Chapter 1: Introduction to Computational Science

Key Notes:

Methods of Doing Science:

Modern science utilizes four basic approaches to the study of how nature works: *observational science, experimental science, theoretical science*, and *computational science*. Of these four, computational science is the newest, made possible by the tremendous improvements in both computer hardware and software over the past 30 years. Computational science, sometimes known as *modeling and simulation* or *scientific computing*, is used extensively in chemistry, and is known as *computational chemistry* or *molecular modeling*. Computational scientists deal with a specific *application*, to which an *algorithm* is defined, which is then computed using a specific *architecture*.

Application:

In computational science, the *application* refers to the type of science being studied, and/or the specific problem being addressed. For example, this *Guide* specifically addresses one application: molecular modeling, also known as computational chemistry or computational quantum chemistry. Specifically, this *Guide* focuses on *electronic structure determination*, which involves trying to solve a specific *algorithm*, known as the *Schrödinger* equation. In electronic structure determinations, chemists calculate the structure, properties, and behavior of atoms and molecules, with an emphasis on the electrons. Other applications include computational astrophysics, computational epidemiology, and numerical weather prediction (NWP).

Algorithm:

In computational science, the scientific problem must be expressed mathematically, known as the *algorithm*. The scientist, oftentimes working collaboratively with a mathematician, must define a *mathematical model* for the problem to be solved. The scientist can use an existing algorithm, modify an existing algorithm, and/or create an algorithm from scratch. Most mathematical models use *approximations* and *assumptions* to the algorithm to help simplify the mathematics. The mathematical model must be tested for how well it represents the science being modeled, and this evaluation happens throughout the modeling process.

Architecture:

Once a suitable algorithm has been determined, that algorithm is translated into one or more *computer programs (software)* and implemented on one or more types of *hardware*. The combination of software and hardware is referred to as the *computational architecture*, or simply *architecture*. The software may be an existing software package, such as Gaussian, GAMESS or MOPAC in the molecular modeling community. Alternatively, one or more computer programmers, working collaboratively with the scientist and the mathematician, might develop the software from scratch. Programming languages such as C++, Java, and high-performance Fortran (HPF) are common programming languages. In terms of hardware, the code might run on a desktop or laptop computer running Windows XP or Macintosh OS X. For larger codes, a *high performance computer* is often used. These machines typically have more than one central processing unit (CPU), can run codes more efficiently, and cost significant amounts of money. There are a number of *supercomputing* centers located across the nation for use in performing large problems with complex codes.

Grand Challenge problems:

Grand Challenge problems are those problems that are complex, and have significant scientific and societal implications. These problems are typically solvable primarily (or only) through a computational approach. Examples of Grand challenge problems include the study of how proteins fold and finding suitable sources of alternative energy. Molecular modeling is one of the

most important technologies in the solution of these Grand Challenge problems, and is not only used in chemistry, but also in biology, environmental sciences, and medicine.

Methods of Doing Science:

The purpose of this *Guide* is to introduce the reader to the *technologies, techniques, and tools* of *molecular modeling,* also known as *computational chemistry*. The use of *computing* and *computation* is one of the ways to do *science*. Science is defined as the study of how nature behaves. Modern scientists agree that there are four methods of doing science:

1. *Observational science*
2. *Experimental science*
3. *Theoretical science*
4. *Computational science*

The purpose of science and scientific inquiry is to answer questions and develop new theories of how nature works, and to develop new technologies that will serve humankind. The fundamental questions in science (especially those with potentially broad social, political, and scientific impact) are sometimes referred to as *Grand Challenge* problems. Some of these problems include:

- Chemistry: how do proteins fold?
- Biology: what is the origin of life?
- Physics: what is the ultimate fate of the universe?
- Neuroscience: what is consciousness?
- Medicine: what is the root cause of cancer?
- Computer science: how can we speed up a computation, and what is the upper limit?

Modern science looks to address these problems, using a wide variety of methods, research protocols, and scientific instruments. In this chapter, the four basic methods of doing science are described, with an emphasis on the newest method -- computational science – that serves as the "parent" for the more specific area of computational chemistry.

1. Observational Science

Francis Collins, one of the scientists who was responsible for the development of the Human Genome Project, states this in his book *The Language of God* (p. 58):

> One of the most cherished hopes of a scientist is to make an observation that shakes up a field of research. Scientists have a streak of closeted anarchism, hoping that someday they will turn up some unexpected fact that will force a disruption of the [scientific] framework of the day. That's what Nobel Prizes are given for.

Observation is the foundation of all science, and the foundation of all methods of doing science. Science begins, and often ends, with observation. Anyone who spends any amount of time with very young children recognizes the inherent ability of these children to make observations. Young children will, with little or no encouragement, spend many hours watching ants build an anthill. Fundamentally, observational science is that area of science that uses a variety of devices, including the naked eye, to collect data about some particular scientific phenomenon. Observations are often, but not always, made with a specific question in mind. Observational science relies on tools such as microscopes, telescopes, satellites, and other devices to improve the quality of the observations. Observations form the basis of the primary scientific method, that being evidence-based reasoning.

[A note to educators: in teaching science, educators sometimes assume that students know how to observe, but this is often not the case. It is probably a bad assumption to trust in the observational abilities of young science students, and as such helping students make observations and make sense of what they see (or even trust what they see) is critically important in science education.]

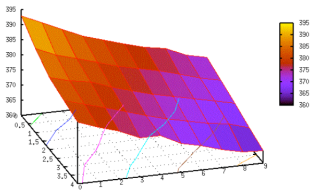
Observational science is, like the other four types of science described here, both *product* and *process*. The observational scientist applies a number of technologies, techniques, and tools to the process of collecting observational data, and, using evidence-based reasoning, looks to suggest possible explanations for the particular set of data collected. The scientists asks questions such as:

1. What *can* we learn, what *do* we learn from these observations?
2. How are we sure these are right?
3. Why should we care?

The observational process, as is the case with most other processes in science, tend to lead to or suggest more observations to be made, perhaps with different instrumentation or from different perspectives.

Observational science is significantly process-focused, but also can be considered as product. The products of observational science are typically datasets of numerical information. Radio telescopes, for example, collect significant amounts of numerical data that are then turned into "observations" by using techniques such as *scientific visualization*. Scientific visualization, a form of computer graphics, turns the numbers into images that can then be used by the observational scientist. For example, the image below shows the results of a small study of a plot of land, where the observational scientist has collected elevation data at 50 different points. The data is then *visualized*, or *rendered*, into a graphic that can be rotated to allow the scientist to see the profile of the land:

393	388	384	382	380.5
390	385	382	380	378
387	383	381	378	376
384	381	378.5	376	373
382	379	376.5	374	373
380	377.5	374.5	372	369.5
378	376	373	370.5	368
377	374	371.5	369	366
374	372	370	368	364.5
372.5	370.5	368.5	366	364



Alternatively, the product of an observational approach might be a field note on the behaviors of gorillas in the field. Regardless of its format, the observational scientist often takes data and looks for *patterns* or *trends*. An observational scientist might analyze data using sophisticated mathematical tools from the fields of pattern recognition and/or statistics. For example, we have now been able to "observe" data of the genetic structure of many living organisms. One particular structure is the genetic code for the protein myoglobin, found in the muscle of almost every animal. Using a variety of mathematical tools, some created specifically for this problem, we can arrange the genetic structures in different fashions. Different arrangements lead to different observations, and to different conclusions. The data and the multiple arrangements are the products of this observational approach.

2. Experimental Science

Experimental science is most typically what students (and the general public) think about when they consider science. The stereotype of the white-coated, slightly disheveled (physically and mentally) loner working in the dark recesses of some laboratory is still surprisingly prevalent. There is also a perception of experimental science as a formula or recipe that one applies to any given question, with an answer being the reward at the end of the application of these steps. Science educators probably perpetuate this through the teaching of the scientific method, which, if one has ever judged a science fair, is clearly evident. Students often think if they follow the four or five steps – make an observation, forma hypothesis, perform some tests, generate a conclusion (with appropriate descriptions of method and materials) – then you are "doing" science. Perhaps, but science, especially experimental science, is significantly more messy than a lock step flowchart-driven system that educators teach in science

classrooms. James Trefil, a well-known scientist who writes extensively about science, writes this in *The Nature of Science: An A-Z Guide to the Laws and Principles Governing Our Universe*:

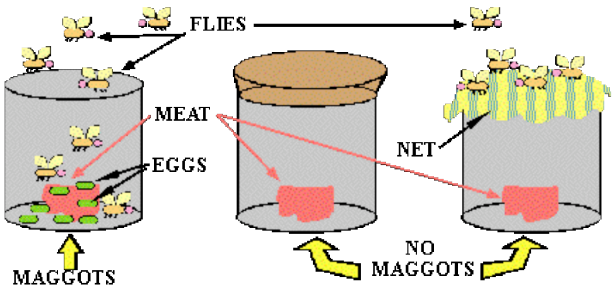
> You will often find, particularly in textbooks, a stepwise procedure that is said to constitute something called "the scientific method." Typically, this is something like, "A scientist first does X, then proceeds to Y, then to Z," and so on. It makes doing science sound like making a batch of cookies from a recipe. The problem with this approach isn't so much that it's completely wrong – scientists often *do* carry out steps X, Y, and Z. Rather, it's that it leaves no room for human creativity, ingenuity, and just plain cussedness that are, and always have been, essential components of the scientific enterprise. Describing the process of science by a "method" is like describing a painting by Rembrandt or Van Gogh solely in terms of where different colors have been applied to the canvas. Science is not the equivalent of painting by numbers.

Experimental science is fundamentally concerned with taking observations and evidence collected using observational techniques, and then conducting specific tests on some aspect of those observations. Experimental science is very much concerned with making measurements in somewhat of a logical and systematic fashion. Experimental science is also concerned with evaluating and investigating cause-and-effect relationships. By identifying and isolating (controlling) specific variables of some phenomenon, the experimentalist can test the effect of one or more variables on the phenomenon. One real problem with experimental science is that it is often difficult to ensure that the variable being tested does not alter the natural behavior. In experimental science, the scientist is primarily concerned with three variables:

1. The *independent variable*, the condition that is being investigated and/or manipulated by the experimentalist
2. The *dependent variable*, the condition which is counted or measured
3. The *confounding variables*, those variables that are not controlled and that may or may not affect the outcome of the experiment.

There are a multitude of good examples of experimental science. One of them is the *Redi Experiment*, conducted around 1670 by the Italian physician Francesco Redi. He wished to prove or disprove the notion of spontaneous generation, the hypothesis that life forms can spontaneously generate from non-living sources, a phenomenon known as *abiogenesis*. Redi's experiment was elegant in its simplicity. His experimental question was the origin of maggots (and whether or not life in the form of maggots could arise from non-life, in this case, meat).

His conjecture, based on observations, was that maggots come from flies. He set up three jars, into which he put some raw meat. Jar 1 was left open, Jar 2 was covered with netting, and Jar 3 was sealed from the outside. In Jar 1, he observed flies laying eggs on the meat, with the subsequent emergence of maggots. On Jar 2, he observed flies laying eggs on the netting, with maggots emerging on the netting. With Jar 3, no flies appeared or laid eggs, and subsequently no maggots emerged.



In this experiment, his *independent variable* was the type of covering on the jars, the *dependent variable* was the production of maggots, and some of the *confounding variables* were temperature, the size of the jars, humidity, and the size and types of the meats.

4. Theoretical Science

Theoretical science is the most esoteric of the four types of science. Middle and high school students, as well as the general public, tend to be able to identify theoretical scientists, but cannot easily define what theoretical science is. The most common reply to the question "what is theoretical science?" is the naming of Albert Einstein, the classic example of someone who *does* theoretical science. Theoretical science has always been a major component of the overall research process, but its complexity has resulted in its lack of prominence as compared with observational and experimental approaches.

Theoretical science is *primarily* mathematical. Theoretical scientists often attempt to mathematically represent some observable or non-observable phenomenon. By applying a variety of mathematical techniques, theoreticians look to "prove" the validity of some hypothesis or conjecture. Oftentimes the mathematical theory is investigated by observational and/or experimentalists to further substantiate or reject the conclusion reached by the theoretician.

The typical end product of the theoretical scientist is one or, more often, a series of relatively complicated mathematical descriptions of some scientific behavior or event. Even coming up with a simple example is challenging. Most laypersons will be familiar with Einstein's $E=mc^2$, a theoretical description of the amount of energy contained in some amount of a substance. The amount of energy (E), according to this theory, is determined by multiplying the mass (m) of a substance times the speed of light (c) squared. It is an excellent theoretical description of the amount of energy inherent in a substance, and has been proven using observational and experimental techniques.

Perhaps a slightly more complicated example comes from theoretical chemistry. It was postulated in the early 1930s that the behavior of an electron as it moves around the nucleus of an atom could be described using mathematics. The mathematical behavior of an electron can be written as a *wavefunction*, so called because electrons behave like the waves you would see in water if you threw a rock into a pond. The wavefunction, called *psi* (represented by a Greek symbol Ψ), is a part of a simple-looking equation created by the great theoretical physicist Erwin Schrödinger:

$$H\Psi = E\Psi$$

In this equation, the H is the *Hamiltonian operator*. There are many mathematical operators, such as the plus sign, the minus sign, and the square root sign. Imagine that the Hamiltonian operator is just a fancy type of square root, and it is doing something to the symbol to its right, *psi*, the wavefunction of the electron.

What happens on the other side? Performing the Hamiltonian operation on the wavefunction produces the *energy* (E) of the electron. If a chemist knows the energy of the electron, s/he can say numerous things about how the atom behaves. Notice, however, that another part of the answer on the right-hand side is a repeat of the wavefunction psi. This equation is a special type of equation known as an *eigenfunction*. In an eigenfunction, performing a mathematical operation on the left results in a repeat of that function multiplied times some constant. In this case, psi is operated on, and it is repeated in the answer multiplied by a constant, which in this case is the energy of the atom.

In case this is not complicated enough, what *is* the Hamiltonian operator? It looks like this:

(1) (2) (3) (4)

$$H = -\frac{1}{2}\sum^{electron}\left(\frac{\partial^2}{\partial x_i^2}+\frac{\partial^2}{\partial y_i^2}+\frac{\partial^2}{\partial z_i^2}\right) - \sum_{i}^{elec}\sum_{I}^{nuclei}\left(\frac{Z_I}{|R_I - r_i|}\right) + \sum_{i}^{elec}\sum_{j<i}^{elec}\left(\frac{1}{|r_i - r_j|}\right) + \sum_{J}^{nuc}\sum_{J<I}^{nuc}\left(\frac{Z_I Z_J}{|R_I - R_J|}\right)$$

This operator is not quite as intimidating as it might appear, *if* it is remembered that particles with like charges repel and particles with opposite charges attract. The numbers above the operator label the four parts of this mathematical equation:

1. Part 1 calculates the *kinetic energy* (K.E.) of the electron; based on equation _ mv^2, with the velocity based on the *partial differentials* of the electron in x-y-z coordinate space. A partial differential is a mathematical property from calculus that measures *change* behavior; in this case the change in the kinetic energy as the position of the electron changes in the x, y, and z directions. The symbol that looks like a "capital E" represents the Greek letter "sigma". This symbol represents *summation*, or adding up. In this case, the equation says to add up all of the changes in kinetic energy in three-dimensional space.
2. Part 2 determines the affinity of a *negatively charged electron* for a *positively charged nucleus*. In this part, *Z* is the *atomic number* of the atom, *R* is the distance that the nucleus is from the electron, and *r* is the distance that the electron is from the nucleus. The *I* and *i* subscripts refer to the number of nuclei and electrons, respectively, contained in the system.
3. Part 3 calculates the *electronic repulsion* of one electron to another. The subscripts are used to differentiate one electron from another.
4. Part 4 calculates the *nuclear repulsion* of two positively charged nuclei. The subscripts again help to differentiate one nucleus from another.

The reader is strongly encouraged to work through this description as a way to grasp not only this particular scientific theory (and one which will be encountered later), but also the mathematical complexities typically found in theoretical science.

The norm is that the experimentalists and observationalists will look to validate this mathematics with experiments and observations, which lead to acceptance, modification, or rejection of the mathematics. This particular equation has been shown repeatedly to predict the properties of atoms and atoms in molecules, with one problem. It is too complicated mathematically to solve for any atomic system with more than a few electrons. This leads to the fourth type of science: *computational science*.

4. Computational Science

It has been suggested (source unknown) that if aircraft technology had advanced at the same rate as that of computer technology, it would be possible to get on a Boeing 747 that would be large enough to carry 12,000 people to the moon in about three hours for the round-trip cost of about $12.00. Computer technology, particular in the areas of increased speed of calculations and more efficient memory storage devices, has improved at a whirlwind pace over the past 30 years. Many of the improvements in computer hardware and in the algorithms (software) that control computers have presented a new tool for investigating scientific problems – *computational science*. Computational science is the newest of the four approaches to scientific inquiry, and is revolutionizing how scientists work and how they think about doing science.

Computational science is the application of *computer science* and *mathematical* techniques to the solution of large and complex problems. Computational science takes advantage of not only the improvements in computer hardware, but probably more importantly, the improvements in computer algorithms and mathematical techniques. Computational science allows scientists to do things that were previously too difficult to do due to the complexity of the mathematics, the large number of calculations involved, or a combination of the two. Computational science allows scientists to build *models* that allow predictions of what might happen in the lab. As such, computational science is *complementary to other methods of science*.

Researchers can use computational techniques to accomplish a number of goals:

1. Perform experiments that might be too dangerous to do in the lab. Scientists can, for example, use computational techniques to predict how a new drug might behave in the body. This allows them to reduce, but not eliminate, the number of animal tests that might have been done prior to the development of these computational pharmacology techniques.
2. Perform experiments that happen too quickly or too slowly. For example, models of global climate change allow environmental scientists to run predictive models many years into the future, looking to determine how past, current, and future human endeavors might impact on the temperature of the Earth.
3. Perform experiments that might be too expensive to do in the lab. Especially in chemistry, there are a number of experiments that require expensive instrumentation. Some of these can now be simulated using computational versions of that instrumentation. While this does not replace the importance of having the actual instrument, it does provide the scientist – and the science student – with a way to interact with the instrument. In areas other than chemistry, flight simulators are a good example of the use of simulation software as a cost-saving method. Flight simulators are significantly less expensive than the actual airplane, and are also safer for the pilot!
4. Perform experiments that are only solvable using computational approaches. Many topics in astrophysics, such as galaxy formation, cannot be observed easily, and certainly are not subject to experimental techniques. Computational models, based on well-understood mathematics, allow the astrophysicist to test a wide variety of parameters and scenarios.

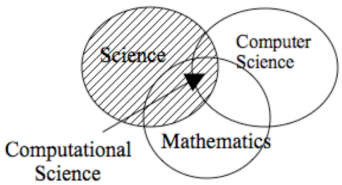
While computational models cannot replace the lab, they have certainly become an integral part of the overall search for scientific knowledge.

There are many definitions of computational science -- most of them describe it as an *interdisciplinary* approach to the solution of complex problems that uses concepts and skills from the disciplines of *science*, *computer science*, and *mathematics*. Of considerable importance is that computational science is not computer science -- computational science is a methodology that allows the study of various phenomenon. Like the other three, it is both a *method of doing science* and a *discipline in and of itself*. For example, one can look at computational science research using the associative law of mathematics:

Computational (science research)
(Computational science) research

In the first example, scientists can use computation as a method of *doing* science research. Computation is used to perform scientific research. Scientists also, however, do research on *how to use* computational science as a tool for inquiry and exploration. As is true of the other three scientific approaches, some scientists *use* the technologies, techniques and tools of observation-experiment-theory, while some researchers *develop* new technologies, techniques, and tools for these areas. In other words, all four types of science are both *process* and *product* in the grand scheme of scientific inquiry.

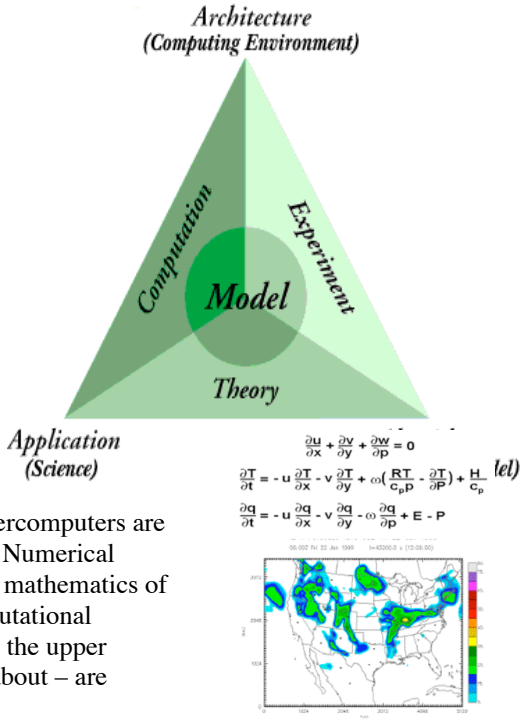
There are other ways to look at computational science. Some will describe it as the intersection of three disciplines:



Computational science is not computer science, which concerns itself with writing software programs and/or the development of new hardware products. Computational science is often defined as being that science that is at the intersection of science, computer science, and mathematics. Alternatively known as *modeling and simulation*, the

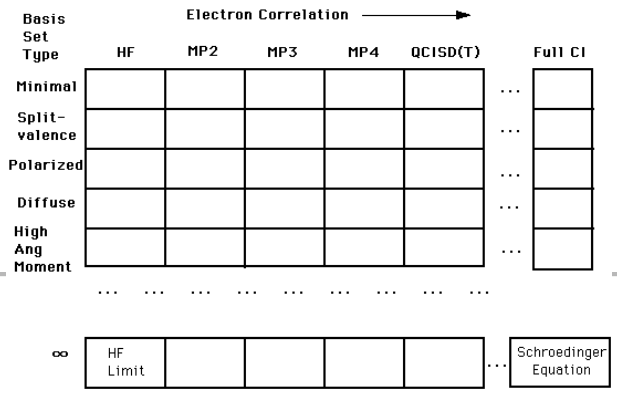
area of computational science looks to create and use computer models as a method of making observations, conducting experiments, and creating or testing new theories. If the calculations or research are being performed on large, high-powered computers, scientists refer to the subject as *high-performance computing (HPC)*, or *supercomputing*. Many of the current and future scientific challenges are going to be significantly dependent on the application of this new methodology.

Another method of describing computational science is: *application*, *algorithm*, and *architecture*. The *application* describes the particular scientific problem to be solved. For example, in the section on theoretical science above, the application would be electronic structure of atoms and molecules. In biology, we might have an application of categorizing all of the genes in the human genome, or predicting the weather in a meteorology application. The next step is to be able to describe that application in mathematical terms, otherwise known as the *algorithm*. A good example of an algorithm is, again, the Schrödinger equation with its complicated Hamiltonian operator shown earlier. The difference now with that equation as compared with its origin in the mid 1900s is that now we have *architecture*, in this case computer hardware and software that allows us to write a program to compute that very complicated problem. The graphic at right attempts to capture this description of computational science. The end product is the computer model, which is supported by, and which supports, experiment, theory, and computation. With the computational model, we can again perform numerical "what if" experiments, test mathematical approximations and theories, and study phenomenon that are difficult if not impossible to study observationally or experimentally.



One challenge to the computational scientist is that it is often the case that the entire algorithm for a particular event is not known, or that the algorithm is so complex that even modern supercomputers are not up to the task. An example of the first is weather prediction. Numerical weather prediction (NWP) modelers have a very good idea of the mathematics of atmospheric chemistry and physics, but not a perfect idea. Computational meteorologists do a good job at modeling large weather effects in the upper atmosphere, but local weather conditions – the ones people care about – are unbelievably complex mathematically.

The electronic Schrödinger problem is an example of the second. Even today, with computers capable of solving a trillion calculations a second, we still have to make approximations to solve Schrödinger's equation. One approximation deals with what mathematics is used to get the equation started, a "priming the mathematical pump" type of approximation. The second approximation deals with how carefully one computes the interaction of one electron as it encounters other electrons. The chart shows the first type down the left-hand side (the basis set type) and the second approximation (the electron correlation) from left to right. The state-of-the-art is currently somewhere towards the bottom right of this graph, but nowhere near being able to completely solve Schrödinger's equation exactly. Approximations to reality rule the day in computational science.



Each of the three aspects – application, algorithm, and architecture – is described in more detail in the following sections.

Application:

In this view, computational science is a scientific endeavor (application) that is supported by the concepts and skills of mathematics (algorithms) and computer science (architecture). Central to any computational science problem is the science itself – what scientific event or problem is of interest? What are its boundaries, what components or factors are part of the system, what assumptions can be made about its behavior, what do we know about other systems that has some similarity to the one we are interested in studying? The application therefore describes the problem to be solved. Some examples of applications are:

- Chemistry: electronic structure determinations
- Biology: genetics and genomics
- Physics: astrophysics and cosmology
- Mathematics: computational geometry
- Medicine: computational epidemiology, computer-assisted drug design
- Environmental science: air quality modeling and numerical weather prediction

Algorithm:

The application, or problem to be solved, allows the researcher to move to the next part of the task: the search for a suitable *algorithm*. In computational science, the algorithm is a *mathematical model* that can be created to represent the behavior described by the parameters of the problem. Often the modeler needs to use one or several *numerical methods*, or numerical "recipes", to begin the solution of the mathematical model generated. Many numerical methods are too complex to calculate by hand and/or require repetitive calculations – *iterations* – to determine an answer. This use of repeated calculations to "hone in" on a solution is known in mathematics as *convergence*, and it is a concept that is used heavily in molecular modeling.

At this point we can use the technologies of computer science to implement our algorithm or mathematical model on some suitable sized computer using some computational software tool. The combination of hardware and software is collectively known as the *architecture*. Computational scientists need to know how to choose the appropriate computing tools to implement the algorithm that is the model.

The process of application-algorithm-architecture can be illustrated with a simple, if nonsensical, problem. The *application* to be solved is a relatively simple one: how does one tie a shoe? This problem is chosen since it requires no scientific understanding – most readers should know how to tie a shoe! Given that the reader understands the problem to be solved, the process of algorithm development can begin.

The first step in the algorithm development is an *assumption* or *approximation*. The shoe has *one* shoestring per shoe, but in tying one's shoe there are functionally two different strings. The two parts can be labeled X and Y or A and B. The assumption being made here is that each part, A and B, is a separate entity, when in fact it is all one shoelace. This simplifying assumption makes it possible to do the algorithm, and fortunately does not alter the answer substantially. Now, we begin to tie the shoe by putting A over B, represented as a "division":

$$TYS = \frac{A}{B}$$

At this stage, we have a new A and a new B, both slightly shorter than they were originally, since some of each is "used up" in the knot. The question is: can we simplify the problem by still calling the laces A and B, as there is not very much of the original A and B used up in the knot? If we can't accept this assumption, the algorithm becomes substantially more complicated and perhaps less useful (but more accurate). Even if we don't accept this assumption, there is going to be some error introduced in the form of round-off error. Because of the way computers store numbers, any time a calculation is performed, some amount of accuracy is lost. Round-off error is the error that occurs "naturally" as a function of the amount of precision you have in your measuring tool and/or in your computational environment. In this case, we can say that the error introduced by ignoring the very little bit of A and B that is in the knot itself is trivial, so we'll continue to call each lace by its original name, A and B.

What is the next part of the algorithm? At this stage, we fold A and B each in half (mathematically multiply them by 0.5), then put the half-folded A over top (division) of the half-folded B. Once this operation is done, the algorithm is complete, and the entire algorithm TYS is as follows:

$$TYS = \frac{A}{B} + \frac{0.5A}{0.5B}$$

One of the real challenges of computational science is developing the ability to "see the world" as a mathematical algorithm. Good computational scientists think about their discipline (chemistry, physics, biology, etc.) in mathematical terms, a very different mindset than that of an observational or experimental scientist. Organizations such as the National Science Foundation (NSF) at the federal level, and funding agencies such as the Burroughs Wellcome Fund (BWF) support the development of resources such as the North Carolina High School Computational Chemistry server and this *Guide* in order to ensure that tomorrow's scientists, mathematicians, and engineers have the opportunity to develop these mathematical and computational mindsets and skills.

Architecture:
Once we have created an algorithm, we can use a wide variety of computational tools (including a pencil!) to generate a solution to the algorithm for a variety of conditions. This mathematical model of tying your shoe allows us to do simulations -- perform "what-if" scenarios by altering the input parameters or initial conditions of the physical event.

Grand Challenge Problems:

There is clearly a *symbiotic relationship* between the four areas. Observations often inspire new experiments, experimental data is used to generate new theoretical constructs, and this mathematics is used to build and validate computational models. The fundamental questions in science (especially those with potentially broad social, political, and scientific impact) are sometimes referred to as *Grand Challenge* problems. Many of the Grand Challenge problems are those that can only be solved computationally. Indeed, there are some definitions of Grand Challenge problems that limit those problems that can *only* be solved by computers. Certainly chemistry problems are considered by all computational scientists to be one of the major Grand Challenge categories. In chemistry, the argument has been made that we have known (since 1928) all of the theoretical mathematics needed to solve every chemical problem. It is only since the birth of computational science (late 1950's) that we have had the tools and technologies needed to solve these complex mathematical equations born from the theorists.

Current Grand Challenge problems in chemistry include these examples:

- Protein structure, particularly protein folding

- Computer-assisted drug design
- Models of environmental chemistry, especially related to global warming and pollution
- Development of carbon-free energy sources

It is clear that the solution to these challenges, all of which are of critical importance to both science and society as a whole, is dependent on the degree to which current and future scientists understand, and can leverage, the technologies, techniques, and tools of computational science and computational chemistry.

Chapter 2: Introduction to Molecular Modeling

Key Notes:

Molecular Modeling Defined:

Molecular modeling is the general term used to describe the use of computers to construct molecules and perform a variety of calculations on these molecules in order to predict their chemical characteristics and behavior. The term molecular modeling is often used synonymously with the term *computational chemistry*. Computational chemistry is a broader term, referring to any use of computers to study chemical systems. Some chemists use the term *computational quantum chemistry* to refer to the use of computers to perform electronic structure calculations, where the electrons in a chemical system are calculated. [NOTE: in this *Guide* the terms molecular modeling, computational chemistry, and electronic structure computing are used synonymously; the reader is advised to recognize that this is not a widely accepted practice!)

Molecular Modelers:

Molecular modelers, or computational chemists, are those scientists who are specially trained in the technologies, techniques, and tools of molecular modeling. This field is quickly becoming a new sub-specialty of chemistry, gaining importance alongside sub-disciplines such as organic, inorganic, analytical, physical, and industrial chemistry. Computational chemists have their own journals (such as the "Journal of Computational Chemistry") and computational chemistry is becoming increasingly recognized as a discipline in its own right. Universities are beginning to offer concentrations and/or majors in computational chemistry.

Fundamental Uses:

Molecular modeling is used to calculate a wide variety of properties of individual atoms and molecules. Chemists typically want to know three things about an individual molecule: its chemical structure (number and type of atoms, bonds, bond lengths, angles, and dihedral angles); its properties (basic characteristics of the molecule, such as its molecular energy, enthalpy, and vibrational frequencies); and its activity (those characteristics that describe how the molecule behaves in the presence of other molecules, such as its nucleophilicity, electrophilicity, and electrostatic potentials).

Methods:

The computational chemist must learn and apply a variety of methods to specific modeling situations. For the purposes of this *Guide*, the focus will be on four general methods: molecular mechanics (MM); *ab initio* quantum chemical methods; semi-empirical quantum chemical methods; and the newest method, Density Functional Theory (DFT). Depending on how broadly one defines molecular modeling or computational chemistry, there are a number of other methods that can be considered.

Tools:

Tools can be described as both software and computer hardware. Computational chemists have an increasing variety of software computer programs and tools with which to perform different types of chemical calculations. Computational chemists are concerned with the choice of a computing platform, ranging from simple desktop computers to the use of very high-performance supercomputers. In performing molecular modeling calculations, chemists are often constrained by what software and computing power they have available. With the advent of desktop computational chemistry software and increasingly powerful desktop machines, computational chemistry is moving from supercomputing centers to university and high school computer labs.

Molecular Modeling Defined:

Molecular modeling is a broadly generic term that defines the use of computers to study chemical systems, with an emphasis on the structure, properties, and activities of molecules. As with many fields of study, there are a number of terms that are used to describe specific aspects or areas of the discipline. The term *computational chemistry* is often used synonymously with molecular modeling. Computational chemistry is considered by some to be a much broader term, referring to *any* use of computers to study chemical behavior or systems. One specific area is *computational quantum chemistry*. This is the largest sub-discipline of computational chemistry, focusing specifically on the determination of the behavior of electrons in chemical systems, including those at the atomic and molecular levels. Another term for this area of computational chemistry is *electronic structure determinations*.

With the exception of a discussion of molecular mechanics, this *Guide* is primarily concerned with electronic structure determinations. There are several reasons for this. First, this Guide is designed to support chemistry students and educators who are using the North Carolina High School Computational Chemistry server as their primary computing platform. Second, electronic structure determinations provide a tremendous amount of breadth and depth in the study of chemistry. As one develops excitement, experience, and expertise with electronic structure calculations, the reader is encouraged to broaden his or her definition of computational chemistry.

Molecular Modelers:

Chemists are increasingly identifying themselves as molecular modelers and/or computational chemists. There are roughly two types of computational chemists: those who *apply* the techniques and tools of computational chemistry to solve interesting problems, and those who work to *improve* the techniques and tools. Many chemists (and other scientists, especially those in medicine) will develop a basic proficiency in molecular modeling, but will seek the advice and counsel of chemists who devote a significant part of their scientific career to the field. In the training of new chemists, new chemists learn a little bit about *most if not all* of the tools and sub-disciplines of the field, and then specialize in specific tools and/or techniques at the graduate and/or post-doctoral levels. Molecular modeling is rapidly becoming one of the important tools to be learned, along with basic wet lab skills (titrating, making solutions, etc.) and the use of specialized chemistry tools such as infrared spectrophotometers and spectroscopy instruments. Increasingly, universities and research institutions are constructing laboratories devoted to molecular modeling applications and research that will advance the field.

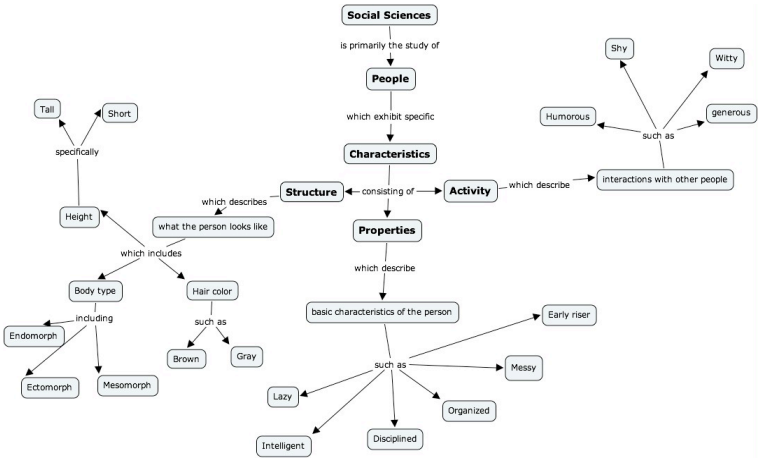
Fundamental Uses:

Molecular modeling allows the user to determine three fundamental items of interest of a molecule or system of molecules:

- the structure, or geometry of the molecule
- the property or properties of a molecule or system of molecules
- the activity, or reactivity, of a molecule or system of molecules.

These determinations can be undertaken to validate experimental studies, or can be carried out to predict experimental results. With increased sophistication of the technologies, molecular modeling can be used to reduce dependence on wet, or traditional, chemical experimental procedures.

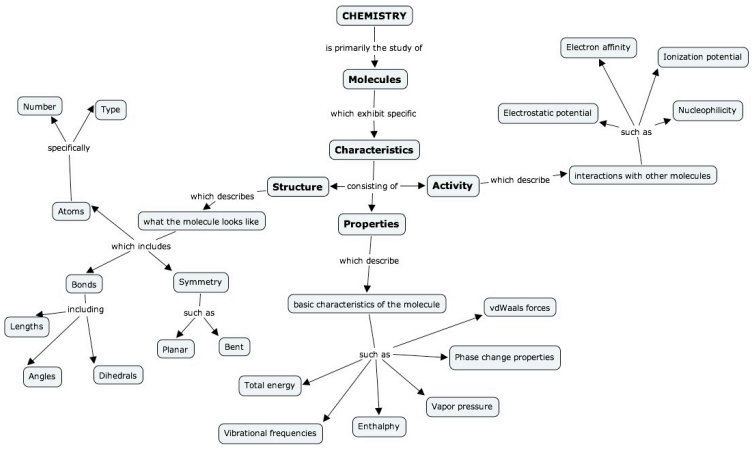
The following concept maps attempt to convey how molecular modeling is used. The first graphic is a concept map that describes the field of social science. The map defines social sciences as the "study of people" that exhibit specific characteristics. The map then divides these characteristics into three categories: those that describe the "structure" of people (mostly focused on physical characteristics); those that describe the "properties" of people (those basic characteristics that the person has all the time, regardless of his or her surrounding environment); and those that describe the "activity" of the person (those characteristics that are only meaningful in the presence of other people). For example, an individual can be described as tall and handsome, messy, and humorous. Tall and handsome help us to visualize his physical appearance. The term "messy" is a basic character trait, and one assumes that the person is messy regardless of his or her situation. The term "humorous", however, only has meaning in the context of being in the presence of other people. One doesn't label himself or herself as humorous. That description only comes from his or her interaction with others.



Likewise, one can construct a similar map for the study of chemistry, which is primarily the study of molecules. Analogously, one can say that one *knows* a molecule if three primary characteristics are known:

1. its structure (physical form)
2. its properties (characteristics that are fundamental)
3. its activity (characteristics that depend on interaction with other molecules)

While both of these characterizations of social sciences and chemistry are highly simplified, thinking about molecules in terms of structure-properties-activity is perhaps a useful organizer.



In using molecular modeling techniques and tools, modelers can calculate structure, properties, and/or activities. The list below provides a *partial* list of the types of calculations that can be performed using the North Carolina High School Computational Chemistry server:

- Single point energies (molecular energies)
- Molecular orbital calculations, including determination of frontier orbitals
- Vibrational frequency calculations
- Reaction mechanisms and reaction path following studies
- Determination of IR and UV-Vis spectra
- Transition structures and activation energy diagrams
- Electron and charge distributions
- Potential energy surfaces (PES)
- Thermodynamic calculations

Methods:

This *Guide* presents four different molecular modeling methods: one that uses classical physics (i.e. mechanics) to study the behavior and interactions of molecules, and three electronic/quantum mechanical areas;

1. *Molecular mechanics*: this method uses traditional classical mechanics. In essence, atoms are considered to be spheres that are connected to other atoms by a spring. Various properties of the molecule can be calculated by measuring the motion of the atoms and the changing energies of the springs.
2. *Ab initio quantum mechanics*: In *ab initio* methods, Schrödinger mathematics are used to calculate a wide variety of quantum chemical properties. The main calculation is the wavefunction determination. Based on the results of calculating the wavefunction, other chemistry properties and activities can be determined. In this method, 100% of the final determination is done mathematically.
3. *Semi-empirical quantum methods*: in semi-empirical methods, a portion of the calculation comes from experimental data, and the rest comes from mathematics. The major advantage of the semi-empirical method is that it is faster and able to perform calculations on larger molecules.
4. *Density Functional Theory (DFT):* DFT is the newest computational method, and is increasing in popularity among computational chemists. Rather than calculate molecular properties based on the determination of the wavefunction, the DFT method determines properties from calculating the electron density. It uses what is known as a functional (a function of a function) to determine the electron density, and by extension, the quantum properties of a molecule.

All of these methods have advantages and disadvantages, most of which are discussed in greater detail in this *Guide*. Likewise, all of these methods have proponents and critics (and the authors of this *Guide* are no different!)

Tools:

There are a wide variety of computational tools available, both commercial (fee-based) and open-source/public domain. In addition to the basic software programs (or *codes*), there are also a wide variety of support tools, such as interfaces to the codes, data visualizers (programs that create images from the computed data), and other support programs. There are three industry-standard quantum chemical codes running on the North Carolina High School Computational Chemistry server:

- GAMESS (General Atomic Molecular Electronic Structure System, in the public domain)
- Gaussian (a high-end commercial package)
- MOPAC (Molecular Orbital PACkage, also in the public domain)

WebMO, a Web-based interface to these three packages, with a built-in Java-based data visualization system, manages all three of these codes.

All computer programs must run, of course, on a computer. The three chemistry software packages and the WebMO interface are installed on a dedicated Dell 2U PowerEdge 2850 EM64T (Extended Memory 64-bit Technology), 2.8 GigaHertz (GHz) server, running two *central processing units* (CPUs). This computer is dedicated to the North Carolina High School Computational Chemistry project. The fact that it has two processors mean that only two jobs can be running at the same time. Other jobs waiting to be run are placed into a *queue*, and are run in turn in the order in which they were submitted. The computer is located in the offices of the Shodor Education Foundation, Inc., in downtown Durham (NC). The name of the server is *chemistry.shodor.org*, and it is this machine that users access when they log in through the North Carolina High School Computational Chemistry web page (http://www.shodor.org/chemistry).



Dell PowerEdge 2850

Chapter 3: A Computational Analogy

Key Notes:

Analogy Overview:

An analogy is fundamentally a comparison between something well understood and something less well understood. In science education, the use of analogies is often useful to help learners develop a framework for understanding a new topic. In describing molecular modeling, an analogy is useful. In this *Guide*, the process of performing a molecular modeling calculation is compared with that of cooking chicken. In cooking chicken, one needs to know what kind of chicken one has, what the end product of the cooking process will be, what cooking tools are available, what recipes are known or available, and what ingredients one has at his or her disposal. In "cooking" molecules, one needs to know the geometry of the molecule, the desired goal of the calculation, the available tools, the methods known or available, and what resources one has at his or her disposal. This chapter describes this analogy in more detail.

Chicken Type/Geometry:

The starting form of the raw chicken usually has a significant bearing on how the chicken might be cooked, and what possible chicken dishes might result. Whole chickens will typically require different recipes or cooking methods than packaged chicken strips. If the chicken is frozen, it is typically assumed that the chicken be thawed before any cooking process begins. Analogously, the starting form of the molecule must be known prior to any calculation. The number of atoms, types of bonds, bond lengths, and the bond angles and dihedrals all describe the molecular geometry, the starting point of any molecular calculation. As with frozen chicken that must be thawed prior to cooking, molecules are typically optimized prior to the performance of any calculation.

End-Product/Desired Goal:

Just as the cook has an idea of what type of end product is desired, so too does the molecular modeler. While a chicken chef might have choices such as baked chicken, boiled chicken, barbeque chicken, and chicken casserole, a modeler needs to know if the desired goal is a molecular energy calculation, a vibrational frequencies determination, a thermochemical calculation, or a natural bond order end result. Just as there are a diverse number of end products for the chef, so too are there a diverse number of calculation types for the chemist.

Cooking Tools/Computational Tools:

A good chef knows his or her cooking tools. Of primary importance to the chef are the main sources of energy: oven, stove, microwave oven, grill, open fire, and the like. The availability of a cooking tool will often dictate the available dishes that can be prepared. Access to all cooking tools provides the chef with unlimited possibilities, while the reverse is also true. Analogously, a molecular modeler is often only as good as the software and hardware tools that are at his or her disposal. Just as professional chefs typically have well-equipped kitchens, professional molecular modelers will have excellent access to high-performance computing tools. Limitations on access to these tools, whether it be cooking or modeling, limit what can be accomplished.

Recipes/Computational Methods:

Recipes provide the main instructions for how to prepare a specific chicken dish. Good cooks are characterized by knowing numerous recipes, typically without the need to refer to a cookbook. They are also able to improvise new recipes by combining different parts of different recipes. Likewise, the molecular modeler must know various computational methods, such as *ab initio*, semi-empirical, density functional theory (DFT), and molecular mechanics methods. These methods form the foundation of most molecular modeling programs and calculations, and the good modeler must know them to be able to perform and interpret a molecular calculation. The good modeler knows how to use these methods in combination to achieve the best result.

Ingredients/Resources:

The last part of the analogy concerns ingredients. In order to prepare a diverse number of chicken dishes, the chef needs a well-stocked kitchen, with a wide variety of spices, seasonings, sauces, and other ingredients besides the chicken. Availability is, however, not sufficient. The good chef also knows what combinations of ingredients work best for a desired result, knows how to modify ingredients as needed, and otherwise is able to choose ingredients intelligently. In the molecular modeling scenario, the good modeler understands his or her ingredients. Primarily, these are the basis sets, the foundational mathematics that described the electrons in atoms and molecules. The choice of a basis set has significant impact on the type and quality of the molecular calculation, and a good chemist is able to choose his or her "ingredients" – the basis sets – intelligently. Just as the choice of a specific cooking ingredient is part science and part art, so too is the choice of a basis set.

Analogy Overview:

The chart below shows five central questions that are common to both the chef and the modeler:

CONSIDERATIONS	CHICKEN	MOLECULES
STARTING SHAPE: What does the thing look like?	**What is the shape of the chicken? Legs, breast, wing, strips?**	**What is the geometry of the molecule? Atoms, bonds, angles and lengths?**
FINAL PRODUCT: What do I want at the end?	**How do I want the chicken cooked? Fried, baked, BBQ, stir-fry, etc.**	**What properties do I want to know? Energy, vibrations, thermodynamics, transition state**
AVAILABLE TOOLS: What tools do I have to work with?	**What cooking devices? Microwave, stove, oven, grill, open fire?**	**What computational software do I have? Gaussian, GAMESS, MOPAC, others**
INSTRUCTIONS: What method(s) do I have or know?	**Which recipe(s) do I know or have?**	**Which methods do I know or have? Ab initio, semi-empirical, MM, DFT, etc.**
RESOURCES: What resources do I have at my disposal?	**What ingredients do I have? Sauces, spices, etc.**	**What type of basis sets do I know and have? Minimal, split-valence, polarized, diffuse, etc.**

Each of these questions, and their answers, are critical components of the two processes. A measure of the competence of each type of person – chef and chemist – lies strongly in the breath and depth of their understanding of these critical questions, their access to and abilities to use specific tools and appropriate supporting resources. This chart (and indeed this analogy) oversimplifies the complexities of being a good chef or a good computational chemist, but it also provides a foundational framework for beginning studies in molecular modeling (or cooking school!). An understanding of this framework will hopefully provide the beginning molecular modeler with a way to organize his or her increasing knowledge of this important field of study.

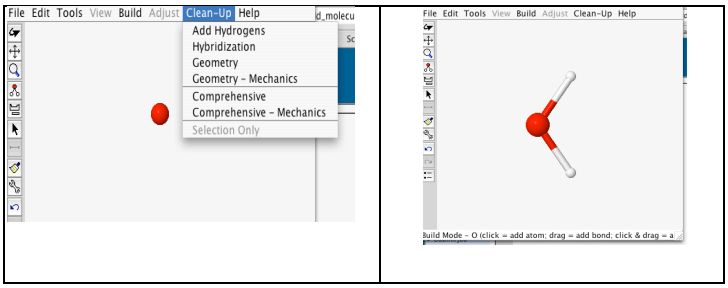
Chicken Type/Geometry:

The first question asks: what does the thing look like? We all know what chicken looks like, whether it comes from a farm or the supermarket. Regardless of where it comes from, most people know that there are different cooking options depending on the "geometry" of the starting raw chicken. In molecular modeling, the starting point is the molecule itself. We need to determine the basic molecular geometry. This typically includes:

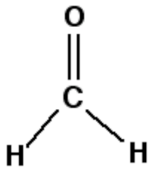
1. Number and types of atoms
2. Number and types of bonds (single, double, triple, etc.)

3. Relevant bond lengths (usually in units of Angstroms (10^{-10} meters)
4. Relevant bond angles (usually in degrees)
5. Relevant dihedral angles (an angle between four atoms, which signifies the 3-dimensional shape of the molecule)

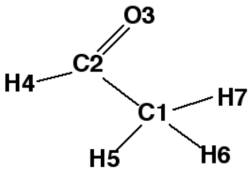
The good news is that most modern software packages come equipped with a *molecular editor*. One simply selects the atoms in the molecule, connects them by drawing lines from one atom to another (or two lines in the case of a double bond), and, often, asks the software to "clean up" the molecule. Cleaning up a molecule typically adds any necessary hydrogens; determines approximate bond lengths, bond angles, and dihedrals; and otherwise provides a reasonable geometry of the molecule. Upon inspection, the molecule looks like it should. For example, in building water, one simply inputs an oxygen atom. Upon performing a "clean up", the two hydrogens are added, the molecule looks like a standard bent molecule, the C-H bond lengths are about 1 angstrom (abbreviated Å), and the H-O-H bond angle is about 109°. Those numbers are not quite exact, but pretty close.



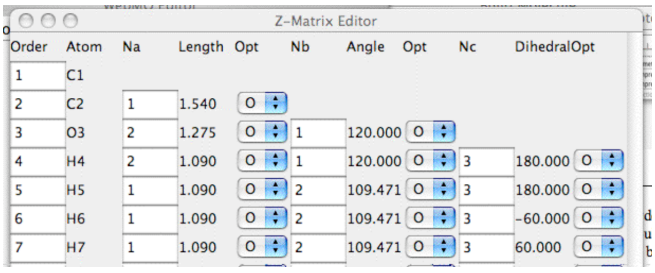
A starting geometry for the formaldehyde molecule (CH_2O) is shown in the diagram. You can see that there is a double bond between the carbon and the oxygen, and that there are single bonds between the carbon and hydrogens. One can easily measure the bond lengths and bond angles for this molecule.



Historically, starting geometries were generated by hand, using what is known as a *z-matrix*. While starting geometries are rarely constructed these days with z-matrices, knowing how to construct simple ones is a useful skill. An example is shown. The graphic shows the molecule acetaldehyde. The atoms are numbered for convenience, starting with the first carbon. One can see the single and double bonds. If this molecule were constructed using a molecular editor (again, standard with almost all software packages), and then cleaned up, the three-dimensional shape of this molecule would be apparent.



The z-matrix for this molecule is shown here. The matrix starts with the first carbon atom (C1). The next line indicates the bonded carbon atom (C2). The 1 following the C2 says that carbon 2 is bonded to atom 1, in this case, C1. The value of 1.540 says that the bond length between the two atoms is 1.540 Å. The next line says oxygen (O3) is attached to carbon 2, with a bond length of 1.275 Å. It then says that the bond angle between O3-C1-C2 is 120.000 degrees. Notice that there is no reference here to the fact that this is a double bond! The bond length strongly suggests a double bond rather than a single bond! The next line describes hydrogen (H4). It is connected to carbon 2 with a bond length of 1.09 Å, forms an angle from H4-C2-C1 of 120 degrees, and forms a *dihedral angle* of 180.0 degrees (that is, laying flat on the plane) between atoms H4-C2-C1-O3. The final three lines describe the remaining hydrogen atoms with bond lengths, bond angles, and bond dihedrals.



Order	Atom	Na	Length	Opt	Nb	Angle	Opt	Nc	DihedralOpt
1	C1								
2	C2	1	1.540	O					
3	O3	2	1.275	O	1	120.000	O		
4	H4	2	1.090	O	1	120.000	O	3	180.000 O
5	H5	1	1.090	O	2	109.471	O	3	180.000 O
6	H6	1	1.090	O	2	109.471	O	3	-60.000 O
7	H7	1	1.090	O	2	109.471	O	3	60.000 O

This type of code represents a starting geometry for a molecule. Again, z-matrices are often foreign to new molecular modelers, given the proliferation of graphically based molecular editors. Pedagogically, however, there is still value in having students construct several z-matrices by hand, and several software packages still provide users with the option to use this method, either as an optional input method and/or as a way to modify a geometry by hand after the work of the molecular editor is completed.

End-Product/Desired Goal:

It is unusual for a chef to not know what the end product of the cooking process will be, whether chicken casserole, or fried chicken, or some other chicken-based meal. Likewise, the molecular modeler typically knows what s/he wants in terms of a desired goal. Some examples of desired goals are as follows:

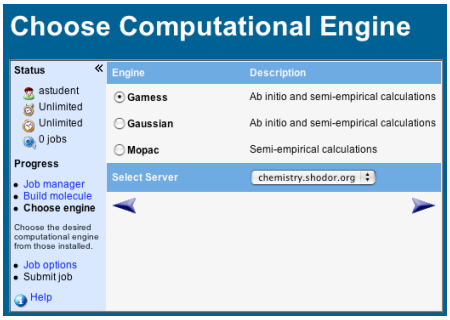
- Electronic structure determinations
- Vibrational frequency calculations
- Transition structures
- Electron and charge distributions
- Potential energy surfaces (PES)
- Rate constants for chemical reactions (kinetics)
- Thermodynamic calculations

In most molecular modeling software packages, the software interface provides a *pull-down menu*, allowing the user to choose the type of calculation desired. Different software packages provide different options. Most if not all provide a "design it yourself" option, allowing the experienced practitioner to go beyond the choices provide as a standard option in the user interface. Obviously, significant experience with the basic options is typically a pre-requisite to being able to custom design a calculation. Only the most experienced chefs can create elegant new dishes!

In subsequent chapters, and in the labs included in this *Guide*, discussions of these various types of calculations will be described and demonstrated in one or more lab activities. Many if not all of these calculations provide significant learning opportunities for new chemistry students. The computing software tools found on the North Carolina High School Computational Chemistry server are the *same tools* as are found in many research labs around the world.

Cooking Tools/Computational Tools:

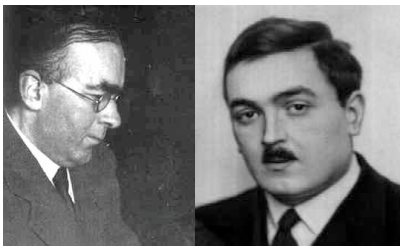
Cooking chicken requires a source of energy. The choices for chefs include an oven; a stove; a microwave oven; an outdoor grill; and even an open fire. While fundamentally the same, each has its own specific uses, nuances, advantages and disadvantages. Likewise, in molecular modeling, there are a number of computer software programs and packages, all running on various types of computer hardware systems. These computational tools can be called *engines*, because these are the tools that make the calculations run. There are many molecular modeling engines. Just as most cooks will have their preferred cooking device, so too do most chemists. The graphic shows an example from the North Carolina Computational Chemistry server of the choices available for computational engines. More detail about these is in subsequent chapters. Suffice it to say that these three represent the most highly used and powerful packages in the computational chemistry field today. They are the *industry standard* tools. The choice of an engine depends on what one wants to do, how the user wants to do it, and how quickly one wants the answer. For example, MOPAC is somewhat analogous to the microwave oven. The calculation is done relatively quickly as compared to other tools, but without some of the accuracy and with less information.



Recipes/Computational Methods:

All good chefs know a variety of recipes for cooking chicken. Expert chefs are those characterized by their understanding of the basic recipes and their abilities to adapt recipes. In molecular modeling, there are also "recipes", known as a *theory*. Unfortunately for the fledgling molecular modeler, the names of these methods are very esoteric sounding, but they are nothing more than recipes to be learned and, hopefully, internalized. The graphic shows a sample window that contains several options. Notice, by the way, that GAMESS is the "source of heat" that this particular chef has chosen with which to cook his chicken! This calculation requests a *vibrational frequency*, shown as the calculation choice. Under that entry, one sees the label "Theory". The option selected there is *RHF*, which stands for *Restricted Hartree-Fock*. Hartree and Fock were two physicists who developed one of the main methods for calculating the properties of molecules. Their method, which will be described in more detail later, is still a popular "recipe" in modern computational chemistry. Hartree is shown on the left, and Fock is on the right. Hartree was an English physicist, while Fock was a physicist from Russia. They developed their computational method independently, and the method was named in both of their honors. Having a unit of energy named after him, as we shall see later, also honored Hartree.





Ingredients/Basis Set:

All good chefs need ingredients besides the raw chicken. If the only ingredient in the pantry is a salt shaker, even the most expert of chefs will be very limited in the chicken dish that he or she will be able to prepare. Most chefs, especially in the finer, better-equipped kitchens, have an abundance of ingredients at their disposal. Not only do they have these ingredients, they have expertise in how to choose the right ingredient(s), how to combine ingredients, and how to adapt and adjust ingredients to produce a specific result. Likewise, the computational chemist has ingredients, the choice of which depends on the knowledge and skill of the user. These ingredients are called *basis sets*, and they represent the beginning mathematical description of where we might find an electron in an atom or a molecule. As with computational methods, these basis sets have very esoteric names, but once you understand the notational system, they are not so intimidating! Also, most if not all basis sets can be categorized as a specific type, providing yet another way to understand and describe these ingredients. Basis sets can be categorized as follows (an incomplete listing):

- Minimal basis sets
- Split-valence basis sets
- Double/triple/quadruple valence basis sets
- Polarized basis sets
- Diffuse basis sets

These types will be described in greater detail in later chapters, and several of the labs provide investigations into the various types of basis sets. In the graphic above entitled **Configure GAMESS Job Options**, one observes that a **Minimal STO-3G** basis set has been chosen for the calculation. This performs the calculation done fairly quickly, and the "chicken" is going to be pretty bland, because an STO-3G basis set is the computational equivalent to only having a salt and pepper shaker in the pantry!

Like most areas, including cooking and molecular modeling, practitioners have their jargon and notational systems. Chefs have jargon such as "sauté", and notations such as "tsp". Molecular modelers like to talk about *model chemistries*, which refers to the method used to perform a calculation and the basis set chosen to support that calculation. For example, in the GAMESS option window above, the modeler has chosen a *Restricted Hartree-Fock method*, using the *STO-3G* basis set. No self-respecting computational chemist would choose that method for significant research purposes, but for educational purposes, or for the researcher needing a quick "first look" calculation, it is an adequate choice. The term "restricted" does not mean limited; it means that the molecule contains all paired electrons, no unpaired electrons in the system. More detailed discussions of the *Hartree-Fock* and other methods are presented in this *Guide*.

Remember also that, just as the chef needs to thaw the chicken before it is cooked, so too must the molecular modeler need to ensure that the starting molecule is the in its most accurate geometric form. This means the bond lengths, angles, and dihedrals (if there are any) are all "correct". This *Guide* will explore what "correct" means in subsequent chapters. This process is known as a *geometry optimization*. It is performed before the main calculation is started. Indeed, there are online libraries where pre-built, optimized molecules can be downloaded, saving one the time and computational energy needed to do this. Most software packages contain a number of pre-built and optimized molecular libraries, which also in many cases include parts of molecules that can be used like Lego blocks.

Regardless, in describing the *model chemistry*, molecular modelers use a notational system. The format is as follows:

Theory used to do the calculation/Calculation basis set//Optimization theory/optimization basis set

As an example, a formaldehyde molecule was built using a molecular editor. It was *optimized* using a *Hartree-Fock method* (1), using the *3-21G basis set* (2). A *vibrational frequencies calculation* was then performed, using the *Becke3-Lee-Yang-Parr (B3LYP) method* (3), with a *6-31G(p) basis set* (4). None of that should make any sense yet! Regardless, in communicating with other molecular modelers, the researcher would use this notation (labeled using the numbering system above):

(3) (4) (1) (2)
B3LYP/6-31G(p)//HF/3-21G

In no time at all, this notational system, and the meaning behind these strange sounding terms will become as comfortable as baste, teaspoon, and broil are to the chef!

Chapter 4: Basis Sets

Key Notes:

Basis Sets Defined:

Basis sets are a series of numbers that are used by the computational chemistry software to begin the process of describing where the electrons are in proximity to the nucleus and to each other. Specifically, basis sets describe where the electrons are in the atomic orbital. There exist a wide variety of basis sets, and the choice of the proper basis set for a calculation is an important decision that the computational chemist needs to make. Basis sets are typically built into modern computational chemistry software, or a chemist can use resources such as the "Gaussian Basis Set Order Form" to find a desired basis set for inclusion in a calculation.

Minimal Basis Sets:

As the name suggests, minimal basis sets are designed to expedite a calculation, and are rarely used for research purposes. These are primarily used to obtain a "first look" at one or more molecular properties. Even though minimal basis sets produce a quantitative result, the experienced computational chemist uses these primarily for qualitative purposes. Example minimal basis sets are STO-3G and STO-6G. Minimal basis sets see all electrons as being of equal importance.

Split Valence Basis Sets:

Split-valence basis sets are the next step up in terms of usefulness and importance. Split-valence basis sets take into account that valence electrons (those on the outermost orbitals) are the electrons that are involved in bonding and chemical reactions, as opposed to the core electrons, which are typically not involved in reactions. Split-valence basis sets perform a quick and dirty approximation of the behavior of the core electrons, and then do a more thorough and careful calculation of the valence electrons. Example split-valence basis sets are 3-21G and 6-31G.

Polarized Basis Sets:

We typically describe the location of electrons in terms of their electron configuration. For example, for carbon we describe the configuration as $1s^2 2s^2 2p^2$. This accounts for all six of the carbon electrons. However, some of those electrons might occasionally "stray" into a "d" orbital, and polarization takes that into account. As such, polarization gives a more accurate description of where the electron is and where the electron can go. A polarized basis set is indicated by a "*" in some places, or by the orbital name in others. For example, we can have a 3-21G* basis set or a 3-21G(d) basis set. Both refer to the same basis set. Chemists would describe this particular example as a polarized split-valence basis set.

Diffuse Basis Sets:

Electrons are typically found close to the nucleus of the atom. We are interested in determining the probability of where electrons are, and that probability is highest near the nucleus. It then holds that as the distance from the nucleus gets larger (bigger atomic radius), there is less probability of finding the electron. At larger distances from the nucleus, we can stop our calculations, because we typically won't find the electrons there. For some systems, however, especially anions and atoms in the excited states, we use diffuse basis sets to extend the distance away from the nucleus that we are looking for electrons. These basis sets are indicated by a "+" symbol, such as 3-21+G. A basis set like 6-312+G(d) would be a polarized diffuse split-valence basis set.

Basis Sets Defined:

Basis sets represent a series of beginning numbers that help the computational chemistry software begin the process of calculating the wavefunction. The wavefunction is a mathematical description of where an electron or group of electrons are in relation to the nucleus and to each other. The wavefunction represents a mathematical approximation at best, but determining the wavefunction correlates well to the calculation of many atomic and molecular properties. The square of the wavefunction, calculated over all of the places where the electron is likely to be, gives us the probability of finding the electron or electrons. The knowledge of that probability is directly translatable into properties such as the energy of the molecule and other characteristics.

A simple example of a basis set is useful at this point. Imagine a really simple system, such as hydrogen, with its lone electron. Let's make that system even simpler, by saying that the electron can only move in one direction – x – from the nucleus. Where do we expect to find that electron? We expect that it will be close to the nucleus, given the charge differences between the nucleus (positive) and the electron (negative). We would imagine that the chance of finding the electron decreases as we move away from the nucleus.

How does the basis set help us to calculate that? Given that we have an extremely simple system, our basis set consists of a single number. Let's say that number is 1.24. We get this number from published literature:

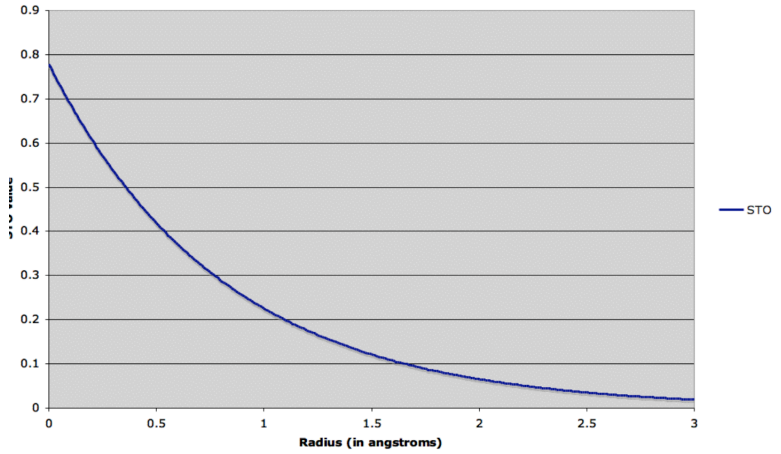
Atom	Zeta value for 1s orbital	Zeta value for 2s and 2p orbitals
H	1.24	
He	1.69	
Li	2.69	0.80
Be	3.68	1.15
B	4.68	1.50
C	5.67	1.72
N	6.67	1.95
O	7.66	2.25
F	8.56	2.55
Ne	9.64	2.88

This is the number that we use to begin our determination of the wavefunction, a mathematical represent of the location of the electron in relation to the nucleus. What do we do with this basis set number? We use it in a formula that represents the wavefunction. There are two "flavors" of this formula, one created by the American physicist J.C. Slater, and known as the "Slater Type Orbital", or "STO"; and one that comes from a type of mathematics known as a Gaussian distribution, or a Gaussian Type Orbital (GTO). You probably know Gaussian curves as a "bell shaped curve" or as a "normal distribution".

Let's examine the STO orbital first. Its formula is shown below. The squiqqly symbol in the equation is the Greek symbol "zeta". You should recognize the symbol for pi, or 3.14. The "e" stands for the mathematical function known as an exponential, and the r represents the radius (in units of angstroms, or 10^{-10} meters). This is the Slater Type Orbital. The value of zeta is the basis set, and in this case that number is 1.24.

$$STO = \left(\frac{\zeta^3}{\pi}\right)^{0.5} e^{(-\zeta r)}$$

If we insert the basis set number into the formula, and then calculate the formula over some distance (say from 0 angstroms to 3 angstroms), we get a graph that looks like this:



What you should observe here is that the wavefunction, or the value of the STO calculation, decreases as the electron moves farther away from the nucleus. Hopefully this is what you would expect to see happen!
You can, by the way, see the exponential nature of the curve as it slopes away to the right, representing a longer distance of the electron from the nucleus.

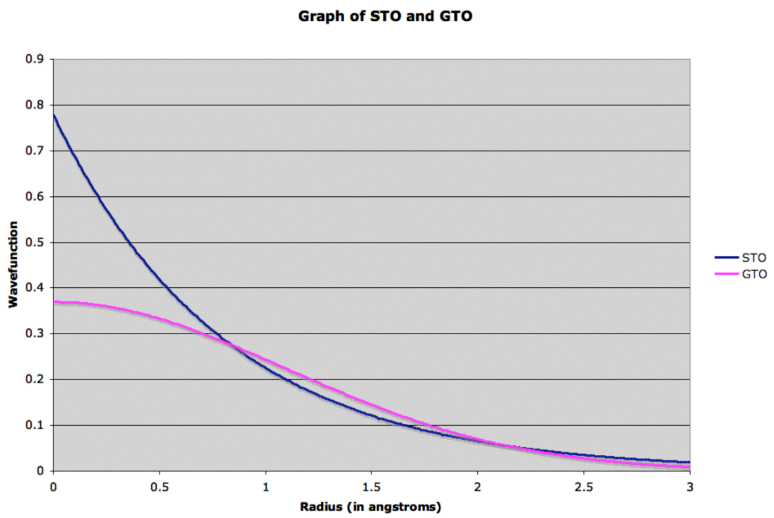
Minimal Basis Sets:

The example above represents a very simple system, and life gets very complicated mathematically once you are in a three-dimensional (x-y-z) system with more than one electron. It turns out that calculating the wavefunction using a Slater Type Orbital becomes extraordinarily complicated for anything more than the very simple system described above. In a world of unlimited computing resources, this would not be an issue, but we don't live in that world! As such, we need some way of approximating the Slater Type Orbital for larger systems.

This is where Gaussian Type Orbitals (GTOs) come into play. A GTO is a different, but similar, mathematical formula. It has the form shown here:

$$GTO = \left(\frac{2\alpha}{\pi}\right)^{0.75} e^{\left(-\alpha r^2\right)}$$

Notice that its form is very similar. You should recognize pi, the exponential function (e), and the radius r. We have a new constant, known as alpha, replacing the zeta value in the STO. For a very simple, one electron system in one dimension, the basis set value of alpha is 0.4166. When this constant value is inserted appropriately, and the formula is plotted over a radius of 0 to 3 angstroms, we get the graph below. This function has been plotted on top of the STO for comparison:



What you should observe is that these are very different plots, especially between 0 and about 1 angstroms. This discrepancy is disturbing, because we really need a better prediction of where the electron is, especially in the vicinity of the nucleus. Assuming that the STO calculation is correct, the GTO approximation fails miserably, at least until we get to one angstrom radius.

We obviously need a fix to this problem. One strategy is to calculate several GTO functions and add them together, with each one contributing to part of the approximation. We can try this with three Gaussian functions, calculating each one, determining its contribution to the overall solution, and then adding them together. Since we now have three functions, we need three different values of alpha. Given that each of the functions contributes to part of the final answer, we also need a value known as the "contraction coefficient". This value will be used in the final approximation. We will show how the contraction coefficient enters into the final calculation a little later.

Where do we get the basis set numbers for this larger problem? We can use the online "Gaussian Basis Set Order Form" (http://www.emsl.pnl.gov/forms/basisform.html), a resource of the Pacific Northwest Laboratories. (NOTE: there is a lab activity that shows how to use this form and how to do the calculation described below using a spreadsheet). Using this resource, we get this series of six numbers:

3.42525091	0.154328970
0.623913730	0.535328140
0.168855400	0.444634540

This is how the basis set appears on the Gaussian Basis Form Order page. The first column of numbers represents the alpha values for each of the three Gaussian calculations. The second column of three numbers represents the contraction coefficient for each of the three calculations. The table below serves to help you understand each of the numbers:

	Alpha (α) value	Contraction coefficient
Numbers for the first GTO	3.42424091	0.154328970
Numbers for the second GTO	0.623913730	0.535328140
Numbers for the third GTO	0.168855400	0.444634540

One way of writing this is as follows:

$$\phi_{1s} = 0.15432897(r, 3.42525091) + 0.53532814(r, 0.62391373) + 0.44463454(r, 0.1688554)$$

This equation calculates the wavefunction (represented by the Greek symbol psi) for the 1s electron. Each of the three GTOs is multiplied by its contraction coefficient. By way of example, the calculation for the first GTO would be as follows:

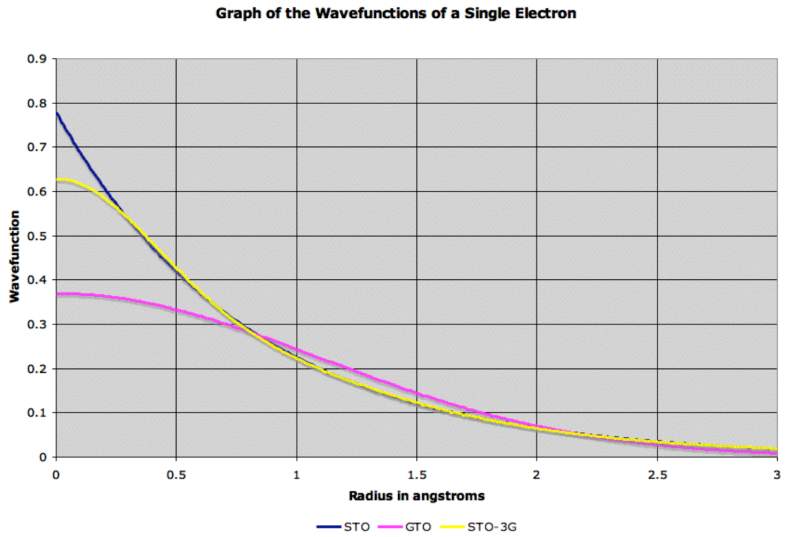
$$GTO1 = \left(\frac{2 * 3.42525091}{\pi}\right)^{0.75} e^{(-3.42525091 * r^2)}$$

So the procedure is as follows:

1. Calculate GTO1, GTO2, and GTO3, using the three alpha values
2. Multiply each of the GTO calculations by its contraction coefficient
3. Add the three products together. The final equation is shown below:

$$\phi_{1s,STO-3G} = (0.15432897 * GTO1) + (0.53532814 * GTO2) + (0.44463454 * GTO3)$$

This basis set is known as a "STO-3G" basis set. This is because it approximates a Slater Type Orbital (STO) by combining three Gaussian functions. If we plot the result of this calculation over top of the original STO calculation and the single GTO calculation, our graph looks like this:



You should notice that we have a considerably better approximation to the "correct" STO calculation with the STO-3G calculation. At the zero radius, the value of the STO is approximately 0.78. With the single GTO calculation, we have a terrible 0.37. With the STO-3G combination calculation, our approximation is close to 0.63, a closer number to 0.78.

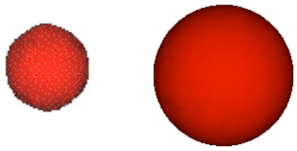
How might we improve upon this? We can add more Gaussian functions as the simplest strategy. For example, we could choose to use a STO-6G basis set. From the Gaussian Basis Set Order Form, you should not be surprised to see six sets of numbers: six alpha values and six contraction coefficients:

35.5232212	0.916359628E-02
6.51314373	0.493614929E-01
1.82214290	0.168538305
0.625955266	0.370562800
0.243076747	0.416491530
0.100112428	0.130334084

There are still, however, considerable problems associated with the STO-nG family (where n is the number of Gaussians used) of "minimal basis sets", some of which will be addressed in the next group of basis sets.

Split-Valence Basis Sets:

One of the major problems with the minimal basis set family (STO-nG), is that it considers all electrons to be equal! Even in beginning chemistry, we differentiate between "inner shell" and "outer shell" electrons, or core and valence electrons (respectively). We use notational systems such as $1s^2 2s^2 2p^2$ to suggest that electrons have different roles in the atomic or molecular systems. In looking at the shapes of these, we see the shape of the 1s and 2s orbitals of carbon:



We need a way to differentiate between electrons in the 1s, 2s, 2p, and other orbitals, and this next set of basis sets provides a way to do that. In the previous section, we improved upon the Gaussian approximation by combining some number of Gaussians, multiplied by the contraction coefficient "fudge factor", to approximate the Slater Type Orbital. We can also improve the accuracy of the STO calculations by doubling the number of STOs. These basis sets are known as "double zeta (DZ)" basis sets, since they use two zeta values. There are also triple and quadruple zeta (TZ, QZ) basis sets, but those are generally beyond the scope of the beginning molecular modeler.

We still have not addressed the fact that the inner shell electrons ("core electrons") and outer shell electrons ("valence electrons") are not the equivalent. A solution to this is to perform a very rough calculation of the contribution of the core electrons, and a more rigorous calculation of the valence electrons. This type of basis set is known as a "split-valence" basis set. An example split-valence basis set is the 3-21G basis set. You should notice that we are still using Gaussian Type Orbitals to approximate the STO, as we did previously. You should also notice, however, that there are two parts to the notation: firstnumber-secondnumberG. The first number – in this case the "3" – says we are going to approximate the Slater equation for the core electrons using a STO-3G basis set. The second number – "21" – doesn't mean we are going to use 21 GTOs for the valence electron. It means we are going to use a double zeta approximation for the valence electrons, and we'll investigate that now.

In the double zeta calculation, we calculate each of the electrons in the valence orbital twice. The first time, we calculate the valence electron(s) using a STO-2G basis set. We then re-calculate the valence electrons with a single STO-1G basis set. You might ask: isn't an STO-2G and an STO-1G simply an STO-3G? No. The reason for this is these different calculations try to approximate that electron orbitals have different sizes. The mathematics becomes complicated quite quickly. All of these calculations require the use of a type of mathematics known as matrix mathematics, especially the use of what is known as a determinant. In this case, the STO-2G approximation calculates the value for a small orbital size, and the STO-1G calculates the value for a larger orbital size. Once all of these are added together – the inner shell STO-3G, the valence shell STO-2G, and the other STO-1G, we are able to get an overall approximation for the wavefunction of the atom or molecule.

Again, from the Gaussian Basis Set Order Form, a 3-21G basis set (in this case, for the atom carbon) is shown below:

```
S   3  1.00
      172.256000          0.617669000E-01
      25.9109000          0.358794000
      5.53335000          0.700713000
SP  2  1.00
      3.66498000         -0.395897000          0.236460000
     0.770545000           1.21584000          0.860619000
SP  1  1.00
     0.195857000           1.00000000           1.00000000
```

You should see three distinct sections: the first set of three numbers represents the alpha values and contraction coefficients for the inner shell STO-3G approximation. So, for example, to calculate the wavefunction of the 1s orbital of carbon, we have:

$$\phi_{1s} = 0.0617 * (r,172.256) + 0.3587 * (r,25,910) + 0.7007 * (r,5.533)$$

The notation being used here is as follows. The first number for each of the three parts is the contraction coefficient. This is multiplied by the numbers in the GTO formula, where r is the radius and the second value is the alpha value. So, for example, in the first part of the formula above, we calculate the GTO formula from 0 to 3 angstroms in radius, using 172.256 as the value of alpha.

