PRAKTIKUM KIMIA KUANTUM

Bagian 1: Persiapan Praktikum

KASMUI

PRAKTIKUM
KIMIA KUANTUM


Buku ini bebas
didownload untuk dibaca
tetapi haram hukummya
untuk diperjualbelikan
oleh orang bukan penyusun
buku ini.

KATA PENGANTAR

Bismillāhirraḥmānirraḥīm.

Segala puji bagi Allah Swt., Tuhan Yang Maha Mengetahui, yang telah menciptakan alam semesta dengan keteraturan, ukuran, hukum, dan harmoni yang memungkinkan manusia membaca tanda-tanda kebesaran-Nya melalui ilmu pengetahuan. Shalawat dan salam semoga senantiasa tercurah kepada Nabi Muhammad saw., teladan agung dalam membangun peradaban ilmu, menuntun manusia agar berpikir jernih, meneliti dengan jujur, serta menggunakan pengetahuan untuk kemaslahatan hidup.

Buku **Praktikum Kimia Kuantum** ini disusun sebagai respons terhadap kebutuhan pembelajaran kimia modern yang semakin menuntut integrasi antara pemahaman teori, keterampilan komputasi, dan kemampuan analisis data. Kimia kuantum selama ini sering dipandang sebagai salah satu cabang kimia fisik yang abstrak, matematis, dan sulit dijangkau oleh mahasiswa pemula. Konsep seperti fungsi gelombang, operator Hamiltonian, orbital molekul, spin elektron, fungsi basis, metode Hartree–Fock, teori fungsional kerapatan, simetri molekul, energi reaksi, dan frekuensi vibrasi sering kali hanya dipelajari melalui persamaan di papan tulis atau uraian teoritis dalam buku teks. Padahal, konsep-konsep tersebut sesungguhnya dapat divisualisasikan, dihitung, dimodelkan, dan dianalisis secara lebih konkret melalui praktikum kimia komputasi.

Perkembangan teknologi komputasi telah mengubah wajah pembelajaran kimia. Laboratorium kimia pada masa kini tidak lagi hanya terbatas pada tabung reaksi, buret, pipet, neraca analitik, spektrofotometer, dan peralatan laboratorium basah lainnya. Kini hadir pula laboratorium digital, laboratorium virtual, dan laboratorium komputasi yang memungkinkan mahasiswa menyelidiki struktur dan sifat materi pada tingkat atomik dan molekuler. Melalui perangkat lunak kimia komputasi, mahasiswa dapat membangun struktur molekul, mengoptimasi geometri, menghitung energi total, memvisualisasikan orbital HOMO dan LUMO, menentukan momen dipol, mensimulasikan spektrum IR, serta menghitung energi reaksi dan perubahan termodinamika. Dengan demikian, pembelajaran kimia tidak hanya berlangsung pada level makroskopik dan simbolik, tetapi juga dapat menjangkau level submikroskopik secara lebih hidup.

Urgensi penyusunan buku ini semakin kuat karena pembelajaran kimia kuantum membutuhkan jembatan antara teori dan praktik. Banyak mahasiswa mampu menghafal bentuk umum Persamaan Schrödinger, tetapi belum tentu memahami bagaimana persamaan tersebut menjadi dasar bagi perhitungan energi molekul. Banyak mahasiswa mengenal istilah orbital molekul, tetapi belum tentu mampu melihat bentuk orbital, menafsirkan distribusi elektron, atau mengaitkannya dengan reaktivitas kimia. Banyak mahasiswa mempelajari simetri molekul, tetapi belum tentu mampu menghubungkan kelompok titik dengan momen dipol dan sifat spektroskopi. Demikian pula, banyak mahasiswa mengenal istilah energi bebas Gibbs, tetapi belum tentu mampu menghitung dan menafsirkan perbedaan kestabilan reaktan dan produk berdasarkan output komputasi.

Di sisi lain, literatur praktikum kimia kuantum yang tersedia sering kali masih terbelah dalam dua kecenderungan. Pertama, buku kimia kuantum teoritis yang sangat kuat secara matematis, tetapi tidak selalu memberi panduan praktikum komputasi yang operasional. Kedua, modul penggunaan

software yang bersifat teknis, tetapi kadang kurang mengaitkan prosedur komputasi dengan landasan teoritis kimia kuantum. Akibatnya, mahasiswa berisiko menjadi pengguna perangkat lunak yang hanya menekan tombol tanpa memahami makna ilmiah dari input, metode, basis set, parameter, grafik, dan file output yang dihasilkan. Inilah kesenjangan yang hendak dijawab oleh buku ini.

Buku ini hadir dengan orientasi yang berbeda. Ia tidak hanya disusun sebagai petunjuk menjalankan software, tetapi sebagai panduan praktikum yang memadukan **kimia kuantum, kimia komputasi, ChemCompute, dan pemrograman Python**. Setiap mata praktikum dirancang agar mahasiswa dapat menempuh dua jalur sekaligus. Jalur pertama adalah penggunaan software kimia komputasi, baik berbasis web seperti ChemCompute maupun software lain seperti Avogadro, ORCA, GAMESS, Gaussian, Gabedit, Jmol, Molden, Multiwfn, Psi4, atau PySCF sesuai ketersediaan fasilitas. Jalur kedua adalah penggunaan kode Python mandiri, sehingga mahasiswa dapat menghitung, memvisualisasikan, dan menganalisis data praktikum secara lebih terbuka, reprodusibel, dan inovatif.

Kehadiran ChemCompute dalam buku ini dimaksudkan untuk memperluas akses praktikum. Tidak semua mahasiswa memiliki komputer dengan spesifikasi tinggi. Tidak semua laboratorium kampus memiliki server komputasi, lisensi software, atau tenaga teknis yang memadai. Platform komputasi berbasis web memberi peluang agar praktikum kimia kuantum dapat dilakukan secara lebih mudah, terutama untuk latihan dasar, eksplorasi molekul sederhana, dan pembelajaran berbasis inkuiri. Namun, penggunaan platform semacam itu tetap perlu diimbangi dengan pemahaman konseptual dan kemampuan membaca output. Oleh sebab itu, buku ini menempatkan ChemCompute bukan sebagai pengganti pemahaman teori, melainkan sebagai sarana untuk menghadirkan pengalaman komputasi yang lebih terbuka dan terjangkau.

Integrasi Python menjadi ciri penting lain dari buku ini. Python dipilih karena bahasa pemrograman ini relatif mudah dipelajari, memiliki ekosistem pustaka ilmiah yang luas, dan sangat sesuai untuk analisis data, visualisasi grafik, simulasi numerik, serta otomasi pengolahan output. Melalui Python, mahasiswa tidak hanya menerima hasil dari software, tetapi juga belajar merancang proses analisisnya sendiri. Mereka dapat membuat grafik fungsi gelombang dan rapat probabilitas, menghitung energi partikel dalam kotak, membaca koordinat XYZ, menghitung panjang dan sudut ikatan, membuat grafik energi terhadap basis set, menghitung gap HOMO-LUMO, menyusun spektrum IR simulasi, serta membuat diagram energi reaksi. Dengan cara ini, praktikum kimia kuantum menjadi lebih aktif, eksploratif, dan kreatif.

Buku ini disusun dalam dua belas bab. Bab pertama memberikan pendahuluan tentang hakikat praktikum kimia kuantum, hubungan kimia kuantum dengan kimia komputasi, serta pentingnya reprodusibilitas dan etika akademik. Bab kedua memperkenalkan ekosistem software kimia komputasi, termasuk ChemCompute, Avogadro, ORCA, GAMESS, Gaussian, dan perangkat pendukung visualisasi. Bab ketiga memberikan fondasi Python untuk praktikum kimia kuantum. Bab keempat hingga bab kesepuluh berisi tujuh mata praktikum utama, yaitu visualisasi fungsi gelombang dan partikel dalam kotak, optimasi geometri molekul sederhana, Hartree–Fock dan fungsi basis, DFT dan HOMO-LUMO, simetri molekul dan momen dipol, frekuensi vibrasi dan spektrum IR, serta energi reaksi, termodinamika, dan efek pelarut. Bab kesebelas membahas

analisis output, validasi data, dan otomasi laporan dengan Python. Bab kedua belas memuat proyek mini, evaluasi akhir, glosarium, dan lampiran kode Python.

Tujuh mata praktikum dalam buku ini dipilih secara bertahap dari konsep paling mendasar menuju aplikasi yang lebih kompleks. Praktikum pertama tentang partikel dalam kotak dimaksudkan untuk menanamkan pengertian bahwa energi pada tingkat mikroskopik bersifat terkuantisasi dan fungsi gelombang mengandung informasi probabilistik. Praktikum kedua tentang optimasi geometri mengantar mahasiswa memahami bahwa struktur molekul stabil berkaitan dengan titik minimum pada permukaan energi potensial. Praktikum ketiga tentang Hartree–Fock dan fungsi basis memperkenalkan mahasiswa pada pendekatan medan rata-rata, orbital molekul, serta hubungan antara akurasi dan biaya komputasi. Praktikum keempat tentang DFT dan HOMO-LUMO mengajak mahasiswa membaca reaktivitas molekul dari perspektif struktur elektronik. Praktikum kelima tentang simetri dan momen dipol menghubungkan bentuk molekul dengan sifat polaritas. Praktikum keenam tentang spektrum IR memperlihatkan hubungan antara struktur molekul dan getaran ikatan. Praktikum ketujuh tentang energi reaksi dan efek pelarut memperluas pemahaman mahasiswa terhadap kestabilan, spontanitas, dan termodinamika molekul.

Kebaruan utama buku ini terletak pada penyatuan tiga pendekatan dalam satu alur praktikum: **teori kimia kuantum, penggunaan software kimia komputasi, dan pemrograman Python mandiri**. Dengan pendekatan ini, mahasiswa tidak diposisikan sebagai operator pasif, tetapi sebagai calon ilmuwan yang mampu bertanya, merancang, menghitung, membandingkan, mengevaluasi, dan menyimpulkan. Setiap angka yang muncul dalam output komputasi harus dibaca secara kritis. Setiap grafik harus ditafsirkan secara kimiawi. Setiap metode harus dipahami batasannya. Setiap hasil harus dapat direproduksi. Setiap kesimpulan harus didukung oleh data, teori, dan argumentasi yang memadai.

Buku ini juga menekankan bahwa praktikum kimia kuantum tidak boleh berhenti pada keterampilan teknis. Penguasaan software dan kode program harus disertai dengan kesadaran epistemologis: bahwa model hanyalah representasi, metode memiliki asumsi, basis set memiliki keterbatasan, dan hasil komputasi harus divalidasi. Mahasiswa perlu memahami bahwa energi total dari dua metode berbeda tidak selalu dapat dibandingkan secara sembarangan, bahwa frekuensi imajiner memiliki makna struktural, bahwa charge dan multiplicity yang keliru dapat merusak seluruh hasil perhitungan, dan bahwa visualisasi orbital bukan sekadar gambar indah, tetapi representasi matematis dari distribusi fungsi gelombang atau kerapatan elektron.

Dalam konteks pendidikan kimia, buku ini diharapkan dapat memperkuat pembelajaran berbasis inkuiri dan proyek. Dosen dapat menggunakan buku ini sebagai modul praktikum satu semester, sedangkan mahasiswa dapat mengembangkannya menjadi proyek mini, tugas akhir, atau latihan riset awal. Setiap praktikum dapat dilakukan dalam laboratorium komputer, ruang kelas berbasis laptop, pembelajaran daring, maupun pembelajaran mandiri di rumah. Fleksibilitas ini penting karena sumber daya pendidikan di berbagai kampus tidak selalu sama. Ada kampus yang memiliki lisensi software komersial, ada yang lebih mengandalkan software gratis, ada yang menggunakan server lokal, dan ada pula yang lebih realistis menggunakan platform berbasis web. Buku ini berusaha menampung keragaman kondisi tersebut.

Sasaran utama buku ini adalah mahasiswa program studi Kimia, Pendidikan Kimia, Teknik Kimia, Farmasi, Biokimia, Material, dan bidang-bidang lain yang membutuhkan pengantar praktikum kimia kuantum dan kimia komputasi. Buku ini juga dapat digunakan oleh dosen, guru, laboran, peneliti pemula, serta siapa pun yang ingin mempelajari hubungan antara struktur molekul, energi, orbital, spektrum, dan reaktivitas secara komputasional. Bagi mahasiswa pemula, buku ini memberikan langkah praktis dan terarah. Bagi mahasiswa tingkat lanjut, buku ini dapat menjadi pijakan untuk merancang proyek komputasi yang lebih kompleks. Bagi pendidik, buku ini dapat menjadi inspirasi untuk menyusun pembelajaran kimia yang lebih visual, analitis, dan berbasis data.

Ruang lingkup buku ini sengaja dibatasi pada praktikum dasar dan menengah. Buku ini tidak dimaksudkan sebagai pengganti buku teks kimia kuantum teoretis yang mendalam, juga bukan manual lengkap untuk semua software kimia komputasi. Buku ini lebih tepat dipahami sebagai jembatan pembelajaran: dari teori menuju praktik, dari persamaan menuju visualisasi, dari output menuju interpretasi, dan dari penggunaan software menuju pemrograman ilmiah. Pembaca yang ingin memperdalam aspek matematis dapat merujuk pada buku-buku kimia kuantum lanjutan. Pembaca yang ingin memperluas aspek komputasi tingkat riset dapat melanjutkan ke metode pasca-Hartree–Fock, dinamika molekul, docking molekuler, QM/MM, material komputasi, atau machine learning dalam kimia.

Penyusunan buku ini juga dilandasi oleh keyakinan bahwa sains dan iman tidak harus dipertentangkan. Alam semesta adalah hamparan ayat-ayat kauniyah yang dapat dibaca melalui observasi, eksperimen, matematika, dan komputasi. Ketika mahasiswa mempelajari keteraturan orbital, kestabilan ikatan, simetri molekul, atau pola spektrum, sesungguhnya mereka sedang menyaksikan keteraturan ciptaan Allah Swt. Namun, refleksi keimanan dalam buku ini ditempatkan secara proporsional: bukan untuk menggantikan penalaran ilmiah, melainkan untuk menumbuhkan kerendahan hati, integritas, dan tanggung jawab moral dalam menggunakan ilmu pengetahuan.

Penulis berharap buku ini dapat membantu mahasiswa mengatasi rasa takut terhadap kimia kuantum. Konsep yang semula abstrak diharapkan menjadi lebih konkret melalui grafik, model, simulasi, tabel, struktur tiga dimensi, dan kode program. Mahasiswa yang semula hanya membaca rumus diharapkan mampu menjalankan perhitungan. Mahasiswa yang semula hanya melihat output diharapkan mampu menafsirkan maknanya. Mahasiswa yang semula hanya mengikuti petunjuk praktikum diharapkan mampu merancang variasi percobaannya sendiri secara kreatif dan bertanggung jawab.

Penulis menyadari bahwa buku ini tentu belum sempurna. Perkembangan software, bahasa pemrograman, pustaka Python, platform komputasi daring, dan metode kimia komputasi berlangsung sangat cepat. Oleh sebab itu, buku ini harus dipandang sebagai fondasi yang terbuka untuk diperbarui. Kritik, saran, koreksi, dan masukan dari dosen, mahasiswa, peneliti, serta praktisi kimia komputasi sangat diharapkan demi penyempurnaan edisi-edisi berikutnya. Terutama, masukan terkait akurasi kode Python, kompatibilitas software, kejelasan prosedur, dan ketepatan analisis kimia akan sangat berharga.

Akhirnya, semoga buku **Praktikum Kimia Kuantum** ini bermanfaat bagi pengembangan pembelajaran kimia di Indonesia. Semoga buku ini dapat menjadi sarana untuk menumbuhkan generasi pembelajar yang tidak hanya mampu memahami teori, tetapi juga terampil menggunakan teknologi, cakap membaca data, jujur dalam melaporkan hasil, kritis dalam menilai metode, dan kreatif dalam merancang eksperimen komputasi. Lebih jauh lagi, semoga buku ini menjadi bagian kecil dari ikhtiar memajukan pendidikan sains yang bermartabat, berkemajuan, dan membawa manfaat bagi kehidupan.

Semarang, Juni 2026

Penulis,

Kasmui

DAFTAR ISI

BAGIAN 1

PERSIAPAN PRAKTIKUM KIMIA KUANTUM

BAB 1

PENDAHULUAN PRAKTIKUM KIMIA KUANTUM

Pengantar Bab 1

Praktikum kimia kuantum merupakan pintu masuk penting untuk memahami bagaimana teori mekanika kuantum bekerja dalam menjelaskan struktur, energi, sifat, dan perubahan molekul. Jika kimia umum memperkenalkan atom, molekul, ikatan, struktur Lewis, bentuk molekul, dan energi reaksi melalui model konseptual, maka kimia kuantum berusaha menjawab pertanyaan yang lebih mendasar: mengapa elektron menempati tingkat energi tertentu, mengapa orbital memiliki bentuk tertentu, mengapa ikatan kimia terbentuk, mengapa suatu molekul lebih stabil daripada yang lain, dan bagaimana sifat molekul dapat diprediksi sebelum eksperimen laboratorium dilakukan.

Bab ini menjadi fondasi seluruh rangkaian praktikum. Sebelum mahasiswa menjalankan ChemCompute, Avogadro, ORCA, GAMESS, Gaussian, Psi4, PySCF, atau kode Python mandiri, mahasiswa perlu memahami terlebih dahulu hakikat praktikum kimia kuantum. Praktikum ini tidak boleh dipahami sekadar sebagai aktivitas menjalankan perangkat lunak, menyalin energi total, menyimpan gambar orbital, atau membuat grafik. Praktikum kimia kuantum adalah proses ilmiah untuk membangun hubungan antara **teori**, **model matematis**, **algoritma numerik**, **data komputasi**, dan **interpretasi kimia**.

Materi dasar bab ini berpijak pada cakupan kimia kuantum yang menempatkan Persamaan Schrödinger sebagai pusat pembahasan. Dalam sumber utama kimia kuantum yang digunakan, Bab 1 secara eksplisit memuat Persamaan Schrödinger, latar belakang historis mekanika kuantum, prinsip ketidakpastian, persamaan bergantung waktu, persamaan tak bergantung waktu, probabilitas, bilangan kompleks, satuan, dan kalkulus. Dasar tersebut sangat relevan dengan penyusunan buku praktikum karena setiap simulasi kimia komputasi, baik sederhana maupun lanjut, pada akhirnya merupakan upaya menyelesaikan atau menghampiri persoalan struktur dan energi sistem kuantum.

1.1 Hakikat Praktikum Kimia Kuantum

Praktikum kimia kuantum dapat didefinisikan sebagai kegiatan pembelajaran ilmiah yang menggunakan model mekanika kuantum, metode numerik, perangkat lunak kimia komputasi, dan/atau pemrograman ilmiah untuk menyelidiki struktur, energi, fungsi gelombang, orbital, sifat elektronik, spektrum, dan reaktivitas molekul. Definisi ini mengandung beberapa unsur penting. Pertama, praktikum kimia kuantum tetap merupakan **praktikum**, karena mahasiswa melakukan aktivitas eksploratif, mengumpulkan data, menganalisis hasil, membuat kesimpulan, dan menyusun laporan. Kedua, praktikum ini bersifat **kuantum**, karena objek yang dikaji adalah perilaku elektron dan inti atom dalam kerangka mekanika kuantum. Ketiga, praktikum ini bersifat **komputasional**, karena penyelesaian persoalan dilakukan dengan bantuan perhitungan numerik,

algoritma, visualisasi, dan perangkat lunak. Keempat, praktikum ini bersifat **interpretatif**, karena angka dan grafik yang dihasilkan tidak bermakna sebelum ditafsirkan berdasarkan konsep kimia.

Dengan demikian, hakikat praktikum kimia kuantum tidak identik dengan praktikum laboratorium basah. Pada laboratorium basah, mahasiswa biasanya mencampur pereaksi, mengukur suhu, menimbang massa, melakukan titrasi, merekam spektrum, atau mengamati perubahan warna. Pada praktikum kimia kuantum, mahasiswa membangun model molekul, memilih metode komputasi, menentukan basis set, menetapkan muatan dan multiplisitas, menjalankan optimasi geometri, menghitung energi, memvisualisasikan orbital, membaca frekuensi vibrasi, serta menafsirkan output. Akan tetapi, keduanya tetap memiliki struktur epistemik yang sama: ada pertanyaan ilmiah, ada prosedur, ada data, ada analisis, ada validasi, dan ada kesimpulan.

Perbedaan mendasar terletak pada jenis realitas yang diakses. Laboratorium basah berhadapan langsung dengan gejala makroskopik yang dapat diamati oleh pancaindra atau instrumen fisik. Praktikum kimia kuantum berhadapan dengan struktur submikroskopik yang tidak dapat dilihat langsung, tetapi dapat dimodelkan melalui teori dan dihitung melalui komputer. Elektron tidak dapat diamati sebagai partikel kecil yang bergerak pada lintasan seperti planet mengelilingi matahari. Orbital bukan lintasan klasik elektron. Fungsi gelombang juga bukan benda fisik yang dapat difoto secara langsung. Fungsi gelombang adalah objek matematis yang memuat informasi tentang keadaan sistem kuantum; makna fisiknya diperoleh melalui operasi matematis, terutama melalui kuadrat modulus fungsi gelombang yang berhubungan dengan rapat probabilitas. Sumber kimia kuantum yang digunakan menegaskan bahwa fungsi gelombang memuat informasi tentang sistem, sedangkan $|\Psi(x,t)|^2\,dx$ memberikan probabilitas menemukan partikel pada daerah tertentu.

Pemahaman ini penting karena salah satu kesalahan konseptual paling umum dalam pembelajaran kimia kuantum adalah memaknai konsep kuantum dengan intuisi klasik. Mahasiswa sering membayangkan elektron sebagai bola kecil yang mengorbit inti pada jalur tertentu. Model semacam itu berguna pada tahap awal, tetapi tidak memadai untuk menjelaskan spektrum atom, bentuk orbital, ikatan kovalen, delokalisasi elektron, aromaticity, transisi elektronik, dan reaktivitas molekul. Kimia kuantum mengajarkan bahwa keadaan elektron lebih tepat dinyatakan melalui fungsi gelombang atau kerapatan elektron, bukan melalui lintasan pasti. Karena itu, praktikum kimia kuantum harus melatih mahasiswa untuk berpindah dari cara berpikir klasik menuju cara berpikir probabilistik, matematis, dan modelistik.

Secara teoritis, fondasi praktikum kimia kuantum terletak pada Persamaan Schrödinger. Bentuk ringkas persamaan tak bergantung waktu dapat ditulis:

$$\hat{\mathbf{H}}\boldsymbol{\psi} = \mathbf{E}\boldsymbol{\psi}$$

Dalam persamaan ini, $\hat{\mathbf{H}}$ adalah operator Hamiltonian yang mewakili energi total sistem, $\boldsymbol{\psi}$ adalah fungsi gelombang keadaan stasioner, dan $\mathbf{E}$ adalah energi yang bersesuaian dengan keadaan tersebut. Persamaan ini tampak sederhana, tetapi konsekuensinya sangat luas. Hampir semua perhitungan struktur elektronik molekul dapat dipahami sebagai upaya mencari fungsi gelombang atau kerapatan elektron yang memberikan energi tertentu bagi sistem. Dalam sistem satu elektron sederhana seperti atom hidrogen, persamaan ini dapat diselesaikan secara analitik. Namun, untuk atom banyak elektron dan molekul, penyelesaian eksak menjadi sangat sulit karena adanya

interaksi elektron–elektron dan kompleksitas geometri inti. Karena itu, kimia kuantum modern berkembang melalui berbagai metode hampiran seperti Hartree–Fock, metode variasi, teori gangguan, Configuration Interaction, Møller–Plesset perturbation theory, Coupled Cluster, dan Density Functional Theory (Levine, 2014; Szabo & Ostlund, 1996; Atkins & Friedman, 2011).

Dari sudut pandang historis, praktikum kimia kuantum merupakan kelanjutan dari revolusi ilmiah awal abad ke-20. Planck memperkenalkan kuantisasi energi untuk menjelaskan radiasi benda hitam. Einstein mengembangkan gagasan kuantum cahaya untuk menjelaskan efek fotolistrik. Bohr menggunakan kuantisasi untuk menjelaskan spektrum atom hidrogen. De Broglie mengusulkan dualisme gelombang-partikel. Heisenberg, Born, dan Jordan mengembangkan mekanika matriks. Schrödinger kemudian merumuskan mekanika gelombang pada tahun 1926. Sumber kimia kuantum yang digunakan menegaskan bahwa mekanika kuantum pertama kali dirumuskan dalam bentuk mekanika matriks oleh Heisenberg, Born, dan Jordan pada tahun 1925, kemudian Schrödinger merumuskan mekanika gelombang pada tahun 1926 dan menunjukkan ekuivalensi keduanya. Perkembangan historis ini penting karena memperlihatkan bahwa kimia kuantum tidak muncul sebagai kumpulan rumus abstrak, tetapi sebagai jawaban atas kegagalan fisika klasik menjelaskan fenomena mikroskopik.

Dalam konteks pendidikan, praktikum kimia kuantum memiliki fungsi pedagogis yang sangat strategis. Banyak konsep kimia kuantum bersifat abstrak karena tidak langsung memiliki padanan pengalaman sehari-hari. Fungsi gelombang, operator, nilai eigen, orbital molekul, spin, determinan Slater, korelasi elektron, fungsi basis, dan permukaan energi potensial adalah konsep yang menuntut kemampuan representasi matematis dan visual sekaligus. Di sinilah praktikum komputasi menjadi penting. Melalui visualisasi orbital, mahasiswa dapat melihat pola distribusi elektron yang sebelumnya hanya berupa simbol ψ. Melalui grafik rapat probabilitas, mahasiswa dapat memahami mengapa probabilitas tidak selalu seragam. Melalui optimasi geometri, mahasiswa dapat melihat bahwa struktur molekul stabil berkaitan dengan minimum energi. Melalui spektrum IR komputasi, mahasiswa dapat menghubungkan getaran ikatan dengan frekuensi. Melalui analisis HOMO-LUMO, mahasiswa dapat menafsirkan kecenderungan donor-akseptor elektron dalam reaksi.

Penting untuk ditegaskan bahwa visualisasi dalam praktikum kimia kuantum bukan sekadar hiasan grafis. Visualisasi adalah jembatan representasi antara model matematis dan pemahaman kimia. Dalam teori representasi pembelajaran kimia, mahasiswa perlu menghubungkan tiga level pemahaman: makroskopik, submikroskopik, dan simbolik (Johnstone, 1991). Level makroskopik berkaitan dengan gejala yang dapat diamati, seperti warna, panas, endapan, atau spektrum. Level submikroskopik berkaitan dengan atom, molekul, elektron, orbital, dan interaksi antarpartikel. Level simbolik berkaitan dengan persamaan, rumus, notasi orbital, diagram energi, dan model matematis. Kimia komputasi dapat menjembatani ketiga level tersebut karena hasil komputasi dapat berbentuk angka, grafik, struktur 3D, spektrum, orbital, dan tabel energi. Dokumen rujukan software kimia komputasi juga menekankan bahwa integrasi komputasi ke dalam kurikulum bukan hanya penggantian alat bantu, melainkan pergeseran epistemologis dalam cara pengetahuan kimia diproduksi, divalidasi, dan ditransmisikan.

Namun, hakikat praktikum kimia kuantum tidak boleh direduksi menjadi “praktikum menekan tombol”. Inilah kritik utama terhadap pembelajaran kimia komputasi yang terlalu prosedural.

Mahasiswa dapat saja berhasil menjalankan job optimasi geometri, memperoleh output “normal termination”, dan menampilkan orbital HOMO-LUMO, tetapi tetap tidak memahami arti metode, basis set, energi total, frekuensi imajiner, atau momen dipol. Praktikum semacam itu hanya menghasilkan operator perangkat lunak, bukan pembelajar kimia kuantum. Karena itu, setiap praktikum harus mengharuskan mahasiswa menjawab pertanyaan konseptual: mengapa metode tersebut dipilih, apa asumsi yang digunakan, apa keterbatasannya, bagaimana memvalidasi hasil, dan bagaimana menghubungkan output dengan konsep kimia.

Kritik ini sangat penting dalam konteks penggunaan software modern. Perangkat lunak seperti Gaussian, ORCA, GAMESS, Psi4, PySCF, Avogadro, Gabedit, Molden, Jmol, Multiwfn, dan platform seperti ChemCompute sangat memudahkan akses perhitungan. Namun, kemudahan antarmuka dapat menimbulkan ilusi pemahaman. Seorang mahasiswa dapat menjalankan DFT/B3LYP/6-31G(d) tanpa memahami arti DFT, B3LYP, 6-31G(d), fungsi basis terpolarisasi, atau korelasi pertukaran. Oleh sebab itu, buku praktikum ini menempatkan software sebagai **instrumen epistemik**, bukan otoritas mutlak. Software membantu menghitung, tetapi mahasiswa tetap harus berpikir. Software menghasilkan angka, tetapi mahasiswa harus menafsirkan. Software menampilkan orbital, tetapi mahasiswa harus memahami maknanya.

Dalam kerangka tersebut, integrasi Python menjadi bagian penting dari hakikat praktikum kimia kuantum. Python bukan sekadar tambahan teknis, melainkan sarana untuk membangun pemahaman algoritmik. Ketika mahasiswa menulis kode untuk menghitung energi partikel dalam kotak, mereka belajar bahwa energi bergantung pada bilangan kuantum, massa partikel, dan panjang kotak. Ketika mereka membuat grafik ψ dan ψ^2, mereka belajar membedakan fungsi gelombang dan rapat probabilitas. Ketika mereka menulis kode untuk membaca koordinat XYZ dan menghitung panjang ikatan, mereka belajar bahwa geometri molekul dapat direpresentasikan sebagai kumpulan koordinat numerik. Ketika mereka menulis kode untuk membuat spektrum IR simulasi, mereka belajar bahwa spektrum dapat dibangun dari daftar frekuensi dan intensitas. Dengan kata lain, Python membantu mahasiswa melihat bahwa hasil kimia komputasi bukan “keajaiban software”, tetapi konsekuensi dari data, rumus, algoritma, dan model.

Praktikum kimia kuantum juga memiliki dimensi metodologis yang khas. Dalam laboratorium basah, pengulangan eksperimen sering dilakukan untuk menilai presisi dan mengurangi kesalahan acak. Dalam praktikum komputasi, reprodusibilitas bergantung pada pencatatan input dan parameter secara lengkap. Dua mahasiswa yang menggunakan molekul sama, tetapi berbeda charge, multiplicity, basis set, atau metode optimasi, dapat memperoleh hasil yang berbeda. Bahkan perbedaan struktur awal dapat memengaruhi konvergensi dan minimum energi yang diperoleh. Oleh karena itu, laporan praktikum kimia kuantum harus mencantumkan struktur awal, metode, basis set, charge, multiplicity, software, versi software jika memungkinkan, kriteria konvergensi, dan file output. Tanpa informasi tersebut, hasil praktikum sulit diverifikasi.

Dari sisi teori, praktikum kimia kuantum mengajarkan bahwa setiap model memiliki batas. Model partikel dalam kotak berguna untuk memahami kuantisasi energi dan sistem elektron π terkonjugasi, tetapi terlalu sederhana untuk menggambarkan molekul nyata secara lengkap. Hartree–Fock memberikan dasar orbital molekul dan medan rata-rata, tetapi tidak memasukkan korelasi elektron secara penuh. DFT efisien dan banyak digunakan, tetapi hasilnya bergantung pada pemilihan fungsional. Metode semiempiris cepat, tetapi mengandung parameterisasi empiris.

Basis set kecil hemat komputasi, tetapi kurang fleksibel menggambarkan distribusi elektron. Basis set besar lebih akurat, tetapi membutuhkan sumber daya lebih tinggi. Kesadaran terhadap batas model inilah yang membedakan praktikum ilmiah dari latihan teknis.

Contoh konkret dapat dilihat pada optimasi geometri air, H_2O. Secara teori VSEPR, molekul air memiliki bentuk bengkok karena pasangan elektron bebas pada atom oksigen menolak pasangan elektron ikatan. Dalam praktikum kimia kuantum, mahasiswa dapat membangun struktur H_2O, mengoptimasi geometri, lalu memperoleh panjang ikatan O–H dan sudut H–O–H. Dari sini mahasiswa dapat membandingkan hasil metode mekanika molekular, semiempiris, Hartree–Fock, dan DFT. Jika hasil sudut ikatan terlalu menyimpang dari nilai eksperimen atau literatur, mahasiswa harus bertanya: apakah struktur telah konvergen, apakah metode memadai, apakah basis set terlalu kecil, apakah charge dan multiplicity benar, dan apakah output dibaca dengan tepat. Aktivitas seperti ini melatih kemampuan ilmiah yang lebih tinggi daripada sekadar menghafal bahwa air berbentuk bengkok.

Contoh lain adalah analisis HOMO-LUMO butadiena. Dalam teori ikatan valensi sederhana, butadiena memiliki sistem ikatan rangkap terkonjugasi. Dalam praktikum kimia kuantum, mahasiswa dapat menghitung energi HOMO dan LUMO, memvisualisasikan bentuk orbital, serta membandingkan gap HOMO-LUMO butadiena dengan etilena. Mahasiswa akan melihat bahwa konjugasi memperkecil gap energi dan memengaruhi reaktivitas serta transisi elektronik. Pada tahap ini, konsep orbital tidak lagi menjadi gambar statis dalam buku teks, tetapi menjadi objek analisis yang dapat dikaitkan dengan stabilitas, delokalisasi, dan respons molekul terhadap radiasi.