This takes care of the two 1s electrons in the carbon atom ($1s^2 2s^2 2p^2$). Now we have to take care of the electrons in the valence shell, in this case the 2s and 2p electrons. The graphic below shows the entire basis set. You should be able to identify the STO-3G numbers in the first two columns on the right. The next column of numbers (3.66498, 0.770545) are the alpha values for the 2s and 2p calculations. Because these orbitals are different size, we have two sets of contraction coefficients – the -0.39/1.21 numbers for the 2s orbitals, and the 0.23/0.86 numbers for the 2p orbitals. These numbers represent the first of the double zeta calculations. The second zeta uses the numbers 0.195857 for the alpha value and 1.0000 for the contraction coefficient for the 2s orbital and 1.0000 for the contraction coefficient for the 2p orbital.

Atom	α_{1s}	Contraction coeff, 1s	$\alpha_{2s,\,2p}$	CC, 2s	CC, 2p	$\alpha'_{2s,2p}$
C	172.256	0.617669E-01	3.66498	-0.395897	-0.2364	0.195857
	25.9109	0.358794	0.770545	1.21584	0.860619	
	5.53335	0.700713				

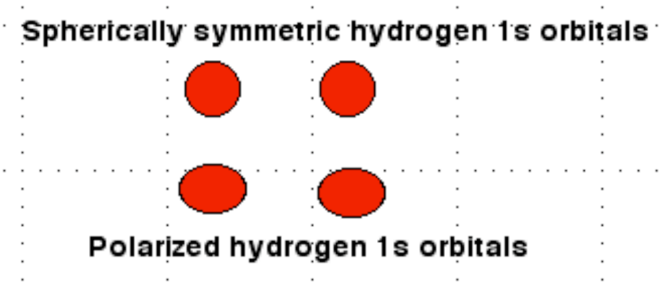
The reader is encouraged to walk through these numbers with patience! It is important to remember that all of these calculations use the GTO formula, where the alpha value is used. It is also important to remember that the formula is calculated over the radius of the electron.

A very popular split basis set is the 6-31G basis set. As the name suggests, this basis set uses an STO-6G calculation for the core electrons, an STO-3G calculation for the first zeta on the valence electrons, and an STO-1G basis for the second zeta of the valence electrons. Most computational chemists use the 6-31G basis set. You will rarely see 3-21G in the research literature, and you will almost never see a minimal basis set (STO-3G) used for research. An example triple zeta (TZ) basis set is the 6-311G basis set. In this set, we approximate the behavior of the core electrons with an STO-6G basis set. We then approximate the behavior of the valence electrons three separate times. Each time we make an allowance for differences in sizes of the orbitals. We do the first calculation using an STO-3G approximation. We do the next calculation using an STO-1G approximation. We do the third, and final, calculation using another STO-1G approximation. Keep in mind that the two STO-1G calculations use different values for alpha, and different contraction coefficients. The basic formula is the same, but the values used in the formula are not.

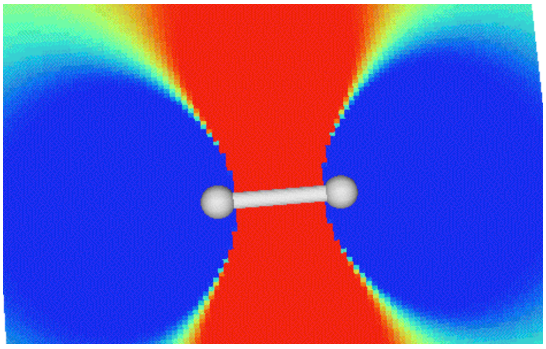
Polarized Basis Sets:

The increase in complexity demonstrated by the split valence basis sets still does not solve all of the problems. Keep in mind that the goal here is to try to approximate what is really happening in the atom or molecule. To do that, we need the best mathematical descriptions of electron behavior possible, given the amount of computational resources we have at our disposal.

One of the problems of minimal and split-valence basis sets is that they don't take into account a phenomenon known as polarization. Imagine two hydrogen atoms, placed near each other. Each hydrogen atom has one electron, located in a 1s orbital. If we ignore polarization, we can move the hydrogen atoms close to each other, and the shapes of neither atom will be distorted. It is as if one atom doesn't know the other one is there! This does not happen to real atoms. As one atom gets close to another, the shape of the orbitals are distorted, depending on the distance, strength, and shape of the neighboring atom. The graphic below shows the two hydrogen atoms, first without polarization and the second with polarization:



Notice that with polarization, the 1s orbitals of each of the two hydrogen atoms are slightly distorted. We can see this effect between two hydrogens that are bonded together. Notice the distortion of the blue circles (which represent part of the 1s orbital). This is due to the effect of polarization.



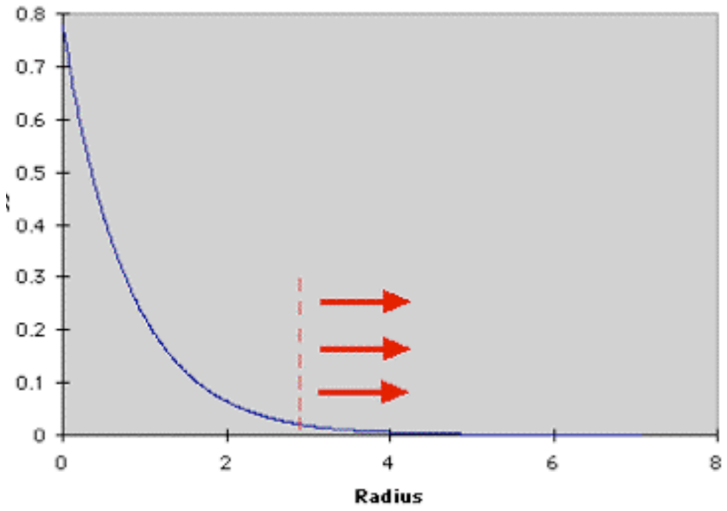
Basis sets that include this additional piece of mathematics are indicated by an "*" symbol. For example, a 3-21G* basis set is a polarized, split-valence basis set.

There is another way of describing polarized basis sets. Once again we return to a simple hydrogen atom. From electron configuration notation, we say that the electron configuration of hydrogen is $1s^1$. We're assuming here that all of the electron is in the s shell, and that a p shell (or orbital) doesn't exist for hydrogen. In adding polarization, we state that some of the 1s electron is in a p-orbital, which doesn't match the $1s^1$ notation used by chemists. Likewise, for elements like carbon, the electron configuration is $1s^2 2s^2 2p^2$. Notice that there is no "d" orbital here. However, to take into account polarization, we give the carbon atom a little "d" flavor. Supposing again that we're adding polarization to a 3-21G split-valence basis set, our notation would be 3-21G*. More chemists, however, are

rejecting that notation in favor of something a little less esoteric! Newer publications use the notation 3-21G(d), where the notation explicitly states that we are adding the d orbital to the calculation. You will also see notation such as 3-21G** or 3-21G(d,p). Both notations mean the same thing. In this case, we are adding a higher level of polarization to the calculation.

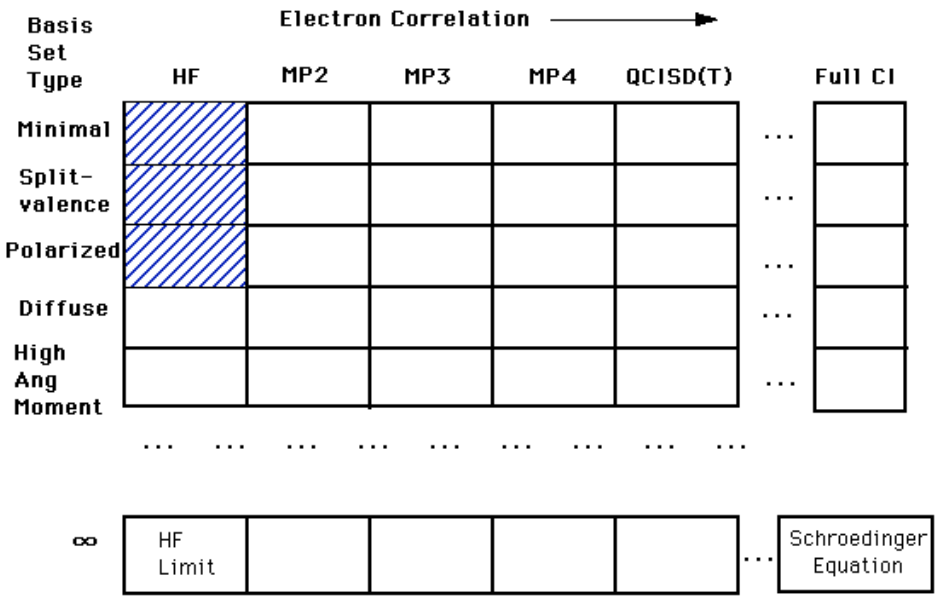
Diffuse Basis Sets:

All of the atoms and molecules we have been talking about have been in the ground state, meaning the electrons are where one normally assumes them to be! In the ground state, the basis sets we have described are adequate most of the time. What about, however, systems that are in the excited state, or those that are anions? In that case, the electrons have the possibility of being found farther away from the nucleus than they would be in the ground state. If you recall, we typically calculated the ground state electron from 0 angstroms to 3 angstroms. Anything beyond that point is probably not significant. With anions and excited systems, however, distances beyond 3 angstrom might indeed be important! As such, with diffuse basis sets, we "add the tail", meaning we include any contributions past 3 angstroms. This gives us a little more confidence that we won't miss parts of the calculation that may indeed be important.



Diffuse basis sets are indicated by a "+" symbol, or a "++" if they are doubly diffuse. As an example, a 3-21+G is a diffuse, split-valence basis set. A 6-31++G(p,d) basis set is a doubly-diffuse, doubly-polarized split-valence set.

The experienced computational chemist understands that the larger the basis set, the more accurate the final calculations will be. The choice of a basis set is determined by the type of calculation being done, the quality of the final calculation desired, and the amount of computing power that one has. In an ideal world, a perfect basis set would yield an exact solution to Schroedinger's equation. As the chart below suggests, we need to have a really big basis set to be able to accomplish that. You should notice the various types of basis sets down the left-hand side of the chart. We have not included high angular momentum basis sets in this discussion. Issues regarding electron correlation will be presented in subsequent chapters.



Finally, many reference manuals for the various computational chemistry software packages provide "quick guide" tables and other references for helping the chemist make appropriate decisions about the choice of a basis set. The table below comes from the Gaussian Users Manual:

Name of Basis Set	Description
STO-3G	Minimal basis set (stripped down in the interest of performance' use for more qualitative results on big systems
3-21G	Double zeta; 2 sets of functions in the valence region provide a more accurate description of orbitals. Use when 6-31G* is too expensive
6-31G* [6-31G(d)]	Adds polarization functions to heavy atoms; use for most jobs on up to medium-size systems (these are the 6-component type d functions)
6-31G** [6-31G(p,d)]	Adds polarization functions to the hydrogens as well; use when the hydrogens are the site of interest (for example, bond energies) and for final, accurate energy calculations
6-31+G*	Adds diffuse functions; important for systems with lone pairs, anions, excited states
6-31+G** [6-31+G(p,d)]	Adds p functions to the hydrogens as well; use when you would use 6-31G** and diffuse functions are needed
6-311+G** [6-311+G(p,d)]	Triple zeta; adds extra valence electrons (3 sizes of s and p functions) to 6-31+G** (five pure component d functions are used)

In practice, you will rarely see a basis set below the 6-31G or 6-31G* levels cited in the research literature. For educational purposes, the use of STO-3G and 3-21G is acceptable, in that it provides the student with the same types of results, just not as accurate. Using one of the smaller basis sets has the advantage, however, of a much faster calculation.

Chapter 5

Mathematics for Computational Chemistry

The fundamental laws necessary for the mathematical treatment of a large part of physics and the whole of chemistry are thus completely known, and the difficulty lies only in the fact that application of these laws leads to equations that are too complex to be solved.l – P.A.M. Dirac

5.1 Key Notes

5.1.1 Fundamental Mathematics:

Molecular modeling is a mathematical science. It attempts to define and describe molecules and their interactions with other molecules in mathematical terms. The computer is then used to solve the mathematical equations, which are numerous and complex. There are two broad categories of mathematics:

1. those that calculate the behavior of the nuclei of the atoms and neglect the behavior of electrons. These are known as **molecular mechanics/molecular dynamics (MM/MD) calculations**.

2. those that calculate the behavior of electrons and neglect the behavior of nuclei. These are known as **electronic structure calculations**.

The *North Carolina High School Computational Chemistry Server* (`http://chemistry.ncssm.edu`) is used to perform these calculations. This chapter, however, deals primarily with electronic structure calculations, with a very short description of MM/DD.

The fundamental mathematical equation to be solved is the **Schrödinger** equation, which looks to be able to help us predict chemical properties by calculating the behavior of electrons. The fundamental property that results from the solution of this equation is the *energy* of the molecule or molecular system.

5.1.2 Fundamental Approximations:

All of the methods use **approximations** to simplify the calculations. Although computer software and hardware are improving, and our ability to describe chemical structure and behavior is improving, we still must accept some simplifications to our models. In electronic structure calculations, there are three primary approximations:

1. the **Born-Oppenheimer approximation**: states that we ignore the motion of nuclei in molecules
2. the **Hartree-Fock (HF) approximation**: states that we can simplify our calculations by aggregating, or combining, the motion of electrons
3. the **Linear Combination of Atomic Orbitals (LCAO)**: states that we can construct molecular orbitals by a relatively straightforward addition of calculated atomic orbitals.

All of these approximations serve to give us less than a perfect solution to whatever calculation we are doing. As the famous statistician George Box declared:

> *All models are wrong, some are useful.*

As the approximations, methods, and tools improve, so too will the accuracy of our calculations.

5.1.3 Solving the Schrodinger Equation:

Despite all of our best efforts, it is still not possible to be able to solve the Schrödinger equation. An exact solution of this equation will result in an energy value that is the lowest possible energy for a given molecular system. Any approximation will result in a higher, and more inaccurate, energy value. In other words, the choice of one or more approximations ensures that our final results will be different from that of the actual molecule. The **variation theorem** states that *the calculated energy will always be greater than the true energy.*

5.2 Fundamental Mathematics:

Molecular mechanics uses mathematics that are the most understandable to most normal human beings! In this method, we apply Newtonian mechanics to determine molecular properties. In Newtonian mechanics, atoms are modeled as if they are hard spheres connected together by flexible springs. By calculating the motion of the balls as the springs are contracted and expanded, using the spring equation (Equation 5.1), we can predict various chemical properties. F represents force, k is the spring constant, and x is the distance between the two hard spheres, representing the length of the bond between two atoms. Using a set of numbers known as a **force field**, molecules can be modeled.

$$F = -kx \tag{5.1}$$

Most molecular modeling tools look to solve the **Schrödinger Equation**. This equation can be used to calculate an approximate idea of the behavior of electrons in an atom or molecule. The equation calculates the **wavefunction** of an electron, which is a mathematical description of the probability of an electron being at some location in relation to the nucleus of the atom. When this equation is calculated, chemists can determine the energy of the atom or molecule. Chemists can also use the calculation results to derive a wide variety of other properties or characteristics of the atom or molecule.

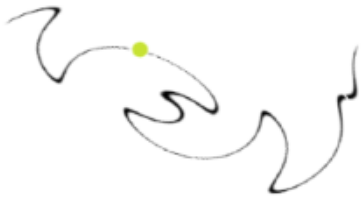


Figure 5.1: Representation of path of electron in Bohr model. The electron follows a precise path.

In discussing the wavefunction, we are interested in the behavior of the electron, particularly its location and motion. In classical (Newtonian) mechanics, we assume that electrons are billiard balls, or particles, that move along a fixed path in the atom, as shown in Figure 5.1. As Niels Bohr postulated, these billiard balls can only exist at very well defined distances from the nucleus. This is the concept of **quantization**. Based on the level at which the electron exists, we can calculate properties such as the energy of the electron and, subsequently, the energy of the atom and of the molecule.

It turns out that the Bohr model does not accurately portray the behavior of electrons. A newer model, known as quantum chemistry, was developed (primarily in the mid-1900s) to address the limitations of the classical mechanics model. In the quantum

model, we cannot predict the specific location of the electron, only the probable location and behavior of the model.

The mathematical formula that looks to predict the probability that an electron might be at some location is known as the wavefunction. We can show this as an **electron cloud** (Figure 5.2). In this graphic, we state that there is a probability that the electron or electrons can be found somewhere inside the cloud.



Figure 5.2: Representation of electron cloud in quantum model. The electron is somewhere in the cloud.

The mathematical equation that represents the calculation of the wavefunction is known as Schrödingers equation (Equation 5.2), named after the Austrian physicist Erwin Schrödinger. In its most well known format, the Schrödinger equation is represented by:

$$H\Psi = E\Psi \tag{5.2}$$

where H is an operator known as the Hamiltonian, Ψ (psi) is the wavefunction, and E is the energy.

This is a very simple looking, but very complicated, mathematical equation! This equation is an **eigenfunction,** a special type of mathematical equation. A short explanation of eigenfunctions follows.

Suppose you have a function such as:

$$f(x) = x^2 \tag{5.3}$$

This is a very simple function. In this case, we perform an operation on the independent value x to obtain a new value, y. In this case, the operation is squaring (raising to the 2nd power).

In calculus, we can perform an operation known as **differentiation**. If we differentiate the function shown above, we get this new result:

$$f'(x) = 2x \tag{5.4}$$

Notice that we now have a new, different function, $2x$ instead of x^2. Since we have a new function as a result of this operation (differentiation), the function $f(x) = x^2$

is not an eigenfunction. Now lets introduce a new function:

$$f(x) = e^{2x} \tag{5.5}$$

Now lets do the same operation (differentiation) on this function. As before, we obtain a new function:

$$f'(x) = 2e^{2x} \tag{5.6}$$

Notice in this new function we have a constant (2) times the original function e^{2x}. Because we have the original function multiplied times a constant, the function $f(x) = e^{2x}$ is known as an **eigenfunction**.

Now take a look back at the Schrödinger equation. This too is an eigenfunction. We perform an operation on the wavefunction (Ψ), using the **Hamiltonian** operator (H). What comes out is a constant, in this case E, the energy of the molecule, and the original function, the wavefunction. Being able to calculate the energy of the molecule is tremendously important, and many other properties of the molecule can be derived from this calculation.

To finish out this mathematics, we should briefly explore the **Hamiltonian** operator (Equation 5.7). An operator is any mathematical action applied to a number, group of numbers, functions, etc. Addition, multiplication are both operations. Finding the square root is an operation. Finding the derivative or evaluating an integral are both operations. The Hamiltonian is also an operator, performing a complex mathematical operation on the atom or molecule.

$$H_{elec} = -\frac{1}{2}\sum^{elec}\left(\frac{\partial^2}{\partial x_i^2}+\frac{\partial^2}{\partial y_i^2}+\frac{\partial^2}{\partial z_i^2}\right)-\sum_i^{elec}\sum_I^{nucl}\left(\frac{Z_I}{R_I-r_i}\right)+\sum_i^{elec}\sum_{j<i}^{elec}\left(\frac{1}{r_i-r_j}\right)+\sum_J^{nucl}\sum_{J<I}^{nucl}\left(\frac{Z_I Z_J}{R_I-R_J}\right) \tag{5.7}$$

The Hamiltonian operator is shown in Equation 5.7. There are four parts to this equation:

1. $-\frac{1}{2}\sum^{elec}\left(\frac{\partial^2}{\partial x_i^2}+\frac{\partial^2}{\partial y_i^2}+\frac{\partial^2}{\partial z_i^2}\right)$: this calculates the kinetic energy of the electrons in the system. The unusual symbol ∂ is a partial differential equation that measures the change in the movement of the electron. The x, y, and z in the denominator represent the location of the electron in three dimensions. The symbol $\sum$ says add all of these calculations together.

2. $\sum_i^{elec}\sum_I^{nucl}\left(\frac{Z_I}{R_I-r_i}\right)$: determines the electron-nuclear attractive energy, that energy that results because of the different charges on the electrons and the nuclei.

The Z notation is the atomic number of the atom ($Z = 1$ for hydrogen), R is the internuclear radius, and r is the interelectron radius.

3. $\sum_i^{elec} \sum_{j<i}^{elec} (\frac{1}{r_i - r_j})$: determines the repulsive forces of electron on electron.

4. $\sum_J^{nucl} \sum_{J<I}^{nucl} (\frac{Z_I Z_J}{R_I - R_J}$: determines the attractive forces of the nuclei in the molecular system

Even a general understanding of this operator should help the reader to understand why computers are necessary. This operation is done thousands, millions, and even billions of times each time a quantum calculation is performed.

5.3 Fundamental Approximations:

Most molecular modeling codes use approximations in order to perform the quantum chemical calculations in a reasonable period of time. An approximation is a mathematical shortcut or simplification of some representation of the problem. Approximations are, in essence, an inexact representation of some reality. In molecular modeling, there are several fundamental approximations:

5.3.1 Born-Oppenheimer approximation:

This approximation states that, due to the size and relative slowness of the nucleus as compared with electrons, most quantum chemical packages assume that it is stationary. In the Hamiltonian above, we only calculate the kinetic and potential energies of the electrons, not that of the nuclei. This significantly reduces the complexity of the calculations, making a calculation possible in a reasonable amount of time for molecules of reasonable size.

5.3.2 Hartree-Fock approximation:

Most chemists also describe the Hartree-Fock approximation. In this approximation we dont correlate the electrons. Suppose we have two electrons in the system, A and B. It is easy to correlate these two electrons, because we can measure the effect that A has on B and the effect that B has on A. However, as soon as we have three electrons (A, B, and C, Figure 5.3)the calculations become much more difficult. We have to measure the effect that A has on B, the effect that A has on C, the effect that B has on A, the effect that B has on C, the effect that C has on A, and the effect that C has on B.

As you can imagine, when we have more than three electrons (and most systems do!), the increase in computations increases exponentially.

In the *Hartree-Fock approximation*, we simplify the calculations by combining two or more electrons and pretending that they are a single electron, albeit a pretty big one. Returning to our three-electron system, we combine B and C now called BC as one electron. Now we first have to calculate the effect that A has on BC and the effect that BC has on A. We then have the electrons switch places. We now combine A and B, forming AB, and calculate the interaction of AB with C.

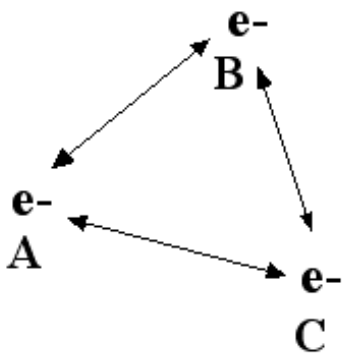


Figure 5.3: A three-electron system, showing the electron-electron interactions.

This system certainly makes the calculations simpler, but it does so at a cost to accuracy of the answer. As is shown in the graphic later in this reading, this approximation establishes a limit (the Hartree-Fock limit) to how accurate our answer can be.

5.3.3 Linear Combination of Atomic Orbitals (LCAO):

In this approximation, we can model a molecular system by calculating the characteristics of the individual atoms in the molecule as if they are by themselves. We then simply add (**linear combination**) the results of the individual atoms to predict the molecular properties. Specifically, we calculate the wavefunction for each atom, otherwise known as the atomic orbital. For example, to calculate the molecular orbitals of water, we calculate the atomic orbital of the oxygen atom and each of the two hydrogen orbitals. Using the LCAO approximation, we then calculate the molecular orbitals of water by relatively simple addition of the atomic orbitals.

It is important to remember that the goal is to solve the Schrödinger equation as exactly as possible. Every approximation used reduces the accuracy of our calculations, and, subsequently, the accuracy of our chemical understanding.

5.4 Solving the Schrodinger Equation:

A more accurate calculation of the Schrödinger equation will always result in a lower energy value for the molecular system. The **variation theorem** states that *the calculated energy ($\overline{E}$) will always be greater than the true energy (E)* :

$$\overline{E} > E \tag{5.8}$$

The graphic shown looks to help visualize this reality.

For a fictitious molecular system, a typical answer results in the Hartree-Fock energy value. Assuming we apply all of the available approximations using the best possible starting numbers, the best we can do is the Hartree-Fock limit. The reader should notice that, in relative terms, this is a very long way away from the exact solution of the Schrödinger equation!

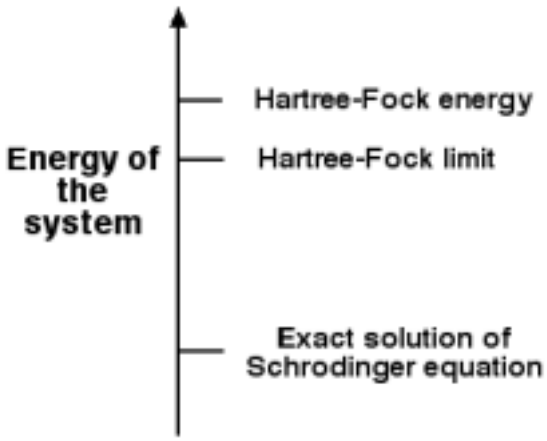


Figure 5.4: The Hartree-Fock Energy Limit

Chapter 6: Molecular Mechanics

Key Notes:

Fundamental Aspects:

Molecular mechanics is the application of classical mechanics to molecules. Classical mechanics is used to describe the motion of macroscopic objects. In molecular mechanics, atoms are treated as spheres whose mass depends on the element. Chemical bonds are treated as springs whose stiffness depends on which elements are bound together, and whether the bond is single, double, or triple. Other types of springs are used to model changes in bond angles, dihedral angles, etc. Each of these various types of springs will have spring constants associated with them. Experimental and theoretical methods are used to determine these parameters. Additional equations from classical physics, such as Coulomb's Law, are used to handle any electrostatic interactions present within a molecule. The sum of all energy terms that apply to a particular molecule are added together to define what is called the "steric" energy, or total potential energy, of the molecule. These energies are not externally referenced (e.g. energies calculated for different molecules cannot be directly compared) and must be used with caution. All of the equations and associated parameters used to calculate each energy term are collectively called the "force field". Different force fields have been developed for different molecular types (e.g. small organic molecules vs. large biomolecules). Since molecular mechanics does not deal directly with electrons and orbitals, we cannot study chemical reactions or predict the reactivity of the molecules studied with this technique.

Applications:

In computational terms, molecular mechanics is the least expensive (fastest) method. It is especially well suited for providing excellent structural parameters in terms of bond distances, angles, etc., for the most stable conformation of a molecule. This so-called "geometry optimization" is often used as the first step before a calculation of another type is performed. This is done to insure that the molecule is in its' lowest energy state so that calculated results can be compared to those done experimentally. Since molecular mechanics is computationally inexpensive, it is often the only method available for use with large molecules, especially those of biochemical interest such as proteins.

Molecular Mechanics Methods:

Although a large number of different methods have been developed, this text will focus on three. For organic molecules with a variety of functional groups, there are two widely used methods known as MM2 and MM3. The MM2 method is a precursor of MM3. The parameters used in these methods were chosen to reproduce the experimental structure and conformational energy differences for individual molecules. The MM3 method has parameters for more atom types and addresses known problems with the MM2 method. The third method is known as OPLS-AA, an acronym for Optimized Potential for Liquid Simulations – All

Atom. This method can also be used with organic molecules but is more widely used in studies of proteins, focusing on reproducing liquid properties, such as heat of vaporization and density.

Molecular Mechanics Software Tools:

The molecular mechanics software program available on the North Carolina High School Computational Chemistry Server is known as Tinker. This program currently utilizes the MM2, MM3, and OPLS-AA molecular mechanics force fields as described above. The program also has the ability to use other, commercially available, force fields.

Advantages:

Perhaps the greatest advantage of molecular mechanics is its computational speed. As the fastest computational chemistry method, this method can be used to study large biomolecules (assuming the required hardware is in place!). For single processor systems like most desktop and laptop machines, calculations on a large biomolecule will still take a prohibitively long time. The main advantage of molecular mechanics with typical computer hardware is in the area of geometry optimization – finding the most stable conformer of a molecule. As long as a good force field is available for the molecule under study, structural results from a molecular mechanics calculation will more closely match experimentally determined structures than other computational chemistry techniques. The method is thus a good choice when studies of molecular geometry are undertaken.

Disadvantages:

The main disadvantage of molecular mechanics, and it is a serious one, is the lack of available parameters for many compound types. Approximately 80% of known compounds do not have parameters available. This severely limits the areas of applicability of the method. Also, since electrons and orbitals are not used in the method, we cannot study chemical reactions or predict reactivity of molecules. Other methods that involve more computational expense must be used to study these properties.

Fundamental Aspects:

The molecular mechanics view of a molecule has spheres of different mass (atoms) connected together by a variety of springs (chemical bonds). If we can find the coordinates of all the atoms at the place where all the springs are at their equilibrium length, this should correspond to the lowest energy state of the molecule. Dynamic behavior of the molecule could also be calculated through application of the laws of classical mechanics.

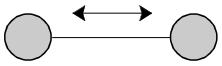
In real molecules, there are other forces present than just those between bonded atoms. There may be charges present that can repel or attract. Repulsions between nonbonded atoms that are close together in space might also occur. These forces may act to change bond angles or cause

twisting around single bonds. To describe the energy of the system, we have to account for all of the different types of interactions that are applicable. The sum of the energy of all of these various components is the basis of a *force field*. A force field allows for calculation of all the forces on the system which in turn gives the energy of the system.

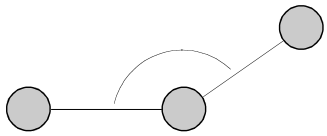
In order to create a force field, we need a mathematical equation for each energy term as well as any required parameters (constants) for these equations. For example, if we are using springs to represent chemical bonds between atoms, we will need to know the strength of each spring (called the spring constant), and the equilibrium distance between atoms, for each different type of bond present. A C-N bond will have a different spring constant and equilibrium distance than an O-H bond. So, these force fields contain a huge number of equations and parameters. The equations come from classical physics, and the parameters come from either experimental data, or from higher level quantum mechanics calculations, which will be discussed in subsequent chapters.

Some of the energy terms that need to be taken into account are:

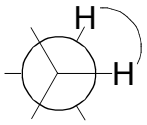
(1) Bond stretching (l):



(2) Bond angle bending (_):

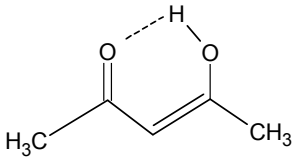


(3) Dihedral angle rotation (_):



(4) van der Waals forces

(5) Hydrogen bonding:



(6) Electrostatic interactions

Since molecular mechanics views chemical bonds as springs, we use an equation from physics called the harmonic oscillator approximation to describe this behavior:

$$E_{stretch} = \frac{k_s}{2}(l - l_0)^2$$

k_s = spring constant; l_0 = equilibrium bond length

Bond angle bending is treated with a similar equation:

$$E_\theta = \frac{k_\theta}{2}(\theta - \theta_0)^2$$

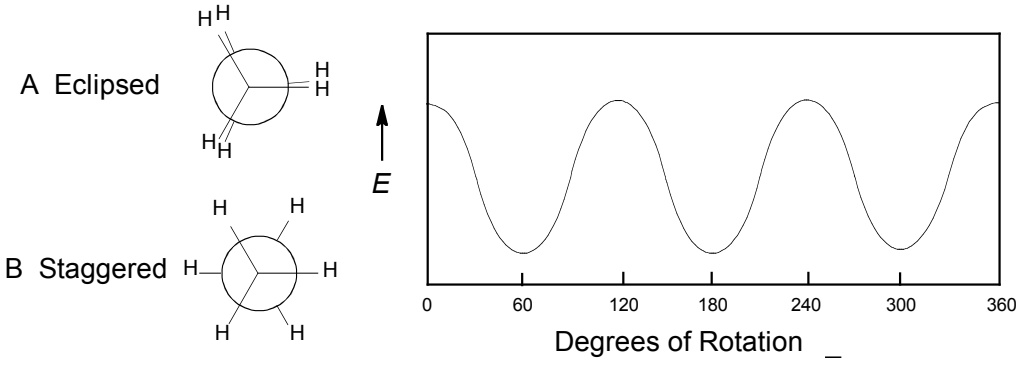
$k_$ = spring force constant; $_0$ = equilibrium bond angle

Rotation about a single bond (torsion) changes the dihedral angle (_) and involves a sum of periodic functions. As the dihedral angle changes from 0° to 360°, the energy profile will begin to repeat itself.

$$E_{torsion} = \frac{1}{2}V_1(1+\cos\phi)+\frac{1}{2}V_2(1+\cos 2\phi)+\frac{1}{2}V_3(1+\cos 3\phi)+\ldots?$$

V_n = dihedral force constant; n = periodicity; _ = dihedral angle

An example is shown below with rotation about the C-C bond in ethane. The eclipsed dihedral angle is taken as 0°, and the staggered form with its lowered steric crowding between H atoms, occurs at 60°:



Neutral atoms undergo a long range attractive van der Waals, or dispersion force. At shorter range, the electron clouds of atoms will begin to repel one another (Pauli repulsion). These two effects are modeled using the Lennard-Jones, or "6-12" potential:

$$E_{vdW} = \frac{A}{r^{12}} - \frac{B}{r^6}$$

A = repulsive term; B = attractive term

Hydrogen bonding is often handled in the van der Waals and electrostatic terms, but is sometimes placed in a separate term. This "10-12" potential decays more rapidly with distance:

$$E_{HB} = \frac{C}{r^{12}} - \frac{D}{r^{10}}$$

Electrostatic interactions are based on Coulomb's Law:

$$E_{electro} = \frac{q_a q_b}{\varepsilon_{ab} r_{ab}}$$

q_n = atomic partial charge; _ = dielectric constant; r = interatomic distance

What is called the steric energy, or total potential energy, of the system is given by a summation of all the energy terms:

$$E_{total} = E_{stretch} + E_{\theta} + E_{torsion} + E_{vdW} + E_{HB} + E_{electro}$$

The force field parameters in all of the above equations are typically determined for an example set of molecules, all of similar type. In order to achieve good results from a molecular mechanics calculation, the molecule of interest should be similar to those used in the parameterization procedure. Some force fields were developed for small organic molecules, while others apply better to proteins, or solid-state oxides, or inorganic molecules.

Applications:

As discussed above the applications of molecular mechanics will depend on the force fields one has available. The North Carolina High School Computational Chemistry Server uses a program called Tinker to do molecular mechanics calculations. This program includes the MM2, MM3, and OPLS-AA force fields. The MM2 and MM3 force fields were designed for use with organic molecules, although some parameters are included for "nonorganic" elements such as Fe, Ni, and Co. The MM3 method was created to deal with some of the problems that became apparent in the MM2 method. Some of these problems include nonbonded H-H repulsions that were too large, inaccurate energy differences in some dihedral angles, and troubles in handling compounds with small (3-, or 4-membered) rings. When developed, the emphasis of both MM2 and MM3 was to model the structures, vibrational frequencies (e.g. infrared spectra), conformational energies and energy barriers, and heats of formation for individual molecules. In a laboratory situation where students are performing calculations on organic molecules, the speed of molecular mechanics combined with the excellent structural parameters obtained through the geometry optimization process make the MM2 and MM3 methods highly attractive for this application. Students could also investigate conformational energy changes of a molecule as single bonds are rotated. The energies reported by the program are not externally referenced - that is, there is no uniform "zero" of energy that all calculated results are calibrated by. Comparisons of energy results between *different* molecules is not likely to work well, although looking at conformational energy differences in the *same* molecule can give useful information.

The OPLS-AA force field was designed to model organic molecules with an emphasis on proteins and condensed phase (liquid or solid) simulations. Unlike the MM2 and MM3 methods, OPLS-AA can also be used to model *inter*molecular (between molecules) as well as *intra*molecular (within the same molecule) interactions. So OPLS-AA is better at condensed phase modeling where intermolecular interactions become more important.

Molecular mechanics methods are widely used to study interactions in biomolecular systems because of the low computational cost. However, before such a task is undertaken it is important to note that the computational time required for these calculations increases as something close to N^2, where N is the number of atoms in the molecule. The calculation speed for a small organic molecule with a few dozen atoms will be *quite different* than that for a small protein that may have several thousand atoms. Calculations on large biomolecules require multiprocessor machines in order to speed things up. Even with the proper hardware, such calculations can take several days!

A typical application for molecular mechanics is in understanding the interactions between potential drug molecules and the sites they interact with, usually in proteins. Changes can be made in the structure of a molecule to see if a stronger interaction with the target site is produced. This is an essential aspect of drug design, and pharmaceutical companies spend a large amount of time and money on computational modeling of these interactions.

Finally, it is up to the user to decide which method to use for a given application. A comparison of structural results for the same molecule obtained using MM2, MM3, and OPLS-AA would be a worthwhile endeavor, especially if the actual experimental results are available. After some practice using the various methods with different molecules, the choice of which method to use in a given situation will become apparent.

Molecular Mechanics Methods:

Both the MM2 and MM3 parameter sets, developed by the Allinger group at the University of Georgia, are targeted at small organic molecules with a range of functional groups, and have been in wide use for over twenty years. These programs are parameterized to provide accurate ground state geometries. Of the various computational methods, molecular mechanics provides the "best" geometrical data that agrees well with experimentally determined geometries. In fact, molecular mechanics is often used to first calculate the optimized geometry of a given molecule before another type of calculation is performed. This is done so that the molecular geometry that the calculation is performed on will be as close as possible to the experimental geometry. If both the calculation and experimentation are performed on a molecule with the same geometry, the calculated and experimental results should be directly comparable.

The OPLS/AA (Optimized Potential for Liquid Simulations/All Atom) parameter set was developed by the Jorgensen group at Yale University and is optimized to fit experimental properties of liquids, such as heat of vaporization and density.

Molecular Mechanics Software Tools:

The TINKER molecular modeling program available on the North Carolina High School Computational Chemistry Server includes MM2, MM3, and OPLS/AA. TINKER also has the ability to use other common parameter sets such as AMBER, CHARMM, and AMOEBA, all of which are commercially available.

Advantages:

The computational speed of molecular mechanics makes this the only viable method for study of large molecules or solid-state materials with many thousands of atoms. Calculations on large systems are tackled using appropriate hardware, typically a high-end multiprocessor machine. Force fields are available for a variety of molecules of interest.

Disadvantages:

In order to perform a molecular mechanics calculation, we must have the required parameters for the molecule of interest. These parameters consist of the set of the various constants from the equations discussed earlier. To have *all* of the constants required for *any* molecule that we might want to study turns out to be highly unlikely. To understand the parameter problem, let's do a crude calculation and see just how many parameters we might need.

The elements that appear most often in typical molecules are a subset of the Periodic Table that includes everything up through Krypton (atomic number 36). Leaving out the Noble Gases (He, Ne, Ar, and Kr) let's pretend that each of the remaining 32 elements could form a bond with every other element and that each element can also form a bond with another atom of the same type. So, with 32 elements we would have:

$$[32(32+1)]/2 = 528$$

We would need 528 spring constants to handle all of the single bonds between these 32 elements. Of course we may also have multiple bonds between elements. One way to look at this is to consider different atom hybridizations (sp^3, sp^2, sp) for single, double, and triple bonds between each of our 32 elements. Now we have 96 different element types, which would lead to:

$$[96(96+1)]/2 = 4656$$

To handle all the bonds would require 4656 different spring constants! We would also need to know the equilibrium bond lengths (l_0) for each of these bonds, giving a total of 9312 parameters. If we include other things we need to know, such as $k_$ and $_0$ values for *every possible* bond angle, various dihedral angle force constants (V_n values), A and B values for van der Waals forces, C and D values for hydrogen bonding forces, atomic charge values (q_n) and dielectric constant values ($_{ab}$) for electrostatic interactions for every possible bond type, etc., we would need something on the order of 10^7 (10 million) parameters! And recall we left out the other ~70 elements of the Periodic Table in our estimate.

The task of experimentally determining (or using theory to calculate) this many parameters is daunting, to say the least! Various approximations are made so that the problem becomes tractable. A simple approach is to have some distance cut-off for terms involving r. Another way to is to limit the number of elements included in a given force field, as discussed previously. Proteins are often simulated via "bead" models that represent each amino acid using two to four

particles, rather than individual atoms. Even with these approximations, ca. 80% of all known molecules do not have adequate parameters for molecular mechanics calculations.

The other drawback of the method is the inability to model certain things of interest to chemists, such as chemical reactions. The harmonic oscillator approximation allows bonds to stretch and compress, but not break and reform new bonds. Also, because we do not have electrons and orbitals in the model, we also can't use molecular mechanics to predict reactivity.

Chapter 7: *Ab Initio* Methods

Key Notes:

Ab Initio Basics:

Ab initio is the Latin term for "from first principles", or, "from scratch". In *ab initio* methods, 100% of the model is done mathematically, based primarily on Schrödinger's equation. Using several constants, such as the speed of light and Planck's constant, and the masses of the electrons and nuclei, we can use *ab initio* methods to calculate a wide variety of properties.

Ab Initio Methods:

The primary goal in the use of *ab initio* methods is the choice of what is known at the model chemistry. The model chemistry describes a mathematical approach to solving the Schrodinger equation for any molecule. In choosing a model chemistry, one proposes a level of theory (such as a Hartree-Fock method) and a basis set (described earlier). At its most basic level, *ab initio* methods state that if one knows the structure of the molecule, one should be able to perform a complete calculation of that molecule completely from mathematical principles.

Applications of *Ab Initio* Methods:

From the mathematics of *ab initio* methods, it is theoretically possible – but often practically difficult – to completely determine everything we might want to know about a molecule. For example, we can determine the energy of the molecule; its vibrational frequencies; its thermodynamic properties; and the values of its molecular orbitals, to name just a few. Our ability to completely describe a molecular system is limited by the computational power available to us at this point in history, and our subsequent need to use approximations to reduce the computational complexity.

Ab Initio Software Tools:

The majority of the currently available software packages have the ability to perform *ab initio* calculations. The commercial package Gaussian (from Gaussian, Inc.) is considered by most to be the "industry standard", although other packages (such as Spartan, CAChe, HyperChem, and others) are challenging Gaussian for computational performance, price, and use by the research community. Gaussian is still, however, the benchmark by which all other *ab initio* codes are measured.

Advantages:

The primary advantage of *ab initio* methods is the accuracy with which calculations are performed. To the degree that a chemist needs to know a property that most accurately matches experimental data, or that most approximates a theoretical prediction, the *ab initio* method is chosen. Fundamentally, *ab initio* is the most accurate and precise of all of the currently available methods in molecular modeling.

Disadvantages:

Ab initio methods can currently only be applied to small molecular systems. As a general guideline, most computational chemists hold the upper limit for use of *ab initio* methods to be around 50 atoms. This upper limit is almost completely dependent on the computational power one has at his or her disposal. As computing power improves (primarily through the use of massively parallel supercomputers), we should be able to come closer to an exact solution of the Schrödinger equation.

The more progress physical sciences make, the more they tend to enter the domain of mathematics, which is a kind of centre to which they all converge. We may even judge the

---

degree of perfection to which a science has arrived by the facility with which it may be submitted to calculation.

Adolphe Quetelet 1796-1874

The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are tthus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble.

P.A.M Dirac 1902-1984

We are perhaps not far removed from the time when we shall be able to submit the bulk of chemical phenomena to calculation.

Joseph Louie Gay-Lussac 1778-1850

***Ab Initio* Basics**:

Ab initio comes from the Latin phrase "from first principles", or, more simply, "from scratch". *Ab initio* is the only computational chemistry method that is 100% mathematical. Unlike other methods that will subsequently defined and described, *ab initio* methods do not use any experimental data or other parameters to attempt to calculate information about a molecule or molecular system. The two quotes shown above describe *ab initio* well: the first states that mathematics can "perfectly" describe a physical system, and this certainly applies to chemistry. The Dirac quote states that (as of 1929, when this quote was made) we know all of the mathematics required to complete describe a chemical system; the only problem being (again, in 1929), we don't have any way to solve them. This is not the case in 2006, now that computers are capable of teraflop (trillions of calculations per second) speeds. And, finally, the Gay-Lussac quote becomes more and more of a reality everyday!

***Ab Initio* Methods**:

Ab initio methods are unarguably the most accurate, as well as the most difficult, of all of the techniques currently in use in the field of molecular modeling. A significant reason for this is that, unlike other methods, the *ab initio* method really does start "from scratch". Beginning with just the molecular structure and a few constants – the speed of light (c), Planck's constant (h), the mass (m_e) and charge (q_e) of the electron – one can calculate a score of chemical properties, make insights into the reactivity of a molecule, and "see" the shapes and sizes of molecular orbitals. All of this comes at a price – both figurative and literal – as is discussed below under "Advantages" and "Disadvantages".

Needless to say, the underlying mathematics of *ab initio* methods are very complicated, involving the solution of integrals, the establishment and solution of complicated matrices, and the establishment of equations that can only be solved through the repetitive abilities of computers. Most of the mathematics found in *ab initio* methods lies well beyond the scope of this Guide, although for a reader who has progressed through a solid year of calculus, the mathematics are accessible.

What is important for all users to understand, is the concept of model chemistry. A model chemistry is a complete mathematical description of the particular calculation. In simplest terms, the model chemistry has two components: the specific theory being used, and the specific basis set that is being used as the starting point for the calculation.

Hartree-Fock (HF) Self-Consistent Field (SCF) Theory:

There are a number of theories, and we will describe a few of them in this reading. The most basic of all theories is the Hartree-Fock method, named after the two physicists (note: not chemists!) who developed the system. The "HF" method is also sometimes known as the "self-consistent field (SCF)" theory, which is a better description of what happens. Most computational chemistry software packages, however, give you pull-down menus that say "Hartree-Fock" or "RHF" (restricted Hartree-Fock, meaning that all of the electrons are paired) or "UHF"

(unrestricted Hartree-Fock, meaning that there are unpaired electrons). Regardless, it is helpful to remember that HF and SCF are referring to the same theory!

So what is the self-consistent field theory? Mathematically, it is quite complicated, but conceptually relatively simple. A procedural description is as follows:

1. Begin with a set of approximate orbitals (a basis set) for all of the electrons in the system
2. Select one electron as a starting electron
3. Calculate the potential (the energy of the system) in which it moves by "freezing" the distribution of all the other electrons by treating their averaged distribution as a single ("centrosymmetric") source of potential
4. Calculate the Schrodinger equation for the selected electron, resulting in a new, more accurate orbital for that electron
5. Repeat the procedure for all the other electrons in the system.
6. A single cycle is complete once each electron has been evaluated
7. Begin the process again with the first electron evaluated, using the newly calculated orbitals as the starting point.
8. Continue this process through the iteration (repeating, or cycling) process until a pass through the calculations does not change the values of the orbitals
9. Declare the calculation to be done, as the orbitals are now considered to be "self-consistent".

Several observations may have come to mind (and if they didn't, you should not be concerned!). If you have not read the chapter on Mathematics, you might consider doing so! In the procedure above, there is no mention of nuclei – the Born-Oppenheimer approximation. The procedure also talks about treating the electrons as "averaged" – the Hartree-Fock approximation. By calculating the energy of an electron as measured against all of the other electrons combined into one big electron, we have an "uncorrelated" system. This lack of electron correlation introduces a fair degree of inaccuracy to our calculations.

Hartree-Fock, or SCF methods, therefore, do not include electron correlation. This limitation is being addressed with the development of newer, "post-SCF" methods that do attempt to take into account electron correlation. Some of these methods are listed below:

- Moller-Plesset (MP) perturbation theory
- Configuration Interaction (CI) theory
- Coupled Cluster (CC) theory

Moller-Plesset Perturbation Theory:
This theory looks to include electron correlation, the first of several methods that work to remove that deficiency from the Hartree-Fock method. The name comes from the basics of the method: in the molecular system, electrons are "perturbated", or moved from a ground state to an excited state, and then allowed to fall back down to the ground state.

- calculate HF wavefunctions (electrons in ground state)
- Move some electrons to excited state
- Calculate wavefunctions of electrons in excited states
- Mix ground and excited states together

There are several levels of the MP theory, indicated by the number following the abbreviation "MP", as in MP2, MP3, etc. The references will often indicate all of these methods with the notation "MP(n)".

Configuration Interaction (CI) theory:

CI Theory has its foundation that is similar, but mathematically different, from that of the MP methods. In this method, an occupied electron orbital – that is, an orbital that is holding electrons – is replaced with a "virtual" orbital, another term for an unoccupied orbital. You might recall that we have done a fair amount of hand waving over the details of the mathematics. We mentioned, for example, that we mentioned the need for very complicated matrices, which are columns and rows of numbers and/or equations. One of the operations that can be performed on a matrix is finding its determinant. In the Hartree-Fock method, the entire wavefunction is represented by a single determinant, shown in the mathematics on the right. In the CI theory, other determinants are formed from these virtual orbitals to expand those found in the Hartree-Fock determinant.

$$\Psi = \frac{1}{\sqrt{n}} \begin{vmatrix} \phi_1(1) & \phi_1(2) & \cdots & \phi_1(n) \\ \phi_2(1) & \phi_2(2) & \cdots & \phi_2(n) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_n(1) & \phi_n(1) & \cdots & \phi_n(n) \end{vmatrix} = |\phi_1 \quad \phi_2 \quad \cdots \quad \phi_n|$$

In CI theory, if we replace a single occupied electron orbital with a single virtual orbital, we call that a "single substitution", and use the notation CIS. Likewise, replacing two occupieds with two virtuals is a double substitution, so indicated by the term CID. Why not replace all of the occupied orbitals with virtual orbitals, which we would label as "Full CI"? As you might be able to determine, the use of Full CI methods is very impractical without a very powerful supercomputer studying very small molecules. The use of single, double, triple, and quadruple substitutions is an acknowledgement of the near-impossibility of using a full CI level of theory.

The problem with doing these substitutions is that it does a fairly poor job of maintaining size consistency. This is a requirement of any theoretical model. This requirement states that the number of errors in a calculation should increase proportionally with the size of the molecule. Another way of describing size consistency is that we can calculate the energy of two non-interacting molecules by adding up the energies of each molecule calculated separately. The molecules would be non-interacting because of their large distance from each other.

Coupled Cluster Theory:

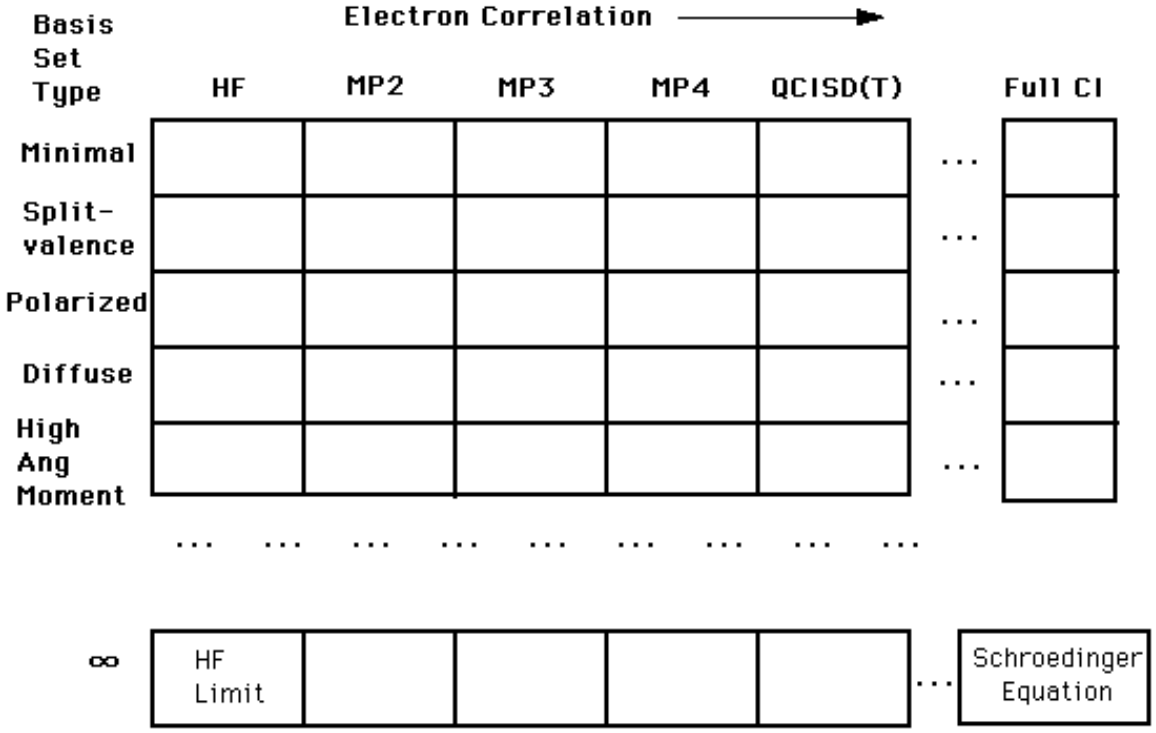
CC methods are the most advanced of the current group of theories. You can identify the coupled cluster theory by a notational system such as CCSD(T), and this method is available on the NC High School Computational Chemistry server, using the Gaussian software package. In this notational system, the "CC" refers to coupled cluster. In the example above, the "SD" refers to the use of a combination of singly and doubly excited electron calculations. The "T" in brackets states that the method also includes a triple virtual orbital, coming from the Moller-Plesset perturbation theory set of mathematics. On the Computational Chemistry server at Shodor, Gaussian and GAMESS offer both CCSD and CCSD(T).

This leads us back to our description of model chemistry. As stated earlier, model chemistry provides a complete mathematical description of how a calculation is to be performed. It consists of our choice of a theory and our choice of a supporting basis set, the numbers used to begin the description of the electron orbital. If, for example, we choose to do a calculation with the Hartree-Fock/SCF theory and a 6-32G* basis set, we would notate our model chemistry as follows:

HF/6-31G*

Our calculation improves if we use a more robust theory – such as one of the electron correlation, or post-SCF methods – and a more robust basis set, such as a triple valence, polarized and diffuse basis set such as 6-311+G(p,d). If it were possible to choose the absolutely best theory and the most powerful basis set, we would reach an exact solution of the Schrodinger equation! We are, however, a long way from reaching that goal. Indeed, an exact analytic solution of Schrödinger's equation is considered by many to be one of the "Holy Grail" areas of modern chemistry.

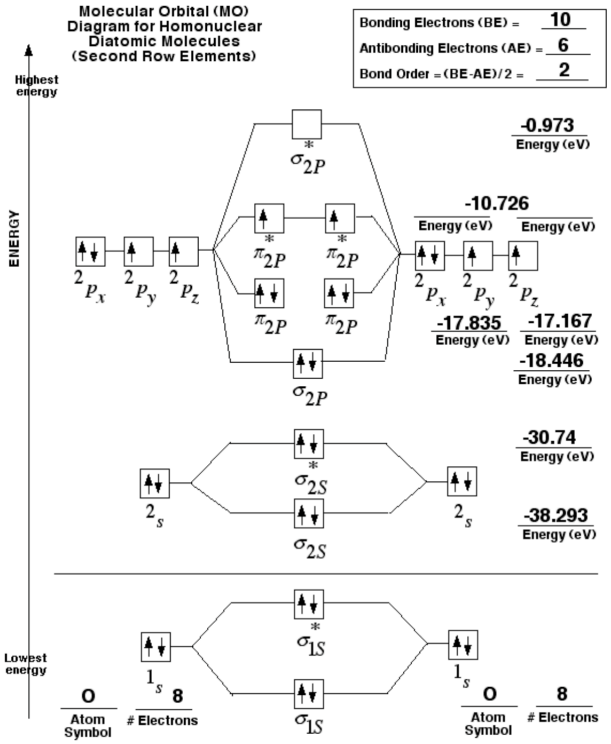
The chart below helps to give the reader an idea of the various model chemistries and their relation to an exact solution to Schrödinger's equation. Each box of the grid represents a unique model chemistry. The discontinuity in the chart implies that there are new model chemistries yet to be discovered!



Applications of *Ab Initio* Methods:

Ab initio methods are the quintessential electron structure determination methods. As such, the primary result of an *ab initio* calculation is the molecular energy. Molecular energies are measured in a unit known as a "Hartree", named in honor of that physicist. This is not a familiar term to most chemists or chemistry students, but the units kilojoules per mole (kJ/mol) or kilocalories per mole (kcal/mole) should be. A Hartree is equivalent to 2625.5 kJ/mol or 627.51 kcal/mol. A Hartree is also equivalent to 27.212 electron-volts (eV), another more familiar energy term.

Atoms have an internal energy, dependent on the number of electrons and the energy levels they occupy. When atoms bond together to form molecules, that bonding changes the energy and orbitals that are occupied by the electrons. The diagram at right is a molecular orbital diagram, constructed for the oxygen (O_2) molecule.



Starting on the left, we show that the oxygen atom has 8 electrons, and we can indicate where they are with the electron configuration notation $1s^2s^2sp^4$. The diagram shows, graphically, the placement of those 8 electrons. Notice, by the way, that we use up and down arrows to represent the electrons. The direction of the arrows is an indication that electrons have spin, _ spin going up and _ spin going down. These are paired electrons. Each of the electrons has an energy value, depending on the energy level of the atomic orbital it occupies. A box represents the atomic orbitals, or AOs. Notice that the 1s atomic orbital is at the lowest, and most stable, energy level. As we move up to the 2s and 2p AOs, the energy level increases.

On the right hand side of the diagram, we show the exact same configuration for the second oxygen atom. Now, what happens when the two atoms of oxygen bond to form molecular oxygen, O_2. (By the way: atomic oxygen is quite toxic, while diatomic oxygen is quite necessary!). Electrons will move into molecular orbitals, or MOs. Starting at the bottom, one electron from the first oxygen atom will move into the σ_{1s} molecular orbital, and one electron from the second oxygen will move to join it. The next two electrons move into the σ^*_{1s} orbital. As we move up the diagram, we have this pairing going on, at least until we get to the "p" levels. At this level, we have 8 atomic electrons. Two of those electrons go into the σ_{2p} molecular orbital, and the next four go into the π_{2p} MO. The last two go into the π_{2p} MO orbital. You should note that these electrons are unpaired. Because of this, the oxygen molecule has a characteristic known as paramagnetism, in this case, diamagnetism.

The diagram also shows the approximate energy levels, in electron-volts (eV) for each of the molecular orbitals (MOs). For example, the σ_{2s} MO has an energy value of -38.293 electron-volts. As we move up the diagram, notice that the energy value gets higher (a smaller negative number). There is also a significance to the use of the asterisk * notation. Any molecular orbital that does not have an asterisk is known as a bonding orbital, whereas those that are marked with an asterisk are anti-bonding orbitals. If we count up the number of electrons in bonding orbitals (10), subtract from that the number of electrons in anti-bonding orbitals (6), and divide that number by 2 (4/2), we get the bond order. In this case, this indicates that molecular oxygen has two oxygen atoms connected with a double bond.

It should be noted that MOs are a mathematical construct, and do not actually exist! They are, however, a useful model. MOs and related concepts (such as Natural Bond Orders, or NBOs) provide the chemistry researcher and chemistry student with an excellent way to predict chemical properties and chemical reactivity. Keeping in mind that MOs are a mathematical representation, and not a physical reality, is a good thing to do.

Ab Initio Software Tools:

Ab initio methods, including both Hartree-Fock (SCF) and post-Hartree-Fock (post-SCF) methods, are found in almost all commercial and free software packages. Of the software packages listed below, the North Carolina High School Computational Chemistry server provides access to two – GAMESS (US) and Gaussian. [Note: the software package NWChem has also been installed, but has not been enabled at this time). Note that some programs, like GAMESS and Gaussian, can also perform semi-empirical, and in the case of Gaussian, molecular mechanics calculations. *Ab initio* calculations are, however, common to most molecular modeling software packages.

Package	Molecular Mechanics	Semi-Empirical	Hartree-Fock	Post-Hartree-Fock methods
ACES	N	N	Y	Y
CADPAC	N	N	Y	Y
COLUMBUS	N	N	Y	Y
DALTON	N	N	Y	Y
GAUSSIAN	Y	Y	Y	Y
GAMESS (UK)	N	Y	Y	Y
GAMESS (US)	N	Y	Y	Y
JAGUAR	Y	N	Y	Y
MOLCAS	Y	Y	Y	Y
MOLPRO	N	N	Y	Y
MPQC	N	N	Y	Y
NWChem	Y	N	Y	Y
PLATO	Y	N	N	N
PQS	Y	Y	Y	Y
PSI	N	N	Y	Y
TURBOMOLE	N	N	Y	Y
Q-Chem	N	N	Y	Y

On the North Carolina High School Computational Chemistry server, users have access to these *ab initio* theories:

- Hartree-Fock

- Moller-Plesset 2
- Moller-Plesset 4
- CCSD
- CCSD(T)

As need arises, more theories will be added to the pull-down menus. The available choices provide the educator and the student researcher with enough variety to explore the various effects of these very different mathematical models. As of this writing (summer 2006), the following *ab initio* basis sets are available:

- STO-3G
- 3-21G
- 6-31G(d)
- 6-311+G(d,p)

Again, these choices are provided to give the user a good, but not overwhelming, sample of very different basis sets. With the five choices of theories and four choices of basis sets, the user can explore in some detail a number of different model chemistries.

Advantages:

It should be clear to the reader that the choice of one of the *ab initio* approaches, which is known as a model chemistry, provides the most accurate computational analysis of a molecule or molecular system possible. Again, as discussed briefly earlier, the use of this methodology allows us, in the words of Gay-Lussac, to "submit the bulk of chemical phenomena to calculation".

Disadvantages:

The disadvantages of this method should not be too much of a surprise! The major disadvantage is that the researcher has significant limitations on the size of the molecule that he or she can study. As a rule of thumb, *ab initio* methods are typically limited to molecules of 50 atoms or less. For the biologist, this, of course, rules out any study of proteins or molecules of biological importance, which are typically thousands of atoms in size.

Even for small molecules, the user must have access to some reasonably significant computing power. While the North Carolina High School Computational Chemistry server is a high-end computing tool, a calculation that has more than 20 atoms and uses one of the electron correlation methods will require run-times that measure in hours. This is not atypical in the computational chemistry community. Educators and student researchers who wish to run calculations of this size will need to request a research account. Classroom accounts, designed to allow educators and students to investigate how the server is used and perform some small calculations, do not provide enough time for the exploration of a model chemistry that incorporates one of the more advanced theories and/or one of the more sophisticated basis sets.

The chart below shows what is known as a "benchmark" test. In this test, we ran the molecule benzene (C_6H_6) using five different levels of theory and four different basis sets, for a total of 20 different and unique model chemistries. The table shows both the amount of computing time required (the "runtime") and the energies of the molecules in units of Hartrees. A careful review of this data should revel that there is a significant change in the runtimes with the triple-zeta (6-311+G(d,p)) basis set, and a reasonable increase with a "standard" basis set such as 6-31G as we increase the level of electron correlation (HF= no correlation to CCSD(T)=substantial correlation).

RUNTIMES (in seconds)

	HF	**MP2**	**MP4**	**CCSD**	**CCSD(T)**
STO-3G	10.8	13.2	16.6	20.9	25.5
3-21G	11.2	14.5	89.0	96.0	172.0
6-31G	14.8	26.7	599.4	393.9	1064.1

6-311+G(d,p)	67.0	214.0	5581.4	3101.6	8390.8

MOLECULAR ENERGIES (in Hartrees)

	HF	**MP2**	**MP4**	**CCSD**	**CCSD(T)**
STO-3G	-227.8905	-228.2386	-228.3095	-228.3129	-228.3211
3-21G	-229.4171	-229.9361	-229.9960	-229.9781	-230.0000
6-31G	-230.7014	-230.7014	-230.7014	-230.7014	-230.7014
6-311+G(d,p)	-230.7551	-230.7551	-230.7551	-230.7551	-230.7551

Chapter 8: Semi-Empirical Methods

Key Notes:

Fundamental Aspects:

Semi-empirical methods represent a "middle road" between the mostly qualitative results available from molecular mechanics and the computationally time-consuming quantitative results available from *ab initio* methods. Semi-empirical methods are a good choice for many users, especially those new to molecular modeling who are less interested in research-quality numerical results, and are more interested in developing their ability to use computing to understand structure, properties, and activities.

Methods:

Semi-empirical methods address the issue of limitations on calculations of large molecules and the length of computing time needed with *ab initio* methods. It does so by making several large assumptions, including ignoring core (non-valence) electrons and making major simplifications of the mathematics. Semi-empirical methods use many of the same mathematics as are found in the Hartree-Fock method, but look to reduce the computing time by replacing some of the mathematics with data (known as *parameters*) derived from experimental and computed data. The various types of semi-empirical methods all use different numbers and types of parameters, which affects the quality of the calculation. The term *semi-empirical* comes from the fact that *some* of the calculations come from *empirical* data. Modern semi-empirical methods include MNDO, AM1, and PM3.

Applications:

Semi-empirical methods are particularly useful in the study of organic chemistry and the structure and reactions of organic molecules. Semi-empirical methods were developed specifically for this area of chemistry, and organic continues to be the primary target for this method. Semi-empirical methods also provide researchers and students with a relatively quick way of studying the structure and behavior of molecules, especially as compared with *ab initio* methods.

Software Tools:

Semi-empirical methods are embedded in most of the modern computational chemistry software packages, including GAMESS and Gaussian, available on the North Carolina High School Computational Chemistry server. MOPAC, also found on the server, is only a semi-empirical tool, and does not perform *ab initio* calculations. MOPAC is the tool of choice for the user solely interested in semi-empirical calculations.

Advantages:

Semi-empirical methods allow the user to obtain qualitative and quantitative results on larger molecules than are possible using *ab initio* methods. It is also an excellent method for use in the study of organic molecules and related reactions. Semi-empirical methods give relatively good results in calculating and visualizing molecular orbitals.

Disadvantages:

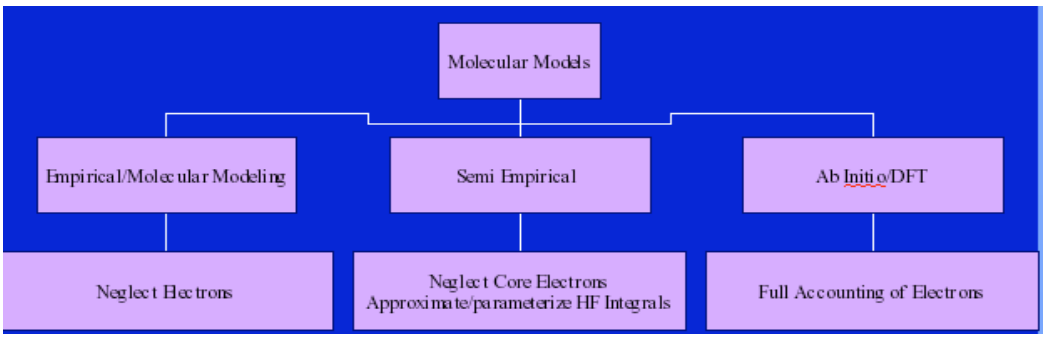
It is not possible to compute all molecules using this method. There are many atoms for which there are not suitable parameters, thus disallowing molecules that contain those atoms. For some parameterized molecules, such as nitrogen, there are well-known inaccuracies in the calculations. Semi-empirical methods do not behave well with hydrogen bonding, transition states, or molecules with non-parameterized atoms.

Fundamental Aspects:

As described in Chapter 7, *ab initio* methods are 100% mathematical, meaning that all of the information generated about an atom, molecule, or reaction comes from the fundamental quantum mechanical calculations (specifically, the Schrödinger equation). This requires significant computing resources; as such, most users are limited to small molecules, typically those consisting of less than 100 atoms.

Semi-empirical methods provide users with a way to study larger molecules. As the name suggests, semi-empirical methods are a combination of *ab initio* methods coupled with the use of data from empirical studies. Through the use of some pre-calculated data, semi-empirical methods allow the user to generate relatively standard information about a molecule. The results obtained are typically less accurate than that of *ab initio* methods, but they are generated more quickly and are possible for larger molecules. A common rule of thumb is that *ab initio* methods scale at a rate of N^4 in terms of computing time (where *N* is the number of basis functions), whereas semi-empirical methods scale at a rate of N^2.

The graphic below shows the relationship between the three basic molecular modeling methodologies: molecular mechanics (empirical modeling), semi-empirical, and *ab initio*/density functional theory (DFT). In previous chapters, this *Guide* has described molecular mechanics, which uses classical physics to describe the motion of nuclei, as if they are attached to springs. With *ab initio* methods, we assume that the nuclei do not move (Born-Oppenheimer approximation), and we focus solely on the behavior of electrons. As is suggested by this graphic, semi-empirical methods try to "split the difference" between these two methodologies.



For scientists, educators, and students, the use of semi-empirical methods provides an efficient way to get meaningful chemical information in a relatively short amount of time that does not put heavy demands on computational resources, especially in a shared/distributed system such as the North Carolina High School Computational Chemistry Server.

Methods:

The graphic above suggests how semi-empirical methods differ from those of molecular mechanics and *ab initio* methods. Molecular mechanics methods do not take electrons into account at all, while *ab initio* methods work to fully account for electron behavior (while fundamentally ignoring nuclear motion and influence). The graphic above suggests that semi-empirical methods do *two* things to improve upon the accuracy of the calculations generated by MM methods, while reducing the significant computing time required for performing *ab initio* calculations:

1. Ignore core electrons
2. Approximate/parameterize the HF (Hartree-Fock) integrals

The first major approximation – ignoring core electrons – contains some chemical common sense. As a general rule, it is the outer electrons – the *valence* electrons – that are of particular chemical interest. Most of the characteristics of the molecule, including its reactivity, come from the specific characteristics and behavior of the valence electrons. In semi-empirical methods, core electrons are not really ignored. Rather, the computations are included with some rather simple (and simplistic) calculations of nuclear behavior. The technique of simplifying calculations on core electrons has the value of significantly reducing computing time, without a tremendous sacrifice on accuracy. For example, in a calculation of methanol (CH_3OH), the semi-empirical calculation only deals with 14 of the 18 total electrons, ignoring the *1s* electrons of carbon and oxygen. The user should appreciate that simply lumping core electrons with nuclei needs its own set of approximations and treatments. These details are beyond the scope of this *Guide*.

It is important and useful for readers to understand that semi-empirical methods are a *type of Hartree-Fock method*, but with some of the Hartree-Fock calculations replaced by the use of empirical (experimental) data. As such, some of the approximations used in HF methods, such as the *Born-Oppenheimer* approximation and others discussed in the previous chapter, also apply to semi-empirical methods. One of the major approximations in semi-empirical methods, however, is the relatively complete neglect of what are known as *two-electron integrals*. These integrals are mathematical representations of the fact that two electrons typically occupy a molecular orbital (for example, a $1s^2$ *orbital*), and that they have a repulsive effect on each other.

It turns out that by eliminating these two-electron integrals, the size of the calculation to be done decreases by a substantial amount. To do this, semi-empirical methods use a type of approximation known as *zero differential overlap*, or ZDO. These methods *parameterize* some of the calculations. Specifically, semi-empirical methods will replace the calculation of the two-electron integrals with data from spectroscopic experimental data. Parameterization means that empirical data is used to create equations or datasets that are stored in the computer code, and accessed at the appropriate point of the semi-empirical calculation.

ZDO actually refers to a family of approximations, grouped according to older and newer methods:

1. Older methods: these methods were developed by John Pople (who also developed the Gaussian software). In this group of methods, data generated by *ab initio* calculations are analyzed using various data fitting algorithms. The results of these data fittings are stored in the software for use during the calculation:
    a) CNDO: **C**omplete **N**eglect of **D**ifferential **O**verlap. Fundamentally, this method does not understand or care that there are bonds between atoms. It simply calculates a wavefunction (molecular orbital) based on the type of atom and its location. Methods such as CNDO/2 are a variant of CNDO.
    b) INDO: **I**ntermediate **N**eglect of **D**ifferential **O**verlap. The "intermediate" part of the method lies in the fact that *some* electron-electron repulsions are ignored, but not those that are centered over the same atom. INDO methods do not have any data (parameters) for atoms with atomic numbers greater than 9, so it cannot be used for molecules containing those atoms. MNDO/3 is a version of an INDO method, not to be confused with MNDO (described below).
2. New methods: all three of these methods use a variant of ZDO known as *Neglect of Differential Diatomic Overlap (NDDO)*. In these methods, the major result of the calculation is that the final energy is reported as a *heat of formation* (ΔH_f), rather than as total energies in units of Hartrees. These methods are attributed primarily to Michael Dewar:

ELEMENT	AM1	PM3	MNDO
H	X	X	X
Li		X	X
Be		X	X
B	X		X
C	X	X	X
N	X	X	X
O	X	X	X
F	X	X	X
Na			
Mg		X	
Al	X	X	X
Si	X	X	X
P	X	X	X
S	X	X	X
Cl	X	X	X
Ca			
Fe			
Cu			
Zn	X	X	X
Ga			
Ge	X	X	X
As	X		
Se	X		
Br	X	X	X
Cd			X
In			
Sn	X	X	X
Sb	X		
Te	X		
I	X	X	X
Hg	X	X	X
Tl			
Pb		X	
Bi			

a) MNDO: **M**odified **N**eglect of **D**ifferential **O**verlap. Parameters for this method come from a statistical analysis (a linear least squares regression fit) of enthalpies of formation and well-known molecular geometries. MNDO methods tend to overestimate repulsive forces between atoms.
b) AM1: Austin Method 1 (so named after the University of Texas – Austin). The AM1 method looks to address the overestimation of repulsive forces by recalculation of the atom-to-atom forces. It does so by multiplying these forces by a sum of Gaussian STO (Slater-type orbital) calculations. In the AM1 method, there are somewhere between 10 and 19 parameters for an individual atom (refer to the list to see which atoms have parameters for a given method)
c) PM3: Parameterization Method 3, developed by J.P. (Jimmy) Stewart in the late 1980s. The "3" comes from the fact that this is the third NDDO method (following MNDO and AM1). PM3 contains many of the same parameters as does AM1, but the parameters were derived more systematically. As compared with MNDO, the parameters are quite different, but the accuracy of the calculation is close to the same. The PM3 method has approximately 18 different parameters for each of the parameterized atoms (11 for hydrogen).

Of the three methods, PM3 is the most robust and the most commonly cited in the computational chemistry literature.

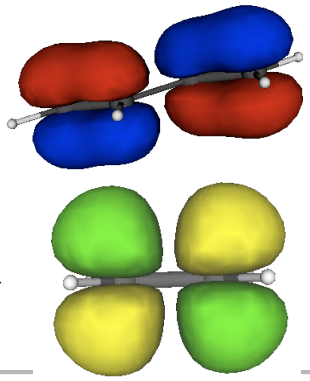
Applications:

Semi-empirical methods are designed specifically for organic systems, and, as such, are particularly well parameterized for those systems. For example, one might use semi-empirical methods for studying the reactivity of organic systems. One such example is the well known (at least to organic chemists!) *Diels-Alder* reaction. This reaction is described as the reaction of an alkene (a C-H compound with one double bond) and a diene (a C-H compound with two double bonds). The alkene is called a *dienophile*, in that it is attracted to the diene. The two compounds react to form a cycloalkene, a ring structure with a double bond. Cycloaddition reactions are an important class of reactions in organic chemistry. Graphically, the reaction is shown as follows:



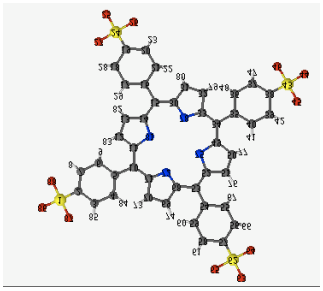
In this reaction, the diene is butadiene (C_4H_6) and the dienophile is ethene (C_2H_4). The two compounds react to from cyclohexene (C_6H_{10}), a six-carbon ring structure.