Aplikasi praktikum kimia kuantum sangat luas. Dalam kimia organik, perhitungan orbital dapat membantu memprediksi pusat nukleofilik dan elektrofilik. Dalam kimia anorganik, perhitungan struktur elektronik membantu memahami spin, geometri kompleks, dan transisi d–d. Dalam kimia fisik, perhitungan frekuensi vibrasi membantu menafsirkan spektrum IR dan termodinamika. Dalam kimia material, perhitungan gap energi membantu memahami sifat semikonduktor dan absorpsi cahaya. Dalam farmasi, optimasi geometri dan analisis muatan dapat menjadi tahap awal pemodelan interaksi obat-reseptor. Dalam pendidikan kimia, visualisasi komputasi dapat membantu mahasiswa memahami konsep abstrak secara lebih konkret.

Implikasi akademik dari praktikum kimia kuantum adalah terbentuknya literasi komputasi dalam pendidikan kimia. Literasi ini mencakup kemampuan membaca input, memahami metode, menjalankan perhitungan, mengekstrak output, menganalisis data, membuat visualisasi, dan menyusun argumentasi berbasis hasil komputasi. Literasi semacam ini semakin penting karena riset kimia modern banyak menggunakan pendekatan komputasional, baik sebagai pelengkap eksperimen maupun sebagai alat prediksi awal. Dokumen software kimia komputasi yang menjadi rujukan buku ini menekankan bahwa mahasiswa dapat bertindak seperti peneliti: membangun struktur molekul, menetapkan basis set, menjalankan optimasi geometri, dan mengekstrak energi bebas Gibbs dari file output komputasi.

Implikasi praktisnya adalah perubahan cara mahasiswa belajar kimia. Mereka tidak lagi hanya menerima data dari buku, tetapi dapat menghasilkan data sendiri. Mereka tidak hanya melihat gambar orbital dari literatur, tetapi dapat membuat visualisasi orbital dari molekul yang mereka pilih. Mereka tidak hanya membaca tabel frekuensi IR, tetapi dapat mensimulasikan spektrum

sendiri. Mereka tidak hanya menghafal konsep polaritas, tetapi dapat menghitung momen dipol dan mengaitkannya dengan simetri molekul. Dengan cara ini, pembelajaran menjadi lebih aktif, berbasis inkuiri, dan berorientasi pemecahan masalah.

Namun, praktikum kimia kuantum juga menuntut sikap kritis terhadap hasil. Hasil komputasi bukan kebenaran absolut. Energi total yang muncul dalam output harus dipahami dalam konteks metode. Frekuensi vibrasi harmonik biasanya perlu faktor skala ketika dibandingkan dengan eksperimen. Struktur minimum harus divalidasi dengan ketiadaan frekuensi imajiner. Perbandingan energi reaksi harus dilakukan dengan metode dan basis set yang konsisten. Perhitungan fase gas tidak selalu mencerminkan sistem larutan. Model pelarut kontinu tidak selalu mampu menggambarkan ikatan hidrogen eksplisit. Semua ini menunjukkan bahwa praktikum kimia kuantum bukan hanya melatih keterampilan teknis, tetapi juga melatih epistemologi sains: bagaimana mengetahui, bagaimana memvalidasi, dan bagaimana membatasi klaim ilmiah.

Dalam perspektif pembelajaran, praktikum kimia kuantum idealnya dirancang sebagai siklus: **pertanyaan → model → input → perhitungan → output → analisis → validasi → refleksi**. Siklus ini membuat mahasiswa memahami bahwa komputasi bukan titik akhir, melainkan bagian dari proses ilmiah. Misalnya, pertanyaan "mengapa CO_2 nonpolar sedangkan H_2O polar?" tidak cukup dijawab dengan bentuk molekul secara verbal. Mahasiswa dapat membangun kedua struktur, mengoptimasi geometri, menentukan simetri, menghitung momen dipol, lalu menganalisis bahwa linearitas CO_2 menyebabkan vektor dipol saling meniadakan, sedangkan bentuk bengkok H_2O menghasilkan momen dipol bersih. Dengan demikian, jawaban konseptual diperkuat oleh data komputasi.

Hakikat praktikum kimia kuantum akhirnya dapat dirumuskan sebagai proses pembelajaran yang mengubah teori abstrak menjadi pengalaman analitis. Teori tetap menjadi fondasi, software menjadi instrumen, Python menjadi alat eksplorasi, data menjadi bahan interpretasi, dan laporan menjadi wujud pertanggungjawaban ilmiah. Praktikum ini tidak menggantikan pemahaman matematis kimia kuantum, tetapi memperkuatnya melalui pengalaman komputasional. Praktikum ini juga tidak menggantikan eksperimen laboratorium basah, tetapi melengkapinya dengan kemampuan memprediksi, menjelaskan, dan menafsirkan fenomena pada tingkat molekuler.

Dengan pemahaman tersebut, subbab berikutnya akan membahas hubungan antara kimia kuantum, kimia komputasi, dan programming ilmiah. Hubungan ini penting karena mahasiswa perlu mengetahui posisi masing-masing: kimia kuantum sebagai teori dasar, kimia komputasi sebagai metode penyelesaian dan pemodelan, serta programming Python sebagai sarana analisis, simulasi, dan inovasi praktikum.

1.2 Kimia Kuantum, Kimia Komputasi, dan Programming Ilmiah

Kimia kuantum, kimia komputasi, dan programming ilmiah merupakan tiga wilayah yang saling berkaitan, tetapi tidak identik. Ketiganya perlu dibedakan secara konseptual agar mahasiswa tidak keliru memahami posisi teori, perangkat lunak, dan kode program dalam praktikum. Kimia kuantum adalah fondasi teoretis yang menjelaskan perilaku elektron, atom, molekul, ikatan kimia,

spektrum, dan reaktivitas berdasarkan prinsip-prinsip mekanika kuantum. Kimia komputasi adalah penerapan metode numerik, algoritma, dan perangkat lunak untuk menyelesaikan persoalan kimia yang terlalu kompleks jika diselesaikan secara analitik. Programming ilmiah adalah keterampilan menyusun instruksi komputasional secara eksplisit melalui bahasa pemrograman, misalnya Python, untuk menghitung, memvisualisasikan, mengolah data, mengotomatisasi analisis, dan menguji kembali hasil perhitungan.

Dalam konteks praktikum ini, ketiga bidang tersebut membentuk satu ekosistem pembelajaran. Kimia kuantum menyediakan pertanyaan dasar: bagaimana energi sistem molekul dihitung, bagaimana orbital terbentuk, mengapa elektron memiliki tingkat energi tertentu, dan bagaimana sifat molekul muncul dari struktur elektroniknya. Kimia komputasi menyediakan metode dan software untuk menjawab pertanyaan tersebut melalui pendekatan numerik. Programming ilmiah menyediakan kemampuan agar mahasiswa tidak hanya menjadi pengguna software, tetapi juga mampu mengolah data, membuat simulasi, memeriksa hasil, dan membangun pemahaman algoritmik. Tanpa kimia kuantum, praktikum komputasi kehilangan dasar teorinya. Tanpa kimia komputasi, kimia kuantum tetap terlalu abstrak bagi banyak mahasiswa. Tanpa programming ilmiah, mahasiswa berisiko menjadi operator pasif yang hanya menerima output tanpa memahami proses analisisnya.

Secara historis, kimia kuantum lahir dari krisis fisika klasik pada awal abad ke-20. Fisika klasik berhasil menjelaskan banyak fenomena makroskopik, tetapi gagal menjelaskan radiasi benda hitam, efek fotolistrik, kestabilan atom, dan spektrum garis atom. Perkembangan mekanika kuantum dimulai dari gagasan kuantisasi energi Planck, penjelasan Einstein tentang efek fotolistrik, model atom Bohr, hipotesis gelombang materi de Broglie, mekanika matriks Heisenberg, dan mekanika gelombang Schrödinger. Dalam sumber utama kimia kuantum yang digunakan, perkembangan mekanika kuantum dijelaskan bermula dari kajian Planck pada tahun 1900, sedangkan pembahasan historis juga dikaitkan dengan sifat gelombang cahaya, difraksi, interferensi, persamaan Maxwell, dan perkembangan konsep radiasi elektromagnetik.

Lompatan penting dari fisika kuantum menuju kimia kuantum terjadi ketika prinsip mekanika kuantum diterapkan pada atom dan molekul. Persamaan Schrödinger memungkinkan energi dan fungsi gelombang suatu sistem dihitung, setidaknya untuk sistem sederhana. Untuk atom hidrogen, penyelesaian persamaan Schrödinger menghasilkan orbital atom dan bilangan kuantum yang menjelaskan struktur spektrum atom. Namun, ketika sistem memiliki lebih dari satu elektron, persoalannya menjadi jauh lebih kompleks karena setiap elektron berinteraksi dengan inti dan dengan elektron lain. Di sinilah kimia kuantum mulai berhadapan dengan keterbatasan analitik. Persamaan dasarnya dapat ditulis, tetapi penyelesaian eksaknya hampir tidak mungkin diperoleh untuk sistem molekul nyata. Karena itu, kimia kuantum berkembang bersama metode hampiran seperti metode variasi, teori gangguan, Hartree–Fock, metode pasca-Hartree–Fock, dan teori fungsional kerapatan.

Kimia komputasi muncul sebagai jawaban praktis terhadap kompleksitas tersebut. Ia tidak mengganti teori kimia kuantum, tetapi menyediakan jalan untuk menerapkan teori tersebut pada sistem nyata. Dokumen rujukan software kimia komputasi menegaskan bahwa akar kimia komputasi tertanam kuat pada dekade 1920-an ketika Schrödinger dan Dirac merumuskan dasar-dasar mekanika kuantum, tetapi disrupsi nyata baru terjadi setelah hadirnya komputer digital

pasca-Perang Dunia II; tonggak penting berikutnya terjadi ketika Gaussian 70 memperluas akses perhitungan ab initio bagi kimiawan. Dengan demikian, kimia komputasi lahir dari pertemuan antara teori kuantum, matematika numerik, algoritma, dan perkembangan perangkat keras komputer.

Definisi kimia komputasi dapat dirumuskan sebagai cabang kimia yang menggunakan teori, model matematis, dan komputasi numerik untuk mempelajari struktur, energi, sifat, dan reaktivitas sistem kimia. Definisi ini mencakup beberapa pendekatan. Pertama, mekanika molekular, yang memperlakukan atom sebagai partikel klasik yang terhubung oleh medan gaya. Kedua, metode semiempiris, yang menyederhanakan perhitungan kuantum dengan parameter empiris. Ketiga, metode ab initio, yang berupaya menghitung sistem molekul dari prinsip pertama tanpa parameter empiris. Keempat, Density Functional Theory atau DFT, yang menggunakan kerapatan elektron sebagai variabel utama. Kelima, dinamika molekul, yang mempelajari perubahan konfigurasi sistem terhadap waktu. Dalam buku praktikum ini, fokus utama berada pada metode struktur elektronik dasar dan menengah yang relevan untuk pembelajaran kimia kuantum.

Perbedaan antara kimia kuantum dan kimia komputasi dapat dianalogikan dengan hubungan antara teori termodinamika dan perangkat lunak simulasi termodinamika. Teori menyediakan hukum, persamaan, konsep, dan batas validitas. Perangkat lunak menyediakan alat untuk menghitung sistem nyata. Dalam kimia kuantum, konsep seperti fungsi gelombang, operator Hamiltonian, nilai eigen, orbital, spin, simetri, dan energi elektronik merupakan perangkat teoritis. Dalam kimia komputasi, konsep tersebut diterjemahkan menjadi input, metode, basis set, algoritma SCF, integral molekuler, optimasi geometri, analisis frekuensi, dan file output. Mahasiswa harus memahami bahwa ketika mereka menjalankan perhitungan HF/STO-3G atau B3LYP/6-31G(d), mereka bukan sekadar memilih menu software, tetapi sedang memilih model matematis tertentu dengan asumsi dan keterbatasan tertentu.

Metode ab initio menjadi contoh penting hubungan antara kimia kuantum dan kimia komputasi. Secara etimologis, ab initio berarti “dari awal” atau “dari prinsip pertama”. Dalam kimia komputasi, istilah ini merujuk pada metode yang menggunakan konstanta fisika fundamental dan prinsip mekanika kuantum tanpa penyesuaian langsung terhadap data eksperimen. Dokumen rujukan software kimia komputasi menjelaskan bahwa pendekatan ab initio berlandaskan penyelesaian Persamaan Schrödinger dengan konstanta fundamental, sedangkan Hartree–Fock menjadi pondasi awal yang mengasumsikan setiap elektron bergerak dalam medan potensial rata-rata yang dibentuk oleh inti dan elektron lain. Bagi mahasiswa, penjelasan ini penting karena Hartree–Fock bukan sekadar nama metode dalam menu software, melainkan sebuah model fisika-matematis tentang bagaimana elektron diperlakukan.

Akan tetapi, Hartree–Fock memiliki keterbatasan. Ia memperlakukan interaksi elektron melalui medan rata-rata, sehingga korelasi elektron tidak sepenuhnya tercakup. Akibatnya, energi yang dihitung dapat menyimpang dari nilai yang lebih akurat, terutama untuk sistem yang memerlukan deskripsi korelasi kuat. DFT kemudian berkembang sebagai pendekatan yang lebih efisien untuk banyak sistem kimia, karena menggunakan kerapatan elektron sebagai pusat perhitungan dan memasukkan efek pertukaran-korelasi melalui fungsional tertentu. Namun, DFT juga tidak bebas masalah. Hasilnya bergantung pada fungsional yang dipilih, dan tidak semua fungsional cocok untuk semua jenis sistem. Karena itu, pemilihan metode dalam kimia komputasi harus selalu

dikaitkan dengan tujuan praktikum, ukuran sistem, jenis sifat yang dihitung, sumber daya komputer, dan tingkat akurasi yang dibutuhkan.

Di sinilah programming ilmiah menjadi penting. Software kimia komputasi seperti GAMESS, ORCA, Gaussian, Psi4, PySCF, dan platform berbasis web seperti ChemCompute memudahkan mahasiswa menjalankan perhitungan. Namun, output yang dihasilkan sering sangat panjang, teknis, dan memerlukan pemrosesan lanjutan. Mahasiswa perlu mengekstrak energi total, panjang ikatan, sudut ikatan, frekuensi vibrasi, intensitas IR, momen dipol, energi HOMO-LUMO, dan parameter termodinamika. Jika semua analisis dilakukan secara manual, prosesnya lambat, rawan salah salin, dan sulit direproduksi. Dengan Python, mahasiswa dapat membuat skrip untuk membaca file, mengekstrak angka penting, menyusun tabel, membuat grafik, menyimpan data, dan membandingkan hasil antar metode.

Programming ilmiah berbeda dari programming aplikasi biasa. Tujuannya bukan terutama membuat aplikasi komersial atau antarmuka pengguna yang rumit, melainkan membuat proses ilmiah menjadi eksplisit, transparan, dan dapat diulang. Dalam praktikum kimia kuantum, Python berfungsi sebagai laboratorium analisis. Mahasiswa dapat menulis kode sederhana untuk menghitung energi partikel dalam kotak, membuat grafik fungsi gelombang, menghitung panjang ikatan dari koordinat XYZ, membuat diagram energi orbital, memplot spektrum IR simulasi, menghitung perubahan energi reaksi, atau membandingkan hasil perhitungan fase gas dan pelarut. Setiap baris kode menjadi bagian dari argumentasi ilmiah karena menunjukkan bagaimana data diolah menjadi kesimpulan.

Hubungan antara software dan Python dapat dipahami melalui pembagian kerja. Software kimia komputasi menjalankan perhitungan struktur elektronik yang kompleks. Python membantu merancang input sederhana, memvalidasi data, mengolah output, membuat grafik, dan membangun simulasi konsep. ChemCompute, misalnya, menyediakan akses komputasi berbasis web untuk pembelajaran dan riset sarjana, sehingga mahasiswa dapat menjalankan perangkat kimia komputasi tanpa harus memasang dan memelihara software atau hardware sendiri. (ChemCompute) Artikel Perri dan Weber menjelaskan bahwa antarmuka web semacam ini dikembangkan untuk memfasilitasi penggunaan GAMESS dan software komputasi kimia bebas lainnya oleh mahasiswa, terutama karena bagian perhitungan sering menjadi bagian paling sulit dan memakan waktu dalam praktik komputasi. (American Chemical Society) Dengan demikian, ChemCompute dapat ditempatkan sebagai gerbang akses, sedangkan Python ditempatkan sebagai alat pemahaman, analisis, dan inovasi.

Kedudukan ChemCompute dalam praktikum perlu dijelaskan secara proporsional. Platform ini bermanfaat untuk memperluas akses, terutama jika mahasiswa tidak memiliki komputer kuat, tidak dapat menginstal software, atau kampus belum memiliki server komputasi. Halaman bantuan ChemCompute juga mencantumkan rujukan sitasi untuk penggunaan ChemCompute Science Gateway, termasuk rujukan Perri dan Weber tahun 2014 tentang antarmuka pengiriman job berbasis web untuk GAMESS. (ChemCompute) Namun, kemudahan akses tidak boleh membuat mahasiswa mengabaikan prinsip dasar perhitungan. Mahasiswa tetap harus memahami charge, multiplicity, metode, basis set, jenis job, geometri awal, dan interpretasi output. ChemCompute tidak menggantikan kimia kuantum; ia hanya mempermudah akses menuju perhitungan kimia kuantum.

Dalam praktiknya, hubungan antara kimia kuantum, kimia komputasi, dan Python dapat dijelaskan melalui satu contoh sederhana: molekul air. Dari sisi kimia kuantum, mahasiswa bertanya mengapa air memiliki bentuk bengkok, mengapa memiliki momen dipol, dan bagaimana energi struktur stabil ditentukan. Dari sisi kimia komputasi, mahasiswa membangun geometri H_2O, menentukan charge = 0 dan multiplicity = 1, memilih metode misalnya HF/STO-3G atau B3LYP/6-31G(d), lalu menjalankan optimasi geometri. Dari sisi Python, mahasiswa membaca koordinat akhir, menghitung dua panjang ikatan O–H, menghitung sudut H–O–H, membuat tabel, membandingkan hasil dengan literatur, dan memplot vektor geometri sederhana. Satu objek molekul yang sama dipahami melalui tiga lapisan: teori, komputasi, dan analisis programatik.

Contoh kedua dapat dilihat pada model partikel dalam kotak. Dari sisi kimia kuantum, model ini mengajarkan kuantisasi energi dan fungsi gelombang. Dari sisi komputasi, mahasiswa dapat menyusun simulasi numerik untuk beberapa bilangan kuantum. Dari sisi Python, mahasiswa menghitung $E_n = n^2h^2 / 8mL^2$, membuat grafik $\psi_n(x)$, membuat grafik $\psi_n^2(x)$, menghitung energi dalam joule dan elektronvolt, lalu melihat hubungan antara panjang kotak dan energi. Model ini memang sederhana, tetapi sangat kuat secara pedagogis karena mahasiswa melihat bahwa perubahan parameter matematis menghasilkan perubahan pola fisik. Ini mengajarkan inti dari pemodelan ilmiah: model bukan sekadar rumus, melainkan alat untuk menguji hubungan sebab-akibat dalam sistem tertentu.

Contoh ketiga adalah analisis HOMO-LUMO. Dari sisi kimia kuantum, HOMO dan LUMO berkaitan dengan orbital molekul terisi tertinggi dan orbital molekul kosong terendah. Dari sisi kimia komputasi, energi HOMO dan LUMO diperoleh dari perhitungan struktur elektronik. Dari sisi Python, mahasiswa memasukkan energi HOMO dan LUMO beberapa molekul, menghitung gap, membuat diagram energi, menghitung parameter reaktivitas global sederhana seperti hardness dan softness, lalu membandingkan pengaruh konjugasi atau substituen. Dengan demikian, mahasiswa tidak hanya melihat gambar orbital, tetapi juga belajar menafsirkan angka menjadi argumen kimia.

Sintesis literatur pendidikan kimia menunjukkan bahwa integrasi komputasi dalam pembelajaran dapat mengubah cara mahasiswa memahami konsep submikroskopik. Dokumen rujukan software kimia komputasi menyebut bahwa integrasi komputasi ke dalam kurikulum bukan sekadar substitusi alat bantu, melainkan pergeseran epistemologis dalam cara pengetahuan kimia diproduksi, divalidasi, dan ditransmisikan. Pernyataan ini penting karena praktikum kimia kuantum berbasis komputasi tidak boleh dipahami hanya sebagai versi digital dari praktikum biasa. Ia mengubah posisi mahasiswa dari penerima data menjadi produsen data. Mahasiswa tidak hanya membaca tabel energi dari buku, tetapi menghasilkan nilai energi sendiri. Mahasiswa tidak hanya menerima gambar orbital, tetapi memvisualisasikan orbital molekul yang dihitung. Mahasiswa tidak hanya mempelajari teori spektrum, tetapi membangun spektrum simulasi dari data frekuensi dan intensitas.

Namun, integrasi komputasi juga memiliki risiko pedagogis. Risiko pertama adalah ilusi presisi. Output komputer sering memiliki banyak angka desimal sehingga tampak sangat akurat. Padahal, akurasi hasil bergantung pada metode, basis set, model, kualitas geometri awal, dan parameter perhitungan. Risiko kedua adalah ketergantungan pada antarmuka grafis. Antarmuka yang mudah dapat membuat mahasiswa kurang memahami file input dan output. Risiko ketiga adalah

keterputusan antara angka dan makna kimia. Mahasiswa dapat memperoleh energi total, tetapi tidak mengetahui apakah nilai tersebut dapat dibandingkan langsung antar molekul atau hanya bermakna dalam konteks tertentu. Risiko keempat adalah kurangnya validasi. Mahasiswa mungkin menganggap perhitungan selesai berarti hasil benar, padahal perlu diperiksa konvergensi SCF, konvergensi optimasi, frekuensi imajiner, charge, multiplicity, dan konsistensi satuan.

Karena itu, programming ilmiah perlu ditempatkan sebagai mekanisme penguatan berpikir kritis. Ketika mahasiswa menulis kode sendiri, mereka dipaksa menyatakan asumsi secara eksplisit. Jika ingin menghitung jarak ikatan, mereka harus memahami rumus jarak Euclidean. Jika ingin menghitung sudut ikatan, mereka harus memahami vektor dan dot product. Jika ingin membuat spektrum IR simulasi, mereka harus memahami hubungan antara frekuensi, intensitas, pelebaran puncak, dan grafik absorbansi. Jika ingin menghitung ΔG reaksi, mereka harus menyusun energi reaktan dan produk secara konsisten. Dengan kata lain, Python bukan hanya alat hitung, tetapi alat pendidikan konseptual.

Dalam riset modern, programming ilmiah juga berkaitan erat dengan reprodusibilitas. Suatu hasil ilmiah tidak cukup hanya dilaporkan; ia harus dapat diperiksa dan diulang. Dalam kimia komputasi, reprodusibilitas menuntut keterbukaan input, metode, basis set, struktur, parameter, dan alur analisis. Kode Python membantu mencatat alur analisis tersebut. Jika mahasiswa membuat grafik energi secara manual, prosesnya sulit dilacak. Jika mahasiswa membuat grafik melalui kode, dosen dapat memeriksa bagaimana data dimasukkan, bagaimana satuan dikonversi, bagaimana grafik dibuat, dan bagaimana kesimpulan diambil. Inilah nilai penting programming ilmiah dalam pendidikan kimia kuantum.

Implikasi akademiknya sangat besar. Mahasiswa yang menguasai hubungan kimia kuantum, kimia komputasi, dan programming ilmiah akan memiliki kemampuan lintas disiplin. Mereka memahami teori kimia, mampu menggunakan software, mampu membaca output, mampu mengolah data, dan mampu menyusun argumen berbasis grafik dan angka. Keterampilan ini relevan untuk riset struktur molekul, spektroskopi, katalisis, kimia material, kimia obat, kimia lingkungan, dan pendidikan kimia. Sumber utama kimia kuantum yang digunakan juga menegaskan luasnya aplikasi mekanika kuantum: kimiawan fisika menggunakannya untuk termodinamika gas dan spektrum molekul, kimiawan organik untuk kestabilan relatif dan mekanisme reaksi, kimiawan analitik untuk spektroskopi, kimiawan anorganik untuk teori medan ligan, dan biokimiawan untuk konformasi serta interaksi biomolekul.

Implikasi praktisnya juga jelas. Dalam kelas praktikum, dosen dapat merancang pembelajaran bertahap. Pertama, mahasiswa diberi konsep kimia kuantum dasar. Kedua, mahasiswa menjalankan perhitungan sederhana melalui ChemCompute atau software lokal. Ketiga, mahasiswa mengambil output dan mengolahnya dengan Python. Keempat, mahasiswa membandingkan hasil dengan teori atau literatur. Kelima, mahasiswa menyusun laporan yang menjelaskan hubungan antara model, metode, hasil, dan keterbatasan. Alur ini membuat praktikum lebih bermakna karena mahasiswa tidak hanya mengikuti petunjuk, tetapi membangun pemahaman melalui siklus ilmiah.

Dalam konteks buku ini, setiap mata praktikum akan memuat dua lapisan kerja. Lapisan pertama adalah **praktikum software**, yaitu penggunaan ChemCompute, Avogadro, ORCA, GAMESS,

Gaussian, Psi4, PySCF, atau software lain untuk menghasilkan data kimia komputasi. Lapisan kedua adalah **praktikum Python**, yaitu penggunaan kode program untuk menghitung, mengekstrak, memvisualisasikan, dan menafsirkan data. Pembagian ini penting agar mahasiswa memahami bahwa hasil software bukan akhir pembelajaran. Output software adalah bahan mentah ilmiah yang harus dibaca, disaring, dianalisis, divalidasi, dan dikomunikasikan.

Sebagai contoh, dalam praktikum optimasi geometri, software menghasilkan koordinat akhir dan energi minimum. Python kemudian digunakan untuk menghitung panjang ikatan dan sudut ikatan dari koordinat tersebut. Dalam praktikum Hartree–Fock dan basis set, software menghasilkan energi untuk beberapa basis set. Python kemudian digunakan untuk membuat grafik konvergensi energi terhadap ukuran basis set. Dalam praktikum DFT dan HOMO-LUMO, software menghasilkan energi orbital. Python digunakan untuk menghitung gap dan membuat diagram energi. Dalam praktikum IR, software menghasilkan frekuensi dan intensitas. Python digunakan untuk membuat spektrum simulasi. Dalam praktikum termodinamika, software menghasilkan energi elektronik dan koreksi termal. Python digunakan untuk menghitung ΔE, ΔH, ΔG, dan diagram energi reaksi.

Pendekatan ini juga mendorong kreativitas mahasiswa. Setelah memahami alur dasar, mahasiswa dapat memodifikasi praktikum. Mereka dapat mengganti molekul, membandingkan metode, menguji pengaruh substituen, mempelajari konformasi, membandingkan fase gas dan pelarut, atau membuat visualisasi baru. Programming ilmiah memberi ruang inovasi karena mahasiswa dapat memperluas kode sesuai pertanyaan riset kecil yang mereka ajukan. Inilah salah satu tujuan utama buku praktikum ini: membentuk mahasiswa yang tidak sekadar mengikuti instruksi, tetapi mampu merancang eksplorasi komputasi secara mandiri, terarah, dan bertanggung jawab.

Akan tetapi, kreativitas harus tetap berada dalam kerangka validitas ilmiah. Mahasiswa tidak boleh sembarangan membandingkan energi dari metode berbeda tanpa alasan. Mereka tidak boleh mengubah basis set di tengah perbandingan tanpa mencatatnya. Mereka tidak boleh menghapus data yang tidak sesuai harapan. Mereka tidak boleh menyimpulkan bahwa suatu molekul “paling stabil” hanya dari satu angka tanpa mempertimbangkan struktur, metode, fase, dan koreksi energi. Programming ilmiah harus mendukung kejujuran akademik, bukan sekadar menghasilkan grafik yang menarik. Kode yang baik adalah kode yang transparan, dapat diperiksa, dan menghasilkan data yang dapat dipertanggungjawabkan.

Hubungan antara kimia kuantum, kimia komputasi, dan programming ilmiah akhirnya dapat dipahami sebagai hubungan antara **prinsip**, **metode**, dan **eksekusi analitis**. Kimia kuantum menjawab “mengapa” dan “apa dasar teorinya”. Kimia komputasi menjawab “bagaimana sistem dihitung dengan metode tertentu”. Programming ilmiah menjawab “bagaimana data dianalisis, divisualisasikan, dan direproduksi”. Ketiganya harus hadir bersama dalam praktikum modern. Jika salah satunya hilang, pembelajaran menjadi tidak utuh. Teori tanpa komputasi menjadi terlalu abstrak. Komputasi tanpa teori menjadi mekanistik. Programming tanpa pemahaman kimia menjadi sekadar manipulasi angka.

Dengan pemahaman ini, mahasiswa siap memasuki subbab berikutnya yang membahas Persamaan Schrödinger sebagai fondasi praktikum. Subbab tersebut akan menunjukkan bahwa seluruh kegiatan praktikum kimia kuantum, mulai dari model partikel dalam kotak hingga perhitungan

orbital molekul, pada dasarnya berakar pada gagasan bahwa keadaan sistem kuantum dapat dijelaskan melalui fungsi gelombang atau kerapatan elektron, dan bahwa energi sistem diperoleh melalui operator Hamiltonian.

1.3 Persamaan Schrödinger sebagai Fondasi Praktikum

Persamaan Schrödinger adalah fondasi konseptual utama dalam praktikum kimia kuantum karena hampir seluruh kegiatan komputasi struktur elektronik molekul dapat dipahami sebagai usaha untuk mencari energi, fungsi gelombang, kerapatan elektron, atau sifat turunan dari suatu sistem atomik dan molekuler. Dalam praktikum kimia kuantum, mahasiswa memang sering berhadapan dengan menu software, file input, file output, nilai energi total, orbital HOMO-LUMO, frekuensi vibrasi, momen dipol, dan geometri hasil optimasi. Akan tetapi, semua data tersebut tidak berdiri sendiri. Di baliknya terdapat prinsip mekanika kuantum yang menyatakan bahwa keadaan sistem mikroskopik dijelaskan oleh fungsi gelombang atau representasi kuantum yang sepadan, sedangkan energi dan sifat fisiknya diperoleh melalui operator tertentu.

Dalam sumber utama kimia kuantum yang menjadi dasar buku ini, Bab 1 secara eksplisit memuat pembahasan Persamaan Schrödinger, latar belakang historis mekanika kuantum, prinsip ketidakpastian, Persamaan Schrödinger bergantung waktu, Persamaan Schrödinger tak bergantung waktu, probabilitas, bilangan kompleks, satuan, dan kalkulus. Ini menunjukkan bahwa Persamaan Schrödinger bukan hanya rumus pembuka dalam kimia kuantum, melainkan pusat gravitasi teoretis yang menghubungkan fisika gelombang, probabilitas, struktur atom, orbital, energi, dan sifat molekul. Dalam buku praktikum ini, Persamaan Schrödinger perlu dipahami bukan hanya sebagai persamaan yang ditulis di awal kuliah, tetapi sebagai landasan yang menjelaskan mengapa software kimia komputasi dapat menghitung struktur dan sifat molekul.

Secara umum, Persamaan Schrödinger memiliki dua bentuk utama, yaitu bentuk bergantung waktu dan bentuk tak bergantung waktu. Bentuk bergantung waktu dapat ditulis secara ringkas sebagai:

$$i\hbar\, \partial\Psi/\partial t = \hat{H}\Psi$$

Dalam persamaan ini, i adalah bilangan imajiner, $\hbar$ adalah konstanta Planck tereduksi, Ψ adalah fungsi gelombang yang bergantung pada posisi dan waktu, t adalah waktu, dan $\hat{H}$ adalah operator Hamiltonian. Bentuk ini menjelaskan bagaimana keadaan kuantum suatu sistem berubah terhadap waktu. Jika suatu sistem mengalami interaksi dengan radiasi, medan luar, atau gangguan yang berubah terhadap waktu, maka bentuk bergantung waktu menjadi sangat penting. Dalam spektroskopi, fotokimia, dan dinamika keadaan tereksitasi, bentuk ini memberikan dasar konseptual untuk memahami transisi dari satu keadaan kuantum ke keadaan lain.

Bentuk tak bergantung waktu dapat ditulis sebagai:

$$\hat{H}\psi = E\psi$$

Bentuk inilah yang paling sering menjadi dasar perhitungan struktur elektronik keadaan stasioner dalam praktikum kimia kuantum dasar. Dalam persamaan ini, ψ adalah fungsi gelombang keadaan stasioner, $\hat{H}$ adalah operator Hamiltonian, dan E adalah energi sistem. Secara matematis,

persamaan ini adalah persamaan nilai eigen. Fungsi gelombang **ψ** bertindak sebagai fungsi eigen, sedangkan energi **E** bertindak sebagai nilai eigen. Dengan kata lain, ketika operator Hamiltonian bekerja pada fungsi gelombang yang tepat, hasilnya adalah fungsi gelombang yang sama dikalikan dengan nilai energi tertentu. Inilah dasar munculnya tingkat-tingkat energi diskret dalam sistem kuantum.

Makna tersebut sangat penting bagi praktikum. Ketika mahasiswa menjalankan perhitungan energi molekul, software pada dasarnya berusaha mencari keadaan elektronik yang memenuhi bentuk hampiran dari Persamaan Schrödinger. Ketika mahasiswa memperoleh energi total dalam file output, nilai itu bukan angka sembarangan; ia adalah hasil penyelesaian numerik atau hampiran dari persoalan nilai eigen. Ketika mahasiswa melihat orbital molekul, orbital tersebut berkaitan dengan fungsi matematika yang digunakan untuk menggambarkan distribusi elektron dalam sistem molekul. Ketika mahasiswa melihat perbedaan energi antara HOMO dan LUMO, mereka sedang melihat selisih tingkat energi elektronik yang muncul dari penyelesaian model kuantum tertentu.

Unsur pertama yang perlu dipahami adalah **operator Hamiltonian**. Dalam mekanika kuantum, besaran fisik seperti energi, momentum, posisi, dan momentum sudut dinyatakan melalui operator. Operator Hamiltonian adalah operator energi total sistem. Untuk sistem sederhana satu partikel dalam satu dimensi, Hamiltonian biasanya terdiri atas dua bagian utama: energi kinetik dan energi potensial. Secara konseptual dapat ditulis:

$$\hat{\mathbf{H}} = \hat{\mathbf{T}} + \hat{\mathbf{V}}$$

Di sini, $\hat{\mathbf{T}}$ adalah operator energi kinetik dan $\hat{\mathbf{V}}$ adalah operator energi potensial. Pada sistem molekul, Hamiltonian menjadi jauh lebih kompleks karena harus mencakup energi kinetik elektron, energi kinetik inti, tarikan elektron–inti, tolakan elektron–elektron, dan tolakan inti–inti. Kompleksitas inilah yang membuat penyelesaian eksak Persamaan Schrödinger untuk molekul nyata hampir tidak mungkin dilakukan secara langsung. Karena itu, semua metode kimia komputasi modern merupakan strategi untuk menyederhanakan, menghampiri, atau memecahkan Hamiltonian molekuler tersebut secara numerik.

Secara konseptual, Hamiltonian dapat dipahami sebagai "mesin matematis" yang memuat seluruh informasi energi dalam sistem. Jika Hamiltonian suatu sistem diketahui dan dapat diselesaikan, maka energi dan fungsi gelombang sistem dapat diperoleh. Namun, dalam praktik kimia, sistem molekul melibatkan banyak elektron dan banyak inti. Setiap elektron berinteraksi dengan inti dan elektron lainnya. Setiap inti juga saling tolak-menolak dengan inti lain. Oleh karena itu, Hamiltonian molekul bukan hanya panjang secara matematis, tetapi juga sulit diselesaikan karena adanya korelasi gerak antarpartikel. Dokumen software kimia komputasi yang menjadi rujukan buku ini menegaskan bahwa secara teoretik sistem yang dapat diselesaikan secara analitik dengan Persamaan Schrödinger hanyalah sistem sangat sederhana seperti atom hidrogen; sistem yang lebih kompleks membutuhkan asumsi dan hampiran, termasuk Aproksimasi Born–Oppenheimer.

Aproksimasi Born–Oppenheimer memiliki peran penting dalam kimia kuantum molekuler. Inti atom jauh lebih berat daripada elektron, sehingga gerak inti jauh lebih lambat dibanding gerak elektron. Berdasarkan perbedaan massa dan skala gerak ini, gerakan elektron dan inti dapat dipisahkan secara hampiran. Dalam praktikum kimia komputasi, pemisahan ini memungkinkan

mahasiswa menghitung energi elektronik pada konfigurasi inti tertentu. Dengan kata lain, untuk setiap susunan geometri inti, software dapat menghitung energi elektroniknya. Ketika geometri inti diubah sedikit demi sedikit, energi juga berubah. Kumpulan energi untuk berbagai geometri membentuk permukaan energi potensial. Optimasi geometri kemudian dapat dipahami sebagai pencarian titik minimum pada permukaan energi potensial tersebut. Walaupun pembahasan rinci jenis perhitungan akan diberikan pada subbab berikutnya, hubungan ini sudah menunjukkan bahwa optimasi geometri pun berakar pada Persamaan Schrödinger.

Unsur kedua adalah **fungsi gelombang**. Fungsi gelombang biasanya dilambangkan dengan **Ψ** atau **ψ**. Dalam interpretasi modern mekanika kuantum, fungsi gelombang bukan gelombang fisik dalam arti gelombang air atau gelombang bunyi. Fungsi gelombang adalah fungsi matematika yang memuat informasi tentang keadaan kuantum sistem. Ia dapat bernilai positif, negatif, atau kompleks. Nilai fungsi gelombang sendiri tidak langsung diamati sebagai besaran fisik. Makna fisiknya muncul melalui operasi tertentu, terutama melalui kuadrat modulus fungsi gelombang. Inilah titik yang sering menimbulkan miskonsepsi dalam pembelajaran kimia kuantum.

Menurut interpretasi Born, $|\Psi|^2$ berhubungan dengan rapat probabilitas. Jika fungsi gelombang bergantung pada koordinat posisi, maka kuadrat modulusnya menunjukkan peluang relatif menemukan partikel pada daerah ruang tertentu. Dalam sumber utama kimia kuantum, pembahasan rapat probabilitas elektron menjelaskan bahwa fungsi gelombang elektronik merupakan fungsi dari koordinat ruang dan spin elektron, sedangkan kuadrat modulus fungsi gelombang digunakan untuk menyatakan peluang menemukan elektron pada elemen volume tertentu. Dengan demikian, kimia kuantum tidak menggambarkan elektron sebagai partikel klasik yang memiliki lintasan pasti, melainkan sebagai entitas kuantum yang distribusi keberadaannya dinyatakan secara probabilistik.

Konsep probabilitas ini sangat mendasar bagi praktikum. Ketika mahasiswa memvisualisasikan orbital atom atau orbital molekul, gambar yang muncul bukan foto lintasan elektron. Visualisasi orbital biasanya menggambarkan permukaan dengan nilai fungsi tertentu atau daerah dengan probabilitas tinggi untuk menemukan elektron. Karena itu, warna, bentuk, fase positif-negatif, dan ukuran orbital harus ditafsirkan secara hati-hati. Orbital bukan “awan elektron” dalam arti benda kabur yang melayang, melainkan representasi matematis dari keadaan elektron. Kesalahan memahami orbital sebagai lintasan klasik akan merusak pemahaman tentang ikatan kimia, delokalisasi elektron, simetri orbital, dan reaktivitas.

Fungsi gelombang juga harus memenuhi syarat matematis tertentu. Ia harus bernilai tunggal, terbatas, kontinu, dan dapat dinormalisasi. Syarat normalisasi berarti total probabilitas menemukan partikel di seluruh ruang harus sama dengan satu. Secara konseptual, jika $|\psi(x)|^2\,dx$ menyatakan probabilitas menemukan partikel dalam interval kecil dx, maka integral seluruh probabilitas pada ruang yang diizinkan harus bernilai satu. Dalam praktikum partikel dalam kotak, syarat ini terlihat jelas. Fungsi gelombang partikel harus bernilai nol di dinding kotak dan hanya bentuk gelombang tertentu yang memenuhi syarat batas. Akibatnya, energi yang diperbolehkan tidak kontinu, tetapi diskret. Sumber utama kimia kuantum menjelaskan bahwa fungsi gelombang keadaan stasioner dan tingkat energi partikel dalam kotak satu dimensi diperoleh dengan menyelesaikan Persamaan Schrödinger tak bergantung waktu.

Konsep ini memperlihatkan hubungan langsung antara matematika dan fenomena kimia. Kuantisasi energi bukan asumsi yang ditempelkan dari luar, melainkan konsekuensi dari syarat batas dan sifat gelombang sistem. Pada partikel dalam kotak, hanya panjang gelombang tertentu yang "pas" di dalam kotak. Karena panjang gelombang berkaitan dengan momentum, dan momentum berkaitan dengan energi kinetik, maka hanya energi tertentu yang diperbolehkan. Inilah salah satu alasan model partikel dalam kotak sangat penting sebagai praktikum pertama dalam buku ini. Model tersebut sederhana, tetapi menunjukkan dengan jelas hubungan antara fungsi gelombang, syarat batas, probabilitas, dan energi.

Unsur ketiga adalah **energi**. Dalam Persamaan Schrödinger tak bergantung waktu, energi muncul sebagai nilai eigen. Ini berarti energi suatu keadaan kuantum tidak selalu dapat mengambil sembarang nilai. Pada sistem terikat seperti elektron dalam atom atau elektron dalam model kotak, energi terkuantisasi. Dalam atom dan molekul, tingkat energi elektronik, vibrasi, dan rotasi memiliki struktur tertentu yang dapat diamati melalui spektroskopi. Ketika molekul menyerap cahaya, transisi terjadi jika energi foton sesuai dengan selisih energi dua keadaan. Dalam sumber utama kimia kuantum, pembahasan teori gangguan bergantung waktu menjelaskan bahwa probabilitas transisi menjadi besar jika frekuensi gangguan memenuhi kondisi resonansi $\mathbf{\Delta E = h\nu}$, yang menjadi dasar serapan dan emisi radiasi oleh atom dan molekul.

Hubungan energi dengan frekuensi ini sangat penting bagi praktikum spektroskopi komputasi. Frekuensi vibrasi, spektrum IR, transisi elektronik, dan serapan UV-Vis semuanya berakar pada perbedaan energi antar keadaan. Ketika software menghitung frekuensi vibrasi molekul, ia sedang menganalisis bentuk permukaan energi potensial di sekitar titik minimum. Ketika software menghitung orbital HOMO dan LUMO, ia memberikan gambaran tingkat energi elektronik yang dapat digunakan untuk menafsirkan reaktivitas dan kemungkinan transisi elektronik. Ketika mahasiswa menghitung energi reaksi, mereka membandingkan energi keadaan awal dan akhir sistem. Semua kegiatan ini memiliki akar yang sama: energi dalam sistem kuantum diperoleh dari struktur Hamiltonian dan keadaan sistem.

Akan tetapi, energi dalam kimia komputasi harus ditafsirkan secara hati-hati. Energi total hasil perhitungan sering bernilai negatif besar dalam satuan Hartree. Nilai absolut energi total tidak selalu bermakna jika dibandingkan secara sembarangan antar sistem yang tidak sepadan. Yang sering lebih bermakna adalah perbedaan energi antara dua keadaan, dua geometri, dua konformer, atau reaktan dan produk yang dihitung dengan metode yang konsisten. Misalnya, energi total H_2O tidak dapat dibandingkan langsung dengan energi total CH_4 untuk menyimpulkan mana yang "lebih stabil" tanpa konteks reaksi atau pembentukan. Sebaliknya, energi dua konformer etanol yang dihitung dengan metode dan basis set sama dapat dibandingkan untuk menentukan konformer relatif lebih stabil. Dengan demikian, praktikum kimia kuantum harus melatih mahasiswa membedakan antara energi absolut dan energi relatif.

Unsur keempat adalah **probabilitas**. Mekanika kuantum mengubah cara ilmuwan memahami keadaan mikroskopik. Dalam mekanika klasik, jika posisi dan momentum awal suatu partikel diketahui dengan tepat, gerak masa depannya dapat diprediksi secara deterministik. Dalam mekanika kuantum, prinsip ketidakpastian dan sifat probabilistik fungsi gelombang membatasi cara kita berbicara tentang posisi dan momentum partikel. Bukan berarti sistem kuantum tidak memiliki keteraturan, tetapi keteraturannya dinyatakan melalui probabilitas, operator, dan

distribusi, bukan melalui lintasan pasti. Hal ini menjadi perubahan epistemologis besar dalam sejarah sains.

Bagi mahasiswa kimia, probabilitas harus dipahami sebagai bagian dari struktur teori, bukan sekadar akibat keterbatasan alat ukur. Ketika kita mengatakan orbital 1s atom hidrogen berbentuk bola, yang dimaksud adalah distribusi probabilitas elektron bersifat simetris bola. Ketika kita mengatakan orbital p memiliki dua lobus dengan bidang nodal, yang dimaksud adalah fungsi gelombang berubah tanda pada daerah tertentu dan probabilitas elektron nol pada bidang nodal. Ketika kita mengatakan elektron terdelokalisasi dalam benzena, yang dimaksud adalah fungsi gelombang atau kerapatan elektron π tersebar pada sistem cincin, bukan elektron bergerak mengelilingi cincin seperti planet kecil. Semua pernyataan ini menuntut cara berpikir probabilistik.

Dalam praktikum berbasis Python, probabilitas dapat dipelajari secara konkret. Pada model partikel dalam kotak, mahasiswa dapat menghitung $\psi(x)$ dan $\psi^2(x)$ untuk beberapa bilangan kuantum. Mereka dapat melihat bahwa fungsi gelombang dapat bernilai positif dan negatif, tetapi rapat probabilitas selalu tidak negatif. Mereka dapat melihat bahwa pada titik simpul, fungsi gelombang bernilai nol sehingga probabilitas menemukan partikel juga nol. Mereka juga dapat menghitung probabilitas menemukan partikel pada bagian tertentu dari kotak dengan mengintegralkan $\psi^2(x)$ pada interval tersebut. Dengan cara ini, konsep probabilitas tidak lagi abstrak, tetapi menjadi objek analisis numerik dan grafik.

Persamaan Schrödinger juga mengajarkan pentingnya **operator** dalam kimia kuantum. Dalam praktikum, mahasiswa sering melihat angka hasil perhitungan tanpa menyadari bahwa setiap angka berasal dari operator tertentu. Energi berasal dari Hamiltonian. Momen dipol berasal dari operator momen dipol. Momentum sudut berasal dari operator momentum sudut. Spin berasal dari operator spin. Nilai yang dapat diamati dalam mekanika kuantum bukan sekadar hasil pembacaan langsung, tetapi hasil operasi matematis terhadap fungsi keadaan. Ini menjelaskan mengapa kimia kuantum memerlukan matematika seperti kalkulus, aljabar linear, bilangan kompleks, matriks, dan persamaan diferensial.

Dalam konteks pembelajaran, hal ini tidak berarti mahasiswa praktikum harus menguasai seluruh rincian matematika tingkat lanjut sebelum menjalankan praktikum. Akan tetapi, mahasiswa harus memahami makna konseptualnya. Mereka harus tahu bahwa energi dari output adalah hasil prosedur matematis, bukan hasil tebakan software. Mereka harus tahu bahwa orbital adalah fungsi matematis, bukan objek fisik sederhana. Mereka harus tahu bahwa probabilitas muncul dari kuadrat fungsi gelombang, bukan dari ketidaktahuan biasa. Mereka harus tahu bahwa metode komputasi berbeda menghasilkan nilai berbeda karena menggunakan model dan hampiran berbeda terhadap Hamiltonian. Dengan kesadaran ini, praktikum menjadi sarana belajar teori, bukan hanya latihan teknis.

Latar historis Persamaan Schrödinger juga memberikan pelajaran metodologis. Schrödinger merumuskan mekanika gelombang pada tahun 1926 sebagai alternatif terhadap mekanika matriks Heisenberg. Keduanya kemudian terbukti ekuivalen dalam menjelaskan fenomena kuantum. Dalam kimia, mekanika gelombang lebih mudah divisualisasikan karena menggunakan fungsi ruang yang dapat dikaitkan dengan orbital dan distribusi elektron. Inilah sebabnya banyak buku kimia kuantum menggunakan Persamaan Schrödinger sebagai jalan utama memperkenalkan

struktur atom dan molekul. Levine, Atkins, McQuarrie, Szabo–Ostlund, dan Jensen sama-sama menempatkan Persamaan Schrödinger sebagai fondasi untuk memahami struktur elektronik, metode hampiran, dan kimia komputasi modern.

Sintesis literatur menunjukkan bahwa tidak ada kimia komputasi modern tanpa Persamaan Schrödinger, tetapi tidak semua praktik kimia komputasi menyelesaikan Persamaan Schrödinger secara langsung dan eksak. Hartree–Fock menyelesaikan bentuk hampiran melalui medan rata-rata elektron. DFT mengganti fungsi gelombang banyak elektron dengan kerapatan elektron sebagai variabel utama. Metode semiempiris menyederhanakan sebagian integral dan memasukkan parameter empiris. Mekanika molekular bahkan tidak memperlakukan elektron secara eksplisit, tetapi menggunakan fungsi energi klasik berbasis ikatan, sudut, torsi, dan interaksi nonikatan. Oleh karena itu, Persamaan Schrödinger harus dipahami sebagai fondasi konseptual, sedangkan metode praktikum adalah berbagai strategi untuk mendekati persoalan kuantum sesuai kebutuhan.

Analisis kritis terhadap Persamaan Schrödinger juga diperlukan. Persamaan ini sangat kuat, tetapi penerapannya pada sistem kimia nyata menghadapi kendala besar. Pertama, persoalan banyak elektron tidak memiliki penyelesaian eksak sederhana. Kedua, fungsi gelombang banyak elektron memiliki dimensi sangat tinggi karena bergantung pada koordinat semua elektron. Ketiga, korelasi elektron sulit dimodelkan secara akurat. Keempat, pemilihan basis set memengaruhi fleksibilitas deskripsi fungsi gelombang. Kelima, dalam banyak kasus kimia, efek relativistik, pelarut, suhu, dan lingkungan molekul juga perlu diperhitungkan. Karena itu, hasil praktikum kimia kuantum harus selalu dipahami sebagai hasil model, bukan kebenaran absolut.

Keterbatasan ini justru menjadi nilai pendidikan yang penting. Mahasiswa perlu menyadari bahwa sains tidak bekerja dengan cara “menghasilkan jawaban sempurna”, tetapi dengan membangun model yang semakin baik dan semakin dapat diuji. Sebuah perhitungan HF/STO-3G mungkin cukup untuk memperkenalkan konsep dasar orbital dan energi, tetapi tidak cukup untuk prediksi termokimia presisi tinggi. Perhitungan DFT/B3LYP/6-31G(d) mungkin memadai untuk praktikum organik sederhana, tetapi belum tentu tepat untuk interaksi dispersi lemah tanpa koreksi tambahan. Perhitungan fase gas dapat menjelaskan kecenderungan dasar, tetapi mungkin berbeda dari kondisi larutan. Dengan demikian, Persamaan Schrödinger mengajarkan dua hal sekaligus: kekuatan teori kuantum dan perlunya kerendahan hati metodologis dalam menggunakan hampiran.

Contoh konkret dapat diberikan pada molekul hidrogen, H_2. Dalam pendekatan sederhana, pembentukan ikatan H–H dapat dipahami sebagai akibat kombinasi orbital atom 1s dari dua atom hidrogen membentuk orbital molekul ikatan dan antiikatan. Elektron yang menempati orbital ikatan menurunkan energi sistem relatif terhadap dua atom hidrogen terpisah. Secara praktikum, mahasiswa dapat menjalankan perhitungan energi H_2 pada beberapa panjang ikatan H–H, lalu membuat grafik energi terhadap jarak. Grafik tersebut akan menunjukkan adanya jarak optimum, yaitu jarak ketika energi minimum. Secara konseptual, grafik ini adalah potongan dari permukaan energi potensial yang berakar pada penyelesaian Hamiltonian elektronik untuk berbagai posisi inti. Dengan cara ini, Persamaan Schrödinger menjadi dasar untuk memahami panjang ikatan sebagai konsekuensi energi minimum.

Contoh lain adalah atom helium. Atom helium memiliki dua elektron, sehingga persoalannya tidak dapat diselesaikan seeksak atom hidrogen karena terdapat tolakan elektron–elektron. Sumber utama kimia kuantum membahas bagaimana teori gangguan dan metode variasi digunakan untuk memperbaiki pendekatan keadaan dasar helium, termasuk pentingnya korelasi gerak kedua elektron dan penggunaan parameter variasi sebagai muatan inti efektif. Contoh helium memperlihatkan mengapa metode hampiran diperlukan dalam kimia kuantum. Jika atom dua elektron saja sudah membutuhkan pendekatan variasi atau gangguan, maka molekul organik, kompleks logam, atau biomolekul tentu memerlukan strategi komputasi yang lebih sistematis.

Dalam praktikum, mahasiswa tidak harus menyelesaikan atom helium secara analitik, tetapi mereka perlu memahami pesan metodologisnya. Semakin banyak partikel, semakin kompleks Hamiltonian. Semakin kompleks Hamiltonian, semakin diperlukan hampiran. Semakin banyak hampiran, semakin penting validasi. Inilah alasan mengapa praktikum kimia kuantum selalu memerlukan dokumentasi metode, basis set, charge, multiplicity, dan parameter. Dua hasil komputasi hanya dapat dibandingkan secara bermakna jika konteks metodologinya jelas.

Persamaan Schrödinger juga menjadi dasar konseptual bagi keterkaitan antara struktur dan spektrum. Dalam spektroskopi, transisi terjadi antara tingkat energi. Setiap spektrum pada dasarnya adalah jejak perbedaan energi antar keadaan kuantum. Spektrum IR berkaitan dengan perubahan tingkat energi vibrasi. Spektrum UV-Vis berkaitan dengan transisi elektronik. Spektrum rotasi berkaitan dengan tingkat energi rotasi. Jika mahasiswa memahami bahwa tingkat energi berasal dari penyelesaian Persamaan Schrödinger, maka mereka akan melihat spektrum bukan sekadar kumpulan puncak, tetapi sebagai peta perbedaan energi molekul. Hal ini akan sangat berguna ketika nanti mahasiswa mempelajari praktikum frekuensi vibrasi dan spektrum IR.

Selain itu, Persamaan Schrödinger menjadi titik awal untuk memahami mengapa struktur elektronik molekul menentukan reaktivitas. Reaksi kimia pada dasarnya melibatkan perubahan distribusi elektron dan perubahan susunan inti. Orbital frontier, yaitu HOMO dan LUMO, sering digunakan untuk menjelaskan kecenderungan donor dan akseptor elektron. Walaupun teori orbital frontier adalah penyederhanaan, ia sangat berguna untuk memahami reaktivitas organik, fotokimia, dan interaksi molekul. Nilai energi HOMO dan LUMO yang diperoleh dari perhitungan komputasi merupakan hasil dari model struktur elektronik. Oleh karena itu, ketika mahasiswa menganalisis gap HOMO-LUMO, mereka sebenarnya sedang menggunakan konsekuensi praktis dari teori kuantum.

Dalam konteks penggunaan ChemCompute dan software kimia komputasi lain, pemahaman terhadap Persamaan Schrödinger akan membantu mahasiswa membaca pilihan metode secara lebih bermakna. Ketika mahasiswa memilih “single point energy”, mereka sedang menghitung energi elektronik pada geometri tertentu. Ketika memilih “geometry optimization”, mereka sedang mencari geometri dengan energi minimum. Ketika memilih “frequency calculation”, mereka sedang menganalisis kelengkungan permukaan energi potensial di sekitar titik minimum. Ketika memilih metode HF, DFT, atau semiempiris, mereka sedang memilih cara tertentu untuk menghampiri penyelesaian persoalan kuantum. Dengan demikian, setiap menu software dapat diterjemahkan kembali ke dalam bahasa teori.

Dari sisi programming Python, Persamaan Schrödinger dapat diperkenalkan secara bertahap melalui model sederhana. Mahasiswa tidak perlu langsung menyelesaikan Hamiltonian molekul banyak elektron. Mereka dapat memulai dari partikel dalam kotak, osilator harmonik, atau visualisasi fungsi gelombang sederhana. Model-model ini mengajarkan struktur umum mekanika kuantum: ada operator, ada fungsi gelombang, ada nilai energi, ada syarat batas, ada probabilitas, dan ada normalisasi. Setelah memahami pola ini, mahasiswa lebih siap menafsirkan output software untuk molekul nyata. Python berfungsi sebagai jembatan antara persamaan dan visualisasi.

Implikasi pembelajaran dari subbab ini adalah bahwa setiap praktikum harus selalu kembali pada empat pertanyaan dasar. Pertama, apa Hamiltonian atau model energi yang digunakan? Kedua, apa bentuk representasi keadaan sistem, apakah fungsi gelombang, orbital, kerapatan elektron, atau koordinat molekul? Ketiga, energi atau sifat apa yang dihitung? Keempat, bagaimana probabilitas, distribusi elektron, atau perbedaan energi ditafsirkan secara kimia? Jika mahasiswa terbiasa menjawab empat pertanyaan ini, maka mereka akan lebih kritis saat menggunakan software dan lebih teliti saat menulis laporan.

Persamaan Schrödinger akhirnya dapat dipahami sebagai fondasi yang menyatukan teori dan praktikum. Ia menjelaskan mengapa energi terkuantisasi, mengapa orbital muncul, mengapa probabilitas menjadi pusat interpretasi, mengapa metode hampiran diperlukan, dan mengapa software kimia komputasi dapat menghitung sifat molekul. Tanpa memahami Persamaan Schrödinger, mahasiswa mungkin tetap bisa menjalankan praktikum, tetapi pemahamannya akan dangkal. Dengan memahami Persamaan Schrödinger, mahasiswa dapat melihat bahwa setiap angka dalam output komputasi memiliki akar teoretis, setiap grafik memiliki makna fisik, dan setiap model memiliki batas validitas.

Subbab berikutnya akan membahas jenis-jenis perhitungan dalam praktikum kimia kuantum, meliputi optimasi geometri, single point energy, frekuensi vibrasi, HOMO-LUMO, momen dipol, muatan atom, dan energi reaksi. Pembahasan tersebut akan menunjukkan bagaimana fondasi Persamaan Schrödinger diterjemahkan menjadi pekerjaan praktikum yang konkret dan dapat dijalankan melalui ChemCompute, software kimia komputasi, maupun kode Python.

1.4 Jenis Perhitungan dalam Praktikum Kimia Kuantum

Jenis perhitungan dalam praktikum kimia kuantum adalah ragam pekerjaan komputasi yang dilakukan untuk memperoleh informasi struktur, energi, distribusi elektron, sifat fisik, sifat spektroskopi, dan kecenderungan reaktivitas suatu sistem atomik atau molekuler. Dalam praktiknya, mahasiswa tidak hanya memilih “metode” seperti Hartree–Fock, DFT, semiempiris, atau mekanika molekular, tetapi juga memilih **jenis pekerjaan komputasi** atau **job type**. Jenis pekerjaan ini menentukan pertanyaan ilmiah yang hendak dijawab. Jika pertanyaannya adalah “bagaimana struktur paling stabil molekul ini?”, maka perhitungannya adalah optimasi geometri. Jika pertanyaannya “berapa energi molekul pada struktur tertentu?”, maka pekerjaannya adalah single point energy. Jika pertanyaannya “apakah struktur ini benar-benar minimum dan bagaimana spektrum IR-nya?”, maka pekerjaannya adalah perhitungan frekuensi. Jika pertanyaannya “bagaimana kecenderungan molekul sebagai donor atau akseptor elektron?”, maka analisis HOMO-LUMO menjadi relevan. Jika pertanyaannya “apakah molekul polar?”, maka momen

dipol perlu dihitung. Jika pertanyaannya “atom mana yang kaya atau miskin elektron?”, maka analisis muatan atom diperlukan. Jika pertanyaannya “reaksi atau konformer mana yang lebih stabil?”, maka energi reaksi dan termodinamika menjadi pusat analisis.

Dalam sumber kimia kuantum yang digunakan sebagai dasar buku ini, bagian penguatan praktikum komputasi kimia kuantum modern menegaskan bahwa praktikum dasar dapat dimulai dari optimasi geometri molekul kecil seperti H_2O, NH_3, CH_4, CO_2, dan etena; setelah geometri konvergen, mahasiswa dapat menghitung frekuensi vibrasi, momen dipol, muatan atom, energi HOMO-LUMO, dan visualisasi orbital molekul. Pernyataan ini sangat penting karena menunjukkan bahwa jenis perhitungan dalam praktikum kimia kuantum harus disusun secara berjenjang. Mahasiswa tidak langsung diminta menghitung energi reaksi kompleks sebelum memahami struktur minimum. Mahasiswa juga tidak seharusnya menafsirkan spektrum IR sebelum memastikan bahwa struktur hasil optimasi benar-benar berada pada titik minimum. Alur kerja yang baik harus bergerak dari struktur awal menuju validasi, lalu menuju analisis sifat molekul.

Secara teoretis, semua jenis perhitungan tersebut berkaitan dengan Persamaan Schrödinger dan Hamiltonian molekuler yang telah dibahas pada subbab sebelumnya. Namun, dalam praktik komputasi, persoalan tersebut diterjemahkan menjadi prosedur numerik yang berbeda. Optimasi geometri berkaitan dengan perubahan energi terhadap koordinat inti. Single point energy berkaitan dengan penyelesaian struktur elektronik pada geometri tetap. Perhitungan frekuensi berkaitan dengan turunan kedua energi terhadap koordinat inti. HOMO-LUMO berkaitan dengan tingkat energi orbital molekul. Momen dipol berkaitan dengan distribusi muatan dan pemisahan pusat muatan positif-negatif. Muatan atom berkaitan dengan partisi kerapatan elektron. Energi reaksi berkaitan dengan perbedaan energi antara spesies yang dihitung secara konsisten. Dengan demikian, satu teori dasar menghasilkan berbagai jenis pekerjaan praktikum yang berbeda.

Secara historis, jenis-jenis perhitungan ini berkembang seiring perkembangan kimia komputasi. Pada masa awal, perhitungan kuantum terutama berfokus pada energi elektronik dan struktur atom sederhana. Setelah komputer digital berkembang, metode Hartree–Fock, semiempiris, dan ab initio mulai digunakan untuk molekul kecil. Perkembangan fungsi basis, algoritma SCF, metode optimasi geometri, dan analisis vibrasi kemudian membuat kimia komputasi semakin praktis digunakan oleh kimiawan. Munculnya Gaussian, GAMESS, ORCA, Q-Chem, NWChem, Psi4, PySCF, dan berbagai paket lain memperluas cakupan perhitungan dari sekadar energi elektronik menuju spektrum, termodinamika, sifat magnetik, sifat optik, reaktivitas, serta sistem besar. Dokumen rujukan software kimia komputasi menegaskan bahwa ORCA, misalnya, menonjol sebagai perangkat lunak akademik yang efisien untuk perhitungan spektroskopi dan sistem open-shell, tetapi pemahaman teori tetap menjadi prasyarat agar mahasiswa tidak menjadi sekadar operator program.

Jenis perhitungan pertama dan paling fundamental dalam praktikum adalah **optimasi geometri**. Optimasi geometri adalah proses mencari susunan koordinat inti atom yang menghasilkan energi minimum pada permukaan energi potensial. Dalam bahasa sederhana, software memindahkan posisi atom sedikit demi sedikit, menghitung energi dan gradien, lalu mencari struktur yang lebih stabil. Proses ini berlangsung berulang hingga perubahan energi dan gaya pada atom memenuhi kriteria konvergensi. Optimasi geometri sangat penting karena sebagian besar sifat molekul

bergantung pada struktur. Panjang ikatan, sudut ikatan, torsi, momen dipol, energi HOMO-LUMO, frekuensi vibrasi, dan energi reaksi dapat berubah jika geometri molekul berubah.

Secara matematis, geometri kesetimbangan diperoleh ketika turunan pertama energi terhadap koordinat inti mendekati nol. Dalam bentuk konseptual:

$\partial E/\partial R_A = 0$ pada geometri kesetimbangan

Di sini, **R_A** menyatakan koordinat inti atom A. Jika turunan pertama energi terhadap semua koordinat inti bernilai nol, struktur berada pada titik stasioner. Namun, titik stasioner belum tentu minimum. Ia bisa berupa minimum lokal, maksimum, atau titik pelana. Karena itu, optimasi geometri idealnya diikuti dengan perhitungan frekuensi untuk memastikan sifat titik stasioner. Sumber kimia kuantum yang digunakan menegaskan bahwa optimasi geometri harus disertai pengecekan frekuensi: semua frekuensi nyata untuk minimum, sedangkan satu frekuensi imajiner menunjukkan keadaan transisi orde pertama.

Dalam praktikum dasar, optimasi geometri dapat dilakukan pada molekul kecil seperti H_2, H_2O, NH_3, CH_4, CO_2, etena, formaldehida, atau benzena. Mahasiswa dapat membandingkan hasil optimasi dengan konsep VSEPR, teori ikatan valensi, teori orbital molekul, dan data eksperimen. Misalnya, pada H_2O, mahasiswa dapat mengoptimasi struktur dan memperoleh panjang ikatan O–H serta sudut H–O–H. Dari sana, mereka dapat membandingkan hasil HF/STO-3G, PM6, dan DFT/B3LYP/6-31G(d). Jika sudut H–O–H berbeda antar metode, mahasiswa perlu memahami bahwa metode berbeda menggambarkan distribusi elektron dengan tingkat akurasi berbeda. Perbedaan ini bukan sekadar “kesalahan software”, melainkan konsekuensi dari model teoritis dan basis set yang digunakan.