We can perform a semi-empirical calculation on both butadiene and ethane, running a molecular orbital calculation using PM3. The graphical results are shown. The top image shows the HOMO (highest occupied molecular orbital) for butadiene, and the bottom image shows the LUMO (lowest unoccupied molecular orbital) for the alkene. The molecular orbitals show lobes above and below the double bonds for the butadiene,



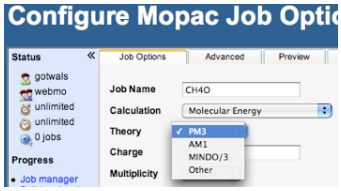
with the blue representing a positive form of the wavefunction and the red representing negative. Likewise, for the ethane we see positive (green) and negative (yellow) wavefunctions above and below the plane of symmetry for the molecule. The significance of these calculations is as follows. Molecular orbital theory tells us that a reaction occurs *if* there is an overlap between a

positive lobe of the HOMO of a molecule and the positive lobe of the LUMO of a molecule, and also between the negative HOMO-negative LUMO. The graphic shows that there is the potential for overlap between blue-green and red-yellow. As such, one can predict that there should be the potential for a reaction between these two compounds, which indeed happens.



Semi-empirical methods can also be used for larger molecules, one of its advantages (as discussed in the appropriate section). The molecule shown is tetraphenylporphine, a complicated molecule composed of a number of carbon and nitrogen ring structures. Typically this compound has a copper (II) atom in the center of the ring structure, and is of particular interest in the study of how metals form complexes with metal ions. Nitrogen compounds, by the way, sometimes give quantitatively questionable results in semi-empirical methods, particularly in the determination of the geometry of the molecule. AM1 methods sometime determine that non-planar molecules are flat, while planar molecules have a pyramidal structure. For all methods, not only semi-empirical, these mathematical "quirks" are part of the business of computational chemistry.

Software Tools:



Semi-empirical methods are found in virtually all major software codes. On the North Carolina High School Computational Chemistry server, semi-empirical methods are a part of GAMESS and Gaussian, and are the central (and only) method available through MOPAC. The three primary Hamiltonians are all from the NDDO family, and include PM3, and AM1. MNDO/3, an earlier semi-empirical technique and part of the INDO family, is also available in MOPAC. For GAMESS there are options for PM3, AM1, and MNDO, while Gaussian only includes PM3 and AM1 (although MNDO and MNDO/3 are available under the keyword system).

Advantages:

The use of semi-empirical methods offers a number of advantages to the user, especially one new to molecular modeling. As is the case with all of the methods – molecular mechanics, *ab initio*, semi-empirical, and density functional theory – there are always trade-offs to be made between accuracy of the answer and the computational resources/time needed to perform the calculation.

The advantages of the semi-empirical method are as follows:

1. Molecule size: Semi-empirical methods came into existence at a time when *ab initio* methods could only be reasonably performed on the smallest of molecules. Even with the increased computing power available now, molecule size is still a major consideration. Semi-empirical methods can produce reasonable results on relatively large molecules (100s of atoms) in a reasonable amount of time. This is especially of importance to the student who might not only have limited computing time (as is the case for classroom accounts on the North Carolina High School Computational Chemistry server) but who also is mostly satisfied with a balance between qualitative and quantitative results.
2. Application to organic molecules: Dewar and his group at the University of Texas – Austin developed semi-empirical methods primarily for the study of organic molecules. Indeed, it was Dewar's goal to develop a type of "molecular orbital spectrometer", specifically as a way to predict whether or not various types of organic reactions would occur. As such, semi-empirical methods are particularly well parameterized for organic systems. Semi-empirical methods can be demonstrated effectively in the study of organic mechanisms such as the

Woodward-Hoffman rules. These rules are of particular importance in the study of pericyclic reactions, which include organic rearrangement reactions.

3. Qualitative and quantitative results: semi-empirical methods give good results in calculating and visualizing molecular orbitals for use in describing the molecular system, in particular in use to predict whether or not a reaction will take place. For molecules for which the atoms are well parameterized, semi-empirical methods give relatively good results, especially as compared with the results one would obtain with *ab initio* methods in the same amount of time.

Disadvantages:

The disadvantages of semi-empirical methods are relatively few, especially for the new user. The major disadvantages are as follows:

1. Accuracy trade-offs: the advantages gained by a quicker calculation are offset by the decrease in accuracy obtained by semi-empirical methods.
2. Problem type limitations: semi-empirical methods tend not to work very well for these types of chemical systems:
    a. Molecules that include hydrogen bonding
    b. Transition structures
    c. Molecules for which the atoms are not parameterized, or for which the atoms are poorly parameterized.

In the balance between advantages and disadvantages, the choice of semi-empirical methods for the computational chemistry student is *highly favored*. Given the relative efficiency of the calculations, coupled with the reasonableness of the results, semi-empirical methods are as close to a "best of both worlds" method as are currently available to the molecular modeler. This is especially true if the goal of the computation is to help the student develop "chemical intuition" about molecular structure, properties, and activities.

Chapter 9: Density Functional Theory (DFT) Methods

Key Notes:

Fundamental Aspects:

Density Functional Theory (DFT) is a computational method that derives properties of the molecule based on a determination of the *electron density* of the molecule. Unlike the *wavefunction*, which is not a physical reality but a mathematical construct, electron density is a physical characteristic of all molecules.

A *functional* is defined as a function of a function, and the energy of the molecule is a functional of the electron density. The electron density is a function with three variables – x-, y-, and z-position of the electrons. Unlike the wavefunction, which becomes significantly more complicated as the number of electrons increases, the determination of the electron density is independent of the number of electrons.

Methods:

There are roughly three types, or categories, of density functional methods. *1) Local density approximation (LDA)* methods assume that the density of the molecule is uniform throughout the molecule, and is typically not a very popular or useful method. *2) Gradient-corrected (GC)* methods look to account for the non-uniformity of the electron density. *3) Hybrid* methods, as the name suggests, attempt to incorporate some of the more useful features from *ab initio* methods (specifically Hartree-Fock methods) with some of the improvements of DFT mathematics. Hybrid methods, such as B3LYP, tend to be the most commonly used methods for computational chemistry practitioners.

Applications:

DFT is a general-purpose computational method, and can be applied to most systems. Like all computational methods, DFT methods are more useful for some types of calculations than others. DFT methods, unlike *ab initio* methods, can be used for calculations involving metals. Hybrid methods, such as B3LYP, are often the method of choice for reaction calculations. Some DFT methods are specifically designed for specific applications, such as the MPW1K hybrid method, designed for determination of kinetics problems.

Software Tools:

DFT methods are now standard in virtually all of the most popular software packages, including Gaussian, GAMESS, HyperChem, and Spartan. On the WebMO interface to the North Carolina High School Computational Chemistry Server, DFT methods are part of the standard options presented to the user. In addition, the user can customize a calculation to include advanced DFT methods, such as time-dependent DFT for determination of excited states.

Advantages:

The most significant advantage to DFT methods is a significant increase in computational accuracy without the additional increase in computing time. DFT methods such as B3LYP/6-31G(d) are oftentimes considered to be a standard model chemistry for many applications.

Disadvantages:

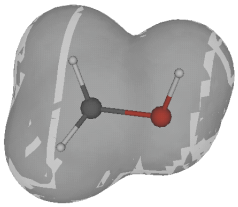
One of the main disadvantages of DFT methods is the challenge in determining the most appropriate method for a particular application. The practitioner should, prior to choosing a DFT method, consult the literature to determine the suitability of that choice

for that particular problem or application. As such, DFT usage tends to favor the more sophisticated user. In general practice (including educational environments), the B3LYP/6-31g(d) model chemistry is considered by most to be a good general-purpose choice.

Fundamental Aspects:

Density functional theory (DFT) is the newest method of the four, although the theory has been around for close to 40 years. It addresses one of the major criticisms of the *ab initio* method. In that method, the energy of the molecule and all of its derivative values depend on the determination of the wavefunction. The problem is that the wavefunction is not a *physical observable*; that is, the wavefunction is purely a mathematical construct. In reality, the wavefunction does not exist! When squared and multiplied by the area of the molecule, the wavefunction is simply a statistical probability that the electron(s) will be at a specific place or part of the molecule. Even though the wavefunction does not exist as a physical, observable property of an atom or molecule, the mathematical determination of the wavefunction (and with it, the atomic and molecular orbitals) has been a good predictor of energy and other actual properties of the molecule.

For a long time, computational chemists have been trying to find some property of atoms and molecules that actually exists, and that can be used to determine the energy and derivative properties of atoms and molecules. Llewellyn Thomas and Enrico Fermi were able to determine that there was a *one-to-one* correspondence between the *electron density* of a molecule and the wavefunction of a molecule with multiple electrons. Thomas was a professor at North Carolina State in the later years of his career, and he died in Raleigh in 1992. All of his teaching and research notes are on special collection at the NCSU Library. Fermi built the first nuclear reactor and was instrumental in the development of the atom bomb as a part of the Manhattan Project.



Because of their work, it was determined that if we can determine the electron density of a molecule, we can say numerous things about the molecule, and this forms the basis for density functional theory. The molecule shown is methanol (CH_3OH), and the electron density mapping is visualized. There are several major advantages of this approach. The first is that the method is based on a property that exists in real molecules, not a purely mathematical invention. The second is that the wavefunction becomes more complicated mathematically as the number of electrons increases. In DFT, the density depends only on the x-y-z coordinates of the individual electron. In practical terms, DFT can be said to *scale* three-dimensionally, or as N^3, where N is the number of basis functions. *Ab initio* methods, on the other hand, scale as N^4, and as a result DFT calculations are faster with better accuracy. There are, of course, still major approximations that are used in DFT that affect the computing time and accuracy of molecules that are evaluated with DFT methods.

The fundamental underlying mathematics of this method is the *functional*. Most readers should have encountered the idea of a *function* in one or more mathematics classes. We say something is a function of some other factor. For example, we might say that "height is a function of age". In this case, the mathematics might be represented by age being the independent variable (represented by "x" on a graph) and height being represented by the dependent variable ("y" on a graph). This function is probably fairly linear, at least until the age of 20 or so. Mathematically, we use the notation *f(x)*, read as the "the function of x". Our age-height function might be represented by the slope equation, where $f(x)=mx+b$, where $f(x)$ is the height (y), m is the slope, x is the age (in years), and b is the y-intercept. In mathematical terms, a function is denoted as follows:

$$y = f(x)$$

A *functional* is a function of a function. Conceptually, this is no more difficult than the concept of a function, but it is not a mathematical concept encountered by most high school or college students. In a function, the dependent variable (y) depends on one or more *single variables*. In the example above, the dependent variable (height) depends on *one* single variable (age). In *ab intio methods*, we are interested in the *wavefunction*. In theory, the wavefunction depends on the electron coordinates – x, y, and z. In practice, however, the wavefunction depends on other variables, and the complexity of the wavefunction increases as we increase the number of electrons. The wavefunction is dependent on *3N* variables, where *N* is the number of atoms. In computational chemistry, a number of approximations need to be made in order to solve the mathematics with our current computational capabilities. Schrödinger's equation states that the energy of the molecule is a function of the wavefunction. Determining the wavefunction becomes the goal of modern computational chemistry.

Mathematically, a *functional* is denoted as follows:

$$y = F[f(x)]$$

In this notation, the value of y is in and of itself dependent on another function. The first *f(x)* function becomes the input for the functional, that is, a function of a function.

In DFT methods, the energy of the molecule is a *functional* of the electron density. Electron density is a function, with three variables: x-position, y-position, and z-position of the electrons. Regardless of the number of electrons, the electron density function is *always* only dependent on those three numbers. The *functional (F)* of electron density gives us the energy of the molecule. The practical advantage is that the mathematics does not spiral out of control as we increase the number of electrons.

$$\text{Electron density} = \rho(x,y,z)$$
$$\text{Energy} = \text{F}[\rho(x,y,z)]$$

The goal in DFT now becomes to find the value of the functional F, and to do this we need to make approximations. Indeed, one of the reasons why there are so many different DFT methods is that there are multiple ways of approximating the functional. For purposes of this *Guide*, and to avoid overwhelming the reader, the number of approximations presented will be limited to three or four.

It is perhaps instructive, however, to delve a little more deeply into the mathematics. In the 1960s, Hohenberg and Kohn were able to use the Thomas-Fermi theorem to develop a more detailed version of the theory. This in turn was adapted by Kohn and Sham (1965) into a practical version of the density functional theory. The KS theory, which describes the mathematics of electron densities and their subsequent correlations to molecular energies, is shown in its simplest form as follows"

$$E_{DFT}[\grave{A}] = T[\grave{A}] + E_{ne}[\grave{A}] + J[\grave{A}] + E_{xc}[\rho]$$

where E is the energy, T is the kinetic energy of the electrons, E_{ne} is the nuclear-electron attraction (Coulombic) energy, J is the electron-electron repulsive (Coulombic) energy, and E_{xc} is the electron-electron exchange-correlation energy. Notice that each of these terms is a function of the function ρ, the electron density, which is itself a function of the three positional coordinates (x, y, and z). As such, each of the terms above – T, E_{ne}, J, and E_{xc} is a functional. The challenge now becomes to determine the value of each of these four functionals.

It turns out that the first three can be determined reasonably well using *ab initio* or semi-empirical methods. This, by the way, makes it difficult to classify DFT. Some computational chemists consider it to be a variation of the *ab initio* method. Some consider it to have connections to semi-empirical, while others

classify it as a completely separate method. For purposes of this *Guide*, DFT is considered to be one of the four basic methods – molecular mechanics, semi-empirical, *ab initio*, and DFT. Regardless of how one determines the first three terms, the last one – the electron exchange-correlation energy – is the term that causes the most concern.

There are a large number of approximations that attempt to calculate the electron exchange-correlation energy. The electron correlation aspect addresses how an electron in an atom or molecule interacts, or "sees", another electron. The electron exchange aspect describes a quantum mechanical property of electrons that is related to their exchange between a fermion and a boson (a *fermionic* or *bosonic* electron). This exchange is well beyond the scope of this *Guide*, but suffice it to say that it is related to the Pauli Exclusion principle, which states that no two electrons can occupy the same energy state.

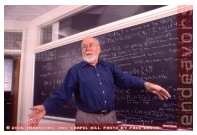
Given this discussion, the next section addresses various methods.

Methods:

Methods in DFT are complicated and diverse, but can roughly be divided into three classes:

1. Methods that use a *local density approximation (LDA)*. The LDA is determined solely based on the properties of the electron density. The *critical assumption* of this approximation is that, for a molecule with many electrons in a gaseous state, the density is uniform throughout the molecule. This is not the case for molecules, where the electron density is decidedly not uniform. This approximation does, however, work well with electronic band structures of solids, which describes the range of energies in which electrons are permitted or not permitted (forbidden). Outside of this application, however, local density approximations are not very satisfactory.
2. Methods that combine the electron density calculations with a *gradient correction factor*. A gradient in mathematics is a function that measures the rate of change of some property. In this case, the gradient looks to account for the non-uniformity of the electron density, and as such is known as *gradient-corrected*. Another term for this is *non-local*.
3. Methods that are a combination of a Hartree-Fock approximation to the exchange energy and a DFT approximation to the exchange energy, all combined with a functional that includes electron correlation. These methods are known as *hybrid* methods, and are currently (Fall of 2006) the most common and popular DFT method used in practice.

The table below provides a good summary of sample methods by name, acronym, and type. [Historical aside: it should be noted that two of the chemists noted are North Carolinians: Bob Parr (shown below) is a quantum chemist at the University of North Carolina at Chapel Hill (read the article: http://research.unc.edu/endeavors/fall2004/parr.html) and Weitao Yang is chemistry faculty at Duke University. Chengteh Lee was Dr. Parr's graduate student and Dr. Yang was his post-doctoral student. Their 1988 paper, "Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density", is one of the most cited papers in the chemistry literature. The paper is routinely referred to as the "LYP" paper.]



Name of the method	Type	Acronym
Hartree-Fock Slater functional	Hartree-Fock with local density approximation exchange	HFS
Vosko, Wilks, and Nusair	Local Density Approximation (emphasis on electron correlation approximation)	VWN
Becke correlation functional; Lee, Yang, Parr electron exchange functional	Gradient-corrected LDA functional	BLYP
Becke 3-term correlation functional; Lee, Yang, and Parr exchange functional	Hybrid DFT	B3LYP

Perdew 1986 functional	Gradient-corrected LDA functional	P86
Becke 3-term correlation functional; Perdew correlation term	Hybrid DFT	P3P86
Modified Perdew-Wang one parameter hybrid for kinetics	Hybrid DFT	MPW1K

For the mathematically inclined readers, the B3LYP functional has this form:

$$E_{xc} = (1-a_0)E_x(LDA) + a_0E_x(HF) + a_xE_x(B88_x) + a_cE_c(LYP88_c) + (1-a_c)E_cVWN80_c$$
$$\text{where } a_0 = 0.2, a_x = 0.72, a_c = 0.81$$

The purpose of including this complex mathematical function (functional) is to suggest the hybrid nature of the mathematics. Note that various approximations – local density approximation (LDA), Hartree-Fock (HF), Becke-1988 (B88), Lee-Yang-Parr 1988 (LYP88), and Vosko,Wilks, Nusair 1980 (VWN80) – are part of this hybrid functional. The lower case *x* refers to the determination of the electron exchange while the lower case *c* refers to the determination of the electron correlation.

Of these methods, the B3LYP functional is considered to be the "industry standard" in terms of practical applications, and it is this method that is available as a pull-down option on the North Carolina High School Computational Chemistry server.

Applications:

Density functional theory is a general-purpose computational chemistry method, and as such, can be applied to most systems. With the number of variations of this method – we have only shown seven in the table above – there is no simple guide to the "correct" choice of a DFT method. On average, there is considerable consensus that DFT methods are "better" than *ab initio* methods, and are generally stated to be more accurate with lower computational expense. There are, of course, some general guidelines for the choice of a DFT method:

1. B3LYP, run with a 6-31G* or better basis set, is *on average* the best choice of a model chemistry for most systems. B3LYP/6-31G* is particular good for organic molecules, but less so for metal-containing compounds.
2. BLYP with most basis sets is the opposite of B3LYP: not particularly accurate with organics, but provides reasonably good energy values for metal-containing compounds.
3. BLYP and B3LYP methods perform close to the same for determination of charge densities on atoms in molecules.
4. Both gradient-corrected and hybrid methods provide high levels of accuracy in the determination of a geometry optimization.
5. B3LYP methods clearly provide better results for reaction chemistry calculations
6. DFT methods are considered by some, but not all, to produce unacceptable results for weak hydrogen bonding interactions.

This incomplete list points to the challenge facing all computational chemists, whether they are using computational chemistry to solve problems or are trying to develop new methodologies. That challenge is finding a method or methods that are a suitable compromise between accuracy and computational expense. Most chemists do not have unlimited computing resources, and as such need to use approximation methods in order to generate results. For theoreticians, the challenge is to develop new approximations that closely mimic the actual chemistry of the atoms and/or molecules, all the while being appreciative and responsive to the needs of the practitioner.

Software Tools:

DFT methods are embedded as "standard" model chemistries in most of the *ab initio* software packages, including GAMESS and Gaussian. On the North Carolina High School Computational Chemistry server, the use of Gaussian is recommended for DFT calculations.

On the WebMO server, one can modify jobs by hand using the "Preview" tab on the Configure window. For example, for excited state chemistry (NMR determinations), the use of the time-dependent (TD) DFT method is recommended. For example, an NMR job for water was set up using the pull-down menus. From the "Preview" tab, we clicked on the Generate button, resulting in the line below:

```
#N B3LYP/6-31G(d)   NMR
```

We modified this input file to include time-dependent DFT, calculating 12 excited states (**nstates**) for the water molecule. Notice we also added a "**+**" to the basis set, indicating the use of a diffuse basis set, recommended for excited state calculations.

```
#N B3LYP  td=nstates=12/6-31+g*
```

In short, the various software packages have a wide range of additional options beyond what is indicated by the available pull-down menus from WebMO. As your experience increases, you can take advantage of this flexibility.

Advantages:

One important advantage of DFT was described early in this chapter: that being that DFT scales three-dimensionally, or as N^3, (N = number of basis functions). *Ab initio* methods, on the other hand, scale as N^4. As a result DFT calculations are slightly faster with better accuracy. Perhaps more importantly, DFT methods overcome one of the main disadvantages of *ab initio* methods such as Hartree-Fock: the complete neglect of electron correlation. Electron correlation is defined as the difference between the Hartree-Fock energy and the exact solution of the Schrödinger equation. DFT methods account for some of this difference with no increase in computational time.

DFT can also perform calculations on some molecules that are not possible with *ab initio* methods, most notably transition metals.

Disadvantages:

Not unlike other methods, the computational chemist must make decisions about which DFT method to use for a particular application. For example, the BLYP method is considered by some (most?) to be appropriate for transition metal applications, but not for organic compounds. B3LYP, on the other hand, has the opposite characteristics. MPW1K, as the name suggests, is particularly well suited for use in modeling kinetics of reactions via determination of transition states. It is, however, a bad choice for stable molecules.

Having said that, we frequently recommend the use of DFT methods to students doing independent research, unless the focus of their research is on comparing model chemistries. For example, we encourage students to use the B3LYP/6-31G(d) model chemistry as the "standard" for projects involving relatively small molecules. With this model chemistry, students can obtain sufficiently accurate results without consuming significant amounts of computing resources.

.

Chapter 10: Basic Molecular Modeling

Key Notes:

Guiding Questions:

Basic molecular modeling requires that the user be able to address two sets of questions. The first set of questions revolves around the chemistry to be solved. The user needs to define the problem, have an idea of what properties or quantities need to be calculated, and have some idea of what the answer might look like and how one might know if the answer is "right". The second set of questions revolves around understanding how molecular modeling will help the user to study a specific chemical behavior. The user must also be able to answer questions related to the choice of methods, computational engines, and other aspects of the calculation. The user must also understand the capabilities and limitations of the software packages available.

Practical Molecular Modeling:

Molecular modeling is a chemistry research tool that is no more difficult than that of traditional chemistry research tools such as test tubes, beakers, or infrared spectrophotometers. Modeling is accessible, "hands on", and does not need to be intimidating. Like any new skill or topic area, there are parts of molecular modeling that are harder than others. Modeling is an important skill area for current and future chemists. It provides some options to the chemistry researcher that are difficult or impossible using traditional experimental techniques. Like any other tool used in chemistry, modeling is used to help chemists explore interesting and challenging problems.

Getting Started:

Students new to molecular modeling can get started by trying to replicate computational studies that have already been solved. By using resources such as lab activities (such as those found at the end of this *Guide*), new users can begin to understand the types of chemistry problems that can be addressed with molecular modeling. They can also begin to learn the terminology and methods used by molecular modelers to address specific types of questions. As the user develops more *competence* and *confidence*, he or she is then able to pose more open-ended questions that eventually lead to doing original research using molecular modeling. In terms of process, the researcher typically follows a four-step approach: define the problem; build the model(s); do the calculation(s); and, analyze the results. Each of these steps requires the user to develop some specific knowledge and skills.

Calculated Properties:

Molecular modeling allows the user to calculate information about a chemical system that is complementary to that information that can be obtained in the laboratory, and also allows the user to calculate information that *cannot* be obtained in the laboratory. Some of the more common properties that can either be calculated *directly* or can be *derived* from computational properties are: single point energies (SPEs), molecular orbital determinations, dipole and higher moments, population analyses, vibrational frequencies and related infrared and Raman spectra and thermodynamic properties, transition structures, UV-VIS spectra, coordinate scans, NMR (nuclear magnetic resonance), electron densities, and electrostatic potential maps. All of these calculations provide us with insight into the structure and/or reactivities of molecules or molecular systems.

Fundamental Units:

Molecular modeling uses its own set of units, which are typically not encountered in standard experimental/laboratory chemistry. Units of length include the *bohr*, *angstroms*, *nanometers* and *picometers*. Units of energy are typically reported in *hartrees*. Dipole moments are reported in units of *debyes*. Bond lengths are typically reported in units of *angstroms* (1×10^{-10} meters, Å), and bond angles and dihedral angles are reported in *degrees*. All of these units can be converted

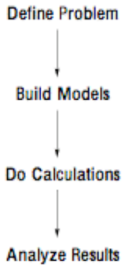
to other units, such as kilocalories per mole (kcal/mol) for energy units or Coulomb-meters for dipole moments expressed in units of debyes.

Verification and Validation:

In molecular modeling, we are anxious to ensure that we have the "right" answer to any molecular calculation. Correctness depends on *validation* and *verification*. Validation answers basic questions about the chemistry being studied. It answers questions such as: do I understand the chemistry I am trying to study? Am I asking the right question? Am I calculating the right property? Verification answers basic questions about the proper use of the computing tools. It addresses questions such as: am I using the software correctly? Have I built the molecule correctly? Do I understand how the software performs its calculations? Is the software working correctly?

Guiding Questions:

Molecular modeling is a scientific tool, and as such, it is designed to find answers to interesting questions. In their online document "Molecular Modeling in Undergraduate Chemistry Education", Warren Hehre and Alan Schusterman, developers of the **Spartan™** computational chemistry software package, describe the process of molecular modeling in this four-step process:



Needless to say, it's not *quite* that simple, but at its core this is an accurate description of the process. The key is the idea of *defining the problem*. Researchers come to the molecular modeling process looking to solve some problem that perhaps cannot be solved in the experimental laboratory. Or, they are looking for insight into the behavior of a molecular or molecular system prior to studying that system in the lab. Regardless, the problem drives the type of molecular modeling they might do, what software or methods they might choose, and what kind of computing power they might require.

For molecular modeling students, however, it is often the case that a problem will be chosen as a way to provide them with the opportunity to learn about a particular method or piece of software. The final answer is less important than the path taken to get to that final answer. Sometimes it is the case that we use molecular modeling in education to reinforce a particular concept, such as bonding, or to help students visualize complex features such as molecular orbitals. As such, in the educational environment, sometimes we are trying to learn *how* to use the tool to study chemistry, sometimes we are *using* the tool to do chemistry, and, most of the time, it's a combination of both. In an analogous way, we teach students the proper techniques for titrating an acid. For the first several times, we are more concerned that the student is using the burette correctly, and less concerned with the underlying chemistry. As the student develops expertise in the use of the tool, we then concentrate on the chemistry to be learned.

For users new to molecular modeling, especially molecular modeling students, there are six general questions that the student should be able to answer. Assuming that these questions can be answered at some level, one can say that the student *knows* something about molecular modeling. [Note to teachers: in our high school course on computational chemistry, these questions are the questions used on the final exam. A copy of the final exam is found in the Lab section of this *Guide*).

1. What is the role and purpose of computational chemistry? What does computational chemistry allow us to do that cannot be done using "traditional" (i.e., wet) chemistry?
2. What is the fundamental mathematical expression that needs to be solved in doing computational chemistry? What are the terms in this equation, what is their significance, what variations can be used?
3. What are the approximations that can be used in doing computational chemistry? What are the pros and cons of the various approximations? How does the choice of approximation affect the results, the computing time, etc.
4. There are roughly four different "flavors" to computational chemistry: *ab initio* methods, semiempirical methods, density functional theory (DFT) and molecular mechanics. What are these methods? How do they differ?
5. What are the fundamental units of measure used by computational chemists? What are some different ways that these fundamental units might be expressed?
6. What are some of the computer codes that one might use to do computational chemistry? What platforms are needed for these codes, what are the strengths and limitations of these codes?

These six questions comprise the basics that a molecular modeler needs to know about his or her craft. It is also assumed that the modeler has some understanding of the chemistry that he or she is investigating. The organic chemist, for example, is often interested in determining what reaction path a particular family of organics might take, given a choice of several different pathways. The inorganic chemist might be interested in finding how a coordination complex might form bonds between a metal atom and other components of the molecule. For all modelers, however, we can generalize by suggesting that there are fundamentally *three things that molecular modeling can tell us*:

1. Structure:
    a. What does the molecule look like?
        - Atoms, bond lengths, bond angles, symmetry
2. Properties:
    a. What characteristics does the molecule have by itself?
        - Molecular energy, dipole moment, structure of the molecular orbitals, frontier orbitals (HOMO and LUMO)
3. Activity:
    a. How does the molecule behave in the presence of other molecules?
        - Electrostatic potential, nucleophilicity, electrophilicity

It is not often the case that we investigate all three of the aspects listed above, but these provide a reasonable cognitive framework for the types of things that molecular modeling can tell us.

Practical Molecular Modeling:

New users are often intimidated by technologies such as molecular modeling. For one thing, the *jargon* is esoteric-sounding and full of abbreviations and acronyms. Some users are new to computing in general, and are worried about which button to click on or are nervous that they will "break" the computer. Molecular modeling is no different than any other field of study. With a little effort, and the use of resources such as this *Guide*, molecular modeling is a very approachable discipline. The structure of this *Guide* is designed to facilitate and encourage the new user: the "Key Notes" section provides *just enough* information to give the new user some *confidence*, while the detailed descriptions are meant to give the user *competence*.

Hehre and Schusterman provide this list of encouragements to the new user:

1. *Modeling is accessible*
    - Anyone can build a useful model.

2. *Modeling is "hands on"*
    - Molecular modeling, like experimental chemistry is a "laboratory" science, and must be learned by "doing" and not just reading.
3. *Modeling does not need to be intimidating*
    - The underpinnings of molecular modeling (quantum mechanics) are certainly intimidating to many chemists, but so too are the underpinnings of NMR. Using molecular modeling should be no more intimidating than obtaining an NMR spectrum.
4. *Modeling is not difficult to learn and do*
    - Molecular modeling is easy to do given currently available software (probably easier than taking an NMR spectrum). The difficulty lies in asking the right questions of the models and properly interpreting what comes out of them.

Hehre and Schusterman also offer some educational encouragements for incorporating molecular modeling into the education of current and future chemists:

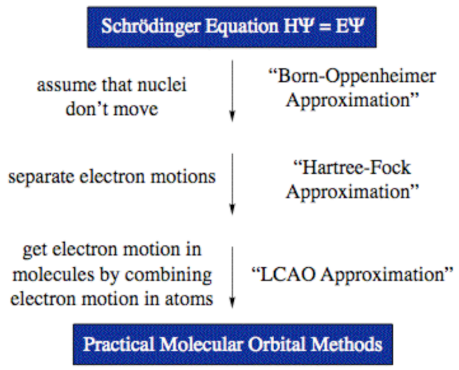
---

1. Models are what we teach. Students need to learn to "think like a molecule". To do this they need to "see" what a molecule sees and "feel" what a molecule feels. Models give us the best and most direct view of the molecular world.
2. Modeling is the best tool for learning about chemical theory. VSEPR, Lewis structures, Hückel MO are all crude attempts to convert good theories into chemical predictions. Modern computational methods give a much more accurate assessment of theoretical predictions.
3. Models are easy to use, inexpensive, safe. Modeling is a student-friendly educational too. It is not just for "experts".

And, since we're quoting these two scholars, here is their answer to the question: should molecular modeling replace experimental chemistry?

> **Of course not**!
> The goals of chemistry are not changed by molecular modeling. On a practical level we want to learn how to make things (synthesis) and how to figure out what things are made of (analysis). On an intellectual level we want to understand the "rules" that describe chemical behavior.
>
> Molecular modeling is, like NMR, a tool for achieving these goals. Since two of the goals - synthesis and analysis - are experimental, they cannot (and should not) be done away with. However, modeling does change the way we do syntheses and analysis. And, it speaks directly to the intellectual goals of chemistry.
>
> A modern chemical education still requires practical training in experimentation, but it requires training in modeling too.

This is perhaps a good time to remind the reader of some of the more theoretical concepts presented in the first several chapters of this *Guide*. In describing the practical aspects of molecular modeling, the reader is reminded that the "Holy Grail" of computing in chemistry is the *exact* solution of Schrödinger's equation. Given the state of the art of molecular modeling today, that's not attainable, and, as such, we have to make a number of approximations to be able to perform a calculation. We also need to remember that the answers we get are not quite "right", but they are correct enough to be useful. The graphic below looks to capture what we would *like* to be able to do, but also shows what we *have to* do to make a practical use of molecular modeling:



Getting Started:

How, then, does one get started? The answer is: the same way one gets started with any chemistry research tool! In chemistry education, we provide our students with laboratory experiences that allow them to learn how to use various tools one at a time. Then we often have them do labs where they have to use the right tool at the right time for the right reason. As we described above, hopefully they are learning some chemistry along the way, but many times the labs are designed to teach the students how to use the tool. Molecular modeling is no different. At the end of this *Guide*, there are a series of labs that are designed to provide students with the opportunity to learn how to use the tools of molecular modeling to solve some problem. As with most labs, we already know the answer to the problem, and it's the student's job to try to see how well they can replicate previously determined experimental results. Molecular modeling is no different in this regard than teaching a student to use basic lab equipment, or a UV-VIS spectrophotometer, or any other of the research tools found in modern chemistry labs.

Molecular modeling, however, has its own techniques and strategies, as well as its own peculiarities, quirks, and idiosyncrasies. The four-step process shown above is, as we stated, the basic procedure: define the problem, build the molecule, perform the calculations and analyze the results. In this next section, we elaborate on this four-step approach.

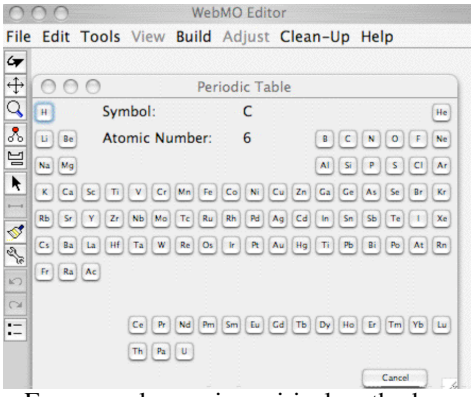
Define the problem: molecular modeling problems can roughly be divided into three categories:

1. Problems that are attempting to determine the structure of the molecule
2. Problems that are attempting to determine a property of the molecule
3. Problems that are attempting to determine the reactivity of a molecule

Each of these three types of problems requires the user to understand what calculations are appropriate, what methods or sequence of methods might be needed, and what a "right" answer might look like. For example, in a problem that wishes to determine the structure of a molecule, what calculation might be needed? In this example, we probably wish to perform a geometry optimization calculation, which tries to find the best configuration of bond lengths, angles and dihedrals that will give us a molecule with the lowest possible energy. What does a "right" answer look like? It looks like an energy value that has converged on the lowest possible value we can expect, given the method we have chosen to perform that calculation.

Or, suppose we are trying to learn something about how a given molecule might react with another chemical. We might perform a molecular orbital calculation. We pay particularly close attention to the characteristics of the *frontier orbitals*, defined as the *highest occupied molecular orbital* (HOMO) and the *lowest unoccupied molecular orbital* (LUMO). Those two molecular orbitals are often the most valuable quantities for determining what reaction might occur between the molecule of interest and other molecules. The point of these two examples is to suggest that the molecular modeler needs to be able to define the chemistry of the problem to be solved, and that problem in turn helps to determine a method to be used.

Build the molecule: this component sounds like the simplest of the four basic steps, and in many ways it is. The modeler must, of course, understand how to use the software to select an atom, add it to the workspace, connect it with one or more other atoms, and otherwise construct the molecule. Many modern software programs, including the WebMO interface used for the North Carolina computational chemistry server, come equipped with *fragment libraries*. A fragment library, as the name suggests, are parts of molecules or entire molecules, usually grouped by categories, such as ring structures, drugs, or polyatomic ions. Most packages, also including WebMO, come with a built-in periodic table, from which the user can select an element for addition to the molecule.



This is an option that is, however, loaded with potential problems! Not all methods are capable of performing calculations on all elements. For example, semi-empirical methods typically cannot deal with elements past xenon (Xe) on the periodic table. The builder will allow you to put those elements into a molecule, but you can count on a "failed job" message if you try to run a semi-empirical job on that molecule. As the saying goes: "Just because you can (include a specific element), doesn't mean you should!" In building the molecule, therefore, the user has to understand the limits of his or her method. Suppose, for example, one wishes to calculate the energy of a molecule containing an element from the second row (Na, Mg, Al, Si, etc.). The user chooses a Hartree-Fock *ab initio* method, with a 3-21G basis set. The user needs to understand that 3-21G does not do a particularly good job with second row elements for geometry optimizations *or* energy calculations. Adding polarization – 3-21G* -- to the choice of the basis set *does,* however, work relatively well for these second row elements. It is necessary for the user, therefore, to develop a sense of which methods can handle which elements from the periodic table option of the molecular builder.

Do the calculations: this step requires the user to set up the job: select the appropriate computational engine, choose the calculation to be performed, choose the method desired, choose (if appropriate) the basis set, and ensure that the computer knows the charge and multiplicity of the molecule. For most new users of molecular modeling, this is the most intimidating part of the process, followed (or preceded) closely by the *analysis* step, described below. This step also implies that the user knows what to do if the job fails, and how to read the output results file to find the error. We address this challenge in some detail in later chapters. Indeed, most of this *Guide* is designed to help the new molecular modeler understand the various parts of the "do the calculation" step.

One of the assumptions that is often implied in the computational chemistry research literature is that of *geometry optimization*. When a researcher reports, for example, that a particular molecule has a specific energy value, it is typically assumed that the molecule has been *optimized*, meaning that a calculation has been run to ensure that the molecule has the correct structure. In the WebMO interface, once the user has built the basic structure of the molecule, he or she typically does a **Comprehensive Cleanup** from the **Cleanup** menu. This action does several things. It automatically adds any missing hydrogens *and* then performs a rough optimization of the molecule. Any calculation that takes place after that point is done on the molecule in this roughly optimized form. Most molecular modelers will perform a geometry optimization calculation on the molecule using a more robust method, and *then* get to the business of running the calculation of interest. In some cases, the geometry optimization *is* the calculation of interest. In most cases, however, we're running molecular orbital or vibrational frequency or transition structure calculations on molecules that have been optimized first. On the Internet, there are resources such as the "Computational Chemistry Comparison and Benchmark Database" (http://srdata.nist.gov/cccbdb/) where one can download optimized geometries for common molecules. This resource saves the user the extra calculation of performing the optimization, but also assumes that the user has some confidence in the geometries available on this page. Note that this is a federal government resource, so you can make your own decision about whether you trust it or not! Joking aside, this resource is the National Institute of Standards and Technology (NIST), a very reputable scientific institution. Regardless of how you do it, however, most of your calculations are performed on optimized

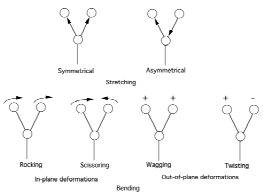
molecules. If your molecule is not optimized, you should either be doing that purposefully or you should be cautious about your results!

Analyze the results: like the *doing the calculation* part, this is often the most challenging aspect of the process, especially for the new modeler. The units for the quantities being calculated are not those often encountered in traditional laboratory chemistry, the graphics produced are challenging to understand, and, as the reader will see in Chapters 18-20 on the software tools, the output files can be overwhelming! Interface software programs such as WebMO significantly aid in the analysis process by filtering out a tremendous amount of the less useful results from a calculation. WebMO also packages the results in a nicely formatted **Calculated Quantities** web page, with well-labeled descriptions. Despite this, the results only make sense based on the user's understanding of the chemistry being studied. In the next section, we present a beginning description of some of these results.

Calculated Properties:

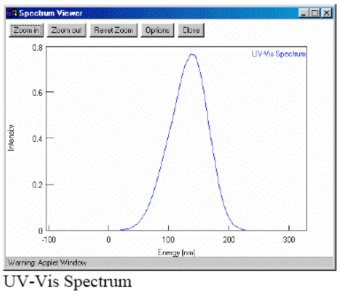
In molecular modeling, there are a variety of basic properties that can be calculated. In the remaining chapters in this section on Techniques, some of these techniques are described in more detail. The basis calculated properties are described here in brief:

1. *Single point energies (SPE)*: also known as the molecular energy, this property represents the energy of a molecule at a *specific geometry*. The term *single point* refers to the fact that the energy value only has meaning for the molecule at that *single* geometry state. Once a bond is shortened or lengthened, or a bond angle changes, the energy also changes, and a new SPE needs to be determined.
2. *Geometry optimization*: this property is not really a property *per se*, but sometimes this is the property we are trying to calculate. As discussed above, however, an optimized molecule is usually the starting point for the calculation. Recalling the "cooking chicken" analogy at the beginning of this *Guide*, a geometry optimization might be analogous to thawing a frozen chicken prior to cooking it. Most cooks assume this step without being told to do so! Likewise, most modelers optimize their molecules before doing a calculation.
3. *Molecular orbitals (MOs):* determining the properties of the MOs – specifically the energies of the individual MOs – is extremely useful in trying to predict other properties and/or the reactivity of a molecule. With the graphics capabilities of most modern software packages (including WebMO), we can visualize the *s* and *p* orbitals of atoms, lone pairs on atoms, and the various types of bonds connecting atoms together. From our calculations of MOs, we can determine bond orders and a variety of other *derived* properties. One of the challenges in doing molecular orbital calculations, however, is to remember that MOs are purely mathematical in nature, and don't exist in real atoms and molecules. Despite this fact, they are exceptionally useful models for predicting the nature and reactivities of molecules.
4. *Dipole and higher multipole moments*: dipole moments are useful in determining the polarity of a molecule. Water, for example, is a relatively strong dipole, and as such is also a relatively strong polar molecule. Polarity is useful in describing chemical behavior such as solubility. Polar compounds will dissolve other polar compounds, and non-polar compounds will dissolve non-polar compounds – "like dissolves like". Calculating dipole moments can also approximate boiling points. Multipole moments (quadropoles, for example) can be used to measure the effect of molecular charge distribution.
5. *Population analysis*: in a molecule, electrons are distributed throughout the molecule, and are not associated with a specific atom. It is useful, however, to *arbitrarily* assign electrons to specific atoms, giving those atoms a property known as a *partial charge*. This practice is known as a *population analysis*, the most common method being attributed to Robert Mulliken (1896-1986, 1966 Nobel Prize in Chemistry). Population analysis studies allow us to get a sense of the contributions of individual atoms to the structure of the molecule.
6. *Vibrational frequencies*: all molecules vibrate, and molecular modeling allows us to compute and visualize the different ways that molecules vibrate. As a general rule, a non-linear molecule has *3N-6* vibrations and a linear molecule has 3N-5 vibrations, where *N* is the number of



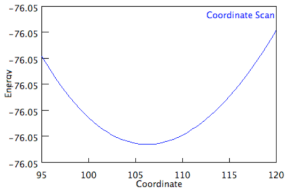
atoms. Water, for example, has 3 vibrational modes. As shown in the graphic, vibrations can be grouped as stretching, in-plane deformations, and out-of-plane deformations. A calculation of vibrational frequencies also results in the generation of an infrared (IR) spectra, considered to be a unique "fingerprint" of a molecule.

7. *Transition structures*: a transition structure is the point at which the energy of two reacting substances is at its highest. Determining the transition structure of a molecule allows chemists to predict the likelihood of a specific product being formed, and is a powerful tool for determining chemical reactivities. Determining transition structures is one of the more challenging problems facing the molecular modeler.
8. *UV-VIS spectra*: most molecular modeling software packages allow the user to create a UV-VIS spectra for a compound. UV-VIS spectra are a standard tool used in analytical chemistry, and is used to determine the properties of specific molecules. It is an especially useful tool to organic chemists who are interested in the structure of complex organic compounds.



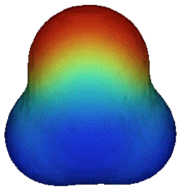
UV-Vis Spectrum

9. *Coordinate scans*: a coordinate scan is created by setting up a job whereby the computer does multiple calculations over a range of bond lengths or angles for a molecule. For example, we can perform a coordinate scan of water, where the H-O-H bond angle is scanned from 95º to 120º at 0.5º intervals. The resulting graph, shown here, tells us that the most stable structure of water is one that has a H-O-H bond angle of approximately 105º, relatively consistent with what we find in chemistry textbooks.



10. *NMR* (nuclear magnetic resonance): NMR is one of the techniques found in *spectroscopy*, an analytical chemistry technique that can be used to determine information about a molecule, such as its structure. As the name suggests, NMR takes advantage of the magnetic properties of nuclei, and can use that attribute to determine the structure of molecules.
11. *Electron Density*: the electron density, typically shown as a visualization (image), gives the user the best sense of the actual shape of the molecule. For example, the graphic here shows the electron density image for the formaldehyde (CH_2O) molecule, giving us a sense of the actual shape of the molecule.
12. *Electrostatic Potential (ESP):* the electrostatic potential gives the user a sense of how the molecule will react with other molecules. Typically, an electrostatic potential is *mapped* onto an electron density image, as shown in the graphic. The color scheme reflects the electrostatic potential, with red being negative and blue being positive. If we have a positively charged object, and move it around the molecule, the electrostatic potential is a measure of the force the positive charge "feels". At the red end of the molecule, the positive charge feels an attractive force, given that the red represents a negative potential. At the blue end of the molecule, the positive charge feels a repulsive force, as a result of the positive electrostatic potential. Electrostatic potential determinations are very useful in describing the interaction of molecules with other molecules.





<u>Fundamental Units</u>:

One of the other challenges facing the novice molecular modeler is that the units of measure used by molecular models are unfamiliar. They are, however, no more difficult than any other unit of measure. It is a simple matter to convert most of the units found in molecular modeling to more familiar units. Most modelers, however, become comfortable with the standard units as they improve their modeling skills. Some example units are described below:

Unit	Conversion
hartree: the basic unit of energy, equal to twice the energy of a ground state hydrogen atom	627.5 kcal $mole^{-1}$
bohr: atomic unit of length (a_0), equal to the radius of the first Bohr orbit for a hydrogen atom	0.529 Å (angstroms)
debye: unit of dipole moment	3.336 10^{-30} Coulomb · meter (C · m)

Verification and Validation:

One of the challenges facing the molecular modeler, and any other use of computational techniques, is that of *validation* and *verification*. The goal of any computational approach is finding the "right" answer. The term "right" is relative, given that we almost always need to use approximations in our model to be able to perform the calculation. It is also the case that any calculation is subject to *round-off error*, small errors that occur in the computing process, which can accumulate for large calculations. Even though all computational calculations are technically "wrong", it is absolutely the case that they are still very useful in understanding the behavior of a physical system. This is certainly the case in computational chemistry, especially as computers become more powerful and our mathematics and software tools become more precise.

Validation and verification helps us to answer two questions:

1. Validation: am I building the right model? This question basically addresses the chemistry of the model. It helps to answer questions such as: do I understand the chemistry I am trying to study? Am I asking the right question? Am I calculating the right property? Is the model I am building valid?
2. Verification: am I building the model right? This question addresses the use of the computing tools. It helps to answer questions such as: am I using the software correctly? Have I built the molecule correctly? Do I understand how the software performs its calculations? Is the software working correctly?

As the novice modeler develops more experience with the technologies, techniques, and tools, so to will the users ability to validate and verify his or her computational results improve.

Chapter 11: Single Point Energies (SPE) and Geometry Optimizations

Key Notes:

Structure-Properties-Activities Revisited:

The overriding purpose of molecular modeling, and perhaps even all of chemistry, is to determine three characteristics of a molecule: its structure, its basic properties, and its reactivity with other molecules. Molecular modeling techniques provide the researcher with an efficient and effective way to determine a molecules structure, properties, and/or activities, in a way that is often complementary to experimental techniques. In many cases, molecular modeling can be used to characterize a molecule that might not be synthesized in the laboratory for a variety of reasons (expense, safety, etc.).

Molecular Geometries:

All molecules possess a geometry, characterized by the number and type of atoms, the number and type of bonds, the bond lengths, bond angles, and bond dihedrals. In molecular modeling, the molecular geometry is the starting point of a computational calculation. Modern software packages typically provide the user with a molecule builder, which automatically determines reasonable bond lengths and angles, based on the atoms and bonds found in the molecule. Molecular geometries can be determined by building the molecule using one or more software programs, or can be found online through files such as PDB files or through online databases.

Single Point Energies:

A single point energy (SPE) is a basic molecular modeling calculation. This calculation determines the energy of a molecule *at a specific molecular geometry*. Single point energies, sometimes known simply as *molecular energies*, are typically in units of Hartrees, which can be converted to more common energy terms such as kilojoules mol^{-1} ($kJ\ mol^{-1}$), kilocalories mol^{-1} ($kcal\ mol^{-1}$), or electron-volts (eV). Any change in a molecular geometry will require that a new single point energy calculation be performed.

Potential Energy Scans (PES):

A *potential energy scan (PES)* is a type of molecular modeling calculation that allows the user to find the lowest energy value for a set of different molecular geometries. Typically, the goal is to find the influence of *one variable*, such as a bond angle, on the molecular energy of a molecule. In performing a PES calculation, the end result is typically a graph showing the change in energy of the molecule as a function in the change of the geometry variable.

Geometry Optimizations:

Prior to performing most molecular modeling calculations, the molecule needs to be *optimized* using a *geometry optimization* calculation, run at a particular *model chemistry*. Optimizing a molecule results in that best combination of bond lengths, angles and dihedrals that results in the molecule having the lowest, and therefore most stable, energy. Some molecules are difficult to optimize, and may have to be optimized several times using progressively more robust mathematical methods. In some cases, the optimization is the focus of the calculation, but in most instances the modeler optimizes the molecule(s) prior to beginning the calculations of interest.

Structure-Properties-Activities Revisited:

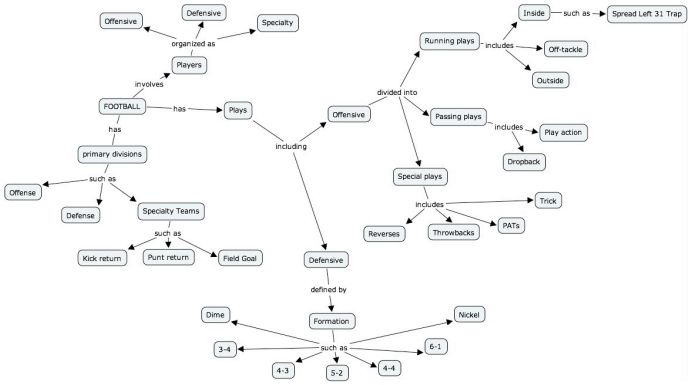
Chapter 2, "Introduction to Molecular Modeling" stated that molecular modeling allows the user to determine three fundamental items of interest of a molecule or system of molecules:

- The structure, or geometry, of a molecule, answering the question "what does the molecule look like?"
- The property or properties of a molecule or system of molecules, answering the question "what characteristics have by itself?"

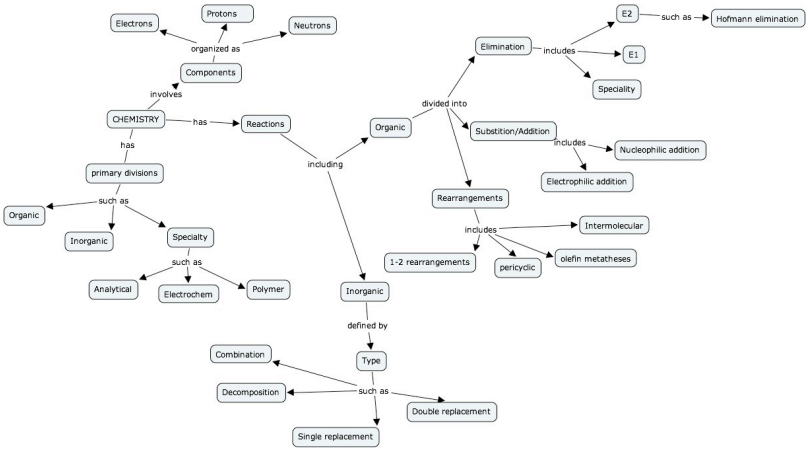
- The activity or activities, also known as *reactivities*, of a molecule or system of molecules, answering the question "how does the molecule behave in the presence of other molecules?"

These characteristics were mentioned again in the previous chapter, "Basic Molecular Modeling". Section II of this *Guide* looks to explore in more detail this model of structure-properties-activities. The reader is advised to be careful about trying to use the structure-properties-activities model too literally. Some of the quantities that can be computed might be considered by some to be "properties", while others might consider them to be more related to reactivity. For example, in this and following chapters, a discussion of molecular orbitals (MOs), particularly *HOMOs* and *LUMO*s, will be presented. While the MOs of a molecule are clearly something that the molecule always has, they also have a role to play in determining reactivity.

Perhaps there is another way to present what chemists do and what chemistry is about. Hopefully most of the readers of this *Guide* have had some exposure to (American) football. You should know that football has players, and that there are two sides, or primary divisions: the offense and defense. You might know that there are also specialty teams, such as the kickoff team and the punt return team. If you are a little more knowledgeable, you might know that there are set plays. The offense has its plays, as does the defense. Focusing on the offense, you might know that there are running plays and passing plays. For the offensive running plays, you might know that there are inside, outside, and "off-tackle" plays. If you *played* or *coached* football, you might know that a "spread left 31 trap" play is an offensive running play to the inside, with the "3-back (the tailback)" running to the left through the "1-hole (the space between the center and the left offensive lineman)". Likewise, the defense has different formations, like a "nickel" formation. This discussion is captured in the concept map shown below (and by no means captures all there is to know about football!):



We can create a similar map for the field of chemistry. Just as the football map doesn't capture all of football, neither does this map capture all of chemistry. Regardless, we see that just as football has players, so too does chemistry have components (electrons, protons, neutrons). Chemistry has primary divisions, such as organic and inorganic chemistry, and "specialty" areas, such as analytical, electrochemistry, and polymer chemistry. Chemistry doesn't have plays, it has *reactions*, and these can be categorized in numerous ways. For example, a *Hoffman elimination* reaction belongs to the organic family of "plays" is one of several elimination reactions, specifically of the *E2* type. Using the concept map below, chemistry can be organized and categorized:



Many of the items on this map (the chemistry one, not the football one!) represent the "stuff" that molecular modeling can help to explore and elucidate. For example, in organic chemistry, we can use molecular modeling to calculate different type of spectra, such as infrared (IR), ultraviolet/visible (UV-VIS), and nuclear magnetic resonance (NMR). Using frontier molecular orbital (FMO) theory, the modeler can study elimination reactions. Coordination complexes in inorganic chemistry can also be modeled, looking to see if we can create mathematically stable coordination complexes, either in support of laboratory work or as a predictive tool.

In this and the following five chapters, this *Guide* presents information on a number of these calculations, with some guidance on how to generate them. All of the calculations presented in this Section are included in one or more of the lab activities found in the Appendix, and readers are encouraged to use these labs for a deeper understanding of the techniques and tools of molecular modeling.

Molecular Geometries:

One of the most basic and important of concepts in all of chemistry, including molecular modeling, is that of a *molecular geometry*. Quite simply, this defines and is defined by the structure of the molecule. In Chapter 3 ("A Computational Analogy"), the components of the molecular geometry were presented in detail, and are listed here by way of review:

1. Number and types of atoms
2. Number and types of bonds
3. Relevant bond lengths (in units of Angstroms, or Å)
4. Relevant bond angles (in units of degrees)
5. Relevant dihedral angles (an angle between four atoms, which signifies the 3-dimensional shape of the molecule).

The reader is encouraged to review these components if necessary. In Chapter 2, the reader was also introduced to the concept of a *Z-matrix*, a mostly historical way of describing molecular geometries. As discussed previously, most modern computational chemistry packages come equipped with a *molecular editor*, a program that allows the user to build the molecule graphically rather than numerically via a text file.

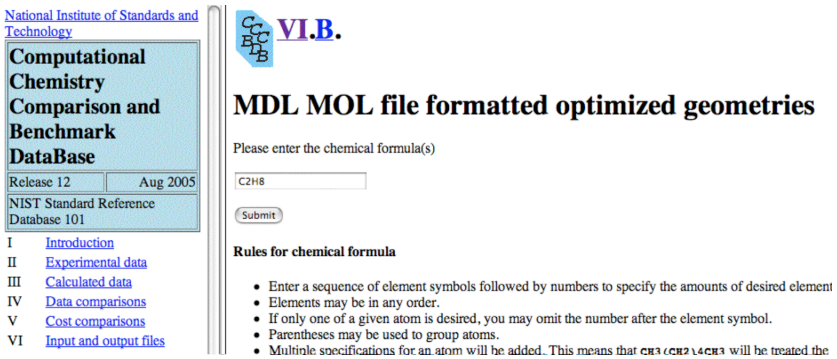
In molecular modeling, one of the most important (and typically one of the first) tasks to be accomplished by the modeler is that of determining the most stable geometry for a molecule. In the experimental world, molecular structures are often determined by techniques such as *x-ray crystallography*, and the experimental molecular geometries for many molecules have been determined and are available through a variety of means. For example, there is a type of computer file known as a "PDB" file, which stands for "Protein Data Bank". A sample PDB file for aspirin would be called "aspirin.pdb". These files can be located using a search engine such as Google. For example, if one knows the name of the molecule, a search on "moleculename.pdb" will often turn up a hit.

Alternatively, one can search Google using the notation "moleculename filetype:pdb". For example, if looking for the molecule benzoic acid, one could search for "benzoicacid.pdb" or "benzoic acid filetype:pdb". In this particular search, a text file was located, and its structure is shown below:

```
COMPND     BENZOIC ACID
AUTHOR     DAVE WOODCOCK  96 07 05
ATOM      1  C         1      1.169  -0.193   0.639  1.00  0.00
ATOM      2  C         1      1.215   1.205   0.782  1.00  0.00
ATOM      3  C         1      0.047   1.967   0.624  1.00  0.00
ATOM      4  C         1     -1.165   1.328   0.325  1.00  0.00
ATOM      5  C         1     -1.213  -0.070   0.183  1.00  0.00
ATOM      6  C         1     -0.043  -0.857   0.336  1.00  0.00
ATOM      7  C         1     -0.043  -2.265   0.198  1.00  0.00
ATOM      8  O         1      1.131  -2.961   0.368  1.00  0.00
ATOM      9  O         1     -1.240  -2.943  -0.112  1.00  0.00
ATOM     10  H         1      2.068  -0.778   0.762  1.00  0.00
ATOM     11  H         1      2.149   1.694   1.014  1.00  0.00
ATOM     12  H         1      0.085   3.041   0.732  1.00  0.00
ATOM     13  H         1     -2.067   1.910   0.203  1.00  0.00
ATOM     14  H         1     -2.149  -0.558  -0.045  1.00  0.00
ATOM     15  H         1     -1.121  -3.476  -1.044  1.00  0.00
TER      16         1
END
```

This structure can be viewed using a variety of free PDB viewers. PDB files can also be imported into the WebMO Molecular Editor, using the **Import Molecule** button, then selecting PDB as the file type.

Molecular geometries that have been computed can also be found on various databases, such as the NIST Computational Chemistry database (http://srdata.nist.gov/cccbdb/). The screenshot below shows a sample search for the ethane (C_2H_8) molecule. This particular database, as is true of most of the online resources, provides a limited number of optimized molecules for downloading and use in a computational chemistry software package.



Regardless of the source, how do we know that the molecular geometry is "right"? A correct molecular geometry is one that has the lowest energy value, and that is the subject of the next section.

Single Point Energies:

Probably the most fundamental of all molecular modeling calculations is the *single point energy (SPE)*, often known as a *molecular energy*. An SPE is the energy of a molecule *at a specific geometry*. For example, if one calculates the molecular energy of the benzoic acid molecule above, the calculation says that the energy is -415.924344571 Hartrees. A *Hartree* is a unit of energy used in computational chemistry, and is easily converted to other, more familiar units, using the conversion table below. This table will appear in a number of chapters in this *Guide*:

1 unit =	hartree	kJ per mole	kcal per mole	eV
hartree	1	2625.5	627.51	27.212
kilojoules (kJ) per mole	3.8088×10^{-4}	1	0.23901	1.0364×10^{-2}
Kilocalories (kcal) per mole	1.5936×10^{-3}	4.1840	1	4.3363×10^{-2}
Electron-volts (eV)	3.6749×10^{-2}	96.485	23.061	1

The most critical concept of the molecular energy calculation is this: if the modeler changes *the position of any atom*, then the molecular energy also changes. The term *single point energy*, which again is synonymous with *molecular energy*, says that the energy value is valid only at a *single point of the geometry*. Change something, and you have to compute a new energy value. Change a bond angle, bond length, dihedral angle, and the previously determined energy is no longer valid.

Given the discussion above, the critical question now becomes: *what combination of bond lengths, angles, and dihedral angles describes a geometry that will have the lowest energy value?* Molecules do a really good job in configuring themselves to find the lowest energy. The lower the energy, the more stable the molecule is. Chemists will often state that the molecule is in an *energetically favored state*, meaning that it has that combination of bond lengths, angles, and dihedrals – its molecular geometry – that is lower than any other combination. Since molecules like to be that way, one of the tasks of the molecular modeler is to try to determine what that combination is. There are a variety of techniques to do that, including performing a *potential energy scan (PES)* calculation.

Potential Energy Scans (PES):

A *potential energy scan (PES)* is a type of molecular modeling calculation that allows the user to find the lowest energy value for a set of different molecular geometries. The result is typically a chart showing the energy of a molecule (dependent variable on the y-axis) as a function of the coordinates of a specific variable (independent variable on the x-axis). In practice, it is typical to isolate *one* variable, such as one bond angle, as the target of interest.

For example, a simple PES calculation involves determining the bond angle of water. This angle is reported in most textbooks as being close to 105.0°, and most students take this as gospel. Is this angle, however, the most energetically favored? In other words, is this the *optimal* bond angle, the one that allows the molecule to have the lowest energy value? Another way of saying this: at the 105.0° bond angle (all of the other coordinates being constant), has the molecule been *optimized?*

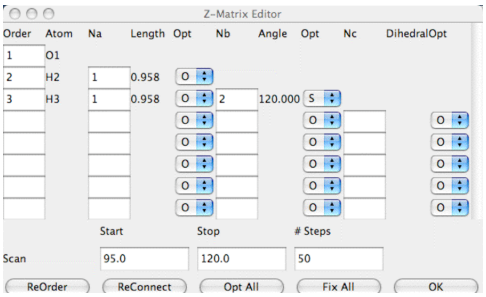
An example is helpful at this point. What is the optimal (optimized) bond angle of water? Is it *really* 105.0°? The answer partly lies with the type of mathematics used to determine this, but at this point the larger concern is the technique. A **coordinate scan** is the type of computation that is appropriate for this task. In this case, the task is to determine which bond angle provides the lowest energy value.

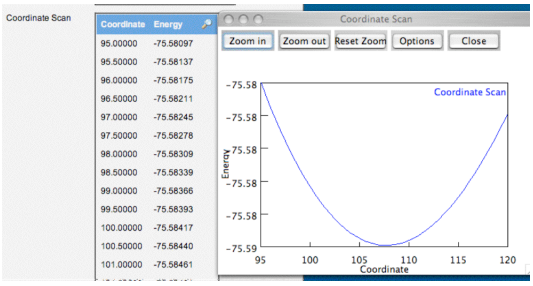
In this experiment, the bond angle range will be from 95.0° to 120°, in increments of 5°. Given that, the coordinate scan calculation will perform 50 molecular energy calculations on the molecule (a relatively lengthy job run). In the WebMO program, the procedure looks like this:

1. In the molecular editor, deposit an oxygen atom (red ball).
2. Perform a **Comprehensive Cleanup** under the **Cleanup** menu.
3. Under **Tools**, go to **Z-matrix.**
4. The goal is to **scan (S)** the bond angle, so the option **S** is chosen next to the bond angle. Then the starting angle, ending angle, and number of steps are entered into the appropriate boxes.
5. Once the molecule has been configured and a computational engine has been selected, the calculation to be chosen (in Gaussian, by the way) is **coordinate scan**. The run now begins, and for water the run requires approximately 20 minutes. For educators, this is not a calculation that can be done by a full class of

students, but can be done as a demonstration and/or by students conducting computational chemistry research.

Once the run is completed, the results show a table of bond angles and energies. The code also automatically generates a graph of the results. In this case, the graph suggests a lowest energy value around 107.0c, higher than the textbook-reported 105.0°. This is due to the mathematical method chosen for this run. The point, however, is that the data clearly shows that the optimal bond angle is somewhere between 100° and 110°. Better mathematics will produce more accurate results.





One of the key points to remember about a PES calculation is that it is typically focused on *one* variable, typically a specific bond angle or dihedral angle. In the next section on *geometry optimizations*, the issue of best geometry is presented.

<u>Geometry Optimizations</u>:

Needless to say, it is inefficient to perform a PES calculation routinely, and most molecules are of such size that it is impractical. It is also the case that *other calculations* are best done on molecules that have the most optimized (lowest energy) structures possible. If the molecule can be found whose geometry has been determined by experimental means, and the modeler is confident of those results, this structure should be used for molecular modeling calculations. The bottom line: computational results make the most sense and are the most "accurate" if the molecule has the most optimal molecular geometry.

It is typically the case, however, that the modeler does *not* have an experimentally determined molecular structure from which to start. As such, one of the first chores that must be performed by the modeler is a *geometry optimization*. As the name suggests, this calculation looks to mathematically determine the best combination of bond lengths, angles, and dihedrals that produces the lowest energy. The quality of the optimization depends upon a number of factors, *primarily the level of theory and the basis set chosen.* A more robust *model chemistry* will produce a lower energy level, i.e., a more optimized geometry. As a reminder from Chapter 2 ("A Computational Analogy") the model chemistry is the combination of the level of theory (Hartree-Fock, Moller-Plesset, B3LYP, etc.) *and* the basis set (3-21G, 6-31G(p,d), etc.) that the user chooses. Molecular modelers use the notation shown below to show the model chemistry being used for the calculation, separated by "//", which is followed by the model chemistry used to optimize the molecule. The "correctness" of the optimization depends, again, on the robustness of the model chemistry used to perform that optimization. In the example below, the optimization was performed with a Hartree-Fock method using the 3-21G basis set. This is fairly basic model chemistry, and as such the geometry of the molecule is not of the highest accuracy. It might, however, be adequate for whatever other calculation this particular modeler is looking to perform:

B3LYP/6-31G(p)//HF/3-21G

In WebMO, as well as with many other computational chemistry software packages, a very rough geometry optimization is performed automatically during the building stage. In WebMO, once the user builds the basic

molecule, a **Comprehensive Cleanup** will add any required hydrogens and adjust the bond lengths and angles to approximate some level of accuracy. Performing a **Comprehensive Cleanup-Mechanics** does the same thing, but uses molecular mechanics mathematics to do the optimization. It is probably a misnomer to call the cleanup an optimization, but the user will often notice that the molecule looks more "correct" after a cleanup has been performed.

In performing a geometry optimization, it might be necessary and/or useful to perform more than one optimization calculation on a molecule. For example, if a molecule is particularly large or has a complicated structure, it may not be possible to optimize the molecule – the user gets a "failed" message from WebMO. One option here is to try to run the optimization several times, using different theories. For example, the first optimization might be run using a molecular mechanics program such as Tinker. That molecule is then optimized again using a semi-empirical technique, such as GAMESS using the PM3 algorithm. That molecule is then optimized using a low level of *ab initio* mathematics, such as HF/STO-3G. Finally, if the molecule has survived these optimizations, it can be optimized using the model chemistry desired for the final solution.

Most codes, such as Gaussian, have *keywords* that are used to help with stubborn optimizations. For example, in the **Advanced** tab of the job manager, the user can use the **MaxCycle**=*N* keyword, where *N* is some number of cycles, typically set at 20. This says that the code will try to run the optimization 20 times, and will fail if it has not reached some level of accuracy. One could set the number higher, thus requesting that the code try harder. This has the effect, of course, of increasing the compute time of your optimization.

In the computational chemistry literature, the majority of the calculations that are performed and reported are done on molecules that have been optimized using some model chemistry. A journal article will almost always report the mathematics used to optimize the molecule(s). When evaluating the quality of the researchers calculations, it is up to the user to decide if the molecule(s) have been optimized to a sufficient level. For the modeler performing the optimization, the main decision is typically one of how much time and resources (particularly valuable compute cycles!) might be devoted to the optimization process. Especially for students on shared systems such as the North Carolina High School Computational Chemistry server, it is probably not expedient to expend significant amounts of computing time to ensuring a quality optimization, unless the calculation is being done for student research purposes. For most lab-type activities, optimizations at a fairly low level of theory (HF/3-21G, for example) are probably adequate.

Chapter 12: Calculating Molecular Properties

Key Notes:

Atomic and Molecular Orbitals

The starting point for understanding molecular properties is the atom. Atoms have two important aspects: the number of electrons and the spatial distribution of these electrons. The number of electrons equals the atomic number of an element for a neutral atom. Solutions of the Schrödinger equation for the hydrogen atom give us the familiar *atomic orbital* shapes which define the electronic spatial distribution. As atoms are combined to make molecules, the wave functions that define the atomic orbitals in the atoms can be mathematically combined (added) to produce new wave functions that describe the wave properties of the electrons in a molecule. These new wave functions are called *molecular orbitals* and result from a linear combination of atomic orbitals (LCAO). This process allows us to calculate the spatial distribution of electrons in a molecule, which, in turn, allows us to calculate various important properties of a molecule.

HOMO/LUMO

These acronyms stand for the *highest occupied molecular orbital* (HOMO), and *lowest unoccupied molecular orbital* (LUMO). The HOMO is the molecular orbital of highest energy that is occupied by electrons. The LUMO is the molecular orbital of lowest energy that is not occupied by electrons. The HOMO and LUMO are important in determining such properties as molecular reactivity and the ability of a molecule to absorb light.

Electron Density

The square of the wave function at a point (r) is interpreted as the probability of finding an electron at that point. If a sum is taken over all occupied molecular orbitals for all space surrounding the nuclei in a molecule, the *electron density* for that molecule can be found. This function indicates the most probable location of electrons in a molecule and is useful for understanding bonding and reactivity. The electron density can also provide an indication of molecular size, or volume.

Electrostatic Potential

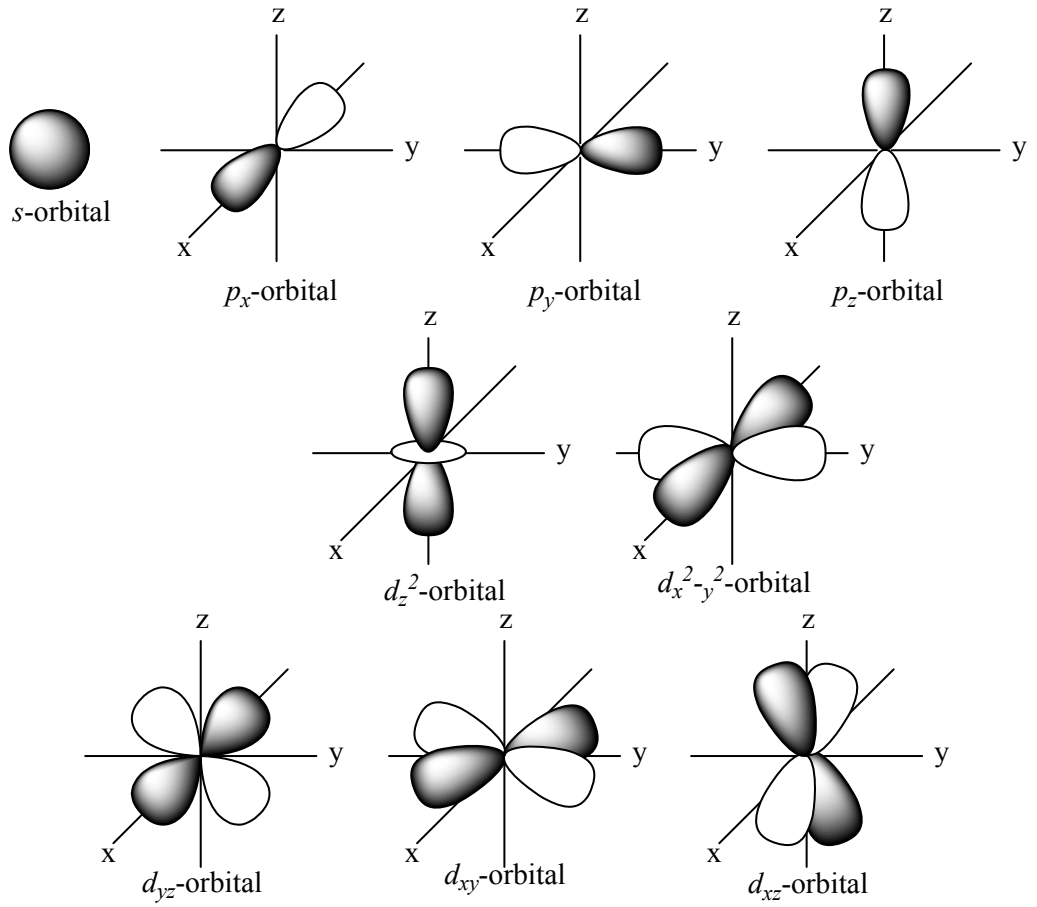
With the electron density and the location of the nuclei known, the *electrostatic potential surface* of a molecule can be calculated. This surface represents the distance from a molecule at which a positive test charge experiences a certain amount of attraction or repulsion. The electrostatic potential can be used to rationalize intermolecular interactions between polar species, define regions of local negative and positive potential in a molecule, and assist in predicting the path charged reactants will take as they approach the molecule.

Reactivity Prediction

A number of approaches can be used to determine the sites of reactivity in a molecule. The choice of approach will depend on the size and type of molecule and the type of reaction. Atomic partial charges are useful in reactions that are charge controlled (e.g. protonations/deprotonations). Electrostatic potentials can be used where polar reagents are involved. The largest lobes of the HOMO and LUMO work best for reactivity prediction when these orbitals are well-separated in energy from the other molecular orbitals. With larger molecules the energy differences between the HOMO or LUMO and the other molecular orbitals becomes smaller. The molecular orbitals close in energy to the HOMO or LUMO help to determine the reactivity, and various reactivity indices (electrophilic, nucleophilic, or radical susceptibilities, depending on the molecule) can be calculated in these cases.

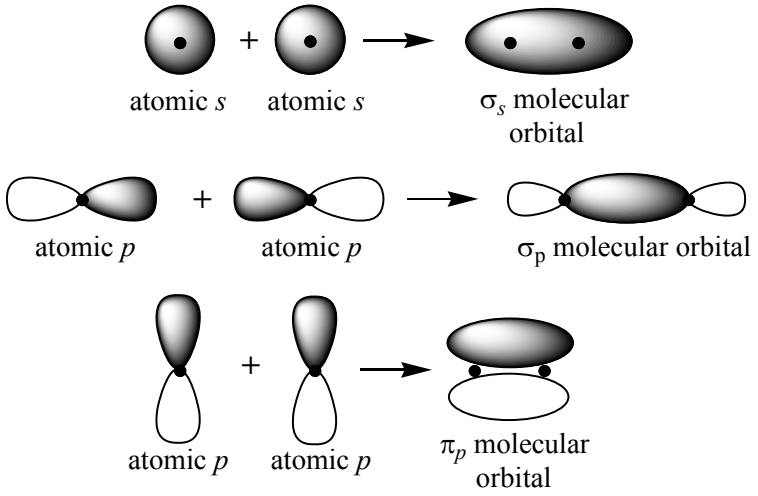
Atomic and Molecular Orbitals

Computational chemistry allows the calculation of a number of interesting molecular properties. In Chapter 5, the Schrödinger equation was introduced. This equation can only be solved exactly for the hydrogen atom. The solutions give various wave functions (Ψ), the square of which (Ψ^2) represents the probability of finding an electron at a certain point in space around the nucleus. The region of space where an electron has a high probability of being found is called an atomic orbital. The square of the different wave functions describe the familiar atomic orbital shapes, some of which are shown below:

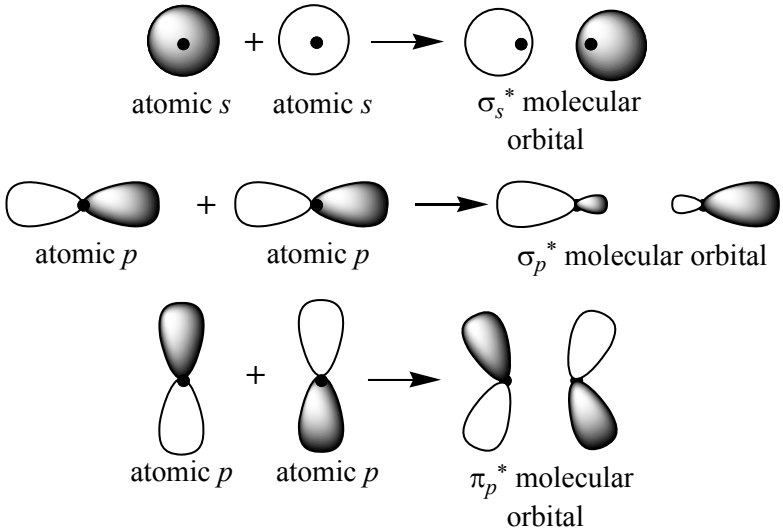


The shading of the orbitals (white or gray) represents the sign of the original wave function. Just as waves on the surface of the ocean can have a positive or negative sign, wave functions that describe electron behavior in atoms also have different phases.