Optimasi geometri juga melatih mahasiswa memahami konsep **konvergensi**. Dalam perhitungan struktur elektronik, konvergensi bukan hanya berarti program selesai berjalan. Konvergensi berarti proses iteratif telah memenuhi batas toleransi tertentu. Pada optimasi geometri, energi elektronik, gradien maksimum, perpindahan atom, dan perubahan energi biasanya diperiksa. Dokumen software kimia komputasi menjelaskan bahwa pada ORCA, optimasi geometri tidak hanya menuntut energi elektronik konvergen, tetapi juga gradien energi terhadap posisi atom; kriteria seperti Opt dan TightOpt digunakan untuk menentukan seberapa ketat proses optimasi dilakukan. Pemahaman ini penting karena struktur yang belum konvergen tidak layak dijadikan dasar untuk analisis spektrum, termodinamika, atau energi reaksi.

Jenis perhitungan kedua adalah **single point energy** atau energi titik tunggal. Single point energy adalah perhitungan energi elektronik pada geometri yang tetap, tanpa mengubah posisi atom. Jika optimasi geometri bertanya “di mana struktur yang paling stabil?”, maka single point energy bertanya “berapa energi struktur ini pada metode tertentu?”. Perbedaan ini sangat penting. Dalam optimasi, software mengubah koordinat atom untuk mencari energi minimum. Dalam single point, koordinat dibekukan, lalu energi dihitung pada konfigurasi tersebut. Dokumen rujukan software kimia komputasi membedakan secara jelas bahwa optimasi geometri memanipulasi koordinat atom untuk mencari posisi paling stabil, sedangkan single point energy membekukan pergerakan atom dan hanya menghitung struktur elektronik serta energi total pada konfigurasi spasial tunggal.

Single point energy sangat berguna dalam strategi bertingkat. Dalam praktik riset, struktur sering dioptimasi dengan metode yang relatif ekonomis, kemudian energinya dihitung ulang dengan metode atau basis set yang lebih tinggi. Misalnya, mahasiswa dapat mengoptimasi molekul dengan B3LYP/def2-SVP, lalu menghitung single point energy dengan B3LYP/def2-TZVP. Strategi ini menghemat waktu karena optimasi geometri dengan basis set besar dapat mahal secara komputasi. Dalam praktikum, strategi sederhana dapat dilakukan dengan mengoptimasi H_2O pada HF/STO-3G, kemudian menghitung energi single point pada basis set yang lebih besar, misalnya 3-21G, 6-31G, dan 6-31G(d), untuk melihat pengaruh basis set terhadap energi total.

Single point energy juga menjadi dasar untuk mempelajari kurva energi potensial. Misalnya, pada molekul H_2, mahasiswa dapat menetapkan beberapa panjang ikatan H–H, menjalankan single point energy pada tiap jarak, lalu membuat grafik energi terhadap panjang ikatan. Grafik tersebut akan menunjukkan adanya jarak optimum ketika energi minimum. Dari grafik ini, mahasiswa dapat memahami makna ikatan kimia secara kuantitatif: ikatan terbentuk ketika susunan inti dan elektron menghasilkan penurunan energi relatif terhadap atom terpisah. Di sini, Python berperan penting untuk membaca data energi, membuat grafik, menentukan titik minimum, dan menafsirkan perubahan energi.

Jenis perhitungan ketiga adalah **perhitungan frekuensi vibrasi**. Perhitungan frekuensi digunakan untuk memperoleh mode normal vibrasi molekul, frekuensi vibrasi, intensitas IR jika tersedia, koreksi zero-point energy, serta besaran termodinamika seperti entalpi dan entropi. Secara teoretis, frekuensi vibrasi diperoleh dari analisis turunan kedua energi terhadap koordinat inti di sekitar titik stasioner. Jika optimasi geometri menggunakan informasi energi dan gradien untuk mencari titik stasioner, maka perhitungan frekuensi memeriksa kelengkungan permukaan energi potensial di sekitar titik tersebut. Kelengkungan positif menunjukkan arah stabil; kelengkungan negatif menghasilkan frekuensi imajiner.

Frekuensi imajiner memiliki makna struktural yang penting. Untuk struktur minimum, semua frekuensi harus nyata atau positif. Jika ada satu frekuensi imajiner, struktur tersebut biasanya merupakan keadaan transisi orde pertama. Jika ada lebih dari satu frekuensi imajiner, struktur belum merupakan minimum maupun keadaan transisi yang benar. Dalam praktikum dasar, mahasiswa harus dibiasakan memeriksa frekuensi setelah optimasi. Struktur yang tampak bagus secara visual belum tentu minimum. Struktur dapat saja berada pada titik pelana atau belum teroptimasi dengan baik. Karena itu, perhitungan frekuensi bukan hanya untuk membuat spektrum IR, tetapi juga untuk validasi struktur.

Perhitungan frekuensi juga memiliki aspek kritis. Banyak metode komputasi menghasilkan frekuensi harmonik, sedangkan vibrasi molekul nyata bersifat anharmonik. Oleh karena itu, frekuensi hasil perhitungan sering perlu dikoreksi dengan faktor skala jika dibandingkan dengan data eksperimen. Dokumen software kimia komputasi memberi contoh bahwa frekuensi vibrasi harmonik yang dihitung dengan Hartree–Fock secara konsisten cenderung lebih tinggi dibandingkan frekuensi fundamental eksperimen; hal ini berkaitan dengan asumsi osilator harmonik dan pengabaian korelasi elektron, sehingga literatur menggunakan faktor skala empiris untuk memperbaiki perbandingan dengan eksperimen. Dalam praktikum, hal ini menjadi kesempatan pedagogis untuk mengajarkan bahwa hasil komputasi perlu divalidasi, dikoreksi, dan ditafsirkan secara ilmiah.

Jenis perhitungan keempat adalah **analisis HOMO-LUMO**. HOMO adalah Highest Occupied Molecular Orbital, yaitu orbital molekul terisi dengan energi tertinggi. LUMO adalah Lowest Unoccupied Molecular Orbital, yaitu orbital molekul kosong dengan energi terendah. Selisih energi antara LUMO dan HOMO sering disebut gap HOMO-LUMO. Dalam banyak konteks, gap ini digunakan secara kualitatif untuk menilai stabilitas elektronik, kecenderungan reaktivitas, kekerasan-kelunakan molekul, dan kemungkinan transisi elektronik. Molekul dengan gap besar umumnya lebih stabil secara elektronik dan kurang mudah mengalami eksitasi, sedangkan molekul dengan gap kecil cenderung lebih mudah mengalami transisi elektronik atau interaksi donor-akseptor.

Analisis HOMO-LUMO harus dilakukan dengan hati-hati. Energi orbital dari metode Hartree–Fock dan DFT tidak selalu dapat diperlakukan sebagai energi ionisasi atau afinitas elektron secara langsung. Dalam Hartree–Fock, teorema Koopmans memberikan hubungan hampiran antara energi HOMO dan energi ionisasi, tetapi tetap memiliki keterbatasan karena relaksasi orbital dan korelasi elektron tidak sepenuhnya dipertimbangkan. Dalam DFT, interpretasi energi orbital Kohn–Sham juga memerlukan kehati-hatian. Karena itu, dalam praktikum, HOMO-LUMO sebaiknya digunakan sebagai alat interpretasi kualitatif dan semi-kuantitatif, bukan sebagai kebenaran mutlak. Mahasiswa perlu memahami bahwa gambar orbital dan nilai gap adalah hasil model, bukan pengukuran langsung.

Contoh konkret dapat diberikan pada etilena, butadiena, dan benzena. Etilena memiliki satu ikatan rangkap dengan sistem π sederhana. Butadiena memiliki dua ikatan rangkap terkonjugasi sehingga delokalisasi elektron lebih luas. Benzena memiliki sistem π aromatik yang lebih stabil. Jika mahasiswa menghitung HOMO-LUMO ketiganya, mereka dapat mengamati perubahan gap dan bentuk orbital. Butadiena biasanya menunjukkan gap lebih kecil daripada etilena karena konjugasi memperluas delokalisasi elektron. Benzena menunjukkan struktur orbital yang mencerminkan simetri cincin aromatik. Dengan Python, mahasiswa dapat memasukkan energi HOMO dan LUMO dari output software, menghitung gap, membuat diagram tingkat energi, serta membandingkan parameter reaktivitas global sederhana seperti hardness dan softness.

Jenis perhitungan kelima adalah **momen dipol**. Momen dipol adalah ukuran pemisahan muatan positif dan negatif dalam molekul. Secara konseptual, molekul memiliki momen dipol jika pusat muatan positif dan pusat muatan negatif tidak berimpit. Besar momen dipol dipengaruhi oleh polaritas ikatan dan geometri molekul. Molekul CO_2 memiliki ikatan C=O yang polar, tetapi bentuknya linear dan simetris sehingga vektor dipol saling meniadakan. Molekul H_2O juga memiliki ikatan O–H yang polar, tetapi bentuknya bengkok sehingga vektor dipol tidak saling meniadakan dan menghasilkan momen dipol bersih. Di sinilah hubungan antara struktur, simetri, dan sifat listrik molekul menjadi jelas.

Dalam perhitungan kimia kuantum, momen dipol dihitung dari distribusi muatan inti dan elektron. Nilainya biasanya dilaporkan dalam Debye. Perhitungan momen dipol bergantung pada kualitas geometri dan metode struktur elektronik. Struktur yang belum teroptimasi dapat memberikan momen dipol yang tidak representatif. Basis set yang terlalu kecil juga dapat memengaruhi distribusi elektron. Karena itu, dalam praktikum, momen dipol sebaiknya dihitung setelah optimasi geometri. Mahasiswa kemudian dapat membandingkan hasil komputasi dengan prediksi VSEPR

dan simetri molekul. Analisis ini sangat baik untuk menghubungkan konsep kimia umum dengan kimia kuantum.

Momen dipol juga dapat dianalisis melalui Python secara sederhana jika tersedia muatan parsial dan koordinat atom. Meskipun momen dipol sebenarnya berasal dari distribusi elektron kontinu, mahasiswa dapat membuat model diskrit sederhana dengan memperlakukan muatan parsial pada atom sebagai titik muatan. Dari koordinat dan muatan parsial, vektor dipol dapat dihitung dan divisualisasikan. Model sederhana ini tidak menggantikan hasil kuantum, tetapi membantu mahasiswa memahami hubungan antara distribusi muatan, posisi atom, dan polaritas molekul. Dengan demikian, Python berfungsi sebagai alat pedagogis untuk menjembatani angka output dengan konsep vektor.

Jenis perhitungan keenam adalah **analisis muatan atom**. Dalam molekul, elektron tidak selalu terbagi merata antar atom. Atom elektronegatif cenderung menarik kerapatan elektron lebih besar. Analisis muatan atom bertujuan membagi kerapatan elektron molekul menjadi kontribusi atomik sehingga diperoleh muatan parsial pada setiap atom. Muatan atom berguna untuk memahami polaritas ikatan, pusat nukleofilik-elektrofilik, interaksi nonkovalen, dan kecenderungan reaktivitas. Namun, muatan atom adalah konsep model, bukan observabel langsung dalam mekanika kuantum. Tidak ada satu-satunya cara yang mutlak benar untuk membagi kerapatan elektron menjadi muatan atom.

Metode analisis muatan yang umum meliputi Mulliken population analysis, Löwdin charge, Natural Population Analysis, Hirshfeld charge, CHELPG, RESP, dan metode berbasis potensial elektrostatik. Setiap metode memiliki asumsi dan kelemahan. Muatan Mulliken sangat bergantung pada basis set dan sering kurang stabil ketika basis set berubah. Muatan berbasis potensial elektrostatik berguna untuk force field dan simulasi molekular, tetapi juga bergantung pada prosedur fitting. Karena itu, mahasiswa tidak boleh memperlakukan muatan atom sebagai "muatan nyata" yang berdiri sendiri. Muatan atom harus ditafsirkan sebagai indikator distribusi elektron menurut skema partisi tertentu.

Dalam praktikum, analisis muatan dapat dilakukan pada molekul seperti formaldehida, metanol, amonia, nitrobenzena, atau asam asetat. Mahasiswa dapat melihat bahwa atom O dalam formaldehida memiliki muatan parsial negatif, sedangkan atom C karbonil memiliki muatan parsial positif. Hal ini membantu menjelaskan mengapa karbon karbonil rentan terhadap serangan nukleofil. Pada nitrobenzena, mahasiswa dapat melihat pengaruh gugus nitro terhadap distribusi muatan cincin aromatik. Namun, analisis harus tetap kritis. Jika hasil muatan berbeda antar metode, mahasiswa perlu menjelaskan bahwa perbedaan tersebut berasal dari cara pembagian kerapatan elektron, bukan karena molekul berubah secara fisik.

Jenis perhitungan ketujuh adalah **energi reaksi dan termodinamika**. Energi reaksi dihitung dari selisih energi produk dan reaktan. Dalam bentuk paling sederhana:

$$\Delta E = \Sigma E_{produk} - \Sigma E_{reaktan}$$

Jika ΔE negatif, produk memiliki energi elektronik lebih rendah daripada reaktan dalam model yang digunakan. Namun, dalam kimia nyata, spontanitas reaksi tidak hanya ditentukan oleh energi

elektronik. Koreksi zero-point energy, entalpi termal, entropi, suhu, dan efek lingkungan juga penting. Karena itu, energi bebas Gibbs sering digunakan untuk menilai kecenderungan termodinamika:

$\Delta G \approx \Delta E\_elektronik + \Delta ZPE + \Delta H\_termal - T\Delta S$

Rumus konseptual ini juga ditampilkan dalam sumber kimia kuantum yang digunakan pada bagian penguatan praktikum komputasi modern.

Dalam praktikum dasar, energi reaksi dapat dipelajari melalui sistem sederhana seperti isomerisasi cis-trans 2-butena, perbandingan konformer etanol, protonasi amonia, hidrasi formaldehida, dimer air, atau dimer asam asetat. Mahasiswa harus memahami bahwa semua spesies yang dibandingkan harus dihitung dengan metode, basis set, dan kondisi yang sama. Jika reaktan dihitung dengan B3LYP/6-31G(d), produk juga harus dihitung dengan metode dan basis set yang sama. Jika satu struktur menggunakan model pelarut, semua struktur yang dibandingkan dalam reaksi tersebut juga harus menggunakan model pelarut yang sama. Konsistensi ini merupakan syarat utama agar perbedaan energi bermakna.

Energi reaksi juga mengajarkan pentingnya stoikiometri. Jika reaksi melibatkan dua molekul reaktan dan satu produk, energi total reaktan harus dijumlahkan sesuai koefisien stoikiometri. Kesalahan umum mahasiswa adalah membandingkan energi satu molekul produk dengan satu molekul reaktan tanpa memperhatikan jumlah spesies. Kesalahan lain adalah mencampur satuan, misalnya membandingkan Hartree dengan kJ/mol tanpa konversi. Dalam Python, mahasiswa dapat menulis skrip sederhana untuk memasukkan energi dalam Hartree, mengonversi ke kJ/mol, menjumlahkan sesuai koefisien reaksi, menghitung ΔE, ΔH, atau ΔG, lalu membuat diagram energi reaksi. Proses ini melatih ketelitian numerik dan pemahaman kimia sekaligus.

Jenis perhitungan energi reaksi juga dapat dikaitkan dengan **efek pelarut**. Banyak reaksi kimia berlangsung dalam larutan, bukan fase gas. Molekul polar atau bermuatan dapat distabilkan oleh pelarut. Model pelarut kontinu seperti PCM, CPCM, atau SMD digunakan untuk menghampiri pengaruh lingkungan pelarut tanpa memodelkan setiap molekul pelarut secara eksplisit. Namun, model ini memiliki keterbatasan, terutama jika interaksi spesifik seperti ikatan hidrogen eksplisit sangat penting. Dalam praktikum, mahasiswa dapat membandingkan energi protonasi amonia atau kestabilan ion dalam fase gas dan pelarut air. Perbandingan ini menunjukkan bahwa lingkungan kimia sangat memengaruhi stabilitas relatif.

Selain tujuh jenis perhitungan utama tersebut, praktikum kimia kuantum juga dapat memuat visualisasi orbital, analisis permukaan potensial elektrostatik, pencarian keadaan transisi, scanning koordinat reaksi, analisis ikatan hidrogen, atau perhitungan sifat spektroskopi lain. Namun, dalam buku praktikum dasar ini, fokus perlu dibatasi agar mahasiswa memperoleh fondasi yang kokoh. Urutan yang paling pedagogis adalah: membangun struktur, mengoptimasi geometri, memvalidasi dengan frekuensi, menghitung sifat elektronik, menganalisis distribusi muatan, lalu membandingkan energi reaksi atau konformer. Urutan ini menghindarkan mahasiswa dari kesalahan menafsirkan hasil yang belum tervalidasi.

Hubungan antara jenis perhitungan dan software juga perlu dipahami. ChemCompute dapat digunakan sebagai jalur berbasis web untuk menjalankan perhitungan dasar menggunakan paket yang tersedia. Avogadro dapat digunakan untuk membangun dan mengedit struktur molekul. ORCA, GAMESS, Gaussian, Psi4, atau PySCF dapat digunakan untuk perhitungan struktur elektronik. Molden, Jmol, Gabedit, GaussView, atau Multiwfn dapat digunakan untuk visualisasi dan analisis. Python digunakan untuk mengolah hasil, membuat grafik, menghitung parameter turunan, menyusun tabel, dan memeriksa konsistensi data. Dengan alur ini, mahasiswa belajar bahwa software bukan satu entitas tunggal, melainkan ekosistem alat yang saling melengkapi.

Dalam rancangan praktikum modern, setiap jenis perhitungan sebaiknya memiliki dua jalur: **jalur software** dan **jalur Python**. Jalur software menghasilkan data utama dari perhitungan kimia komputasi. Jalur Python membantu mahasiswa memahami, memeriksa, dan menafsirkan data tersebut. Misalnya, pada optimasi geometri, software menghasilkan koordinat akhir, sedangkan Python menghitung panjang ikatan dan sudut ikatan. Pada single point energy, software menghasilkan energi total, sedangkan Python membuat grafik energi terhadap parameter tertentu. Pada frekuensi vibrasi, software menghasilkan daftar frekuensi dan intensitas, sedangkan Python membuat spektrum IR simulasi. Pada HOMO-LUMO, software menghasilkan energi orbital, sedangkan Python menghitung gap dan membuat diagram energi. Pada energi reaksi, software menghasilkan energi masing-masing spesies, sedangkan Python menghitung ΔE, ΔH, ΔG, dan diagram reaksi.

Pendekatan ganda ini memiliki implikasi pendidikan yang penting. Mahasiswa tidak hanya belajar menggunakan software, tetapi juga belajar memvalidasi hasil. Mereka dapat memeriksa apakah jarak ikatan masuk akal, apakah sudut ikatan sesuai geometri, apakah gap HOMO-LUMO konsisten dengan sistem konjugasi, apakah frekuensi imajiner muncul, apakah momen dipol sesuai simetri, dan apakah energi reaksi dihitung dengan stoikiometri benar. Proses ini membangun kebiasaan ilmiah yang kritis. Mahasiswa belajar bahwa output komputasi tidak boleh disalin begitu saja ke laporan. Output harus dibaca, dipilih, diuji, ditafsirkan, dan dikaitkan dengan teori.

Analisis kritis terhadap jenis perhitungan juga perlu diberikan sejak awal. Pertama, tidak semua jenis perhitungan cocok untuk semua sistem. Molekul kecil dapat dihitung dengan metode relatif tinggi, sedangkan sistem besar memerlukan kompromi metode. Kedua, hasil perhitungan sangat bergantung pada input. Struktur awal yang buruk dapat menyebabkan optimasi gagal atau menuju minimum yang tidak diinginkan. Ketiga, basis set dan metode memengaruhi hasil. Keempat, beberapa sifat lebih sensitif daripada sifat lain. Energi relatif sering lebih sensitif terhadap metode dibanding panjang ikatan sederhana. Kelima, hasil komputasi harus dibandingkan dengan data eksperimen atau literatur jika tersedia. Dokumen rujukan kimia kuantum menekankan bahwa laporan praktikum perlu memuat perbandingan dengan data eksperimen atau literatur bila tersedia, dan perbedaan harus dianalisis dari sisi metode, basis set, korelasi elektron, efek pelarut, serta keterbatasan model.

Contoh komprehensif dapat diberikan melalui molekul formaldehida, H_2CO. Pertama, mahasiswa membangun struktur formaldehida. Kedua, mereka melakukan optimasi geometri untuk memperoleh struktur minimum. Ketiga, mereka menjalankan perhitungan frekuensi untuk memastikan tidak ada frekuensi imajiner dan memperoleh puncak C=O. Keempat, mereka menghitung HOMO-LUMO untuk melihat orbital frontier. Kelima, mereka menganalisis muatan

atom untuk memahami mengapa karbon karbonil bersifat elektrofilik. Keenam, mereka menghitung momen dipol untuk menilai polaritas molekul. Ketujuh, mereka dapat membandingkan energi formaldehida dengan produk hidrasi untuk mempelajari energi reaksi. Satu molekul yang sama dapat menjadi pusat berbagai jenis perhitungan yang saling berkaitan.

Contoh lain adalah nitrogen, N_2, yang relevan dengan pembelajaran kinetika dan ikatan kimia. Dokumen software kimia komputasi menyebut contoh penggunaan ORCA untuk memodelkan gas nitrogen: mahasiswa dapat melakukan optimasi geometri, menghitung single point energy, mengkaji celah HOMO-LUMO, dan menilai kekuatan ikatan rangkap tiga N≡N; pada tingkat lanjut, mahasiswa dapat mencari keadaan transisi untuk pemutusan ikatan nitrogen. Contoh ini menunjukkan bahwa jenis perhitungan tidak berdiri sendiri. Optimasi geometri memberi struktur, single point memberi energi, HOMO-LUMO memberi gambaran elektronik, dan pencarian keadaan transisi memberi gambaran kinetika. Dalam konteks pendidikan, alur ini membantu mahasiswa memahami mengapa Proses Haber-Bosch memerlukan kondisi keras dan katalis.

Aplikasi jenis perhitungan ini meluas ke berbagai bidang. Dalam kimia organik, optimasi geometri dan HOMO-LUMO membantu memahami reaktivitas gugus fungsi. Dalam kimia anorganik, momen dipol, spin, dan struktur elektronik membantu menjelaskan kompleks logam. Dalam kimia fisik, frekuensi vibrasi dan termodinamika memperkuat pemahaman spektrum dan energi bebas. Dalam kimia analitik, spektrum komputasi membantu interpretasi data eksperimen. Dalam farmasi, muatan parsial dan distribusi elektrostatik membantu memahami interaksi molekul obat. Dalam material, gap energi dan struktur elektronik membantu memprediksi sifat semikonduktor. Dalam pendidikan kimia, semua jenis perhitungan tersebut menjadi alat visualisasi dan inkuiri.

Implikasi metodologisnya adalah bahwa laporan praktikum harus mencerminkan jenis perhitungan yang dilakukan. Untuk optimasi geometri, laporan harus memuat struktur awal, metode, basis set, energi akhir, panjang ikatan, sudut ikatan, dan status konvergensi. Untuk single point energy, laporan harus memuat geometri yang digunakan dan alasan metode. Untuk frekuensi, laporan harus memuat frekuensi imajiner atau ketiadaannya, puncak utama, intensitas, dan faktor skala jika dipakai. Untuk HOMO-LUMO, laporan harus memuat energi HOMO, energi LUMO, gap, gambar orbital, dan interpretasi reaktivitas. Untuk momen dipol, laporan harus memuat nilai dipol dan arah vektor jika tersedia. Untuk muatan atom, laporan harus menyebut metode analisis muatan. Untuk energi reaksi, laporan harus memuat persamaan reaksi, energi semua spesies, satuan, koreksi termal, dan perhitungan ΔE atau ΔG.

Dalam praktik pengajaran, dosen perlu menegaskan bahwa satu jenis perhitungan tidak otomatis menjawab semua pertanyaan. Optimasi geometri tidak otomatis memberikan termodinamika lengkap. Single point energy tidak membuktikan struktur minimum. Frekuensi IR tidak otomatis menjelaskan reaktivitas. HOMO-LUMO tidak cukup untuk menyimpulkan mekanisme reaksi secara lengkap. Muatan atom tidak boleh digunakan sebagai satu-satunya dasar klaim polaritas atau reaktivitas. Energi reaksi tidak boleh ditafsirkan sebagai laju reaksi; laju reaksi memerlukan energi aktivasi dan analisis keadaan transisi. Pembedaan ini penting agar mahasiswa tidak membuat klaim berlebihan dari data yang terbatas.

Dari sisi filsafat ilmu, jenis-jenis perhitungan dalam praktikum kimia kuantum mengajarkan bahwa sains bekerja melalui model bertingkat. Setiap jenis perhitungan menjawab aspek tertentu

dari realitas molekuler. Struktur menjawab susunan atom. Energi menjawab kestabilan relatif. Frekuensi menjawab respons vibrasi. Orbital menjawab distribusi keadaan elektronik. Muatan menjawab partisi kerapatan elektron menurut model tertentu. Momen dipol menjawab pemisahan muatan. Energi reaksi menjawab perubahan kestabilan antara keadaan awal dan akhir. Tidak ada satu angka yang menjelaskan seluruh kimia molekul. Pemahaman muncul dari sintesis berbagai jenis data.

Dengan demikian, jenis perhitungan dalam praktikum kimia kuantum harus dipahami sebagai instrumen ilmiah yang memiliki tujuan, dasar teori, prosedur, output, dan batas validitas masing-masing. Mahasiswa perlu belajar memilih jenis perhitungan sesuai pertanyaan ilmiah, bukan sekadar mengikuti menu software. Mereka harus memahami mengapa suatu job dilakukan, data apa yang dihasilkan, bagaimana data divalidasi, dan bagaimana hasilnya dihubungkan dengan konsep kimia. Jika kompetensi ini terbentuk, praktikum kimia kuantum akan benar-benar menjadi sarana pembelajaran ilmiah yang mendalam, bukan sekadar latihan menjalankan program.

Subbab berikutnya akan membahas format file dalam kimia komputasi, meliputi XYZ, MOL, SDF, PDB, input GAMESS, input ORCA, input Gaussian, file log, dan output visualisasi. Pembahasan tersebut penting karena setiap jenis perhitungan yang telah dijelaskan pada subbab ini selalu memerlukan format data yang benar. Kesalahan format file dapat menyebabkan perhitungan gagal, hasil keliru, atau output sulit dianalisis.

1.5 Format File dalam Kimia Komputasi

Format file dalam kimia komputasi adalah standar penulisan data digital yang digunakan untuk menyimpan, memindahkan, membaca, memvisualisasikan, dan menjalankan informasi molekul pada perangkat lunak komputasi. Dalam praktikum kimia kuantum, format file bukan sekadar persoalan teknis administratif. Format file menentukan apakah struktur molekul dapat dibaca dengan benar, apakah ikatan dikenali, apakah koordinat atom tepat, apakah muatan dan multiplisitas tercatat, apakah metode komputasi dipahami oleh program, dan apakah output dapat dianalisis ulang. Kesalahan format file dapat menyebabkan perhitungan gagal, struktur molekul terbaca keliru, ikatan hilang, muatan salah, atau hasil tidak dapat direproduksi.

Dalam ekosistem kimia komputasi, file berfungsi sebagai bahasa penghubung antara manusia, perangkat lunak visualisasi, mesin komputasi, dan alat analisis data. Mahasiswa dapat menggambar molekul melalui Avogadro, Jmol, JSmol, GaussView, Gabedit, ChemCompute, atau WebMO, tetapi struktur grafis itu harus diterjemahkan ke dalam format data agar dapat dihitung oleh ORCA, GAMESS, Gaussian, Psi4, PySCF, NWChem, atau program lain. Dokumen rujukan software kimia komputasi menegaskan bahwa penyediaan file input merupakan jembatan kritis yang menghubungkan laboratorium visual tiga dimensi dengan mesin komputasi mekanika kuantum seperti ORCA, Gaussian, dan GAMESS; kesalahan pada tahap ini dapat menimbulkan kegagalan logis atau penghentian perhitungan secara prematur.

Secara konseptual, format file dapat dibagi menjadi tiga kelompok besar. Pertama, **file struktur molekul**, yaitu file yang menyimpan atom, koordinat, dan kadang informasi ikatan, seperti XYZ, MOL, SDF, dan PDB. Kedua, **file input komputasi**, yaitu file yang memuat struktur molekul sekaligus perintah perhitungan, metode, basis set, charge, multiplicity, memori, jumlah prosesor,

dan jenis pekerjaan, seperti input GAMESS, input ORCA, input Gaussian, input Psi4, atau input NWChem. Ketiga, **file output**, yaitu file hasil perhitungan yang memuat energi, status konvergensi, geometri akhir, frekuensi vibrasi, orbital, momen dipol, muatan atom, dan pesan teknis lain. Ketiga jenis file ini harus dipahami karena praktikum kimia kuantum selalu bergerak dari struktur awal menuju input, dari input menuju output, dan dari output menuju analisis.

Latar historis format file kimia komputasi berkaitan dengan perkembangan pemodelan molekul digital. Pada awal perkembangan komputasi kimia, struktur molekul banyak ditulis secara manual dalam bentuk koordinat internal, Z-matrix, atau koordinat Cartesian. Ketika perangkat lunak visualisasi berkembang, mahasiswa dan peneliti dapat menggambar molekul secara grafis, kemudian menyimpannya dalam format yang dapat ditukar antarprogram. Perkembangan ini memudahkan pembelajaran, tetapi juga menimbulkan tantangan baru, yaitu interoperabilitas. Satu software dapat menyimpan data ikatan, sementara software lain hanya membaca koordinat. Satu format dapat menyimpan muatan formal, sementara format lain tidak. Satu file dapat cocok untuk visualisasi, tetapi belum cukup untuk perhitungan kuantum. Oleh sebab itu, mahasiswa harus memahami jenis informasi apa yang disimpan oleh setiap format.

Format pertama yang paling sederhana adalah **XYZ**. File XYZ adalah format koordinat Cartesian yang menyimpan daftar atom dan posisinya dalam ruang tiga dimensi. Struktur dasarnya sangat sederhana: baris pertama berisi jumlah atom, baris kedua biasanya berisi komentar atau nama molekul, dan baris berikutnya berisi simbol atom serta koordinat X, Y, dan Z. Format ini banyak digunakan karena mudah dibaca manusia, mudah dibuat dengan Python, dan mudah dipindahkan antarprogram. Dokumen rujukan software kimia komputasi menjelaskan bahwa format XYZ adalah format koordinat Cartesian paling primitif; ia menyimpan jumlah atom pada baris pertama dan matriks koordinat X, Y, Z untuk setiap unsur, tetapi tidak menyimpan orde ikatan maupun muatan formal molekul.

Contoh sederhana file XYZ untuk molekul air adalah sebagai berikut:

```
3
water molecule
O     0.000000    0.000000    0.000000
H     0.758602    0.000000    0.504284
H    -0.758602    0.000000    0.504284
```

Contoh ini tampak sederhana, tetapi mengandung informasi penting. Angka **3** menunjukkan jumlah atom. Baris kedua adalah komentar. Baris berikutnya menunjukkan atom O dan dua atom H beserta koordinatnya. Jika file ini dibaca oleh program visualisasi, molekul air akan dapat ditampilkan. Jika dibaca oleh program analisis Python, panjang ikatan O–H dan sudut H–O–H dapat dihitung. Namun, file ini tidak memberi tahu secara eksplisit bahwa atom O terikat pada dua atom H. Program visualisasi biasanya menebak ikatan berdasarkan jarak antaratom dan radius kovalen. Untuk molekul sederhana, tebakan ini sering benar. Untuk kompleks logam, ion, keadaan transisi, radikal, atau struktur tidak biasa, tebakan ikatan dapat keliru.

Kelebihan utama XYZ adalah kesederhanaan dan transparansi. Mahasiswa dapat membuka file XYZ dengan Notepad, VS Code, atau editor teks biasa. Mereka dapat melihat langsung atom dan koordinatnya. Mereka dapat membuat atau memodifikasi file ini dengan Python. Dalam praktikum

optimasi geometri, format XYZ sangat berguna untuk menyimpan struktur awal dan struktur akhir. Dalam praktikum Python, format ini ideal untuk latihan membaca koordinat, menghitung jarak, menghitung sudut, membuat visualisasi 3D sederhana, dan membandingkan geometri awal dengan geometri hasil optimasi. Namun, kelemahannya juga harus ditekankan: XYZ tidak cukup untuk menyimpan semua informasi kimia, terutama konektivitas, orde ikatan, muatan formal, stereokimia, dan metadata perhitungan.

Format kedua adalah **MOL** atau **MDL Molfile**. Berbeda dari XYZ, file MOL menyimpan informasi yang lebih kaya. File ini tidak hanya memuat koordinat atom, tetapi juga tabel konektivitas yang menjelaskan atom mana berikatan dengan atom mana dan jenis ikatan apa yang menghubungkannya. Dokumen rujukan menjelaskan bahwa file MOL menyimpan tabel konektivitas, koordinat ruang, serta definisi eksplisit atom yang berikatan dan jenis ikatannya; hal ini penting untuk visualisasi awal dan transfer data antarperangkat lunak. Dengan demikian, MOL lebih dekat dengan representasi struktur kimia yang biasa dipahami mahasiswa: atom, ikatan, dan orde ikatan.

Dalam praktikum kimia kuantum, file MOL berguna ketika struktur kimia perlu dipindahkan dari satu program ke program lain tanpa kehilangan informasi ikatan. Misalnya, mahasiswa menggambar etanol atau benzena di Avogadro, menyimpannya sebagai MOL, lalu membukanya di program lain untuk memastikan ikatan tunggal, rangkap, atau aromatik terbaca benar. Untuk molekul organik kecil, format ini sangat membantu karena konektivitasnya eksplisit. Namun, MOL juga memiliki keterbatasan, terutama untuk sistem yang tidak mudah direpresentasikan dengan ikatan valensi klasik. Kompleks logam transisi, struktur resonansi, interaksi lemah, keadaan transisi, dan sistem delokalisasi tertentu dapat lebih sulit direpresentasikan secara memadai melalui tabel ikatan sederhana.

Format ketiga adalah **SDF** atau **Structure Data File**. SDF dapat dipahami sebagai perluasan dari format MOL yang mampu menyimpan banyak molekul dalam satu file beserta data tambahan. Format ini banyak digunakan dalam basis data kimia, kimia obat, cheminformatics, dan virtual screening. Satu file SDF dapat memuat ratusan atau ribuan struktur molekul, masing-masing dengan properti seperti nama senyawa, massa molekul, logP, aktivitas biologis, ID basis data, atau data hasil perhitungan. Dalam praktikum kimia kuantum dasar, SDF tidak selalu menjadi format utama untuk perhitungan ab initio, tetapi sangat berguna ketika mahasiswa bekerja dengan banyak molekul sekaligus atau ingin menghubungkan kimia kuantum dengan kimia informatika.

Aplikasi SDF dalam buku praktikum ini dapat diarahkan pada proyek mini. Misalnya, mahasiswa dapat mengambil beberapa molekul turunan benzena dari basis data, menyimpannya sebagai SDF, mengonversinya ke XYZ, lalu menjalankan optimasi geometri dan analisis HOMO-LUMO. Dengan cara ini, mahasiswa belajar bahwa format file bukan hanya alat menyimpan molekul tunggal, tetapi juga dapat menjadi wadah dataset molekuler. Dalam era kimia data dan machine learning, kemampuan membaca dan mengolah SDF menjadi semakin relevan. Namun, mahasiswa harus tetap berhati-hati karena struktur dalam SDF dari basis data belum tentu sudah memiliki geometri tiga dimensi yang teroptimasi secara kuantum. Banyak struktur awal perlu ditambahkan hidrogen, dibersihkan, dan dioptimasi sebelum digunakan dalam perhitungan serius.

Format keempat adalah **PDB** atau **Protein Data Bank format**. Format PDB awalnya dikembangkan untuk menyimpan struktur biomolekul seperti protein, DNA, RNA, dan kompleks makromolekul. PDB menyimpan informasi atom, residu, rantai, nomor atom, koordinat, faktor suhu, dan kadang informasi konektivitas. Format ini sangat penting dalam biokimia komputasi, docking molekuler, dinamika molekul, dan pemodelan biomolekul. Walaupun buku praktikum kimia kuantum ini lebih berfokus pada molekul kecil, PDB tetap perlu dikenalkan karena banyak mahasiswa kimia, farmasi, biokimia, dan pendidikan kimia akan bertemu struktur protein atau biomolekul dalam penelitian lanjut.

Dalam praktikum kimia kuantum dasar, PDB dapat digunakan sebagai jembatan menuju sistem besar. Misalnya, mahasiswa dapat membuka struktur protein kecil atau ligan dalam situs aktif enzim, lalu mengekstrak fragmen molekul kecil untuk perhitungan kuantum. Namun, PDB juga memiliki tantangan. Struktur PDB hasil kristalografi sinar-X sering tidak menyertakan semua atom hidrogen. Keadaan protonasi residu dapat ambigu. Muatan sistem tidak selalu jelas. Jika mahasiswa langsung mengambil fragmen dari PDB tanpa memeriksa hidrogen, protonasi, dan muatan, perhitungan kuantum dapat menghasilkan data yang keliru. Karena itu, PDB harus diperlakukan sebagai titik awal struktur, bukan input final yang selalu siap dihitung.

Setelah memahami file struktur, mahasiswa perlu memahami **file input komputasi**. File input adalah file yang memberi instruksi kepada mesin komputasi. Di sinilah perbedaan antara visualisasi molekul dan perhitungan kuantum menjadi jelas. Struktur 3D saja tidak cukup. Program harus diberi tahu pekerjaan apa yang dilakukan: optimasi geometri, single point energy, frekuensi vibrasi, analisis orbital, atau perhitungan lain. Program juga harus diberi tahu metode, basis set, muatan total, multiplisitas spin, alokasi memori, jumlah prosesor, serta kadang detail teknis lain. Dokumen rujukan menyebut bahwa format input seperti .inp atau .gjf merupakan format hibrida yang memuat koordinat Cartesian atau Z-matrix sekaligus blok perintah atau kata kunci mekanika kuantum, seperti basis set, batas iterasi, dan alokasi memori.

Salah satu file input yang penting dalam praktikum adalah **input GAMESS**. GAMESS menggunakan struktur input berbasis grup atau blok, biasanya diawali dengan tanda dolar seperti `$CONTRL`, `$SYSTEM`, `$BASIS`, `$DATA`, dan diakhiri dengan `$END`. Dalam konteks ChemCompute, mahasiswa dapat menjalankan GAMESS melalui antarmuka web tanpa harus menginstal program secara lokal. Dokumen rujukan menjelaskan bahwa ChemCompute memungkinkan mahasiswa sarjana menjalankan eksperimen menggunakan paket komputasi seperti GAMESS dan NWChem tanpa instalasi Linux yang rumit; mahasiswa cukup menggambar molekul, memilih metode seperti B3LYP/6-31G*, mengirim job, dan hasil dikembalikan melalui peramban web.

Contoh konseptual input GAMESS untuk molekul air dapat ditulis sebagai berikut:

```
 $CONTRL SCFTYP=RHF RUNTYP=OPTIMIZE DFTTYP=B3LYP $END
 $BASIS GBASIS=N31 NGAUSS=6 NDFUNC=1 $END
 $DATA
Water molecule
C1
O  8.0   0.000000   0.000000   0.000000
H  1.0   0.758602   0.000000   0.504284
H  1.0  -0.758602   0.000000   0.504284
 $END
```

Contoh ini menunjukkan beberapa hal penting. Baris `$CONTRL` menentukan jenis perhitungan, tipe SCF, dan fungsional DFT. Baris `$BASIS` menentukan basis set. Blok `$DATA` memuat nama molekul, simetri, dan koordinat atom. Mahasiswa harus memahami bahwa perubahan kecil pada input dapat mengubah jenis perhitungan secara drastis. Jika `RUNTYP=OPTIMIZE`, program melakukan optimasi geometri. Jika diganti menjadi `RUNTYP=ENERGY`, program melakukan single point energy. Jika diganti menjadi `RUNTYP=HESSIAN`, program dapat digunakan untuk analisis frekuensi. Dengan demikian, input adalah representasi eksplisit dari keputusan ilmiah.

Format input kedua yang penting adalah **input ORCA**. ORCA banyak digunakan dalam pendidikan dan riset karena sintaksnya relatif ringkas. Input ORCA biasanya memuat baris perintah yang diawali tanda seru, misalnya `! B3LYP def2-SVP Opt Freq`, kemudian blok koordinat yang diawali `* xyz charge multiplicity`. Contoh input ORCA untuk air adalah:

```
! B3LYP def2-SVP Opt Freq

* xyz 0 1
O    0.000000    0.000000    0.000000
H    0.758602    0.000000    0.504284
H   -0.758602    0.000000    0.504284
*
```

Dalam contoh ini, `B3LYP` adalah fungsional DFT, `def2-SVP` adalah basis set, `Opt` berarti optimasi geometri, dan `Freq` berarti perhitungan frekuensi. Bagian `* xyz 0 1` menunjukkan bahwa koordinat ditulis dalam format XYZ, muatan total molekul adalah 0, dan multiplisitas spin adalah 1. Kesalahan menulis charge atau multiplicity dapat membuat hasil keliru walaupun perhitungan selesai. Misalnya, molekul air netral harus memiliki charge 0 dan multiplicity 1. Jika mahasiswa salah memasukkan multiplicity 2, program akan memperlakukan sistem sebagai radikal doublet, yang tidak sesuai untuk air keadaan dasar.

Keunggulan input ORCA bagi praktikum adalah keterbacaannya. Mahasiswa dapat melihat dengan cepat hubungan antara kata kunci dan jenis perhitungan. Namun, kesederhanaan ini tidak boleh membuat mahasiswa mengabaikan detail. ORCA memiliki banyak opsi tambahan untuk SCF, grid DFT, dispersi, pelarut, optimasi, dan analisis spektrum. Pada praktikum dasar, input sebaiknya dibuat sederhana terlebih dahulu. Setelah mahasiswa memahami struktur dasar input, mereka dapat belajar menambahkan opsi seperti `TightSCF`, `Grid5`, `D3BJ`, `CPCM(water)`, atau pengaturan paralel. Penambahan opsi harus selalu disertai alasan ilmiah, bukan sekadar meniru input dari internet.

Format input ketiga adalah **input Gaussian**, yang biasanya menggunakan ekstensi `.gjf` atau `.com`. Struktur input Gaussian umumnya terdiri atas bagian alokasi sumber daya, route section, judul, charge dan multiplicity, serta koordinat molekul. Contoh sederhana input Gaussian untuk optimasi dan frekuensi air adalah:

```
%chk=water.chk
%nprocshared=4
%mem=2GB
# B3LYP/6-31G(d) Opt Freq
```

```
Water molecule optimization and frequency

0 1
O     0.000000     0.000000     0.000000
H     0.758602     0.000000     0.504284
H    -0.758602     0.000000     0.504284
```

Pada contoh ini, `%chk` menentukan file checkpoint, `%nprocshared` menentukan jumlah prosesor, `%mem` menentukan memori, dan baris `# B3LYP/6-31G(d) Opt Freq` menentukan metode, basis set, dan jenis pekerjaan. Setelah judul, baris `0 1` menunjukkan charge dan multiplicity. Input Gaussian memperlihatkan bahwa file input tidak hanya memuat molekul, tetapi juga pengaturan komputasi. Dalam praktikum, mahasiswa perlu membaca bagian ini dengan cermat karena banyak kesalahan perhitungan berasal dari route section yang tidak sesuai, muatan yang keliru, atau koordinat yang tidak rapi.

Selain koordinat Cartesian, beberapa input menggunakan **Z-matrix** atau koordinat internal. Z-matrix menyatakan struktur molekul melalui panjang ikatan, sudut ikatan, dan sudut dihedral. Format ini historis dan sangat penting dalam optimasi geometri, pemindaian koordinat reaksi, serta pembelajaran struktur molekul. Sumber kimia kuantum yang digunakan juga memuat latihan penggunaan Z-matrix atau koordinat internal untuk menyusun input perhitungan geometri. Z-matrix membantu mahasiswa memahami bahwa geometri molekul tidak hanya dapat direpresentasikan sebagai koordinat X, Y, Z, tetapi juga sebagai relasi internal antaratom.

Contoh konseptual Z-matrix air adalah:

```
O
H 1 rOH
H 1 rOH 2 angleHOH

rOH=0.96
angleHOH=104.5
```

Representasi ini menyatakan bahwa atom H pertama berjarak `rOH` dari atom O, sedangkan atom H kedua berjarak `rOH` dari atom O dan membentuk sudut `angleHOH` terhadap H pertama. Z-matrix sangat baik untuk praktikum pemindaian sudut atau panjang ikatan karena parameter seperti `rOH` atau `angleHOH` dapat diubah secara sistematis. Namun, untuk molekul besar, Z-matrix dapat menjadi rumit dan rentan kesalahan urutan atom. Karena itu, koordinat Cartesian lebih umum digunakan dalam banyak workflow modern, sedangkan Z-matrix tetap penting untuk pemahaman struktur internal.

Setelah file input dijalankan, program menghasilkan **file output**. File output adalah sumber data utama praktikum. Ia dapat berisi ratusan hingga ribuan baris, tergantung jenis perhitungan. Output biasanya mencantumkan ringkasan input, informasi metode, basis set, jumlah elektron, charge, multiplicity, energi SCF, status konvergensi, geometri awal dan akhir, gradien, frekuensi vibrasi, intensitas IR, momen dipol, populasi muatan, orbital molekul, serta pesan peringatan atau error. Mahasiswa harus belajar membaca output secara selektif dan kritis. Output bukan dokumen yang hanya disalin, melainkan data ilmiah yang harus ditambang.

Bagian terpenting dari output untuk praktikum dasar biasanya mencakup enam informasi. Pertama, apakah perhitungan selesai normal. Kedua, apakah SCF konvergen. Ketiga, apakah optimasi geometri konvergen jika job berupa optimasi. Keempat, berapa energi total akhir. Kelima, bagaimana geometri akhir. Keenam, apakah ada frekuensi imajiner jika dilakukan perhitungan frekuensi. Sumber kimia kuantum yang digunakan menekankan bahwa dokumentasi input dan output sangat penting; setiap praktikum harus menyertakan struktur awal, muatan, multiplicity, metode, basis set, kriteria konvergensi, serta file output yang dapat diperiksa ulang, karena tanpa dokumentasi tersebut hasil sulit diverifikasi.

Dalam konteks laporan, file output harus diperlakukan sebagai bukti primer. Mahasiswa tidak cukup menulis "hasil optimasi berhasil". Mereka harus menunjukkan indikator keberhasilan, misalnya pesan konvergensi, energi akhir, kriteria gradien, dan ketiadaan frekuensi imajiner. Jika menggunakan Gaussian, mahasiswa dapat mencari frasa seperti "Normal termination" atau bagian frekuensi. Jika menggunakan ORCA, mahasiswa dapat mencari "ORCA TERMINATED NORMALLY", energi akhir, dan bagian vibrational frequencies. Jika menggunakan GAMESS, mahasiswa dapat mencari bagian energi total dan status normal termination. Setiap program memiliki format output berbeda, tetapi prinsip validasinya sama: perhitungan harus selesai, konvergen, dan konsisten dengan tujuan job.

File output juga menjadi bahan utama untuk **parsing dengan Python**. Pada tahap lanjut, mahasiswa dapat menulis kode untuk membaca file log dan mengekstrak angka tertentu. Misalnya, Python dapat digunakan untuk mencari baris yang memuat energi total, mengambil koordinat akhir, membaca daftar frekuensi, atau menyusun tabel energi beberapa molekul. Hal ini penting karena analisis manual rentan salah salin, terutama jika jumlah data banyak. Dalam praktikum Hartree–Fock dan basis set, mahasiswa mungkin memiliki lima output energi untuk satu molekul. Dalam praktikum energi reaksi, mahasiswa mungkin memiliki output untuk beberapa reaktan, produk, dan pelarut. Tanpa otomasi, peluang kesalahan meningkat. Dengan parsing Python, alur analisis menjadi lebih transparan dan dapat diulang.

Selain input dan output utama, beberapa program menghasilkan **file visualisasi**. File visualisasi dapat berupa file orbital, cube file, molden file, checkpoint file, atau file lain yang dibaca oleh program visualisasi. File semacam ini digunakan untuk menampilkan orbital HOMO-LUMO, rapat elektron, permukaan potensial elektrostatik, mode vibrasi, atau peta densitas. Dalam praktikum, output visual sangat penting karena membantu mahasiswa menghubungkan angka dengan bentuk molekul. Namun, visualisasi harus dipahami sebagai representasi data, bukan gambar artistik bebas. Warna orbital menunjukkan fase fungsi gelombang, bukan muatan positif-negatif secara langsung. Permukaan potensial elektrostatik menunjukkan daerah kaya atau miskin elektron relatif, bukan peta moral "baik-buruk" reaktivitas. Mode vibrasi menunjukkan pola perpindahan atom, bukan gerak molekul dalam arti klasik sederhana.

Dokumen software kimia komputasi juga menekankan pentingnya visualisasi sifat molekul seperti MEP atau Molecular Electrostatic Potential. Visualisasi MEP dapat mengubah cara mahasiswa memprediksi mekanisme reaksi organik karena polaritas molekul menjadi bukti empiris yang tampak melalui pemodelan in silico. Dalam konteks praktikum, visualisasi MEP dapat digunakan untuk memahami mengapa bagian tertentu dari molekul lebih rentan terhadap serangan nukleofil

atau elektrofil. Namun, mahasiswa harus tetap mengaitkan gambar MEP dengan data kuantitatif seperti muatan atom, momen dipol, orbital frontier, dan konteks reaksi.

Format file juga berkaitan erat dengan **interoperabilitas**. Interoperabilitas adalah kemampuan data molekul berpindah dari satu perangkat lunak ke perangkat lunak lain tanpa kehilangan informasi penting. Avogadro dapat digunakan untuk menggambar molekul dan menghasilkan input ORCA atau Gaussian. ChemCompute dapat digunakan untuk menggambar molekul melalui antarmuka web dan menjalankan perhitungan berbasis paket seperti GAMESS. Jmol atau JSmol dapat digunakan untuk visualisasi molekul di peramban. Python dapat digunakan untuk membaca XYZ, CSV, atau file log. Dokumen rujukan menyebut bahwa Avogadro memiliki keunggulan pedagogis sebagai "penerjemah" melalui menu ekstensi, misalnya untuk menghasilkan input ORCA secara otomatis dari struktur molekul.

Namun, interoperabilitas tidak selalu sempurna. Ketika file dikonversi dari satu format ke format lain, sebagian informasi dapat hilang. Jika SDF dikonversi ke XYZ, informasi konektivitas dan properti tambahan mungkin hilang. Jika PDB dikonversi ke XYZ, informasi residu dan rantai hilang. Jika MOL dibaca oleh program yang tidak mendukung aromatic bond dengan baik, ikatan aromatik dapat berubah menjadi pola tunggal-rangkap tertentu. Jika file input dibuat otomatis dari software visualisasi, charge dan multiplicity mungkin masih harus diperiksa manual. Oleh karena itu, mahasiswa harus selalu memeriksa file setelah konversi. Prinsipnya sederhana: jangan percaya bahwa konversi file selalu benar tanpa verifikasi.

Kesalahan format file dalam praktikum kimia kuantum dapat dikelompokkan menjadi beberapa jenis. Pertama, kesalahan jumlah atom pada file XYZ. Jika baris pertama menyatakan 10 atom, tetapi daftar atom hanya 9 atau 11, file dapat gagal dibaca. Kedua, kesalahan simbol atom, misalnya menulis "CL" bukan "Cl" untuk klorin. Ketiga, kesalahan satuan koordinat, misalnya mencampur Ångström dan Bohr tanpa menyadari perbedaannya. Keempat, kesalahan charge dan multiplicity. Kelima, kesalahan basis set atau metode yang tidak dikenali program. Keenam, kesalahan format blok input, misalnya lupa menutup `$END` pada GAMESS atau lupa tanda `*` penutup pada ORCA. Ketujuh, kesalahan struktur awal, misalnya atom terlalu berdekatan sehingga SCF sulit konvergen.

Dampak kesalahan format tidak selalu langsung terlihat. Kadang program menolak berjalan dan memberikan pesan error. Ini relatif mudah diperbaiki. Yang lebih berbahaya adalah ketika program tetap berjalan tetapi inputnya salah secara kimia. Misalnya, molekul radikal dimasukkan sebagai singlet, ion positif dimasukkan sebagai netral, atau struktur dengan atom H hilang tetap dihitung. Dalam kasus seperti ini, output mungkin tampak normal, tetapi kesimpulan ilmiahnya salah. Oleh sebab itu, pemeriksaan format file harus menjadi bagian dari etika dan disiplin praktikum. Mahasiswa harus memeriksa struktur visual, jumlah atom, muatan, multiplisitas, metode, basis set, dan jenis job sebelum mengirim perhitungan.

Dalam praktikum berbasis ChemCompute, manajemen file memiliki karakter khusus. Platform web membantu mahasiswa menghindari instalasi rumit dan mengurangi kebutuhan perangkat keras lokal. Dokumen rujukan menjelaskan bahwa ChemCompute memindahkan beban komputasi dari komputer lokal ke server jarak jauh yang diakses melalui peramban web, sehingga mahasiswa dapat melakukan eksplorasi molekuler tanpa batasan perangkat keras yang berat. Namun,

penggunaan platform web menuntut kedisiplinan dalam menyimpan input dan output. Mahasiswa harus mengunduh file hasil, memberi nama file secara sistematis, menyimpan tangkapan layar bila diperlukan, dan memastikan data tidak hilang setelah sesi web berakhir.

Nama file juga merupakan bagian penting dari manajemen praktikum. Nama file yang baik harus informatif, ringkas, dan konsisten. Misalnya:

```
h2o_b3lyp_631gd_optfreq.inp
h2o_b3lyp_631gd_optfreq.log
h2o_b3lyp_631gd_final.xyz
h2o_b3lyp_631gd_ir.csv
```

Nama seperti ini memberi informasi molekul, metode, basis set, jenis job, dan jenis file. Sebaliknya, nama seperti `hasil1.log`, `coba_final_baru_banget.log`, atau `praktikum_fix_final2.log` tidak memenuhi standar akademik karena sulit ditelusuri. Dalam penelitian komputasi, penamaan file yang buruk dapat menyebabkan kebingungan, salah analisis, dan hilangnya reprodusibilitas. Praktikum harus melatih mahasiswa sejak awal agar terbiasa dengan manajemen data ilmiah.

Struktur folder juga perlu ditata. Untuk setiap mata praktikum, mahasiswa sebaiknya membuat folder seperti:

```
praktikum_02_optimasi_geometri/
├── input/
├── output/
├── xyz/
├── python/
├── gambar/
├── tabel/
└── laporan/
```

Struktur ini membantu memisahkan file input, output, koordinat, kode Python, gambar orbital, tabel hasil, dan laporan. Dalam praktikum dengan banyak molekul atau banyak metode, struktur folder sangat membantu. Jika mahasiswa membandingkan H_2O, NH_3, CH_4, dan CO_2, masing-masing molekul dapat diberi subfolder. Jika mahasiswa membandingkan beberapa basis set, file dapat dinamai sesuai basis set. Kebiasaan sederhana ini memperkuat reprodusibilitas dan profesionalisme ilmiah.

Format file juga berkaitan dengan **satuan**. Koordinat dalam file XYZ umumnya dinyatakan dalam Ångström, tetapi beberapa program dapat menggunakan Bohr. Energi dapat muncul dalam Hartree, eV, kJ/mol, kcal/mol, atau cm^{-1} tergantung jenis data. Frekuensi vibrasi biasanya dinyatakan dalam cm^{-1}. Momen dipol biasanya dalam Debye. Muatan atom dalam satuan muatan elektron. Jika mahasiswa tidak memahami satuan, analisis menjadi keliru. Misalnya, energi total dalam Hartree tidak boleh langsung dimasukkan ke tabel kJ/mol tanpa konversi. Dalam Python, konversi satuan harus ditulis eksplisit agar pembaca laporan mengetahui bagaimana angka dihitung.

Dalam perspektif teori informasi, format file adalah bentuk representasi. Satu molekul nyata dapat direpresentasikan sebagai gambar 2D, model bola-tongkat 3D, koordinat XYZ, tabel konektivitas MOL, dataset SDF, struktur PDB, input ORCA, input GAMESS, atau output Gaussian. Setiap representasi memilih aspek tertentu dan mengabaikan aspek lain. Gambar 2D menonjolkan konektivitas, tetapi tidak selalu geometri 3D. XYZ menonjolkan koordinat, tetapi tidak menyimpan ikatan. MOL menyimpan ikatan, tetapi belum tentu cocok untuk keadaan elektronik kompleks. Input kuantum menyimpan instruksi metode, tetapi perlu program tertentu untuk dijalankan. Output menyimpan hasil, tetapi sering sulit dibaca tanpa pemahaman struktur file. Kesadaran representasional ini penting agar mahasiswa tidak menyamakan file dengan molekul itu sendiri. File adalah model digital dari molekul, bukan molekul fisik.

Implikasi akademik dari penguasaan format file sangat besar. Mahasiswa yang memahami format file dapat berpindah antarsoftware dengan lebih mandiri. Mereka dapat menggambar molekul di Avogadro, mengekspor XYZ, membuat input ORCA, menjalankan job, membaca output, mengekstrak koordinat akhir, memvisualisasikan orbital, lalu menganalisis data dengan Python. Mahasiswa yang tidak memahami format file akan sangat bergantung pada tombol otomatis dan mudah berhenti ketika terjadi error. Dengan demikian, literasi format file adalah bagian dari literasi komputasi kimia.

Implikasi praktisnya juga jelas dalam penyusunan laporan. Setiap laporan praktikum harus menyertakan file input dan output sebagai lampiran atau setidaknya ringkasan bagian pentingnya. Untuk praktikum berbasis Python, file kode juga harus dilampirkan. Jika hasil berupa grafik, sumber data grafik harus jelas. Jika hasil berupa tabel energi, file output asal energi harus dapat ditelusuri. Jika hasil berupa gambar orbital, file visualisasi atau metode pembuatannya harus dicatat. Ini sejalan dengan prinsip bahwa reprodusibilitas bergantung pada input lengkap, versi program, parameter komputasi, dan file output.

Dalam pembelajaran, dosen dapat menjadikan format file sebagai objek praktikum tersendiri. Mahasiswa dapat diberi tugas membandingkan informasi yang tersimpan dalam XYZ, MOL, SDF, dan PDB untuk molekul yang sama. Mereka dapat diminta membuka setiap file dengan editor teks dan menjelaskan bagian-bagiannya. Mereka dapat diminta mengonversi file dari MOL ke XYZ lalu memeriksa informasi apa yang hilang. Mereka dapat diminta membuat input ORCA atau GAMESS dari struktur sederhana dan menjelaskan arti setiap baris. Tugas semacam ini akan menumbuhkan pemahaman yang jauh lebih kuat dibanding hanya meminta mahasiswa mengunggah file ke software.

Sebagai contoh latihan, mahasiswa dapat memulai dari molekul etanol. Pertama, gambar etanol di Avogadro. Kedua, simpan sebagai `ethanol.mol` dan `ethanol.xyz`. Ketiga, buka kedua file dengan editor teks. Keempat, jelaskan perbedaan informasi yang tersimpan. Kelima, buat input ORCA untuk optimasi geometri. Keenam, jalankan perhitungan melalui ORCA lokal atau platform yang tersedia. Ketujuh, simpan output sebagai `ethanol_b3lyp_def2svp_opt.log`. Kedelapan, ekstrak koordinat akhir ke file XYZ. Kesembilan, gunakan Python untuk menghitung panjang ikatan C–C, C–O, O–H, serta beberapa sudut ikatan. Latihan ini menyatukan literasi struktur, input, output, dan analisis.

Contoh lain dapat menggunakan ChemCompute. Mahasiswa menggambar air atau amonia pada antarmuka ChemCompute, memilih metode dan basis set, lalu menjalankan job. Setelah selesai, mereka mengunduh output dan menyimpannya dengan nama sistematis. Kemudian mereka membaca output untuk mencari energi akhir, geometri akhir, dan jika tersedia momen dipol atau frekuensi. Selanjutnya, mereka memasukkan data tersebut ke Python untuk membuat tabel. Dengan cara ini, ChemCompute tidak hanya menjadi tombol submit, tetapi bagian dari alur data ilmiah yang dapat diperiksa.

Analisis kritis terhadap format file juga harus memasukkan aspek keamanan dan keberlanjutan data. File yang disimpan di komputer pribadi, flashdisk, atau platform web dapat hilang jika tidak dicadangkan. File yang diberi nama tidak jelas dapat tertukar. File output besar dapat sulit dibuka dengan editor teks biasa. File dari software komersial mungkin memiliki format biner tertentu yang tidak mudah dibaca oleh program lain. Oleh karena itu, praktikum perlu mendorong penggunaan format terbuka jika memungkinkan, seperti XYZ, CSV, TXT, atau format log teks. File biner tetap boleh digunakan, tetapi harus didampingi file teks atau data ringkas yang dapat diperiksa.

Dalam kaitannya dengan Python, format file terbuka sangat penting. Python dapat dengan mudah membaca XYZ, CSV, TXT, dan banyak log output karena berbasis teks. Untuk format yang lebih kompleks, pustaka seperti ASE, RDKit, cclib, Open Babel, atau pandas dapat digunakan jika tersedia. Namun, pada tahap dasar, mahasiswa sebaiknya belajar membaca format sederhana terlebih dahulu. Misalnya, membaca file XYZ dengan fungsi `open()`, memisahkan baris, mengambil simbol atom dan koordinat, lalu menghitung jarak. Setelah itu, mahasiswa dapat naik ke parsing output yang lebih rumit. Pendekatan bertahap ini membuat programming ilmiah lebih mudah dipahami.

Format file dalam kimia komputasi akhirnya dapat dipahami sebagai fondasi teknis bagi seluruh praktikum. Tanpa file yang benar, perhitungan tidak dapat berjalan. Tanpa input yang terdokumentasi, hasil tidak dapat diverifikasi. Tanpa output yang tersimpan, analisis tidak dapat diperiksa. Tanpa format data yang rapi, Python tidak dapat digunakan secara efektif. Tanpa manajemen file yang baik, laporan praktikum menjadi lemah. Karena itu, mahasiswa harus memperlakukan file bukan sebagai lampiran sekunder, tetapi sebagai bagian inti dari praktik ilmiah.

Subbab berikutnya akan membahas prinsip reprodusibilitas praktikum. Pembahasan tersebut merupakan kelanjutan langsung dari format file karena reprodusibilitas hanya mungkin dicapai jika struktur awal, metode, basis set, charge, multiplicity, software, versi program, parameter, input, dan output dicatat secara lengkap serta dapat diperiksa ulang.

1.6 Prinsip Reprodusibilitas Praktikum

Reprodusibilitas praktikum adalah kemampuan suatu hasil praktikum untuk diperiksa, diulang, dan diperoleh kembali oleh orang lain dengan menggunakan data, metode, parameter, perangkat lunak, dan prosedur yang sama atau setara. Dalam praktikum kimia kuantum, reprodusibilitas bukan sekadar kelengkapan administrasi laporan, melainkan inti dari keabsahan ilmiah. Hasil komputasi yang tidak dapat direproduksi tidak memiliki kekuatan akademik yang memadai, meskipun tampak rapi, memiliki grafik menarik, atau menghasilkan angka dengan banyak desimal.

Suatu energi total, panjang ikatan, sudut ikatan, frekuensi vibrasi, gap HOMO-LUMO, momen dipol, muatan atom, atau nilai ΔG hanya bernilai ilmiah jika orang lain dapat menelusuri bagaimana angka itu diperoleh.

Dalam praktikum kimia kuantum, reprodusibilitas menuntut pencatatan lengkap tentang **struktur awal, metode, basis set, charge, multiplicity, software, versi program, parameter komputasi, file input, dan file output**. Rujukan kimia kuantum yang menjadi dasar buku ini menegaskan bahwa praktikum komputasi harus menyertakan struktur awal, muatan, multiplicity, metode, basis set, kriteria konvergensi, serta file output yang dapat diperiksa ulang; tanpa dokumentasi input dan output, hasil komputasi sulit diverifikasi. Rumusan ringkasnya dapat dinyatakan sebagai berikut:

Reprodusibilitas = input lengkap + versi program + parameter komputasi + file output

Rumus sederhana ini memiliki konsekuensi luas. Ia berarti bahwa laporan praktikum tidak boleh hanya berisi hasil akhir. Laporan harus memuat jejak ilmiah yang memungkinkan dosen, mahasiswa lain, atau peneliti berikutnya memahami jalan yang ditempuh dari struktur awal sampai kesimpulan. Dalam kimia komputasi, angka bukan sekadar angka. Angka adalah hasil dari rangkaian keputusan metodologis. Perubahan satu keputusan kecil, misalnya mengganti basis set dari STO-3G menjadi 6-31G(d), mengubah charge dari 0 menjadi +1, atau mengganti multiplicity dari singlet menjadi doublet, dapat menghasilkan output yang sama sekali berbeda. Oleh karena itu, reprodusibilitas menuntut transparansi penuh terhadap seluruh keputusan tersebut.

Secara historis, isu reprodusibilitas menjadi semakin penting ketika sains memasuki era komputasi dan data besar. Dalam eksperimen klasik, peneliti biasanya mencatat bahan, alat, suhu, tekanan, konsentrasi, volume, waktu reaksi, dan prosedur kerja. Dalam eksperimen komputasi, unsur yang harus dicatat berubah bentuk menjadi file input, versi software, metode numerik, parameter konvergensi, basis set, model pelarut, grid integrasi, algoritma optimasi, dan kode analisis. Dengan kata lain, laboratorium komputasi juga membutuhkan buku catatan laboratorium, tetapi bentuknya lebih digital. File input, file output, script Python, struktur folder, dan metadata perhitungan adalah "catatan laboratorium" dalam praktikum kimia kuantum.