Molecular orbital theory describes the formation of molecules in terms of overlap of the constituent atomic orbitals. As atoms come together and form molecules, their atomic orbitals overlap to form molecular orbitals. This is called a linear combination of atomic orbitals (LCAO). Atomic orbitals can overlap in three different ways. *Positive overlap* occurs when both orbitals have the same sign. Since the wave functions are in phase, they combine together (constructive interference). Bonding molecular orbitals, with electron density located between the two nuclei, are formed. Several possibilities are shown below in a pictorial representation of the process:

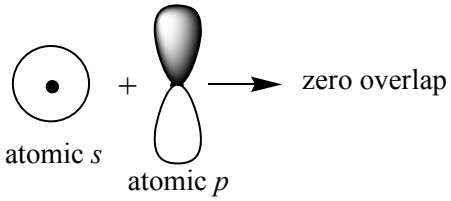


Negative overlap occurs when the two atomic orbitals have different signs. In this case, the wave functions are out of phase and they cancel one another out (destructive interference). Antibonding molecular orbitals are formed, and are denoted using the (*) superscript. Antibonding molecular orbitals have no electron density between the nuclei. Several possibilities and shown here:



Zero overlap occurs when there are equal regions of positive and negative overlap. In this case

there is no *net* bonding. One possibility of zero overlap is show below:



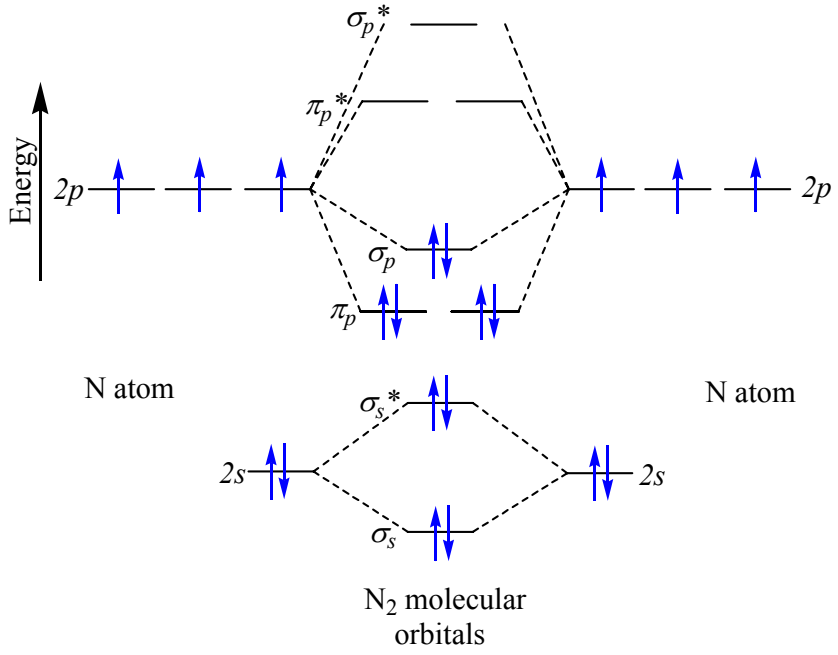
Notice that when positive or negative overlap occurs, *two* atomic orbitals can combine in *two* different ways to produce *two* molecular orbitals, one of which is bonding while the other is antibonding. Bonding molecular orbitals are lower in energy than the atomic orbitals they are formed from, while antibonding molecular orbitals are higher in energy than the atomic orbitals they are formed from.

If we have two atoms forming a bond, the above pictures can be used to understand molecular orbital formation. With a large number of atoms forming a more complex molecule, the ability to use a pictorial representation is limited. Computationally, the atomic orbitals are represented by basis functions (Chapter 4) which are mathematically combined (linear combination) to form the molecular orbitals. Computational software packages can then show us a 3D representation of the resulting molecular orbitals.

Molecular orbitals, just like atomic orbitals, are filled with two electrons each. The order of filling molecular orbitals begins with the lowest energy orbital and continues until all electrons are accounted for. This gives the ground state electron configuration of the molecule.

HOMO/LUMO

To further illustrate the process of molecular orbital formation, let's construct the molecular orbitals for the dinitrogen molecule, N_2. A nitrogen atom has a ground state electron configuration of $1s^2 2s^2 2p^3$. We will focus on the valence, or outermost, atomic orbitals (*2s* and *2p*) that will be involved in bonding. With five valence electrons per nitrogen atom, we will have a total of ten valence electrons in the N_2 molecule. In the molecular orbital diagram shown below, we start with a nitrogen atom on the left, and another on the right. As the atoms are brought closer together, the atomic orbitals overlap to form the molecular orbitals, depicted in the center. (See the above drawings for the molecular orbital shapes). The orbital energy increases in the vertical direction. Electrons (spin up or down) are shown in blue:



N_2 molecular orbitals

For the N_2 molecule, the σ_p molecular orbital is the *highest occupied molecular orbital* (HOMO), while the π_p* orbital is the *lowest unoccupied molecular orbital* (LUMO).

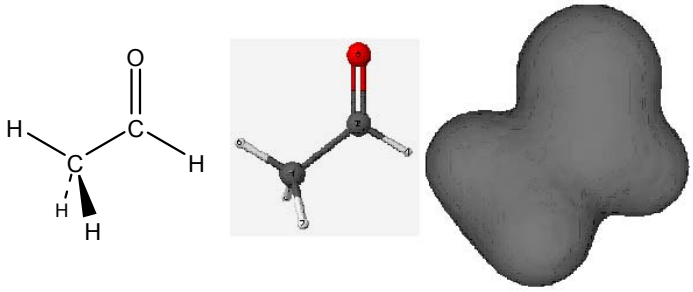
Electron Density

The wave function (Ψ) describes the behavior of an electron in a molecule. The square of the wave function (Ψ^2) at some point (r) is interpreted as the probability of finding an electron at that point. The *total* electron density at some point (r) involves summing up the probabilities for all occupied orbitals. For a molecule with n electrons and $n/2$ occupied orbitals, the electron density at some point (r) can be mathematically expressed as:

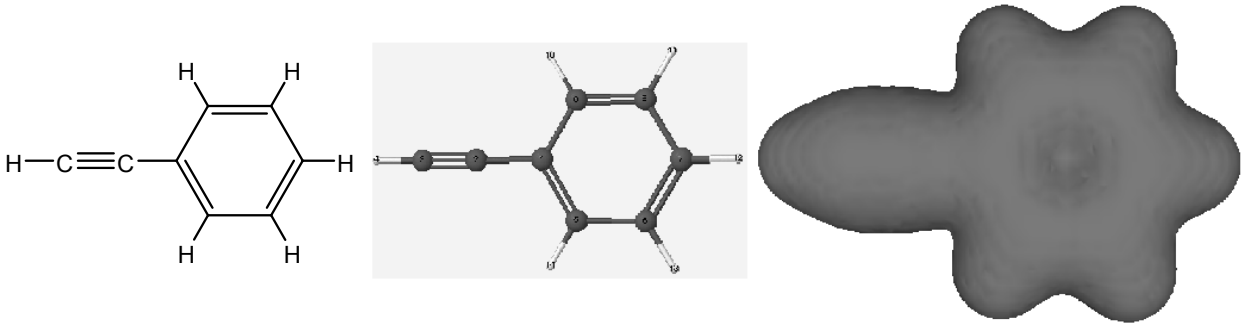
$$\rho(r) = 2\sum_{i=1}^{n/2} |\psi_i(r)|^2$$

Computational chemistry software packages will check the result of an electron density calculation by summing up the electron density function over all space. The result of this summation should equal the number of electrons. This number is reported in the output file.

The electron density provides an indication of molecular volume or size. An electron probability density of $0.01e^-/Å^3$ is similar to the space filling models that many programs provide, as shown below for the acetaldehyde molecule. (Note: The electron density and electrostatic potential pictures below are from B3LYP/3-21G level calculations).



The electron density also provides information on the location of the electrons in a molecule. While chemists often draw aromatic rings with localized double bonds, the π-electrons are actually delocalized through the ring. Phenylacetylene, pictured below, demonstrates this:



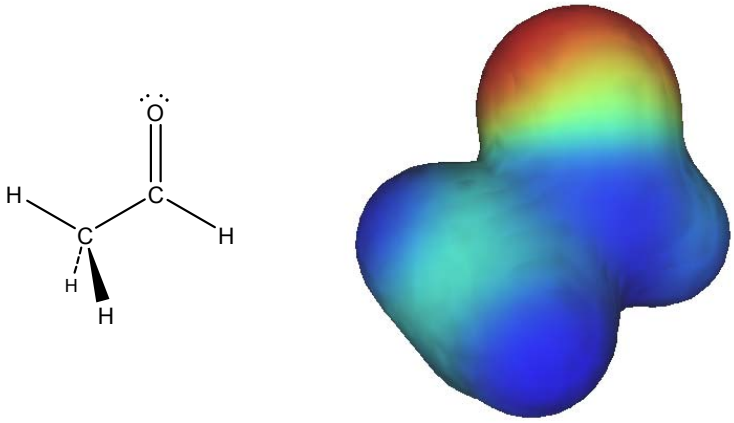
Electrostatic Potential

Imagine having two strong bar magnets, one of which is wrapped in enough cloth to mask its orientation. Using the unwrapped magnet, you would be able to hold it close to the wrapped one and feel the attractive and repulsive forces. If your unwrapped magnet had the poles labeled "N" and/or "S", you could do some experiments and determine the orientation of the wrapped magnet without having ever seen it.

Now, imagine that you have a proton (H^+) in one hand and a polar molecule in the other (you have small hands!). The molecule is "wrapped up" so you cannot see where the atoms lie. As with the magnets, you could use the proton to probe regions of electrical attraction and repulsion. Since opposite charges attract, portions of the molecule with more electron density (partially negative) will pull on the proton, while those areas with less electron density (partially positive) would repel the proton. With careful experimentation, you would be able to learn something about the charge distribution of the hidden molecule. You could also measure the strength of the attraction or repulsion for the area surrounding a molecule and produce a map of your results.

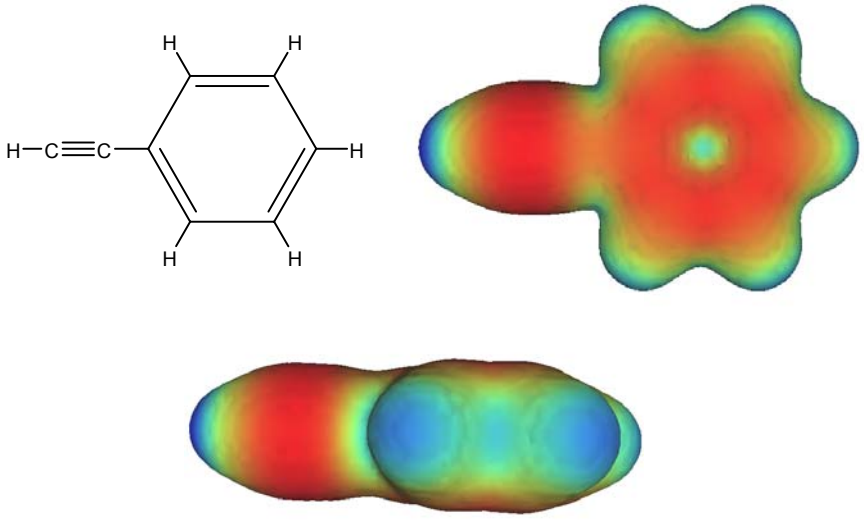
In essence, this is the procedure used to computationally determine the electrostatic potential of a molecule. Instead of using a proton, the software uses a positive point charge (small size). Typically, the amount of attraction or repulsion is determined at the electron density surface described above in a point-by-point manner. The test charge is interacting with *both* the negatively charged electron cloud *and* the positively charged atomic nuclei. The amount of attraction or repulsion the test charge experiences will depend on the types of atoms present

(nuclei), the sort of bonding between atoms (electron density), and presence of lone pairs (another form of electron density). The calculations sum up the relative amounts of electron density and the positive charges from nearby nuclei at a large number of points, and the results are color coded. The color red represents an area of high electron density (partially negative charge). At the other extreme, blue represents an area of low electron density (partially positive charge). The electrostatic potential of acetaldehyde, mapped onto the electron density, is shown below:



The top portion of the oxygen atom appears red do to the presence of two lone pairs of electrons. Oxygen is also more electronegative than carbon. The areas where the hydrogen atoms "stick out" appear blue since there is little electron density.

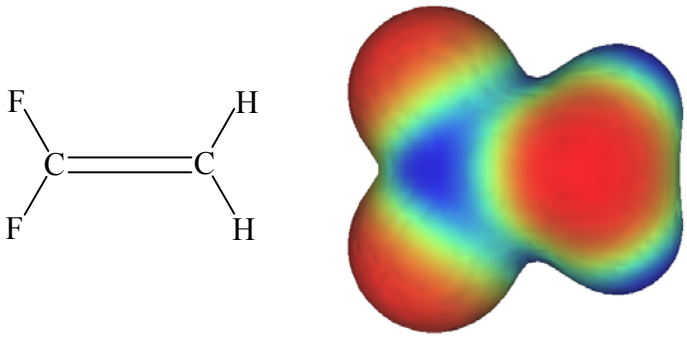
Another example, phenylacetylene, is shown below:



The aromatic ring, and (especially) the triple bond, both show a negative potential as is expected.

The last example is 1,1-difluroethylene (also called 1,1-difluoroethene). The electrostatic potential is interesting for this molecule because of the very different results for the two carbon

atoms. As seen below, the carbon on the left has a more positive potential due to the influence of the highly electronegative fluorine atoms attached to it. The carbon on the right has a relative negative potential.



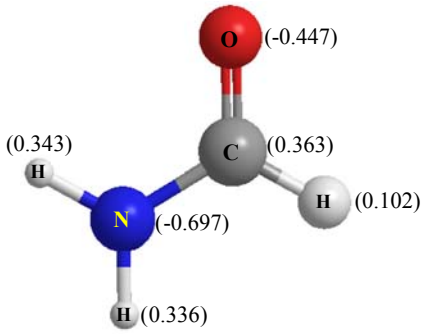
The ability to calculate and view the electrostatic potential can help chemists understand and predict some types of chemical reactivity, as discussed in the next section.

Reactivity Prediction

Several different computational approaches can be used to help predict the chemical reactivity of a molecule. It is sometimes useful to use more than one approach, and make sure the different methods predict the same thing! Of course, chemical intuition is also of crucial importance. In this section we will discuss the use of: (1) partial charge, (2) electrostatic potential, (3) HOMO/LUMO, and (4) reactivity indices. Each case will describe the area of applicability and will also provide an example. [Note: Calculations in this section were performed using B3LYP/6-31G(d)].

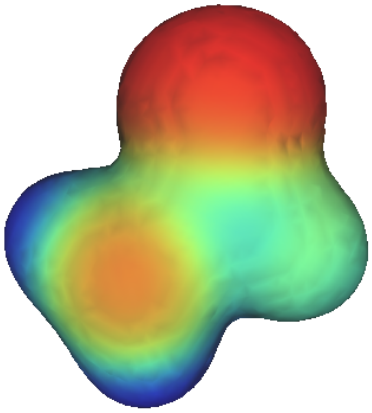
Computational chemistry software packages will provide the *partial, or total atomic charge* for each atom in a molecule. Mathematically, the program partitions the electron density onto individual atoms and then, after taking into account the atomic number (number of protons in the nucleus), the charge on each atom is calculated. Different software packages use different methods in partitioning the electron density between atoms, and the results can depend on the basis sets used in the calculation. The partial charges calculated can sometimes be misleading, and care should be taken in their interpretation. With that caveat, partial charges are often useful in understanding charge-controlled reactions involving "hard" electrophiles and nucleophiles. Electrophiles are electron pair acceptors that seek areas of a molecule with increased electron density in order to form a covalent bond. A proton is an example of a "hard" electrophile, in the Lewis acid sense, due to its high charge density. Nucleophiles are electron pair donors that will react to form covalent bonds with areas of a molecule depleted in electron density.

Let's apply the concept of partial charges to predict the site of protonation in the formamide molecule. The partial charges are shown below:

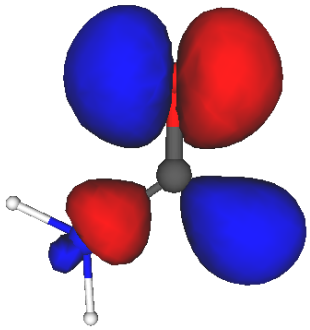


Because a proton has a positive charge, it should be most strongly attracted to the nitrogen atom in formamide, since it has the most negative charge. This is actually NOT the most likely site of protonation. The nitrogen atom has three atoms attached to it (two hydrogens and one carbon) that all carry partial positive charges. In addition, there are two lone pairs of electron on the oxygen atom.

Instead of partial charges, let's explore the electrostatic potential of the formamide molecule, shown below (same orientation as above).

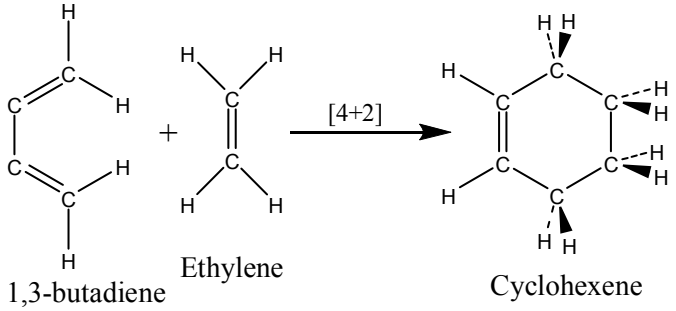


The electrostatic potential predicts that the site of protonation will be the oxygen atom, which makes more sense. The orbital that contains the electron pair that the proton seeks should be the HOMO. This orbital is shown below:



Both the location of the HOMO and the electrostatic potential confirm that the oxygen atom will be the site of protonation. For more complex reactions, we can use the HOMO and LUMO, sometimes referred to as *frontier orbitals*, to understand the location of bond formation. A

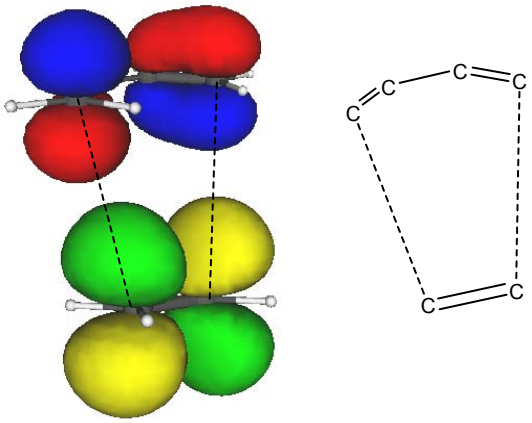
simple example is the Diels-Alder reaction between ethylene and 1,3-butadiene to form cyclohexene, as shown below:



1,3-butadiene + Ethylene → Cyclohexene

This reaction is known as a [4+2] cycloaddition reaction because four π-electrons from butadiene and two π-electrons from ethylene are involved. Two new carbon-carbon single bonds are formed in the product. The important orbitals involved here (and in other reactions of this type) are the HOMO of 1,3-butadiene (red/blue) and the LUMO of ethylene (green/yellow):

1,3-butadiene - HOMO

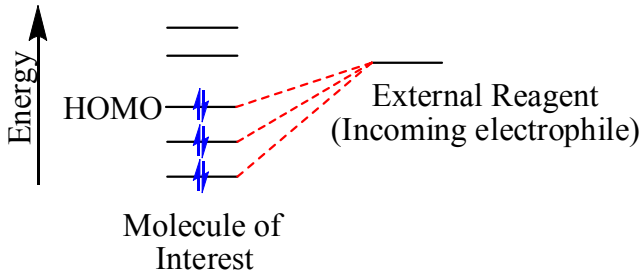


Ethylene - LUMO

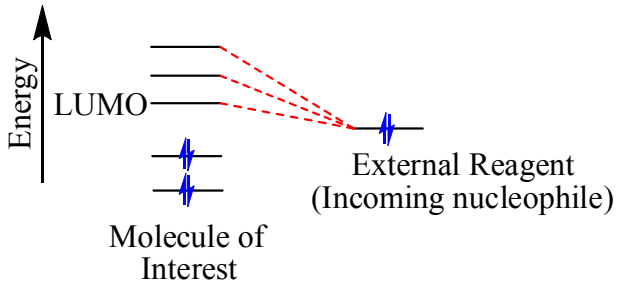
Positive overlap occurs between the red (HOMO) - green (LUMO) and blue (HOMO) - yellow (LUMO) combination of lobes. Note that the lobes of the HOMO are largest on the terminal carbon atoms, and it is these atoms that become bound to the carbon atoms of the ethylene.

In larger molecules the HOMO and LUMO are close in energy to a number of other molecular orbitals. Because these other molecular orbitals will contribute to the reactivity, it is no longer sufficient to just look at the HOMO or LUMO. In these cases, a *reactivity index* can be calculated, and the result is mapped onto the electron density surface, just as the electrostatic potential is. These indices are known as *susceptibilities*, and are of three types: electrophilic, nucleophilic, and radical. The *electrophilic susceptibility* of a molecule will identify sites on that molecule most likely to react with an incoming electrophilic reagent. Similarly, the *nucleophilic susceptibility* of a molecule will identify sites where an incoming nucleophile is most likely to react. Finally, the *radical susceptibility* of a molecule will show sites where an incoming free radical is most likely to react.

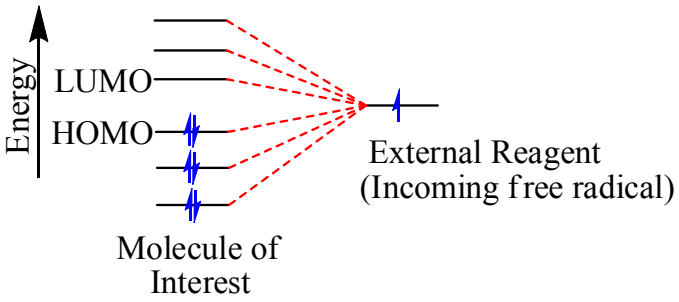
To determine the *electrophilic susceptibility* of a molecule, the computer program sums up the contributions of the HOMO and the occupied orbitals that are close in energy to the HOMO on a point by point basis and displays the result on the electron density surface. Schematically, the process is shown below:



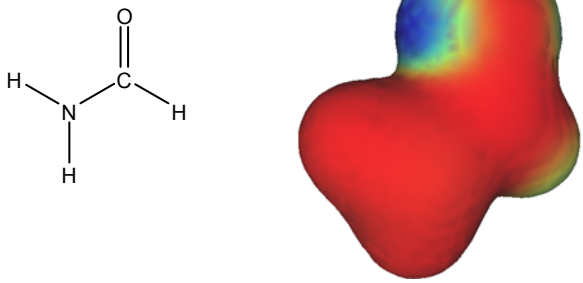
To determine the *nucleophilic susceptibility* of a molecule, the computer program sums up the contributions of the LUMO and the unoccupied orbitals that are close in energy to the LUMO on a point by point basis and displays the result on the electron density surface. Schematically, the process is shown below:



The *radical susceptibility* of a molecule is found by taking an average of the electrophilic and nucleophilic susceptibilities and the result is displayed on the electron density surface. Schematically, this process is shown below:

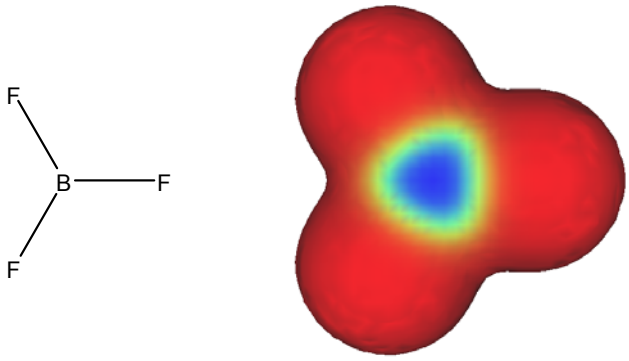


Shown below is the electrophilic susceptibility of formamide, which we earlier studied by different means: (Note: The molecule is in the same orientation as the earlier figures).



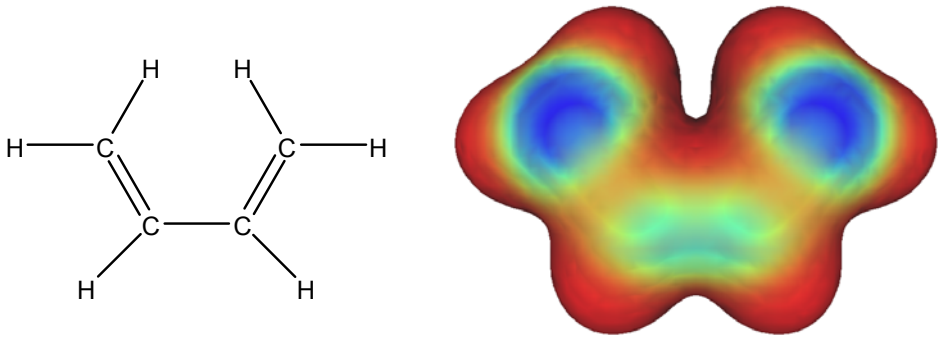
The blue "bullseye" on the oxygen atom again shows that an electrophilic reagent such as a proton is most likely to react here.

Boron-containing molecules are often electron deficient and are good lone pair acceptors (Lewis acids). The nucleophilic susceptibility of BF_3 is shown below:



Clearly, the blue "bullseye" on boron shows that an incoming nucleophile will react here.

An example of radical susceptibility is shown below for the butadiene molecule. Again, the darker blue "bullseye" on the terminal carbon atoms indicates that a free radical will react at these carbon atoms.



In summary, chemical intuition should always be used in conjunction with computation when determining molecular reactivity. When possible, at least two of the different methods described above should be used to make sure the predictions are consistent.

Chapter 13: Calculating Spectra: Prediction of Vibrational Frequencies (Infrared) and Electronic Transitions (Ultraviolet-Visible)

Key Notes:

Fundamental Aspects:

The absorption of infrared (IR) light by a molecule will cause excitation of the vibrational motions of the atoms present. Different types of bonds in the molecule will absorb light of different wavelengths, thus allowing *qualitative* identification of certain bond types in the sample. The total number of degrees of internal motion for a nonlinear molecule with N atoms is given by $3N - 6$. A linear molecule will have $3N - 5$ total degrees of internal motion. Most of these internal degrees of motion are experimentally observable vibrational modes. IR spectroscopy is a widely used type of qualitative analysis for sample identification.

The absorption of ultraviolet or visible (UV-Vis) light by a molecule can excite electrons from lower energy, filled molecular orbitals to higher energy, empty molecular orbitals. This process is called an electronic transition. The specific wavelengths of light absorbed depend on the types of functional groups present in the molecule, and this information can help with qualitative identification of these functional groups. Ultraviolet-Visible spectroscopy is more commonly used for *quantitative* analysis, where the identity of the absorbing species is known, and its solution concentration is sought. The Beer-Lambert Law is used to relate the amount of light absorbed by the sample to its solution concentration.

Applications of Infrared Spectroscopy:

Infrared data is used to help determine molecular structure. Correlation tables are available that match certain organic functional groups with absorbed wavelengths of light. A detailed comparison between a sample spectrum and reference spectra of pure compounds allows the identity of a substance to be determined. Infrared spectroscopy is often used in conjunction with other spectroscopic techniques to identify organic compounds.

Calculating Infrared Spectra:

Any of the methods discussed in Chapters 6 – 9 can be used to calculate vibrational frequencies. In order for the calculated and experimental results to be directly comparable, the *structure must first be optimized* before a frequency calculation is performed. Frequency calculations should be computed with the same method and level of theory used to optimize the geometry. Density functional theory gives the best overall agreement with experimental data. With larger molecules, the computational expense of DFT can become prohibitive. While other less expensive methods may have poorer accuracy, the results can still be useful for comparison with experimental data. The calculated results with many software packages allow animation of each molecular vibration and can help students better comprehend what is happening at the molecular level.

Applications of Ultraviolet-Visible Spectroscopy:

Since certain functional groups present in organic molecules absorb light at characteristic wavelength in the UV-Vis region, the technique can be used to qualitatively identify the presence of these groups. Correlation tables are available for this purpose. For systems that contain conjugated double bonds, there are sets of rules that allow prediction of the wavelength of maximum absorbance (λ_{max}). The most common use of UV-Vis spectroscopy is quantitative analysis, where the concentration of the dissolved analyte is calculated from the Beer-Lambert Law, which states that the amount of light absorbed by a sample is directly proportional to its concentration. Standard solutions of the analyte are used to produce a working curve of light absorbed versus concentration. Solutions of unknown concentration are then analyzed.

Calculating Ultraviolet-Visible Spectra:

The ability to calculate the UV-Vis spectrum of a molecule can help in interpretation of an experimental spectrum, will show the orbitals involved in a given electronic transition, and can shed light on the electronic structure of the molecule. The *geometry must first be optimized*, the ground state wavefunction is calculated, and then a calculation is performed that mixes some of the higher energy, empty (virtual) molecular orbitals into the ground state wavefunction. This is the basis of what is called the configuration interaction method. From these calculations an approximation to the energy of the excited electronic states can be calculated. The energy difference between the ground state and various excited states is then found to give the frequency of each transition.

Fundamental Aspects

Spectroscopy is the interaction of light with matter. In this context "light" refers to the entire electromagnetic spectrum. Light can be treated as a wave where the speed of light (c) is equal to the wavelength (λ) in meters multiplied by the frequency (ν) in hertz (cycles per second, or s^{-1}):

$$c = \lambda\nu \approx 3.00 \times 10^{8}\ m{\cdot}s^{-1}$$

Light can also be treated as a particle (photon) whose energy is equal to Planck's constant (h) multiplied by the frequency:

$$E = h\nu = hc/\lambda \qquad h = 6.63 \times 10^{-34}\ J{\cdot}s$$

The electromagnetic spectrum shown below summarizes the various wavelengths, frequencies, and regions.

Different types of spectroscopy utilize different regions of the electromagnetic spectrum. Light at longer wavelengths, beyond the red end of the visible spectrum, lies in the infrared region (*ca*. 25,000 – 2500 nm). Typical units used to report IR data are reciprocal centimeters (cm^{-1}), and the most common IR spectrophotometers cover the mid-IR region, 4000 to 400 cm^{-1}. Infrared light can be absorbed by a molecule and will excite the vibrational modes. For a nonlinear polyatomic molecule with N atoms, there are $3N - 6$ degrees of internal motion. Linear molecules will have $3N - 5$ degrees of internal motion. Bond stretching typically occurs at higher energies. Vibrational modes that involve changes in bond angles usually have lower energies. Some of these motions are shown below:

In order for the infrared light to be absorbed, there must be a change in the dipole moment of the molecule as it vibrates. This condition may not be met by *all* the possible modes of internal motion, so some modes are not observable using IR spectroscopy and are *IR inactive*. Another less common technique used to explore molecular vibrations is Raman spectroscopy. In this technique, only those vibrations which result in a change in the polarizability of a molecule are observed. In some cases, vibrations that are IR inactive will be *Raman active*, so both techniques can be used. When the vibrational modes of a molecule are calculated and are compared with experimental results, it is important to remember that some modes may be IR inactive. Some modes will only be observable using Raman spectroscopy. In depth analyses of molecular symmetry using group theory can distinguish between IR and Raman active modes, but these details will not be covered here.

A detailed analysis of a Infrared (IR) spectroscopy is typically used in *qualitative analysis*, meaning that the technique can provide information about the chemical *identity* of a sample under study. Samples can be solutions, pure liquids, solids, and (more commonly) pastes or mulls made from a solid and mineral oil (Nujol®). Different types of chemical bonds present in a sample absorb characteristic wavelengths of infrared light. With practice, a scientist can interpret the infrared spectrum of a sample and learn what types of chemical bonds, or functional groups, are present. In conjunction with additional information about the sample, a scientist may be able to identify exactly what molecule is present. The ability to calculate the infrared spectrum of a given molecule can greatly assist in the interpretation of an experimental spectrum. The area of an IR spectrum between *ca*. 1300 – 900 cm^{-1} is called the fingerprint region, and often contains many peaks that are difficult to assign to particular vibrational modes. The calculated frequencies can be animated to show the corresponding molecular motions, and more peaks in the spectrum can be identified.

At shorter wavelengths than the IR region is the visible region, 700 to 400 nm. Beyond the violet end of the visible region lies the ultraviolet, 400 – 200 nm. Light in the ultraviolet-visible region of the electromagnetic spectrum can be absorbed by a molecule and will excite electrons from lower energy, filled molecular orbitals to higher energy, empty molecular orbitals. This is called an electronic transition. The wavelength at which the maximum absorption of light occurs (λ_{max}) is characteristic of the electronic structure of the molecule. The electronic structure, in turn, depends on the functional groups present in an organic molecule, or on the metal ion (and its oxidation state) for a transition metal containing inorganic molecule or ion. Thus, UV-Vis spectroscopy can be used in *qualitative analysis* to help identify the molecular structure of the sample under study. More commonly, UV-Vis spectroscopy is used in *quantitative analysis*, where the solution concentration of a known substance is found. The Beer-Lambert Law states that the amount of light absorbed by a sample is proportional to the solution concentration of the sample. The ability to calculate the UV-Vis spectrum of a given molecule can help in the interpretation of an experimental spectrum and can show which molecular orbitals are involved in each electronic transition. These calculations can also help to understand the electronic structure of a molecule.

Applications of Infrared Spectroscopy

Infrared data is used to help determine molecular structure. The ability to compare experimental and computed spectra can aid in the correlation of specific vibrational motions of the molecule

with observed spectral peaks. Correlation tables are available that match certain organic functional groups to absorbed wavelengths. There exist reference IR spectra of pure compounds. The fingerprint region (*ca*. 1300 – 900 cm^{-1}) of the reference spectra can be compared with an experimental spectrum to identify a particular molecule. Computed IR data can also help in identifying transition state structures (Chapter 14). Other purposes for performing these calculations include computation of force constants for geometry optimization (Chapter 11) and determination of zero point vibration and thermal energy corrections to the total energies, as well as other thermodynamic quantities of interest (Chapter 15).

Calculating Infrared Spectra

Any of the methods discussed previously (molecular mechanics, *ab initio*, semiempirical, or density functional theory) can be used to calculate IR spectra. The geometry of the molecule *must be optimized* before the frequency calculation is done so that the calculated results and experimental results are done on molecules with the most similar geometry possible. Molecular mechanics can give usable results if the molecule being studied is similar in structure to those used to create the force field. Since many molecules will not have adequate parameters in the standard force fields available, molecular mechanics is not the best method for these calculations. Both semiempirical and *ab initio* methods have known, systematic errors that can be compensated for. Once the calculation is complete, the wavelengths obtained are multiplied by a scaling factor in order to bring them into better agreement with experimental data. The best method for calculating vibrational frequencies is density functional theory. Although calculated results using DFT give the best overall agreement with experimental values, the computational costs associated with DFT can be prohibitive.

Applications of Ultraviolet-Visible Spectroscopy

Since certain functional groups present in organic molecules absorb light at characteristic wavelengths in the UV-Vis region, the technique can be used to qualitatively identify the presence of these groups. Tables of absorption data for various functional groups are available. If the molecule of interest contains a conjugated system of double bonds, a set of simple rules, called the Woodward-Fieser Rules, can be used to predict the wavelength of maximum absorption (λ_{max}). Many transition metal ion containing complexes absorb light in the visible region of the spectrum. The light absorbed causes electronic transitions of the d-electrons. Different groups attached to the metal will change the electronic energy levels, thus changing the color (value of λ_{max}). Quantitative analysis of the electronic structure of inorganic molecules can be performed in this manner. The more common application of UV-Vis spectroscopy is in quantitative determination of the concentration of a dissolved analyte, found through application of the Beer-Lambert law:

$$A = \varepsilon bc$$

where A = amount of light absorbed,
ε = molar absorptivity or extinction coefficient ($L \cdot mol^{-1} \cdot cm^{-1}$)
b = pathlength of light through the sample (cm)
c = concentration in units of molarity ($mol \cdot L^{-1}$)

Calculating Ultraviolet-Visible Spectra

The ability to calculate the UV-Vis spectrum of a molecule can help in the interpretation of an experimental spectrum, will show the orbitals involved in a given electronic transition, and can also shed light on the electronic structure of the molecule. The first step in the calculation is to perform a geometry optimization. Any of the four methods (molecular mechanics, *ab initio*, semiempirical, or density functional theory) could be used for this. Next, the ground state wavefunction is calculated. This generates occupied (filled with two electrons each) and unoccupied (virtual) molecular orbitals. Molecular mechanics cannot be used for this step, since no electrons or orbitals are used in this method. Any of the other techniques could be used. Next, a calculation is performed that mixes some of the virtual orbitals into the ground state wavefunction while the geometry is held constant. This process provides an approximation to the energy of the excited electronic states at the fixed geometry of the ground state. The transition frequency is calculated by finding the difference between the excited state and ground state energies.

Chapter 14: Transition States

Key Notes:

Fundamental Aspects

Chemists have devised a number of theories to help explain and understand chemical reactions. *Collision theory* assumes that reactant molecules must collide with the proper orientation and with energy greater than some minimum value known as the activation energy (E_a). *Transition-state theory* states that two colliding molecules form an activated complex, or transition state, which is an unstable grouping of atoms. The activated complex can fall apart in different ways and revert to reactants or yield products. The *Arrhenius theory* describes the relation between the rate constant k, the temperature T, and E_a. The activation energy is the difference between the energy of the *transition structure* and the energy of the reactants. The transition structure and its energy can be calculated, as well as the energy of the reactants and products. Computational chemistry has become an important tool in advancing our knowledge of how reactions occur.

Potential Energy Surfaces

A plot of how the potential energy of a system changes as a function of structure is called a potential energy surface. Potential energy surfaces have several important features. The energy *minima*, or valleys, correspond to the equilibrium structures of a given molecule, or to the relative energy of a reactant and product. For the potential energy surface of a simple reaction $A \rightarrow B$, the path between the minima for A and for B is the *reaction path* or *profile*, sometimes called the *reaction coordinate*. The reaction coordinate will typically have a maximum in energy at a *saddle point*. The geometry found at the saddle point is the *transition structure*, and the energy of this structure can be used in finding the activation energy for a reaction.

Reaction Coordinate

The potential energy diagrams that plot energy vs. reaction coordinate (or "progress of reaction") that chemists regularly employ are cross sections of the full potential energy surface that describes the system. These simplified diagrams are useful in visualizing activation energies and whether reactions are exothermic or endothermic. They are also commonly used to illustrate the decrease in activation energy in catalyzed chemical reactions.

Calculating Transition Structures

Finding the correct transition structure geometry can be difficult and involves trial and error. There are various methods that can be used, and chemical intuition is an important factor. Many computational software packages have "transition state optimization" routines that can be helpful. Once a possible structure is identified, an IR calculation (Chapter 13) should reveal one imaginary (negative) frequency. This frequency corresponds to atomic motions that lead towards lower energy structures. If one imaginary frequency is found for the candidate structure,

this means the structure is indeed at a saddle point, but it may *not* be the saddle point on the reaction path connecting reactants and products! If the correct transition structure has been found, animation of the normal vibrational mode that corresponds to the imaginary frequency should show that the atomic motions lead in the direction of reactants and products.

Fundamental Aspects

Chemists would like to know the details of how chemical reactions occur on an atomic level. Why do certain reactions take place, but not others? If a single molecule undergoes some type of rearrangement, how are the old bonds broken and the new bonds formed? If a given molecule has equivalent bonds, do they break simultaneously or one at a time? If a reaction gives a mixture of products, what factors are responsible for the product ratio? Why are some reactions fast while others are slow, and why do reaction rates depend on temperature? A number of theories have been devised over the years in order to provide answers to questions such as these. Experimentally, this is a difficult task given the small distances and very short timescales involved. Recent work in femtosecond (10^{-15} s) spectroscopy has begun to answer some of these questions, and computational results on transition structures continue to provide important insight.

The rate of a chemical reaction depends on the temperature. For the reaction:

$$2\ NO_2(g)\ +\ F_2(g)\ \rightarrow\ 2\ NO_2F(g)$$

The experimental rate law, which relates the rate of a reaction to the concentration of reactants, is given by:

$$\text{Rate} = k[NO_2][F_2]$$

The square brackets denote molar concentration. Experiments show that the rate constant (k) varies with temperature. In general, the reaction rate increases with increasing temperature. *Collision theory* explains the temperature dependence of rate constants by assuming that reactant molecules must collide with energy greater than some minimum value (the activation energy E_a) and that the collision orientation must be correct for a reaction to occur. As the temperature increases, the gas phase molecules in the above reaction will be moving faster resulting in more collisions. Increased molecular speed means increased kinetic energy, so more of the collisions will have energy greater than E_a. The orientation requirement is temperature independent. *Transition-state theory* goes further in postulating that colliding reactants form an activated complex (transition state) that can break up to either reform reactants or go on to form products. In thermodynamic terms, this theory relates the rate constant to the *Gibbs free energy of activation* ($\Delta G^{\ddagger}$) which is equal to the difference in Gibbs free energy between the transition state and the reactants. The geometry of the *transition state* referred to here is at the peak of the Gibbs free energy (G) reaction profile, <u>not</u> the potential energy reaction profile. Note that the geometry of the *transition structure* discussed above occurs at the maximum of the potential energy reaction profile. While we typically refer to the "transition state", what we often really mean is the "transition structure".

From the computational standpoint, one approach is to use the Arrhenius theory which relates the rate constant (k) to the activation energy (E_a) via:

$$k = Ae^{-E_a/RT}$$

where A is the frequency factor
e is the base of the natural logarithm
R is the ideal gas constant
T is the absolute temperature

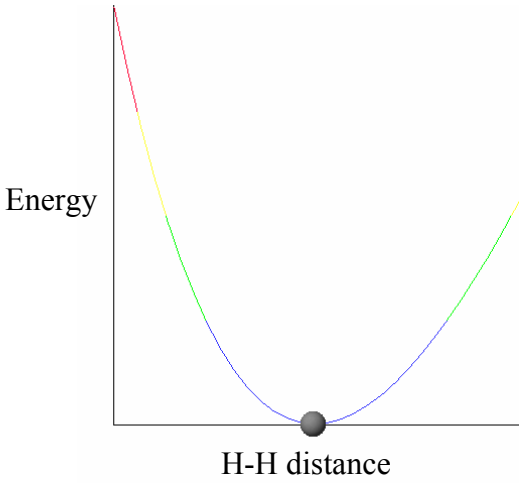
Experimentally, a plot of ln k vs. $1/T$ should give a straight line whose slope is $-E_a/R$ and intercept is ln A. We can also *calculate* the energy of a transition structure and obtain E_a via:

$$E_a = E_{\text{transition structure}} - E_{\text{reactants}}$$

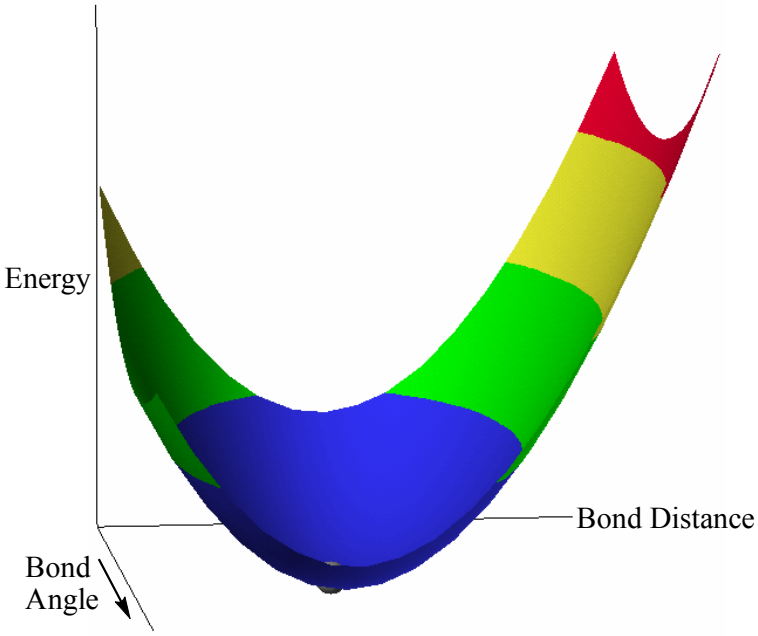
The Arrhenius theory is approximate and absolute values of the frequency factor A are hard to calculate. It is thus very difficult to calculate absolute rate constants with any degree of accuracy. If we can make A values cancel out (e.g. by looking at ratios of rate constants) we can obtain quite useful results.

Potential Energy Surfaces

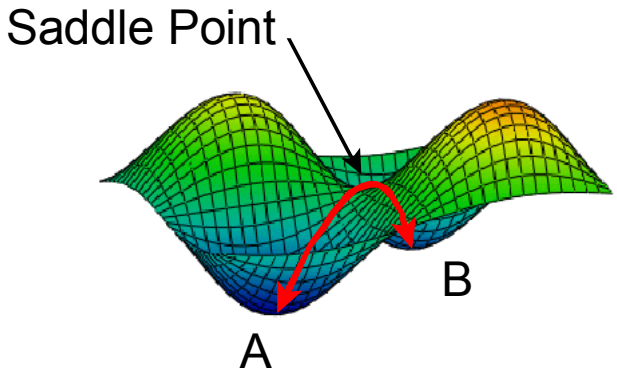
A potential energy surface shows how the energy depends on structure. The surface will have as many dimensions as the total number of degrees of internal motion of the system being studied. A diatomic molecule ($N = 2$) will have only one degree of internal motion, which is the bond length ($3N - 5 = 1$). The potential energy surface in this case is a simple two-dimensional plot of energy vs. bond distance, as shown below for the H_2 molecule.



For a symmetric nonlinear triatomic molecule like H_2O, a three-dimensional potential energy surface plots energy (z-axis) vs. O-H bond length (assuming both O-H bond lengths are identical) and H-O-H bond angle (x and y axes).



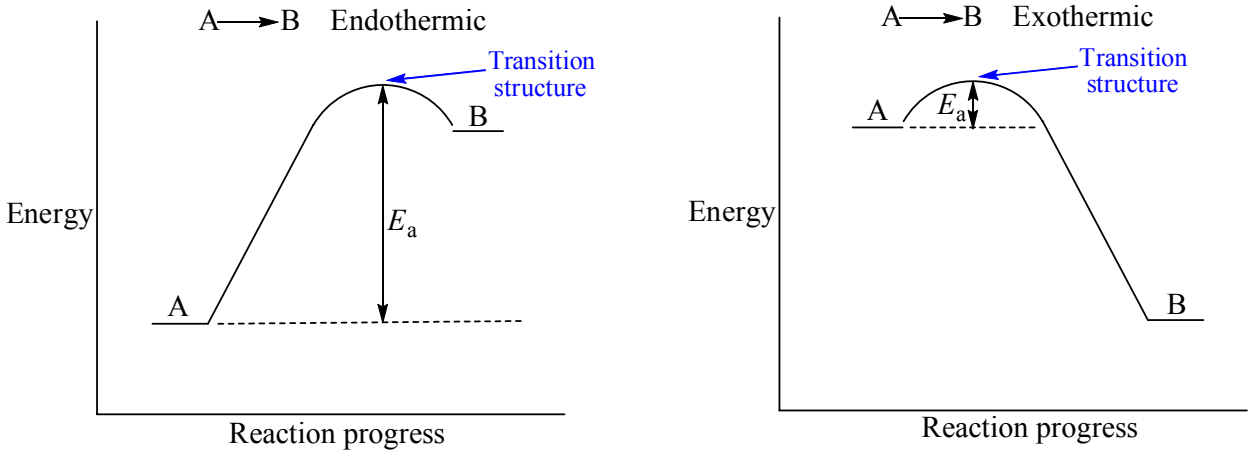
Systems involving more than two structural parameters cannot be plotted. For a simple reaction such as A → B, a potential energy surface can show the relative energies of reactant and product, and the minimum energy path, or reaction coordinate (shown in red below) between them:



The highest energy structure along the reaction coordinate shown above is the transition structure, which lies on the saddle point. A saddle point lies at a maximum in energy along one direction (the reaction coordinate) and is at a minimum of energy in all directions other than the reaction coordinate (see above). If we correctly identify the transition structure, it should have one imaginary frequency whose normal mode reflects atomic motions that lead to reactants or products.

Reaction Coordinate

Chemists often simplify the figure above showing the potential energy change of the general reaction A → B by looking at a cross section of the diagram in two dimensions. From the above diagram, it appears that the reaction is endothermic (reactants have lower energy than products; reaction absorbs heat), as shown on the left:



A general exothermic reaction (reactants have higher energy than products; reaction releases heat) is shown on the right. These diagrams make clear that E_a is the difference in energy between the reactant(s) and the transition structure. An analogous physical situation involves rolling a heavy ball from one valley, over a hill, then letting it roll down into another valley. The amount of work done "rolling uphill" represents E_a. Our ability to computationally model a transition structure and calculate its energy (and the energy of reactants and products), is much easier than going through the experimental process described above. Comparison of experimental and calculated E_a values shows that, depending on the level of theory we use and the computational expense we are willing to deal with, that very good results can be obtained. Reactions involving dangerous substances and those that are otherwise difficult to follow experimentally can now be explored using computer technology.

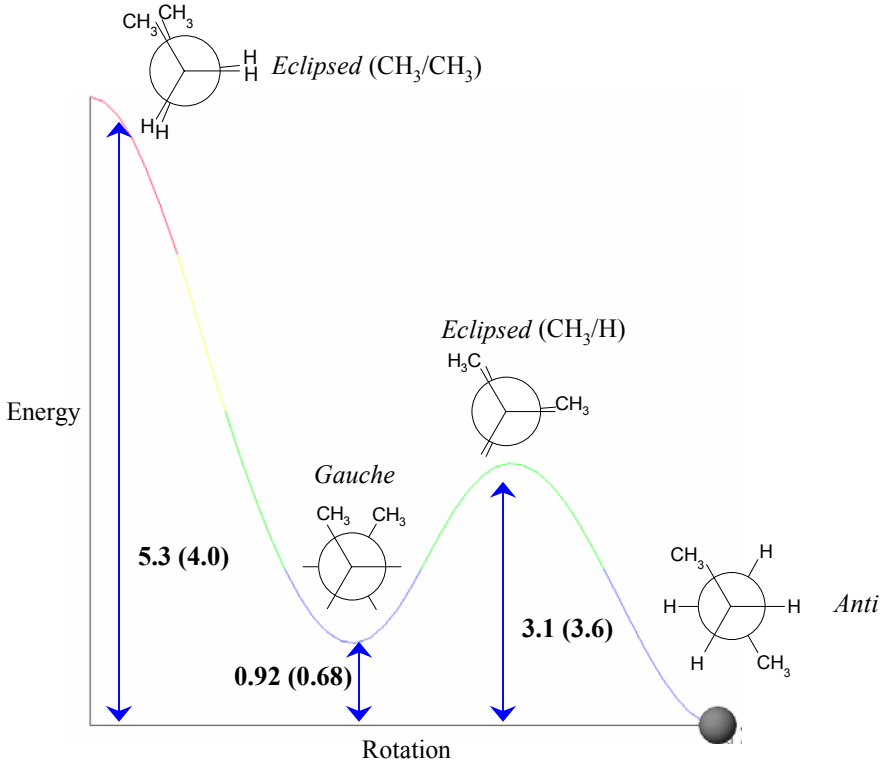
Calculating Transition Structures

As chemists work to further their understanding of chemical reactivity, the ability to compute transition structures has provided great insight. However, the calculations can be difficult, and the methods used all involve some trial and error. Chemical intuition can assist in providing a framework for an initial guess. The difficulty lies in finding the correct geometry for the transition structure. We can verify that our transition structure may a good one by calculating the IR spectrum (Chapter 13). If the structure we have found is indeed at a saddle point on the potential energy surface, we should see one imaginary (negative) frequency value. However, it is possible we are at the *wrong* saddle point on the potential energy surface – *not* the one that connects reactants and products! Many programs used to calculate vibrational frequencies allow animation of the atomic motions for a given normal mode. If the transition structure we have found is indeed at the correct saddle point that connects reactants and products, animation of the imaginary frequency should reveal atomic motions that lead in the direction of reactants (bond breaking) and products (bond formation).

As discussed in Chapter 11, the geometry optimization process leads to what is called a

stationary point on the potential energy surface. A stationary point may be a minimum (local or global) on the potential energy surface, or a maximum (local or global). If we are at a minimum, all of the calculated vibrational frequencies will be positive values. A transition structure will lie at a saddle point (local maximum) and can be identified by one imaginary (negative) vibrational frequency. The trick is to build a geometry that is close enough to the transition state so that optimization will find the correct transition state geometry. Most software programs include a "transition state optimization" routine. Once a good guess at the transition state geometry is built, this routine can assist in finding the transition state, but this approach may not always work. Finding a good transition state will often involve some trial and error. Always perform an IR calculation and check for one imaginary frequency. Explore the animation of this frequency and see if the atomic motions lead towards reactants and products. Several examples of increasing complexity are provided below.

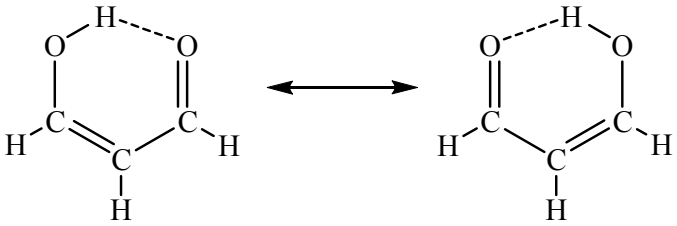
Example #1: Conformations of n-*butane.* As we rotate around the C_2-C_3 bond in butane, we form various conformers of different energy. The highest energy conformer (global maximum) occurs when the two methyl groups are eclipsed, due to the van der Waals (steric) repulsion between them. The minimum energy conformer (global minimum) occurs when the two methyl groups are pointing in opposite directions, or *anti*. Another maximum (local maximum) occurs when the methyl groups are eclipsed with hydrogen atoms. A local minimum is found in the *gauche* conformer, as shown below:



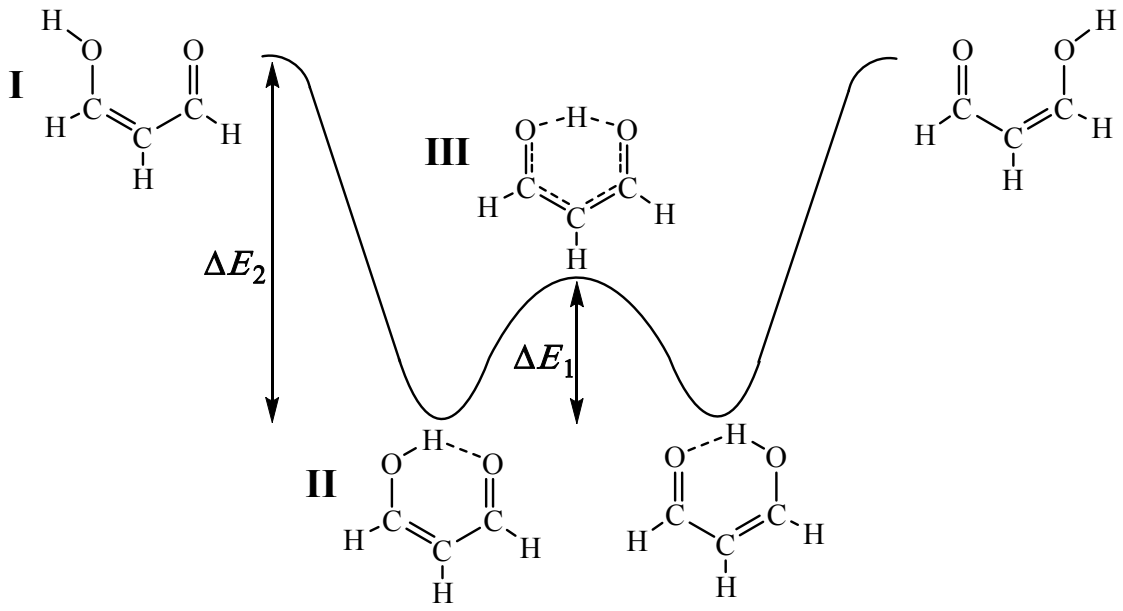
The numbers for the relative energies of the different conformers were calculated using DFT (B88-LYP, DZVP) and are in kcal mol^{-1}. Experimental values are in parentheses. Each of the four structures shown in the above figure was constructed, and a geometry optimization was performed in each case to obtain the energy. The C_1-C_2-C_3-C_4 angle was fixed at 120° for the

lower energy eclipsed conformation, at 60° for the *gauche*, and at 0° for the highest energy eclipsed conformation. To ensure that minima were located for the *anti* and *gauche* conformations, IR frequency calculations were performed on the optimized structures, and all frequencies found were positive. Similarly, IR frequency calculations performed on the two eclipsed structures each gave one imaginary (negative) frequency value, confirming that transition structures had indeed been found. Differences in energy values were calculated, and the experimental values were found on the National Institute of Standards and Technology's Computational Chemistry Comparison and Benchmark Database website (See: http://srdata.nist.gov/cccbdb/exprotbar.asp).

Example #2: Proton Transfer in Malonaldehyde Enol. This compound is the simplest system that has an *intra*molecular proton transfer between two oxygen atoms, as shown below:



It's pretty easy to guess what the transition state would look like. The proton would be equally shared between the two oxygen atoms. In the two structures shown above, we would also expect some hydrogen bonding between the oxygen-bound proton and the carbonyl (doubly bonded) oxygen. This is indicated with the dashed "weak bond" line. If the proton bound to the oxygen were turned away from the carbonyl oxygen, no intramolecular hydrogen bonding would be present, and we would expect such a molecule to be at higher energy. So, we have three structures whose energy needs to be calculated. Two of them will also have IR frequency calculations performed to make sure they are minima and maxima (saddle point) on thepotential energy surface. The calculations were performed using density functional theory [B3LYP/6-31G(d)]. The results are shown below:

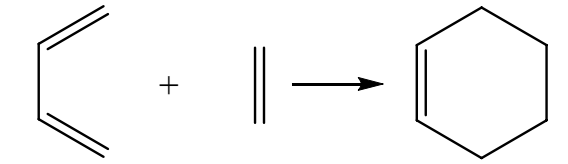


The difference in energy between compounds **II** and **III** in the above diagram (ΔE_1) is 3.5 kcal mol^{-1} (lit. value 4.0). Compounds **I** and **II** differ in energy (ΔE_2) by 15.0 kcal mol^{-1} (lit. value 12.4). To further understand the changes in structure which occur during the proton transfer, various bond lengths for compounds **I**, **II**, and **III** are collected in the following table. "Left and "right" bond lengths refer to the structures above.

Bond length (Å)	**Cmpd. I**	**Cmpd. II**	**Cmpd. III**
O-H (left)	0.969	1.005	1.215
O-H (right)	n/a	1.689	1.215
C-O (left)	1.346	1.322	1.282
C-O (right)	1.219	1.244	1.282
O- - - - - -O	2.865	2.590	2.378
C-C (left)	1.350	1.367	1.401
C-C (right)	1.465	1.439	1.400

From the values in the first row of the table, the O-H bond length increases as the bonding to the second O atom increases. The C-O bond on the left shortens with increasing double bond character, while the C-O bond on the right lengthens. The two O atoms move substantially closer in the transition state as the proton becomes equally shared just before transfer. The C-C bond lengths also change during the process. It is these details of a reaction revealed by computation that greatly increases our understanding of chemical reactivity.

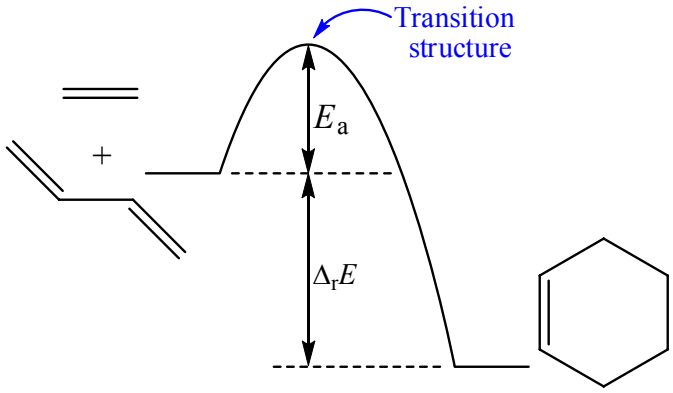
Example 3. Diels-Alder Addition of Butadiene and Ethylene. Our last example involves two separate reactant molecules that combine to form a single product. The Diels-Alder reaction is a synthetically useful example of such a system. In this reaction a conjugated diene reacts with an alkene (the dienophile) to form a 6-membered ring:



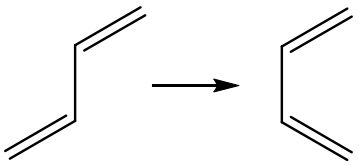
Diene (butadiene) Dienophile (ethene) Cyclohexene

The diene has four π electrons while the dienophile has two π electrons. This reaction is thus known as a [4 + 2] cycloaddition. Four of the six π electrons present in the reactants are used to make the two new σ bonds in the product.

The reaction can be summarized in the following potential energy diagram:

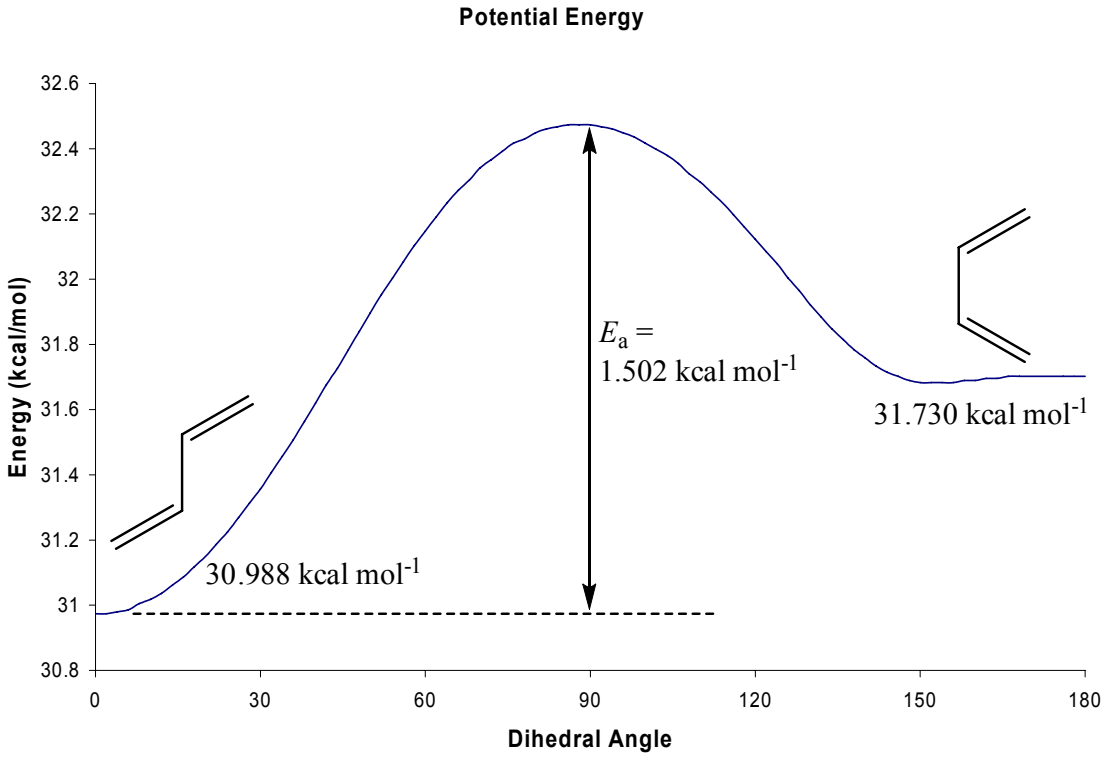


The most stable form of butadiene is the *s-trans* form. In order for the cycloaddition reaction to occur, butadiene must be in the higher energy *s-cis* form:

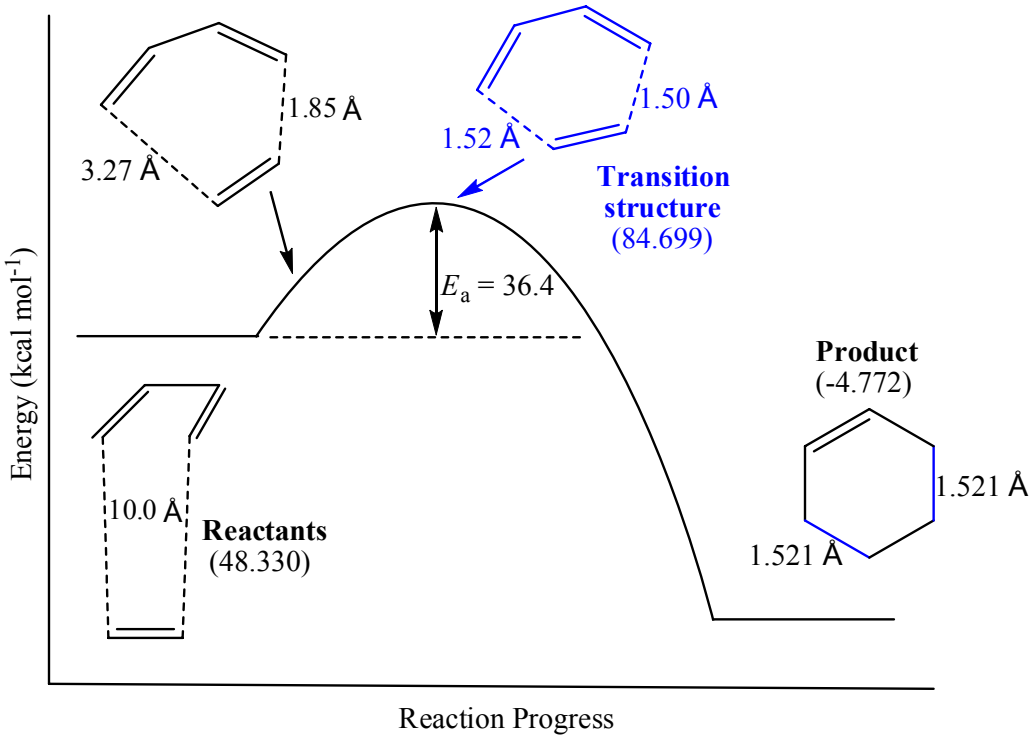


s-trans *s-cis*

To find the energy difference between these two conformers, we follow a procedure similar to that from Example #1 above. The calculation was performed using the semiempirical PM3 method:



To find E_a for the cycloaddition reaction, the potential energy of the reactants is calculated when they are well-separated and also as they are brought closer and closer together. An energy maximum is reached (transition structure), and then the product is formed. The calculation was again performed using the PM3 method. Energy results for reactants, transition state, and product are given in parentheses. Carbon-carbon distances for the new bonds are also given:



The E_a value of 36.4 kcal mol^{-1} does not compare very well with the literature value of 27 ± 2 kcal mol^{-1}, but accurate quantitative results are not to be expected using the PM3 method.

Chapter 15: Thermochemistry

Key Notes:

Fundamentals Aspects

Thermochemistry is an area of chemistry that deals with the relationship between energy and chemical reactions. Chemists would like to know if a proposed chemical reaction is going to be endo- or exothermic, if it will proceed to give products, or if it will produce a mixture of reactants and products. Thermochemistry seeks to answer these questions by looking at thermodynamic properties of the substances involved in a reaction. Computational chemistry allows one to calculate properties for these substances in order to provide the answers chemists require, without resorting to experimental means of determining this information.

Internal Energy

The *internal energy* (U) is the sum of the kinetic and potential energies of the particles that make up a system. Kinetic energy involves the motion of the electrons, nuclei, and the molecules themselves while potential energy is present in chemical bonds between atoms and in intermolecular forces. Internal energy is a *state function*. A state function depends only on the present state of the system and is completely determined by variables such as temperature and pressure. As a system changes from one state to another, the internal energy changes from one definite value to a new definite value. The change in internal energy (ΔU) equals the difference in internal energy between the final and initial states. In terms of a chemical reaction, the change in internal energy is the difference between the internal energy of the products and that of the reactants.

Enthalpy

The *enthalpy* (H) is a property of a substance that can be used to calculate the heat produced or absorbed in a chemical reaction. Enthalpy is also a state function, and we calculate the enthalpy change for a chemical reaction by finding the difference in enthalpy between the products and reactants. Enthalpy is related to internal energy via its precise definition:

$$H = U + PV$$

where P is the pressure and V is the volume. Chemical reactions that release energy (exothermic) are favored.

Entropy

The *entropy* (S) is a measure of the amount of disorder, or randomness, in a system. Entropy is another state function. For reactions involving different phases, we can often predict the sign of the entropy change. Solids have a more ordered structure since the constituent units (atoms, molecules, or ions) have definite locations. Liquids have less order (more entropy) as the units now move

freely within the liquid volume. Gases have the least order (most entropy), as the constituent units are free to move throughout the volume of the container that holds them. Chemical reactions that involve an increase in entropy are favored.

Gibbs Free Energy

The *Gibbs free energy* (G), or Gibbs energy, is the thermodynamic quantity defined by the equation:

$$G = H - TS$$

where T is the temperature. As a chemical reaction proceeds, both H and S change. These changes, denoted using the Δ symbol, allow the change in the Gibbs energy to be calculated:

$$\Delta G = \Delta H - T\Delta S$$

If ΔG for a reaction is positive, the reaction is nonspontaneous. If ΔG is negative, the reaction is spontaneous. A spontaneous reaction, once started, will continue to completion.

Calculating Thermochemical Parameters

Previous chapters have discussed various calculations that were all performed on single molecules in the gas phase at a temperature of 0 Kelvin. The thermochemical properties discussed above are macroscopic in nature, and arise from large collections of like molecules. *Statistical mechanics* relates the microscopic calculations done earlier to the macroscopic properties we are now interested in. Statistical mechanics depends on the *partition function*. The partition function for a single molecule is a sum of exponential terms involving all possible quantum energy states. These energy states involve contributions from translational, rotational, vibrational, and electronic modes. Once the various contributions to the partition function are calculated, a number of thermochemical and macroscopic observables can be calculated, including U, H, S, and G. Calculations performed on the molecules involved in a chemical reaction of interest allow chemists to determine if a particular reaction is endo- or exothermic, spontaneous or nonspontaneous, etc.

Fundamental Aspects

Chemical reactions either produce heat (exothermic) or absorb heat (endothermic). In planning a chemical synthesis, especially for a large scale reaction, it is important to know the amount of heat that will be produced or absorbed as an appropriate method for removing or supplying the heat must be devised. Chemists are also concerned about the extent of a given reaction, or where the position of equilibrium lies. Some reactions proceed entirely to products, others produce an equilibrium mixture of reactants and products, and some do not produce an appreciable amount of products at all. Before performing a reaction in the lab, it is important to know if the planned synthesis is feasible, that is, if a substantial amount of product will be formed. Thermochemistry, also called chemical thermodynamics, can guide the chemist in choosing

among several possible reactions that will produce a given product. Extensive tables (see Chapter 22) of thermochemical data exist. The values supplied in these tables can be used to determine ΔH, ΔS, and ΔG for a reaction of interest. The problem is that many compounds have not had thermochemical parameters determined, or a chemist may be interested in synthesizing an entirely new class of compounds for which this data is unavailable. In either case, computational chemistry can be used to calculate the quantities of interest. Before these calculations can be discussed, a review of some of the parameters and concepts involved is in order.

Internal Energy

We have defined the internal energy (U) as the sum of the kinetic and potential energies of the particles that make up a system. In a molecular system, kinetic energy is present in the motion of the electrons, nuclei, and the translational movement of the molecules themselves. Potential energy is present in the chemical bonds in a molecule and also in forces between molecules (intermolecular forces). When a chemical reaction occurs, the internal energy of the system will change from one definite starting or initial value to a definite final value. Since U is a state function, the details of how the changes in internal energy occur are not important. That is, we are not concerned with exactly how the reaction happens. The change in the internal energy (ΔU) is found by taking the difference between the final value (U_f) and the initial value (U_i):

$$\Delta U = U_f - U_i$$

For a given reaction, the changes in U can be determined by noting the energy exchange between the thermodynamic system (the mixture of reactants) and its surroundings (solvent, container, etc). The energy exchange involves both heat and work.

Heat (q) will flow into or out of the system due to a temperature difference between the system and its surroundings. By convention, heat flowing out of a system is given a negative sign, and heat flowing into a system is given a positive sign. Exothermic reactions produce heat ($-q$) which we detect as a temperature increase in the surroundings. Endothermic reactions absorb heat from the surroundings ($+q$), which we detect as a temperature decrease. Careful measurement of the amount of reactants used and the temperature change of the surroundings allows quantitative data to be obtained.

The other form of energy, work (w), is the result of some force (F) moving an object through some distance (d), or:

$$w = F \Delta d$$

In chemistry, work is usually associated with the production of a gas, so-called expansion work. For example, combustion of the gasoline/air mixture in the cylinder of a car engine produces heat (exothermic) and gases that perform work by pushing against the piston. Work done *by* the system (pushing the piston) is given a negative sign, while work done *on* the system is given a positive sign.

Energy, whether in the form of heat or work, is expressed in units of joules (J). During an exothermic chemical reaction, such as combustion of gasoline, some of the potential energy

stored in the bonds of the molecules making up the gasoline mixture is converted into heat and work. The *first law of thermodynamics* states that energy cannot be created or destroyed, but can be converted from one form into another. In terms of the change in the internal energy of a system (ΔU), we can restate the first law as:

$$\Delta U = q + w$$

The change in the internal energy of a system is equal to the sum of the heat and work.

Enthalpy

For a reaction carried out at a constant pressure, such as the pressure of the atmosphere for a reaction done in an open beaker, the heat of the reaction is known as q_p. It turns out that the change in enthalpy for a reaction (ΔH) is equal to q_p:

$$\Delta H = H_f - H_i = q_p$$

where H_f and H_i are the final (product) and initial (reactant) enthalpy values, respectively. If the enthalpy values of the reactant and product molecules are known, it is possible calculate ΔH for any reaction of interest. Units for enthalpy are typically joules per mole (J/mole) or kilojoules/mole (kJ/mole).

Enthalpy cannot be measured directly for a substance. Enthalpy changes for many reactions have been measured, and these values have been used to create tables of standard enthalpies of formation (ΔH_f°) for a variety of substances. The superscript ° denotes standard conditions (298K, 1 atm. pressure, 1*M* solution concentrations). These tables can be used to calculate the standard enthalpy change (ΔH°) for a reaction:

$$\Delta H^\circ = \Sigma\, n\Delta H_f^\circ(\text{products}) - \Sigma\, m\Delta H_f^\circ(\text{reactants})$$

where the "Σ" symbol means the "sum of" the heats of formation values for either reactants or products, and n and m refer to the stoichiometric coefficients from the balanced chemical equation. For the reaction:

$$4\ NH_3(g) + 5\ O_2(g) \rightarrow 6\ H_2O(g) + 4\ NO(g)$$

the relevant ΔH_f° values (kJ/mole) are: $NH_3(g)$, -46.2; $O_2(g)$, 0; $H_2O(g)$, -241.8; $NO(g)$, 90.4. Substitution of these values into the above equation yields:

$$\Delta H^\circ = [6(-241.8) + 4(90.4)] - [4(-46.2) + 5(0)] = [-1089.2] - [-184.8] = -904.4\ \text{kJ/mol}$$

The negative value for ΔH° shows that this is an exothermic reaction. The standard enthalpy of formation of elements in their most stable states is equal to zero. (Note the value of zero for $O_2(g)$ above).

Entropy

The thermodynamic quantity entropy (*S*) measures the amount of randomness or disorder in a system. Since entropy is a state function, the amount of entropy for a given amount of a substance is determined by variables such as temperature and pressure. If these variables are fixed, the amount of entropy will also be fixed. Typical units for entropy are joules/Kelvin _ mole (J/K _ mole).

The entropy change for a process, _*S*, is calculated the same way _*H* was calculated. The change in entropy is given by:

$$\_S = S_f - S_i$$

where S_f is the final entropy and S_i is the initial entropy. The *second law of thermodynamics* states that the total entropy of a system and its surroundings always increases for a spontaneous process. This means that entropy is quite distinct from energy. Energy cannot be created or destroyed during a chemical change, while entropy is created during a spontaneous process.