Reprodusibilitas juga berkaitan erat dengan konsep **traceability**, yaitu keterlacakan data dari kesimpulan akhir kembali ke sumber awalnya. Jika sebuah laporan menyatakan bahwa "molekul A lebih stabil daripada molekul B sebesar 18,5 kJ/mol", maka pembaca harus dapat menelusuri dari mana angka 18,5 kJ/mol berasal. Apakah angka itu berasal dari energi elektronik saja, atau sudah memuat zero-point energy? Apakah energi dihitung dalam Hartree lalu dikonversi ke kJ/mol? Apakah struktur A dan B dioptimasi dengan metode yang sama? Apakah keduanya memiliki jumlah atom yang sama? Apakah perbandingan dilakukan pada fase gas atau menggunakan model pelarut? Apakah ada frekuensi imajiner pada salah satu struktur? Tanpa jawaban atas pertanyaan ini, klaim stabilitas menjadi lemah.

Dalam kimia kuantum, sumber pertama reprodusibilitas adalah **struktur awal**. Struktur awal adalah koordinat molekul sebelum perhitungan dijalankan. Walaupun optimasi geometri bertujuan mencari struktur minimum, hasil optimasi sering bergantung pada struktur awal, terutama untuk molekul fleksibel, sistem konformer, kompleks logam, agregat ikatan hidrogen, atau keadaan transisi. Dua mahasiswa dapat menggambar molekul butana dengan torsi awal berbeda. Setelah

optimasi, satu dapat berakhir pada konformer anti, sedangkan yang lain pada konformer gauche. Keduanya mungkin sama-sama “konvergen”, tetapi tidak merepresentasikan struktur yang sama. Karena itu, struktur awal harus disimpan, diberi nama jelas, dan dilampirkan dalam format yang dapat diperiksa, misalnya XYZ, MOL, SDF, atau file input asli.

Struktur awal juga harus diperiksa secara kimia sebelum perhitungan. Mahasiswa tidak boleh menganggap struktur dari software visualisasi selalu benar. Atom hidrogen dapat kurang, valensi dapat tidak wajar, jarak atom dapat terlalu dekat, ikatan dapat salah terbaca, atau keadaan protonasi dapat tidak sesuai. Pada molekul organik sederhana, kesalahan ini mungkin mudah terlihat. Pada molekul ionik, radikal, kompleks logam, atau biomolekul, kesalahan lebih sulit dikenali. Reprodusibilitas bukan hanya memastikan orang lain dapat mengulang kesalahan yang sama, tetapi juga memastikan bahwa input yang digunakan memang layak secara kimia. Oleh karena itu, pemeriksaan struktur awal harus menjadi langkah wajib sebelum mengirim job ke ChemCompute, ORCA, GAMESS, Gaussian, Psi4, PySCF, atau program lain.

Unsur kedua adalah **metode komputasi**. Metode menentukan model teoritis yang digunakan untuk menghitung sistem. Dalam praktikum kimia kuantum, mahasiswa dapat menggunakan mekanika molekular, semiempiris, Hartree–Fock, DFT, atau metode lain sesuai tujuan praktikum. Metode tidak boleh ditulis secara umum seperti “dihitung dengan kimia komputasi” atau “dihitung dengan software ORCA”. Software bukan metode. ORCA, Gaussian, GAMESS, atau ChemCompute adalah alat atau platform, sedangkan metode adalah model perhitungannya. Penulisan yang benar harus menyebutkan, misalnya, “optimasi geometri dilakukan dengan DFT menggunakan fungsional B3LYP dan basis set def2-SVP” atau “single point energy dihitung pada tingkat HF/6-31G(d)”. Tanpa menyebut metode, hasil tidak dapat direproduksi.

Pemilihan metode juga harus konsisten dengan tujuan. Untuk praktikum pengantar, metode sederhana seperti HF/STO-3G atau PM6 dapat digunakan untuk memahami alur kerja. Untuk hasil struktur elektronik yang lebih baik, DFT seperti B3LYP atau PBE0 dengan basis set sedang dapat dipilih. Untuk sistem besar, metode semiempiris atau DFTB mungkin lebih realistis. Namun, laporan harus menjelaskan alasan pemilihan tersebut. Dokumen software kimia komputasi menekankan bahwa mahasiswa dapat bertindak seperti peneliti: membangun struktur, menetapkan basis set seperti 6-31G(d), menjalankan optimasi geometri, dan mengekstrak nilai energi bebas Gibbs dari output komputasi. Aktivitas ini hanya bermakna ilmiah jika metode yang digunakan dicatat dan dipahami.

Unsur ketiga adalah **basis set**. Basis set adalah kumpulan fungsi matematika yang digunakan untuk merepresentasikan orbital atom atau orbital molekul dalam perhitungan kuantum. Basis set yang berbeda memberikan fleksibilitas berbeda dalam menggambarkan distribusi elektron. Basis set kecil lebih cepat, tetapi kurang akurat. Basis set besar lebih fleksibel, tetapi lebih mahal secara komputasi. Dalam praktikum, basis set harus ditulis lengkap, misalnya STO-3G, 3-21G, 6-31G, 6-31G(d), 6-31+G(d,p), def2-SVP, def2-TZVP, atau cc-pVDZ. Penulisan “menggunakan basis set standar” tidak cukup karena standar setiap program atau metode dapat berbeda.

Basis set juga memengaruhi reprodusibilitas perbandingan. Jika mahasiswa membandingkan energi dua konformer, keduanya harus dihitung dengan basis set yang sama. Jika mahasiswa membandingkan pengaruh basis set, maka geometri, metode, charge, multiplicity, dan parameter

lain harus dijaga konsisten agar perubahan hasil dapat dikaitkan secara masuk akal dengan perubahan basis set. Kesalahan umum adalah mencampur data dari berbagai metode dan basis set dalam satu tabel tanpa penjelasan. Misalnya, membandingkan energi molekul A dari HF/STO-3G dengan molekul B dari B3LYP/6-31G(d) lalu menyimpulkan stabilitas relatif adalah praktik yang tidak valid. Reprodusibilitas menuntut bukan hanya catatan lengkap, tetapi juga desain perbandingan yang adil.

Unsur keempat adalah **charge** atau muatan total sistem. Muatan total menentukan jumlah elektron yang dihitung. Molekul netral, kation, anion, radikal kation, dan radikal anion memiliki struktur elektronik berbeda. Kesalahan charge adalah salah satu sumber kesalahan fatal dalam praktikum kimia kuantum. Misalnya, amonia netral NH_3 memiliki charge 0, sedangkan amonium NH_4^+ memiliki charge +1. Jika mahasiswa menghitung NH_4^+ sebagai molekul netral, jumlah elektron menjadi salah dan hasilnya tidak merepresentasikan sistem yang dimaksud. Demikian pula, asetat CH_3COO^- harus dihitung sebagai anion dengan charge -1. Dalam laporan, charge harus ditulis eksplisit bersama struktur kimia yang dihitung.

Unsur kelima adalah **multiplicity** atau multiplisitas spin. Multiplicity dihitung dari rumus:

Multiplicity = 2S + 1

dengan **S** adalah total spin elektron. Molekul dengan semua elektron berpasangan umumnya memiliki multiplicity 1 atau singlet. Radikal dengan satu elektron tak berpasangan biasanya memiliki multiplicity 2 atau doublet. Spesies dengan dua elektron tak berpasangan dapat memiliki triplet, tergantung susunan spin. Kesalahan multiplicity dapat menyebabkan hasil yang secara kimia tidak bermakna. Misalnya, oksigen molekuler O_2 keadaan dasar adalah triplet, bukan singlet. Jika O_2 dihitung sebagai singlet tanpa alasan, hasilnya tidak merepresentasikan keadaan dasar yang benar. Karena itu, multiplicity harus dicatat dan dipertanggungjawabkan.

Charge dan multiplicity sering muncul bersama dalam file input. Pada ORCA, misalnya, baris `* xyz 0 1` berarti koordinat XYZ dengan charge 0 dan multiplicity 1. Dokumen rujukan software kimia komputasi menjelaskan bahwa blok geometri ORCA dimulai dengan tanda bintang diikuti format input, muatan, dan multiplisitas spin, misalnya `* xyz 0 1` untuk molekul air netral. Kalimat sederhana ini mengandung prinsip reprodusibilitas yang sangat penting: pembaca laporan harus dapat mengetahui keadaan elektronik apa yang dihitung. Tanpa charge dan multiplicity, file input tidak lengkap secara kimia.

Unsur keenam adalah **software dan versi program**. Dalam laporan praktikum, mahasiswa harus menyebutkan perangkat lunak yang digunakan, misalnya ChemCompute, GAMESS, ORCA, Gaussian, Psi4, PySCF, Avogadro, Gabedit, Jmol, Molden, Multiwfn, atau Python. Namun, menyebut nama software saja belum ideal. Versi program juga sebaiknya dicatat karena algoritma, basis set bawaan, default grid, parameter konvergensi, atau format output dapat berubah antarversi. Misalnya, ORCA 4 dan ORCA 5 dapat memiliki perbedaan default tertentu. Python 3.10 dan Python 3.12 mungkin memiliki perbedaan kompatibilitas paket. Pustaka seperti numpy, scipy, pandas, matplotlib, RDKit, ASE, Psi4, atau PySCF juga memiliki versi yang memengaruhi eksekusi kode.

Dalam praktikum berbasis ChemCompute, mahasiswa perlu mencatat platform, tanggal akses jika relevan, paket yang dipilih, dan file output yang diunduh. ChemCompute memudahkan akses komputasi karena mahasiswa dapat menjalankan job melalui web tanpa instalasi lokal yang rumit. Namun, kemudahan ini tidak menghilangkan kewajiban dokumentasi. Justru karena perhitungan dijalankan pada platform jarak jauh, mahasiswa harus lebih disiplin mengunduh file input dan output, menyimpan job ID bila tersedia, mencatat metode, basis set, dan memastikan hasil tidak hanya tersimpan sementara di akun web. Reprodusibilitas di platform web bergantung pada kemampuan mahasiswa menyimpan jejak perhitungan secara mandiri.

Unsur ketujuh adalah **parameter komputasi**. Parameter ini meliputi kriteria konvergensi SCF, kriteria optimasi geometri, grid DFT, koreksi dispersi, model pelarut, suhu, tekanan, faktor skala frekuensi, jumlah prosesor, memori, batas iterasi, dan opsi khusus lain. Dalam praktikum dasar, tidak semua parameter perlu dimodifikasi. Namun, jika parameter digunakan atau diubah dari default, ia harus dicatat. Misalnya, jika perhitungan DFT menggunakan koreksi dispersi D3BJ, hal itu harus ditulis. Jika model pelarut CPCM(water) digunakan, harus ditulis. Jika frekuensi IR dikalikan faktor skala 0,96, harus ditulis. Jika optimasi menggunakan `TightOpt`, harus ditulis.

Parameter komputasi penting karena dapat memengaruhi hasil. Grid DFT yang terlalu kasar dapat memengaruhi energi dan gradien. SCF yang belum konvergen membuat energi tidak dapat dipercaya. Model pelarut dapat mengubah stabilitas spesies polar atau bermuatan. Faktor skala frekuensi mengubah posisi puncak IR simulasi. Koreksi dispersi dapat memengaruhi interaksi nonkovalen, konformer, dan dimer. Karena itu, laporan yang baik tidak hanya menyebut “DFT/B3LYP”, tetapi juga menyebut detail yang relevan dengan sifat yang dihitung. Untuk praktikum dasar, pencatatan dapat dibuat ringkas, tetapi tetap harus memungkinkan perhitungan diulang.

Unsur kedelapan adalah **file input dan file output**. File input adalah instruksi perhitungan. File output adalah bukti hasil. Keduanya harus disimpan. Dalam laporan praktikum, mahasiswa dapat melampirkan file input lengkap dan ringkasan bagian penting output. Untuk buku praktikum ini, setiap mata praktikum sebaiknya mengharuskan mahasiswa menyimpan file dalam struktur folder standar, misalnya `input/`, `output/`, `xyz/`, `python/`, `gambar/`, `tabel/`, dan `laporan/`. File input dan output harus diberi nama sistematis, misalnya:

```
h2o_b3lyp_def2svp_optfreq.inp
h2o_b3lyp_def2svp_optfreq.out
h2o_b3lyp_def2svp_final.xyz
h2o_b3lyp_def2svp_ir.csv
```

Nama file seperti ini sudah memuat informasi molekul, metode, basis set, dan jenis perhitungan. Sebaliknya, nama seperti `coba1.out`, `final_baru.log`, atau `praktikum_fix_fix.out` tidak memenuhi standar reprodusibilitas karena sulit ditelusuri.

Reprodusibilitas juga menuntut **pencatatan status konvergensi**. Perhitungan yang menghasilkan angka belum tentu sahih. Mahasiswa harus memeriksa apakah SCF konvergen, apakah optimasi geometri konvergen, apakah frekuensi menunjukkan struktur minimum atau keadaan transisi, dan apakah tidak ada error penting dalam output. Sumber kimia kuantum yang digunakan menyatakan bahwa hasil valid jika SCF dan optimasi konvergen serta tidak ada frekuensi imajiner untuk

struktur minimum. Pernyataan ini harus menjadi prinsip kerja praktikum. Jangan mengambil energi akhir dari output yang gagal konvergen lalu menyajikannya sebagai hasil final. Jangan menyimpulkan struktur minimum jika masih ada frekuensi imajiner, kecuali memang sedang mencari keadaan transisi dan jumlah frekuensi imajiner tepat satu.

Pemeriksaan frekuensi adalah contoh penting reprodusibilitas sekaligus validasi. Struktur hasil optimasi dapat tampak wajar secara visual, tetapi tetap berada pada titik pelana. Jika perhitungan frekuensi menunjukkan semua frekuensi nyata, struktur dapat dianggap minimum pada tingkat teori tersebut. Jika ada satu frekuensi imajiner, struktur mungkin keadaan transisi orde pertama. Jika ada lebih dari satu frekuensi imajiner, struktur biasanya belum valid sebagai minimum maupun keadaan transisi yang benar. Dokumen rujukan software kimia komputasi menekankan bahwa pencarian keadaan transisi dengan ORCA menggunakan `OptTS` harus divalidasi dengan perhitungan frekuensi; tepat satu frekuensi imajiner menunjukkan vektor getaran sepanjang koordinat reaksi, sedangkan tidak ada atau lebih dari satu frekuensi imajiner menunjukkan hasil yang tidak sesuai untuk klaim keadaan transisi.

Dalam praktikum dasar, mahasiswa sebaiknya diberi format catatan reprodusibilitas. Catatan ini dapat berupa lembar metadata yang harus diisi untuk setiap perhitungan. Isinya minimal:

Identitas perhitungan

- Nama praktikum.
- Nama mahasiswa atau kelompok.
- Tanggal perhitungan.
- Nama molekul atau sistem.
- Tujuan perhitungan.

Struktur dan keadaan sistem

- File struktur awal.
- Format file struktur.
- Charge.
- Multiplicity.
- Jumlah atom.
- Catatan protonasi, konformer, atau keadaan spin.

Metode dan parameter

- Software.
- Versi software.
- Metode.
- Basis set.
- Jenis job.
- Model pelarut jika ada.
- Koreksi dispersi jika ada.
- Kriteria konvergensi jika diubah.

- Jumlah prosesor dan memori jika relevan.

Output dan validasi

- Nama file output.
- Status SCF.
- Status optimasi.
- Energi akhir.
- Frekuensi imajiner.
- Momen dipol jika dihitung.
- Catatan error atau warning.
- File Python analisis.
- Tabel atau grafik yang dihasilkan.

Format catatan seperti ini tampak sederhana, tetapi memiliki nilai pedagogis besar. Mahasiswa belajar bahwa hasil ilmiah tidak hanya bergantung pada angka akhir, tetapi juga pada rekam jejak proses. Dalam penelitian profesional, metadata semacam ini sering menentukan apakah suatu hasil dapat dipercaya atau tidak. Dalam pembelajaran, metadata melatih mahasiswa berpikir sistematis dan bertanggung jawab.

Reprodusibilitas juga berkaitan dengan **konsistensi satuan**. Energi dapat muncul dalam Hartree, eV, kJ/mol, kcal/mol, atau cm^{-1}. Panjang ikatan dapat ditulis dalam Ångström atau Bohr. Frekuensi vibrasi biasanya dalam cm^{-1}. Momen dipol dalam Debye. Jika mahasiswa tidak mencatat satuan, perhitungan lanjutan dapat salah. Misalnya, 1 Hartree setara sekitar 2625,5 kJ/mol. Kesalahan mengonversi Hartree ke kJ/mol dapat menghasilkan kesimpulan energi reaksi yang sangat keliru. Oleh karena itu, setiap tabel hasil harus memuat satuan pada judul kolom. Kode Python juga harus mencantumkan konstanta konversi secara eksplisit, bukan menyembunyikannya dalam angka tanpa penjelasan.

Reprodusibilitas dalam praktikum Python memiliki tantangan tambahan. Kode Python harus disimpan, diberi nama jelas, dan dapat dijalankan ulang. File data yang dibaca oleh kode harus disertakan. Versi pustaka Python sebaiknya dicatat, terutama jika menggunakan pustaka khusus seperti RDKit, ASE, cclib, Psi4, PySCF, atau scipy. Kode harus memiliki komentar yang cukup, tetapi tidak berlebihan. Variabel harus diberi nama informatif. Jalur file sebaiknya relatif, bukan absolut, agar kode dapat dijalankan di komputer lain. Misalnya, gunakan `data/h2o_energy.csv`, bukan `C:\Users\Nama\Desktop\praktikum\h2o_energy.csv`. Hal-hal teknis ini merupakan bagian dari reprodusibilitas, bukan sekadar gaya pemrograman.

Dalam konteks Python, prinsip reprodusibilitas dapat dinyatakan sebagai berikut:

```
Kode + data input + versi pustaka + instruksi eksekusi = analisis yang dapat direproduksi
```

Jika mahasiswa hanya menyertakan grafik tanpa kode dan data, dosen tidak dapat memeriksa bagaimana grafik dibuat. Jika mahasiswa menyertakan kode tetapi tidak menyertakan file CSV yang dibaca, kode tidak dapat dijalankan. Jika mahasiswa menyertakan data tetapi tidak menjelaskan satuan, hasil dapat disalahartikan. Jika mahasiswa menggunakan pustaka khusus

tetapi tidak mencatat cara instalasi, pengguna lain dapat gagal menjalankan kode. Oleh karena itu, setiap mata praktikum dalam buku ini perlu menyertakan instruksi sederhana tentang paket Python yang digunakan, file input yang dibutuhkan, output yang dihasilkan, dan cara menjalankan kode.

Reprodusibilitas juga menuntut **perbandingan dengan data rujukan** jika tersedia. Dalam laporan praktikum, mahasiswa sebaiknya membandingkan panjang ikatan, sudut ikatan, frekuensi vibrasi, momen dipol, atau energi relatif dengan data eksperimen atau literatur. Rujukan kimia kuantum yang digunakan menegaskan bahwa laporan praktikum perlu memuat perbandingan dengan data eksperimen atau literatur bila tersedia; perbedaan antara hasil komputasi dan data rujukan harus dianalisis dari sisi metode, basis set, korelasi elektron, efek pelarut, dan keterbatasan model. Perbandingan ini tidak bertujuan mencari "salah-benar" secara dangkal, tetapi melatih mahasiswa memahami batas metode.

Misalnya, jika mahasiswa menghitung frekuensi regangan O–H air dengan metode HF/6-31G(d), hasil komputasi dapat lebih tinggi daripada nilai eksperimen karena pendekatan harmonik dan keterbatasan metode. Jika mahasiswa menghitung sudut H–O–H dengan basis set kecil, nilai dapat berbeda dari eksperimen. Jika mahasiswa menghitung momen dipol dalam fase gas, nilainya dapat berbeda dari kondisi larutan. Perbedaan tersebut harus dianalisis, bukan ditutupi. Dalam sains, perbedaan antara model dan eksperimen sering menjadi sumber pemahaman. Reprodusibilitas membantu mahasiswa melihat bahwa hasil komputasi adalah hasil model yang perlu dikaji, bukan angka final yang kebal kritik.

Prinsip reprodusibilitas juga berkaitan dengan **validitas perbandingan**. Perhitungan dapat saja direproduksi, tetapi desain perbandingannya tetap tidak valid. Misalnya, mahasiswa dapat mereproduksi energi etanol pada B3LYP/6-31G(d) dan energi aseton pada HF/STO-3G, tetapi perbandingan langsung keduanya tidak menjawab pertanyaan stabilitas relatif karena sistem dan metode berbeda. Reprodusibilitas adalah syarat perlu, tetapi bukan syarat cukup bagi validitas ilmiah. Validitas menuntut desain yang benar: sistem yang dibandingkan harus relevan, metode konsisten, satuan benar, dan kesimpulan tidak melampaui data.

Dalam praktikum kimia kuantum, ada beberapa kesalahan reprodusibilitas yang sangat umum. Pertama, mahasiswa hanya menyalin energi dari layar software tanpa menyimpan output. Kedua, mahasiswa lupa mencatat metode dan basis set. Ketiga, mahasiswa mengganti struktur setelah perhitungan tetapi tidak memperbarui file laporan. Keempat, mahasiswa menyebut "menggunakan DFT" tanpa menyebut fungsional. Kelima, mahasiswa menyebut "menggunakan basis set 6-31G" padahal file input memakai 6-31G(d). Keenam, mahasiswa tidak mencatat charge dan multiplicity. Ketujuh, mahasiswa mengambil hasil dari perhitungan yang tidak konvergen. Kedelapan, mahasiswa mencampur satuan Hartree dan kJ/mol. Kesembilan, mahasiswa membuat grafik dengan data yang tidak dilampirkan. Kesepuluh, mahasiswa tidak menyimpan kode Python yang digunakan untuk analisis.

Untuk mencegah kesalahan tersebut, setiap praktikum sebaiknya menggunakan **alur kerja reprodusibel**. Alur kerja ini dapat dirumuskan sebagai:

```
Pertanyaan ilmiah
→ struktur awal
→ verifikasi struktur
```

```
→ pemilihan metode dan basis set
→ penentuan charge dan multiplicity
→ penyusunan file input
→ eksekusi job
→ pemeriksaan konvergensi
→ ekstraksi data output
→ analisis Python
→ perbandingan literatur
→ interpretasi kimia
→ penyimpanan input-output-kode
→ laporan
```

Alur ini tampak panjang, tetapi sebenarnya mencerminkan proses ilmiah yang bertanggung jawab. Mahasiswa perlu dilatih menjalankannya secara konsisten sejak praktikum pertama. Setelah terbiasa, alur ini akan menjadi kebiasaan kerja. Kebiasaan ini sangat penting bagi mahasiswa yang kelak menulis skripsi, tesis, artikel ilmiah, atau laporan penelitian berbasis kimia komputasi.

Reprodusibilitas juga memiliki dimensi sosial dan akademik. Dalam kelas praktikum, dosen harus dapat memeriksa pekerjaan mahasiswa. Teman sejawat harus dapat memahami dan mengulang hasil. Peneliti lain harus dapat menilai apakah kesimpulan didukung oleh data. Dalam publikasi ilmiah, editor dan reviewer menuntut metode yang jelas. Dalam pendidikan, reprodusibilitas menjadi sarana membangun kejujuran akademik. Mahasiswa yang menyimpan input, output, dan kode dengan baik lebih sulit melakukan fabrikasi data karena jejak perhitungan dapat diperiksa. Sebaliknya, laporan yang hanya berisi angka tanpa file pendukung rentan terhadap kesalahan, manipulasi, atau plagiarisme.

Dari perspektif pedagogis, reprodusibilitas memperkuat pembelajaran aktif. Mahasiswa tidak lagi menjadi penerima data yang pasif, tetapi menjadi penghasil data yang dapat dipertanggungjawabkan. Dokumen software kimia komputasi menekankan bahwa makmal maya mendorong pergeseran dari pembelajaran pasif menuju active learning dan inkuiri saintifik; mahasiswa membangun struktur, menetapkan basis set, menjalankan optimasi geometri, dan mengekstrak sendiri nilai energi bebas Gibbs dari output komputasi. Kegiatan seperti ini hanya akan menghasilkan pembelajaran mendalam jika disertai dokumentasi reprodusibel. Tanpa dokumentasi, aktivitas inkuiri berubah menjadi aktivitas coba-coba yang sulit dievaluasi.

Reprodusibilitas juga mendukung **kolaborasi**. Dalam proyek kelompok, setiap anggota dapat mengerjakan molekul berbeda, metode berbeda, atau analisis berbeda. Agar hasil dapat digabung, format pencatatan harus seragam. Jika satu mahasiswa menulis energi dalam Hartree, yang lain dalam eV, dan yang lain dalam kJ/mol tanpa keterangan, tabel kelompok menjadi kacau. Jika satu mahasiswa menyimpan output tetapi yang lain hanya menyimpan tangkapan layar, validasi menjadi tidak merata. Oleh karena itu, buku praktikum ini perlu menetapkan template tabel dan folder yang sama untuk seluruh mata praktikum. Standar ini bukan untuk membatasi kreativitas, tetapi untuk memastikan kerja kolaboratif tetap tertib.

Contoh konkret dapat diberikan pada praktikum optimasi geometri H_2O. Laporan yang tidak reprodusibel hanya menulis: “Air dioptimasi menggunakan DFT dan diperoleh sudut 104,7°.” Laporan semacam ini tidak cukup. Laporan yang reprodusibel harus menulis bahwa struktur awal disimpan sebagai `h2o_initial.xyz`, optimasi dilakukan dengan ORCA versi tertentu

menggunakan B3LYP/def2-SVP, charge 0, multiplicity 1, job `Opt Freq`, SCF konvergen, optimasi konvergen, tidak ada frekuensi imajiner, energi akhir sekian Hartree, struktur akhir disimpan sebagai `h2o_final.xyz`, panjang ikatan O–H dan sudut H–O–H dihitung dengan script Python `analisis_geometri.py`, dan hasil dibandingkan dengan data literatur. Laporan kedua jauh lebih kuat secara ilmiah karena dapat diperiksa.

Contoh lain adalah praktikum energi reaksi. Misalnya mahasiswa menghitung energi isomerisasi cis-2-butena menjadi trans-2-butena. Agar reprodusibel, laporan harus mencatat file input kedua isomer, metode dan basis set yang sama, charge dan multiplicity sama, status optimasi dan frekuensi, energi elektronik, koreksi zero-point energy jika digunakan, energi bebas Gibbs jika dihitung, satuan konversi, serta script Python yang menghitung ΔE atau ΔG. Jika salah satu isomer memiliki frekuensi imajiner, hasil energi relatif tidak boleh langsung digunakan. Jika satu isomer dihitung dengan model pelarut dan yang lain fase gas, perbandingan tidak sah. Contoh ini menunjukkan bahwa reprodusibilitas juga melatih konsistensi logika kimia.

Dalam praktikum HOMO-LUMO, reprodusibilitas menuntut pencatatan orbital mana yang disebut HOMO dan LUMO, metode yang digunakan, satuan energi orbital, dan cara menghitung gap. Jika energi orbital diambil dari output DFT, mahasiswa perlu menyadari bahwa interpretasinya bersifat model-dependent. Jika diagram orbital dibuat dengan Python, data energi harus disertakan. Jika gambar orbital diambil dari Molden, Jmol, GaussView, atau Multiwfn, file orbital atau output sumbernya harus dicatat. Tanpa catatan ini, gambar orbital hanya menjadi ilustrasi tanpa basis data yang jelas.

Dalam praktikum frekuensi IR, reprodusibilitas menuntut pencatatan frekuensi mentah, intensitas, faktor skala jika digunakan, metode pelebaran puncak dalam spektrum simulasi, dan satuan sumbu grafik. Jika mahasiswa membuat spektrum IR dengan Python, mereka harus menjelaskan apakah puncak dibuat dengan fungsi Gaussian atau Lorentzian, berapa lebar puncak yang digunakan, dan apakah intensitas dinormalisasi. Dua spektrum dari data frekuensi yang sama dapat tampak berbeda jika parameter pelebaran berbeda. Oleh karena itu, grafik juga harus direproduksi, bukan hanya angka frekuensi.

Reprodusibilitas juga berkaitan dengan **versi laporan**. Dalam pengerjaan praktikum, mahasiswa sering memperbaiki struktur, mengganti metode, mengulang perhitungan, atau memperbaiki kode. Setiap perubahan perlu dikelola. File final harus jelas. Jika ada revisi, versi lama jangan dicampur dengan versi baru tanpa keterangan. Untuk pembelajaran dasar, penamaan sederhana seperti `v1`, `v2`, atau tanggal dapat digunakan. Untuk proyek lebih lanjut, sistem version control seperti Git dapat diperkenalkan. Namun, pada tahap awal, yang paling penting adalah mahasiswa tidak menimpa file penting tanpa cadangan dan tidak mencampur output dari perhitungan berbeda.

Dalam konteks reprodusibilitas, **catatan kegagalan** juga penting. Tidak semua perhitungan berhasil. SCF dapat gagal konvergen. Optimasi dapat berhenti pada struktur tidak wajar. Frekuensi dapat menunjukkan banyak frekuensi imajiner. Job dapat berhenti karena memori kurang. Mahasiswa sering tergoda menghapus kegagalan dari laporan. Padahal, dalam pendidikan ilmiah, kegagalan dapat menjadi bahan belajar. Catatan kegagalan membantu mahasiswa memahami apa yang salah: struktur awal buruk, metode tidak cocok, basis set terlalu besar, charge salah, multiplicity salah, atau parameter konvergensi perlu disesuaikan. Laporan praktikum dapat

memuat ringkasan kegagalan dan langkah perbaikan, terutama jika kegagalan tersebut relevan dengan proses pembelajaran.

Reprodusibilitas juga menuntut **kejelasan batas klaim**. Jika suatu hasil hanya dihitung dengan satu metode sederhana, kesimpulan harus dibatasi. Misalnya, mahasiswa boleh menulis "pada tingkat teori B3LYP/def2-SVP, konformer anti butana lebih stabil daripada gauche sebesar sekian kJ/mol." Kalimat ini lebih ilmiah daripada "konformer anti pasti lebih stabil." Kalimat pertama menyebut konteks metode. Kalimat kedua terlalu absolut. Dalam kimia komputasi, setiap klaim harus dikaitkan dengan tingkat teori yang digunakan. Ini bukan kelemahan, melainkan bentuk kejujuran metodologis.

Prinsip ini sejalan dengan literatur kimia komputasi modern. Cramer menekankan bahwa model kimia komputasi harus dipahami bersama asumsi dan batas validitasnya. Jensen menegaskan pentingnya pemilihan basis set, korelasi elektron, dan perbandingan metodologis dalam interpretasi hasil. Szabo dan Ostlund menunjukkan bahwa struktur elektronik molekul dibangun dari pendekatan matematis yang memiliki konsekuensi terhadap hasil energi dan fungsi gelombang. Levine, Atkins, dan McQuarrie menempatkan mekanika kuantum sebagai teori yang kuat, tetapi penerapannya pada sistem nyata memerlukan hampiran. Dalam konteks praktikum, sintesis literatur ini mengarah pada satu prinsip: hasil komputasi harus terdokumentasi, tervalidasi, dan ditafsirkan dalam batas metode.

Agar prinsip reprodusibilitas dapat diterapkan dalam buku ini, setiap mata praktikum perlu memiliki bagian **Checklist Reprodusibilitas**. Checklist tersebut dapat dibuat ringkas tetapi wajib. Misalnya:

1. Struktur awal disimpan.
2. File input dilampirkan.
3. Charge dan multiplicity ditulis.
4. Metode dan basis set ditulis lengkap.
5. Software dan versi dicatat.
6. Parameter khusus dicatat.
7. Output disimpan.
8. SCF/optimasi/frekuensi divalidasi.
9. Data dianalisis dengan Python.
10. File kode dan data disertakan.
11. Satuan ditulis jelas.
12. Hasil dibandingkan dengan literatur bila tersedia.
13. Kesimpulan menyebut batas metode.

Checklist ini membantu dosen menilai laporan dan membantu mahasiswa menghindari kelalaian. Dalam jangka panjang, checklist membentuk budaya kerja ilmiah. Budaya ini jauh lebih penting daripada sekadar menyelesaikan satu praktikum, karena akan terbawa ke tugas akhir, penelitian, dan publikasi.

Reprodusibilitas juga dapat dihubungkan dengan prinsip **FAIR data**, yaitu data sebaiknya mudah ditemukan, dapat diakses, interoperabel, dan dapat digunakan ulang. Walaupun konsep FAIR

sering dibahas dalam konteks data penelitian, prinsipnya relevan untuk praktikum. File harus diberi nama jelas agar mudah ditemukan. File harus disimpan dalam folder yang dapat diakses. Format file sebaiknya terbuka atau umum agar interoperabel. Metadata harus lengkap agar data dapat digunakan ulang. Dalam praktikum kimia kuantum, FAIR bukan slogan teknis, tetapi kebiasaan menyimpan input, output, kode, dan data secara rapi.

Secara praktis, dosen dapat menilai reprodusibilitas dengan mencoba menjalankan ulang sebagian pekerjaan mahasiswa. Misalnya, dosen memilih satu file input dari laporan, menjalankannya kembali, lalu memeriksa apakah hasil energi mendekati output mahasiswa. Atau dosen menjalankan script Python mahasiswa dengan data yang disertakan untuk melihat apakah tabel dan grafik dapat dibuat ulang. Penilaian seperti ini akan mendorong mahasiswa menulis laporan lebih jujur dan rapi. Mahasiswa akan memahami bahwa laporan bukan sekadar narasi, tetapi dokumen ilmiah yang dapat diuji.

Reprodusibilitas juga memiliki dimensi etika yang akan dibahas lebih lanjut pada subbab berikutnya. Ketika mahasiswa tidak menyimpan output, mengubah data tanpa catatan, menyalin hasil teman, atau membuat angka fiktif, masalahnya bukan hanya teknis, tetapi etis. Reprodusibilitas menjadi benteng awal melawan fabrikasi, falsifikasi, dan plagiarisme. Dengan input, output, dan kode yang lengkap, proses ilmiah menjadi terbuka untuk diperiksa. Dengan proses yang terbuka, kejujuran akademik lebih mudah ditegakkan. Oleh karena itu, reprodusibilitas dan etika akademik tidak dapat dipisahkan.

Pada akhirnya, prinsip reprodusibilitas praktikum kimia kuantum dapat diringkas dalam satu sikap ilmiah: **setiap angka harus memiliki jejak**. Energi harus dapat ditelusuri ke output. Output harus dapat ditelusuri ke input. Input harus dapat ditelusuri ke struktur awal, metode, basis set, charge, multiplicity, dan parameter. Grafik harus dapat ditelusuri ke data dan kode. Kesimpulan harus dapat ditelusuri ke analisis. Jika jejak ini lengkap, praktikum memiliki kualitas ilmiah. Jika jejak ini terputus, hasil menjadi lemah walaupun tampak meyakinkan.

Subbab berikutnya akan membahas etika akademik dalam praktikum komputasi. Pembahasan tersebut merupakan kelanjutan langsung dari reprodusibilitas, karena kejujuran data, larangan fabrikasi output, sitasi software, dan tanggung jawab akademik dalam pelaporan hanya dapat ditegakkan jika seluruh proses praktikum terdokumentasi dengan baik.

1.7 Etika Akademik dalam Praktikum Komputasi

Etika akademik dalam praktikum komputasi adalah seperangkat prinsip moral, ilmiah, dan profesional yang mengatur bagaimana mahasiswa merancang perhitungan, memperoleh data, membaca output, menggunakan software, menulis kode, menyusun laporan, mengutip sumber, serta mempertanggungjawabkan kesimpulan. Dalam praktikum kimia kuantum, etika akademik tidak boleh dipahami secara sempit sebagai larangan menyontek laporan teman. Etika akademik mencakup kejujuran data, transparansi metode, penghormatan terhadap karya ilmiah orang lain, kepatuhan terhadap lisensi software, tanggung jawab terhadap hasil komputasi, serta kesediaan menyatakan keterbatasan model yang digunakan.

Praktikum komputasi memiliki tantangan etik yang khas. Dalam laboratorium basah, pelanggaran etika dapat berupa memalsukan hasil titrasi, mengubah angka absorbansi, membuang data yang tidak sesuai harapan, atau menyalin laporan teman. Dalam praktikum kimia kuantum, bentuk pelanggarannya dapat lebih halus: menyalin output orang lain, mengubah energi total agar sesuai teori, membuat grafik tanpa data asli, mengambil gambar orbital dari internet tanpa sitasi, menyatakan job "konvergen" padahal output menunjukkan kegagalan, menghapus frekuensi imajiner dari laporan, menulis metode yang berbeda dari input sebenarnya, menggunakan software bajakan, atau menyalin kode Python tanpa memahami dan menyebut sumbernya. Karena itu, etika akademik dalam praktikum komputasi harus dibahas secara eksplisit sejak awal.

Etika ini berhubungan langsung dengan reprodusibilitas yang telah dijelaskan pada subbab sebelumnya. Reprodusibilitas adalah syarat teknis agar data dapat diperiksa, sedangkan etika akademik adalah komitmen moral agar data tersebut dilaporkan secara jujur. Sumber kimia kuantum yang menjadi dasar buku ini menegaskan bahwa setiap praktikum komputasi harus menyertakan struktur awal, muatan, multiplicity, metode, basis set, kriteria konvergensi, serta file output yang dapat diperiksa ulang; tanpa dokumentasi input dan output, hasil komputasi sulit diverifikasi. Prinsip tersebut sekaligus menjadi prinsip etis: mahasiswa tidak boleh menampilkan hasil tanpa jejak proses yang dapat ditelusuri.

Secara historis, etika akademik berkembang bersama perkembangan ilmu pengetahuan modern. Ketika sains menjadi kegiatan kolektif yang melibatkan publikasi, peer review, pendanaan, reputasi institusi, dan dampak sosial, kejujuran ilmiah menjadi prasyarat mutlak. The National Academies dalam *On Being a Scientist* menekankan bahwa integritas ilmiah bertumpu pada kejujuran dalam memperoleh, menganalisis, dan melaporkan data (National Academies, 2009). Singapore Statement on Research Integrity menegaskan empat prinsip utama integritas riset: kejujuran, akuntabilitas, profesionalisme, dan tata kelola yang baik. ALLEA dalam *The European Code of Conduct for Research Integrity* menekankan reliabilitas, kejujuran, penghormatan, dan akuntabilitas sebagai pilar praktik riset yang baik. Prinsip-prinsip tersebut sangat relevan untuk praktikum kimia kuantum, meskipun konteksnya berada pada pembelajaran mahasiswa.

Dalam perspektif kimia komputasi, etika akademik berkaitan erat dengan sifat data komputasi. Data komputasi tampak mudah diperoleh, mudah disalin, dan mudah dimodifikasi. Sebuah file output dapat dikirim melalui pesan singkat. Sebuah grafik dapat dibuat ulang dengan mengganti beberapa angka. Sebuah struktur molekul dapat diambil dari database tanpa mencatat sumber. Sebuah kode Python dapat disalin dari internet dan dijalankan tanpa pemahaman. Kemudahan ini memberi manfaat besar bagi pembelajaran, tetapi juga membuka peluang pelanggaran akademik. Oleh sebab itu, praktikum kimia kuantum harus menanamkan budaya bahwa data komputasi sama seriusnya dengan data eksperimen basah. Keduanya harus jujur, lengkap, dapat diperiksa, dan dapat dipertanggungjawabkan.

Bentuk pelanggaran pertama adalah **fabrikasi data**. Fabrikasi berarti membuat data yang sebenarnya tidak pernah diperoleh. Dalam praktikum komputasi, fabrikasi dapat terjadi ketika mahasiswa menulis nilai energi, frekuensi, momen dipol, atau gap HOMO-LUMO tanpa benar-benar menjalankan perhitungan. Fabrikasi juga terjadi ketika mahasiswa membuat file tabel seolah-olah berasal dari output, padahal angka-angkanya dikarang. Pelanggaran ini sangat serius

karena merusak dasar keilmuan. Sains bertumpu pada data yang diperoleh melalui prosedur yang dapat diperiksa. Jika data dibuat tanpa prosedur, laporan kehilangan nilai ilmiah.

Bentuk pelanggaran kedua adalah **falsifikasi data**. Falsifikasi berarti mengubah, memilih, menghapus, atau memanipulasi data sehingga laporan memberikan gambaran yang menyesatkan. Dalam praktikum kimia kuantum, falsifikasi dapat berupa menghapus frekuensi imajiner dari tabel agar struktur tampak sebagai minimum, mengganti energi akhir dengan angka dari perhitungan lain, menyatakan optimasi konvergen padahal output menunjukkan gagal konvergen, mengubah label metode agar tampak lebih tinggi, atau membuang data konformer yang tidak sesuai dugaan. Falsifikasi sering tampak lebih "ringan" daripada fabrikasi karena mahasiswa memang melakukan sebagian perhitungan, tetapi dampaknya tetap serius karena kesimpulan menjadi tidak jujur.

Bentuk pelanggaran ketiga adalah **plagiarisme**. Plagiarisme dalam praktikum komputasi tidak hanya berupa menyalin paragraf teori dari internet atau laporan teman. Plagiarisme juga dapat berupa menyalin file input, output, grafik, gambar orbital, tabel energi, atau kode Python tanpa menyebut sumber. Jika mahasiswa menggunakan struktur molekul dari database, kode dari repositori publik, atau template input dari dokumentasi software, sumber tersebut perlu disebut sesuai konteks. Dalam pembelajaran, penggunaan template diperbolehkan sepanjang mahasiswa memahami, memodifikasi secara sah, dan memberi atribusi. Yang tidak diperbolehkan adalah menyajikan karya orang lain sebagai karya sendiri.

Bentuk pelanggaran keempat adalah **misrepresentasi metode**. Ini terjadi ketika laporan menyebut metode yang tidak sama dengan input yang sebenarnya digunakan. Misalnya, laporan menulis "perhitungan dilakukan dengan DFT/B3LYP/6-31G(d)", padahal input menggunakan PM6. Atau laporan menyebut "frekuensi dihitung", padahal job yang dijalankan hanya optimasi geometri. Atau laporan menulis "model pelarut air digunakan", padahal perhitungan dilakukan dalam fase gas. Misrepresentasi metode dapat terjadi karena ketidaktahuan atau kesengajaan. Dalam kedua kasus, hasilnya tetap bermasalah karena pembaca tidak dapat menilai validitas data secara benar.

Bentuk pelanggaran kelima adalah **penggunaan software tidak sah**. Dalam kimia komputasi, sebagian software bersifat komersial, sebagian gratis untuk akademik, dan sebagian open-source. Dokumen rujukan software kimia komputasi menjelaskan bahwa institusi pendidikan menghadapi pilihan strategis antara software komersial dan ekosistem open-source atau free-for-academia; keputusan ini berkaitan dengan aksesibilitas, pendanaan, dan filosofi keterbukaan sains. Mahasiswa harus memahami bahwa software berlisensi tidak boleh digunakan secara bajakan. Jika kampus memiliki lisensi Gaussian, GaussView, atau software komersial lain, penggunaannya harus sesuai ketentuan lisensi. Jika tidak memiliki lisensi, mahasiswa dapat menggunakan alternatif legal seperti ORCA untuk akademik sesuai ketentuannya, GAMESS, Psi4, PySCF, Avogadro, Jmol, Multiwfn, atau platform seperti ChemCompute bila tersedia.

Etika penggunaan software juga mencakup **sitasi software**. Banyak mahasiswa mengutip buku teori dan artikel jurnal, tetapi lupa mengutip software yang digunakan. Padahal, software adalah karya ilmiah dan teknis yang dikembangkan oleh komunitas peneliti. Setiap software biasanya memiliki panduan sitasi resmi. ORCA, Gaussian, GAMESS, Psi4, PySCF, Avogadro, Jmol, Multiwfn, RDKit, ASE, cclib, numpy, scipy, pandas, dan matplotlib memiliki rujukan atau dokumentasi sitasi masing-masing. ChemCompute juga menyediakan panduan sitasi untuk

penggunaan platformnya. Sitasi software bukan formalitas; ia adalah pengakuan terhadap kerja intelektual pengembang dan bagian dari transparansi metode.

Dalam laporan praktikum, sitasi software dapat ditulis secara sederhana pada bagian metode. Misalnya: “Struktur awal dibangun menggunakan Avogadro, optimasi geometri dilakukan menggunakan ORCA melalui tingkat teori B3LYP/def2-SVP, visualisasi orbital dilakukan dengan Jmol, dan analisis data dilakukan menggunakan Python dengan pustaka numpy, pandas, dan matplotlib.” Jika laporan bersifat formal, rujukan software dicantumkan dalam daftar pustaka. Untuk praktikum, dosen dapat menyediakan format sitasi standar agar mahasiswa tidak bingung. Namun, mahasiswa tetap perlu memahami bahwa setiap software yang berperan dalam menghasilkan data harus disebut secara jujur.

Etika akademik juga menuntut **kejelasan peran manusia dan mesin**. Dalam praktikum modern, mahasiswa dapat menggunakan platform web, software otomatis, dan bahkan bantuan AI untuk menyusun kode atau memperbaiki laporan. Penggunaan bantuan digital tidak selalu salah, tetapi harus transparan dan sesuai kebijakan kelas. Jika mahasiswa menggunakan AI untuk membantu merapikan kode Python, ia tetap bertanggung jawab memahami kode tersebut, memeriksa hasilnya, dan memastikan tidak ada klaim palsu. AI tidak boleh dijadikan alasan untuk menyerahkan laporan yang tidak dipahami. Dalam konteks pendidikan, tujuan praktikum bukan hanya menghasilkan laporan, tetapi membentuk kompetensi ilmiah mahasiswa.

Landasan etika akademik dalam praktikum kimia kuantum dapat diperkuat melalui perspektif nilai Islam. Ilmu dalam Islam tidak hanya dipandang sebagai kumpulan informasi, tetapi amanah. Kejujuran, tanggung jawab, dan larangan mengikuti sesuatu tanpa pengetahuan merupakan prinsip moral yang sangat relevan dengan praktik ilmiah. Al-Qur’an menegaskan:

وَلَا تَقْفُ مَا لَيْسَ لَكَ بِهِۦ عِلْمٌ ۚ إِنَّ ٱلسَّمْعَ وَٱلْبَصَرَ وَٱلْفُؤَادَ كُلُّ أُو۟لَٰٓئِكَ كَانَ عَنْهُ مَسْـُٔولًا

Terjemah: “Dan janganlah engkau mengikuti sesuatu yang tidak engkau ketahui. Sesungguhnya pendengaran, penglihatan, dan hati, semuanya itu akan dimintai pertanggungjawaban.” (QS. Al-Isrā’ [17]: 36)

Ayat ini sangat relevan dengan praktikum komputasi. Mahasiswa tidak boleh menulis kesimpulan yang tidak didukung data. Tidak boleh menyatakan struktur minimum tanpa memeriksa frekuensi. Tidak boleh menulis metode tertentu tanpa benar-benar menggunakannya. Tidak boleh menyalin hasil tanpa mengetahui asal-usulnya. Setiap klaim ilmiah harus berbasis pengetahuan yang dapat dipertanggungjawabkan.

Al-Qur’an juga melarang pencampuran kebenaran dengan kebatilan:

وَلَا تَلْبِسُوا۟ ٱلْحَقَّ بِٱلْبَـٰطِلِ وَتَكْتُمُوا۟ ٱلْحَقَّ وَأَنتُمْ تَعْلَمُونَ

Terjemah: “Dan janganlah kamu campuradukkan kebenaran dengan kebatilan dan janganlah kamu sembunyikan kebenaran, sedangkan kamu mengetahui.” (QS. Al-Baqarah [2]: 42)

Dalam praktikum komputasi, mencampur kebenaran dan kebatilan dapat berbentuk mencampur data asli dengan data karangan, menampilkan output berhasil tetapi menyembunyikan output gagal, atau menulis sebagian metode secara benar tetapi menutupi parameter penting yang mengubah hasil. Ayat ini mengingatkan bahwa pengetahuan harus disampaikan secara utuh, bukan dipilih hanya untuk mendukung kesimpulan yang diinginkan.

Prinsip keadilan dan objektivitas juga ditegaskan dalam Al-Qur'an:

يَـٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ كُونُوا۟ قَوَّٰمِينَ لِلَّهِ شُهَدَآءَ بِٱلْقِسْطِ ۖ وَلَا يَجْرِمَنَّكُمْ شَنَـَٔانُ قَوْمٍ عَلَىٰٓ أَلَّا تَعْدِلُوا۟ ۚ ٱعْدِلُوا۟ هُوَ أَقْرَبُ لِلتَّقْوَىٰ ۖ وَٱتَّقُوا۟ ٱللَّهَ ۚ إِنَّ ٱللَّهَ خَبِيرٌۢ بِمَا تَعْمَلُونَ

Terjemah: "Wahai orang-orang yang beriman, jadilah kamu penegak kebenaran karena Allah, menjadi saksi dengan adil. Janganlah kebencianmu terhadap suatu kaum mendorong kamu untuk tidak berlaku adil. Berlaku adillah, karena adil itu lebih dekat kepada takwa. Dan bertakwalah kepada Allah. Sesungguhnya Allah Maha Teliti terhadap apa yang kamu kerjakan." (QS. Al-Mā'idah [5]: 8)

Dalam konteks akademik, adil berarti menilai data sebagaimana adanya. Jika hasil komputasi tidak sesuai dugaan, mahasiswa tidak boleh memaksakan kesimpulan. Jika metode sederhana menghasilkan penyimpangan besar, mahasiswa harus mengakuinya dan menganalisis penyebabnya. Jika data literatur berbeda dari hasil perhitungan, mahasiswa harus membandingkannya secara objektif. Keadilan ilmiah menuntut kesetiaan pada data, bukan pada keinginan pribadi.

Al-Qur'an juga memerintahkan ucapan yang benar:

يَـٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ ٱتَّقُوا۟ ٱللَّهَ وَقُولُوا۟ قَوْلًا سَدِيدًا

Terjemah: "Wahai orang-orang yang beriman, bertakwalah kepada Allah dan ucapkanlah perkataan yang benar."
(QS. Al-Aḥzāb [33]: 70)

Dalam laporan praktikum, "perkataan yang benar" berarti kalimat yang akurat, tidak berlebihan, dan sesuai bukti. Mahasiswa sebaiknya menulis, "Pada tingkat teori B3LYP/def2-SVP, hasil perhitungan menunjukkan…" bukan "molekul ini pasti…", jika data hanya berasal dari satu metode. Kalimat ilmiah harus berhati-hati karena setiap klaim memiliki batas.

Hadis Nabi saw. juga memberikan dasar kuat bagi etika akademik. Rasulullah saw. bersabda:

مَنْ غَشَّنَا فَلَيْسَ مِنَّا

Terjemah: "Barang siapa menipu kami, maka ia bukan termasuk golongan kami." (HR. Muslim)

Hadis ini secara langsung relevan dengan larangan manipulasi data. Menipu dalam praktikum komputasi dapat berupa menyerahkan output teman sebagai milik sendiri, mengubah angka energi,

atau membuat hasil palsu. Praktikum adalah bagian dari amanah ilmu. Menipu dalam ilmu bukan hanya melanggar aturan kelas, tetapi juga merusak karakter ilmuwan.

Rasulullah saw. juga bersabda:

إِنَّ ٱلصِّدْقَ يَهْدِي إِلَى ٱلْبِرِّ، وَإِنَّ ٱلْبِرَّ يَهْدِي إِلَى ٱلْجَنَّةِ، وَإِنَّ ٱلرَّجُلَ لَيَصْدُقُ حَتَّى يَكُونَ صِدِّيقًا، وَإِنَّ ٱلْكَذِبَ يَهْدِي إِلَى ٱلْفُجُورِ وَإِنَّ ٱلْفُجُورَ يَهْدِي إِلَى ٱلنَّارِ

Terjemah: "Sesungguhnya kejujuran membawa kepada kebaikan, dan kebaikan membawa ke surga. Seseorang senantiasa berlaku jujur hingga ia dicatat sebagai orang yang sangat jujur. Sesungguhnya dusta membawa kepada kefasikan, dan kefasikan membawa ke neraka." (HR. al-Bukhari dan Muslim)

Hadis ini menempatkan kejujuran sebagai kebiasaan karakter, bukan sekadar tindakan sesaat. Dalam praktikum, mahasiswa harus membiasakan jujur ketika job gagal, jujur ketika data berbeda dari teori, jujur ketika tidak memahami kode, dan jujur ketika menggunakan bantuan sumber lain.

Dalam hadis lain, Nabi saw. bersabda:

كَفَىٰ بِٱلْمَرْءِ كَذِبًا أَنْ يُحَدِّثَ بِكُلِّ مَا سَمِعَ

Terjemah: "Cukuplah seseorang dianggap berdusta apabila ia menceritakan setiap apa yang ia dengar."
(HR. Muslim)

Hadis ini relevan dengan penggunaan sumber digital. Mahasiswa tidak boleh memasukkan teori, angka, struktur, atau kode dari internet tanpa memeriksa kebenarannya. Tidak semua tutorial, forum, blog, atau kode daring dapat dipercaya. Dalam kimia komputasi, input yang tampak berjalan belum tentu benar secara kimia. Karena itu, setiap sumber perlu diverifikasi.

Rasulullah saw. juga bersabda:

كُلُّكُمْ رَاعٍ، وَكُلُّكُمْ مَسْئُولٌ عَنْ رَعِيَّتِهِ

Terjemah: "Setiap kalian adalah pemimpin, dan setiap kalian akan dimintai pertanggungjawaban atas yang dipimpinnya."
(HR. al-Bukhari dan Muslim)

Dalam praktikum, mahasiswa bertanggung jawab atas data, file, kode, dan kesimpulannya. Dosen bertanggung jawab membimbing, memberi standar, dan menilai secara adil. Kelompok praktikum bertanggung jawab memastikan kerja kolaboratif dilakukan jujur. Etika akademik bukan hanya tugas individu, tetapi budaya bersama.

Dalam perspektif pendidikan, etika akademik harus diajarkan melalui praktik, bukan hanya nasihat. Dosen perlu menetapkan standar laporan yang menuntut lampiran input, output, dan kode Python. Mahasiswa harus diminta menjelaskan bagian penting output, bukan hanya

mengumpulkan angka. Tugas praktikum sebaiknya dirancang agar setiap kelompok memiliki variasi molekul, metode, atau parameter, sehingga penyalinan mentah lebih mudah terdeteksi dan kreativitas lebih terdorong. Selain itu, rubrik penilaian harus memberi bobot pada kejujuran proses, validasi output, dan kemampuan interpretasi, bukan hanya pada "hasil akhir yang benar".

Etika akademik juga perlu dibangun melalui **budaya mengakui kegagalan**. Dalam komputasi kimia, kegagalan job bukan aib. SCF dapat gagal konvergen. Optimasi dapat berhenti di struktur yang tidak wajar. Frekuensi dapat menunjukkan struktur bukan minimum. Model pelarut dapat tidak cocok. Basis set dapat terlalu besar untuk sumber daya yang tersedia. Mahasiswa yang melaporkan kegagalan dengan analisis penyebab justru menunjukkan kedewasaan ilmiah. Sebaliknya, mahasiswa yang menyembunyikan kegagalan dan hanya menampilkan hasil yang tampak bagus belum tentu memiliki integritas ilmiah yang baik. Karena itu, laporan praktikum dapat memuat bagian "catatan kendala dan perbaikan" untuk menumbuhkan budaya ilmiah yang jujur.

Etika dalam praktikum komputasi juga mencakup **penggunaan sumber daya komputasi secara bertanggung jawab**. Platform seperti ChemCompute, server kampus, atau komputer laboratorium memiliki kapasitas terbatas. Mahasiswa tidak boleh mengirim job besar tanpa kebutuhan pedagogis yang jelas, menjalankan perhitungan berulang tanpa memeriksa input, atau membebani server dengan sistem yang tidak sesuai tingkat praktikum. Tanggung jawab komputasi berarti memilih metode dan basis set yang proporsional dengan tujuan. Untuk latihan konsep, molekul kecil dan metode sedang lebih tepat. Untuk proyek mini, perlu diskusi dengan dosen agar sistem, metode, dan sumber daya seimbang.

Dalam konteks software open-source dan free-for-academia, etika juga berarti menghargai keterbukaan ilmu. Dokumen rujukan software kimia komputasi menunjukkan bahwa perangkat lunak open-source dan gratis untuk akademik membuka akses lebih luas bagi institusi pendidikan, terutama yang memiliki keterbatasan biaya lisensi. Namun, keterbukaan bukan berarti bebas tanpa aturan. Lisensi tetap harus dihormati. Jika software mensyaratkan penggunaan akademik nonkomersial, syarat itu harus dipatuhi. Jika software meminta sitasi, sitasi harus diberikan. Jika kode dimodifikasi dari sumber terbuka, lisensi dan atribusi harus dijaga.

Etika akademik dalam penggunaan Python juga perlu ditegaskan. Menyalin kode dari internet tidak selalu salah jika kode tersebut berlisensi terbuka dan sumbernya disebut. Yang salah adalah menyalin kode tanpa atribusi, tidak memahami kode, lalu mengakuinya sebagai karya sendiri. Dalam praktikum, mahasiswa boleh menggunakan pustaka seperti numpy, pandas, matplotlib, scipy, ASE, atau RDKit jika tersedia. Namun, mereka harus menuliskan pustaka yang digunakan dan memahami fungsi utamanya. Jika menggunakan kode bantuan dari dosen, buku ini, dokumentasi resmi, atau sumber lain, mereka harus menyebutkan sumber sesuai aturan kelas. Kode adalah bagian dari karya ilmiah, bukan sekadar alat teknis yang bebas dari etika.

Salah satu masalah yang sering muncul adalah **overclaiming**, yaitu membuat klaim yang melampaui kapasitas data. Misalnya, mahasiswa menghitung gap HOMO-LUMO satu molekul dengan satu metode sederhana lalu menyimpulkan molekul tersebut "paling baik sebagai obat". Klaim seperti itu tidak sah. Gap HOMO-LUMO tidak cukup untuk menilai aktivitas farmakologis. Contoh lain, mahasiswa menghitung energi reaksi fase gas lalu menyimpulkan reaksi pasti

berlangsung spontan dalam larutan biologis. Klaim seperti itu mengabaikan efek pelarut, suhu, entropi, kinetika, dan lingkungan. Etika akademik menuntut mahasiswa menyatakan kesimpulan sesuai ruang lingkup data.

Overclaiming juga dapat terjadi dalam interpretasi visualisasi. Gambar orbital yang indah tidak otomatis membuktikan mekanisme reaksi. Peta potensial elektrostatik tidak otomatis menentukan seluruh jalur reaksi. Momen dipol tidak sendirian menentukan kelarutan. Muatan atom dari satu skema populasi tidak boleh dianggap sebagai muatan absolut. Sumber kimia kuantum yang digunakan menekankan bahwa hasil perhitungan harus dibaca secara kritis: konvergensi SCF, konvergensi optimasi, frekuensi imajiner, nilai energi, satuan, dan konsistensi metode harus diperiksa sebelum interpretasi kimia dilakukan. Etika ilmiah berarti mengakui batas setiap metode dan setiap jenis data.

Etika akademik juga mencakup **keadilan dalam kerja kelompok**. Dalam praktikum komputasi, sering kali satu mahasiswa lebih terampil menggunakan software atau Python, sedangkan anggota lain hanya menumpang nama. Ini tidak sehat secara akademik. Kerja kelompok harus membagi tugas secara jelas: satu anggota dapat menyiapkan struktur, satu menjalankan perhitungan, satu membaca output, satu membuat kode Python, satu menyusun pembahasan, tetapi semua anggota harus memahami alur umum dan mampu menjelaskan hasil. Laporan kelompok sebaiknya memuat pembagian peran agar akuntabilitas lebih jelas. Dosen dapat meminta presentasi singkat untuk memastikan setiap anggota memahami hasil.

Dalam konteks penilaian, etika akademik juga menuntut dosen bersikap adil dan transparan. Rubrik penilaian perlu disampaikan sejak awal. Mahasiswa perlu tahu bahwa yang dinilai bukan hanya hasil numerik, tetapi juga kerapian input-output, validasi, kejujuran data, sitasi, interpretasi, dan kemampuan menjelaskan keterbatasan. Jika hanya nilai akhir yang ditekankan, mahasiswa terdorong mengejar angka “sesuai teori” dan menyembunyikan proses. Jika proses ilmiah dinilai, mahasiswa terdorong bekerja jujur. Dengan demikian, sistem penilaian harus mendukung budaya integritas.

Etika akademik juga berkaitan dengan **keselamatan dan tanggung jawab sosial**. Praktikum komputasi memang tidak menghasilkan limbah kimia, tidak memerlukan reagen berbahaya, dan lebih aman dibanding banyak eksperimen basah. Dokumen rujukan software kimia komputasi menjelaskan bahwa makmal maya dapat mengeliminasi risiko keselamatan fisik ketika mahasiswa mensimulasikan reaksi ekstrem, gas beracun, atau katalis logam transisi langka, sekaligus selaras dengan prinsip pencegahan limbah dalam Green Chemistry. Namun, keamanan fisik yang lebih tinggi tidak berarti bebas tanggung jawab. Mahasiswa tetap harus bertanggung jawab terhadap penggunaan sumber daya, kebenaran hasil, dan dampak klaim ilmiah yang dibuat.

Dalam pendidikan kimia modern, etika akademik perlu ditempatkan sebagai bagian dari pembentukan karakter ilmuwan. Mahasiswa kimia kuantum dan kimia komputasi tidak hanya belajar menghitung energi molekul. Mereka belajar bagaimana bersikap terhadap kebenaran. Mereka belajar bahwa angka yang bagus tetapi palsu tidak bernilai. Mereka belajar bahwa data yang gagal tetapi jujur lebih berguna daripada data yang dimanipulasi. Mereka belajar bahwa software canggih tidak menggantikan integritas. Mereka belajar bahwa keterampilan teknis harus disertai tanggung jawab moral.

Untuk menerapkan etika akademik secara konkret, setiap laporan praktikum dalam buku ini sebaiknya memuat **Pernyataan Integritas Praktikum**. Contohnya:

```
Saya menyatakan bahwa perhitungan, output, data, grafik, kode Python, dan pembahasan dalam laporan ini disusun secara jujur. Setiap sumber, software, kode, data, dan bantuan pihak lain telah disebutkan sesuai ketentuan. Saya memahami bahwa fabrikasi, falsifikasi, plagiarisme, dan penggunaan software tidak sah merupakan pelanggaran etika akademik.
```

Pernyataan ini bukan sekadar formalitas, tetapi pengingat bahwa praktikum adalah amanah ilmiah. Jika mahasiswa menandatangani pernyataan tersebut, ia harus siap mempertanggungjawabkan file input, output, dan kode yang digunakan.

Selain pernyataan integritas, laporan juga perlu memuat **bagian sitasi software dan sumber data**. Contohnya:

```
Struktur molekul dibangun dengan Avogadro. Perhitungan optimasi geometri dilakukan menggunakan ORCA pada tingkat teori B3LYP/def2-SVP. Analisis data dan grafik dilakukan menggunakan Python dengan pustaka numpy, pandas, dan matplotlib. Visualisasi orbital dilakukan menggunakan Jmol. Semua file input, output, dan kode Python dilampirkan.
```

Kalimat seperti ini membuat proses ilmiah menjadi transparan. Pembaca mengetahui alat yang digunakan, metode yang dipakai, dan file yang tersedia. Jika laporan lebih formal, rujukan software dimasukkan dalam daftar pustaka pada akhir bab atau akhir laporan.

Etika akademik juga memerlukan **bahasa laporan yang jujur**. Mahasiswa perlu menghindari kalimat yang berlebihan. Sebaiknya menulis “hasil menunjukkan kecenderungan”, “pada tingkat teori yang digunakan”, “berdasarkan model ini”, “perbedaan dapat disebabkan oleh”, atau “hasil ini perlu divalidasi lebih lanjut” ketika data memang terbatas. Bahasa ilmiah yang hati-hati bukan tanda kelemahan, tetapi tanda kedewasaan. Sebaliknya, bahasa absolut seperti “terbukti pasti”, “selalu”, atau “sangat akurat” harus dihindari jika tidak didukung data memadai.

Dalam praktik, dosen dapat membuat daftar pelanggaran etik yang jelas. Misalnya, pelanggaran berat meliputi fabrikasi output, falsifikasi data, plagiarisme laporan, penggunaan output kelompok lain tanpa izin, dan penggunaan software bajakan. Pelanggaran sedang dapat mencakup tidak mencantumkan sitasi software, tidak melampirkan input-output, atau tidak menyebut sumber kode. Pelanggaran ringan dapat berupa salah format sitasi atau penamaan file kurang jelas. Klasifikasi ini membantu mahasiswa memahami konsekuensi dan mendorong perbaikan sejak awal.

Etika akademik dalam praktikum komputasi pada akhirnya tidak hanya melindungi nilai mata kuliah, tetapi juga melindungi masa depan ilmiah mahasiswa. Mahasiswa yang terbiasa memalsukan data pada praktikum kecil berisiko membawa kebiasaan itu ke skripsi, tesis, artikel, atau pekerjaan profesional. Sebaliknya, mahasiswa yang terbiasa jujur terhadap output gagal, teliti mencatat input, disiplin menyebut software, dan rendah hati dalam menyimpulkan akan memiliki dasar kuat sebagai peneliti, guru, dosen, analis, atau praktisi kimia komputasi. Integritas adalah kompetensi ilmiah yang sama pentingnya dengan kemampuan menjalankan software.

Dalam konteks buku ini, etika akademik menjadi penutup Bab 1 karena seluruh fondasi praktikum bermuara pada tanggung jawab. Hakikat praktikum kimia kuantum telah dijelaskan sebagai kegiatan ilmiah berbasis teori, software, dan analisis. Hubungan kimia kuantum, kimia komputasi, dan programming ilmiah telah menunjukkan pentingnya integrasi pengetahuan dan keterampilan. Persamaan Schrödinger telah diletakkan sebagai fondasi teoritis. Jenis perhitungan telah dikenalkan sebagai bentuk kerja praktikum. Format file telah dijelaskan sebagai infrastruktur data. Reprodusibilitas telah dibahas sebagai syarat validasi. Etika akademik kemudian menegaskan bahwa seluruh rangkaian itu harus dijalankan dengan kejujuran, tanggung jawab, dan penghormatan terhadap ilmu.