The experimental determination of entropy for a substance involves measurement of its heat capacity at different temperatures. This method finds its basis in the *third law of thermodynamics* which states that a perfectly crystalline substance at 0 K has an entropy of zero. As the temperature of a substance is increased, it absorbs heat and becomes more disordered. The increase in entropy is gradual, but large, sharp increases occur during phase changes. The standard, or absolute entropy (*S*°) is the entropy value for the standard state of a species, with units of J/mole _ K.

Standard entropies for the substances involved in a chemical reaction can be used to determine the change in entropy (_S°) for the reaction:

$$\_S^\circ = \_\ nS^\circ(\text{products}) - \_\ mS^\circ(\text{reactants})$$

where *n* and *m* refer to the stoichiometric coefficients from the balanced chemical equation. For the reaction:

$$4\ NH_3(g)\ +\ 5\ O_2(g)\ \_\ 6\ H_2O(g)\ +\ 4\ NO(g)$$

the relevant *S*° values (J/mol _ K) are: $NH_3(g)$, 192.5; $O_2(g)$, 205.0; $H_2O(g)$, 188.7; NO(g), 210.6. Substitution of these values into the above equation yields:

$$\_S^\circ = [6(188.7) + 4(210.6)]\ -\ [4(192.5) + 5(205.0)]\ =\ [1974.6]\ -\ [1795.0] = 179.6\ J/K$$

The positive value for _*S*° could have been predicted for this reaction as 9 total moles of gaseous reactants forms 10 total moles of gas in the products. The production of more moles of gas means that an increase in entropy is expected.

Gibbs Free Energy

In planning a chemical synthesis, a chemist wants to know if a proposed reaction will actually proceed to give products, that is, if the reaction is spontaneous. What is needed is a way to

determine reaction spontaniety. A quantity that provides a direct way to do this is the Gibbs free energy (G). The Gibbs energy is defined by the equation:

$$G = H - TS$$

where T is the temperature in Kelvin. Gibbs energy has units of kilojoules per mole (kJ/mole). As a chemical reaction proceeds at constant temperature and pressure, changes in both the enthalpy and entropy occur. The changes in H and S result in a change in the Gibbs energy as well, given by:

$$\Delta G = \Delta H - T\Delta S$$

If the reactants and products involved in a reaction are in their standard states, the standard free energy change can be calculated using the equation:

$$\Delta G^\circ = \Delta H^\circ - T\Delta S^\circ$$

For a given reaction we can look up the required ΔH_f° and S° values (if available) and calculate ΔH° and ΔS°. If ΔG° is negative, then the reaction is spontaneous. If ΔG° is positive, the reaction is nonspontaneous. If ΔG° equals 0, the reaction is at equilibrium.

The standard free energy of formation, ΔG_f°, for a substance is defined as the free energy change that occurs when 1 mole of a substance is formed from its elements in their most stable states at standard temperature and pressure. Standard free energy of formation values can be used to directly calculate ΔG° for a reaction using the equation:

$$\Delta G^\circ = \Sigma\, n\Delta G_f^\circ(\text{products}) - \Sigma\, m\Delta G_f^\circ(\text{reactants})$$

where n and m are the stoichiometric coefficients from the balanced chemical reaction. For the reaction we looked at previously:

$$4\ NH_3(g) + 5\ O_2(g) \rightarrow 6\ H_2O(g) + 4\ NO(g)$$

the relevant ΔG_f° values (kJ/mole) are: $NH_3(g)$, -16.7; $O_2(g)$, 0; $H_2O(g)$, -228.6; NO(g), 86.7. Substitution of these values into the above equation yields:

$$\Delta G^\circ = [6(-228.6) + 4(86.7)] - [4(-16.7) + 5(0)] = [-1024.8] - [-66.8] = -958.0\ \text{kJ/mol}$$

The negative value for ΔG° shows that this is a spontaneous reaction. Just as with standard enthalpies of formation, standard free energies of formation for elements in their most stable states are equal to zero. (Note the value of zero for $O_2(g)$ above).

Spontaneous chemical reactions can be harnessed to perform useful work. Think of our gasoline combustion example. The terminology "free" energy is used because the free energy change is the maximum energy available (free) to do useful work. In theory, if a reaction were carried out to obtain the maximum useful work (w_{max}), and no entropy were produced, then:

$$\Delta G = w_{max}$$

In reality the maximum amount of work is never obtained and some entropy is always created.

One of the most important reasons for calculating $\Delta G°$ for a reaction is in determining the extent of reaction as given by the value of the equilibrium constant K. Recall that for the generic reaction:

$$aA + bB \rightleftharpoons cC + dD$$

the equilibrium constant is given by:

$$K = [C]^c[D]^d \div [A]^a[B]^b$$

where the square brackets denote solution concentration units of molarity. If gases are involved in the reaction, the concentration is instead expressed using the partial pressure of the gas in units of atmospheres. Large values of K mean that the reaction favors products while small values indicate that reactants are favored. Values close to 1 mean that the equilibrium mixture will contain both reactants and products. Once $\Delta G°$ has been calculated, the value of K can be found using the relation:

$$\Delta G° = -RT \ln K$$

If a reaction occurs under nonstandard state conditions, we use the thermodynamic reaction quotient Q in place of K. The value for Q is obtained the same way as that for K. The relationship between $\Delta G°$, ΔG, and Q is given by:

$$\Delta G = \Delta G° + RT \ln Q$$

Applying the above relationships to our earlier reaction:

$$4\ NH_3(g)\ +\ 5\ O_2(g) \rightleftharpoons 6\ H_2O(g)\ +\ 4\ NO(g)$$

and solving for the value of K gives:

$$K = e^{(\Delta G°/-RT)} = e^{(386.855)} = 1.021 \times 10^{168}$$

The very large value of K tells us that this reaction greatly favors products and proceeds to completion as written.

Calculating Thermochemical Parameters

In the preceding discussion we have seen how useful the parameters $\Delta H_f°$, $S°$, and $\Delta G_f°$ are as they relate to chemical reactions. What if we are interested in reactions that include species that these quantities are not available for? What if we can find these values, but our proposed reaction conditions will be at some other temperature than 298 K? In these situations we can *calculate* a number of these parameters for various chemical species and use the results to learn

more about any reaction we are interested in. This section looks at some of the details of these calculations.

In previous chapters we have calculated single point energies, electron densities, infrared spectra, etc. All of these calculations were performed on single molecules in the gas phase at 0 K. The thermochemical parameters covered in this chapter result from the interactions of large collections of like molecules. In other words, we need to relate the results of a calculation performed on a single molecule to the macroscopic properties enthalpy, entropy, and Gibbs free energy. *Statistical mechanics* will help us make this connection.

In earlier chapters quantum mechanics and the Schrödinger equation were introduced. Quantum mechanics depends on the wavefunction _. Application of the appropriate operator to _ allows the calculation of the various properties of interest. For example, the Hamiltonian (H) applied to _ gives the energy (E) according to:

$$H\_ = E\_$$

Statistical mechanics depends on the *molecular partition function*, q (not to be confused with heat!). For a single molecule, q is the sum of exponential terms involving all possible quantum energy states $\__i$:

$$q = \sum_{i}^{\text{all states}} e^{-\varepsilon_i / k_B T}$$

where k_B is the Boltzmann constant and T is the temperature in K. Once the molecular partition function (q) is known, the partition function (Q) for N identical molecules can easily be found via:

$$Q = q^N / N!$$

Once the partition function Q is found, a number of thermochemical parameters and macroscopic observables can be calculated. The thermochemical parameters include H, S, and G as shown below:

$$H = U + PV = k_B T^2 \left(\frac{\partial \ln Q}{\partial T} \right)_V + k_B T V \left(\frac{\partial \ln Q}{\partial V} \right)_T$$

$$S = \frac{U - A}{T} = k_B T \left(\frac{\partial \ln Q}{\partial T} \right)_V + k_B \ln Q$$

$$G = H - TS = k_B T V \left(\frac{\partial \ln Q}{\partial V} \right)_T - k_B T \ln Q$$

In order to find Q, we first need to determine the molecular partition function, q. To get q we need to know all the possible quantum energy states. The total molecular energy ($\__{tot}$) can be approximated as a sum of various contributions. These include translational, rotational, and vibrational motions as well as electronic energy levels:

$$\varepsilon_{tot} = \varepsilon_{trans} + \varepsilon_{rot} + \varepsilon_{vib} + \varepsilon_{elec}$$

The total molecular partition function (q_{tot}) then becomes a product of terms:

$$q_{tot} = q_{trans} q_{rot} q_{vib} q_{elec}$$

The total enthalpy (H_{tot}) and total entropy (S_{tot}) involve ln(q) and can be expressed as a sum of the various contributions:

$$H_{tot} = H_{trans} + H_{rot} + H_{vib} + H_{elec}$$
$$S_{tot} = S_{trans} + S_{rot} + S_{vib} + S_{elec}$$

To determine thermochemical parameters for a molecule, the vibrational frequencies (*i.e.* the infrared spectral results) are first calculated. As discussed in Chapter 13, a bad starting geometry will give bad vibrational frequencies. This will result in incorrect thermochemical parameters, so the geometry must first be optimized, then the vibrational frequencies should be calculated *using the same level of theory*. Once the vibrational frequencies have been determined, most computational chemistry programs will calculate the various thermochemical parameters since only a small amount of CPU time is necessary.

The translational component of the molecular partition function (q_{trans}) is given by:

$$q_{trans} = \left(\frac{2\pi M k_B T}{h^2} \right)^{3/2} V$$

where M is the molecular weight of the molecule, V is the molar gas volume, and h is Planck's constant. The rotational component (q_{rot}) is given by the equation:

$$q_{rot} = \frac{\sqrt{\pi}}{\sigma} \left(\frac{8\pi^2 k_B T}{h^2} \right)^{3/2} \sqrt{I_1 I_2 I_3}$$

where _ is a number related to the symmetry of the molecule and I_i are the moments of inertia for the molecule. The vibrational component (q_{vib}) is a product of partition functions for each vibration (_$_i$), given by:

$$q_{vib} = \prod_{i=1}^{3n-6} \frac{\exp\left(-h\nu_i / 2k_B T \right)}{1 - \exp\left(-h\nu_i / k_B T \right)}$$

where *3n-6* is the total number of vibrations for a nonlinear molecule composed of *n* atoms (See Chapter 13). The final component of the molecular partition function is the electronic portion (q_{elec}). Since excited electronic states typically lie much higher in energy than the ground state, only the ground state need be considered. This means:

$$q_{elec} = 1$$

In comparing the energies of individual molecules (calculated at 0 K with fixed nuclei) to experimental results (typically gathered at ~298 K with vibrating nuclei), two corrections are required: (1) the zero point energy (ε_0), and (2) the thermal energy correction.

At 0 K a molecule will still have vibrational energy. This energy is known as the zero point energy (ε_0) and can be calculated by summing the energy contributed by each vibrational mode (ν_i):

$$\varepsilon_0 = H_{vib}(0) = \frac{1}{2} h \sum_{\text{i}}^{\text{normal modes}} \nu_{\text{i}}$$

Most computational chemistry programs automatically calculate the zero point energy and add it to the reported total energy. The program output will typically list all of the contributions to the total energy and will also specify the value of ε_0.

As a molecule is heated from 0 K, it gains translational, rotational, and vibrational energy. The amount of energy gained depends on the temperature of interest. At some temperature (T) above 0 K, the change in enthalpy is given by:

$$\Delta H(T) = H_{trans}(T) + H_{rot}(T) + \Delta H_{vib}(T) + RT$$

where R is the ideal gas constant. The translational and rotational components are easily calculated:

$$H_{trans}(T) = \frac{3}{2} RT \quad \mathrm{H_{rot}}(T) = \frac{3}{2} RT \quad (= RT \text{ if linear})$$

The vibrational component again involves a summation over all normal modes (ν_i), taking the temperature of interest into account:

$$\Delta H_{vib}(T) = H_{vib}(T) - H_{vib}(0) = Nh \sum_{\text{i}}^{\text{normal modes}} \frac{\nu_i}{e^{h\nu_i / k_B T} - 1}$$

The output obtained will depend on the particular program and method used to do the calculation. As discussed in Chapter 6, the energies reported from a molecular mechanics calculation are not externally referenced, and will therefore not be of use in investigating the energetics of chemical reactions.

The semiempirical methods (Chapter 8) MINDO/3, AM1, and PM3 available in MOPAC (See Chapter 20) have been parameterized to give reasonable energy values. The program will report enthalpy and entropy values at a variety of temperatures (200 – 400K), including individual partition function values for vibration, translation, and rotation. The most useful number reported is the ΔH_f° value. This value can be calculated for all reactants and products, and ΔH° for the reaction can be calculated as demonstrated above. The ΔH° values obtained in this way are semi-quantitative at best (See example below).

Hartree-Fock and Density Functional Theory methods (Chapters 7 & 9, respectively), using either the Gaussian or GAMESS programs (Chapters 18 & 19) report enthalpy, entropy, and free energy values with individual partition function values for vibration, translation, and rotation as the result of a vibrational frequency calculation. The energy values reported by these programs are not externally referenced, that is, they cannot be directly compared to experimental values. The *relative* energy values, however, are useful and can be used to determine $\Delta H°$ for reactions of interest. Excellent results can be obtained for *isodesmic* reactions, where the number of each type of chemical bond is the same in both reactants and products. The similarity in bond types leads to a cancellation of errors. An example of such a reaction is:

$$CO_2 + CH_4 \rightarrow 2\,H_2CO$$

The reaction has an experimentally determined $\Delta H° = 59.9 \pm 0.2$ kcal/mole. The table below shows representative calculated values:

Method	Result (kcal/mole)
AM1	25.7
PM3	29.9
B88-LYP	58.4

Chapter 16

Applications of Molecular Modeling to Other Disciplines

16.1 Key Notes

16.1.1 Applying molecular modeling to other disciplines

The use of molecular modeling/computational chemistry is not restricted to researchers and students who are solely interested in studying traditional chemistry topics, such as molecular structure, kinetics, reaction mechanisms, and thermodynamics. To a large extent, molecular modeling is becoming an increasingly important tool to researchers in other scientific disciplines, such as materials science, the environmental sciences, life sciences, and medicine. This chapter presents a number of examples of how computational chemistry technologies, techniques, and tools are applied to disciplines related to but outside the traditional definition of pure and applied chemistry.

16.1.2 Materials Science

Materials science is a relatively new discipline that looks to apply a variety of disciplines – chemistry, physics, biology, mathematics, computing, and others – to the characterization of compounds that have usefulness in society. Materials science includes sub-domains such as nanotechnology, biopolymers, and crystallography. Products developed by material scientists have a wide variety of uses in all areas of everyday life. Computational chemistry is an integral part of this new discipline, one that is found in many research labs and universities. Examples in this chapter include the delineation of quantum dots (QDs) as potential semiconductors and an investigation of how water can be embedded between graphite layers to increase the absortive and adhesive characteristics of the compound.

16.1.3 Environmental Science

Modeling in the environmental sciences is multi-faceted, with many of the models being in the category of *dynamic* models, those that calculate the change of one or more variables as a function of time. These models, built on the mathematics of differential equations, allow environmental

scientists and researchers to project and predict the consequences of some action (such as burning of fossil fuels) on some consequence (such as global warming). Computational chemistry also plays a role in the environmental sciences, allowing researchers to investigate chemical issues such as reaction mechanics and kinetics, thermodynamics, and other structure-property-activity conditions of environmentally-important compounds. Examples in this chapter include examples from the atmospheric sciences, a paper on NO_x compounds and a paper on transition states of compounds that play a role in photochemical smog production.

16.1.4 Life Sciences

Computational chemistry plays a significant role in the life sciences, especially given the developments in the areas of molecular biology, genomics, and proteomics, among others. The ability to study important biological compounds computationally allows the life scientist to better understand the role of specific molecules in the biological process. As the name suggests, molecular biology research is particularly supported by the contributions of the computational chemist, and molecular biology research often includes research scientists who are specifically trained in computational chemistry.

16.1.5 Medicine

Medicine, particularly pharmacology and the sub-discipline of *medicinal chemistry*, is another area that takes significant advantage of computational chemistry. In what is now known as *rational drug design*, computational chemistry is probably the *central* tool for the chemist looking to develop a new pharmaceutical compound. Using techniques such as *quantitative structure-activity relationships* (QSAR), researchers can eliminate a large number of compounds that will not succeed as pharmaceuticals prior to synthesizing them in the laboratory and, more importantly, subjecting animals to experimental drug research.

16.2 Applying molecular modeling to scientific problems

A significant portion of this *Guide* is directed towards educating the reader on how computational chemistry is done, with an emphasis on mathematics, methods, and tools that are used by the practicing computational chemist. Some portions of this *Guide* look at the types of calculations that can be done, with some description of the types of problems that those calculations are designed to solve. This chapter, however, looks to present some examples of real-world problems that researchers are studying, with the help of the technologies, techniques, and tools of computational chemistry.

As was discussed in previous chapters, some researchers focus their attention on research that will improve the way computational chemistry is done, what was described as *(computational chemistry) research*. Indeed, this *Guide* spends most of its pages describing how one goes about *doing* computational chemistry. The majority of researchers, however, see computational chemistry as one of many *tools* and *approaches* that can be used to solve interesting problems. Certainly many of these problems are in the realm of *pure and applied chemistry*, but an increasing number of problems outside of the traditional domain of chemistry are being subjected to a computational approach. The terminology of *computational (science research)*, in which computation is used as a means to the end of scientific research, is quickly becoming standard operating procedures in many research labs in both the private sector and in academia.

This chapter presents several areas that make significant use of computational chemistry as a research tool and methodology. These examples are only the tip of the iceberg, but should serve to help the reader appreciate how this technology is used to find solutions to interesting and complex problems. The examples in each of the following sections come from independent research projects done by junior and senior high school students at the North Carolina School of Science and Mathematics (NCSSM) in Durham, NC. These projects were completed as the final project for the class "Introduction to Computational Chemistry".

16.3 Materials Science

Materials science is a broad interdisciplinary field that spans a multitude of sciences. Wikipedia provides a concise yet thorough synopsis of this relatively new scientific discipline:

> Materials science or materials engineering is an interdisciplinary field involving the properties of matter and its applications to various areas of science and engineering. This science investigates the relationship between the structure of materials at atomic or molecular scale and their macroscopic properties. It includes elements of applied physics and chemistry, as well as chemical, mechanical, civil and electrical engineering. With significant media attention to nanoscience and nanotechnology in recent years, materials science has been propelled to the forefront at many universities. It is also an important part of forensic engineering and forensic materials engineering, the study of failed products and components.
> Source: *http://en.wikipedia.org/wiki/Materials_science*)

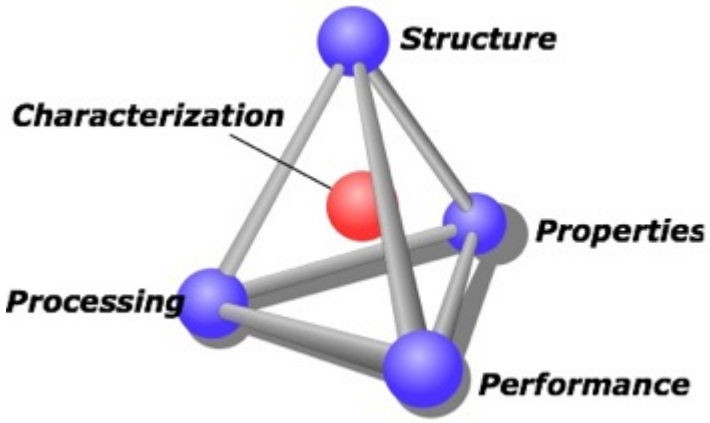


Figure 16.1: Visual representation of the materials science discipline

The graphic from Wikipedia, shown in Figure 1, shows strong similarities with one of the foundational premises of this *Guide*, the emphasis on the structure-properties-activities paradigm. The field of materials science is quite broad, with a variety of sub-categories, including nanotechnology, crystallography, microelectronics, and biomaterials. Foundational disciplines used by materials scientists include quantum mechanics, thermodynamics, kinetics, and other traditional chemistry

areas. Topics from physics, such as mechanics and solid-state physics, are also strong components of materials science. It cannot be emphasized enough the *interdisciplinary* nature of this research field. Most major universities now have departments or even schools of materials science, and the proliferation of peer-reviewed journals (including, by way of example, a journal entitled *Computational Materials Science*) is growing at a staggering rate.

To illustrate the use of computational chemistry in the materials science, we present two projects done by NCSSM computational chemistry research students. In the first project, Sam Powers (now at MIT), was interested in the materials science properties of *quantum dots (QDs)*, which act as semiconductors and have significant potential to improve the efficiency and effectiveness of transistors, LEDs, and other optical/imaging devices. Sam's research work looked to determine an optimal structure for a quantum dot compound – in this case, a cadmium selenide (CdSe) compound – by characterizing the structure and then trying to find an optimal geometry for this structure. As one can see from the abstract (Figure 2), getting this structure to optimize was difficult, requiring more computational power than she had access to for this particular project, a reality not understood until she ran her calculations.

Figure 3 shows the structure of the CdSe compound. This compound also contains a number of phosphorus compounds. The goal of this computational work, once the molecule was structurally optimized, was to see if it could be determined how electrons migrate through the compound, contributing to its ability to serve as a semiconductor. The molecule will fluoresce under proper conditions, and It should be noted that student also synthesized this compound in the lab and was able to get it to do just that. The ultimate goal of this overall work, the topics that first attracted this student to this project, was to see if quantum yields and quantum efficiencies could be determined experimentally and computationally. As a side note, the student hopes to continue this project upon matriculation to MIT in the fall of 2008.

Two other students, Makani Dollinger and Lindsay Koloff (now both at UNC Chapel Hill), looked to determine the adhesive properties of a water molecule inserted between graphite ($C_{16}H_{10}$) bilayers, an important consideration in the field of nanotechnology. Their abstract is as follows:

Abstract

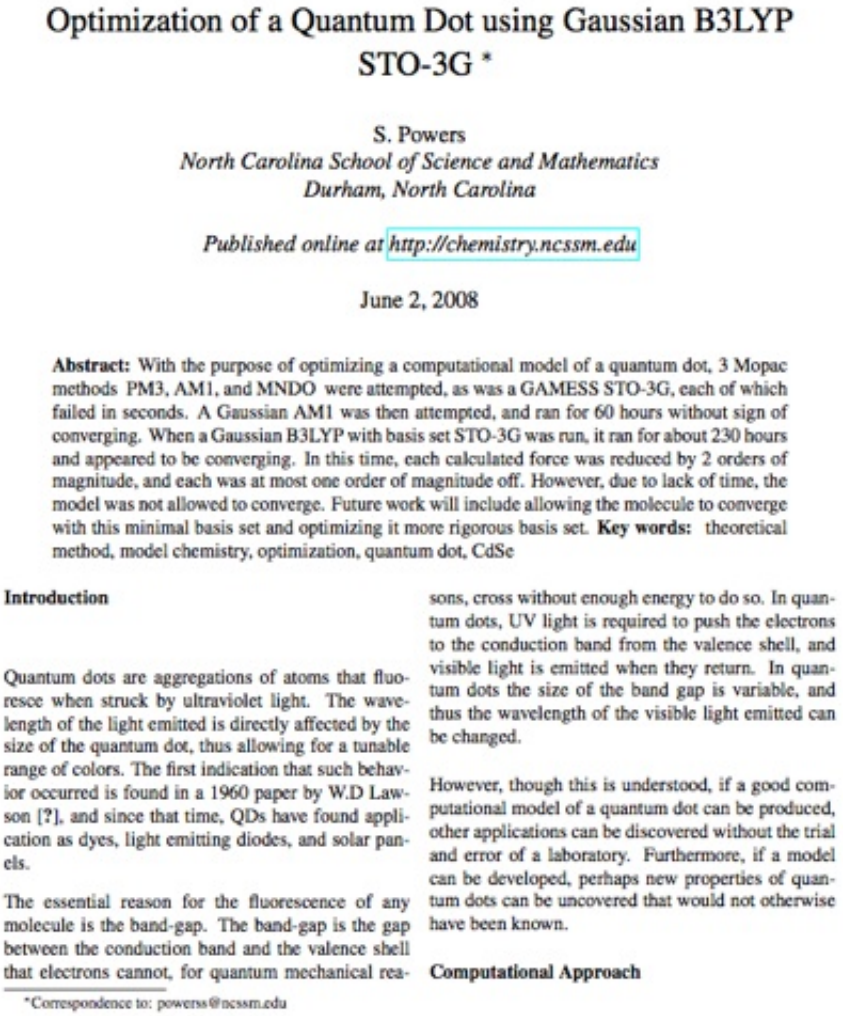
Interactions between a water molecule and one- and two-layer graphite were calculated by ab initio B3LYP method. The eclipsed pyrene dimer molecule $(C_{16}H_{10})_2$ was used as the two-layer graphite model, to study the effects on a water molecule between two graphite layers. A one-layer graphite model ($C_{16}H_{10}$) was used as a reference to investigate the influence on the molecular orbitals and adhesion sites. HOMO and LUMO properties were analyzed for adsorption properties.

Key Words: graphite, molecular orbitals, HOMO, LUMO, adsorption

Their compound, shown in Figure 4, was optimized with the B3LYP/3-21G *model chemistry*. Electrostatic potential maps were then calculated, looking to identify the optimal orientation for a water molecule "sandwiched" between two layers of graphite. The students were particularly interested in determining the orientation of the hydrogen atoms in relation to the two graphite bilayers.

16.4 Environmental Science

The environmental sciences are another area where computation plays a central and significant role. Much of the modeling done in the environmental sciences consists of dynamic models, those that



Optimization of a Quantum Dot using Gaussian B3LYP STO-3G *

S. Powers
North Carolina School of Science and Mathematics
Durham, North Carolina

Published online at http://chemistry.ncssm.edu

June 2, 2008

Abstract: With the purpose of optimizing a computational model of a quantum dot, 3 Mopac methods PM3, AM1, and MNDO were attempted, as was a GAMESS STO-3G, each of which failed in seconds. A Gaussian AM1 was then attempted, and ran for 60 hours without sign of converging. When a Gaussian B3LYP with basis set STO-3G was run, it ran for about 230 hours and appeared to be converging. In this time, each calculated force was reduced by 2 orders of magnitude, and each was at most one order of magnitude off. However, due to lack of time, the model was not allowed to converge. Future work will include allowing the molecule to converge with this minimal basis set and optimizing it more rigorous basis set. **Key words:** theoretical method, model chemistry, optimization, quantum dot, CdSe

Introduction

Quantum dots are aggregations of atoms that fluoresce when struck by ultraviolet light. The wavelength of the light emitted is directly affected by the size of the quantum dot, thus allowing for a tunable range of colors. The first indication that such behavior occurred is found in a 1960 paper by W.D Lawson [?], and since that time, QDs have found application as dyes, light emitting diodes, and solar panels.

The essential reason for the fluorescence of any molecule is the band-gap. The band-gap is the gap between the conduction band and the valence shell that electrons cannot, for quantum mechanical reasons, cross without enough energy to do so. In quantum dots, UV light is required to push the electrons to the conduction band from the valence shell, and visible light is emitted when they return. In quantum dots the size of the band gap is variable, and thus the wavelength of the visible light emitted can be changed.

However, though this is understood, if a good computational model of a quantum dot can be produced, other applications can be discovered without the trial and error of a laboratory. Furthermore, if a model can be developed, perhaps new properties of quantum dots can be uncovered that would not otherwise have been known.

Computational Approach

*Correspondence to: powerss@ncssm.edu

Figure 16.2: Screenshot of Sam Powers' paper on quantum dots

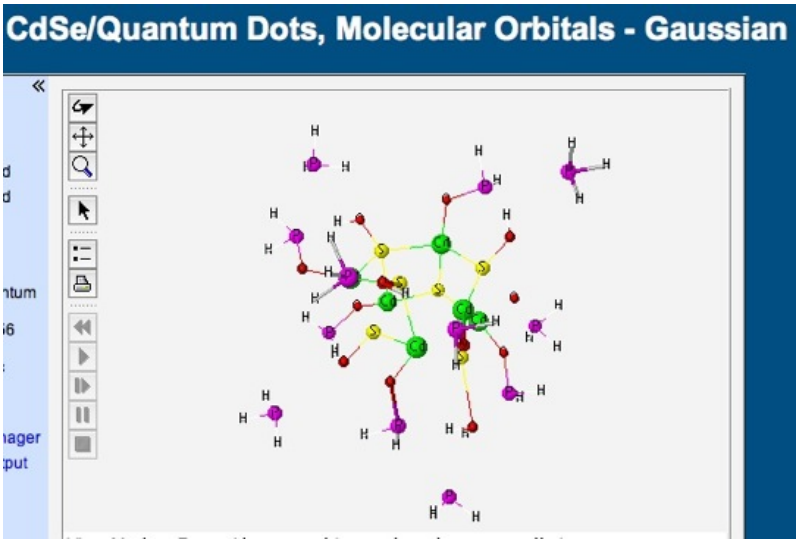


Figure 16.3: Cadmium selenide quantum dots compound

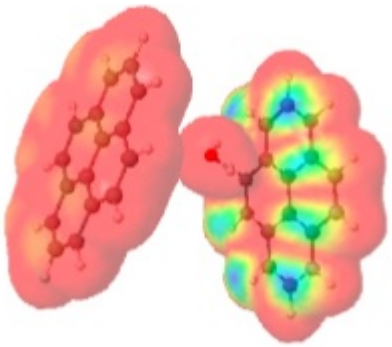


Figure 16.4: Electrostatic potential map for graphite bilayers with water molecule in the middle

are concerned with the change of some phenomena over time, represented mathematically by the differential equation $\Delta X/\Delta t$, or dX/dt, where X is some measurable (such as the concentration of a pollutant in a body of water) and t represents time. *Global climate change models* and *numerical weather prediction (NWP)* models are well known examples of these types of environmental science models.

Computational chemistry technologies, techniques, and tools are also prevalent in the environmental sciences. The following two papers from student projects demonstrates how computation is applied to the environmental sciences. The abstract for the first paper is shown in Figure 5, and addresses how nitrogen-containing compounds (such as NO_x, including NO_2) contribute to pollution in tropospheric chemistry. This paper applied *molecular orbital theory*, including the determination of the HOMO/LUMO gap, as a way to evaluate and compare a variety of NO_x compounds.

Reactive Activities of Nitrogen-Containing Pollutants in the Environment

T. Hicks, D. Britt, & C. Nance
North Carolina School of Science and Mathematics, Durham, NC

June 5, 2007
Published online on Comp Chem Moodle (moodle.ncssm.edu)

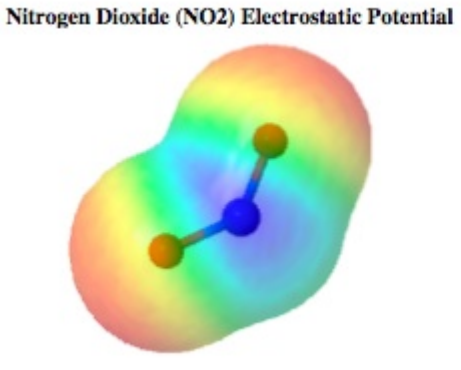
Abstract: We often use computational chemistry methods to solve real world problems. One of the many problems we face today is global warming and air pollution. We decided to look at different nitrogen-containing compounds that are known to have harmful effects on the environment and determine why some are more harmful than others, specifically nitrogen dioxide (NO2). Using the molecular editor builder of WebMO on the North Carolina High School Computational Chemistry Server allowed us to have a greater understanding of why NO2 is more harmful and reactive in the environment. Using Gaussian software package at the Hartree-Fock level with a 3-21G basis set we were able to retrieve useful data relating to the EHOMO and ELUMO correlation between the several nitrogen molecules being tested. We began researching each compound by running geometry optimization calculations followed by molecular orbital calculations. We used the molecular orbital calculations to retrieve the HOMO-LUMO gap and to compare the different molecules, which allowed us to draw reasonable conclusions. In the end it became clear that NO2 was an outlier when compared to the other nitrogen molecules. This difference could be a major factor in how it reacts in the environment when compared to the others.

Key words: Nitrogen Dioxide, Gaussian, Basis Set, EHOMO, ELUMO, HOMO-LUMO Gap

Figure 16.5: Abstract from student paper in the environmental sciences

Figure 6 shows a snapshot of a portion of the student's journal article. All student journal articles at NCSSM are required to be written following the editorial guidelines of the *Journal of Computational Chemistry* (Wiley). In this research, the students created HOMO/LUMO molecular orbital graphics and electrostatic potential maps to evaluate the different compounds, and applied a number of statistical techniques to their numerical results. In the graphic in Figure 6, we see both an example of an electrostatic potential map and a table containing HOMO/LUMO gap results, calculated using the formula:

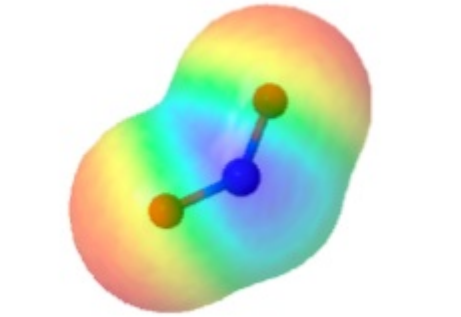
$$E_{HOMO/LUMOgap} = E_{HOMO} - E_{LUMO} \tag{16.1}$$



Molecules	CPU Time (secs)	EHOMO (Hartrees)	ELUMO (Hartrees)	HOMO-LUMO Gap
NO	10.5	-0.44079	0.13111	0.5719
NO_2	10.5	-0.4892	0.05264	0.54184
NO_3^{-1}	10.6	-0.19062	0.36486	0.55548
N_2O_5	10.7	-0.5077	0.0249	0.5326

We also used Logger Pro Version 3.4.1 to graph the number of atoms in each compound vs. the EHOMO-ELUMO gap of each compound. This data resulted in a scatter plot displayed below:

Figure 16.6: Sample results section from MO theory analysis of NO_x compounds



Molecules	CPU Time (secs)	EHOMO (Hartrees)	ELUMO (Hartrees)	HOMO-LUMO Gap
NO	10.5	-0.44079	0.13111	0.5719
NO_2	10.5	-0.4892	0.05264	0.54184
NO_3^{-1}	10.6	-0.19062	0.36486	0.55548
N_2O_5	10.7	-0.5077	0.0249	0.5326

We also used Logger Pro Version 3.4.1 to graph the number of atoms in each compound vs. the EHOMO-ELUMO gap of each compound. This data resulted in a scatter plot displayed below:

Figure 16.7: Sample results section from MO theory analysis of NO_x compounds

In the second paper, written by Khan, Gromlich, and Somervell, transition state methods are applied to the atmospheric chemistry of photochemical smog. By applying transition state calculations to the atmospheric degradation of a variety of compounds, the paper looked to determine if some of those chain reaction mechanisms might be susceptible to manipulation by external means. Figure 8 shows a screenshot of the abstract for this paper.

Figure 9 shows a section of the results for the transition state calculations on photochemical smog.

Transition States of Reactions Involved in Photochemical Smog Creation

Monis Khan, Matt Gromlich, Lenora Somervell
North Carolina School of Science and Mathematics, Durham, NC

Received 1 June, 2008; Accepted 2 June, 2008
Published online on Comp Chem Moodle (moodle.ncssm.edu)

Abstract: Atmospheric reactions are commonly-occuring processes which often lead to tropospheric air pollution. These atmospheric reactions often create free radicals which propagate rapidly, creating more free radical molecules. Transition states, which are only found through computational methods, are useful in learning the kinetics of these atmospheric reactions and free radicals. Once the kinetics for the reaction are found, there is a possibility of developing processes that stop transition states from forming. If such a possibility existed, harmful atmospheric reactions could be eliminated altogether. To determine these transition states, semi-empirical MOPAC/PM3 saddle calculations, transition state optimizations, and vibrational frequency calculations were performed for each reactant and product in three chosen atmospheric degradation equations. WebMO was used as an interface to build the molecules involved in the atmospheric reactions. These calculations provided information on the heat of formation values (H_f) for each molecule. A comparison was made and the H_f of each transition state was found to be equivalent to the H_f of each product. The equivalence of these values depicts the rapid rate of the reactions in the atmosphere.

Key words: Transition state, ozone, kinetics, heat of formation, free radical

Figure 16.8: Transition states of photochemical smog

In the C_3H_6OH transition state, the C-C and C-O bond lengths were 0.8 angstroms and 0.09 angstroms less than those in C_3H_6OH. The C-H bond lengths were 0.02 angstroms greater than those in C_3H_6OH. The C-C-C, C-C-O, and some of the C-C-H bonds were larger than those in C_3H_6OH. The C-C-H bonds ranged from 108.51 – 120.06 degrees.



	C3H6OH	C3H6OH TS	C3H6OH TS
C-C bond (A)	1.54	1.46	1.48
C-O bond (A)	1.5	1.41	
C-H bond (A)	1.09	1.09	1.11
O-H bond (A)	1.05	0.95	
C-C-C angle °	109.47	119.32	
C-C-O angle °	109.47	116.13	
C-C-H angle °	109.47	108.51	112.12
	H2O2	H2O2 TS	
O-O bond (A)	1.46	1.47	
O-H bond (A)	1.05	0.94	
O-O-H angle °	109.47	100.67	

Conclusions

For most reactions, the transition state heat of formation value is either at a maximum or a

Figure 16.9: Sample results of transition state calculations on smog formation

16.5 Life Sciences

Computational chemistry is found often in the life sciences domain, particularly in the areas of molecular biology, genomics, proteomics, and biochemistry. Increasingly, the artificial line between chemistry and biology is being eroded, and the modern biologist must be well versed in the chemical sciences, and vice versa.

In their computational science research paper, Newsome and Jones performed a computational analysis of the structure and properties of *cytochrome c oxidase*, an enzyme of significant biochemical importance. Figure 10 shows the paper abstract and keywords. In this paper, the researchers used molecular orbital theory to determine similarities and differences between five *cytochrome c oxidase* inhibitors. By analysis of the *frontier orbitals*, particulary the LUMO, the researchers determined that there were potentially measurable differences in the potencies of the various inhibitors based on the value of the LUMO.

Figure 11 shows a screenshot of the *cytochrome c oxidase* protein. Note that this is a large structure, and the student researchers focused their research work on the efffect of various compounds, including azide, formaldehyde, cyanide, hydrogen sulfide and carbon monoxide as potential *cytochrome c oxidase* inhibitors (CCOIs). These molecules are small molecules, and hence very acceptable for performing *ab initio* calculations.

Comparison of Reactive Molecular Orbitals Found in Cytochrome C Oxidase Inhibitors

Alex Newsome and Michael Jones
North Carolina School of Science and Mathematics, Durham, NC

Received 2 June, 2008

Abstract: A category of very potent poisonous drugs is the Cytochrome C Oxidase Inhibitor family. These drugs target a vital protein in cells of organisms, and interact with the component in ways that prevent it from being used in biological processes. Once this protein has been disabled, the body is unable to function and death will inevitably occur. These Oxidase inhibitors vary in structure, but have common trends in their molecular make-up that allow them to continue their function as highly electronegative disrupters in the mitochondria. The purpose of this experiment is to observe five Cytochrome C Oxidase Inhibitors and draw conclusions based on correlations between their potency in the body and their reactive molecular orbitals. The hypothesis is that the LUMO of these molecules will show a strong correlation between their energy values and potency. After calculating these molecular orbitals and other attributes of the molecules using HF/6-311*G(d,p)// HF/6-311*G(d,p), it was found that the LUMO did indeed provide strong evidence in determining the potency of the compounds, proving the hypothesis. It was also noted that the HOMO and LUMO provided more accurate correlations together than either did when separate. It was also confirmed that there was no accurate trend between the electrostatic potential or dipoles of the molecules and their potency.

Keywords: Computational Chemistry, Cytochrome C Oxidase, Electron Transport Chain, HOMO, LUMO, Cytochrome C Oxidase Inhibitors, LD_{50}, QSAR

Figure 16.10: Abstract from student paper in the life sciences

Figure 1.1: Structure of Cytochrome C Oxidase

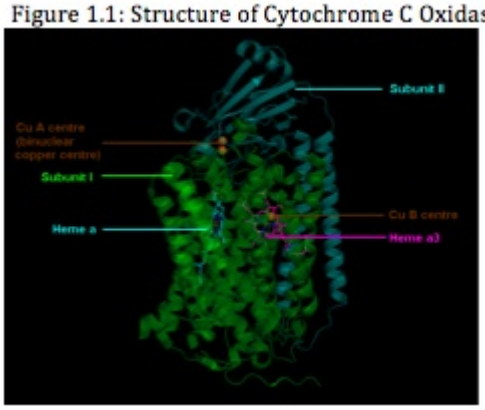


Figure 16.11: Screenshot of the *cytochrome c oxidase* protein

16.6 Medicine

One of the biggest disciplines for application of computational methods is in the medical field, especially in the pharmaceutical sciences, such as medicinal chemistry. One of the most important techniques in medicinal chemistry is that of *quantitative structure-activity relationships*, or QSAR. In QSAR, the research attempts to quantify some biological activity, such as the ability of a compound to have a medically therapeutic effect, on one or more measurable structural characteristics. QSAR researchers will develop a test set of compounds that have some relationship to the target activity, and attempt to develop a linear regression equation that can then be used predict the potential that a new compound will have similar or better results for biological activity. Figure 12 shows a sample test dataset, where π, σ, and $E_s(meta)$ are all structural characteristics that can be measured in the laboratory and/or computationally. The biological activity is shown as log(1/C), where C is the minimum concentration needed to cause a therapeutic effect. Equation 16.2 shows the QSAR linear regression representation of this dataset.

Once the researcher has the QSAR equation, s/he can then look to find those compounds that have the optimal values of the structural characteristics (in this case, π, σ, and $E_s(metat)$) that will result in a specific value for the biological activity (log(1/C). For example, if the target for log(1/C) was 10.250, and using the equation in Equation 16.2, the researcher would need to develop a compound that had the structural characteristics of $\pi = 1.446$, $\sigma = 1.160$, and $E_s(meta) = 0.00$. A compound with these characteristics would likely have a biological activity that is at or near the target set at 10.25.

						Calc.	Calc.
m-X	*p*-Y	π	σ+	$E_{s(meta)}$	$\log(1/C)_{obs}$	$\log(1/C)_a$	$\log(1/C)_b$
H	H	0.00	0.00	1.24	7.46	7.82	7.88
F	H	0.13	0.35	0.78	7.52	7.45	7.43
H	F	0.15	-0.07	1.24	8.16	8.09	8.17
Cl	H	0.76	0.40	0.27	8.16	8.11	8.05
Cl	F	0.91	0.33	0.27	8.19	8.38	8.34
Br	H	0.94	0.41	0.08	8.30	8.30	8.22
I	H	1.15	0.36	-0.16	8.40	8.61	8.51
Me	H	0.51	-0.07	0.00	8.46	8.51	8.36
Br	F	1.09	0.34	0.08	8.57	8.57	8.51
H	Cl	0.70	0.11	1.24	8.68	8.46	8.60
Me	F	0.66	-0.14	0.00	8.82	8.78	8.65

Figure 16.12: Sample QSAR test dataset

$$log(1/C) = 1.259\pi - 1.460\sigma + +0.208E_s(meta) + 7.619 \tag{16.2}$$

A QSAR project by Gold, Nutz, and Ravindranatha looked at trying to determine which computational method would be the best predictor of heats of formation (ΔH_f) for various components

of the aspirin (acetylsalicylic acid) reaction mechanism. In this paper, "best" was described as most closely approximating the experimental values found in the pharmacology laboratory. The overall goal of this research as to measure the accuracy and computational effectiveness of various computational methods on medicinal compounds. Figure 13 shows a sample of the results found by the student researchers in this project.

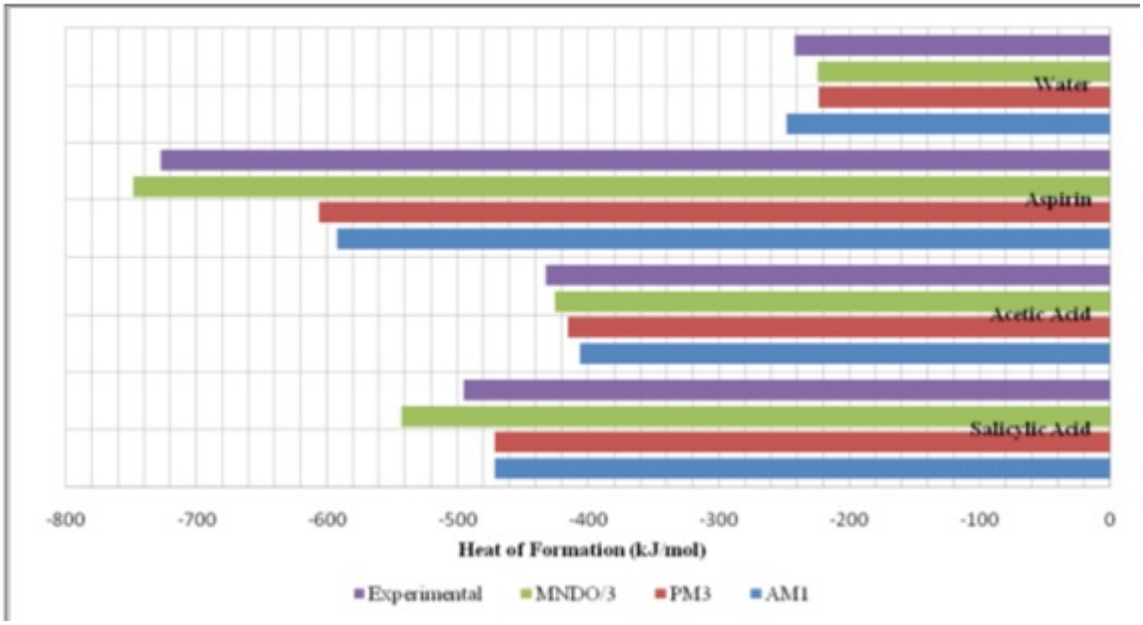


Figure 16.13: Comparison of ΔH_f of Water, Aspirin, Acetic Acid, and Salicylic Acid with MNDO/3, PM3, and AM1 with Experimental Values

Chapter 18: Gaussian

Key Notes:

Gaussian Basics:

Gaussian is a very high-end quantum chemical software package, available commercially through Gaussian, Inc. The software runs on virtually all computer platforms, including Microsoft Windows, Macintosh OS, and all variants of Unix. In addition, it can be accessed through Web-based interface tools such as WebMO. Gaussian is the most powerful software available to educators and student researchers through the North Carolina High School Computational Chemistry server. Currently, Gaussian03 (G03) is available. The "03" refers to the year – 2003 – in which the software was published. G03 is the most recent version.

Running Gaussian Jobs:

In most molecular modeling software programs, a submitted calculation is known as a **job**. Gaussian jobs are simple to run using the WebMO interface. Once you have built the molecule, and you have chosen Gaussian as the computational engine, the WebMO interface provides you with a series of options via pull-down menus. The Shodor computational chemistry staff has customized the pull-down menus to provide a reasonable, but not overwhelming, number of options for the educator and student researcher. Using the Advanced tab, further customization can be made by the user using a variety of keyword options.

Gaussian Keywords:

Gaussian has a long list of keywords, which are additional options and extensions to a calculation. For example, if we are trying to determine where electrons are distributed throughout a molecule (a computation known as a population analysis), we can specify which specific type of population analysis we might wish to run. Without a keyword choice (entered using the **Advanced** tab in the WebMO Job configuration window), the software defaults to (automatically chooses) a Mulliken population analysis.

Interpreting Gaussian Output:

One of the advantages of the WebMO interface that links the user to Gaussian is that almost all of the important results that come from running a Gaussian calculation are automatically displayed in the **View Calculated Quantities** window in WebMO. It is possible, however, to view the entire text-based output file that is generated with a Gaussian calculation. A short tutorial on how to read this output is included later in this chapter.

Troubleshooting Gaussian Jobs:

Most Gaussian jobs will run to completion successfully and will not fail, assuming that the molecule being calculated is one that is reasonable in size and structure. Occasionally, a job will fail for no apparent reason, and can simply be restarted. Other troubleshooting tips are described in greater detail later in this chapter.

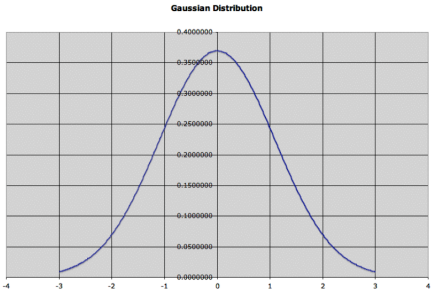
Gaussian Support Tools:

There are a number of support software programs that help the Gaussian user to get input and output files to and from the Gaussian software program. With the North Carolina High School Computational Chemistry server, the WebMO program, a Web-based interface, provides the user with the majority of those helper tools. These include a Java-based molecular editor for building molecules and a separate Java-based program entitled "MOViewer". MOViewer automatically loads when the user selects the magnifying glass option for molecular orbital calculations. In the professional community, increasing numbers of researchers are using the WebMO interface. Other programs, such as GaussView, provide very much the same capabilities, but are expensive. Other graphical interfaces for Gaussian include PC Model, Chem3D, and CAChe.

Gaussian Basics:

Gaussian, a commercial quantum chemical software package from Gaussian, Inc., is considered by many (including the authors of this resource) to be the industry standard in the area of molecular modeling and computational chemistry. Gaussian is capable of running all of the major methods in molecular modeling, including molecular mechanics; *ab initio*; semi-empirical; and density functional theory (DFT). It is probably best known for its robustness in running *ab initio* and DFT calculations. Gaussian also does several *compound methods* such as MPx and Gx, where x is a number that indicates the level of the method. For example, there is MP2 and MP4, defined as Moller-Plesset 2nd order and Moller-Plesset 4th order.

The name Gaussian comes from the use of the Gaussian Type Orbitals that Gaussian's originator, John Pople, used to try to overcome the computational difficulties that arose from the use of Slater Type Orbitals (for a discussion of these, see the chapter on Basis Sets). Most readers will know the Gaussian mathematics by two other names – a normal distribution, or perhaps as a *bell-shaped curve.* A number of researchers, such as S.F. Boys and Isaiah Shavitt, Pople, quite brilliantly, recognized that the (relatively) simple substitution of a series of Gaussian functions for the Slater function would greatly simplify the rest of the calculation of the Schrödinger equation. Pople's work resulted in the standard use of these Gaussian functions. Virtually every other developer of *ab initio* computational chemistry software uses this technique. Pople, by the way, was awarded the 1998 Nobel Prize in Chemistry (along with Walter Kohn, who you will hear about later) for this work.



Running Gaussian Jobs:

Like almost all molecular modeling packages, the basic concept of running a calculation is known as a **job**. Gaussian jobs are submitted by sending a text-based input file to the Gaussian processor, with results returned as one or more text-based output files. With the WebMO interface, the actual input file is not seen by the user, but is generated through the WebMO interface. As an example, in WebMO, we created the formaldehyde molecule and requested a molecular orbital calculation using the HF/STO-3G model chemistry (Hartree-Fock method with a STO-3G basis set). When we requested a molecular orbital calculation, the command POP=FULL was added. This asks that all of the calculated molecular orbitals be printed out in "full" in the output file. The 6D and 10F notation gives the software a few more specifics about the orbitals to be calculated. We built the molecule with the molecular editor, and we gave the job the title "Formaldehyde molecule" on the WebMO page.

```
a.  #N HF/STO-3G SP GFINPUT POP=FULL 6D 10F

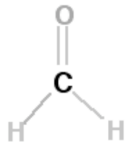
b.  Formaldehyde molecule

c.  0 1
d.  C
e.  O 1 B1
f.  H 1 B2 2 A1
g.  H 1 B3 2 A2 3 D1

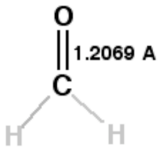
h.  D1 180.00000
i.  B1 1.2069500
j.  B2 1.0832400
k.  A1 122.52865
l.  B3 1.0832400
m.  A2 122.52865
```

A description of each of the lines is as follows (NOTE: we have added the a, b, c notation on the left of the input file for easy reference. This is not part of a Gaussian input file).

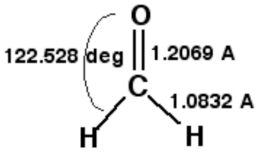
a. The *route* line states that we would like Normal output (as compared to terse or verbose); our model chemistry is a Hartree-Fock calculation using a STO-3G basis set; we are requesting a Single Point energy calculation; and we wish to see a *full* output of all of the molecular orbitals.
b. This is the title of our job, as indicated from the input box in the WebMO inter
c. This line states that our molecule – formaldehyde – has no charge and a *spin multiplicity* of 1. Spin multiplicity, or simply multiplicity, is a measure of the pairing of electrons. The formula for multiplicity is 2*S*+1, where *S* is the spin. If electrons are paired, as is typically the case, the first electron has a spin of -_, and the second has a spin of _, for a total spin of 0. This gives the multiplicity as 2*0+1, or 1, otherwise known as a *singlet*. If there is one unpaired electron, then the multiplicity is a doublet (2*_+1=2).



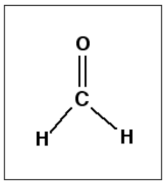
d. This line is the start of a *Z*-matrix, or a description of the geometry of the molecule. The starting atom is (arbitrarily) selected as carbon, indicated here with a C.
e. This line shows that oxygen (O) is connected to the carbon at a bond length of 1.2069 A (angstroms). The "B1" is simply a reference value, with the actual value of B1 shown in the lookup table in Row i. This is a very common notational system used in *Z*-matrices and in Gaussian.



f. This line shows the bond length and bond angle of a hydrogen connected to the central carbon.
g. This line shows the bond length, angle, and dihedral angle of the other hydrogen connected to the central carbon.



h. Lines h through m are the lookup tables for the bond lengths, angles, and dihedrals. "B's" refers to bond length, "A's" refer to bond angles, and the single "D" refers to the single dihedral angle found in formaldehyde. This is very difficult, however, to show on paper. The formaldehyde molecule is flat, or *planar*. We show the molecule here lying completely flat on a piece of paper, or a *plane*. The angle from the bottom side of the paper to the top of the paper is 180 degrees, and that is the dihedral angle of this particular molecule.



Gaussian and other researchers often publish example input files for various types of calculations. Reviewing these files can be very instructive! It should be noted that one of the goals of using an interface such as WebMO, with its pull-down menus, is the avoidance of having to write these input files by hand! For the student researcher, however, understanding the structure of these input files allows him or her to make sense of published literature articles, use Gaussian reference materials, and, even in the WebMO interface, customized a Gaussian job beyond the choices provided by the pull-down menus.

Users can preview the input file in WebMO by clicking on the **Preview** tab in the Job Options window. Clicking on the "Generate" button creates the input file for inspection.

Gaussian Keywords:

Like many computational chemistry codes, Gaussian uses a keyword system. Keywords are short and typically cryptic instructions to the software that describe what the user wishes to do. Most of the keywords that are needed by the user on the North Carolina High School Computational Chemistry server are programmed into the various input windows as pull-down menus. For example, there are pull-down menus for the method (theory), the choice of basis set, and the job type (calculation). Others, such as property keywords, are added automatically. For the more experienced user, additional keywords can be added by hand.

There are four types of keywords in Gaussian:

1. Method: this is an indication of the theory that is requested. If, for example, you choose the **Hartree-Fock** level of theory from the pull-down menu, the keyword "HF" will be keyed into the input file.
2. Basis Set: this keyword is also obtained from the pull-down menu.
3. Job Type: again, this is chosen from the pull-down menu. The difference is that this pull-down menu presents the choices as completely spelled-out terms, such as **Geometry Optimization** or **Vibrational Frequencies**. Representative keywords are shown below:
    a. SP: Single point energy, on the pull-down menu as **Molecular Energy**
    b. OPT: Geometry optimization, so listed on the pull-down menu
    c. FREQ: Vibrational frequencies, so listed on the pull-down menu
4. Properties: this keyword option is typically added automatically. For experienced users, properties keywords can be added using the **Advanced** tab in the WebMO input windows. Some examples:
    a. POP=FULL: this requests that all of the molecular orbitals, and a description of how the electrons are distributed among those orbitals, be printed in their entirety. This option is shown below in the annotated Gaussian output.
    b. AIM: Atoms In Molecules, a keyword that calculates bond order for a given molecule.
    c. NMR: this keyword generates a Nuclear Magnetic Resonance (NMR) scan of your molecule.

As before, one of the advantages of using the WebMO interface is that the user does not need to know the cryptic form of the keywords. It is helpful, however, for Gaussian users to understand the keyword concept. Students performing research projects will likely need to use one or more keywords (again, using the **Advanced** tab in WebMO) to customize Gaussian jobs.

Interpreting Gaussian Output:

This reading describes in basic detail the various parts of a relatively simple output file. *In most cases*, the user of the North Carolina High School Computational Chemistry server will obtain *most* if not all of his or her calculation results from the standard output window provided by the WebMO software. There is, however, a great deal of information contained in the "Raw Output" file, available in the Job Manager window. Also, if you click on the "All Files" link on the Job Manager, you can see parts of the output file. These are basic text files that can be saved and opened using any standard text editor or word processor.

Comments or output information that is pertinent at this stage are in **bold**, embedded in the output. The actual output file is shown indented, using a slightly different font. Some sections may be omitted, especially a large amount of the copyright information and other legalese at the beginning of the output file!

A NOTE TO THE READER! This next section requires some mental heavy lifting. The reader is encouraged, however, to work through this section carefully. Doing so will help you to interpret the results in the standard WebMO **View Calculated Quantities** window as well as the more detailed results available from the **Raw Output** link on the WebMO Job Manager window.

This shows that you are using Gaussian 2003, otherwise known as G03:

```
Entering Gaussian System, Link 0=/usr/local/g03/g03
 Initial command:
 /usr/local/g03/l1.exe /tmp/webmo/3563/Gau-9299.inp -scrdir=/tmp/webmo/3563/
 Entering Link 1 = /usr/local/g03/l1.exe PID=      9301.
```

If a student does a research paper or other report, the citation below is how the vendors wish to be cited:

```
Cite this work as:
 Gaussian 03, Revision C.02,
 M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria,
 M. A. Robb, J. R. Cheeseman, J. A. Montgomery, Jr., T. Vreven,
 K. N. Kudin, J. C. Burant, J. M. Millam, S. S. Iyengar, J. Tomasi,
```

```
V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega,
G. A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota,
R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao,
H. Nakai, M. Klene, X. Li, J. E. Knox, H. P. Hratchian, J. B. Cross,
C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev,
A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, P. Y. Ayala,
K. Morokuma, G. A. Voth, P. Salvador, J. J. Dannenberg,
V. G. Zakrzewski, S. Dapprich, A. D. Daniels, M. C. Strain,
O. Farkas, D. K. Malick, A. D. Rabuck, K. Raghavachari,
J. B. Foresman, J. V. Ortiz, Q. Cui, A. G. Baboul, S. Clifford,
J. Cioslowski, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz,
I. Komaromi, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham,
C. Y. Peng, A. Nanayakkara, M. Challacombe, P. M. W. Gill,
B. Johnson, W. Chen, M. W. Wong, C. Gonzalez, and J. A. Pople,
Gaussian, Inc., Wallingford CT, 2004.
```

This next section repeats back to you what was requested in setting up the job. This shows the model chemistry and the requested calculation. The "N" stands for normal output, a model chemistry of Hartree-Fock theory using the STO-3G basis set, a molecular energy calculation (SP=single point energy), an instruction (GFINPUT) that asks to print out the basis sets, and a notation that says to print out the full population analysis. The 6D and 10F describe the level of detail for the molecular orbitals:

```
******************************************
Gaussian 03:  IA32L-G03RevC.02 12-Jun-2004
                4-Jul-2006
******************************************
Default route:  Maxdisk=40GB
---------------------------------------
#N HF/STO-3G SP GFINPUT POP=FULL 6D 10F
---------------------------------------
```

This next section explains to the programmer the different parts of the program that the calculation is using. It is not of importance to the majority of users, but can be helpful in troubleshooting. It is known as the *route* section:

```
1/38=1/1;
2/17=6,18=5,40=1/2;
3/6=3,8=22,11=9,16=1,24=10,25=1,30=1/1,2,3;
4//1;
5/5=2,32=1,38=5/2;
6/7=3,28=1/1;
99/5=1,9=1/99;
```

The title of your job, as you specified in setting up your calculation comes next:

```
----------------
Formaldehyde MOs
----------------
```

The *Z*-matrix represents how the software knows the molecular geometry (structure). Notice that the molecule has no charge and a multiplicity of 1 (all paired electrons). More details are available in the "Computational Analogy" chapter:

```
Symbolic Z-matrix:
Charge =  0 Multiplicity = 1
C
O                    1    B1
```

```
H                     1    B2       2    A1
H                     1    B3       2    A2       3    D1       0
     Variables:
 D1                  180.
 B1                    1.20695
 B2                    1.08324
 A1                  122.52865
 B3                    1.08324
 A2                  122.52865
```

The structure is also represented as a more standard X-Y-Z coordinate system. The central atom is carbon, placed at the origin of the X-Y-Z grid:

```
                         Input orientation:
---------------------------------------------------------------------
Center     Atomic     Atomic              Coordinates (Angstroms)
Number     Number      Type              X           Y           Z
---------------------------------------------------------------------
   1          6             0        0.000000    0.000000    0.000000
   2          8             0        0.000000    0.000000    1.206953
   3          1             0        0.913307    0.000000   -0.582483
   4          1             0       -0.913307    0.000000   -0.582483
---------------------------------------------------------------------
```

The distance matrix shows the distance of each atom from the other, in units of angstroms:

```
                  Distance matrix (angstroms):
                  1          2          3          4
    1  C    0.000000
    2  O    1.206953   0.000000
    3  H    1.083243   2.009032   0.000000
    4  H    1.083243   2.009032   1.826614   0.000000
```

The stoichiometry is CH2O, just as you already knew:

```
Stoichiometry    CH2O
```

The framework group talks about the symmetry of the molecule. In this example, formaldehyde has C_{2V} symmetry. We won't be discussing what is known as *point group symmetry* in this *Guide*. The information following the framework line also relate more information about the symmetry of the molecule to the user:

```
Framework group  C2V[C2(CO),SGV(H2)]
Deg. of freedom     3
Full point group                 C2V     NOp   4
Largest Abelian subgroup         C2V     NOp   4
Largest concise Abelian subgroup C2      NOp   2
```

In case you have forgotten, the output lists the *standard orientation*, or the geometry of the molecule:

```
                        Standard orientation:
---------------------------------------------------------------------
Center     Atomic     Atomic              Coordinates (Angstroms)
Number     Number      Type              X           Y           Z
---------------------------------------------------------------------
   1          6             0        0.000000    0.000000   -0.530666
   2          8             0        0.000000    0.000000    0.676287
   3          1             0        0.000000    0.913307   -1.113149
```

```
   4          1             0        0.000000   -0.913307   -1.113149
 ---------------------------------------------------------------------
Rotational constants (GHZ):    300.5858276     38.7849258     34.3523975
```

The next section shows you details about the basis set. We've chosen a STO-3G basis set:

```
Standard basis: STO-3G (6D, 10F)
AO basis set in the form of general basis input:
```

This is the STO-3G basis set for carbon. This data is built into the Gaussian software, but it can also be found through the Gaussian Basis Set Order Form (http://www.emsl.pnl.gov/forms/basisform.html). A lab on basis sets contained in this volume explains how to use this resource:

```
 1 0
S   3 1.00       0.000000000000
     0.7161683735D+02  0.1543289673D+00
     0.1304509632D+02  0.5353281423D+00
     0.3530512160D+01  0.4446345422D+00
SP   3 1.00       0.000000000000
     0.2941249355D+01 -0.9996722919D-01  0.1559162750D+00
     0.6834830964D+00  0.3995128261D+00  0.6076837186D+00
     0.2222899159D+00  0.7001154689D+00  0.3919573931D+00
****
```

This is the STO-3G basis set for oxygen:

```
 2 0
S   3 1.00       0.000000000000
     0.1307093214D+03  0.1543289673D+00
     0.2380886605D+02  0.5353281423D+00
     0.6443608313D+01  0.4446345422D+00
SP   3 1.00       0.000000000000
     0.5033151319D+01 -0.9996722919D-01  0.1559162750D+00
     0.1169596125D+01  0.3995128261D+00  0.6076837186D+00
     0.3803889600D+00  0.7001154689D+00  0.3919573931D+00
****
```

Finally, the STO-3G basis sets for the two hydrogens. Notice that the alpha values (first column of three numbers) and contraction coefficients (second column) are identical:

```
 3 0
S   3 1.00       0.000000000000
     0.3425250914D+01  0.1543289673D+00
     0.6239137298D+00  0.5353281423D+00
     0.1688554040D+00  0.4446345422D+00
****
 4 0
S   3 1.00       0.000000000000
     0.3425250914D+01  0.1543289673D+00
     0.6239137298D+00  0.5353281423D+00
     0.1688554040D+00  0.4446345422D+00
****
```

This next section gives a little more information about the basis functions, referring again to symmetry considerations:

```
There are     7 symmetry adapted basis functions of A1  symmetry.
```

```
There are     0 symmetry adapted basis functions of A2  symmetry.
There are     2 symmetry adapted basis functions of B1  symmetry.
There are     3 symmetry adapted basis functions of B2  symmetry.
Integral buffers will be    262144 words long.
Raffenetti 1 integral format.
Two-electron integral symmetry is turned on.
```

This next section provides some important information, particularly that there are 12 basis functions. This is important for several reasons. The first is that it tells you that the 16 electrons are distributed over 12 molecular orbitals (as we shall see later). The second is that the number of basis functions is a good predictor of how long the calculation will take. There is a lab activity that explains how to predict the number of basis functions, and, as a result, the approximate run times for the calculation.

Notice also that there are 36 primitive Gaussians. This should make sense. We are using an STO-3G basis function. This means we are calculating the Gaussian Type Orbital (GTO) equation 3 times for *each* of the 12 Slater basis functions. 1 STO = 3 GTOs, so 12 STOs=36 (primitive) GTOs:

```
    12 basis functions,    36 primitive gaussians,    12 cartesian basis functions
```

This next line says you have 16 electrons. (Remember: 8 from oxygen, 6 from carbon, 1 each from the two hydrogens). All electrons have spin, either up or down. 8 of these electrons are spin-up (alpha electrons), 8 are spin-down (beta electrons). Since there is an equal number of both, they are paired, no unpaired electrons:

```
     8 alpha electrons        8 beta electrons
```

The output reports the nuclear repulsion energy, in units of hartrees. This means that the energy of all four nuclei trying to repel each other is 31.411 hartrees. The unit of hartrees is simply an energy unit, just as is kcalories per mole, kilojoules per mole, and electron-volts. A conversion table is shown below. This table is not included in the output file:

```
        nuclear repulsion energy        31.4113804717 Hartrees.
```

1 unit =	hartree	kJ per mole	kcal per mole	eV
hartree	1	2625.5	627.51	27.212
kJ per mole	3.8088×10^{-4}	1	0.23901	1.0364×10^{-2}
kcal per mole	1.5936×10^{-3}	4.1840	1	4.3363×10^{-2}
eV	3.6749×10^{-2}	96.485	23.061	1

This next section is of little importance to most users. It describes some of the internal mathematics used by Gaussian to do its calculations:

```
NAtoms=    4 NActive=    4 NUniq=    3 SFac= 2.05D+00 NAtFMM=   60 Big=F
One-electron integrals computed using PRISM.
NBasis=    12 RedAO= T  NBF=     7     0     2     3
NBsUse=    12 1.00D-06 NBFU=     7     0     2     3
Harris functional with IExCor=  205 diagonalized for initial guess.
ExpMin= 1.69D-01 ExpMax= 1.31D+02 ExpMxC= 1.31D+02 IAcc=1 IRadAn=         1 AccDes= 1.00D-
06
HarFok:  IExCor= 205 AccDes= 1.00D-06 IRadAn=         1 IDoV=1
ScaDFX=  1.000000  1.000000  1.000000  1.000000
Initial guess orbital symmetries:
      Occupied  (A1) (A1) (A1) (A1) (B2) (A1) (B1) (B2)
      Virtual   (B1) (A1) (B2) (A1)
The electronic state of the initial guess is 1-A1.
Warning!  Cutoffs for single-point calculations used.
```

```
Requested convergence on RMS density matrix=1.00D-04 within 128 cycles.
Requested convergence on MAX density matrix=1.00D-02.
Requested convergence on            energy=5.00D-05.
No special actions if energy rises.
Keep R1 integrals in memory in canonical form, NReq=      422883.
```

This next section provides the first real data of the calculation. The Schrödinger equation is fundamentally designed to determine the *energy* of the molecule, and that is reported here. The energy of the formaldehyde molecule is -112.353556547 hartrees. The calculation required 5 cycles, or iterations, to achieve this value. It decided that it had reached the "right" answer when the difference between two energy values was about -0.000044. The energy value is one of the most important results of this particular calculation:

```
Convergence on energy, delta-E=-4.40D-05
SCF Done:  E(RHF) =  -112.353556547     A.U. after    5 cycles
```

The "-V/T" is a mathematical construct known as the *virial theorem*. If the value of –V/T was exactly 2, then we would have an exact solution to the Schrödinger equation. We don't have exactly 2, which means that all of our answers are close, but not exact:

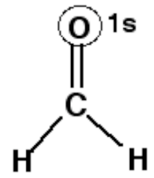
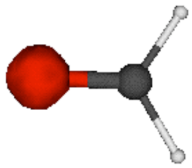
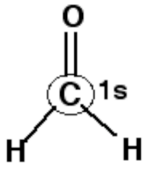
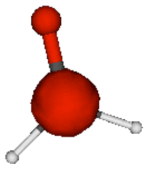
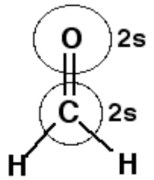
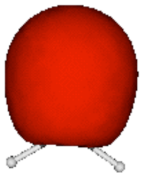
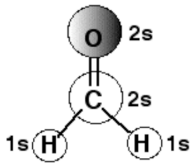
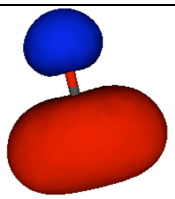
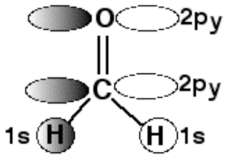
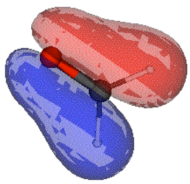
```
Convg  =    0.2507D-03             -V/T =  2.0085
S**2   =   0.0000
```

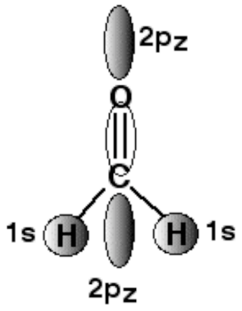
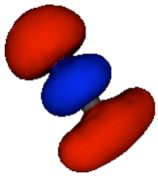
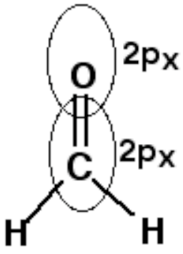
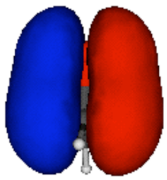
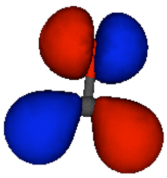
This next section provides most of the information about the calculation we requested. The calculation distributes electrons over 8 molecular orbitals (2 electrons per orbital), but shows us information about 12 orbitals. There are 8 occupied orbitals and 4 unoccupied, or virtual, orbitals. The two most important orbitals are the HOMO (Highest Occupied Molecular Orbital) and the LUMO (Lowest Unoccupied (virtual) Molecular Orbital. The HOMO and the LUMO can be identified by finding those two orbitals where the symmetry changes from *occupied* (HOMO) to *virtual* (LUMO). In this case, the HOMO is MO number 8, with a value of -0.35591. The LUMO is MO number 9, with a value of 0.28473. The process of distributing electrons is known as a *population analysis* (in this case, a *Mulliken* population analysis, the standard method used in most software packages):

```
**********************************************************************
           Population analysis using the SCF density.
**********************************************************************
Orbital symmetries:
      Occupied  (A1) (A1) (A1) (A1) (B2) (A1) (B1) (B2)
      Virtual   (B1) (A1) (B2) (A1)
The electronic state is 1-A1.
Alpha  occ. eigenvalues --  -20.31521 -11.12653  -1.34474  -0.81363  -0.64195
Alpha  occ. eigenvalues --   -0.54927  -0.44913  -0.35591
Alpha virt. eigenvalues --    0.28473   0.64987   0.75458   0.92564
```

This next section provides a great deal of information, somewhat overwhelming at first. As above, we see the 12 molecular orbitals listed, of which the first 8 are occupied. The last four are unoccupied, or virtual. The eigenvalue is the energy value of that MO, in units of hartrees. For each orbital, the output describes which shell (1s, 2s, $2p_x$, etc) is the most predominant.

The table below shows all 8 occupied orbitals, with their symmetry notation (A1, B1, B2); a description of what you are seeing; the MO number for reference; a drawing of the various parts of the orbital; and a graphic of the MO from the Gaussian output. Use this table as a reference for your study of the molecular orbitals shown in the Gaussian output below. In the graphics, the red and blue colors are used by molecular models to represent different phases of the orbital, red being negative and blue being positive. We will walk you through several of these:

Symmetry	Description of Orbital	MO #	Diagram	Graphic
A1. The "A" notation means that the orbital is symmetric	This orbital represents a 1s orbital surrounding the oxygen (O) atom.	1		
A1	This orbital represents a 1s orbital surrounding the carbon (C) atom.	2		
A1	This orbital represents one of the bonds – the sigma bond – between the carbon and the oxygen. A sigma bond is a chemical bond where one orbital overlaps another, forming a bond. The two 2s orbitals overlap to form one big sigma *bonding* orbital. The two circles shown in the diagram overlap each other to form the large red orbital in the graphic. The oxygen 2s is slightly bigger than the carbon 2s, giving the red orbital the odd shape.	3		
A1	This orbital has two parts. The first is *bonding* between the carbon and the hydrogens. The C and two H circles all overlap, giving the hot-dog shaped red orbital. You also see an oxygen lone pair coming off of the oxyen atom.	4		
B2. The "B" notation here means the orbital is anti-symmetric.	This orbital is also carbon-hydrogen bonding, with oxygen lone pairs. The like-colored parts of the orbitals overlap, so all of the gray parts on the left of the drawing overlap to form the odd-shaped blue, and the white parts of the diagram overlap to form the red part.	5		

A1	This orbital is also carbon-hydrogen bonding, along with some influences from oxygen lone pairs.	6		
B1	Orbital 7 is the pi (π) bond, the second bond in the C=O double bond.	7		
B2	The last orbital, the HOMO (Highest Occupied Molecular Orbital) also reflects C-H bonding, along with oxygen lone pairs.	8		

```
Molecular Orbital Coefficients
                           1         2         3         4         5
                         (A1)--O   (A1)--O   (A1)--O   (A1)--O   (B2)--O
    EIGENVALUES --    -20.31521 -11.12653  -1.34474  -0.81363  -0.64195
  1 1   C  1S          0.00052   0.99258  -0.12428  -0.18482   0.00000
  2        2S         -0.00739   0.03333   0.27696   0.56993   0.00000
  3        2PX         0.00000   0.00000   0.00000   0.00000   0.00000
  4        2PY         0.00000   0.00000   0.00000   0.00000   0.53508
  5        2PZ        -0.00640   0.00047   0.15875  -0.23263   0.00000
  6 2   O  1S          0.99427   0.00011  -0.21884   0.09878   0.00000
  7        2S          0.02613  -0.00578   0.76449  -0.42734   0.00000
  8        2PX         0.00000   0.00000   0.00000   0.00000   0.00000
  9        2PY         0.00000   0.00000   0.00000   0.00000   0.43728
 10        2PZ        -0.00576   0.00158  -0.17391  -0.15950   0.00000
 11 3   H  1S          0.00022  -0.00668   0.03282   0.26580   0.29855
 12 4   H  1S          0.00022  -0.00668   0.03282   0.26580  -0.29855
```

Take a look at MO 1, with an eigenvalue of -20.31521. Ignoring the sign (negatives), what is the largest number? We've underlined it for you, the 0.99427. This suggests that MO 1 is primarily made up of a 1s orbital on the oxygen atom. From the reference table, you can see that this agrees with the graphical results shown in the far right-hand column.

Now take a look at MO 5. Scanning down the column, we see four relatively big numbers: 0.53508, 0.43728, 0.29855, and -0.29855 (remember to ignore the negative sign). Following those numbers over to the left, this

orbital is built from a carbon $2p_y$, an oxygen $2p_y$, and two hydrogen 1s orbitals. You can see the p-orbitals running between the carbon and oxygen. The extra bulge over the two hydrogens comes from the effect of the 1s orbitals.

As another example, take a look at MO 7, with an *eigenvalue* of -0.44913. For carbon, all of that orbital is a $2p_x$. If you look down the column, you will see that the oxygen is also completely $2p_x$. This suggests that MO 7 represents the pi (π) bond between the carbon and the oxygen.

				6	7	8	9	10
				(A1)--O	(B1)--O	(B2)--O	(B1)--V	(A1)--V
	EIGENVALUES --			-0.54927	-0.44913	-0.35591	0.28473	0.64987
1	1	C	1S	0.03389	0.00000	0.00000	0.00000	-0.20805
2			2S	-0.10843	0.00000	0.00000	0.00000	1.33023
3			2PX	0.00000	**0.61055**	0.00000	0.82141	0.00000
4			2PY	0.00000	0.00000	-0.18082	0.00000	0.00000
5			2PZ	-0.44400	0.00000	0.00000	0.00000	-0.46201
6	2	O	1S	-0.09491	0.00000	0.00000	0.00000	0.02748
7			2S	0.50574	0.00000	0.00000	0.00000	-0.15764
8			2PX	0.00000	**0.67258**	0.00000	-0.77144	0.00000
9			2PY	0.00000	0.00000	0.87285	0.00000	0.00000
10			2PZ	0.68065	0.00000	0.00000	0.00000	0.23607
11	3	H	1S	0.15459	0.00000	-0.35885	0.00000	-0.90620
12	4	H	1S	0.15459	0.00000	0.35885	0.00000	-0.90620

				11	12
				(B2)--V	(A1)--V
	EIGENVALUES --			0.75458	0.92564
1	1	C	1S	0.00000	-0.09697
2			2S	0.00000	0.65670
3			2PX	0.00000	0.00000
4			2PY	1.16534	0.00000
5			2PZ	0.00000	1.17988
6	2	O	1S	0.00000	0.11727
7			2S	0.00000	-0.88515
8			2PX	0.00000	0.00000
9			2PY	-0.32174	0.00000
10			2PZ	0.00000	0.93140
11	3	H	1S	-0.85753	0.14684
12	4	H	1S	0.85753	0.14684

At this stage, we can ignore the density matrix:

DENSITY MATRIX.

				1	2	3	4	5
1	1	C	1S	2.07194				
2			2S	-0.22071	0.82891			
3			2PX	0.00000	0.00000	0.74554		
4			2PY	0.00000	0.00000	0.00000	0.63802	
5			2PZ	0.01736	-0.08081	0.00000	0.00000	0.55299
6	2	O	1S	0.01269	-0.00273	0.00000	0.00000	-0.04388
7			2S	-0.00922	-0.17409	0.00000	0.00000	-0.00789
8			2PX	0.00000	0.00000	0.82129	0.00000	0.00000
9			2PY	0.00000	0.00000	0.00000	0.15230	0.00000
10			2PZ	0.15145	-0.42556	0.00000	0.00000	-0.58535
11	3	H	1S	-0.10920	0.28718	0.00000	0.44928	-0.25053
12	4	H	1S	-0.10920	0.28718	0.00000	-0.44928	-0.25053

				6	7	8	9	10

```
 6 2   O  1S          2.11045
 7        2S         -0.46305   2.04710
 8        2PX         0.00000   0.00000   0.90473
 9        2PY         0.00000   0.00000   0.00000   1.90617
10        2PZ        -0.09605   0.55856   0.00000   0.00000   1.03802
11 3   H  1S          0.00924  -0.02055   0.00000  -0.36534   0.11422
12 4   H  1S          0.00924  -0.02055   0.00000   0.36534   0.11422
                         11         12
11 3   H  1S          0.62715
12 4   H  1S         -0.24448   0.62715
```

Likewise, we can dispense with the *Full Mulliken population analysis*:

```
 Full Mulliken population analysis:
                         1         2         3         4         5
 1 1   C  1S          2.07194
 2        2S         -0.05482   0.82891
 3        2PX         0.00000   0.00000   0.74554
 4        2PY         0.00000   0.00000   0.00000   0.63802
 5        2PZ         0.00000   0.00000   0.00000   0.00000   0.55299
 6 2   O  1S          0.00000  -0.00010   0.00000   0.00000  -0.00275
 7        2S         -0.00034  -0.06382   0.00000   0.00000  -0.00351
 8        2PX         0.00000   0.00000   0.17486   0.00000   0.00000
 9        2PY         0.00000   0.00000   0.00000   0.03243   0.00000
10        2PZ        -0.00938   0.13759   0.00000   0.00000   0.18358
11 3   H  1S         -0.00691   0.14205   0.00000   0.17788   0.06326
12 4   H  1S         -0.00691   0.14205   0.00000   0.17788   0.06326
                         6         7         8         9        10
 6 2   O  1S          2.11045
 7        2S         -0.10961   2.04710
 8        2PX         0.00000   0.00000   0.90473
 9        2PY         0.00000   0.00000   0.00000   1.90617
10        2PZ         0.00000   0.00000   0.00000   0.00000   1.03802
11 3   H  1S          0.00005  -0.00153   0.00000  -0.01374  -0.00842
12 4   H  1S          0.00005  -0.00153   0.00000  -0.01374  -0.00842
                        11        12
11 3   H  1S          0.62715
12 4   H  1S         -0.03848   0.62715
```

We do, however, want to get an idea of the *Gross orbital population*. This shows where Gaussian thinks electrons are distributed. Adding up all of the decimals results in roughly 6 electrons for carbon, 8 for oxygen, 1 for each hydrogen, and 16 for the entire molecule:

```
  Gross orbital populations:
                         1
 1 1   C  1S          1.99359
 2        2S          1.13186
 3        2PX         0.92040
 4        2PY         1.02620
 5        2PZ         0.85682
 6 2   O  1S          1.99808
 7        2S          1.86677
 8        2PX         1.07960
 9        2PY         1.91111
10        2PZ         1.33297
11 3   H  1S          0.94129
12 4   H  1S          0.94129
```

If you add up the numbers across the row for each atom, you will get slightly more or less than the integer values for the atoms (C=6, O=8, H=1):

```
          Condensed to atoms (all electrons):
               1          2          3          4
    1  C    4.727770   0.448556   0.376278   0.376278
    2  O    0.448556   7.787260  -0.023644  -0.023644
    3  H    0.376278  -0.023644   0.627146  -0.038485
    4  H    0.376278  -0.023644  -0.038485   0.627146
```

Formaldehyde is a neutral molecule, meaning that it has no charge. Gaussian thinks so too! Notice that the individual atoms have *partial charges* on them, not the integer values we typically assign to them:

```
Mulliken atomic charges:
               1
    1  C    0.071118
    2  O   -0.188528
    3  H    0.058705
    4  H    0.058705
Sum of Mulliken charges=   0.00000
Atomic charges with hydrogens summed into heavy atoms:
               1
    1  C    0.188528
    2  O   -0.188528
    3  H    0.000000
    4  H    0.000000
Sum of Mulliken charges=   0.00000
Electronic spatial extent (au):  =    57.9282
Charge=     0.0000 electrons
```

This next section shows the dipole moment calculations. Since this is a planar (flat) molecule, we only have a dipole in the Z direction. The total dipole for the molecule is 1.5144 debyes:

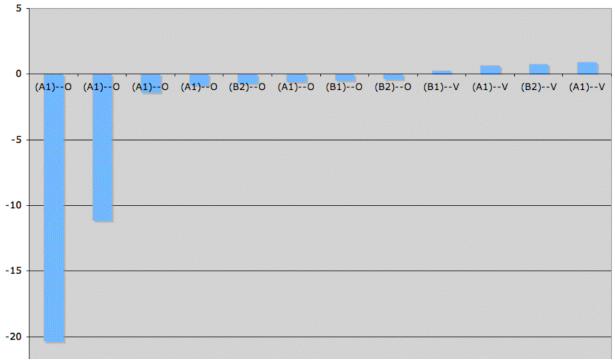
```
Dipole moment (field-independent basis, Debye):
   X=     0.0000    Y=     0.0000    Z=    -1.5144  Tot=     1.5144
```

Other moments are also calculated. These are known as *multipoles*, but are rarely of interest to most users:

```
Quadrupole moment (field-independent basis, Debye-Ang):
  XX=   -10.4132   YY=   -10.5607   ZZ=   -11.3350
  XY=     0.0000   XZ=     0.0000   YZ=     0.0000
Traceless Quadrupole moment (field-independent basis, Debye-Ang):
  XX=     0.3565   YY=     0.2089   ZZ=    -0.5654
  XY=     0.0000   XZ=     0.0000   YZ=     0.0000
Octapole moment (field-independent basis, Debye-Ang**2):
 XXX=     0.0000  YYY=     0.0000  ZZZ=     2.6983  XYY=     0.0000
 XXY=     0.0000  XXZ=     1.6091  XZZ=     0.0000  YZZ=     0.0000
 YYZ=     0.5076  XYZ=     0.0000
Hexadecapole moment (field-independent basis, Debye-Ang**3):
XXXX=    -6.8715 YYYY=   -14.3946 ZZZZ=   -39.3151 XXXY=     0.0000
XXXZ=     0.0000 YYYX=     0.0000 YYYZ=     0.0000 ZZZX=     0.0000
ZZZY=     0.0000 XXYY=    -3.7589 XXZZ=    -7.5879 YYZZ=    -8.2179
XXYZ=     0.0000 YYXZ=     0.0000 ZZXY=     0.0000
N-N= 3.141138047168D+01 E-N=-3.277399462147D+02  KE= 1.114023324343D+02
Symmetry A1   KE= 1.009796377183D+02
Symmetry A2   KE= 0.000000000000D+00
Symmetry B1   KE= 3.554148585134D+00
Symmetry B2   KE= 6.868546130918D+00
```

The most important numbers below are the orbital energies. We have roughly two electrons per occupied level (8 levels for 16 electrons), starting at -20.31521 hartrees. In the chart (not part of the Gaussian output), you can see the HOMO (orbital 8) and the LUMO (orbital 9) below and above the 0 (zero) line, respectively:

```
 Orbital energies and kinetic energies (alpha):
                          1         2
      1  (A1)--O     -20.31521  28.66366
      2  (A1)--O     -11.12653  15.65155
      3  (A1)--O      -1.34474   2.44519
      4  (A1)--O      -0.81363   1.53255
      5  (B2)--O      -0.64195   1.23161
      6  (A1)--O      -0.54927   2.19686
      7  (B1)--O      -0.44913   1.77707
      8  (B2)--O      -0.35591   2.20267
      9  (B1)--V       0.28473   2.37493
     10  (A1)--V       0.64987   2.50467
     11  (B2)--V       0.75458   2.42369
     12  (A1)--V       0.92564   4.02320
 Total kinetic energy from orbitals= 1.114023324343D+02
```



This next section shows a very cryptic regurgitation of what you requested and some of the more basic results. This bit of data goes into an archive file, which is typically not needed by most users:

```
1\1\GINC-CHEMISTRY\SP\RHF\STO-3G\C1H2O1\WWWRUN\04-Jul-2006\0\\#N HF/ST
O-3G SP GFINPUT POP=FULL 6D 10F\\Formaldehyde MOs\\0,1\C\O,1,1.206953\
H,1,1.0832433,2,122.52865\H,1,1.0832433,2,122.52865,3,180.,0\\Version=
IA32L-G03RevC.02\State=1-A1\HF=-112.3535565\RMSD=2.507e-04\Dipole=0.,0
.,-0.5958009\PG=C02V [C2(C1O1),SGV(H2)]\\@
```

Gaussian always gives you a "fortune cookie", one of the little quirks that endears Gaussian to many computational chemists:

```
 A SOLDIER'S LIFE IS A LIFE OF HONOR, BUT A DOG WOULD NOT LEAD IT.
 -- PRINCE RUPERT, FOUNDER OF THE HUDSON'S BAY COMPANY
```

This last section shows a final accounting of the statistics of your job. This was a short job – 10.2 seconds – that used very little space for the read-write file (RWF), the electron integral file (INT), the two-electron derivative file (D2E), the checkpoint file (CHK), and the scratch file (SCR):
]

```
 Job cpu time:  0 days  0 hours  0 minutes 10.2 seconds.
 File lengths (MBytes):  RWF=     11 Int=      0 D2E=      0 Chk=      7 Scr=      1
```

The final message in the output file is a really good thing to see. It states that your job completed as requested, with no "failure to converge" message or other problems, as is described in the next section -- Troubleshooting:

```
 Normal termination of Gaussian 03 at Tue Jul  4 19:53:26 2006.
```

<u>Troubleshooting Gaussian Jobs</u>:

There are fundamentally *two* types of *failed* jobs in Gaussian:

1. jobs for which you actually get a "Failed" message (in red!) in the WebMO Job Manager
2. jobs for which you get one or several results that are clearly not reasonable chemical answers

Both types of failed jobs are described here.

Failed Job Messages

Even for the most experienced Gaussian user, failed jobs can be difficult to diagnose. Gaussian has a nice feature known as a *restart*. In the graphic, we show three failed jobs in the WebMO interface. Note, by the way, the run times for these jobs! Under **Actions**, there are two icons. Clicking on the first icon displays the raw output file in text format. Scrolling to the bottom of the output file might show a message such as this:




```
>>>>>>>>>> Convergence criterion not met.
 SCF Done:  E(RHF) =  -112.176862421     A.U. after  129 cycles
            Convg  =    0.5429D-05             -V/T =  2.0034
            S**2   =   0.0000
 Convergence failure -- run terminated.
 Error termination via Lnk1e in /usr/local/g03/l502.exe at Fri Jul  7 11:47:27 2006.
 Job cpu time:  0 days  0 hours  0 minutes  9.4 seconds.
 File lengths (MBytes):  RWF=     11 Int=     0 D2E=      0 Chk=      1 Scr=      1
```

This message says that, after repeating (iterating) through the calculation cycle 129 times (!), the software could not converge, or narrow in on, a correct or reasonable solution. An experienced Gaussian user would recognize that the error is in a part of the program known as a *link*. In this case, the error is in Link 502, which is defined as:

```
L502 Iteratively solves the SCF equations (conven. UHF & ROHF, all direct methods, SCRF)
```

If you look carefully in the error message above, you should find the Link 502 error! The software, for whatever reason, was not able to perform a *self-consistent field* (SCF) convergence on your molecule.

All of that is nice, but what do you do about it? There are several options:

1. Restart your molecule. You can do this clicking on the second icon in the graphic above, the one that looks like a yellowish clock. This will restart your job, using *some* of the previously computed results, sort of like starting in the middle. You will have to redefine some of your requests, using the same windows you did to start the job in the first place.
2. Try building the molecule again. Start a new job, and this time try using a smaller basis set, like STO-3G. If that works, then you can re-run the job with the results of that calculation, but using a larger, more powerful basis set.

A note especially to teachers:

1. Students will sometimes build nonsensical molecules, try to run jobs on them, and then express surprise when the job fails! The molecular builder will let you build any molecule you want. It doesn't know that the molecule is nonsensical! The actual computing software, Gaussian in this case, is smarter than the builder. It will balk on molecules that are really not geometrically or chemically reasonable molecules! Failed jobs don't hurt anything, other than consuming time from the users time allocation, described in the next item.
2. Jobs will also fail if the user does not have enough allocated time to run the job. Users of the North Carolina High School Computational Chemistry server should remember that all users are allocated time on the machine. Most users will have a *time limit for a specific job*, and a *total time limit*. For example, if the user has a job time limit of 5 minutes, and the requested calculation needs 6 minutes, the job will fail. The job will also fail if the user exceeds his or her allotted total time. In the research chapter, there will be a discussion of how to estimate job and total time requests for research projects.
3. System administrators and computational chemists at Shodor, who monitor the use of the server, take special note of failed jobs. As appropriate, they can help diagnose the failure.

Jobs that provide questionable results

Jobs, especially jobs that are submitted by users new to molecular modeling and Gaussian, can sometimes return results that have questionable chemical accuracy. As with failed jobs, identifying questionable results typically requires an experienced eye. The most obvious example is when students are running calculations on a number of related molecules. For example, suppose a group of 10 users are running calculations on 10 slightly different, but related, molecules such as a substituted phenol. If, in the course of comparing the results of all 10 calculations in terms of molecular energy, one or two of the results have energy values that are way out of line, then you should suspect that the calculation was not done correctly. The software will give you what you ask for, so the user should go back, check the output file, and otherwise try to make sure that the calculation was set up properly. With WebMO, a common mistake is to build a molecule, add something like a bonded chlorine atom, and then decide to remove the bonded atom. New users will often simply remove the bond, and forget to also remove the chlorine atom! This extra atom will really impair the calculation.

For North Carolina teachers and students, the computational chemistry support staff can provide assistance with results of questionable accuracy. Contact information is available on the main server page (http://www.shodor.org/chemistry).

Gaussian Support Tools:

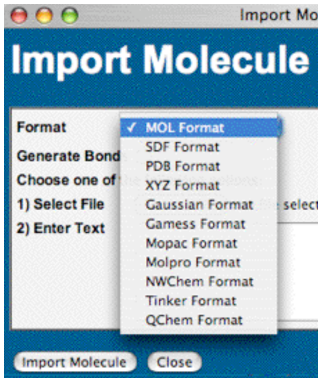
There are a number of stand-along support tools for Gaussian users. GaussView, for example, provides many of the same functions to the user as the WebMO interface. It provides a graphical way to build molecules, submit jobs, view results, display graphical images such as spectra and molecular orbitals, and allow the user to perform troubleshooting tasks.

There are also a number of Gaussian utilities available, most of them free of charge. These utilities are primarily of use to users who are running Gaussian in a UNIX environment on large machines. An example utility is *freqmem*, which allows a user to determine how much computer memory might be required to run a vibrational frequency calculation. Some of these utilities are useful to the (student) researcher who needs to customize a job, restart a complicated job, or check a text-based output file for some particular feature. The WebMO interface, however, provides most if not all of the support that is needed by its users.

WebMO always requires the use of an external graphics viewer, known as MOViewer. This is a Java applet, a type of computer program that runs over the Web. The actual MOViewer software resides on the WebMO computational chemistry server. The end-user must, however, have the Java software installed on his or her browser. This is almost also installed automatically with your Internet browser, such as Internet Explorer or Netscape. Sometimes, however, Java is not installed or up to date. A new browser, Mozilla Firefox, probably does the best job of all of the browsers in keeping the Java engine up to date, and that is the software that is recommended for use with WebMO.

There is one more generic support tool, and that is a file converter. Input and output files in molecular modeling come in various types. Just as there are different word processing file types – Microsoft Word, WordPerfect, AppleWorks – there are also different types in molecular modeling. Common file types are listed below:

1. XMol XYZ Coordinate files
2. SYBYL MOL2 files
3. Protein Data Bank (PDB) files
4. SMILES formatted files



There are a number of resources, including Web-based resources that allow the user to convert from one format to another. WebMO, with its import feature, can read files written in many of the common formats found in the molecular modeling community. For most situations, WebMO does a good job of importing various file formats.

Chapter 19: GAMESS

Key Notes:

GAMESS Basics:

GAMESS is a general *ab initio* quantum chemistry package. It is one of the better-known codes in the molecular modeling community, and has been around since the late 1970s. The name is an acronym, and stands for **G**eneral **A**tomic and **M**olecular **E**lectronic **S**tructure **S**ystem. GAMESS can run standard semi-empirical, Hartree-Fock/SCF, post-SCF, and DFT methods. GAMESS is capable of running only quantum chemical calculations. It does not have the capability, as is found in Gaussian, to run molecular mechanics calculations, although one can do a molecular mechanics job by integrating GAMESS with another code entitled Tinker. It has a significant advantage, however, over Gaussian: it is available for free! GAMESS runs on just about every type of computer, and it has specific versions for the different types of computers, or platforms. As a result, users can download GAMESS and use it directly on their own machine, although running GAMESS through the WebMO server is still the easiest option!

Running GAMESS Jobs:

GAMESS jobs are simple to run using the WebMO interface, and run reasonably quickly. As such, it is a good intermediate choice for a computational engine – faster than Gaussian, but also capable of running the majority of the quantum chemical calculations (semi-empirical, *ab initio*, and DFT). Once you have built the molecule and have chosen GAMESS as the computational engine, the WebMO interface provides you with a series of options via pull-down menus. GAMESS jobs can be modified using the **Advanced** tab on the WebMO interface. GAMESS input files can be viewed by using the **Preview** tab, and then clicking on the **Generate** button.

GAMESS Keywords:

Like most computational chemistry codes, GAMESS has a long list of keywords. These keywords allow the user to customize his or her input file, the instructions sent to the program to perform a calculation. As is the case with most other codes, these keywords are reasonably cryptic, and have very specific rules for how they can be used. WebMO removes the need for users to know the most common keywords because they are available from the pull-down menus. More advanced users, including student researchers, will need to learn how to customize GAMESS jobs by using the keyword system.

Interpreting GAMESS Output:

One of the advantages of the WebMO interface that links the user to GAMESS is that almost all of the important results that come from running a GAMESS calculation are automatically displayed in the **View Calculated Quantities** window in WebMO. It is possible, however, to view the entire text-based output file that is generated with a GAMESS calculation. A short tutorial on how to read this output is included later in this chapter.

Troubleshooting GAMESS Jobs:

Most GAMESS jobs will run to completion successfully and will not fail, assuming that the molecule being calculated is one that is reasonable in size and structure. Jobs do, however, fail in GAMESS with the same frequency as that of other tools. The text-based output that is available through the WebMO **Job Manager** interface provides relatively easy-to-read descriptions as to the source of the failure. Troubleshooting tips are described in greater detail later in this chapter.

GAMESS Support Tools:

With the North Carolina High School Computational Chemistry server, the WebMO program provides the user with the majority of those helper tools, including the Java-based program entitled “MOViewer”. GAMESS interfaces well with external programs such as MOLDEN, which provides an interface for GAMESS input and output files. There are platform-specific tools

such as MacMolPlt, a Macintosh program that allows the visualization of GAMESS output files. WebMO provides virtually all of the support needed to build input files and analyze results.

GAMESS Basics:

GAMESS is the **G**eneral **A**tomic and **M**olecular **E**lectronic **S**tructure **S**ystem software package, and it has its home at the University of Iowa at Ames. GAMESS is one of the best known and most highly used of all of the quantum chemical software packages. Its major advantage over codes such as Gaussian (described in the previous chapter) is that it is available for free, and there are specific versions for every variety of computer system, large and small. There are currently two versions, GAMESS (US) and GAMESS (UK), and there are differences between the two. The North Carolina High School Computational Chemistry server uses GAMESS (US), and that is the code we describe in this chapter.

GAMESS is capable of running *ab initio* calculations using Hartree-Fock methods. It can also do Density Functional Theory (DFT) as well as a number of more advanced types of electronic structure determination methods. GAMESS does not do molecular mechanics methods, although there are methods to allow GAMESS to take advantage of the molecular mechanics capabilities of codes like Tinker. On the North Carolina High School Computational Chemistry server, users can run Tinker as a separate code.

Running GAMESS Jobs:

The basic concept of running a GAMESS calculation is known as a job. GAMESS jobs are submitted by sending a text-based input file to the GAMESS processor, with results returned as one or more text-based output files and a variety of graphical displays. The WebMO software serves as the bridge between the user and the GAMESS code, allowing the user to build the molecule using the Java-based molecular editor/builder, configure the job for the desired method and type of calculation, and view the calculated results. In the example below, we have again built the molecule formaldehyde (CH_2O), and have requested a molecular orbital calculation using the Hartree-Fock method with a STO-3G basis set. The GAMESS input file is shown below:

```
a.   $CONTRL SCFTYP=RHF RUNTYP=ENERGY ICHARG=0 MULT=1 COORD=ZMTMPC $END
b.   $BASIS GBASIS=STO NGAUSS=3 $END
c.   $DATA
d.  Formaldehyde MOs
e.  C1 1
f.  C 0.0000000 0 0.0000000 0 0.0000000 0 0 0 0
g.  1.2069530   1 0.0000000 0 0.0000000 0 1 0 0
h.  H 1.0832433 1 122.52865 1 0.0000000 0 1 2 0
i.  H 1.0832433 1 122.52865 1 180.00000 1 1 2 3
j.   $END
```

A description of each of the lines is as follows (NOTE: we have added the a, b, c notation on the left of the input file for easy reference. This is not part of a GAMESS input file).

a. The first line is the CONTRoL line. It describes the method to be used and what calculation is requested. Note that the first character is a dollar sign($) for each of the lines with GAMESS commands on them! Note also that the lines with commands on them are indented. In this particular case, we want to run a *self-consistent field* calculation, with the run calculating the molecular energy – `SCFTYP=RHF RUNTYP=ENERGY`. Specifically, we want a Restricted Hartree-Fock calculation (RHF), the restricted meaning that all of the electrons are paired. The line continues by stating the molecule has no charge and a spin multiplicity of 1 (`ICHARG=0 MULT=1`). Finally, it says that the coordinate system to be used is a MOPAC-style *Z*-matrix (`COORD=ZMTMPC`). Recalling the previous chapter, input files can be of several types, and GAMESS typically defaults to describing the structure of molecules using this format. The line ends with the `$END` command.

b. The second line describes the basis set desired. In this case, we are asking for an STO-3G basis set, shown in the `GBASIS=STO NGAUSS=3` command structure. In this example, the "N" in "NGAUSS" refers to the number of Gaussians used. In this case, N=3.
c. This line states that what follows next is the data describing the geometry of the molecule.
d. This is the title of the job
e. Lines e through i describe the geometry of the molecule, using the MOPAC format. The term "C1 1" refers to the point group symmetry of the molecule. These are difficult to determine, but the WebMO molecular builder/editor will determine that for you. Readers should recognize what appears to be a X-Y-Z coordinate system. The other integer values are part of what is known as a connection table. Fortunately, modern users don't have to construct these by hand! WebMO is able to import different types of geometry input files, and it does the conversion for you.
f. Geometry continued
g. Geometry continued
h. Geometry continued
i. Geometry continued
j. `$END` ends the data section, and, in this case, ends the input file.

As was described in the previous chapter on Gaussian, the following statements also apply to GAMESS input files:

1. GAMESS and other researchers often publish example input files for various types of calculations.
2. Reviewing these files can be very instructive
3. One of the goals of using an interface such as WebMO, with its pulldown menus, is the avoidance of having to write these input files by hand
4. For the student researcher, understanding the structure of these input files allows the student to:
    a. Make sense of published literature articles
    b. Use GAMESS reference materials
    c. Even in the WebMO interface, customize a GAMESS job beyond the choices provide by the pull-down menus.

GAMESS Keywords:

GAMESS keywords are short, typically cryptic, instructions to the software that describe what the user wishes to do. Most of the keywords that are needed by the WebMO user on the North Carolina High School Computational Chemistry server are programmed into the various input windows as pull-down menus.

As before, one of the advantages of using the WebMO interface is that the user does *not* need to know the cryptic form of the keywords. It is helpful, however, for GAMESS users to understand the keyword concept. Students performing research projects will likely need to use one or more keywords (again, using the **Advanced** tab in WebMO) to customize GAMESS jobs.

Like most molecular modeling codes, there are some general categories of keywords. Each of the items below defines an input group, under which one would put one or several other specific keywords, keyword fragments, or other modifiers:

1. $CONTRL: this establishes the basic parameters of the calculation, as seen above in the sample input file
2. $SYSTEM: this optional input group describes the specifics of the run, such as the amount of memory to be used in the calculation, the time limit, and whether or not the code is running in parallel (more than one CPU). If not included (as is the case above), then GAMESS uses the built-in defaults.
3. $BASIS: as one would expect, this input group allows the user to define the basis set
4. $DATA: this group is used to define the geometry of the molecule. Also, if the user is using a basis set that is not considered to be standard, the non-standard basis set can be defined here

Some example keywords are as follows. These are shown as keyword fragments, meaning that the user would enter the entire fragment to perform the function:

- `$DFT DFTTYP=B3LYP:` this keyword fragment states that the user wishes to use the Density Functional Theory (DFT) method, using the Becke-3-Lee-Yang-Parr method.

- `$CONTRL RUNTYP=OPTIMIZE:` this keyword fragment instructs GAMESS to perform a geometry optimization on the molecule.
- `$BASIS GBASIS=N31 NGAUSS=6 NDFUNC=1:` use a 6-31+G basis set – the "+" referring to a diffuse basis set.
- `$ELDENS IEDEN=1:` this fragment requests that an electron density map be generated. The "1" after the keyword `IEDEN` is a toggle – 0 for off, 1 for on.
- `$ELPOT IEPOT=1:` this fragment requests that an electrostatic potential map be generated.

Interpreting GAMESS Output:

GAMESS has the reputation in the computational chemistry community of having the most verbose, and helpful, output files of all of the electronic structure software packages. This reading describes the various parts of a relatively simple output file. *In most cases*, the user of the North Carolina High School Computational Chemistry server will obtain *most* if not all of his or her calculation results from the standard output window provided by the WebMO software. There is, however, a great deal of information contained in the **Raw Output** file, available in the Job Manager window.

Readers are invited to skim this particular section, returning to it as needed for reference.

Comments or output information that is pertinent at this stage are in **bold**, embedded in the output. The actual output file is shown indented, using a slightly different font. Some sections may be omitted.

This shows that you are using GAMESS, the 2004 (most recent) version:

```
Distributed Data Interface kickoff program.
Initiating 1 compute processes on 1 nodes to run the following command:
/usr/local/gamess/gamess.00.x /var/webmo/gotwals/3860/input

          ******************************************************
          *         GAMESS VERSION = 22 NOV 2004 (R1)          *
          *             FROM IOWA STATE UNIVERSITY             *
          * M.W.SCHMIDT, K.K.BALDRIDGE, J.A.BOATZ, S.T.ELBERT, *
          *   M.S.GORDON, J.H.JENSEN, S.KOSEKI, N.MATSUNAGA,   *
          *          K.A.NGUYEN, S.J.SU, T.L.WINDUS,           *
          *       TOGETHER WITH M.DUPUIS, J.A.MONTGOMERY       *
          *         J.COMPUT.CHEM.  14, 1347-1363(1993)        *
          ****************** PC-UNIX VERSION ******************
```

We start! The following section is an echo of our input file, so indicated with the INPUT CARD notation:

```
EXECUTION OF GAMESS BEGUN Sat Jul  8 08:35:55 2006

           ECHO OF THE FIRST FEW INPUT CARDS -
INPUT CARD> $CONTRL SCFTYP=RHF RUNTYP=ENERGY
INPUT CARD>       ICHARG=0 MULT=1 COORD=ZMTMPC $END
INPUT CARD> $BASIS GBASIS=STO NGAUSS=3 $END
INPUT CARD> $DATA
INPUT CARD>Formaldehyde MOs
INPUT CARD>C1 1
INPUT CARD>C 0.0000000 0 0.0000000 0 0.0000000 0 0 0 0
INPUT CARD>O 1.2069530 1 0.0000000 0 0.0000000 0 1 0 0
INPUT CARD>H 1.0832433 1 122.52865 1 0.0000000 0 1 2 0
INPUT CARD>H 1.0832433 1 122.52865 1 180.00000 1 1 2 3
INPUT CARD> $END
INPUT CARD>
```

GAMESS is chatty about telling you what is happening:

```
..... DONE SETTING UP THE RUN .....
  40000000 WORDS OF MEMORY AVAILABLE

    BASIS OPTIONS
    -------------
    GBASIS=STO          IGAUSS=       3      POLAR=NONE
    NDFUNC=       0     NFFUNC=       0     DIFFSP=       F
    NPFUNC=       0      DIFFS=       F

    RUN TITLE
    ---------
Formaldehyde MOs
```

What is known as point group symmetry is described. The code thinks the symmetry is C1, but it is probably C_{2v}. We will not describe point group symmetry in this Guide:

```
THE POINT GROUP OF THE MOLECULE IS C1
THE ORDER OF THE PRINCIPAL AXIS IS     1
```

Thankfully, the geometry of the molecule is shown in a more normal *Z*-matrix fashion, as compared with that of the input geometry:

```
YOUR FULLY SUBSTITUTED Z-MATRIX IS
C
O      1   1.2069530
H      1   1.0832433  2  122.5286
H      1   1.0832433  2  122.5286  3  180.0000  0

THE MOMENTS OF INERTIA ARE (AMU-ANGSTROM**2)
IXX=    13.030   IYY=    14.712   IZZ=     1.681
```

The next section shows the same geometry using a standard X-Y-Z Cartesian coordinate system. Notice that this is a planar (flat) molecule, so most of the dimensions are on the Z-axis. There is no Y-axis component. Notice that the units are in units of Bohrs. One Bohr is 5.29 x 10^{-11} meters (0.529 Angstroms, Å):

```
ATOM      ATOMIC                    COORDINATES (BOHR)
          CHARGE         X                  Y                  Z
C           6.0     0.0000000000       0.0000000000      -1.1416869603
O           8.0     0.0000000000       0.0000000000       1.1391234897
H           1.0     1.7258999428       0.0000000000      -2.2424201487
H           1.0    -1.7258999428       0.0000000000      -2.2424201487
```

GAMESS shows the internuclear distances in units of Angstroms (Å, 10^{-10} meters):

```
          INTERNUCLEAR DISTANCES (ANGS.)
          ------------------------------
                    C             O             H             H
 1  C              0.0000000     1.2069530 *   1.0832433 *   1.0832433 *
 2  O              1.2069530 *   0.0000000     2.0090323 *   2.0090323 *
 3  H              1.0832433 *   2.0090323 *   0.0000000     1.8266140 *
 4  H              1.0832433 *   2.0090323 *   1.8266140 *   0.0000000
```

```
 * ... LESS THAN  3.000
```

The output explicitly lists the alpha values and contraction coefficients for the STO-3G basis set, for carbon, oxygen, and the two hydrogens:

```
    ATOMIC BASIS SET
    ----------------
THE CONTRACTED PRIMITIVE FUNCTIONS HAVE BEEN UNNORMALIZED
THE CONTRACTED BASIS FUNCTIONS ARE NOW NORMALIZED TO UNITY

SHELL TYPE PRIMITIVE    EXPONENT          CONTRACTION COEFFICIENTS

C

  1   S    1           71.6168373      0.154328967295
  1   S    2           13.0450963      0.535328142282
  1   S    3            3.5305122      0.444634542185

  2   L    4            2.9412494     -0.099967229187      0.155916274999
  2   L    5            0.6834831      0.399512826089      0.607683718598
  2   L    6            0.2222899      0.700115468880      0.391957393099

O

  3   S    7          130.7093214      0.154328967295
  3   S    8           23.8088661      0.535328142282
  3   S    9            6.4436083      0.444634542185

  4   L   10            5.0331513     -0.099967229187      0.155916274999
  4   L   11            1.1695961      0.399512826089      0.607683718598
  4   L   12            0.3803890      0.700115468880      0.391957393099

H

  5   S   13            3.4252509      0.154328967295
  5   S   14            0.6239137      0.535328142282
  5   S   15            0.1688554      0.444634542185

H

  6   S   16            3.4252509      0.154328967295
  6   S   17            0.6239137      0.535328142282
  6   S   18            0.1688554      0.444634542185
```

This next section shows an accounting of the number of basis functions (12), number of electrons (16), charge, multiplicity, spin up and spin down electrons, and, finally, the nuclear repulsion energy. Note that all of these values are the same as seen in the Gaussian output (shown in the previous chapter):

```
TOTAL NUMBER OF BASIS SET SHELLS             =    6
NUMBER OF CARTESIAN GAMESS BASIS FUNCTIONS   =   12
NUMBER OF ELECTRONS                          =   16
CHARGE OF MOLECULE                           =    0
SPIN MULTIPLICITY                            =    1
NUMBER OF OCCUPIED ORBITALS (ALPHA)          =    8
NUMBER OF OCCUPIED ORBITALS (BETA )          =    8
TOTAL NUMBER OF ATOMS                        =    4
THE NUCLEAR REPULSION ENERGY IS       31.4113829018
```

This next section describes the computational aspects of the run. Notice that we're running this job on one CPU (1 processor), and that 40 Megawords (MW, or 40000000 Words) of memory is available for running this job. The "PARALL= F" $SYSTEM option shows that this job is not being run on a parallel computing system. Larger jobs may require that some of these parameters be adjusted by the user:

```
    $CONTRL OPTIONS
    ---------------
SCFTYP=RHF          RUNTYP=ENERGY       EXETYP=RUN
MPLEVL=       0     CITYP =NONE         CCTYP =NONE
MULT  =       1     ICHARG=       0     NZVAR =       0     COORD =ZMTMPC
ECP   =NONE         RELWFN=NONE         LOCAL =NONE         NUMGRD=       F
ISPHER=      -1     NOSYM =       0     MAXIT =      30     UNITS =ANGS
PLTORB=       F     MOLPLT=       F     AIMPAC=       F     FRIEND=
NPRINT=       7     IREST =       0     GEOM  =INPUT
NORMF =       0     NORMP =       0     ITOL  =      20     ICUT  =       9
INTTYP=BEST         GRDTYP=BEST         QMTTOL= 1.0E-06

    $SYSTEM OPTIONS
    ---------------
 REPLICATED MEMORY=    40000000 WORDS (ON EVERY NODE).
DISTRIBUTED MEMDDI=           0 MILLION WORDS IN AGGREGATE,
MEMDDI DISTRIBUTED OVER   1 PROCESSORS IS           0 WORDS/PROCESSOR.
TOTAL MEMORY REQUESTED ON EACH PROCESSOR=    40000000 WORDS.
TIMLIM=      525600.00 MINUTES, OR   365.00 DAYS.
PARALL= F  BALTYP=  NXTVAL  KDIAG=    0  COREFL= F
```

This next section describes some of the programming commands. For brevity, we've deleted a large chunk of this information:

```
        ----------------
          PROPERTIES INPUT
          ----------------

    MOMENTS             FIELD            POTENTIAL           DENSITY
IEMOM =       1   IEFLD =       0   IEPOT =       0   IEDEN =       0
WHERE =COMASS     WHERE =NUCLEI     WHERE =NUCLEI     WHERE =NUCLEI
OUTPUT=BOTH       OUTPUT=BOTH       OUTPUT=BOTH       OUTPUT=BOTH
IEMINT=       0   IEFINT=       0                     IEDINT=       0
                                                    MORB  =       0
```

Now we see some of the results of our SCF (self-consistent field) calculation. Our nuclear repulsion energy is 31.41138 hartrees. One of the nice things of GAMESS output files is that it shows the data for each iteration. Notice that the energy of the molecule at the beginning is `-112.168549217.` **Through 9 iterations, we see the energy getting lower. Also notice the second column, where we see the change in energy. At the point where we have no change (0.0000000), the calculations stop. This process of reaching a point where the energy changes from one iteration to the next is known as *convergence*. In the Troubleshooting section, we show a run that fails to converge.**

```
        -------------------------
               RHF SCF CALCULATION
        -------------------------
    NUCLEAR ENERGY =        31.4113829018
    MAXIT =   30     NPUNCH=    2
    EXTRAP=T  DAMP=F  SHIFT=F  RSTRCT=F  DIIS=F  DEM=F  SOSCF=T
    DENSITY MATRIX CONV=  1.00E-05
    SOSCF WILL OPTIMIZE      32 ORBITAL ROTATIONS, SOGTOL=   0.250
```

```
    MEMORY REQUIRED FOR RHF STEP=     31304 WORDS.

ITER EX DEM     TOTAL ENERGY        E CHANGE  DENSITY CHANGE     ORB. GRAD
  1  0  0     -112.168549217    -112.168549217   0.540318378   0.000000000
         ---------------START SECOND ORDER SCF---------------
  2  1  0     -112.340149900    -0.171600683   0.124075012   0.082743342
  3  2  0     -112.350638373    -0.010488473   0.051006698   0.029083644
  4  3  0     -112.353551287    -0.002912914   0.002774858   0.001367645
  5  4  0     -112.353558921    -0.000007634   0.000726379   0.000287474
  6  5  0     -112.353559205    -0.000000284   0.000048597   0.000027764
  7  6  0     -112.353559210    -0.000000005   0.000029290   0.000009094
  8  7  0     -112.353559210    -0.000000001   0.000014179   0.000003300
  9  8  0     -112.353559211     0.000000000   0.000002417   0.000001070

         -----------------
         DENSITY CONVERGED
         -----------------
    TIME TO FORM FOCK OPERATORS=       0.0 SECONDS (       0.0 SEC/ITER)
    TIME TO SOLVE SCF EQUATIONS=       0.0 SECONDS (       0.0 SEC/ITER)

FINAL RHF ENERGY IS     -112.3535592105 AFTER   9 ITERATIONS
```

This next section shows the information about the 12 molecular orbitals, just as we saw in the Gaussian output. NOTE: we will not annotate these to the level of detail shown in the Gaussian chapter. Please refer to that chapter for details:

```
         ------------
         EIGENVECTORS
         ------------
                    1         2         3         4         5
                -20.3152  -11.1264   -1.3447   -0.8136   -0.6419
                   A         A         A         A         A
  1  C  1  S    0.000521  0.992581 -0.124271 -0.184826  0.000000
  2  C  1  S   -0.007394  0.033327  0.276941  0.569940  0.000000
  3  C  1  X    0.000000  0.000000  0.000000  0.000000  0.535073
  4  C  1  Y    0.000000  0.000000  0.000000  0.000000  0.000000
  5  C  1  Z   -0.006397  0.000468  0.158715 -0.232591  0.000000
  6  O  2  S    0.994269  0.000106 -0.218840  0.098791  0.000000
  7  O  2  S    0.026135 -0.005783  0.764515 -0.427376  0.000000
  8  O  2  X    0.000000  0.000000  0.000000  0.000000  0.437374
  9  O  2  Y    0.000000  0.000000  0.000000  0.000000  0.000000
 10  O  2  Z   -0.005760  0.001577 -0.173941 -0.159552  0.000000
 11  H  3  S    0.000217 -0.006684  0.032808  0.265787  0.298505
 12  H  4  S    0.000217 -0.006684  0.032808  0.265787 -0.298505

                    6         7         8         9        10
                -0.5493   -0.4491   -0.3559    0.2848    0.6499
                   A         A         A         A         A
  1  C  1  S    0.033861  0.000000  0.000000  0.000000 -0.208062
  2  C  1  S   -0.108262  0.000000  0.000000  0.000000  1.330286
  3  C  1  X    0.000000  0.000000 -0.180847  0.000000  0.000000
  4  C  1  Y    0.000000  0.610653  0.000000  0.821332  0.000000
  5  C  1  Z   -0.443907  0.000000  0.000000  0.000000 -0.461904
  6  O  2  S   -0.094886  0.000000  0.000000  0.000000  0.027491
  7  O  2  S    0.505605  0.000000  0.000000  0.000000 -0.157731
  8  O  2  X    0.000000  0.000000  0.872795  0.000000  0.000000
  9  O  2  Y    0.000000  0.672484  0.000000 -0.771524  0.000000
 10  O  2  Z    0.680751  0.000000  0.000000  0.000000  0.236111
```

```
 11  H  3  S    0.154589   0.000000  -0.358907   0.000000  -0.906199
 12  H  4  S    0.154589   0.000000   0.358907   0.000000  -0.906199

                   11         12
                 0.7546     0.9257
                   A          A
  1  C  1  S    0.000000  -0.096966
  2  C  1  S    0.000000   0.656626
  3  C  1  X    1.165341   0.000000
  4  C  1  Y    0.000000   0.000000
  5  C  1  Z    0.000000   1.179967
  6  O  2  S    0.000000   0.117274
  7  O  2  S    0.000000  -0.885175
  8  O  2  X   -0.321770   0.000000
  9  O  2  Y    0.000000   0.000000
 10  O  2  Z    0.000000   0.931305
 11  H  3  S   -0.857520   0.146887
 12  H  4  S    0.857520   0.146887
```

We're done. Notice that the GAMESS calculation took 0.1 seconds, as compared with 10.2 seconds for Gaussian! Classroom teachers are encouraged to take note of this difference when running jobs in laboratory classes:

```
 ...... END OF RHF CALCULATION ......
STEP CPU TIME =     0.01 TOTAL CPU TIME =        0.1 (    0.0 MIN)
TOTAL WALL CLOCK TIME=        0.0 SECONDS, CPU UTILIZATION IS 100.00%
```

GAMESS does a nice reporting of the contributions of the energy calculations to the final total energy. Our -V/T ratio (*virial theorem*) is not exactly 2.0, so we know that we didn't solve Schrödinger's equation exactly:

```
          -----------------
          ENERGY COMPONENTS
          -----------------

        WAVEFUNCTION NORMALIZATION =       1.0000000000

              ONE ELECTRON ENERGY =    -216.3373904965
              TWO ELECTRON ENERGY =      72.5724483841
         NUCLEAR REPULSION ENERGY =      31.4113829018
                                    ------------------
                    TOTAL ENERGY =    -112.3535592105

ELECTRON-ELECTRON POTENTIAL ENERGY =      72.5724483841
 NUCLEUS-ELECTRON POTENTIAL ENERGY =    -327.7395695989
  NUCLEUS-NUCLEUS POTENTIAL ENERGY =      31.4113829018
                                    ------------------
           TOTAL POTENTIAL ENERGY =    -223.7557383130
             TOTAL KINETIC ENERGY =     111.4021791024
               VIRIAL RATIO (V/T) =       2.0085400494

 ...... PI ENERGY ANALYSIS ......

ENERGY ANALYSIS:
           FOCK ENERGY=    -71.1924888293
         BARE H ENERGY=   -216.3373904965
    ELECTRONIC ENERGY =   -143.7649396629
        KINETIC ENERGY=    111.4021791024
         N-N REPULSION=     31.4113829018
```

```
          TOTAL ENERGY=   -112.3535567611
       SIGMA PART(1+2)=   -134.6135275556
             (K,V1,2)=    107.8483151961   -306.7810676479     64.3192248961
          PI PART(1+2)=     -9.1514121073
             (K,V1,2)=      3.5538639063    -20.9585019511      8.2532259374
 SIGMA SKELETON, ERROR=   -103.2021446538      0.0000000000
            MIXED PART= 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
....... END OF PI ENERGY ANALYSIS ......
```

As with Gaussian, we now get a *Mulliken population analysis*. This is the standard method used in most software packages for estimating where electrons are distributed throughout the molecule. GAMESS also includes a *Lowdin population analysis*, another popular method. The output shows both methods in comparison:

```
          ---------------------------------------
          MULLIKEN AND LOWDIN POPULATION ANALYSES
          ---------------------------------------

     MULLIKEN ATOMIC POPULATION IN EACH MOLECULAR ORBITAL

                        1         2         3         4         5

                   2.000000  2.000000  2.000000  2.000000  2.000000

    1             -0.001501  2.002605  0.505300  1.020206  0.925208
    2              2.001496 -0.000710  1.469012  0.323857  0.501888
    3              0.000002 -0.000948  0.012844  0.327969  0.286452
    4              0.000002 -0.000948  0.012844  0.327969  0.286452

                        6         7         8

                   2.000000  2.000000  2.000000

    1              0.455440  0.920662  0.100990
    2              1.404409  1.079338  1.409200
    3              0.070076  0.000000  0.244905
    4              0.070076  0.000000  0.244905

               ----- POPULATIONS IN EACH AO -----
                             MULLIKEN      LOWDIN
              1  C  1  S      1.99359     1.98707
              2  C  1  S      1.13183     1.03989
              3  C  1  X      1.02620     1.04257
              4  C  1  Y      0.92066     0.92248
              5  C  1  Z      0.85663     0.92888
              6  O  2  S      1.99808     1.99725
              7  O  2  S      1.86675     1.74520
              8  O  2  X      1.91109     1.91446
              9  O  2  Y      1.07934     1.07752
             10  O  2  Z      1.33324     1.37890
             11  H  3  S      0.94130     0.98289
             12  H  4  S      0.94130     0.98289

          ----- MULLIKEN ATOMIC OVERLAP POPULATIONS -----
          (OFF-DIAGONAL ELEMENTS NEED TO BE MULTIPLIED BY 2)

             1          2          3          4
```

```
   1    4.7277211
   2    0.4486310   7.7871609
   3    0.3762790  -0.0236505   0.6271622
   4    0.3762790  -0.0236505  -0.0384912   0.6271622

        TOTAL MULLIKEN AND LOWDIN ATOMIC POPULATIONS
     ATOM         MULL.POP.    CHARGE          LOW.POP.     CHARGE
   1 C             5.928910    0.071090         5.920895    0.079105
   2 O             8.188491   -0.188491         8.113326   -0.113326
   3 H             0.941299    0.058701         0.982889    0.017111
   4 H             0.941299    0.058701         0.982889    0.017111
```

This next section describes the bond orders and valence distributions. Notice that we have slightly more than a double bond between atoms 1 and 2 (the carbon and the oxygen), slightly less than single bonds between the carbon and the hydrogens. In the Total Valence chart, note the size of the bonds – not quite 4 for carbon, as are generally described:

```
          ------------------------------
          BOND ORDER AND VALENCE ANALYSIS     BOND ORDER THRESHOLD=0.050
          ------------------------------

                  BOND                        BOND                        BOND
ATOM PAIR DIST  ORDER      ATOM PAIR DIST  ORDER      ATOM PAIR DIST  ORDER
  1   2  1.207  2.036        1   3  1.083  0.937        1   4  1.083  0.937

                    TOTAL       BONDED        FREE
     ATOM           VALENCE     VALENCE     VALENCE
   1 C                3.909       3.909       0.000
   2 O                2.126       2.126       0.000
   3 H                0.997       0.997       0.000
   4 H                0.997       0.997       0.000
```

Dipole moments, entitled *electrostatic moments* here, are described:

```
          --------------------
          ELECTROSTATIC MOMENTS
          --------------------
POINT   1           X           Y           Z (BOHR)    CHARGE
                0.000000    0.000000    0.000000        0.00 (A.U.)
        DX          DY          DZ         /D/  (DEBYE)
    0.000000    0.000000   -1.513771    1.513771
```

Final wrap-up comments. We only used 1 MW (megawords) of memory, the job required 0.1 seconds of server time, and, most importantly, we got an "`EXECUTION OF GAMESS TERMINATED NORMALLY`**" message:**

```
...... END OF PROPERTY EVALUATION ......
STEP CPU TIME =     0.00 TOTAL CPU TIME =        0.1 (    0.0 MIN)
TOTAL WALL CLOCK TIME=        0.0 SECONDS, CPU UTILIZATION IS 100.00%
100000 WORDS OF DYNAMIC MEMORY USED
EXECUTION OF GAMESS TERMINATED NORMALLY Sat Jul  8 08:35:55 2006
DDI: 920 bytes (0.0 MB / 0 MWords) used by master data server.
---------------------------------------
CPU timing information for all processes
=======================================
0: 0.062990 + 0.009998 = 0.072988
---------------------------------------
ddikick.x: exited gracefully.
```

Troubleshooting GAMESS Jobs:

There are fundamentally *two* types of "failed" jobs in GAMESS:

1. jobs for which you actually get a "Failed" message, shown in red print in the WebMO Job Manager
2. jobs for which you get one or several results that are clearly not reasonable chemical answers

Both types of failed jobs are described here.

Failed Job Messages

Like any computational chemistry program, there are times when the job fails. This can happen for a variety of reasons. Failed jobs can be difficult to diagnose. For users of the North Carolina WebMO resource, support staff can help in diagnosing job failures, and keep a regular watch on the jobs running on the server.

For a failed job in GAMESS, clicking on the file icon under the **Actions** label of the Job Manager will give you a text-based listing of the entire output file. Scrolling to the bottom, information about the job failure is given. We have deleted some of the output in the example below. In this failed job, the calculation failed after 132.6 seconds (2.2 minutes) of run time, giving you the message "`EXECUTION OF GAMESS TERMINATED -ABNORMALLY- AT Wed Jul 12 06:47:00 2006`". Why did this job fail? GAMESS is relatively clear spoken about the problem in this particular case. It states that the "`ENERGY DID NOT CONVERGE...ABORTING HESSIAN`". In attempting to find the lowest energy value and the best geometry for this molecule, it could no longer continue calculating. When the program is doing a *self-consistent field* (SCF) calculation, it constructs a type of mathematical equation known as a *matrix*. In most cases, a specific type of matrix known as a *Hessian matrix* is used. For the mathematically oriented reader, a Hessian matrix is a square matrix constructed from the *second partial derivatives* of the energy of the molecule. For those readers who have not studied calculus, it is enough to known that a failing Hessian is typically a bad thing!

```
          ---------------------
          ELECTROSTATIC MOMENTS
          ---------------------

POINT   1           X           Y           Z (BOHR)    CHARGE
                0.000000    0.000000    0.000000        0.00 (A.U.)
        DX          DY          DZ         /D/  (DEBYE)
-3273.979737 -443.217695   90.957048 3305.095827
...... END OF PROPERTY EVALUATION ......
STEP CPU TIME =     0.59 TOTAL CPU TIME =      132.6 (    2.2 MIN)
TOTAL WALL CLOCK TIME=      133.8 SECONDS, CPU UTILIZATION IS  99.16%

ENERGY DID NOT CONVERGE...ABORTING HESSIAN

EXECUTION OF GAMESS TERMINATED -ABNORMALLY- AT Wed Jul 12 06:47:00 2006
695053 WORDS OF DYNAMIC MEMORY USED
STEP CPU TIME =     0.00 TOTAL CPU TIME =      132.6 (    2.2 MIN)
TOTAL WALL CLOCK TIME=      133.8 SECONDS, CPU UTILIZATION IS  99.16%
A fatal error occurr on DDI Process 0.
ddikick.x: application process 0 quit unexpectedly.
ddikick.x: Fatal error detected.
The error is most likely to be in the application, so check for
input errors, disk space, memory needs, application bugs, etc.
ddikick.x will now clean up all processes, and exit...
ddikick.x: Sending kill signal to DDI processes.
ddikick.x: Execution terminated due to error(s).
```

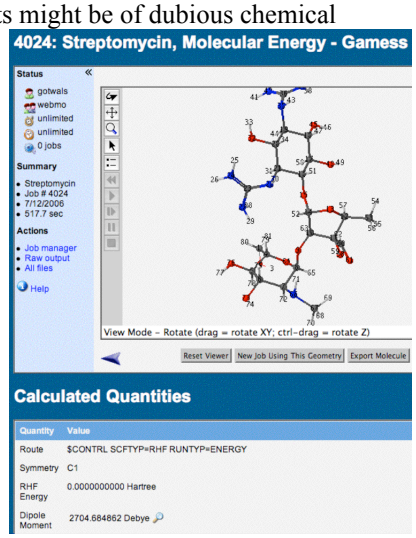
Given this particular error, what does one do? Sometimes simply restarting the calculation (using the **Restart** icon under the **Actions** option) will suffice. By way of example, we ran a GAMESS job on the drug molecule streptomycin. We downloaded this molecule as a PDB file (streptomycin.pdb) that we found through a Google search. Using the Import feature of the editor, we imported this molecule, and did a vibrational frequency calculation. That job failed. Our first thought was that there was something wrong with the structure we had imported, so we restarted the job and requested a geometry optimization, as an attempt to clean up the molecule. This too failed. We then went back to the molecular editor and opened the molecule. Using the **Comprehensive** option under the **Cleanup** menu, we adjusted the geometry of the molecule. What we noticed after doing so was that the builder was attempting to put charges on various atoms and, as a consequence, the entire molecule. Using the **Adjust** option under the **Tools** menu, followed by using the **Charges** option under the **Adjust** menu, we changed charged atoms to neutral atoms. At this point, we ran a **Molecular Energy** calculation under GAMESS, and the job ran to completion.

Streptomycin	Molecular Energy - Gamess	7/12/2006 8:48	Complete	8:37	
streptomycin	Geometry Optimization - Gamess	7/12/2006 7:18	Failed	2:13	
Streptomycin	Vibrational Frequencies - Gamess	7/12/2006 6:44	Failed	2:12	

A note especially to teachers: we recommend that you refer to Chapter 18: Gaussian for a discussion of specific run-time problems that are common to all of the software packages on the North Carolina High School Computational Chemistry server.

Jobs that provide questionable results

Even if GAMESS jobs run to completion, it is always possible that the results might be of dubious chemical accuracy. Outputs that show an enormous dipole moment, unusual energy values, large bond lengths and degrees, and other suspicious values should give the user pause. Users of *any* computational chemistry code should remember that all calculations use one or more approximations, and as such have some amount of error. Grossly distorted or unusual values, however, should encourage the user to disregard that job, even though one gets an "`EXECUTION OF GAMESS TERMINATED NORMALLY`" message. Unusual errors are most evident when a user or group of users are calculating some property of related molecules. For example, if a group of students are performing molecular energy calculations on a group of carboxylic acids, and one of the runs is significantly different from the others, a healthy dose of skepticism about that job is probably in order! Hopefully, re-doing that molecule and re-running the calculation should solve the problem.



Again, for North Carolina teachers and students, the computational chemistry support staff can provide assistance with results of questionable accuracy. Contact information is available on the main server page (http://www.shodor.org/chemistry).

GAMESS Support Tools:

The WebMO interface provides most if not all of the support functionality needed by GAMESS users. Readers are encouraged again to read the previous chapter on Gaussian. Many of the comments made regarding support tools for Gaussian also apply to GAMESS.

As described in the previous chapter, one of the most confusing aspects to new users is the variety of file formats used. Different codes require different types of formats, particularly regarding how the geometry of a molecule is described. Some codes use MOL format, some PDB, some XYZ Cartesian coordinates. WebMO can import and

export most formats generated by other codes. For example, using the WebMO **Export Molecule** feature, one can export the molecule in seven different formats, including:

1. MOL
2. PDB
3. XYZ
4. Gaussian
5. Gaussian Cartesian
6. MOPAC
7. MOPAC Cartesian

One tool that is mentioned frequently in the research literature is "Babel", a free software program that runs on all computers. Babel, so named for the Biblical references to many languages, is designed to convert any format to any other format. Users can download Babel and use it to convert files that cannot be directly imported into the WebMO interface. The most highly used formats, however, are available in WebMO. For North Carolina users, the support staff can assist with difficult or unusual format conversions.

Chapter 20: MOPAC

Key Notes:

MOPAC Basics:

MOPAC stands for **M**olecular **O**rbital **Pac**kage, and is primarily the work of J.P. Stewart, of the U.S. Air Force Academy. MOPAC is a very popular semi-empirical software package, and it is integrated into many commercial codes, such as CAChe. MOPAC runs all of the basic semi-empirical methods, including AM1, PM3, and MNDO/3. It has the ability to run basic quantum chemical calculations: geometry optimizations, molecular energies, molecular orbitals, vibrational frequencies, coordinate scans, thermodynamics (thermochemistry), and transition structures. In reporting energies, MOPAC reports heats of formation (in kcal/mole) instead of energies in hartrees. The North Carolina High School Computational Chemistry server uses MOPAC7. MOPAC 2003 is a commercial version of the software, while MOPAC7 is still available without charge. MOPAC is a useful tool for calculations on larger molecules, and, as compared with the other codes on the North Carolina server, runs calculations very quickly.

Running MOPAC Jobs:

MOPAC jobs run from the WebMO interface in exactly the same manner as Gaussian and GAMESS. The astute user will note that there is no option to choose a *basis set*. The user simply chooses the theory – PM3, AM1, or MNDO/3. The **Advanced** and the **Preview** tabs provide the advanced user with the option to customize and preview input files. Once the job is completed, clicking on the name of the job or the magnifying glass displays the **View Job** window with the text and graphical results. MOPAC jobs run very quickly as compared with Gaussian and GAMESS. As such, MOPAC is a good choice for large groups of students and/or larger molecules.

MOPAC Keywords:

MOPAC follows the keyword system found in most computational chemistry software packages. Keywords allow the user to customize an input file. In a system like WebMO, the *standard* keywords are built into the pull-down menus, and provide most of the functionality needed by most users.

Interpreting MOPAC Output:

MOPAC output files are significantly less detailed than that of Gaussian or GAMESS. As such, MOPAC output files are useful as a resource to help beginning users with interpreting output files. An annotated output file is included in this chapter.

Troubleshooting MOPAC Jobs:

MOPAC jobs run to completion a majority of the time, and the code is very stable. Error messages can be found in the **Raw output** file in the **Job Manager**, and are very specific about the type of error and keyword remedies to those errors.

MOPAC Basics:

MOPAC is a well-known semi-empirical software package, developed primarily by J. P. Stewart of the U.S. Air Force Academy. The WebMO server in North Carolina runs MOPAC7, the most recent non-commercial version of the software. MOPAC 2003 is found *bundled* in a number of commercial computational chemistry software packages, such as Fujitsu's CAChe program. The name has an interesting history, found on the developer's MOPAC page:

```
The name MOPAC should be understood to mean "Molecular Orbital
PACkage". The origin of the name is somewhat unusual, and might
be of general interest: The original program was written in
```

```
Austin, Texas. One of the roads in Austin is unusual in that the
Missouri-Pacific railway runs down the middle of the road. Since
this railway was called the MO-PAC, when names for the program
were being considered, MO-PAC was an obvious contender.
```

MOPAC7 runs several of the basic techniques found in semi-empirical quantum chemical methods:

1. AM1: the Austin Method 1 (developed at the University of Texas at Austin). AM1 semi-empirical methods can only calculate molecules with elements shown in the list at right.
2. PM3: the Parameterized Model 3, an improved version of AM1. PM3 methods will not calculate molecules with elements not on the elements list shown here.
3. MNDO and MNDO/3: these two methods are older semi-empirical methods. The North Carolina WebMO server provides access to MNDO/3.

ELEMENT	AM1	PM3
H	X	X
Li		X
Be		X
B	X	
C	X	X
N	X	X
O	X	X
F	X	X
Na		
Mg		X
Al	X	X
Si	X	X
P	X	X
S	X	X
Cl	X	X
Ca		
Fe		
Cu		
Zn	X	X
Ga		
Ge	X	X
As	X	
Se	X	
Br	X	X
Cd		
In		
Sn	X	X
Sb	X	
Te	X	
I	X	X
Hg	X	X
Tl		
Pb		X
Bi		

A review of the documentation for MOPAC (available from a variety of sites) lists the following as *capabilities* of MOPAC7. This list is provided *verbatim* here as a teaching tool. As your understanding of computational chemistry increases, the terminology on this list, and lists like it, are hopefully becoming less daunting! The following list of MOPAC capabilities may be used to assess your understanding of previous chapters in this Guide.

1. MNDO, MINDO/3, AM1, and PM3 Hamiltonians.
2. Restricted Hartree-Fock (RHF) and Unrestricted Hartree-Fock (UHF) methods.
3. Extensive Configuration Interaction
    a. 100 configurations
    b. Singlets, Doublets, Triplets, Quartets, Quintets, and Sextets
    c. Excited states
4. Geometry optimizations, etc., on specified states
5. Single SCF calculation
6. Geometry optimization
7. Gradient minimization
8. Transition structure location
9. Reaction path coordinate calculation
10. Force constant calculation
11. Normal coordinate analysis
12. Transition dipole calculation
13. Thermodynamic properties calculation
14. Localized orbitals
15. Covalent bond orders
16. Bond analysis into sigma and pi contributions
17. One dimensional polymer calculation
18. Dynamic Reaction Coordinate calculation
19. Intrinsic Reaction Coordinate calculation

Running MOPAC Jobs:

Running MOPAC jobs from the WebMO interface follows the standard procedure of:

1. building the molecule using the Java-based molecular editor
2. choosing MOPAC as the computational engine
3. selecting the desired calculation and theory (PM3, AM1, MNDO/3)
4. ensuring that the charge and multiplicity are set correctly

Using the **Advanced** tab, the user can change other parameters, and type in extra keywords to customize a job. The **Preview/Generate** capabilities allow the user to see the actual input file being sent to the MOPAC software. The input file for our molecular orbital calculation of formaldehyde, using the *PM3* method, is shown below:

```
a.  PM3 1SCF GRAPH VECTORS BONDS CHARGE=0 SINGLET
b.  Formaldehyde MOs
c.  C 0.0000000 0 0.0000000 0 0.0000000 0 0 0 0
d.  O 1.2070000 1 0.0000000 0 0.0000000 0 1 0 0
e.  H 1.0832466 1 122.52961 1 0.0000000 0 1 2 0
f.  H 1.0832466 1 122.52961 1 180.00000 1 1 2 3
```

A description of each of the lines is as follows (NOTE: we have added the a, b, c notation on the left of the input file for easy reference. This is not part of a MOPAC input file).

a. The first line describes the specifics of the calculation:
    1. `PM3`: the theory, or method, to be used. PM3 is the parameterized model 3, the most powerful semi-empirical method available in MOPAC
    2. `1SCF`: this keyword tells MOPAC to perform one complete *self-consistent field* (SCF) calculation. Hopefully the SCF calculation will converge.
    3. `GRAPH`: this keyword command tells MOPAC to generate data needed to create a variety of graphics. In WebMO, the graphics are viewable using the MOViewer program, which loads automatically by clicking on the various magnifying glass icons.
    4. `VECTORS`: this keyword command requests MOPAC to print out all of the *eigenvectors*, or *eigenvalues*, for the molecular orbitals.
    5. `BONDS`: this keyword command asks MOPAC to print out the *bond order* information found in the molecule.
    6. `CHARGE=0`: the charge on formaldehyde is still zero!
    7. `SINGLET`: the *spin multiplicity* of the formaldehyde molecule is 1, otherwise known as a singlet.
b. This is the title of the job, and came from the **Job Title** box in the WebMO interface
c. Lines c through f describe the geometry of the molecule, using the MOPAC format. The geometry input format is similar to a *Z*-matrix, along with a *connection table* on each line of the geometry data. Bond lengths are shown in angstroms (Å), and bond angles and dihedrals are shown in degrees.
d. Geometry continued
e. Geometry continued
f. Geometry continued

MOPAC Keywords:

MOPAC follows the method used by many molecular modeling packages in its use of a keyword notation. As described earlier, the most basic of these keywords are programmed into the WebMO interface, and are automatically inserted into the input file when you select an option from a pulldown menu.

There are three general categories of keywords in MOPAC:

1. *Control* keywords directly affect the chemical results, such as the calculated heat of formation (ΔH_f). Examples of control keywords are:
    a. AM1: use the AM1 method
    b. SADDLE: optimize the transition structure by attempting to find its *saddle point*
    c. THERMO: perform a *thermodynamics* calculation
    d. XYZ: instead of using the MOPAC geometry format, use a Cartesian XYZ format for the input file
2. *Output* keywords, which determine what information will be printed in the **Raw Output** file. Users should note that the output shown in the WebMO **View Job/Calculated Quantities** window is not changed if the user adds in output keywords by hand. Those quantities can only be viewed in the **Raw Output** file. Examples of output keywords are as follows:
    a. 1ELECTRON: this prints the *one-electron matrix*

b. DENSITY: print the *density matrix*
c. GRADIENTS: print all of the *gradients* for the heat of formation calculation
d. LARGE: prints an expanded output
e. MULLIK: print the *Mulliken population analysis* information

3. *Working* keywords, which affect how the job will be handled by the computer. Examples of working keywords are as follows:
   a. GNORM: this keyword allows the user to modify how precisely a geometry optimization calculation will be.
   b. ITRY=NN: this keyword stands for *iteration tries*, and is normally set (defaulted) to 200. Increasing this value requests that the calculation increase the number of iterations that it performs in trying to find the lowest energy value.
   c. DUMP: this keyword is sometimes useful if calculating a particularly large molecule. With this keyword, intermediary results are *dump*ed to a temporary file, to be used if the job fails at some point. The default dump time is 14400 seconds (four hours), but can be changed using DUMP=NN, where NN is time in seconds.

Interpreting MOPAC Output:

MOPAC has the reputation in the computational chemistry community of having verbose, and helpful, output files. This reading describes in basic detail the various parts of a relatively simple output file. *In most cases*, the user of the North Carolina High School Computational Chemistry server will obtain *most* if not all of his or her calculation results from the standard output window provided by the WebMO software. There is, however, a great deal of information contained in the **Raw Output** file, available in the Job Manager window.

Readers are invited to skim this particular section, returning to it as needed for reference.

Comments or output information that is pertinent are in **bold**, embedded in the output. The actual output file is shown indented, using a slightly different font. MOPAC files do not have large sections describing legal issues, citation suggestions, as are found in Gaussian and GAMESS:

This shows that you are using MOPAC, Version 7, still in the public domain:

```
 ******************************************************************************
 **                      MOPAC (PUBLIC DOMAIN)                                 **
 ******************************************************************************
                                PM3 CALCULATION RESULTS
 ******************************************************************************
 *          MOPAC:  VERSION  7.00              CALC'D. Sat Jul  8 08:56:41 2006
```

This section shows the keywords that WebMO automatically entered from the pulldown menus. In this job, we requested a molecular orbitals calculation using the PM3 *Hamiltonian*, on the molecule formaldehyde, with a charge of 0 and a spin multiplicity of singlet:

```
 *  VECTORS  - FINAL EIGENVECTORS TO BE PRINTED
 *  BONDS    - FINAL BOND-ORDER MATRIX TO BE PRINTED
 *  GRAPH    - GENERATE FILE FOR GRAPHICS
 *  SINGLET  - SPIN STATE DEFINED AS A SINGLET
 *
 *                 CHARGE ON SYSTEM =  0
 *
 *   T=      - A TIME OF 14400.0 SECONDS REQUESTED
 *  DUMP=N   - RESTART FILE WRITTEN EVERY 14400.0 SECONDS
 *  1SCF     - DO 1 SCF AND THEN STOP
 *  PM3      - THE PM3 HAMILTONIAN TO BE USED
```

```
*************************************************************************400BY400
```

This next section repeats your control line and title from your input file:

```
PM3 1SCF GRAPH VECTORS BONDS CHARGE=0 SINGLET
Formaldehyde MOs
```

This next section shows the geometry of the molecule, with bond lengths in angstroms (Å) and bond angles in degrees. The geometry is shown with a MOPAC coordinate system as well as in Cartesian (XYZ) coordinates. The reader is encouraged to try to decipher the notational system being used here:

```
  ATOM   CHEMICAL  BOND LENGTH    BOND ANGLE     TWIST ANGLE
 NUMBER  SYMBOL    (ANGSTROMS)     (DEGREES)      (DEGREES)
  (I)                  NA:I          NB:NA:I      NC:NB:NA:I     NA   NB   NC

    1      C
    2      O         1.20695  *                                  1
    3      H         1.08324  *     122.52865  *                 1    2
    4      H         1.08324  *     122.52865  *  180.00000  *   1    2    3

         CARTESIAN COORDINATES

  NO.       ATOM         X         Y         Z
   1         C         .0000     .0000     .0000
   2         O        1.2070     .0000     .0000
   3         H        -.5825     .9133     .0000
   4         H        -.5825    -.9133     .0000
```

The point group symmetry of the molecule – C_{2v} – is identified:

```
    MOLECULAR POINT GROUP   :   C2V
H: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
C: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
O: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
```

The output states that we are doing a *Restricted Hartree-Fock* calculation, with all electrons paired, and that there are six orbitals containing 12 electrons. MOPAC ignores the first two orbitals; we actually have 8 occupied orbitals, each with 2 electrons, for a total of 16 electrons:

```
          RHF CALCULATION, NO. OF DOUBLY OCCUPIED LEVELS =  6
```

This table shows the *interatomic distances* in units of angstroms (Å):

```
          INTERATOMIC DISTANCES
    C  1       O  2       H  3       H  4
-----------------------------------------------------
    C    1    .000000
    O    2   1.206953    .000000
    H    3   1.083243   2.009032    .000000
    H    4   1.083243   2.009032   1.826614    .000000

-------------------------------------------------------------------------------
```

Once more, a reminder of what is being requested:

```
PM3 1SCF GRAPH VECTORS BONDS CHARGE=0 SINGLET
Formaldehyde MOs
```

This next section fundamentally amounts to a "success" message. The program was able to converge on a heat of formation, using a *self-consistent field* calculation. As such, it did not need to use the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) optimizer. The BFGS optimizer is an alternative way of determining the best, or optimized, geometry of a molecule:

```
1SCF WAS SPECIFIED, SO BFGS WAS NOT USED
SCF FIELD WAS ACHIEVED

                          PM3    CALCULATION
                                                VERSION  7.00
                                                Sat Jul  8 08:56:41 2006
```

This next section shows our first results. Notice that we get the energy of the molecule in terms of heats of formation, in units of kcal per mole. For users new to molecular modeling, these are terms much more familiar than esoteric terms such as "hartrees". As such, MOPAC is often a good starting tool for novice molecular modelers. Note also, in the output below, other energy values are reported in units of electron-volts (eV). Total energy is equal to electronic energy plus core-core repulsion energy (T.E. = E.E. + C-C.E):

```
FINAL HEAT OF FORMATION =        -33.97167 KCAL

TOTAL ENERGY            =       -442.70353 EV
ELECTRONIC ENERGY       =       -834.28321 EV
CORE-CORE REPULSION     =        391.57968 EV
```

The next line states that the Highest Occupied Molecular Orbital (HOMO) has an energy value (in kcal/mole) of 10.65128. *Koopman's Theorem* states that the value of the HOMO is equal to the *ionization potential* (IP) of the molecule, and it is the ionization potential that is reported here. The ionization potential represents the energy needed to remove an electron from the outermost orbital of the molecule:

```
IONIZATION POTENTIAL    =         10.65128
```

If you are asking why there are only 6 filled levels (total of 12 electrons) and not 8 (total of 16 electrons), remember that MOPAC only shows the valence electrons:

```
NO. OF FILLED LEVELS    =          6
```

It's nice of MOPAC to do this math for us!

```
MOLECULAR WEIGHT        =     30.026
```

As requested with the *1SCF* keyword, MOPAC did one complete SCF calculation, and it converged. Note the computation time. As we described in the "Computational Analogy" chapter, software programs are analogous to ovens, stoves, microwave ovens, and grills. MOPAC is a microwave!

```
SCF CALCULATIONS  =               1
COMPUTATION TIME =    .000 SECONDS
```

We did not ask MOPAC to optimize this molecule, so it reports back that the bond lengths, angles, and dihedrals are as they were previously. Cartesian coordinates and interatomic distances are also reported, also all unchanged:

```
  ATOM   CHEMICAL  BOND LENGTH    BOND ANGLE     TWIST ANGLE
 NUMBER  SYMBOL    (ANGSTROMS)     (DEGREES)      (DEGREES)
  (I)                  NA:I          NB:NA:I       NC:NB:NA:I     NA   NB   NC

    1      C
    2      O         1.20695  *                                    1
    3      H         1.08324  *     122.52865  *                   1    2
    4      H         1.08324  *     122.52865  *  180.00000  *     1    2    3


          INTERATOMIC DISTANCES
0
                 C  1       O  2       H  3       H  4
 ------------------------------------------------------
    C    1    .000000
    O    2   1.206953    .000000
    H    3   1.083243   2.009032    .000000
    H    4   1.083243   2.009032   1.826614    .000000

     MOLECULAR POINT GROUP   :   C2V
```

The output file now reports out the *eigenvectors*, the details of the molecular orbitals. Unlike the other output files (Gaussian and GAMESS), we only get 10 orbitals here, not 12. As before, the output shows us the A1-B1-B2 symmetry of the orbital; the energy in kcal/mole; and a detailed description of the distribution of electrons in each of the orbitals. By way of reminder, in MO-1 ("Root" 1), most of the C atom is found in the s orbital, most of the O atom is also s, and both of the hydrogens are s orbitals. In Root 5 (MO-5), most of the carbon atom is in the p_z orbital, as is the oxygen. The hydrogens are still in s orbitals:

```
              EIGENVECTORS

 Root No.     1       2       3       4       5       6       7       8
            1 A1    2 A1    1 B2    3 A1    1 B1    2 B2    2 B1    4 A1
         -38.019 -24.326 -17.087 -16.430 -14.279 -10.651    .790   2.747
S   C  1   .4747  -.6029   .0000  -.0017   .0000   .0000   .0000   .6121
Px  C  1   .2739   .2772   .0000   .5400   .0000   .0000   .0000  -.1649
Py  C  1   .0000   .0000   .6234   .0000   .0000  -.2740   .0000   .0000
Pz  C  1   .0000   .0000   .0000   .0000  -.5951   .0000   .8037   .0000
S   O  2   .7807   .4739   .0000  -.3095   .0000   .0000   .0000  -.0588
Px  O  2  -.2201   .2349   .0000  -.7181   .0000   .0000   .0000   .2202
Py  O  2   .0000   .0000   .5818   .0000   .0000   .7882   .0000   .0000
Pz  O  2   .0000   .0000   .0000   .0000  -.8037   .0000  -.5951   .0000
S   H  3   .1444  -.3740   .3693  -.2201   .0000  -.3896   .0000  -.5226
S   H  4   .1444  -.3740  -.3693  -.2201   .0000   .3896   .0000  -.5226

 Root No.     9      10
            3 B2    5 A1
           4.155   6.655
S   C  1   .0000   .1911
Px  C  1   .0000   .7275
Py  C  1  -.7323   .0000
Pz  C  1   .0000   .0000
S   O  2   .0000  -.2581
Px  O  2   .0000   .5763
Py  O  2   .2004   .0000
Pz  O  2   .0000   .0000
S   H  3   .4602   .1331
S   H  4  -.4602   .1331
```

MOPAC shows the *partial charges* and the *electron density* on each of the atoms:

```
                NET ATOMIC CHARGES AND DIPOLE CONTRIBUTIONS

           ATOM NO.   TYPE          CHARGE        ATOM  ELECTRON DENSITY
             1          C           .2996          3.7004
             2          O          -.3100          6.3100
             3          H           .0052           .9948
             4          H           .0052           .9948
```

Dipole moments, in units of debyes, are shown, with a breakdown of that distribution:

```
DIPOLE           X         Y         Z       TOTAL
POINT-CHG.    -1.826      .000      .000     1.826
HYBRID         -.349      .000      .000      .349
SUM           -2.175      .000      .000     2.175

    CARTESIAN COORDINATES

   NO.       ATOM              X         Y         Z
    1         C                  .0000     .0000     .0000
    2         O                 1.2070     .0000     .0000
    3         H                 -.5825     .9133     .0000
    4         H                 -.5825    -.9133     .0000
```

MOPAC shows how electrons are distributed across each of the 10 orbitals. If we add up these values, we come up with the value of 12, the number of valence electrons in the formaldehyde molecule:

```
            ATOMIC ORBITAL ELECTRON POPULATIONS

  1.17769    .88696    .92750    .70824   1.85982   1.23873   1.91968   1.29176
   .99481    .99481
```

As was requested in the input file (without us asking it to do so!), MOPAC gives us a *bond order* report. Note especially the *degrees of bonding* information: It reports that carbon has approximately 4 bonds, roughly what we typically state. Likewise, oxygen has approximately 2 bonds, and the hydrogens have almost 1 (0.935034):

```
            BONDING CONTRIBUTION OF EACH M.O.

  1.1280  1.2835  1.4365  1.3682  1.8297   .8712 -1.8297 -2.1644 -1.9917 -1.9313
  nelecs nclose nopen nopn   12    6    6    0

  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
          *          STATISTICAL POPULATION ANALYSIS         *
  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                  DEGREES OF BONDING
0
                 C  1      O  2      H  3      H  4
 ---------------------------------------------------
    C    1   3.547085
    O    2   1.983610  10.556429
    H    3    .935034    .039974    .989652
```

```
     H    4    .935034    .039974    .024966    .989652

      SELF-Q    ACTIV-Q   TOTAL-Q   VALENCE   FREE-VA STAT.PROM MULL.PROM

1   -.15330   3.85368   3.70038   3.85368    .00000   -.29962   -.29962
2   4.24644   2.06356   6.30999   2.06356    .00000    .30999    .30999
3   -.00516    .99997    .99481    .99997    .00000   -.00519   -.00519
4   -.00516    .99997    .99481    .99997    .00000   -.00519   -.00519
```

MOPAC tells us that all of the electrons are paired (*closed shell*), that data was written for us to view graphics (available from the WebMO interface), and that the job was done in 0.00 seconds:

```
CLOSED SHELL

          DATA FOR GRAPH WRITTEN TO DISK

TOTAL CPU TIME:              .00 SECONDS

== MOPAC DONE ==
```

<u>Troubleshooting MOPAC Jobs</u>:

There are fundamentally *two* types of "failed" jobs in MOPAC:

1. jobs for which you actually get a "Failed" message (in red!) in the WebMO Job Manager
2. jobs for which you get one or several results that are clearly not reasonable chemical answers

Both types of failed jobs are described here.

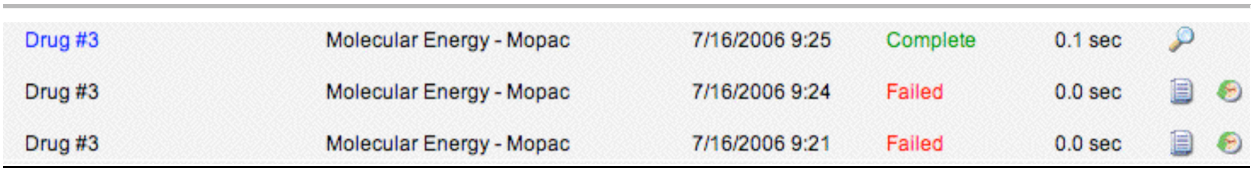
<u>Failed Job Messages</u>

Like any computational chemistry program, there are times when the job fails. This can happen for a variety of reasons. Failed jobs can be difficult to diagnose. For users of the North Carolina WebMO resource, support staff can help in diagnosing job failures, and keep a regular watch on the jobs running on the server.

For this section, we purposefully created two jobs that would probably fail, and they did:

1. a drug compound that contains a peptide group , defined as a H-N-C=O group in a molecule. MOPAC doesn't like peptide groups, and says so. In the **Raw output** file, it says "this system contains –HNCO- groups. It also tells us what to do. In the **Advanced** menu of the **Job Manager**, we can add the keyword "MMOK", which says it is "OK" to use Molecular Mechanics methods to calculate this part of the molecule. Fundamentally, all of the semi-empirical methods (PM3, AM1, MNDO, and MNDO/3) underestimate the way that peptide bonds rotate, and the molecular mechanics fix allows the code to adjust for that reality.

This graphic shows our attempts at running this molecule. The first time we ran the molecule as it had been built. We then tried to run it again after cleaning up the geometry in the builder. Finally, we added the "MMOK" keyword in the **Advanced** keywords box, and it ran to completion:



The code fragment below shows us that MOPAC has found a peptide group (HNCO), and what to do about it!

```
    MOLECULAR POINT GROUP   :   C1
 H: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
 C: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
 N: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).
 O: (PM3): J. J. P. STEWART, J. COMP. CHEM.     10, 209 (1989).


         RHF CALCULATION, NO. OF DOUBLY OCCUPIED LEVELS = 26
THIS SYSTEM CONTAINS -HNCO- GROUPS.
       YOU MUST SPECIFY "NOMM" OR "MMOK" REGARDING MOLECULAR MECHANICS CORRECTION
```

2. For the second example, we ran a MOPAC job on the drug molecule streptomycin. We downloaded this molecule as a PDB file (streptomycin.pdb) that we found through a Google search. We simply imported it into the molecular editor and attempted to run a MOPAC energy calculation on it. The job failed, and the raw output stated that it was suspicious of the geometry of the molecule. Assuming we have faith in the structure of the molecule that we had downloaded, we can run the molecule again, putting the keyword "GEO-OK" in the **Extra keywords** box under the **Advanced** tab. As you might guess, this is telling MOPAC that the geometry is "OK". Keep in mind that the geometry might *not* be OK, and whatever results you get are based on that!

```
 ATOMS 17 AND 10 ARE SEPARATED BY   .4725 ANGSTROMS.
 TO CONTINUE CALCULATION SPECIFY "GEO-OK"
```

A note especially to teachers: we recommend that you refer to Chapter 18: Gaussian for a discussion of specific run-time problems that are common to all of the software packages on the North Carolina High School Computational Chemistry server.

Jobs that provide questionable results

Readers are encouraged to read the discussions on questionable results in the previous two chapters (Gaussian and GAMESS). As with any molecular modeling software programs, it *is* possible to get success completions with absolutely bogus results. We leave you with this excerpt from Henry Schaefer's Quantum Chemistry: The Development of *Ab Initio* Methods in Molecular Electronic Structure Theory (Dover Press, 1984). Note, by the way, the discussion of "deck of computer cards" and the references to Gaussian 70!

> Nevertheless, one must concede that there are great perils associated with the application of [molecular modeling] methods to chemical problems. When an experimentalist turns in his latest new compound to the departmental n.m.r. facility, at least he is reasonably certain that the output will be an n.m.r. spectrum. He may have to be careful that the n.m.r. spectrum refers to a single compound rather than a mixture, but this is an uncertainty that chemists have learned to deal with. In striking contrast, it is very easy to submit a deck of computer cards to a standard quantum mechanical program such as GAUSSIAN 70 and receive as output totally meaningless results. In fact, it is probably not an exaggeration to state that there are literally hundreds of such error-ridden calculations which have actually been published in the chemical literature. A cynical view of this situation is provided by the statement that 'running a few molecular orbital calculations doesn't make one an electronic structure theorist any more than sleeping in one's garage makes him/her an automobile' (anonymous, 1981).

Again, for North Carolina teachers and students, the computational chemistry support staff can provide assistance with results of questionable accuracy. Contact information is available on the main server page (http://www.shodor.org/chemistry).

Chapter 21: Web-Based Tools

Key Notes:

WebMO

WebMO is a web-based interface available in both freeware and commercial (Pro) versions that provides simple molecule building, set-up of calculations, and viewing of results. The program provides an easy-to-use interface to a variety of computational chemistry software packages, such as Gaussian, GAMESS, MOPAC, and Tinker. WebMO Pro, along with these four computational chemistry programs, is available on the North Carolina High School Computational Chemistry server. For more information on WebMO, see: http://www.webmo.net/.

Gaussian Basis Set Order Form

The Molecular Science Computing Facility of the Environmental and Molecular Sciences Laboratory (EMSL) is a part of the Pacific Northwest Laboratory located in Richland, Washington. This National Laboratory is funded by the U.S. Department of Energy. The EMSL Basis Set Order Form allows the user to extract Gaussian basis sets, formatted appropriately for a wide variety of popular computational chemistry software packages, including Gaussian and GAMESS. The site is accessible at: http://www.emsl.pnl.gov/forms/basisform.html.

Computational Chemistry Comparison and Benchmark Database

A service of the National Institute of Standards and Technology (NIST), the Computational Chemistry Comparison and Benchmark Database (CCCBDB) provides a collection of both experimental and calculated thermochemical parameters for a selected set of molecules. These data are provided to help software developers evaluate program performance and to compare the results from different computational methods. The site also provides a useful source of data for educators and their students. The site is available at: http://srdata.nist.gov/cccbdb/.

Protein Data Bank

The Protein Data Bank (PDB) is a repository of structural information on large biological molecules such as proteins and nucleic acids. The structures have mostly been solved using single crystal X-ray diffraction, nuclear magnetic resonance (NMR), and cryo-electron microscopy (cryo-EM) techniques. A detailed understanding of the 3-D structure of these biomolecules can lead to an understanding of their function in a living organism. Atomic coordinates of the structures (in .pdb file format) can also be downloaded for use in computational chemistry programs. The Protein Data Bank web site is available at: http://www.rcsb.org/pdb/home/home.do.

Molecular Libraries

Reciprocal Net (http://www.reciprocalnet.org/) is funded by the U.S. National Science Foundation (NSF) as part of the National Science Digital Library (NSDL) project. Reciprocal Net maintains a distributed database of crystallographic information focused on smaller molecules of general interest. The structural data can be viewed in a variety of formats, and the data files can be downloaded and imported into programs such as WebMO so that calculations can be performed.

The Mathmol library has 3-D structures for molecules that are often presented in introductory biology and chemistry textbooks. These structures can be downloaded (.pdb file format) and imported into WebMO. The library can be accessed at: http://www.nyu.edu/pages/mathmol/library/.

Another site that has downloadable structural files is 3DChem.com. Inorganic, organic, therapeutic drug, and biomolecular structures can be downloaded (.mol file format) and imported. The site is available at: http://www.3dchem.com/.

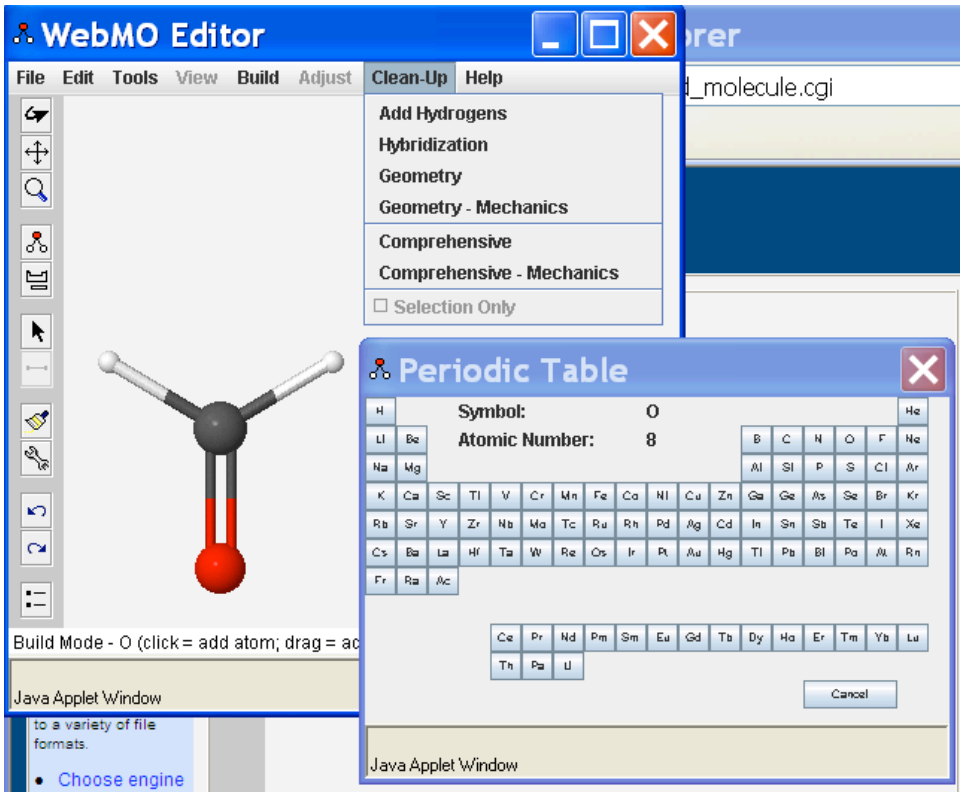
Computational Chemistry for Chemistry Educators

Computational Chemistry for Chemistry Educators (CCCE) is the companion Web-based resource for the computational chemistry workshops offered through the Shodor Education Foundation's National Computational Science Institute (NCSI). Funding for NCSI was provided (in part) by the Burroughs Wellcome Fund and the NSF. CCCE is designed to provide educators with sufficient information so they can begin to use molecular modeling tools in the chemistry classroom to improve the education of their students. Perhaps the most useful aspect of this site is the collection of computational laboratory exercises available for various software packages, including WebMO. The site can be accessed at: http://www.computationalscience.org/ccce/.

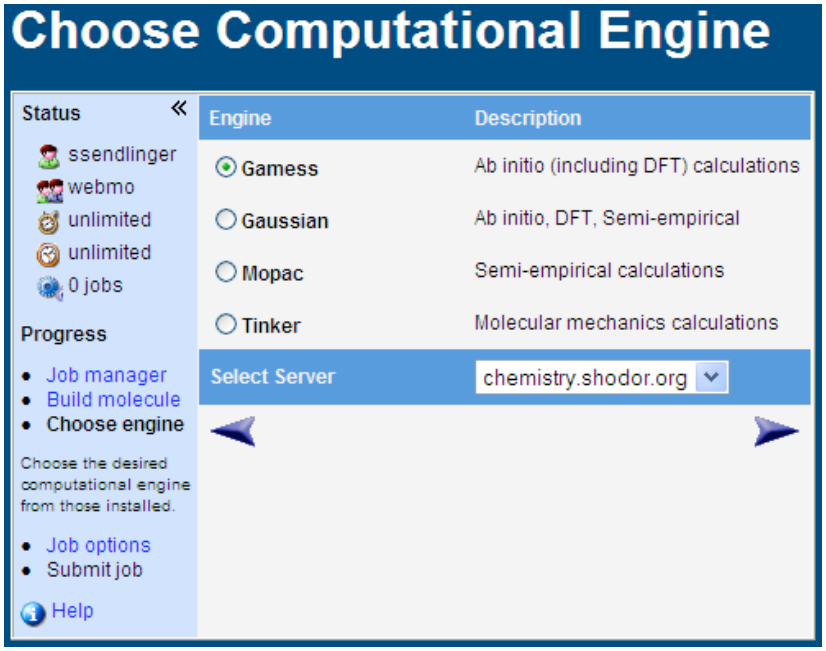
WebMO

WebMO is a free download that allows users to access computational chemistry software via a Windows, Macintosh, Unix, or Linux based web browser. WebMO Pro is a commercial product that lets the user edit the Z-matrix and also allows visualization of molecular orbitals, electron densities, electrostatic potentials, and electrophilic or nucleophilic susceptibility surfaces. The screen shots that follow were made using WebMO Pro.

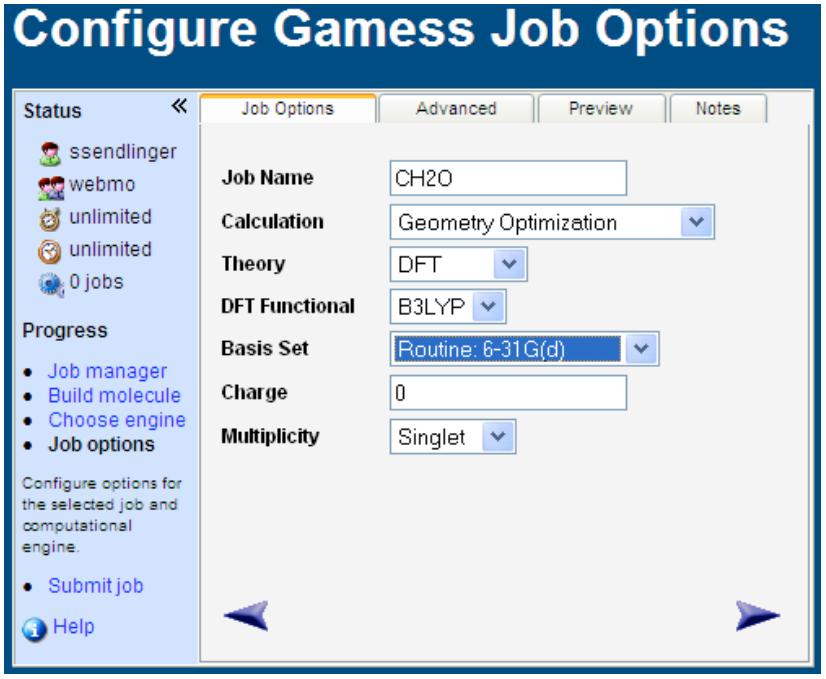
WebMO is loaded onto a server that also has at least one computational chemistry software package. Current software supported includes Gaussian, GAMESS, MOPAC, Tinker, MolPro, NWChem, and QChem. (Note that GAMESS, MOPAC, Tinker, and NWChem are *freely available* downloads). WebMO allows users to login via a web browser. Molecules can be easily drawn using a 3D java editor or can be imported in various common file formats (.pdb, .mol, .sdf, and .xyz) or in the formats produced by any of the aforementioned computational chemistry packages. The editor allows rotation, translation, and magnification of a molecule as well as display and adjustment of bond lengths, bond angles, and dihedral angles. A "clean-up" feature will add hydrogen atoms, set hybridization, and give idealized bond lengths and angles. Molecular mechanics is available in the editor to help with more complex structures. A view of the editor window with the "Clean-Up" menu, Periodic Table (for atom selection), and a formaldehyde molecule ($H_2C{=}O$) that was built is shown below:



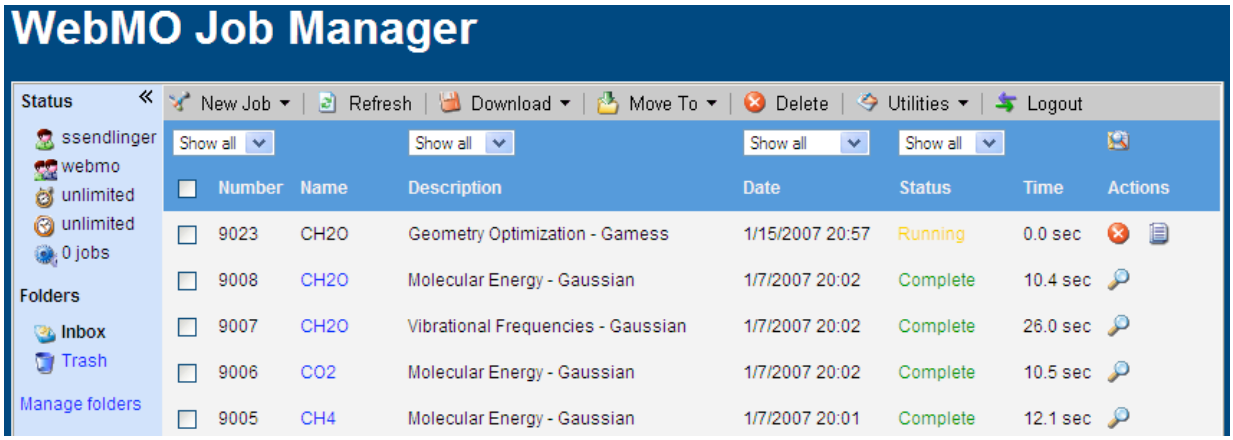
Once the molecule has been constructed, the computational software is chosen by simply clicking a button next to the name. The same screen also allows the server to be chosen if more than one is available (see below).



The final screen allows the user to choose a job name, specify the type of calculation to be performed, and pick the theory, DFT functional, and basis set to be used (if DFT or *ab initio*). The species charge and the spin multiplicity can also be entered (see below). An advanced window can be used to set molecular symmetry, choose solvents for solution-phase studies, and provide additional keywords that are specific to the computational software to be used.



The job is then submitted and will be listed in the WebMO Job Manager queue (shown below) where its status (queued, running, complete, or failed) is shown. The results of a successful run are viewed by clicking on the file name. The calculated quantities are displayed and links to the raw output and other files are provided. Using the Job Manager, a user can create folders to store results, pick molecules to run additional calculations on, and delete files.



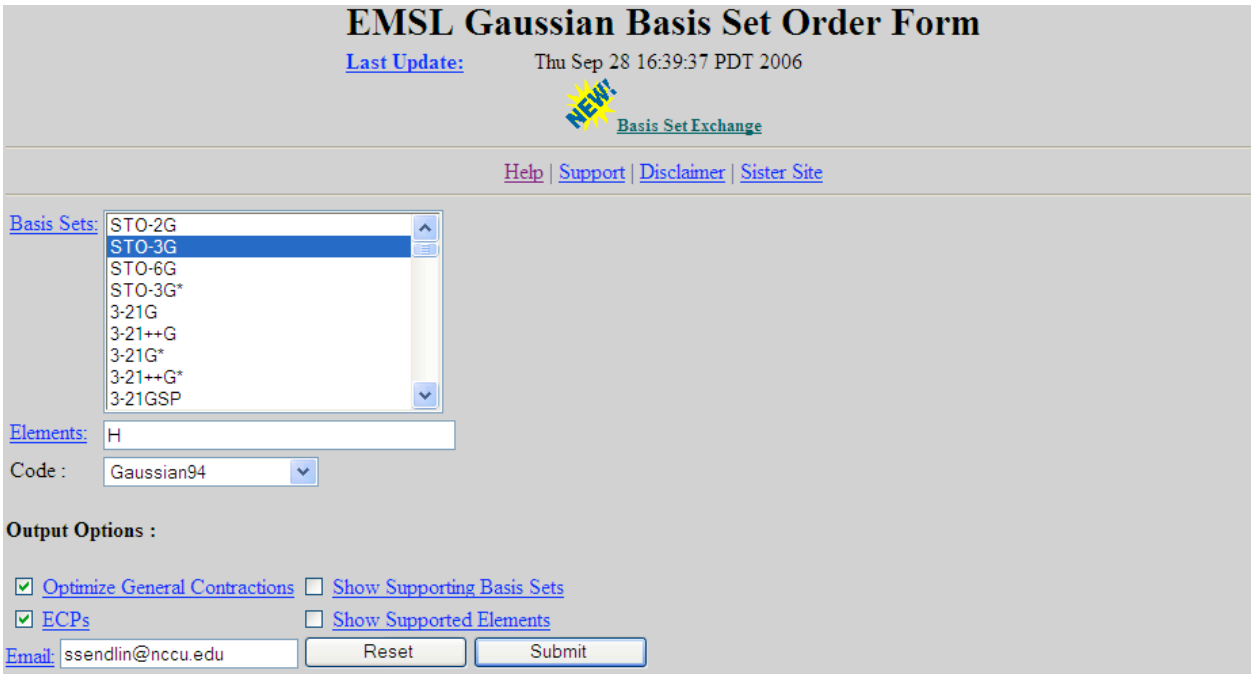
WebMO is simple and intuitive to use. The software is loaded onto one server and is available to anyone who has login credentials and a computer with web-access. Administrative tasks, such as setting up user accounts, are also done via web-accessible screens in a straightforward manner. A class list can even be imported to create new users. A free version of WebMO is available, and it can be used with free computational engines (GAMESS, MOPAC, Tinker, and NWChem). The only cost is that associated with a single server.

The use of WebMO makes computational chemistry easier for beginners with its shallow learning curve. Instructors can perform calculations and access the results in any location that has a computer with web access. Since the software is all placed on one machine, it lowers the cost associated with having a dedicated computer lab with multiple copies of licensed computational software. System maintenance time and expense are both greatly reduced.

A WebMO Demonstration Server allows users to run small jobs to earn how the software works. The server can be accessed at: http://www.webmo.net/demo/index.html.

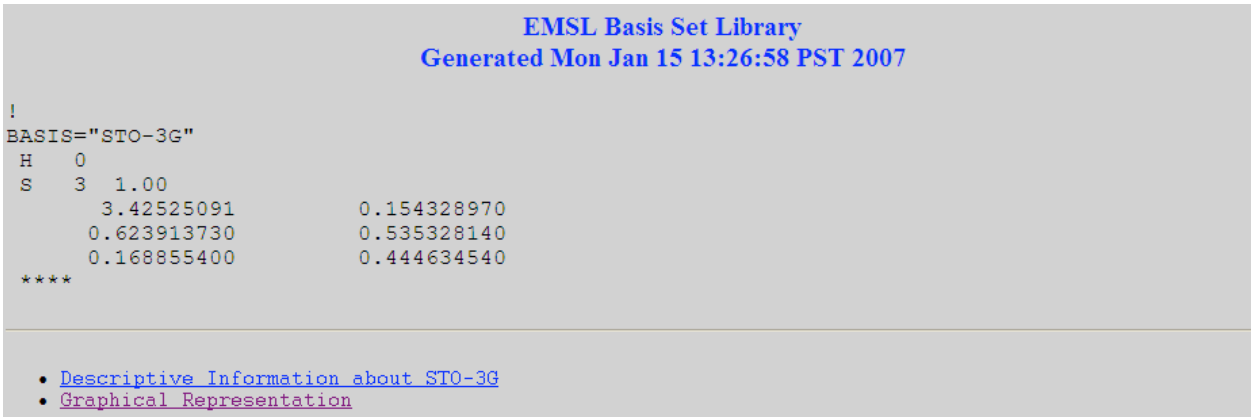
Gaussian Basis Set Order Form

Chapter 4 describes the importance and construction of the basis sets used in *ab initio* calculations. Most computational chemistry software packages will have several different basis sets included with them to choose from. A large number of basis sets have been reported in the literature and many have been collected in one location by the Molecular Science Computing Facility of the Environmental and Molecular Sciences Laboratory (EMSL), part of the Department of Energy's Pacific Northwest Laboratory. The EMSL Gaussian Basis Set Order Form (see below, available at: http://www.emsl.pnl.gov/forms/basisform.html) allows a large number of basis sets for many different elements to be downloaded in formats appropriate for a variety of computational chemistry software packages, including GAMESS, Gaussian, Molpro, and NWChem.

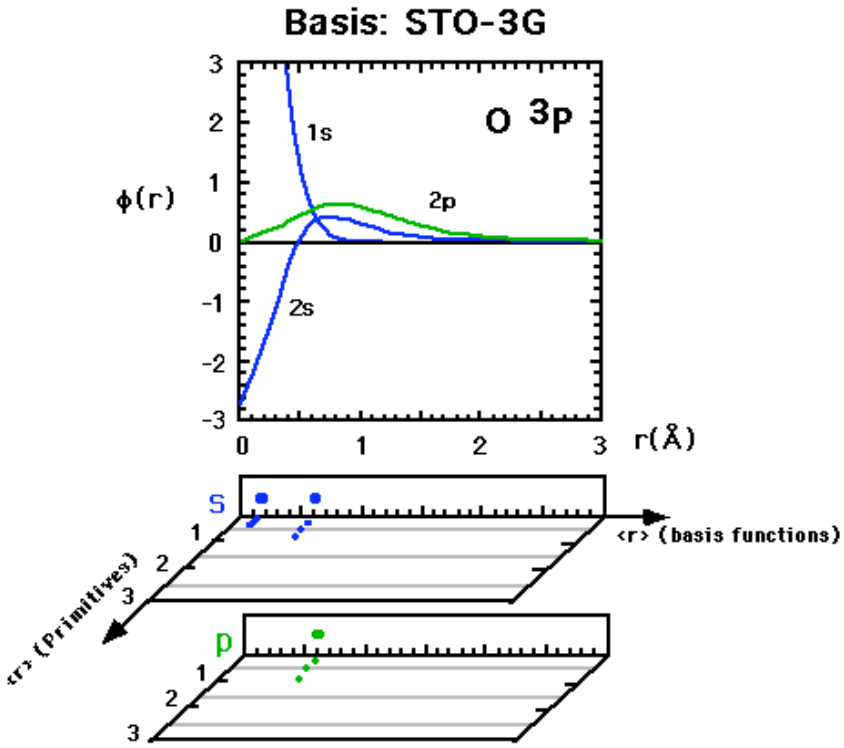


Once the basis set, element of interest, and software type is chosen, the user's e-mail is provided (in case errors are discovered in the data) and the form is submitted. The basis set data appears in a new window, as shown below.

STO-3G Data in Gaussian 94 format for Hydrogen



The columns of numbers on the left are the alpha values (used in the exponent) and the second column contains the contraction coefficients (See Chapter 4). Additional information about the basis set, including original literature references, is available in the "Descriptive Information" link. A graphical representation of the data is also provided, as shown below (STO-3G for Hydrogen):



Different basis sets can be downloaded and tried if those available with a particular program do not provide results of sufficient accuracy. Studies of the basis set dependence of a particular property of a given molecule can also be performed.

Computational Chemistry Comparison and Benchmark Database

As a part of the U.S. Department of Commerce, the National Institute of Standards and Technology (NIST, formerly the National Bureau of Standards) has a mission to promote innovation and competitiveness through advancement of measurement science, standards, and technology. As a part of the standards provision, the NIST Computational Chemistry Comparison and Benchmark Database (CCBDB), available at: http://srdata.nist.gov/cccbdb/, is a collection of experimental and calculated thermochemical properties for a selected set of molecules.

The two main goals of the CCCBDB are to provide benchmark data to: (1) Assist in the evaluation of computational methods, and (2) Allow comparison between thermochemical properties calculated using different computational methods. Thermochemical values include enthalpies of formation ($_{f}H$) and entropies (S) discussed in Chapter 15. Other values provided are those needed to compute thermochemical values: Geometries, vibrational frequencies, barriers to internal rotation, and electronic energy levels. A large number of other properties are also reported with both experimental and calculated values. A few of these properties include: Atomic charges, atomization enthalpies, bond lengths, dipole moments, geometries, HOMO-LUMO gaps, ionization energies, and zero point energies. Currently, the results of over 100,000 calculations are provided on molecules that contain 6 or fewer heavy (i.e. non-H) atoms and 20 or fewer total atoms. Additional results are added to the site as they become available.

Users are able to search by property, molecule, or by method of calculation and basis set [e.g. B3LYP/6-31G(d)]. An example molecular search window for experimental data is shown below. Note that a search for data on methane (CH_4) is about to be submitted.

II.A.1. (XV.A.)

Experimental Data

Please enter the chemical formula

CH4 Submit

Rules for chemical formula

- Enter a sequence of element symbols followed by numbers to specify the amounts of desired elements (e.g., C6H6).
- Elements may be in any order.
- If only one of a given atom is desired, you may omit the number after the element symbol.
- Parentheses may be used to group atoms.
- Multiple specifications for an atom will be added. This means that `CH3(CH2)4CH3` will be treated the same as `C6H14`.
- To specify one or more of a given atom, use a question mark (?) after the element symbol.
- To specify any number (including zero) of given element, use an asterisk (*) after the element symbol.
- A comma delimited list of several species may be entered.

Species in the CCCBDB

- No atoms with atomic number greater than 18 (Argon).
- Six or fewer heavy atoms and twenty or fewer total atoms. *Exception:* Release version 8 and higher have a few substituted benzenes with more than six heavy atoms.

The above search for methane returns a long, interactive table of values, a portion of which are shown below:

Listing of experimental data for CH_4 (Methane)

Other names

Biogas; Fire damp; Marsh gas; Methane; Methyl hydride; R 50; UN 1971; UN 1972;

Enthalpy of formation (Hfg), Entropy, Integrated heat capacity (0 K to 298.15 K) (HH), Heat Capacity (Cp)

Property	Value	Uncertainty	units	Reference	Comment
Hfg(298.15K) $\Delta_f H°$	-74.60	0.30	kJ mol^{-1}	Gurvich	
Hfg(0K) $\Delta_f H°$	-66.63	0.30	kJ mol^{-1}	Gurvich	
Entropy (298.15K) $S°$	186.37		J K^{-1} mol^{-1}	Gurvich	
Integrated Heat Capacity (0 to 298.15K) $H°–H°(T)$	10.02		kJ mol^{-1}	Gurvich	
Heat Capacity (298.15K) $C_p°$	35.69		J K^{-1} mol^{-1}	Gurvich	

Information can also be found for this species in the NIST Chemistry Webbook

Vibrational levels (cm^{-1})

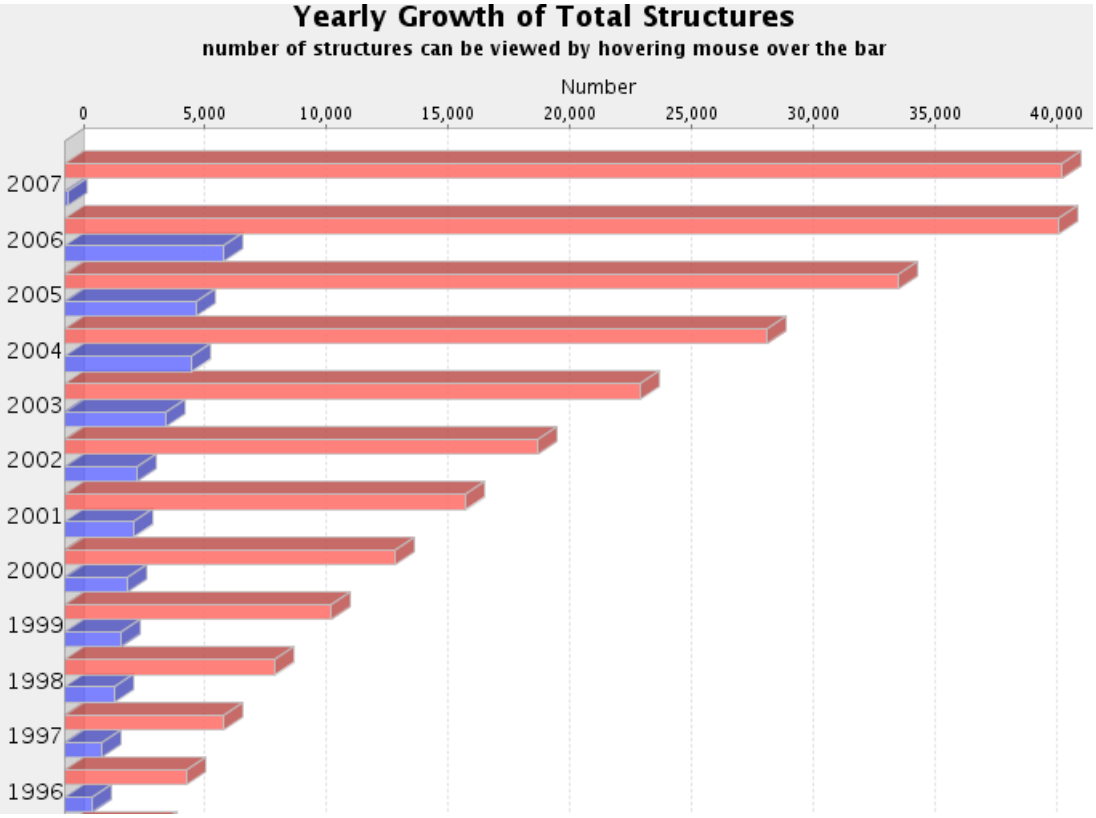
Mode Number	Symmetry	Frequency (cm^{-1})	Frequency Reference	Intensity (km mol^{-1})	Int. unc.	Intensity Reference	Comment
1	A_1	2917	Shim				
2	E	1534	Shim				
3	T_2	3019	1982vibint	69.7	2.1	1982vibint	
4	T_2	1306	1982vibint	33.4	1.0	1982vibint	

The site also contains a number of other useful features such as information on different energy units and their interconversion, an interactive feature that will provide isodesmic reaction suggestions (Chapter 15) for molecules

the user specifies, an interactive glossary of computational terms, and transition state enthalpies, geometries, and vibrational frequencies (Chapter 14).

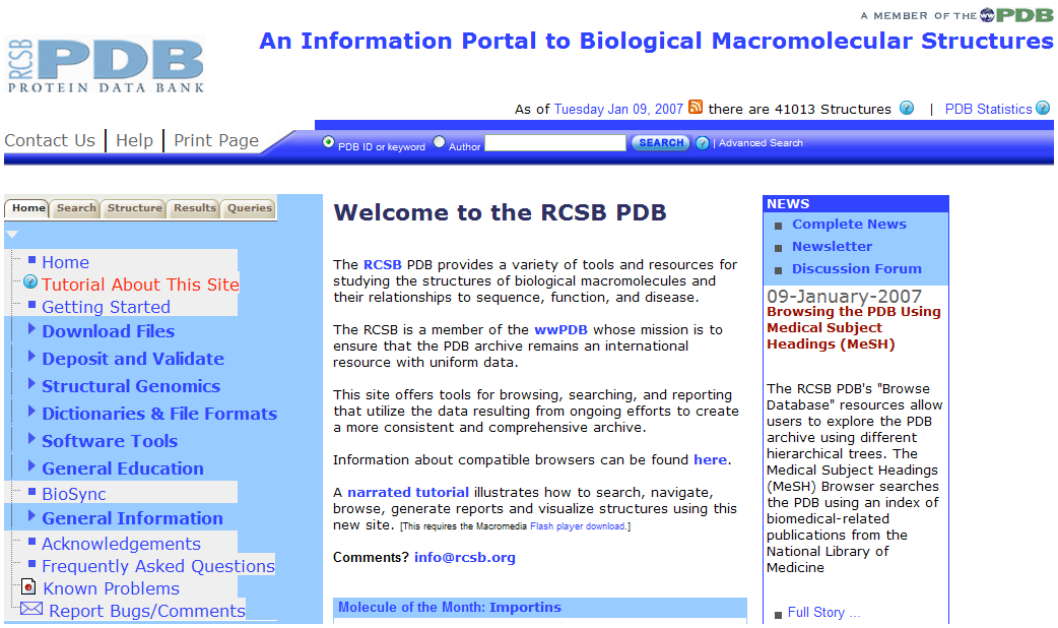
Protein Data Bank

The Protein Data Bank (PDB) serves as the worldwide repository for structural information on large biomolecules, mainly proteins and nucleic acids. The PDB was started in 1971 with seven structures. As improvements in computer power and data collection have advanced, the growth of the PDB has been rapid in recent years (see below – total structures in pink, yearly contributions in blue). Today there are over 41,000 structures in the PDB.

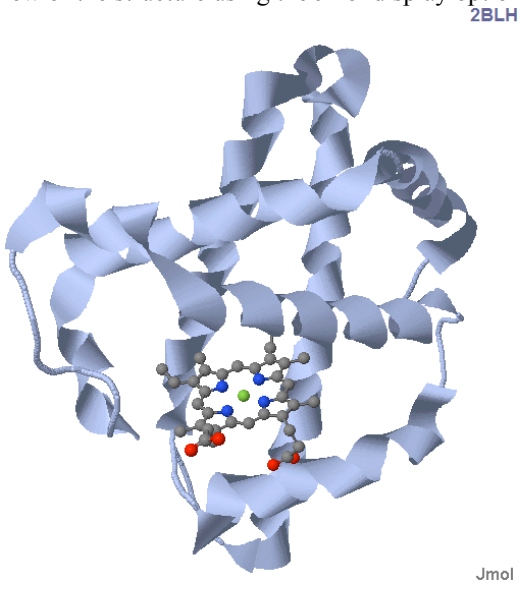


Researchers who determine these structures share their data to enable advances in biological and medicinal research. One of the important steps in understanding the biological function of a large molecule is to determine its 3D structure. Today, finding the amino acid sequence of a protein is now an automated process. Scientists are trying to create a reliable method of predicting the correct 3D structure of a protein from the amino aid sequence. With more structural data available, these scientists can further refine their models and will hopefully find a reliable method of protein structure prediction in the near future.

In addition to structural information, the PDB also provides amino acid sequence details, crystallization conditions, 3D images in various formats, and numerous links to other resources, including the original literature citation where the structure was reported. The site is available at: http://www.rcsb.org/pdb/home/home.do. The main page is shown below:



A keyword search for "myoglobin" (the mammalian O_2 storage protein) gives a list of 240 structures, the first of which happens to be from a sperm whale. All PDB structures have a four character alphanumeric identification code, and this one is "2LBH". A view of the structure using the Jmol display option is shown below:

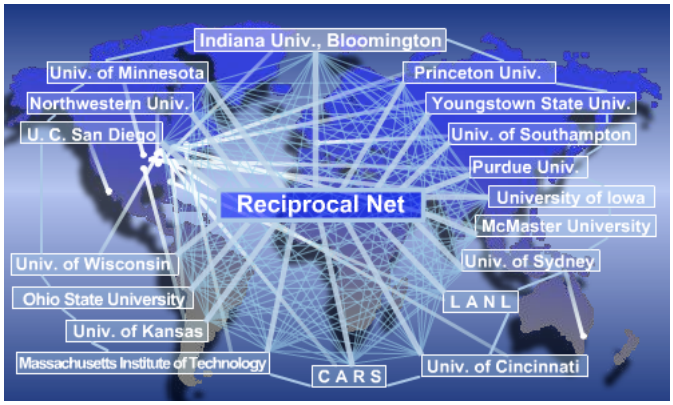


The "heme" area of the protein is shown in ball and stick format, while the protein backbone is indicated by the gray ribbon. The iron atom is green and the four heme nitrogen atoms are blue. Structure files can be downloaded in .pdb file format, imported into WebMO, and a calculation could be performed. Interactive links provided for "2LBH" give the amino acid sequence, further biological and structural details, relations to other types of proteins, etc.

Molecular Libraries

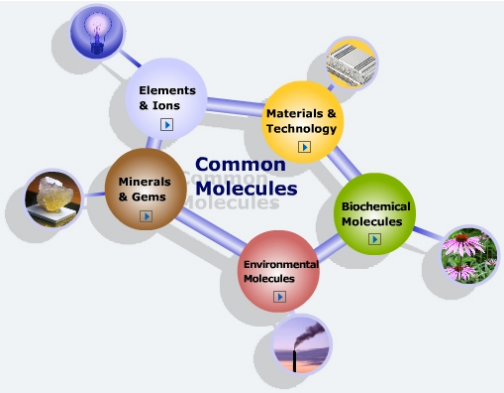
The Protein Data Bank provides structural details for large biomolecules. There are other repositories of structural information for smaller molecules, several of which are discussed here.

Reciprocal Net (http://www.reciprocalnet.org/) receives funding from the National Science Foundation (NSF) as part of the National Science Digital Library (NSDL) project. Reciprocal Net is led by the Indiana University Molecular Structure Center, with partner sites throughout the world, as shown below:



This site focuses on structures of more general interest and usefulness. The structural data, obtained by X-ray crystallographic analysis at the sites listed above, is converted into standard formats. A number of these formats allow interactive exploration of the structure online. Other formats allow production of high quality printable images. One goal of the site is to provide structures that can be used in constructing educational modules, thus bringing the general public into closer contact with current chemical research.

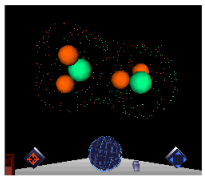
A useful portion of the site is known as the "Common Molecules Collection", shown below:



Clicking on one of the categories leads to an interactive list of compound classes, each of which contains an interactive, alphabetical list of compounds. Clicking on a compound name leads to a page with information concerning the molecule in layperson's terms, and a java applet that allows the molecule to be interactively viewed. Links to more sophisticated applets are present for higher quality images, if needed. Structural data can also be downloaded (.pdb file format), imported into WebMO, and calculations can then be performed.

The MathMol (Mathematics and Molecules) home page declares the site "is designed to serve as an introductory starting point for those interested in the field of molecular modeling". A link to the library of 3-D molecular structures (http://www.nyu.edu/pages/mathmol/library/) provides a gateway to five areas (see below): Water and Ice, Carbons, Hydrocarbons, Molecules of Life, and Drugs.

Library of 3-D Molecular Structures



If you are using CosmoPlayer click on the above image of a water dimer

[About the Database]

To enter the library, click on the appropiate buttons below.

Water and Ice

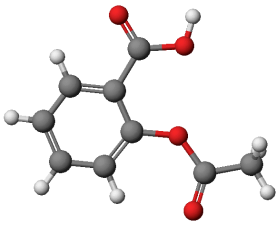
Carbons

Hydrocarbons

Molecules of Life

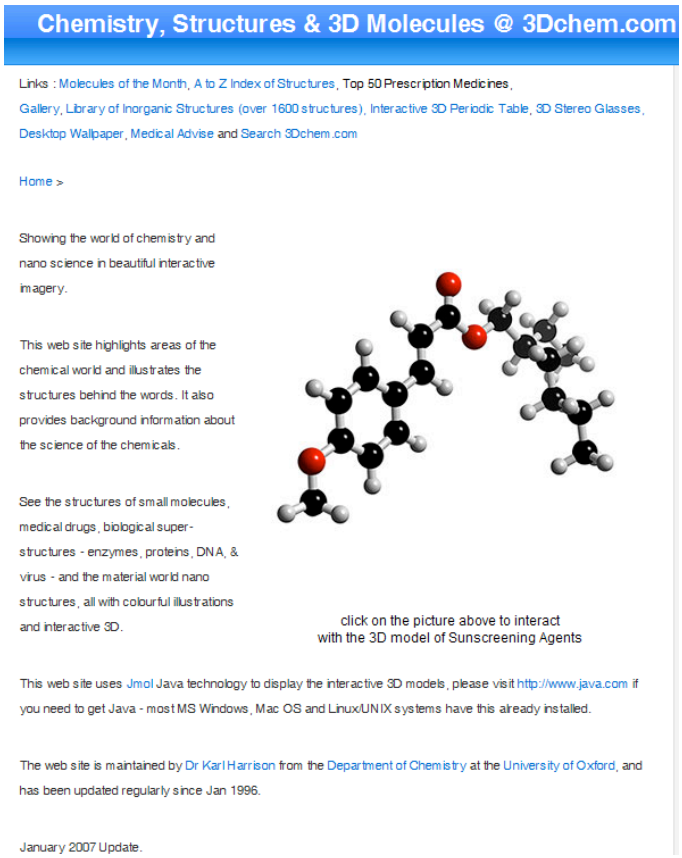
Drugs

Water and Ice has several structures ranging from a single water molecule to a water "box" containing ~34 water molecules. The Carbons section has the three elemental forms of carbon: diamond, graphite, and the fullerene C_{60}. The Hydrocarbons section contains methane through octane along with several other common molecules such as benzene, cyclohexane, etc. The Molecules of Life section includes amino acids, nucleotides, lipids, sugars, and molecules involved in photosynthesis. The Drugs section has structures of a number of common drugs, such as aspirin, shown here:



All structures can be downloaded in the .pdb file format and imported into computational chemistry programs, such as WebMO.

Thousands of structures are available at 3DChem.com, a commercial site. A large number of inorganic structures, the top 50 prescription medicines, and an alphabetical list of mainly organic structures are available (see: http://3dchem.com/).

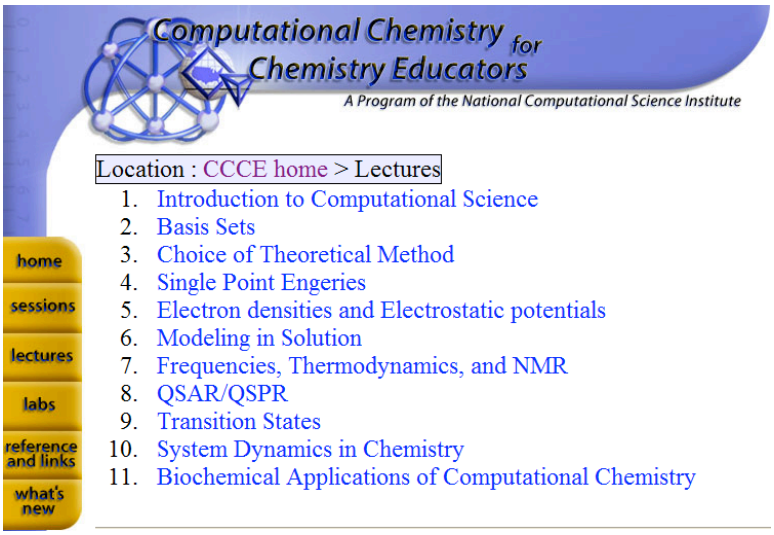


Clicking on a molecule opens a new window that contains information about the uses of the molecule along with a static image of the structure. A second click on the structure image opens an interactive view of the molecule using the Jmol viewer. Clicking again on the molecule displayed in the Jmol format with the right mouse button opens a menu that includes the .mol file format. Choosing this file format opens a new view of the molecule. Finally, right-clicking on this molecular view allows one to save the file in the standard MDL .Mol file format. Many computational chemistry software packages can open MDL .mol files, and these files can be imported into WebMO. If you don't mind a few advertisements and having to click four times to save a file, this is a useful site.

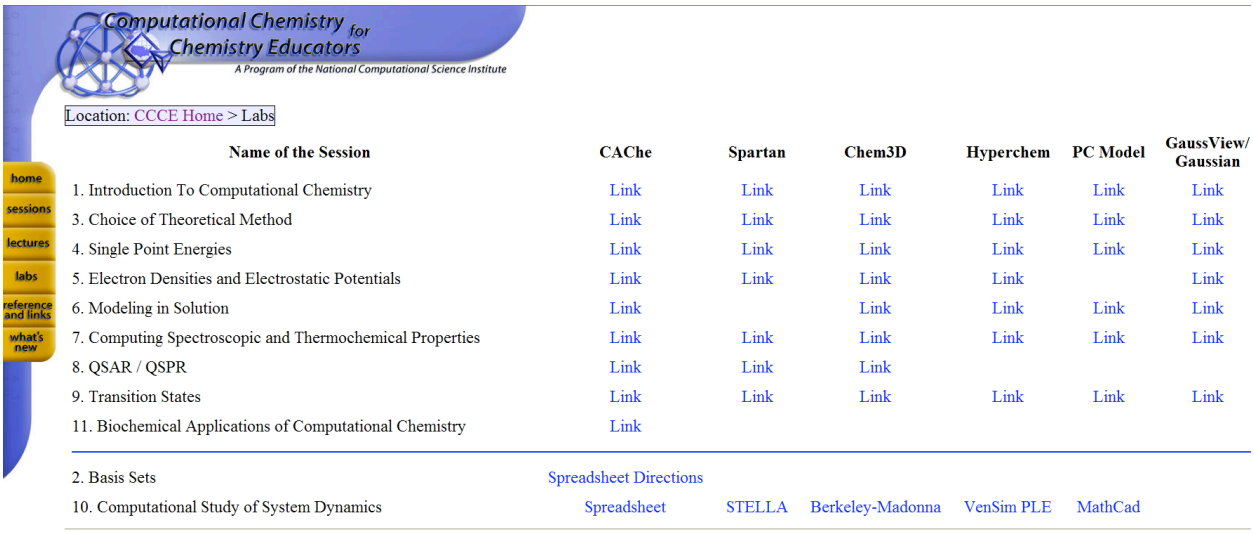
Computational Chemistry for Chemistry Educators

In 2001, the Shodor Education Foundation, with financial support provided in part by the Burroughs Wellcome Fund and the National Science Foundation began the National Computational Science Institute (NCSI). The first few years of NCSI focused primarily on interdisciplinary computational science workshops. Computational Chemistry for Chemistry Educators (CCCE) was the first discipline-specific set of workshops under the NCSI umbrella. The CCCE web site, which is still under development, was designed as the companion web-based resource for the in-person CCCE workshops. The purpose of CCCE is to provide educators with the competence and confidence to begin using computational chemistry as an educational tool with their students.

The site offers eleven sets of lecture notes on the topics listed below. Many of these topics are also covered in this manual.



Each of the links shown above opens a .pdf file which contains the lecture notes that were used in the summer workshops. All eleven lectures have associated laboratory exercises that serve to demonstrate the material covered in the lecture. These exercises are listed below:



Name of the Session	CAChe	Spartan	Chem3D	Hyperchem	PC Model	GaussView/ Gaussian
1. Introduction To Computational Chemistry	Link	Link	Link	Link	Link	Link
3. Choice of Theoretical Method	Link	Link	Link	Link	Link	Link
4. Single Point Energies	Link	Link	Link	Link	Link	Link
5. Electron Densities and Electrostatic Potentials	Link	Link	Link	Link		Link
6. Modeling in Solution	Link		Link	Link	Link	Link
7. Computing Spectroscopic and Thermochemical Properties	Link	Link	Link	Link	Link	Link
8. QSAR / QSPR	Link	Link	Link			
9. Transition States	Link	Link	Link	Link	Link	Link
11. Biochemical Applications of Computational Chemistry	Link					
2. Basis Sets	Spreadsheet Directions					
10. Computational Study of System Dynamics	Spreadsheet	STELLA	Berkeley-Madonna	VenSim PLE	MathCad	

Each column shown above contains links to instructions that are appropriate for the specified software package. Several of the laboratory exercises have been translated for use with WebMO and are included in this manual.

Chapter 22

Integrating Computational Chemistry into Existing Activities

22.1 Key Notes

22.1.1 Getting Started

Getting started with a new tool or technology is challenging regardless of what it may entail. In getting started with computational chemistry, educators are encouraged to start with small projects, labs, or activities, with the goal of having a successful and rewarding experience. Many resources are available in terms of labs and other classroom activities, including many found at the North Carolina High School Computational Chemistry server web page (http://chemistry.ncssm.edu). Educators are also encouraged to not think of computational chemistry as something to be done *in addition* to the things already expected in most chemistry classrooms. Rather, educators are encouraged to think about using computational cheistry *instead of* more traditional teaching strategies. Educators should also leverage the computer skills that many students have, allowing students to learn how to use the interface, and then teach other students and/or the teacher! While most educators will want to use computational chemistry to introduce and/or reinforce basic chemical concepts, some teachers may wish to help students learn the underlying concepts of how calculations are performed.

22.1.2 Planning and Preparation

Accounts on the North Carolina computational chemistry server and/or the national computational chemistry server are available free of charge to both students and teachers at the pre-college levels. Instructions and resources are available at the North Carolina site (http://chemistry.ncssm.edu) for teachers to request accounts. Student researchers are required to submit formal proposals for projects that require more computing resources than are typically allocated to students. There are also labs on the North Carolina site that teachers and students can download for a variety of purposes.

22.1.3 Managing the Lab

Managing a computational lab is to a large extent no different than managing a more "traditional" lab. The educator needs to understand the purpose of the lab, what pedagogical goals there might be, and what the students need to know beforehand. Like traditional labs, educators need to have a sense of timing. With computational labs, the calculations can be submitted to the server as a "pre-lab" activity or, if the calculations are small enough and/or the class size is small, can be submitted during the lab period. Unlike traditional labs, it might be the case that students are more familiar and independent with the "equipment" – in this case, a computer – than the educator, and students can often figure out how to do things that the teacher might not yet have learned. Depending on the purpose of the lab, students might spend more time analyzing results and data in a computational lab than they might in a more traditional lab, where a stronger emphasis is typically placed on getting equipment set up and working, preparing solutions and samples, and using lab equipment to collect relevant data. The approach to a computational lab can be a *black box* approach, where the students don't know (and don't care) how the calculations are being performed, or students can learn some of the foundational methods and mathematics underlying the calculations.

22.1.4 Activities for advanced chemistry students

More advanced students can and should be expected to do more open-ended and less well-structured computational labs. A good strategy, especially for advanced students who have mastered information that other students are still working on, is to provide the advanced student with a journal article, such as might be found in the *Journal of Chemical Education*. Advanced students can try to understand how the researcher applied computation to achieve his or her results, and perhaps attempt to replicate these efforts. Advanced students can also be asked to engage in independent research projects. These can be short- or long-duration projects. In these situations, the advanced student will typically have to develop an understanding of concepts such as a *model chemistry*, and be able to apply that to an interesting chemical research question. Educators should be familiar with what computational resources might be needed for students wishing to do independent research projects, and can get help and advice from the computational chemistry support team at NCSSM.

22.1.5 Activities for lower level and/or younger students

Younger students can develop a tremendous sense of the power of computation while performing fairly low level computational tasks such as building molecules, making bond length and angle measurements, and otherwise developing a sense of the structure of molecules. Calculations such as determination of vibrational frequencies are also very good activities for younger students to do. Not only can they see the molecules vibrate, but these vibrations can be related to experiences the students might already have had, such as seeing IR spectra on shows such as *CSI*. Younger students also enjoy building molecules as "art", and can be very creative in this regard.

22.1.6 Assessing Student Work

Assessing computational labs is not significantly different from that of more traditional labs. Computational labs *on average* require more analysis of data, but getting that data is also *on average* easier than that of wet chemistry labs. The use of grading rubrics is very helpful, both for the

student and the educator. The student has a clear idea of how s/he will be assessed, and the rubric provides consistency and simplification of the grading process for the educator. One of our favorite assessment strategies is to have students write journal-style abstracts for every lab, and short journal articles for several of the larger labs or independent projects. Students and teachers can also compare their computational results to experimental data and *benchmarked* computational data at the Computational Chemistry Comparison and Benchmark Database (http://cccbdb.nist.gov/).

22.1.7 Formal courses in computational chemistry

For schools with the flexibility to add courses, educators might consider offering one or more formal courses in computational chemistry, such as is the case at the North Carolina School of Science and Mathematics (NCSSM). At NCSSM, we offer a 30-hour elective course in computational chemistry, a 15-hour seminar series in medicinal chemistry, and a year-long sequence of courses entitled "Research in Computational Science", which includes a chemistry option. Educators are encouraged to contact chemistry faculty at NCSSM if interested in starting a course. Course materials developed at NCSSM are available to educators who wish to start one or more courses.

22.2 Getting Started

For most educators, it is probably a true statement to say that we *teach the way we were taught.* As such, most classroom educators (at least the ones over 30) did not use computational technologies, techniques, and tools as a part of their own educational program. The fact that you, the educator, are reading these words suggests that you *have* used computational tools or are interested in doing so! This section, and indeed this entire book, is written with this type of educator in mind.

There are several statements which we believe to be true, and are offered here for your consideration:

1. The use of computation is not something you do *in addition* to what you are already doing. We see computation as *another way* of teaching what you are already teaching. Many educators are overwhelmed with what they are already doing, and are hesitant to try something new.

2. A successful strategy, especially if you are hesitant about using computation, is to *start small.* The next section provides ideas and tips on planning and preparation. A successful first or second integration leads to larger and more ambitious activities and projects.

3. As we discussed in earlier chapters, computational methods are now a part of the *mainstream* of modern scientific research, and as such are just as important to future scientists as is "traditional" laboratory science. We believe that students must learn how to do basic science (including chemistry) using computation. We also believe that students must learn *about* computational chemistry, meaning understanding what it can and cannot do, how it is used in the research environment, and how it can be applied to the understanding of interesting chemical problems.

4. While many educators, especially older ones, may be uncomfortable with technology, students are not. If the educator is willing to give up some control to the students by asking them for help in managing the technology, the experience will be much more enjoyable for all. The role of the instructor in this scenario is to help the students understand the chemistry. Many

educators will not integrate technology until they can answer every single question about every single button or feature in the software. Students, on average, will figure out how to use the software much more quickly than will the educator! Willingness to leverage that reality works to the advantage of both teacher and student.

There are two ways to think about the integration of computational chemistry into your classroom. With a tongue-in-cheek nod to our mathematical colleagues, this might be called the "Associative Law of Computational Chemistry Education":

1. **Computational (chemistry education)**: this refers to the use of computing as a way to teach the concepts of chemistry found in traditional chemistry courses. Computation becomes the teaching environment where students can learn traditional concepts such as acid/base chemistry, reaction kinetics, thermodynamics and thermochemistry, and other topics found in most standard courses of study. With this method, the computer is often treated as a black box: the students will likely not understand the inner workings of the comptuational method, underlying mathematics, or processes involved. They are interested in what answer is provided by the computer and how that answer gives them insight into the chemical system of interest.

2. **(Computational chemistry) education**: this refers to the learning the student does *about* computational chemistry. Students are educated *about* computational chemistry. In this approach, the focus is more on the underlying mathematics, computational methods, run times, and other aspects of getting the calculation to work. The final answer may or may not have any consequence at all. Students who are learning about computational chemistry are more concerned with trying to get a particular calculation on a chemical system to run efficiently.

As with all scientific tools that a teacher might use for student learning, the teacher needs to make the pedagogical decision as to what s/he is trying to accomplish. In a later section of this chapter, we will present background information on our formal courses "Introduction to Computational Chemistry" and "Research in Computational Chemistry". In this first course, the emphasis is more on how to do computational chemistry, with lesser emphasis on solving interesting problems. In the second course, the emphasis switches to applying what was learned in the first course to challenging and interesting chemical systems.

22.3 Planning and Preparation

As in most things, success breeds success. Given that, educators are encouraged to start small by doing small projects, either as demonstrations or in the computer lab. The instructions listed below assume that you will be using one of the two Web-based computational chemistry servers, either the North Carolina High School Computational Chemistry server (for North Carolina teachers and students, located at *http://chemistry.ncssm.edu*), or the national server (for non-North Carolina teachers and students, located at
http://cli.globalgridexchange.com/).

1. **Request a teacher account:** educators can receive a free account (all accounts are free) by sending email to Bob Gotwals at gotwals@ncssm.edu. Educators are provided with unlimited

accounts, meaning that there is no limit on the size of the job that can be run and no limit on the total amount of CPU time. Requests for accounts are typically fulfilled in 24 hours or less.

2. **Download and explore one of the online labs:** a number of online lab activities are available on the main website (*http://chemistry.ncssm.edu)*. These labs have all been tested with a variety of students, and most address concepts found in all chemistry courses. The labs cover a variety of topics. Some of those topics are ones found in most if not all honors/AP level classes, including general structure of molecules, Lewis dot structure configurations, and the like. Others are designed to teach the ins and outs of computational chemistry, focusing on helping the students understand the different *model chemistries*, basis sets, advantages and disadvantages of various software packages, and other issues related to the hows and whys of computational chemistry.

3. **Request student accounts:** student accounts are available free of charge. To request student accounts, download the file *classroom.xls*, an Excel spreadsheet, from the Teacher section of the main website *http://chemistry.ncssm.edu*. On this spreadsheet, provide a list of usernames, all in lowercase. You can, for example, list students by last name-first initial, such as millerb for Brad Miller. Alternatively, you can request usernames that have the initials of the school, followed by a number, such as jshs1, jshs2, etc. for Jordan Senior High School. You can provide a generic password for these accounts. Students can change their passwords under **Utilities/Edit Profile** in WebMO. All student accounts receive time limits of 4 minutes of per job time (meaning any job that takes longer than 4 minutes will fail), and a total CPU time limit of 20 minutes (meaning once 20 minutes of total CPU time has been used by the students, all subsequent jobs will fail). Providing email accounts is optional and generally not worth the time it takes to type them into the spreadsheet. Once you have completed the spreadsheet, send it to Bob Gotwals at gotwals@ncssm.edu. Accounts are generally activated within 24 hours, and the requestor is notified of activation.

4. **Submit a request for a specific lab activity:** if you are unable to find a lab that specifically addresses a concept you are interested in teaching, you can submit a request for a lab to Bob Gotwals at gotwals@ncssm.edu.

22.4 Managing the Lab

Managing a computational lab is not significantly different from managing a more traditional "wet" chemistry lab. The computational lab has the advantage of not requiring the preparation of solutions or the gathering and setting up of equipment, but it also requires an understanding of timing and resource management. Like the traditional lab, there is a learning curve for both students and teacher. Some considerations for managing the lab are as follows:

1. **What are your goals for the lab?** Your goal for the lab may be several-fold, but you would no more do a computational lab without having some guiding pedagogy than you would a traditional lab. Example goals might include:

   (a) introduce students to the use of computing in chemistry

   (b) introduce some basic chemistry concept, such as bonding or molecular structure

(c) reinforce a concept presented in lecture

2. **Is your lab qualitative or quantitative?** By definition, all computational labs are quantitative, in that the server generates a significant amount of numerical data that is then presented through the WebMO interface to the student. However, if your goal is for students to get a "feel" for the computation, and accurate data is not necessary, you can run the calculations with less accurate *model chemistries*. Student results are then more qualitatively useful, and less accurate quantitatively. This helps significantly in terms of resource management, especially if you have a large group of students. Students doing research, however, might need to use more sophisticated (and resource-consuming) calculations, resulting in more accurate quantitative results.

3. **How much time can you devote to the lab?** One of the frequent comments from teachers new to computing is the problem of trying to "cover" all of the material in a typical standard course of study. Adding a new activity is oftentimes a challenge. Many educators, especially those new to the classroom, assume that some topic needs to be deleted in orer to add a computational activity. Experience has strongly suggested, however, that the integration of a computational lab significantly augments and solidifies basic concepts. It is also often the case that a computational lab requires less time than a typical lab activity, and thus can have the benefit of opening up time for other activities. With a computational lab, students can submit a job at the beginning or end of the class, and then do the analysis the next day, during the next lab block, and/or as a "take home" lab activity. Once the computation is complete, it stays in the student's folder until such time as s/he is ready to do the analysis work. With large classes, this approach is especially relevant. Depending on the size of the molecule, the type of calculation required, and the size of the model chemistry, run times can become quite long. Best practices suggest a submit and move on approach – students submit their jobs, and then return to some other classroom and/or lab activity. Most jobs should be completed by the next class period. As a reminder, students do not need to be logged in for jobs to run, and most servers (including the North Carolina and national servers referenced in this *Guide* run 24/7.

4. **Do your students work in teams?** Most chemistry educators have their students work in pairs in traditional labs, and the same approach should be taken with computational labs. In a team of two or three students, *one* student should submit the job to the server for calculation. Other students in the group can build the molecule, and set the calculation up, but not submit it for computation. This reduces the number of jobs to be run, thus preserving resources for other students and teachers.

One key to success for educators new to computational chemistry is to become partners with the students! Students who have grown up with computers and video games have a natural intution about how to make computers do what they want. Educators who insist on knowing every single option that is available will likely be frustrated with computational chemistry, and it will be an unpleasant experience for both teacher and students. Make it clear that you too are learning, and let your students show you things that they have figured out how to do! Not only is this empowering for the students, but your skills will improve rapidly in this manner!

22.5 Activities for advanced chemistry students

Advanced chemistry students, such as those found taking AP or other advanced/honors chemistry courses, benefit significantly from a computational approach to chemistry. With these students, more focus on the underlying mathematics can be undertaken, and some of the more challenging concepts, such as transition states, thermochemistry, and reaction kinetics can be studied computationally. With lower level students, the computer is more of a black box; with more advanced students, one can be more explicit and rigorous about how the calculations are completed.

With advanced students, it is also possible to have more open-ended computational activities, ones that require more initiative, imagination, and/or teamwork than some of the simpler "cookbook" computational labs. For example, advanced students can be given a computationally-based article from journals such as the *Journal of Chemical Education*, and be asked to replicate the results in this article. For example, Figure 1 shows an example article from Professor Bumpus and colleagues at the University of Northern Iowa, one that is quite popular with most high school students! This requires students to do a careful analysis of the article – figuring out what the researcher was trying to accomplish, determining the computational approach used, and then trying to replicate that approach. It is often the case that journal articles don't report failures and other challenges, and the students will need to figure out how the journal authors were able to get their calculations to work! Even if unsuccessful, students develop a tremendous understanding and appreciation of the *process* of doing research computationally.

Advanced chemistry students also, more typically than lower level students, engage in some type of independent research work. Computational research projects are becoming more commonplace in the better known scientific competitions, such as Siemens, Intel, and the Junior Science and Humanities Symposium. Judges in past years have not been open to computationally-based student projects, but that is changing rapidly. With the North Carolina and national servers, student researchers are expected to submit a research proposal to the administrators of the two servers, describing their project – goals, research question, thoughts on their computational approach, etc. This requirement mimics that of professional researchers, many of whom will submit proposals to one or more supercomputing centers in order to gain access to high performance computing resources. It is sometimes the case that a student project will exceed the resources available on the North Carolina and national servers, and students will need to apply for compute resources at a national or state supercomputing center. Educators at the North Carolina School of Science and Mathematics (NCSSM) can provide assistance and advice to students needing more advanced computing resources.

22.6 Activities for lower level and/or younger students

Computational chemistry is not just for high school students! The authors have had excellent experiences with younger students, down to about fifth grade. Students are very excited to be able to build molecules, rotate them, and otherwise interact with the molecular editors. We have also had a number of students perform vibrational frequency calculations. Students at the lower levels very much enjoy seeing how molecules vibrate in the infrared. With the popularity of forensics shows such as *CSI*, we have also had middle school students learn about IR spectra, and do some very rudimentary analysis of IR spectra. This requires a basic (but brief) introduction to organic functional groups. We provide students with a chart showing functional groups and some of the more basic stretches.

Characterization of High Explosives and Other Energetic Compounds by Computational Chemistry and Molecular Modeling

Experiments for the Undergraduate Curriculum

John A. Bumpus,* Anne Lewis, and Corey Stotts
Department of Chemistry, University of Northern Iowa, Cedar Falls, IA 50614; *john.bumpus@uni.edu

Christopher J. Cramer
Department of Chemistry and Supercomputing Institute, University of Minnesota, Minneapolis, MN 55455-0431

Heat of formation is one of a variety of properties used to assess and compare the energy content of high explosives[1] *(1)*. We present a series of introductory and advanced experiments in which heats of formation are predicted from calculation. These experiments can be used to introduce students to commercially available computational chemistry and molecular modeling programs. They also help reinforce several concepts in thermodynamics and computational chemistry. In the initial experiment several compounds are arranged such that students are provided structures with increasing levels of complexity so that they may master the drawing programs in a stepwise manner. Ab initio quantum mechanical (QM) calculations are used to model simple compounds. However, as structural complexity increases, students quickly learn time and expense are considerations in the use of such calculations. In the second experiment students are shown how to set up and use semiempirical calculations, the Austin Model 1 (AM1), the Modified Neglect of Differential Overlap (MNDO) Model, and the Parametric Model 3 (PM3), to estimate the heats of formation for a series of energetic compounds, including some that are used as high explosives. In the third experiment, selected results from computational studies are compared with those predicted using an older, but still useful, group additivity method *(2, 3)*. Both approaches are in these experiments. The remaining three experiments introduce students to new and old computational methods used to calculate heats of formation.

Experiment 1: Molecular Model Construction

Students are provided the structures of a number of high explosives and other selected energetic compounds and are asked first to draw them in any one of several molecular modeling programs that are commercially available. For example, we have available PCSpartan (Wavefunction, Inc.), HyperChem 7.0 (Hypercube, Inc.), and ChemDraw Standard 7.0, which is used in conjunction with Chem 3D Ultra 7.0 (CambridgeSoft Corp). Manufacturers of these software programs provide relatively straightforward instructions for



nitroglycerin tetranitromethane

Figure 22.1: A computational article from *Journal of Chemical Education*

Students also enjoy performing molecular orbital calculations and then displaying graphics, such as electrostatic potentials. Even with middle school students, one can discuss concepts such as acidity, and then use electrostatic potential maps to help them "see" the acidic hydrogen.

Students can also do computational chemistry "art", such as the graphic of a "man" shown in Figure 2. Recognizing that this requires either a significant understanding of molecular orbitals and/or a significant amount of luck, students do enjoy trying to design interesting looking works of art!

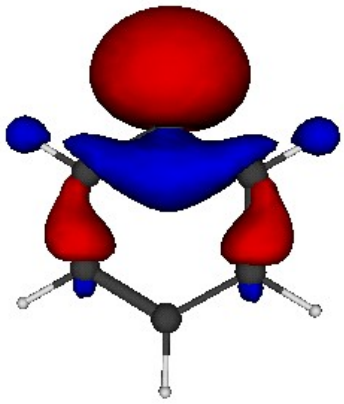


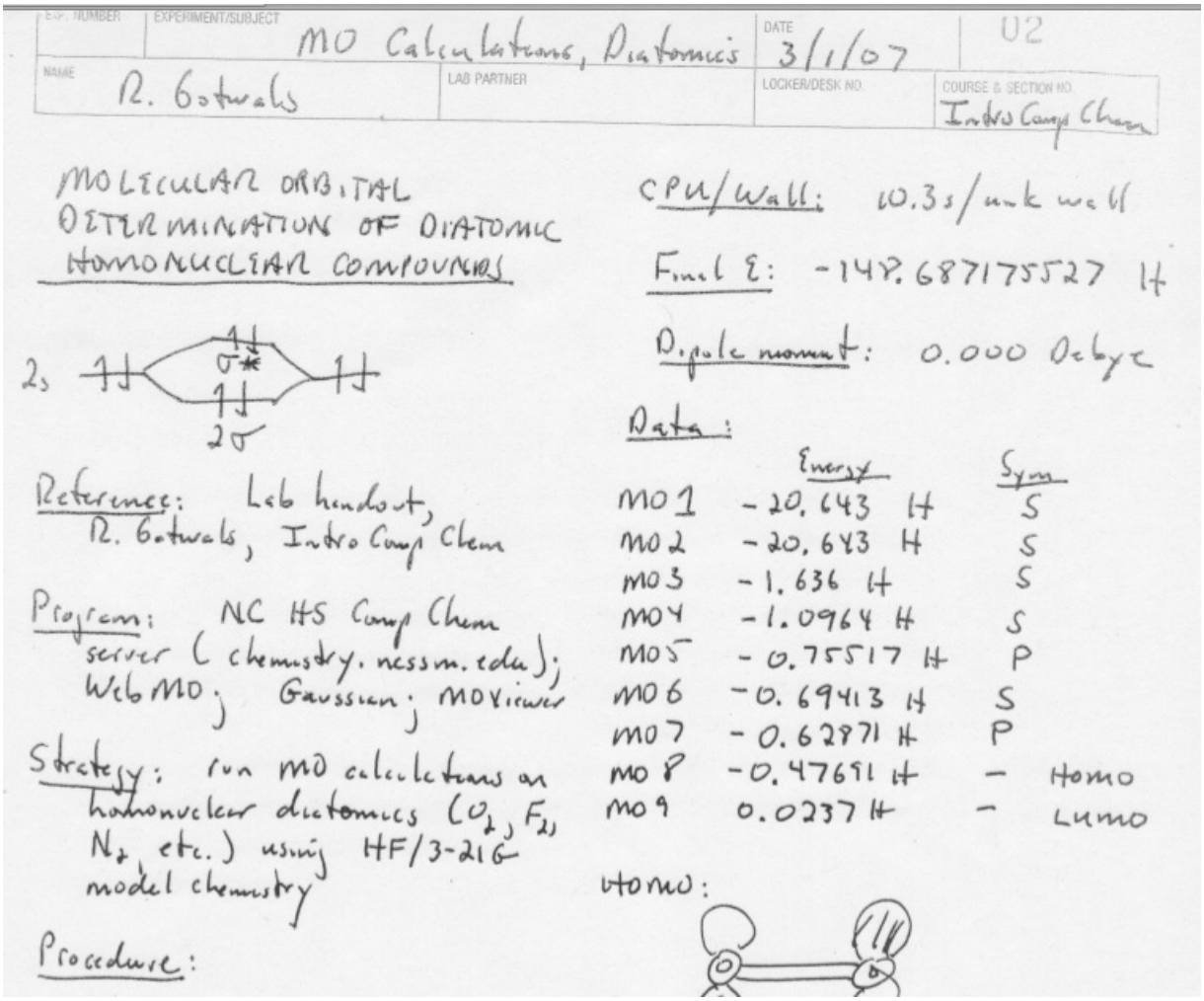
Figure 22.2: A computational chemistry "man"

22.7 Assessing Student Work

Computational labs can be assessed in a manner very similar to those of traditional "wet" chemistry labs. Students can complete a prelab activity, perform a computation, compare their results to literature or experimental values, determine a percent error, and otherwise assess the value of their work. We require students to keep a lab notebook, following a specific format. A portion of a typical computational lab page is shown in Figure 3.

For students wishing to compare their computational results (with other computations and with experimental data), an extensive database – the Computational Chemistry Comparison and Benchmark Database (http://cccbdb.nist.gov/) – is a great resource. This resource provides students with *benchmarks* by which to self-assess their computational work. A screenshot of the menu bar is shown in Figure 4. Depending on the level of student and the time available for the lab, students can try to determine the best method for approximating a literature value. Students can also prepare lab reports following standard and/or preferred formats.

We also require students to write a lab abstract for every computational lab, describing the goal of the lab, the computational approach, sample results from the calculations, and a short conclusion statement. Abstracts are typically 250-300 words, written in passive voice. A significant part of the students' grade comes from a well-crafted lab abstract. We require every student to write his or her own abstract, although students typically do the lab in teams of two students and are encouraged to discuss their results.


EXP. NUMBER | EXPERIMENT/SUBJECT: MO Calculations, Diatomics | DATE: 3/1/07 | 02

NAME: R. Gotwals | LAB PARTNER | LOCKER/DESK NO. | COURSE & SECTION NO.: Intro Comp Chem

MOLECULAR ORBITAL DETERMINATION OF DIATOMIC HOMONUCLEAR COMPOUNDS

Reference: Lab handout, R. Gotwals, Intro Comp Chem

Program: NC HS Comp Chem server (chemistry.ncssm.edu); WebMO; Gaussian; MOViewer

Strategy: run MO calculations on homonuclear diatomics (O_2, F_2, N_2, etc.) using HF/3-21G model chemistry

Procedure:

CPU/Wall: 10.3s/unk wall

Final E: -148.687175527 H

Dipole moment: 0.000 Debye

Data:

	Energy	Sym	
MO 1	-20.643 H	S	
MO 2	-20.643 H	S	
MO 3	-1.636 H	S	
MO 4	-1.0964 H	S	
MO 5	-0.75517 H	P	
MO 6	-0.69413 H	S	
MO 7	-0.62871 H	P	
MO 8	-0.47691 H	-	Homo
MO 9	0.0237 H	-	Lumo

Homo:

Figure 22.3: A computational chemistry lab page

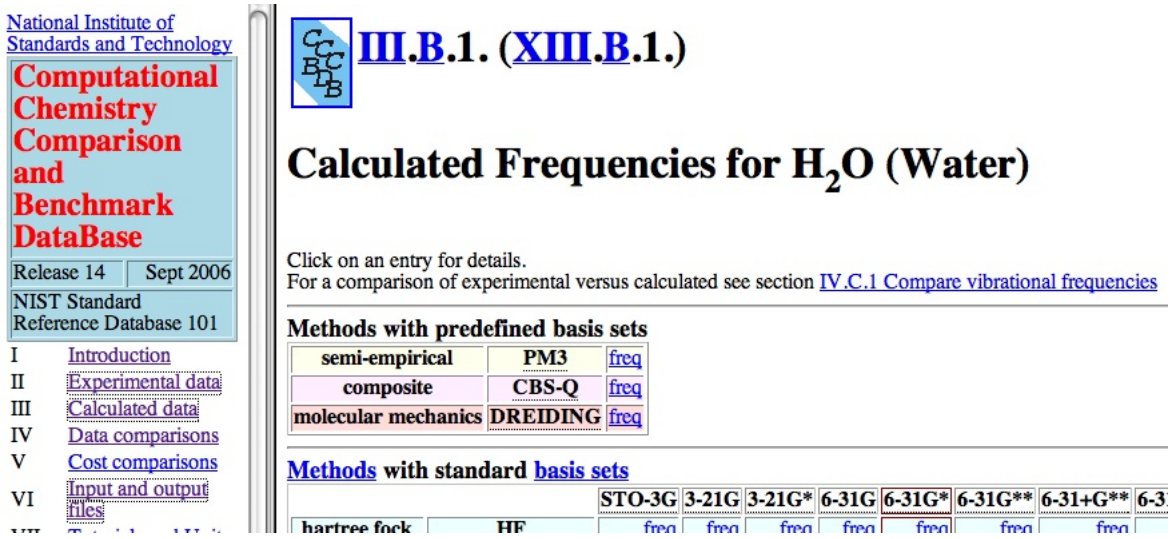


Figure 22.4: Menu bar and sample data comparison page from NIST Computational Chemistry Comparison and Benchmark Database

A sample lab abstract is shown below:

Abstract: The choice of a theoretical method in computational chemistry is a critical consideration for the computational chemistry practitioner. The choice of a theoretical method is evaluated as applied to the protonation of the pyridine molecule, a benzene-like cyclic organic compound containing a single nitrogen atom in the ring. Geometry optimizations are performed on both a neutral pyridine molecule and a positively charged pyridinium cation, where a proton is attached to the nitrogen atom. The optimizations are performed using two semi-empirical methods, AM1 and PM3, using the MOPAC software. Geometry optimizations are also performed using a density functional theory (DFT) hybrid functional (B3LYP/6-31G(p,d) on both of the organics. Single point energy calculations are performed on the proton using AM1 and PM3. A comparison of the heats of formation for the reaction, calculated using Hess Law, shows that the DFT theoretical method is significantly better (1.6% error as compared to the experimental value) than that of the PM3 method (5% error) or the AM1 method (10% error).

Students also very much appreciate the use of a grading rubric, which they receive prior to the activity. Figure 5 shows a portion of a sample grading rubric for lab abstracts (in this case, a rubric for an experimental laboratory project). Students who are poor writers will improve fairly rapidly from the use of a grading rubric, and, even though the choice of a "5" or "4" is very subjective on the part of the grader, students do not complain about grading "fairness" or other real or perceived grading slights!

Lab Abstract Grading Rubric

	Purpose	Description	Analysis	Results/ Conclusions	Grammar/ Mechanical Clarity	Understanding
	The purpose of the lab…	*The experimental description …*	*The analysis section…*	*The numerical results…*	*The abstract contains…*	*The writer demonstrates…*
5	is elegantly stated.	adequately describes how the measurements were obtained without extraneous detail.	adequately describes how the reported results were obtained without extraneous detail.	are correctly reported; all appropriate quantitative comparisons are made among the results and with any expected values.	no errors in sentence structure, spelling, word choice, or punctuation. The writing is elegant and clear.	a superior understanding of the experiment and offers original insights .
4	is adequately stated.	describes how the measurements were obtained, but some details are missing.	describes how the reported results were obtained, but some details are missing	are reported; some quantitative comparisons are made among the results and with any expected values.	no more than 1 error in sentence structure, spelling, word choice, or punctuation. The writing is clear.	a good understanding of the experiment.
3	is poorly stated.	is incomplete - key information is missing.	is incomplete - key information is missing.	are adequately reported but comparisons are not reported.	no more than 2 errors in sentence structure, spelling, word choice, or punctuation.	an adequate understanding of the experiment.

Figure 22.5: Abstract grading rubric

22.8 Formal courses in computational chemistry

At the North Carolina School of Science and Mathematics (NCSSM), we offer a 10-week, 30-hour course in computational chemistry entitled "Introduction to Computational Chemistry". In this course, students learn the basics of computational chemistry, with an emphasis on (computational chemistry) education, and less emphasis on learning basic or even advanced chemistry concepts. A copy of the syllabus is available under "Teacher" resources on the main web page (http://chemistry.ncssm.edu). Students spend a significant amount of time learning about the foundational mathematics, the advantages and disadvantages of different model chemistries, the advantages and disadvantages of different software packages, and how computational chemistry is applied to a variety of chemical problems. We use a framework of *six guiding questions* to help the students to organize their learning:

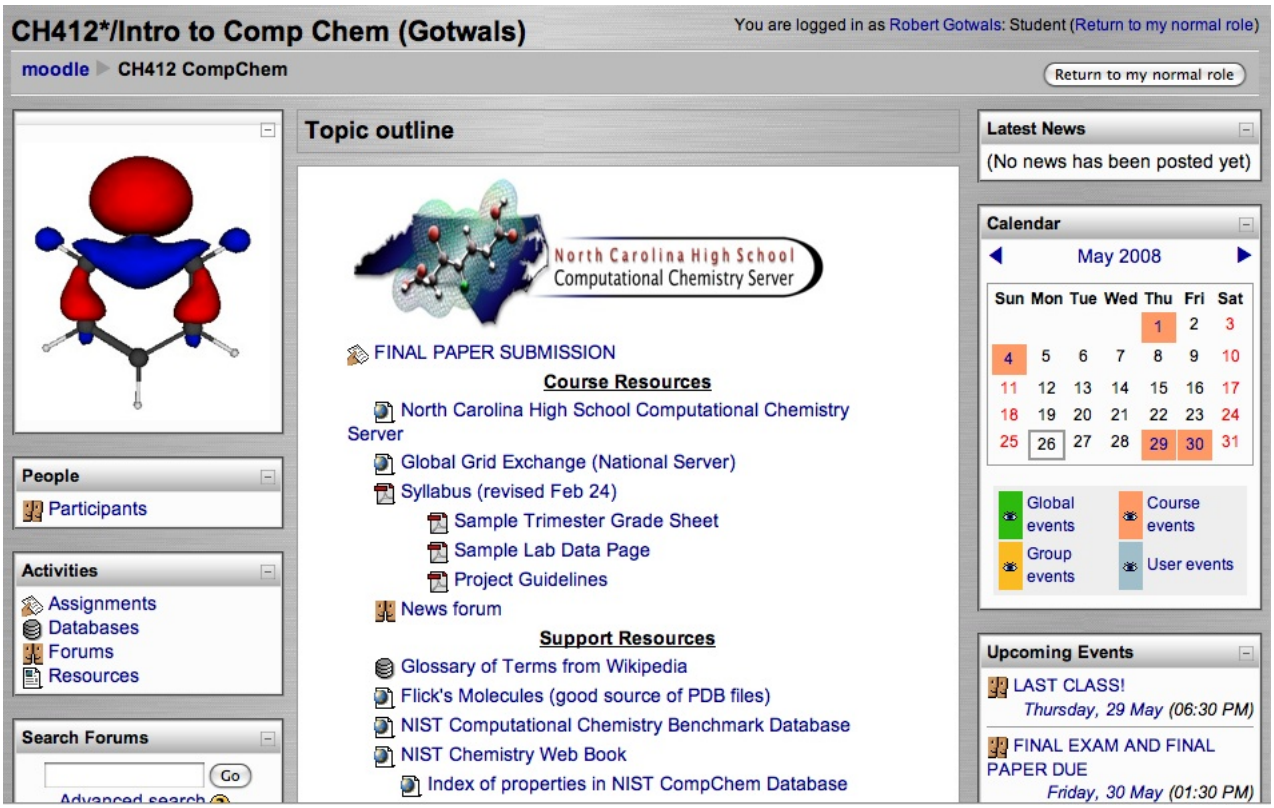


Figure 22.6: Top of Comp Chem "Moodle" page

1. What is the role and purpose of computational chemistry? What does computational chemistry allow us to do that cannot be done using "traditional" (i.e. wet) chemistry?
2. What is the fundamental mathematical expression that needs to be solved in doing computational chemistry? What are the terms in this equation, what is their significance, what variations can be used?

3. What are the approximations that can be used in doing computational chemistry? What are the pros and cons of the various approximations? How does choice of approximation affect the results, the computing time, etc.

4. There are roughly four different "flavors" to computational chemistry: ab initio methods, semi-empirical methods, density functional theory (DFT) and molecular mechanics/molecular dynamics. What are these methods? How do they differ?

5. What are the fundamental units of measure used by computational chemists? What are some different ways that these fundamental units might be expressed?

6. What are some of the computer codes that one might use to do computational chemistry? What platforms are needed for these codes, what are the strengths and limitations of these codes?

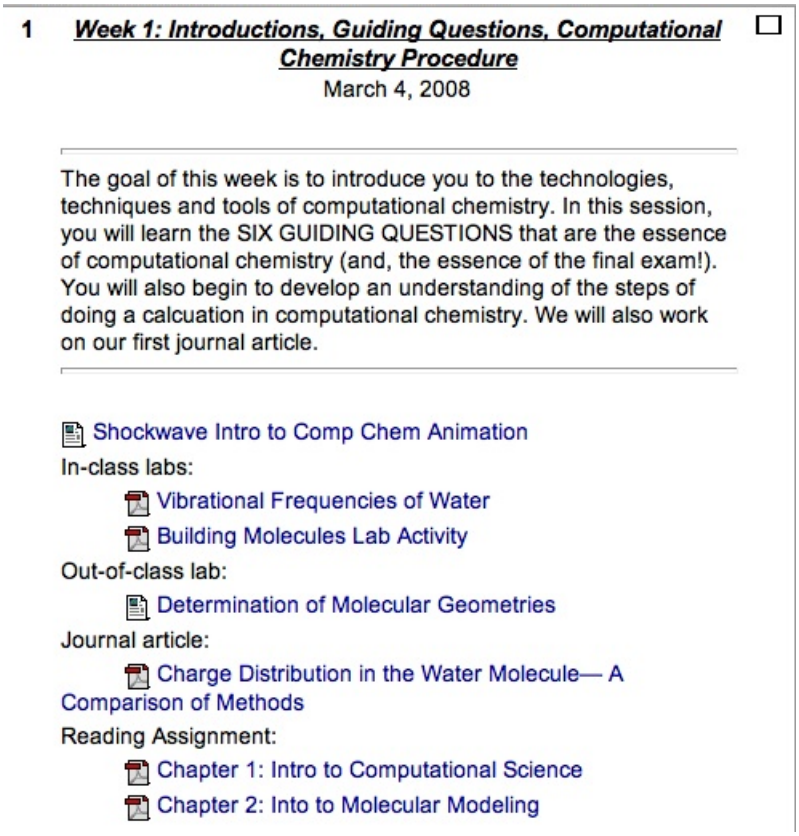


Figure 22.7: A typical week's activities

Figure 6 shows a screenshot of the "Moodle" page of NCSSM's Introduction to Computational Chemistry course. Figure 7 shows a screenshot from the Moodle page, showing a typical week's activities. During a given week, students attend lectures on the topic for the week, perform at least

one in-class lab, read and review a journal article, and work collaboratively in teams of two or three students on an out-of-class lab. They also have a weekly reading assignment, and take one 15-20 minute quiz each week on the work completed in the previous week.

NCSSM also offers two online versions of the introductory course. The first is offered to mixed groups of teachers and students, and is conducted weekly for 90 minutes over full-motion, two-way videoconferencing. The second version is a more "traditional" online course (http://online.ncssm.edu/compchem.htm), where students interact with the instructor and each other through a variety of online media, including email, a course management web tool, chat rooms, and personal videoconferencing. At the time this is being written (Summer 2008), this course is only available to select students from North Carolina. Figure 8 shows the NCSSM Online Web page with a list of course offerings. Note that this program also offers a course in "Medicinal Chemistry". Labs in this course are done completely with computational methods; the emphasis here, however, is less on how calculations are performed and more on the significance of the calculation results to issues in drug design and pharmacology.

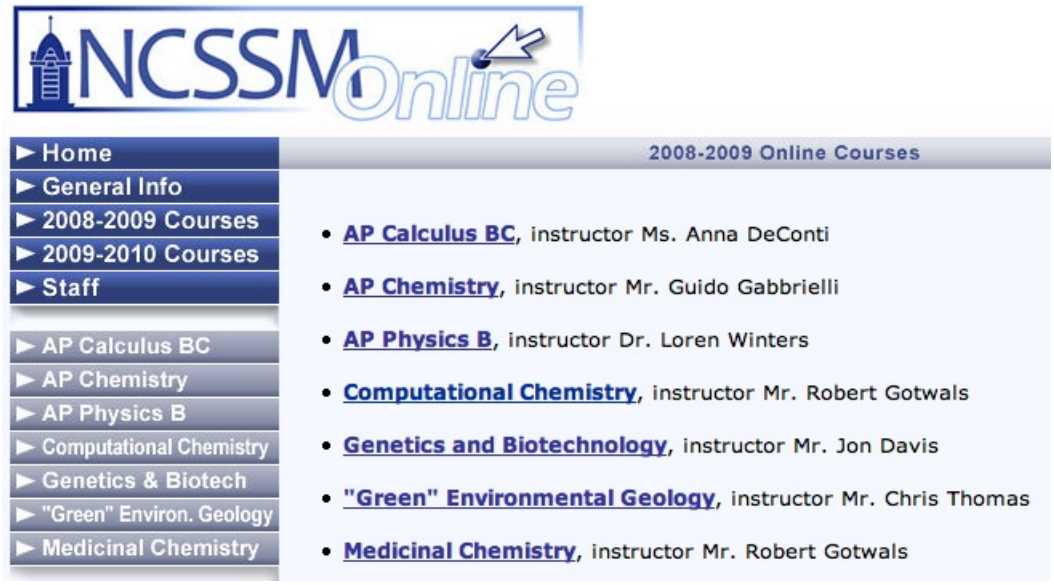


Figure 22.8: Main page from NCSSM Online program

NCSSM also offers a series of courses entitled "Research in Computational Science", a part of the "R"-series of research courses (Research in Chemistry, Research in Physics, and Research in Biology). These programs are multi-trimester, and provide students with a chance to engage in long-term research, often with scientific mentors at local universities and research labs. A description of the computational science courses is available under "Teacher" resources on the main web page (http://chemistry.ncssm.edu).

Chapter 23

Computational Chemistry Research

23.1 Key Notes

23.1.1 Choosing a Research Problem:

Students wishing to engage in computational chemistry research should have reasonable practical experience in building molecules, submitting calculations (jobs), and analyzing results. The student researcher should also have a good idea of the types of problems that can, and that cannot, be solved using available computational resources. While there is no magic number, most students wishing to do an independent project in computational chemistry will have done several dozen or more smaller projects, often under the direction of the classroom teacher.

In choosing a research problem, there are four guiding questions that the student can use to help narrow down the choices:

1. **What area of chemistry are you most interested in?** Answers to this question include organic chemistry, inorganic chemistry, medicinal chemistry, and environmental chemistry, among others.

2. **What resources do you have at your disposal?** Answers to this question include the amount of computing time, access to mentors, etc.

3. **How much time to you have?** Specifically, this means what is the duration of the research? Is this a one-week project, or something being done over the course of a semester or summer research program? Guidelines for how many jobs might be run given the length of the project in days is presented in the reading below.

4. **Is there a particular category of computations that is of most interest?**

Beginning student researchers might be intrigued by determination of transition state geometries, or calculation of various spectra for a molecule or series of molecules.

23.1.2 Applying for a Research Account:

Computational chemistry researchers in the scientific community often need to request computing time on large supercomputers, located at a variety of supercomputing centers. To do so, they submit a research proposal, describing the research to be conducted, the expected results, the amount of computing time need (in terms of CPU minutes or hours), and other issues related to the research. Likewise, student researchers wishing to use the computational chemistry resources described in this document must submit an online research proposal. The proposal is evaluated by the computational chemistry server support staff, and computing time is appropriately awarded. The proposal includes components that include a working time, an abstract describing the work, software requirements, per job time limit, a total CPU time limit, and the name(s) of a teacher or mentor who is supporting or sponsoring the student research.

23.1.3 Choosing a Model Chemistry:

One of the most critical components of student research is the choice of a computational approach for the research problem. In other words, a description of the calculations to be performed, and, for quantum methods, a description of the *model chemistry* that might be used to do the calculations. The critical guideline is that the student researcher should **choose the simplest model chemistry that produces the data needed to answer the research question.** For the beginning researcher, this is a challenging issue. The goal of any good computational chemistry research is to get the best data possible with the least amount of computing time. This is an important consideration in professional computational chemistry research, and important in student-based computational chemistry research.

23.1.4 The Computational Chemistry Notebook:

Good researchers keep good lab notebooks, and that requirement does not change for computationally-based research. The basics of keeping a computational chemistry notebook are similar to that of a notebook kept in a traditional wet-chemistry lab, but with some differences. Computational chemistry lab notebooks will include discussions of the computational approach, the programs or software used to perform the calculations, the types of calculations performed, specific file names for submitted jobs, and drawings of the chemistry being evaluated.

23.1.5 Presenting Results:

Upon completion of a research project, students can present their results using one of three formats typically found in the professional scientific community. The first is the creation of a poster, for display at a poster session at a scientific conference. Student researchers typically stand next to the poster, and answer questions as other scientists or student scientists visit the poster. Students can also prepare a lab abstract of the research. Lab abstracts are short narratives, approximately 250-300 words in length, and contain the purpose, computational approach, example data and sample results, and a conclusion. For larger computational chemistry projects, students can also create a journal-type article. Articles are typically 3-10 pages in length, and follow a specific set of guidelines. At the North Carolina School of Science and Mathematics (NCSSM), student computational chemistry researchers follow the *Journal of Computational Chemistry* author guidelines.

23.1.6 Sample Project Titles:

In the reading below, sample project titles are listed to help the educator and student researcher get a sense of the types of projects and project titles might be suitable for student research.

23.2 Choosing a Research Problem:

What do you want to know? Any research effort starts with this question. While it sounds like a simple question, the answer is anything but simple. Generating a research question that is appropriate to the research situation is both an art and a science. What, however, does *appropriate to the research situation* mean? For the educational audience (for whom this book is written), this typically means a research project being done to satisfy the requirements for a course in computational chemistry, or for a project that might be submitted to one or more science competitions. In this chapter, the focus will be on student projects of both short- and long-duration.

All good research starts with an understanding of the **technologies, techniques, and tools of computational chemistry**. By the time a student researcher is considering a research project, s/he should have a good practical understanding of what types of problems computational chemistry can solve (*technologies*), an understanding of how to use the available software, submit jobs, and analyze results (*techniques*), and be able to choose the most appropriate software for the problem (*tools*). By the time the student is ready for research, s/he should have run a dozen or more jobs using several different software packages, and have successfully completed both structured labs (teacher directed) and smaller, open-ended (teacher guided) lab activities.

In developing a research question for a student project, there are several guiding questions that can help direct the student:

- **What area of chemistry are you most interested in?** Answers to this question might be topics such as these (with examples):

  1. organic chemistry (structures and/or mechanisms)
  2. medicinal chemistry (comparison of energies of substituted lead drugs)
  3. environmental chemistry (rates of hydroxyl radical degradation in the atmosphere)
  4. inorganic chemistry (excitation states of coordination complexes)
  5. reaction chemistry (prediction of kinetics, thermodynamics, transition states, etc.)

- **What resources do you have at your disposal?** Resources specifically means computing resources, both time and software availability. If the researcher is using a shared (distributed) resource such as the North Carolina High School Computational Chemistry server, then there has to be a recognition that the student is sharing computing time and cannot submit numerous jobs, especially simultaneously-submitted jobs. The student researcher (with support from the teacher-mentor or external mentor) needs to coordinate the submission of jobs with the server administrative team. In terms of software, the student needs enough familiarity with the various software packages to know what types of problems can be solved, and perhaps more importantly, what types of problems *cannot* be solved with the available computing resources.

- **How much time do you have?** For an end-of-course project, time constraints put significant limitations on the scope of the project. The number of jobs that need to be completed for the project to be successful probably needs to be 20 or less, with each job requiring 10 minutes or less of compute time per job. For longer term projects (such as a research course or science competition project), the number and length of jobs scales appropriately. The table below shows guidelines for how to determine the number of jobs and the size of the jobs. For example, if the project must be completed in a week (as in a small classroom assignment), the student should probably consider running no more than 8 jobs, each requiring on average no more than 66 seconds (2.1 minutes) per job. Larger research projects, such as one lasting 10 weeks (perhaps as a summer research project or independent school year project) might result in 80 or more submitted jobs, with single jobs requiring 21 minutes or more. Assuming the multiplier of **number of jobs** *times* **time per job**, 84 jobs times 21 minutes suggests a total CPU computing time of about 30 hours. For larger research projects, one run might require that much time. **As a rule-of-thumb, any research project that anticipates needing more than 100 total CPU hours of time must coordinate the scheduling of jobs with the server system administrative team!**

Available Time (Calendar days)	Number of Jobs	Time per Job (minutes)
2	2	0.6
4	5	1.2
7	8	2.1
10	12	3.0
14	17	4.2
21	25	6.3
28	34	8.4
35	42	10.5
42	50	12.6
49	59	14.7
70	84	21.0

Table 23.1: Guidelines for determining number and size of jobs, based on research time available

Graphically, this scaling can be represented as in Figure 24.1. The graphic shows the number of days available for the project on the x-axis and the total CPU time suggested on the y-axis. It is important to remember that the table and related chart are simply suggestive, not prescriptive!

- **Is there a particular category of computations that is of most interest?** By this, we mean the following three items:

    1. **Structure:** projects based on *structure* typically focus on trying to determine the optimum, or optimized, geometry of one or more molecules. A research project might, for example, try to determine which *model chemistry* might work best for a specific type of molecule or molecular family, such as cycloalkanes. Or, a project might try to evaluate how well computation predicts the known experimental structure, perhaps using data from the Computational Chemistry Comparison and Benchmark Database (*http://srdata.nist.gov/cccbdb/*). Structure-based projects might also consider trying to determine a suitable transition state structure for a specific reaction mechanism.
    2. **Property:** projects based on *property* typically are looking to determine those characteristics of a molecule or group of molecules that exist in that molecule regardless of the presence of other molecules. For example, a property-based research project might look at determining the vibrational frequencies and IR spectra of an organometallic compound. Another project might look at determining the gas-phase basicities of a compound or family of compounds. Property-based projects might also include investigations to find fundamental quantum descriptors such as dipole moments, polarizabilities, molecular orbital energies, and

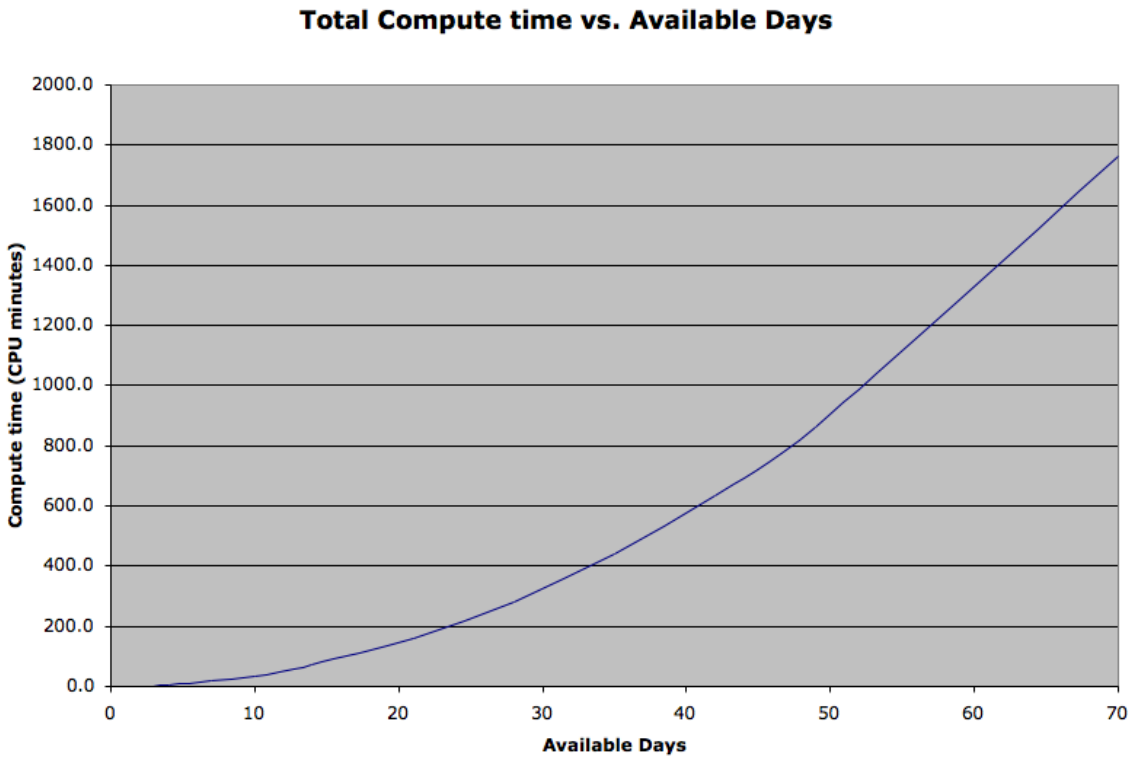


Figure 23.1: Scaling of available time in days vs. total CPU compute times

spectroscopic information. One category of property-based projects involves the use of the techniques of QSPR (quantitative structure-property relationships), in which regression mathematics is used to try to capture the mathematical relationships between some property of the molecule and one or more structure-based quantum descriptors.

3. **Activity:** projects based on *activity* are probably better described as *reactivity*. Activity-based projects typically look at applying computational techniques to reaction mechanisms. A project that looks to investigate the S_N1 and S_N2 reaction mechanisms in organic chemistry is a good example of an activity-based problem. Other projects might look at determining heats of formation and other thermodynamic parameters of some reaction.

23.3 Applying for a Research Account:

As more computational chemistry tools become available (or affordable, as the case may be) for individual computers, the need to apply for a research account will no longer be

necessary. In many cases, however, a researcher applies for time on a large computer. Student researchers requiring the resources of a distributed computing system such as the North Carolina High School Computational Chemistry server must submit a research proposal to the scientific team that maintains the server. This is excellent practice for students who will continue into the area of high performance computing, also known as supercomputing. It is still common practice for professional scientific researchers to submit proposals to institutions such as the National Center for Supercomputing Applications (NCSA) and the San Diego Supercomputing Center (SDSC) for supercomputing time.

Student researchers requiring an account on the NC HS Computational Chemistry server must also submit a research proposal in order to get an account. Project proposals should contain the following pieces of information:

1. **Paper Title:** at the proposal stage, this can be a working title (i.e., not necessarily the final title). The title should be descriptive enough to provide the server administrators with a sense of what problem is the focus of the study. An example title might be: `Determination of the Transition State of the Decomposition of Formaldehyde`.

2. **Paper Abstract:** the student researcher should prepare a short abstract describing the main goal(s) of the research, along with a description of how the researcher might approach the solution of the problem (in other words, the *computational approach*). Of critical importance at this stage is an indication of what *model chemistry* the student researcher might be using. Model chemistry selection is described in greater detail below.

3. **Software requirements:** in submitting a proposal, the researcher should also indicate which software package or packages s/he might require for the research. The support team can often advise the researcher whether or not the choice of a software tool is the most appropriate.

4. **Per job time limit:** in any distributed computing system (i.e., a computer that supports multiple users), one of the primary responsibilities of the server support team is to ensure that there are enough computing resources for all of the users. In attempting to predict the per job time limit, the student shows whether or not s/he has an realistic understanding of how long the computations will take.

5. **Total CPU time limit:** as with per job time limit, the student researcher should attempt to predict the total computing time necessary. A simple algorithm such as the number of total jobs that will be needed multiplied by the average amount of time for each job should work for predicting total time. Student researchers (and even experienced researchers) tend to underestimate both the per job and total time limits, so it is not uncommon to scale, or increase, the predicted time limits by 10% or more.

6. **Name of teacher(s)/mentor(s):** until a researcher acquires *principle investigator* status, s/he typically conducts research under the watchful eye of one or more mentors. High school student researchers, just like undergraduate or graduate students, are hopefully being guided by a research mentor, such as the school chemistry educator and/or a mentor at a local university. The amount of support that the student researcher will receive, along with the name(s) of the mentor(s), should be described in the proposal.

Preparing a research proposal not only prepares the student researcher for the real world, but also provides the computational chemistry support team with the information needed to ensure adequate computational resources for all users.

23.4 Choosing a Model Chemistry:

For all but the most basic research projects, one of the critical determinations is the choice of a *model chemistry*. The student researcher needs to make decisions about which level of theory s/he might need to obtain the appropriate data for a specific research question. The goal is always to choose the simplest level of theory that will yield adequate results. If, for example, the student can use a *semi-empirical* approach rather than an *ab initio* approach, then that is the most appropriate approach. The student must work to **choose a level of theory that results in the most accurate data requiring the least amount of computational time**.

If the student researcher is using an *ab initio* or *DFT* quantum method, s/he also needs to choose a basis set that provides good data at a minimal computational expense. For large research projects that are intended for submission to one or more scientific competitions, it is suggested that the student use at least a 6-31G or higher basis set (unless, of course, the research project is looking at evaluating the proper choice of a basis set!). Students are encouraged to use smaller, less accurate basis sets for their preliminary calculations in order to get a sense of what the results might look like. When, however, they are ready for *production runs*, meaning the final calculations that will be analyzed and reported, larger basis sets are encouraged.

Most published research articles report a model chemistry that is used to ensure that the molecule is geometry optimized and then the model chemistry used for the calculation(s). For example, the student might indicate that s/he will use an HF/6-31G model chemistry for the geometry optimization, and a B3LYP/6-31G(p,d) model chemistry for the final calculations. If the student does not have a basic understanding of model chemistries, s/he is probably not ready to engage in a large-scale computational chemistry research project.

23.5 The Computational Chemistry Notebook:

Keeping a notebook in computational chemistry research is just as important as keeping a notebook in any other research program. There are, however, some differences in how one might keep a notebook in computational research versus that in a wet lab. Dr. John Hanson of the University of Puget Sound has developed an excellent model for keeping a computational science notebook. Figure 24.2 shows an example computational chemistry notebook page with numeric labels:

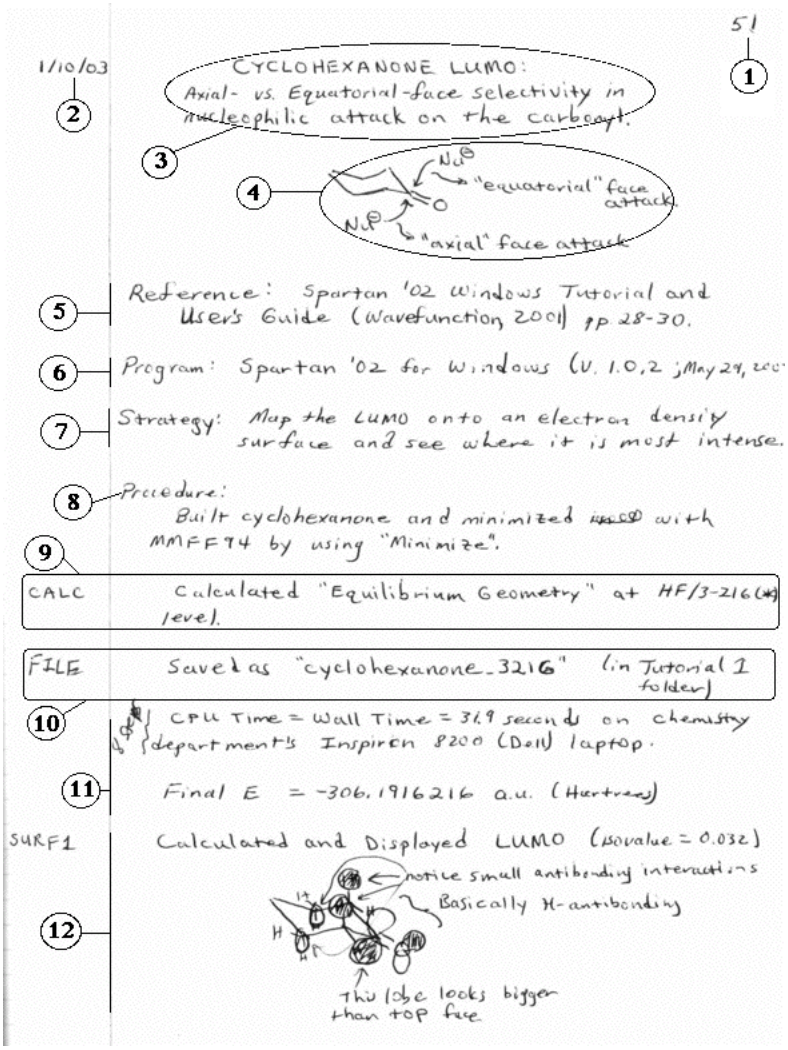


Figure 23.2: Notebook example courtesy of Dr. John Hanson, University of Puget Sound, *http://www2.ups.edu/faculty/hanson/c455.07/intro.htm*

1. **Page number (mandatory)**
2. **Date (mandatory)**
3. **Title of the lab or project (mandatory)**
4. **Drawing of the project (optional):** for most computational labs, a drawing of the molecule(s) and/or the reaction mechanism is useful and appropriate. In this case, the drawing shows the mechanism of the nucleophilic attack on the the carbonyl.
5. **References to the literature (optional, but normally a part of the notebook)**
6. **Program(s) used to perform the calculation(s) (mandatory)**
7. **Computational approach (mandatory):** The computational approach is listed here as the strategy. The strategy is not a procedure; rather, it describes the overall plan for the project
8. **Procedure (mandatory):** as typical, the procedure describes in some detail how the calculations will be performed
9. **Calculation (mandatory):** a description of the model chemistry used in the lab. A good notational system is `HF/6-31G(d)//B3LYP/6-31G(p,d)`, where the first model chemistry describes how the molecule was optimized, and the second model chemistry describes how the calculations were performed
10. **File (mandatory):** this section lists the file name(s) for the calculation(s)
11. **Basic data (mandatory):** this section lists the primary data for the calculations, including how long the runs required (shown here as both CPU time and wall, or clock, time), and the final energy of the molecule
12. **Data (mandatory):** this section might include a table of data and/or a drawing of the visualizations. For example, in this lab notebook, the researcher has sketched the molecular orbitals of the molecules, describing the interaction of the MOs
13. **Summary (mandatory):** not included in this example, the lab should conclude with a summary statement

23.6 Presenting Results

There are primarily three ways to present the results of a computational chemistry research project:

1. Poster
2. Lab abstract
3. Journal-type article

Each is described in more detail as follows.

23.6.1 Poster

Most educators and students are familiar with this option. Posters and poster sessions are common occurrences for most students who have participated in a science fair-type activity. Experienced researchers also regularly create posters to present information. Posters are typically set up at a wide variety of scientific meetings. The researcher will often stand next to the poster, explaining the research as other scientists browse through the poster area. Figure 24.3 shows a photograph of a scientific poster session. It is the rare scientific conference that does not sponsor at least one poster session!

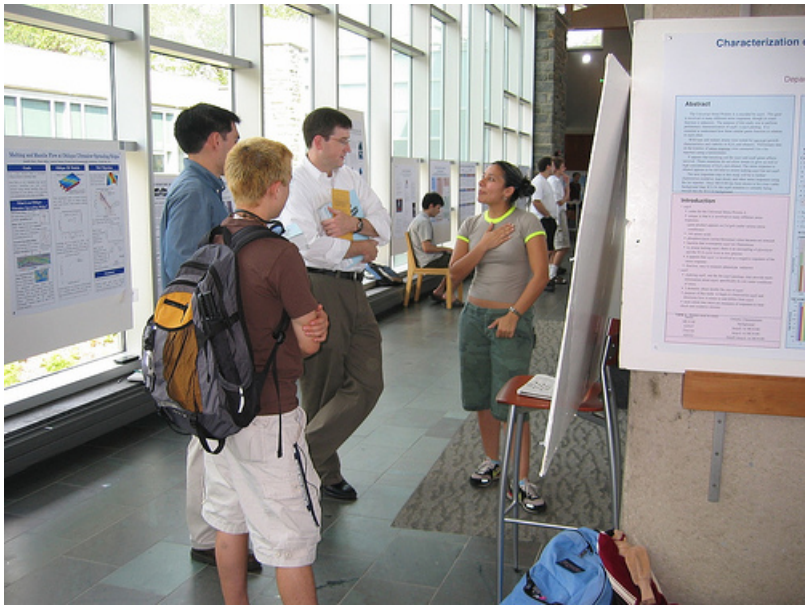


Figure 23.3: A scientific poster session. Image courtesy of Swarthmore College, http://www.swarthmore.edu/NatSci/cpurrin1/posteradvice.htm

In designing a poster for a computational chemistry project, one may or may not follow the typical hypothesis-based approach described in most readings on the scientific method. A computational chemistry poster should *at a minimum* contain the following sections:

1. Abstract
2. Purpose of the Project
3. Scientific Background
4. Computational Approach
5. Data/Example Data
6. Results and Discussion
7. References

23.6.2 Lab Abstract

Most if not all journal articles begin with a short (250-300 word) abstract describing the research work. For smaller research projects, such as a small class project or shorter independent research project, the presentation of results in a journal abstract form is a very effective tool. Students often comment that it is easier to write a complete report rather than the shorter lab abstract!

As an example, students in a computational chemistry class were given a small, one-week research project to evaluate which theoretical method, or *model chemistry* might provide the best value for the heat of formation for the protonation reaction of pyridine[1]. A sample lab abstract for this project is shown below:

> **Abstract:** The choice of a theoretical method in computational chemistry is a critical consideration for the computational chemistry practitioner. The choice of a theoretical method is evaluated as applied to the protonation of the pyridine molecule, a benzene-like cyclic organic compound containing a single nitrogen atom in the ring. Geometry optimizations are performed on both a neutral pyridine molecule and a positively charged pyridinium cation, where a proton is attached to the nitrogen atom. The optimizations are performed using two semi-empirical methods, AM1 and PM3, using the MOPAC software. Geometry optimizations are also performed using a density functional theory (DFT) hybrid functional (B3LYP/6-31G(p,d) on both of the organics. Single point energy calculations are performed on the proton using AM1 and PM3. A comparison of the heats of formation for the reaction, calculated using Hess' Law, shows that the DFT theoretical method is significantly better (-231.9557007 kcal/mol, or 1.6% error as compared to the experimental value

[1] *Introduction to Computational Chemistry*, The North Carolina School of Science and Mathematics, Spring 2007. Project design courtesy of Dr. Clyde Metz, Department of Chemistry, College of Charleston, SC

of -219.2 kcal/mol) than that of the PM3 method (-196.69071 kcal/mol, or 5% error) or the AM1 method (-162.79073 kcal/mol, or 10% error).

The basic format for the lab abstract is as follows:

1. **Purpose:** typically, a one- or two-sentence description of the goal of the project. In the sample abstract, the writer describes why the work is important and provides a broad description of what the research evaluates

2. **Computational approach:** the writer describes the type of calculations performed, in this case through geometry optimization and single point energy calculations using semi-empirical and hybrid density functional theory (DFT) techniques

3. **Example data:** given the size of this research, the abstract does not report example data. For a larger work, the abstract would try to give the reader a flavor of the types of data that was collected from the computations

4. **Sample results:** again, given the size of this project, the results section of the abstract reports all of the final results. In this project, the writer reports the final heats of formation for reaction for the three theoretical methods and the percent error calculations from the experimental value

5. **Conclusion(s):** the last line of this abstract, which also contains the sample results, states the conclusions, reporting that DFT is better than the AM1 or PM3 semi-empirical methods for this particular chemical reaction

In classes conducted at the North Carolina School of Science and Mathematics, students are required to submit a copy of their lab notebook and a final abstract for all labs performed. Typically, students conduct their computations in teams of two or three students, can collaborate and consult with each other on the data analysis, but are required to write their own lab abstract. This practice helps to ensure that each student learns how to write clearly and concisely.

23.6.3 Journal-type article

For larger classroom projects, and certainly for independent research projects, students are required to prepare a complete journal-type article. At the North Carolina School of Science and Mathematics, students use the *Journal of Computational Chemistry* as the model for the journal article.

1. **Title:** students need to be creative in finding a title that appropriately captures the research work without being too long. We counsel students to avoid titles such as *A Computational Study of .....*

2. **Author(s):** in the classroom, this section provides us with an opportunity to introduce students to the idea of *first and second authors*, a phenomenon which they will hopefully experience at the undergraduate, graduate, and profession levels.

3. **Institution:** writers list their institutions. If more than one institution is involved, the order follows the order of the authorship

4. **Keywords:** students are required to identify 3-8 keywords that best capture the focus of the research. We encourage students to do this last.

5. **Abstract:** the abstract follows the same format as described above. Students are encouraged to write the abstract last. For the final exam in the NCSSM computational chemistry class, we often give students an article from the *Journal of Computational Chemistry* with the abstract missing and give them 90 minutes to read the article and prepare a suitable abstract for the article.

6. **Introduction:** in this section, the writer presents the chemistry of the research. The challenge in this section is to provide the reader with enough background information such that s/he can then read the rest of the article, but not have so much background information that the article becomes lengthy. Student writers need to be able to make some assumptions about the readership of the journal for which the article might be submitted.

7. **Computational Approach:** in this section, the writer describes in some detail the types of calculations performed, the model chemistries chosen (if appropriate), and the specific software tools used. This section should contain enough detail such that another computational chemist could reasonably duplicate the work. Again, the student writer needs to make some assumptions about what the reader already knows how to do.

8. **Results and Discussion:** in this section, the writer presents most if not all of the data. Data can be presented in appropriately labeled and captioned data tables and graphical formats, with a written discussion referencing the data tables and graphics. For larger projects, students need to make decisions about how much data to report in the article.

9. **Conclusion(s):** the conclusion section provides the writer with a chance to present his or her analysis of the data results, with the overall goal of providing the reader with some important and/or useful understanding of the question being evaluated. The writer might also suggest ideas for further evaluation based on the results of this work.

10. **Acknowledgment:** the writer recognizes any individual mentor or other organization providing support for the work. For students at NCSSM, we require them to

recognize the funding support provided by the Burroughs Wellcome Fund and the North Carolina Science, Mathematics and Technology Center with this notation in the Acknowledgments section:

> Appreciation is also extended to the Burroughs Wellcome Fund and the North Carolina Science, Mathematics and Technology Center for their funding support for the North Carolina High School Computational Server.

Student researchers using the Global Grid Exchange computational chemistry server would follow suit, using an acknowledgment as follows:

> Appreciation is also extended to the Global Grid Exchange, Parabon Computation, and Cisco for their funding support for the Computational Chemistry Server for Pre-College Students.

This type of requirement helps students understand the role of funding agencies that support science and scientific research.

11. **References:** learning how to properly prepare a references section is an important skill for student writers. The school media specialist can often help with this aspect. For computational chemistry articles, the student must cite the use of specific software packages, servers, etc. A standard citation format is used for these entities. The reader is encouraged to look at the example journal article written for the pyridine protonation lab, found on this website, for examples of how to cite WebMO, the North Carolina High School Computational Chemistry server, and the various software tools.

23.7 Sample Project Titles

Below is a list of project titles for a variety of student-designed research projects conducted over the past several years.

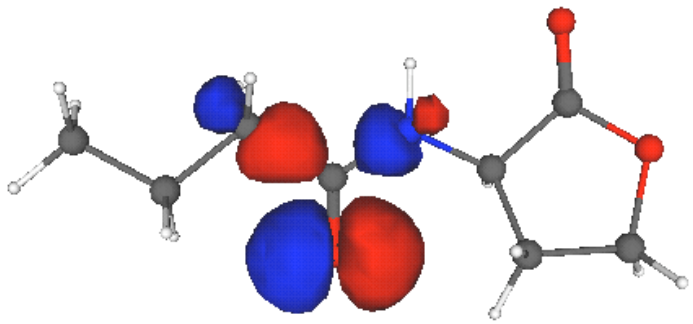
- Transition State of a Creatine Molecule during Dehydration
- The Diels-Alder Reaction
- Comparison of Relative Sweetness to Molecular Properties of Artificial and Natural Sweetners
- Comparative Study: Sarin and VX
- Is there a transition state for the insertion of ethylene into the Ziegler-Natta catalyst?
- Gaussian94 Analysis of C60

- GAMESS Animation Study of LiH
- Transition State Study of a Diels-Alder Reaction
- Transition State Study of a Cocaine Molecule
- Basicities of Amines
- Comparison of the Bonding Properties of Serotonin and Lysergic Acid
- Conformational Anaysis Study of n-Butane
- Transitional State Study of ATP
- Potential Energy Scan of an Ester using Gaussian94

North Carolina High School
Computational Chemistry Server

Chapter 24

Computational Chemistry Research

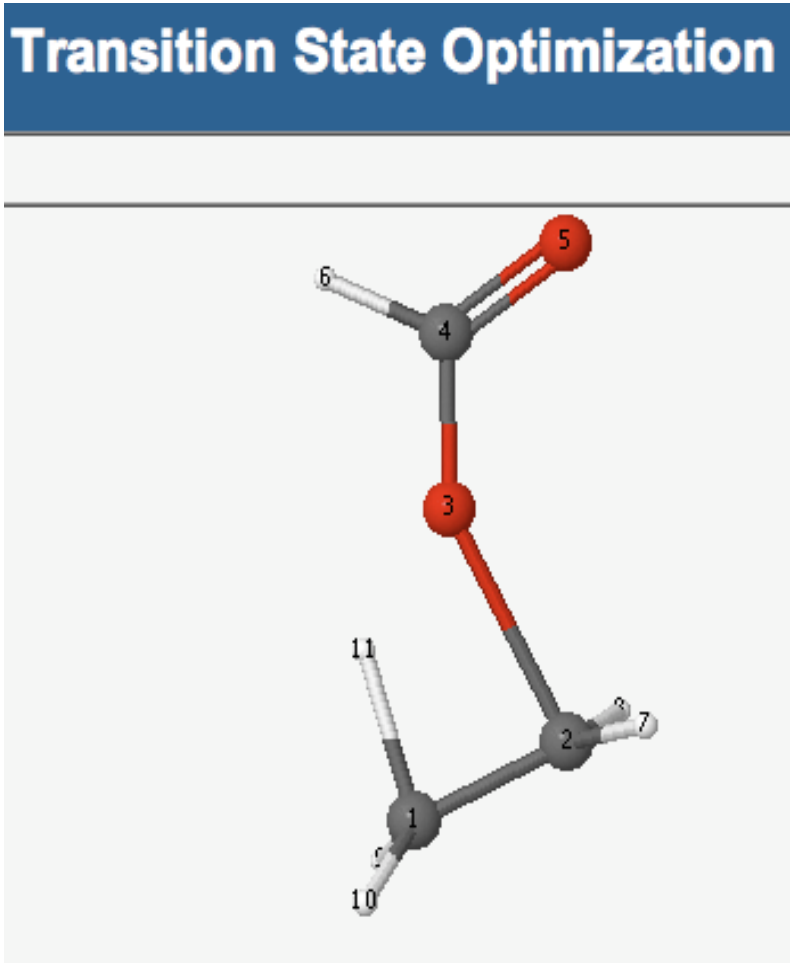


 Copyright© 2007 All Rights Reserved

North Carolina High School Computational Chemistry Server

Contents

- Choosing a Research Problem
- Applying for a research account
- Choosing a Model Chemistry
- The Computational Chemistry Notebook
- Presenting Results
- Sample Project Titles



 Copyright© 2007 All Rights Reserved



Choosing a Research Problem

- Challenge:
  - Choosing a project appropriate to the research situation
    - Appropriateness defined
      - Do-able in the amount of time available
      - Within the cognitive and experiential abilities of the student researcher
      - Within the limits of available resources
        - » Software, job limits, other resources

 Copyright© 2007 All Rights Reserved



Research Guiding Questions

- What area of chemistry is most interesting?
  - Organic chemistry
    - Structures and mechanisms
  - Medicinal chemistry
    - Comparison of energies of lead drugs
  - Environmental chemistry
    - Rates of hydroxyl radical degradation in the atmosphere
  - Inorganic chemistry
    - Exitation states of coordination complexes
  - Reaction chemistry
    - Prediction of kinetics, thermodynamics, transition states

 Copyright© 2007 All Rights Reserved



Research Guiding Questions

- What resources do you have at your disposal?
  - Software
    - Comp chem server cannot do protein folding/docking, crystal structures, etc.
  - Job time limits
    - Computational resources are not unlimited
  - Mentors
    - Are there intellectual resources available, i.e smart people?
      - University mentors, online mentors, local educators, etc.

 Copyright© 2007 All Rights Reserved



Research Guiding Questions

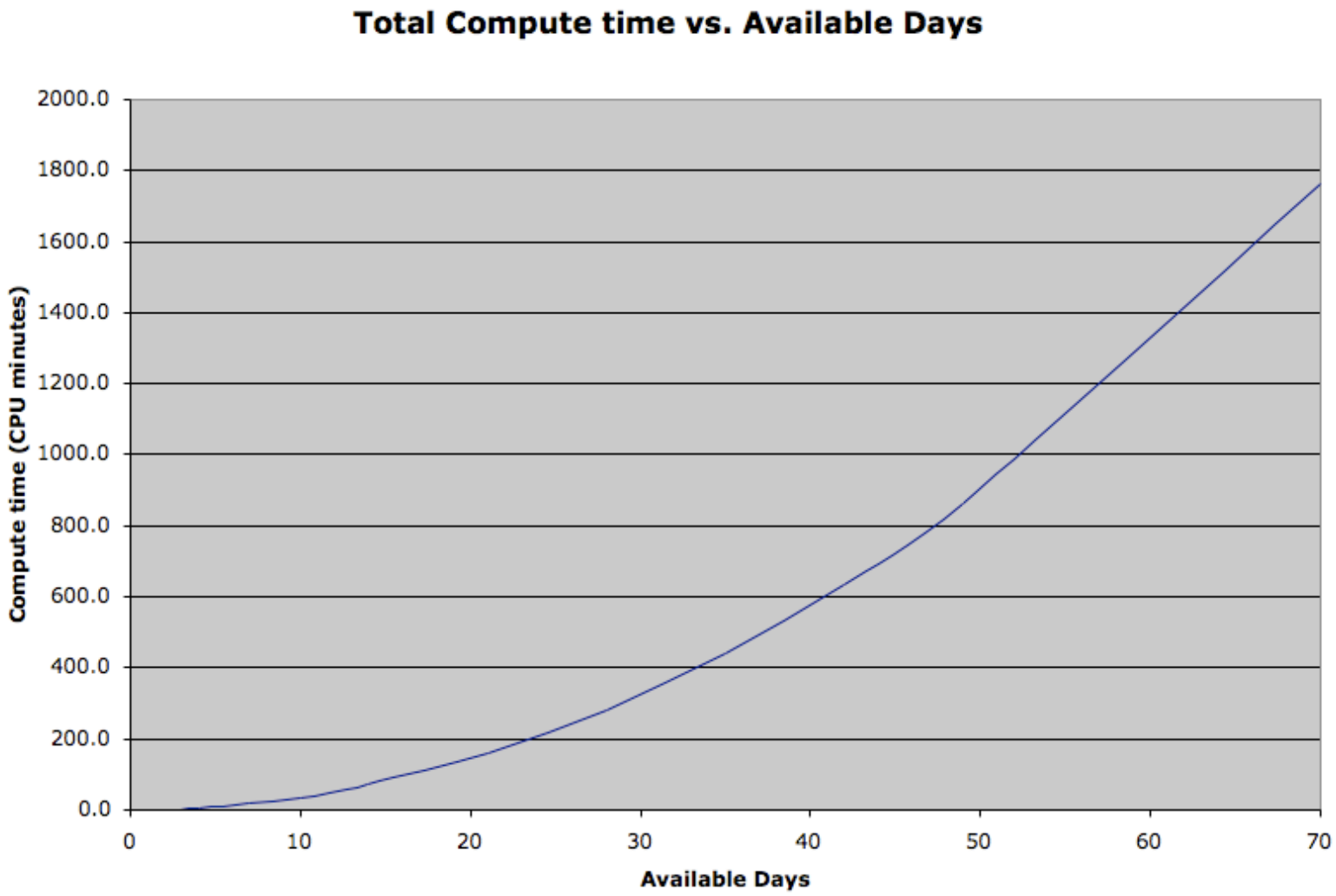
- How much time do you have?

Amount of time available for the project (calendar days)	Number of Jobs	Length per job (minutes)	Total Compute time (CPU minutes)	Total Compute time (CPU hours)
2	2	0.6	1.4	0.0
4	5	1.2	5.8	0.1
7	8	2.1	17.6	0.3
10	12	3.0	36.0	0.6
14	17	4.2	70.6	1.2
21	25	6.3	158.8	2.6
28	34	8.4	282.2	4.7
35	42	10.5	441.0	7.4
42	50	12.6	635.0	10.6
49	59	14.7	864.4	14.4
70	84	21.0	1764.0	29.4

 Copyright© 2007 All Rights Reserved

North Carolina High School
Computational Chemistry Server

Research Time Chart



 Copyright© 2007 All Rights Reserved



Research Guiding Questions

- Is there a particular category of computations that is of most interest?
  - Structure:
    - Geometry optimizations based on model chemistry
    - Comparison of computational results to experimental results
    - Transition state geometries
  - Property:
    - Determination of spectra
    - Calculation of quantum descriptors (QSPR)
  - Activity:
    - Reaction mechanisms
    - QSAR-types of problems

 Copyright© 2007 All Rights Reserved



Applying for a research account

- Real world:
  - Researchers apply for computing time on large machines
    - How? By submitting a *research proposal*
- Student researchers:
  - Also required to submit research proposal
  - Done online via online form

- Components:
  - Paper Title
  - Paper Abstract (250-300 words)
  - Software Requirements
  - Per job time limit
  - Total CPU time
  - Name of teacher/mentor

 Copyright© 2007 All Rights Reserved



Choosing a Model Chemistry

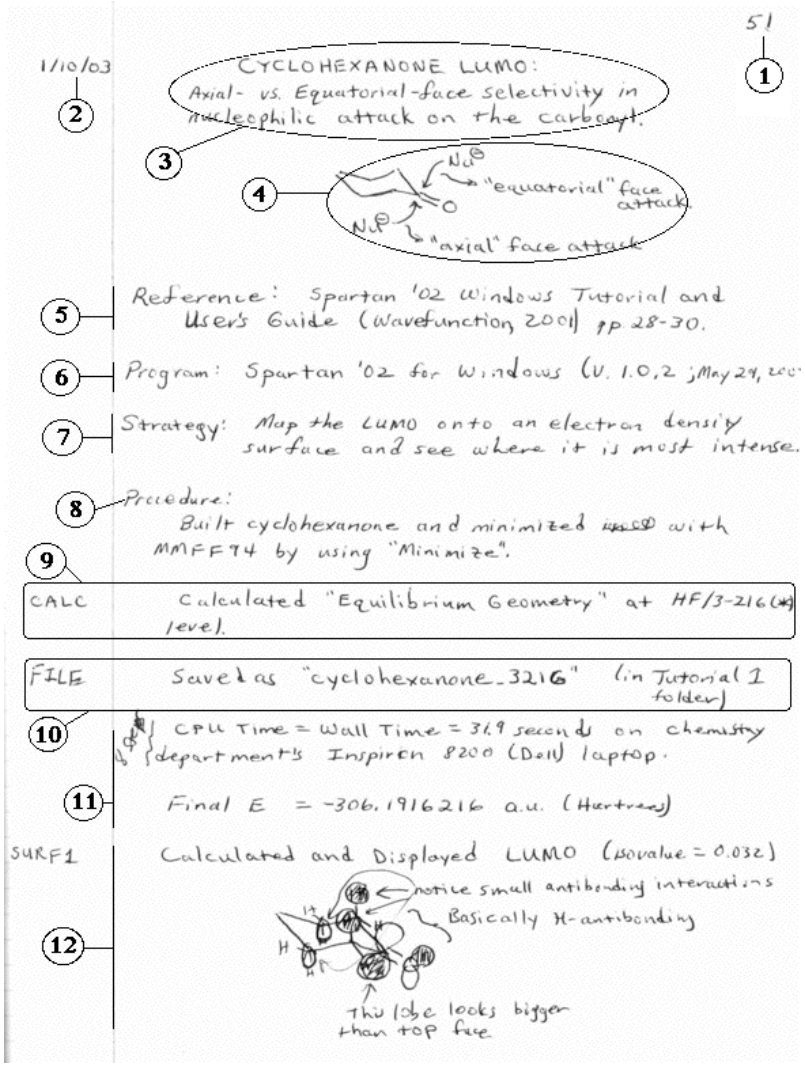
- Goal
  - Choose the simplest model chemistry that produces the data needed to answer the research question
- Tradeoff
  - Computational accuracy vs. compute time (resource stewardship)
- Classroom research projects
  - Semi-empirical typically adequate
    - AM1 or PM3
  - Smaller basis sets are adequate
    - STO-3G
    - 3-21G
- Independent research
  - 6-31G basis sets for final runs
  - Can do prelims with smaller basis set

 Copyright© 2007 All Rights Reserved

North Carolina High School
Computational Chemistry Server

The Computational Chemistry Notebook

- Parts
  1. Page number
  2. Date
  3. Title of project
  4. Drawing of project
  5. References to the literature
  6. Program(s) used in calculations
  7. Computational approach
  8. Procedure
  9. Calculation type
  10. File name(s)
  11. Basic data results
  12. Data
  13. Summary

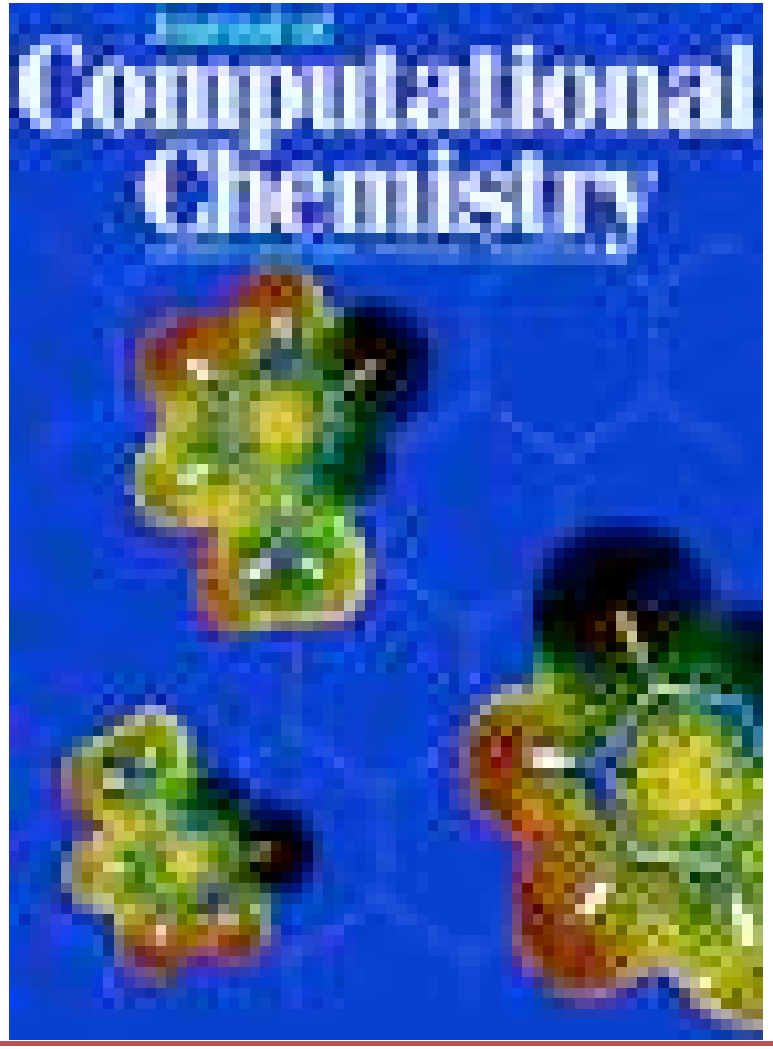


 Copyright© 2007 All Rights Reserved

North Carolina High School Computational Chemistry Server

Presenting Results

- Three methods:
  - Poster
  - Lab Abstract
  - Journal-type article
- Upcoming
  - *Student Journal of Computational Chemistry*



 Copyright© 2007 All Rights Reserved

North Carolina High School Computational Chemistry Server

Poster

- Standard format at scientific professional meetings
- Can use standard tri-fold cardboard display
- PowerPoint formatted - ~$150 at Kinkos for printing



 Copyright© 2007 All Rights Reserved

North Carolina High School Computational Chemistry Server

Lab Abstract

- Contains the basic parts of the research
  - Purpose
  - Computational approach
  - Example data
  - Sample results
  - Conclusion(s)
- Typically 250-350 words

Abstract: The choice of a theoretical method in computational chemistry is a critical consideration for the computational chemistry practitioner. The choice of a theoretical method is evaluated as applied to the protonation of the pyridine molecule, a benzene-like cyclic organic compound containing a single nitrogen atom in the ring. Geometry optimizations are performed on both a neutral pyridine molecule and a positively charged pyridinium cation, where a proton is attached to the nitrogen atom. The optimizations are performed using two semi-empirical methods, AM1 and PM3, using the MOPAC software. Geometry optimizations are also performed using a density functional theory (DFT) hybrid functional (B3LYP/6-31G(p,d) on both of the organics. Single point energy calculations are performed on the proton using AM1 and PM3. A comparison of the heats of formation for the reaction, calculated using Hess' Law, shows that the DFT theoretical method is significantly better (1.6% error as compared to the experimental value) than that of the PM3 method (5% error) or the AM1 method (10% error).

 Copyright© 2007 All Rights Reserved



Journal-type article

- Length: 3-10 pages
- Contents
  - Title
  - Author(s)
  - Institution
  - Keywords
  - Abstract
- Introduction
- Computational Approach
- Results and Discussion
- Conclusion(s)
- Acknowledgement
- References

 Copyright© 2007 All Rights Reserved



Sample Project Titles

- Symmetry in Superconductors
- Transition State of a Creatine Molecule during Dehydration
- The Diels-Alder Reaction
- Comparison of Relative Sweetness to Molecular Properties of Artificial and Natural Sweetners
- Comparative Study: Sarin and VX
- Is there a transition state for the insertion of ethylene into the Ziegler-Natta catalyst?
- Gaussian94 Analysis of C60
- Potential Energy Scan of an Ester using Gaussian94
- GAMESS Animation Study of LiH
- Transition State Study of a Diels-Alder Reaction
- Transition State Study of a Cocaine Molecule
- Basicities of Amines
- Comparison of the Bonding Properties of Serotonin and Lysergic Acid
- Conformational Anaysis Study of n-Butane
- Transitional State Study of ATP

 Copyright© 2007 All Rights Reserved

Determination of the Choice of Theoretical Method for a Pyridine / Protonated Pyridine System

R. Gotwals
North Carolina School of Science and Mathematics, Durham, NC

Received 3 April, 2007; Accepted 26 April, 2007
Published online on Comp Chem Moodle (moodle.ncssm.edu)

Abstract: The choice of a theoretical method in computational chemistry is a critical consideration for the computational chemistry practitioner. The choice of a theoretical method is evaluated as applied to the protonation of the pyridine molecule, a benzene-like cyclic organic compound containing a single nitrogen atom in the ring. Geometry optimizations are performed on both a neutral pyridine molecule and a positively charged pyridinium cation, where a proton is attached to the nitrogen atom. The optimizations are performed using two semi-empirical methods, AM1 and PM3, using the MOPAC software. Geometry optimizations are also performed using a density functional theory (DFT) hybrid functional (B3LYP/6-31G(p,d) on both of the organics. Single point energy calculations are performed on the proton using AM1 and PM3. A comparison of the heats of formation for the reaction, calculated using Hess' Law, shows that the DFT theoretical method is significantly better (1.6% error as compared to the experimental value) than that of the PM3 method (5% error) or the AM1 method (10% error).

Key words: theoretical method, model chemistry, pyridine, pyridinium, protonation, heats of formation

Introduction

One of the most critical decisions to be made by the practitioner of computational chemistry is that of choosing the most appropriate *theoretical method*. The chemist must make a decision as to which theoretical method is best suited for both the problem to be solved, the accuracy of the data needed to address the research question, and the computational resources (software and computational time) available to perform the calculations.

Most computational chemists have the knowledge and resources to choose from one of four methods: 1) molecular mechanics/molecular dynamics; 2) semi-empirical methods; 3) ab *initio* methods; and 4) density functional theory (DFT) methods. These four areas represent, in the broadest terms, the four levels of theory, or theoretical methods that can be applied to any computational chemistry research problem. The choice of a theoretical method is often predicated on logistical considerations such as the amount of time one has to perform the calculations, the amount of computing resources available, the need to share resources with other researchers, and the size of the molecule. As an example of the last item, it is conventional wisdom that *ab initio* quantum chemical techniques are generally limited to molecules that are under 100 atoms in size. The practical number for *ab initio* calculations tends to be closer to 50 atoms or less.

The choice of a theoretical method is also referred to as the *model chemistry*. In choosing a model chemistry, one describes both the specific form of the broad-level theoretical method. For example, under the category of *ab initio* quantum methods, one can choose a variety of methods, such as *Hartree-Fock, Moller Plessett,* or one of the Configuration Interaction (CI) methods. These are typically referred to in describing the model chemistry by the commonly accepted acronyms, such as HF, MP2, or CIS, respectively.

The model chemistry description, particularly for the quantum methods (*ab initio* and DFT), also includes a notation of the *basis set* used in the calculation. The basis set represents that set of

numbers that is used to begin the determination of the wavefunction, which in term determines the atomic orbitals (AOs) and, using the *linear combination of atomic orbitals (LCAO)* approximation, the molecular orbitals (MOs). As a general rule, the larger the basis set, the more accurate the results of the calculation (and also the more computing time needed to perform those calculations!).

In describing the model chemistry, a standard format is used. The level of theory and the basis set used to ensure that the molecule has been optimized geometrically is first reported, and then the level of theory used to perform the calculations (separated by a double //) is reported. For example, supposing that a molecule has been optimized geometrically using a Hartree-Fock level of theory with a STO-3G basis set, then the calculations performed with a Hartree-Fock level of the theory and a 6-31G(p,d) basis set, one would describe the calculation as follows:

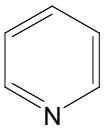
HF/STO-3G//HF/6-31(p,d)

The choice of a theoretical method, or model chemistry, is of considerable interest to the computational practitioner. As more methods are developed and more experience are gained in the use of a variety of theoretical methods, so too will our abilities to choose the right method for the right problem. Our interest here is to evaluate the optimal model chemistry for studying the protonation of organic ring structures. As a model organic structure, we have chosen pyridine, a benzene-like ring structure that has a substituted nitrogen atom, replacing one of the carbons in the ring. Like benzene, pyridine has an alternating double bond structure reflecting the resonance of the molecule. In this research, we are modeling the protonation of the nitrogen, with the goal of determining the change in the *heat of formation (ΔH_f)*. The reaction heat of formation will be calculated using Hess' Law:

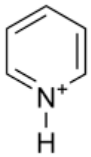
$$\Delta H_{f_{rxn}} = \Sigma \Delta H_{f_{products}} - \Sigma \Delta H_{f_{reactants}} \quad (1)$$

Computational Approach

Using the molecular editor builder of WebMO[1] on the North Carolina High School Computational Chemistry Server[2], the molecules pyridine, protonated pyridine, and a proton (H+) were built. The structure of the pyridine molecule is shown below:

pyridine 

The protonated version of this molecule, known as a pyridinium cation, adds a hydrogen to the nitrogen atom, with a subsequent increase in the charge of the molecule to a cation (positively-charged ion):



Following building, each molecule was optimized using the "comprehensive cleanup" molecular mechanics package found in the WebMO molecular editor. Following this rough optimization, pyridine and protonated pyridine were optimized with MOPAC[3], a semi-empirical software package. Each of two molecules was optimized three times from an initial build followed by a comprehensive cleanup. The first optimization was performed using MOPAC with the AM1 basis set. The second was performed with MOPAC using the PM3 basis set. Finally, the third optimization was performed using Gaussian 03[4] using a hybrid density functional theory (DFT) model chemistry, specifically the B3LYP/6-31G(p,d) model chemistry. In addition, a proton (H^+) was built. Using MOPAC, single point energies (molecular energies) were calculated using both the AM1 and PM3 basis sets.

Results and Discussion

For MOPAC calculations, all energies are reported as heats of formation with units of kilocalories per mole (kcal/mol). DFT calculations resulted in energies reported in units of hartrees (E_h). These energy values were converted to kcal/mol energy units using the conversion factor 1 E_h = 627.51 kcal/mol.

The data results are shown in Table 1.1.

Molecule	AM1 (kcal/mol)	PM3 (kcal/mol)	B3LYP/6-31G(d) (Hartrees)
Pyridine	32.03855	30.3684	-248.2849731
Protonated	184.16282	187.26355	-248.6569778
Proton	314.915	353.58586	1.481
ΔHf	-162.79073	-196.69071	-231.9557007
Percent error	**25.73%**	**10.27%**	**5.82%**
	Experimental	-219.2	kcal/mol

Table 1.1 Computational Results

The value for the proton for the DFT calculation was obtained from literature[5]. The overall ΔH_f for the reactions were calculated using Hess' Law (Equation 1), based on the reaction:

Protonated pyridine --> Pyridine + H^+

Figure 1.1 shows a comparison of the ΔH_f for the three theoretical methods.

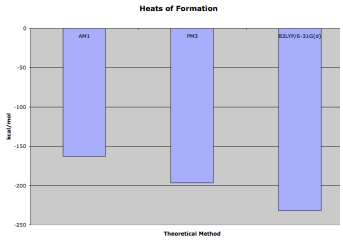


Figure 1.1. ΔH_f for methods

Percent error determinations for the three methods were calculated using the experimental value of –219.2 kcal/mol for the heat of formation for the reaction. The percent error calculations are reported in Table 1.1.

Conclusions

Based on the data and data analysis, the results suggest that the more powerful quantum chemical theoretical method – specifically, the choice of the DFT hybrid functional – provides a significant improvement in the accuracy of the ΔH_f for the protonation of pyridine. Both semi-empirical methods using the AM1 and PM3 basis sets resulted in a percent error determination greater than 10% (25.73% and 10.27%, respectively). It can be stated, therefore, that the choice of the DFT hybrid theoretical method is the most superior choice for organic protonation methods. Further studies on other organic moieties to substantiate this finding are currently underway

Acknowledgement

The author thanks Dr. Clyde Metz of the College of Charleston, SC, and Dr. Shawn Sendlinger of North Carolina Central University for assistance with this work. Appreciation is also extended to the Burroughs Wellcome Fund and the North Carolina Science, Mathematics and Technology Center for their funding support for the North Carolina High School Computational Chemistry Server.

References

1. Schmidt, J.R.; Polik, W.F. *WebMO Pro*, version 7.0; WebMO LLC: Holland, MI, USA, 2007; available from http://www.webmo.net (accessed April 2007).
2. The North Carolina High School Computational Chemistry Server, http://chemistry.ncssm.edu (accessed April 2007).
3. MOPAC Version *7.00*, J. J. P. Stewart, Fujitsu Limited, Tokyo, Japan.
4. Gaussian 03, Revision C.02, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, J. A. Montgomery, Jr., T. Vreven, K. N. Kudin, J. C. Burant, J. M. Millam, S. S. Iyengar, J. Tomasi, V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega, G. A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, M. Klene, X. Li, J. E. Knox, H. P. Hratchian, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, P. Y. Ayala, K. Morokuma, G. A. Voth, P. Salvador, J. J. Dannenberg, V. G. Zakrzewski, S. Dapprich, A. D. Daniels, M. C. Strain, O. Farkas, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. V. Ortiz, Q. Cui, A. G. Baboul, S. Clifford, J. Cioslowski, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, C. Gonzalez, and J. A. Pople, Gaussian, Inc., Wallingford CT, 2004.
5. JC Traeger, RG McLoughlin *J.Am.Chem.Soc,* **103**, 3647-3652 (1981)