Dengan demikian, mahasiswa yang memasuki praktikum kimia kuantum harus membawa tiga bekal utama: pemahaman teori, keterampilan teknis, dan integritas akademik. Tanpa teori, praktikum menjadi prosedur kosong. Tanpa keterampilan teknis, teori sulit diterapkan. Tanpa integritas, seluruh hasil kehilangan makna. Bab berikutnya akan membawa mahasiswa masuk ke ekosistem software, ChemCompute, dan alur kerja praktikum komputasi, tetapi semua penggunaan software tersebut harus tetap berpijak pada prinsip etika yang telah dibahas di sini.

Penutup Bab 1

Bab 1 telah memperkenalkan landasan umum buku **Praktikum Kimia Kuantum: Panduan Praktikum In Silico Berbasis ChemCompute, Python, dan Software Kimia Komputasi untuk Memahami Struktur Elektronik Molekul**. Bab ini menegaskan bahwa praktikum kimia kuantum tidak cukup dipahami sebagai kegiatan menjalankan software, memilih menu, atau menyalin angka energi dari output. Praktikum kimia kuantum harus dipahami sebagai proses ilmiah yang menghubungkan **teori mekanika kuantum**, **model struktur elektronik**, **representasi matematis molekul**, **perangkat lunak komputasi**, **analisis data**, dan **interpretasi kimia**.

Gagasan utama Bab 1 adalah bahwa kimia komputasi memberikan cara baru untuk mempelajari molekul melalui pendekatan in silico. Dalam pendekatan ini, molekul tidak hanya dipahami melalui rumus struktur dua dimensi, model bola-tongkat, atau persamaan teoritis di papan tulis, tetapi juga melalui perhitungan numerik yang menghasilkan energi, geometri, orbital, muatan, momen dipol, spektrum, dan berbagai sifat molekul lain. Dengan demikian, praktikum kimia kuantum menjadi jembatan antara teori abstrak dan pengalaman analitis yang konkret.

Bab ini juga menegaskan pentingnya literasi komputasi bagi mahasiswa kimia. Pada masa kini, pemahaman kimia modern semakin banyak melibatkan kemampuan membaca data, menjalankan simulasi, mengolah hasil komputasi, membuat visualisasi, serta menilai validitas metode. Mahasiswa tidak hanya dituntut memahami konsep orbital, energi, ikatan, dan struktur molekul, tetapi juga harus mampu menjelaskan bagaimana konsep tersebut dihitung, dengan metode apa, menggunakan basis set apa, melalui software apa, dan dengan keterbatasan apa.

ChemCompute, Python, dan software kimia komputasi seperti GAMESS, ORCA, Gaussian, Psi4, PySCF, Avogadro, atau perangkat sejenis diposisikan sebagai alat pembelajaran. Alat-alat tersebut tidak menggantikan teori, tetapi membantu mahasiswa mengalami teori secara operasional. ChemCompute memudahkan akses perhitungan berbasis web. Python membantu mahasiswa

mengolah data, membuat tabel, membuat grafik, dan menyusun analisis. Software kimia komputasi menjalankan perhitungan struktur elektronik yang menjadi inti praktikum. Dengan kombinasi ini, mahasiswa belajar bahwa kimia kuantum bukan hanya kumpulan persamaan, tetapi juga suatu cara berpikir ilmiah yang dapat diuji, dihitung, divisualisasikan, dan dikritisi.

Bab 1 juga menekankan bahwa setiap hasil komputasi harus dibaca secara hati-hati. Energi total, panjang ikatan, sudut ikatan, orbital molekul, momen dipol, atau spektrum IR tidak boleh dianggap sebagai hasil mutlak tanpa konteks. Setiap output selalu bergantung pada metode, basis set, geometri awal, parameter perhitungan, status konvergensi, dan kualitas input. Oleh karena itu, mahasiswa perlu dilatih sejak awal untuk menulis laporan secara lengkap: menyebut molekul yang dihitung, metode yang digunakan, basis set, software, jenis job, status konvergensi, satuan, serta interpretasi kimianya.

Secara pedagogis, Bab 1 menjadi pintu masuk menuju seluruh rangkaian praktikum. Bab ini menyiapkan mahasiswa agar tidak hanya menjadi pengguna software, tetapi menjadi pembaca kritis hasil komputasi. Sikap kritis ini penting karena software dapat menghasilkan angka, tetapi tidak dapat menggantikan penalaran kimia. Mahasiswa harus mampu membedakan antara hasil yang valid dan hasil yang meragukan, antara visualisasi yang informatif dan visualisasi yang menyesatkan, serta antara kesimpulan yang didukung data dan kesimpulan yang terlalu berlebihan.

Dengan demikian, Bab 1 meletakkan fondasi konseptual, metodologis, dan pedagogis bagi bab-bab berikutnya. Setelah memahami orientasi umum praktikum kimia kuantum in silico, mahasiswa siap memasuki praktikum dasar tentang instalasi, pengenalan software, pembangunan struktur molekul, optimasi geometri, metode Hartree–Fock, fungsi basis, DFT, orbital molekul, momen dipol, spektrum vibrasi, dan analisis sifat molekul lainnya.

Bab ini dapat disimpulkan dengan satu pernyataan utama: **praktikum kimia kuantum adalah latihan berpikir ilmiah berbasis model, data, dan interpretasi**. Software hanyalah alat; pemahaman kimia tetap menjadi pusatnya.

Rangkuman Bab 1

1. **Kimia kuantum** mempelajari struktur dan sifat molekul berdasarkan prinsip mekanika kuantum, terutama perilaku elektron dalam atom dan molekul.
2. **Kimia komputasi** memungkinkan konsep kimia kuantum diterapkan melalui perhitungan numerik menggunakan software, sehingga mahasiswa dapat mempelajari energi, geometri, orbital, muatan, dan sifat molekul secara lebih konkret.
3. **Praktikum in silico** merupakan praktikum berbasis simulasi dan komputasi yang melengkapi praktikum laboratorium basah, terutama untuk memahami konsep yang sulit diamati langsung.
4. **ChemCompute** digunakan sebagai platform web untuk memudahkan akses perhitungan kimia komputasi tanpa instalasi lokal yang rumit.
5. **Python** digunakan sebagai alat analisis data, pembuatan tabel, visualisasi grafik, perhitungan numerik, dan dokumentasi hasil praktikum.

6. **Software kimia komputasi** seperti GAMESS, ORCA, Gaussian, Psi4, PySCF, Avogadro, dan perangkat lain digunakan sesuai tujuan praktikum, tingkat kesulitan, dan ketersediaan fasilitas.
7. **Hasil komputasi harus selalu ditafsirkan dengan konteks**, yaitu metode, basis set, geometri, muatan, multiplicity, jenis job, status konvergensi, dan batasan teoritis.
8. **Tujuan utama praktikum kimia kuantum bukan sekadar mendapatkan angka**, tetapi memahami hubungan antara model teoritis, prosedur komputasi, output numerik, dan makna kimia.
9. **Laporan praktikum harus bersifat ilmiah, tertib, dan reprodusibel**, sehingga data, input, output, kode, tabel, grafik, dan kesimpulan dapat ditelusuri ulang.
10. **Bab 1 menjadi dasar bagi seluruh bab berikutnya**, terutama dalam membangun sikap kritis, literasi data, dan keterampilan menggunakan software kimia komputasi secara bertanggung jawab.

Implikasi Pembelajaran Bab 1

Bab 1 memberikan beberapa implikasi penting bagi pembelajaran kimia kuantum dan kimia komputasi.

Pertama, mahasiswa perlu memahami bahwa konsep kimia kuantum yang abstrak dapat dipelajari secara lebih konkret melalui simulasi. Orbital molekul, distribusi elektron, energi total, dan spektrum tidak lagi hanya menjadi konsep teoritis, tetapi dapat divisualisasikan dan dianalisis melalui perangkat komputasi.

Kedua, mahasiswa perlu mengembangkan keterampilan teknis dan konseptual secara bersamaan. Kemampuan menjalankan software tanpa memahami teori akan menghasilkan penggunaan yang mekanis. Sebaliknya, pemahaman teori tanpa keterampilan komputasi akan membuat mahasiswa sulit menerapkan konsep dalam konteks modern. Praktikum ini berupaya menggabungkan keduanya.

Ketiga, mahasiswa harus dibiasakan dengan prinsip reprodusibilitas. File input, output, kode Python, tabel CSV, grafik, dan laporan harus disusun secara tertib. Praktikum kimia komputasi yang baik bukan hanya menghasilkan jawaban, tetapi juga meninggalkan jejak ilmiah yang dapat diperiksa ulang.

Keempat, pembelajaran kimia komputasi harus menanamkan sikap kritis. Mahasiswa harus memahami bahwa setiap metode memiliki batas. Hartree–Fock, DFT, semiempiris, molecular mechanics, dan metode lain memiliki asumsi dan wilayah penerapan masing-masing. Karena itu, pemilihan metode harus selalu disesuaikan dengan tujuan.

Kelima, praktikum kimia kuantum dapat menjadi sarana penguatan literasi data. Mahasiswa belajar membaca output, menyusun dataset kecil, menghitung perubahan energi, membuat grafik, dan menarik kesimpulan berdasarkan bukti. Keterampilan ini penting tidak hanya dalam kimia komputasi, tetapi juga dalam riset sains modern secara umum.

Arah Menuju Bab 2

Setelah Bab 1 memberikan orientasi umum, Bab 2 dapat diarahkan pada pengenalan lingkungan kerja praktikum, instalasi perangkat lunak, struktur folder, format file molekul, penggunaan ChemCompute, penggunaan Python, serta prinsip dasar pengelolaan data praktikum. Bab berikutnya perlu memastikan bahwa mahasiswa siap secara teknis sebelum masuk ke perhitungan molekul yang lebih spesifik.

Dengan demikian, Bab 1 berfungsi sebagai fondasi konseptual, sedangkan Bab 2 mulai mengarahkan mahasiswa pada kesiapan operasional. Mahasiswa yang memahami Bab 1 akan lebih siap mengikuti praktikum karena telah mengetahui mengapa software digunakan, apa yang harus dicatat, bagaimana hasil harus ditafsirkan, dan mengapa setiap perhitungan harus dilaporkan secara lengkap.

Daftar Pustaka Bab 1

Atkins, P. W., & Friedman, R. S. (2011). *Molecular Quantum Mechanics* (5th ed.). Oxford University Press.

Cramer, C. J. (2004). *Essentials of Computational Chemistry: Theories and Models* (2nd ed.). John Wiley & Sons.

Engel, T., & Reid, P. (2013). *Physical Chemistry* (3rd ed.). Pearson.

Feller, D., & Dixon, D. A. (2001). The impact of larger basis sets and explicitly correlated wave functions on the coupled cluster calculation of small molecules. *The Journal of Physical Chemistry A, 105*(25), 6201–6211.

Foresman, J. B., & Frisch, Æ. (2015). *Exploring Chemistry with Electronic Structure Methods* (3rd ed.). Gaussian, Inc.

Gordon, M. S., & Schmidt, M. W. (2005). Advances in electronic structure theory: GAMESS a decade later. In C. E. Dykstra, G. Frenking, K. S. Kim, & G. E. Scuseria (Eds.), *Theory and Applications of Computational Chemistry: The First Forty Years* (pp. 1167–1189). Elsevier.

Helgaker, T., Jørgensen, P., & Olsen, J. (2000). *Molecular Electronic-Structure Theory*. John Wiley & Sons.

Jensen, F. (2017). *Introduction to Computational Chemistry* (3rd ed.). John Wiley & Sons.

Koch, W., & Holthausen, M. C. (2001). *A Chemist's Guide to Density Functional Theory* (2nd ed.). Wiley-VCH.

Leach, A. R. (2001). *Molecular Modelling: Principles and Applications* (2nd ed.). Prentice Hall.

Levine, I. N. (2014). *Quantum Chemistry* (7th ed.). Pearson.

Lewars, E. G. (2016). *Computational Chemistry: Introduction to the Theory and Applications of Molecular and Quantum Mechanics* (3rd ed.). Springer.

McQuarrie, D. A., & Simon, J. D. (1997). *Physical Chemistry: A Molecular Approach*. University Science Books.

Neese, F. (2012). The ORCA program system. *WIREs Computational Molecular Science, 2*(1), 73–78.

Neese, F. (2022). Software update: The ORCA program system—Version 5.0. *WIREs Computational Molecular Science, 12*(5), e1606.

Ochterski, J. W. (2000). *Thermochemistry in Gaussian*. Gaussian, Inc.

Parr, R. G., & Yang, W. (1989). *Density-Functional Theory of Atoms and Molecules*. Oxford University Press.

Roothaan, C. C. J. (1951). New developments in molecular orbital theory. *Reviews of Modern Physics, 23*(2), 69–89.

Schmidt, M. W., Baldridge, K. K., Boatz, J. A., Elbert, S. T., Gordon, M. S., Jensen, J. H., Koseki, S., Matsunaga, N., Nguyen, K. A., Su, S. J., Windus, T. L., Dupuis, M., & Montgomery, J. A. (1993). General atomic and molecular electronic structure system. *Journal of Computational Chemistry, 14*(11), 1347–1363.

Sherrill, C. D. (2010). Frontiers in electronic structure theory. *The Journal of Chemical Physics, 132*(11), 110902.

Sun, Q., Berkelbach, T. C., Blunt, N. S., Booth, G. H., Guo, S., Li, Z., Liu, J., McClain, J. D., Sayfutyarova, E. R., Sharma, S., Wouters, S., & Chan, G. K.-L. (2018). PySCF: The Python-based simulations of chemistry framework. *WIREs Computational Molecular Science, 8*(1), e1340.

Szabo, A., & Ostlund, N. S. (1996). *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Dover Publications.

Young, D. C. (2001). *Computational Chemistry: A Practical Guide for Applying Techniques to Real-World Problems*. John Wiley & Sons.

BAB 2

EKOSISTEM SOFTWARE, CHEMCOMPUTE, DAN ALUR KERJA PRAKTIKUM KOMPUTASI

Pengantar Bab 2

Praktikum kimia kuantum modern tidak dapat dilepaskan dari ekosistem software. Jika Bab 1 telah membangun fondasi konseptual tentang hakikat praktikum kimia kuantum, hubungan kimia kuantum dengan kimia komputasi dan programming ilmiah, Persamaan Schrödinger, jenis perhitungan, format file, reprodusibilitas, serta etika akademik, maka Bab 2 mulai membawa mahasiswa ke wilayah operasional: perangkat apa yang digunakan, bagaimana alur kerjanya, bagaimana memilih software sesuai tujuan praktikum, dan bagaimana menghubungkan software visualisasi, platform komputasi daring, serta Python dalam satu sistem kerja yang terarah.

Ekosistem software kimia komputasi perlu dipahami sebagai jaringan alat yang saling melengkapi. Tidak ada satu software yang ideal untuk semua kebutuhan praktikum. Ada software yang kuat untuk membangun dan memvisualisasikan struktur molekul, ada yang berfungsi sebagai mesin perhitungan struktur elektronik, ada yang berperan sebagai antarmuka grafis, ada yang cocok untuk analisis orbital dan rapat elektron, ada yang bekerja berbasis web, dan ada pula yang dirancang sebagai pustaka Python untuk pengembangan metode serta otomasi perhitungan. Oleh karena itu, mahasiswa tidak cukup hanya mengetahui nama software. Mahasiswa harus memahami fungsi, keunggulan, keterbatasan, konteks penggunaan, serta posisi setiap software dalam alur praktikum.

Bab ini menempatkan **ChemCompute** sebagai salah satu gerbang penting karena platform tersebut memungkinkan praktikum kimia komputasi dilakukan berbasis web. ChemCompute relevan untuk pendidikan karena dapat memperluas akses mahasiswa terhadap perhitungan kimia komputasi tanpa instalasi lokal yang rumit. ChemCompute juga menyediakan rujukan sitasi untuk penggunaan *web-based job submission interface* bagi GAMESS, serta mengaitkannya dengan praktik komputasi kimia berbasis web untuk pembelajaran. (ChemCompute) Artikel terbaru tentang ChemCompute juga menggambarkannya sebagai platform web gratis yang menyediakan akses ke berbagai perangkat lunak struktur elektronik dan dinamika molekul sumber terbuka untuk mengeksplorasi struktur serta sifat sistem kimia dalam konteks laboratorium sarjana. (Springer Nature Link)

Namun, ChemCompute tidak berdiri sendiri. Dalam buku praktikum ini, ChemCompute dipadukan dengan Avogadro untuk membangun struktur molekul, ORCA/GAMESS/Gaussian/Psi4/PySCF untuk perhitungan struktur elektronik, Jmol/Molden/Multiwfn/Gabedit/GaussView untuk visualisasi dan analisis, serta Python untuk memproses data, membuat grafik, menyusun tabel, dan melakukan validasi hasil. Dokumen rujukan software kimia komputasi juga menegaskan bahwa integrasi komputasi ke dalam kurikulum bukan sekadar mengganti alat bantu, tetapi mengubah cara pengetahuan kimia diproduksi, divalidasi, dan ditransmisikan. Karena itu, pembahasan software dalam bab ini tidak

diarahkan menjadi katalog teknis, melainkan peta epistemik: software dilihat sebagai bagian dari proses ilmiah.

2.1 Peta Software Kimia Komputasi

Peta software kimia komputasi adalah pengelompokan perangkat lunak berdasarkan fungsi ilmiah, cara penggunaan, tingkat akses, jenis perhitungan, format file, dan peran pedagogisnya dalam praktikum. Peta ini penting karena mahasiswa sering kali mengenal software secara terpisah, misalnya Avogadro untuk menggambar molekul, Gaussian untuk menghitung energi, Jmol untuk melihat struktur, atau Python untuk membuat grafik. Padahal dalam praktikum yang benar, software-software tersebut bekerja sebagai satu rantai alur kerja: struktur dibangun, input disusun, perhitungan dijalankan, output dibaca, data diekstrak, hasil divisualisasikan, lalu kesimpulan ditulis secara ilmiah.

Secara garis besar, software kimia komputasi dalam buku ini dapat dipetakan ke dalam enam kategori. Pertama, **platform komputasi berbasis web**, yaitu ChemCompute. Kedua, **editor dan visualizer struktur molekul**, seperti Avogadro, GaussView, Jmol, dan Molden. Ketiga, **mesin perhitungan kimia kuantum**, seperti ORCA, GAMESS, Gaussian, Psi4, dan PySCF. Keempat, **antarmuka grafis dan penghubung input-output**, seperti Gabedit dan GaussView. Kelima, **alat analisis fungsi gelombang, orbital, dan rapat elektron**, seperti Multiwfn, Molden, Jmol, dan sebagian fitur GaussView/Gabedit. Keenam, **lingkungan programming ilmiah**, terutama Python, yang walaupun dibahas lebih rinci pada bab berikutnya, tetap perlu ditempatkan dalam peta software sejak awal.

Peta ini dapat diringkas sebagai berikut.

Kelompok Software	Contoh Software	Fungsi Utama dalam Praktikum
Platform komputasi web	ChemCompute	Menjalankan job komputasi melalui web tanpa instalasi lokal rumit
Editor molekul	Avogadro, GaussView, Gabedit	Membangun struktur, menambah hidrogen, membuat input awal
Mesin perhitungan	ORCA, GAMESS, Gaussian, Psi4, PySCF	Menghitung energi, optimasi geometri, frekuensi, orbital, sifat molekul
Visualisasi struktur	Jmol, Molden, Avogadro, GaussView	Menampilkan struktur 3D, orbital, mode vibrasi, hasil output
Analisis lanjutan	Multiwfn, Molden, Gabedit	Analisis orbital, rapat elektron, muatan, potensial elektrostatik

Kelompok Software	Contoh Software	Fungsi Utama dalam Praktikum
Programming ilmiah	Python, Jupyter Notebook	Mengolah data, membuat grafik, parsing output, simulasi konsep

Peta tersebut tidak dimaksudkan untuk mengunci fungsi software secara kaku. Banyak software memiliki fungsi ganda. Avogadro, misalnya, bukan hanya editor molekul, tetapi juga visualizer dan pembuat input untuk beberapa program. Gabedit dapat berperan sebagai antarmuka grafis, pembuat input, sekaligus post-processor. GaussView tidak hanya membangun molekul, tetapi juga menyiapkan input Gaussian dan memvisualisasikan hasil. Psi4 dan PySCF bukan hanya mesin hitung, tetapi juga dapat diintegrasikan dengan Python untuk workflow yang dapat diotomatisasi. Dengan demikian, peta software harus dipahami secara fungsional, bukan semata-mata berdasarkan nama program.

1. ChemCompute: Platform Komputasi Berbasis Web

ChemCompute ditempatkan pertama karena buku ini memang dirancang agar praktikum dapat dijalankan oleh mahasiswa dengan hambatan instalasi seminimal mungkin. Dalam banyak kampus, masalah utama praktikum kimia komputasi bukan hanya teori, tetapi infrastruktur. Instalasi software kuantum dapat rumit, terutama jika memerlukan Linux, konfigurasi PATH, pustaka numerik, memori cukup besar, atau lisensi tertentu. ChemCompute membantu mengurangi hambatan tersebut dengan menyediakan akses berbasis web untuk menjalankan paket komputasi tertentu, terutama dalam konteks pembelajaran dan riset sarjana. ChemCompute sendiri menyediakan panduan sitasi untuk penggunaan platformnya, termasuk rujukan Perri dan Weber tentang antarmuka web untuk pengiriman job GAMESS. (ChemCompute)

Dalam praktikum, ChemCompute dapat berfungsi sebagai “laboratorium komputasi daring”. Mahasiswa dapat menyusun atau mengunggah input, mengirim job, menunggu perhitungan selesai, lalu mengunduh output. Keunggulan utamanya adalah aksesibilitas. Mahasiswa tidak harus memasang GAMESS atau paket lain secara lokal. Mereka juga tidak harus langsung berhadapan dengan konfigurasi sistem operasi. Dengan demikian, energi belajar dapat difokuskan pada pemahaman konsep: struktur molekul, metode, basis set, charge, multiplicity, jenis job, output, dan interpretasi. Artikel tentang ChemCompute juga menekankan potensinya untuk memperkenalkan laboratorium kimia komputasi berkualitas tinggi pada tingkat sarjana tanpa hambatan finansial besar. (Springer Nature Link)

Akan tetapi, ChemCompute tidak boleh dipahami sebagai alat ajaib yang membebaskan mahasiswa dari pemahaman input. Justru karena platform web menyederhanakan akses, dosen harus memastikan mahasiswa tetap membaca file input dan output. Mahasiswa perlu mengetahui apakah job yang dijalankan adalah optimasi, single point energy, frekuensi, atau perhitungan lain. Mahasiswa juga harus mencatat metode dan basis set, menyimpan output, serta memeriksa status konvergensi. Dengan demikian, ChemCompute berperan sebagai sarana demokratisasi akses, bukan pengganti literasi kimia komputasi.

Dalam buku ini, ChemCompute akan ditempatkan sebagai jalur praktikum utama untuk mahasiswa yang belum memiliki instalasi lokal. Namun, setiap hasil dari ChemCompute tetap harus dianalisis dengan Python atau software visualisasi lain. Dengan pola ini, mahasiswa memperoleh dua kompetensi sekaligus: menjalankan perhitungan komputasi dan menganalisis hasilnya secara mandiri.

2. Avogadro: Editor dan Visualizer Molekul

Avogadro adalah salah satu software paling penting untuk tahap awal praktikum. Ia digunakan untuk membangun struktur molekul, menambah atom hidrogen, membersihkan geometri awal, mengatur konformasi, menyimpan file XYZ/MOL/SDF, dan dalam banyak kasus membuat input untuk program komputasi lain. Dokumentasi Avogadro menyebutnya sebagai editor molekul dan alat visualisasi yang bebas serta open-source, digunakan dalam kimia komputasi, pemodelan molekul, pendidikan kimia, bioinformatika, ilmu material, dan bidang lain. (Avogadro)

Keunggulan pedagogis Avogadro terletak pada kemudahan visualnya. Mahasiswa pemula sering kesulitan membayangkan struktur molekul tiga dimensi hanya dari rumus struktur dua dimensi. Dengan Avogadro, mahasiswa dapat menggambar air, amonia, metana, etanol, benzena, formaldehida, atau nitrobenzena, lalu memutar struktur dalam ruang 3D. Mereka dapat melihat bentuk molekul, sudut ikatan, torsi, dan posisi atom. Dalam konteks pembelajaran kimia kuantum, pengalaman visual semacam ini penting karena banyak sifat elektronik sangat bergantung pada geometri.

Dokumen rujukan software kimia komputasi juga menyinggung penggunaan Avogadro dalam model pembelajaran BYOD (*Bring Your Own Device*) bersama ORCA. Pada tahap awal, mahasiswa dapat mengunduh dan menjalankan simulasi termokimia dasar seperti optimasi geometri H_2O atau frekuensi vibrasi etanol di laptop masing-masing, sehingga keterbatasan jumlah komputer laboratorium dapat dikurangi. Ini menunjukkan bahwa Avogadro tidak hanya berguna sebagai alat gambar, tetapi juga sebagai pintu masuk pedagogis menuju penalaran komputasional.

Namun, Avogadro tetap memiliki batas. Struktur yang dibuat di Avogadro adalah struktur awal, bukan hasil kuantum final. Fitur “optimize geometry” berbasis force field dapat membantu memperbaiki struktur kasar, tetapi tidak sama dengan optimasi geometri kuantum menggunakan HF, DFT, atau metode ab initio. Mahasiswa harus memahami bahwa struktur Avogadro sebaiknya digunakan sebagai input awal, kemudian dihitung lebih lanjut dengan software struktur elektronik. Jika struktur awal sangat buruk, perhitungan kuantum dapat gagal atau menuju minimum yang tidak diinginkan. Karena itu, penggunaan Avogadro harus disertai pemeriksaan kimiawi: valensi benar, hidrogen lengkap, muatan masuk akal, dan konformasi sesuai tujuan praktikum.

3. ORCA: Mesin Perhitungan Struktur Elektronik Akademik

ORCA adalah paket kimia kuantum yang banyak digunakan untuk perhitungan struktur elektronik, spektroskopi, sistem open-shell, kompleks logam transisi, DFT, Hartree–Fock, metode pasca-Hartree–Fock, dan berbagai sifat molekul. Situs resmi ORCA menjelaskan bahwa ORCA merupakan paket software kimia kuantum yang kuat dan serbaguna, dikembangkan terutama oleh kelompok Prof. Frank Neese, serta gratis untuk penggunaan akademik. (FACCTs)

Dalam praktikum kimia kuantum, ORCA memiliki beberapa keunggulan. Sintaks inputnya relatif ringkas, sehingga mahasiswa dapat memahami hubungan antara baris perintah dan jenis perhitungan. Misalnya, input `! B3LYP def2-SVP Opt Freq` sudah menunjukkan metode DFT B3LYP, basis set def2-SVP, optimasi geometri, dan perhitungan frekuensi. ORCA juga sangat cocok untuk mengenalkan mahasiswa pada logika file input berbasis teks. Mahasiswa dapat melihat bahwa perhitungan kimia kuantum bukan sekadar klik menu, tetapi instruksi eksplisit yang dapat ditulis, diperiksa, dan disimpan.

Dokumen rujukan software kimia komputasi menempatkan ORCA sebagai salah satu software penting dalam perhitungan ab initio dan DFT; ORCA digambarkan menonjol karena gratis untuk kepentingan akademik, efisien dalam komputasi paralel, dan kuat pada perhitungan spektroskopi serta sistem molekul bercangkang terbuka. Namun, keunggulan ORCA juga menuntut kedisiplinan. Mahasiswa harus memahami charge, multiplicity, metode, basis set, kriteria konvergensi, serta pesan output. ORCA dapat memberikan hasil sangat kaya, tetapi output yang kaya hanya bermanfaat jika mahasiswa tahu bagian mana yang perlu dibaca.

ORCA sangat cocok digunakan dalam buku ini untuk praktikum optimasi geometri, frekuensi vibrasi, HOMO-LUMO, momen dipol, dan energi reaksi. Jika laboratorium memiliki komputer lokal dengan ORCA terpasang, mahasiswa dapat belajar alur kerja profesional: membuat input, menjalankan job di terminal, membaca output, memvisualisasikan hasil, lalu menganalisis data dengan Python. Jika instalasi lokal belum tersedia, prinsip input ORCA tetap dapat diajarkan sebagai literasi file komputasi.

4. GAMESS: Paket Ab Initio dan DFT yang Kuat untuk Pembelajaran

GAMESS atau *General Atomic and Molecular Electronic Structure System* adalah paket kimia kuantum umum untuk perhitungan struktur elektronik. Situs resmi kelompok Gordon di Iowa State University menyebut GAMESS sebagai paket kimia kuantum ab initio umum yang dipelihara oleh anggota kelompok riset Gordon. (MSG Chem) Ames Laboratory juga menjelaskan GAMESS sebagai kode struktur elektronik serbaguna untuk kimia komputasi yang berfokus pada ab initio, serta dapat melakukan DFT, semiempiris, QM/MM, dan menangani efek pelarut. (Ames Lab)

Dalam konteks buku ini, GAMESS penting karena berhubungan erat dengan ChemCompute. ChemCompute menyediakan jalur web untuk menjalankan GAMESS sehingga mahasiswa tidak harus melakukan instalasi lokal. Ini membuat GAMESS sangat relevan untuk praktikum sarjana, terutama pada topik optimasi geometri, single point energy, frekuensi, dan perhitungan sifat molekul sederhana. Struktur input GAMESS memang lebih teknis dibanding ORCA, karena menggunakan blok seperti `$CONTRL`, `$BASIS`, dan `$DATA`. Namun, justru di situlah nilai pedagogisnya: mahasiswa belajar bahwa perhitungan kimia komputasi tersusun dari blok instruksi yang eksplisit.

Kelebihan GAMESS adalah kematangan metodologis dan cakupan perhitungan yang luas. Kekurangannya bagi pemula adalah sintaks input yang dapat terasa lebih kaku. Oleh karena itu, dalam praktikum, GAMESS lebih cocok diperkenalkan melalui ChemCompute atau template input yang sudah dijelaskan baris demi baris. Mahasiswa tidak boleh hanya menyalin template.

Mereka perlu memahami arti `RUNTYP`, `SCFTYP`, basis set, simetri, koordinat, charge, dan multiplicity. Dengan demikian, penggunaan GAMESS dapat melatih kedisiplinan membaca input.

5. Gaussian dan GaussView: Standar Komersial dalam Struktur Elektronik dan Visualisasi

Gaussian adalah salah satu paket kimia komputasi paling berpengaruh dalam sejarah perhitungan struktur elektronik. Dokumen rujukan software kimia komputasi menyebut bahwa rilis Gaussian 70 oleh John Pople menjadi tonggak penting karena memperluas akses perhitungan ab initio bagi ahli kimia. Situs resmi Gaussian menggambarkan Gaussian 16 sebagai bagian dari seri program struktur elektronik yang digunakan oleh kimiawan, insinyur kimia, biokimiawan, fisikawan, dan ilmuwan lain. (Gaussian)

Gaussian kuat untuk berbagai jenis perhitungan struktur elektronik, seperti optimasi geometri, frekuensi, DFT, metode pasca-HF, perhitungan termokimia, spektrum, dan sifat molekul. Dalam pendidikan kimia komputasi, Gaussian sering dipadukan dengan GaussView. GaussView adalah antarmuka grafis yang digunakan untuk membangun molekul, menyiapkan input Gaussian, menjalankan job dalam lingkungan tertentu, serta memvisualisasikan hasil seperti struktur teroptimasi, orbital molekul, kerapatan elektron, mode vibrasi, dan potensial elektrostatik. Sumber dokumentasi vendor menyebut bahwa GaussView dapat memvisualisasikan berbagai hasil Gaussian, termasuk struktur teroptimasi, orbital molekul, rapat elektron, dan potensial elektrostatik. (Additive Net)

Bagi praktikum, Gaussian dan GaussView sangat baik jika institusi memiliki lisensi resmi. Mahasiswa dapat belajar alur visual yang lengkap: membangun molekul, memilih metode, menjalankan perhitungan, dan melihat hasil. Namun, karena Gaussian dan GaussView merupakan software komersial, penggunaannya harus mengikuti lisensi. Buku ini tidak menjadikan Gaussian sebagai satu-satunya jalur, tetapi menempatkannya sebagai salah satu opsi bagi kampus yang memiliki akses legal. Jalur alternatif seperti ChemCompute, GAMESS, ORCA, Psi4, dan PySCF tetap disediakan agar praktikum lebih inklusif.

Secara pedagogis, penggunaan Gaussian/GaussView perlu dijaga agar tidak berubah menjadi *button-clicking chemistry*. Mahasiswa harus tetap melihat file input `.gjf` atau `.com`, memahami route section, charge, multiplicity, metode, basis set, dan jenis job. GaussView memudahkan proses, tetapi pemahaman input-output tetap wajib. Dalam laporan, mahasiswa harus mencantumkan tingkat teori secara lengkap, misalnya B3LYP/6-31G(d), bukan hanya "menggunakan Gaussian".

6. Gabedit: Antarmuka Grafis untuk Preprocessing dan Postprocessing

Gabedit adalah antarmuka grafis yang dirancang untuk membantu preprocessing dan postprocessing berbagai software kimia komputasi. Artikel Allouche (2011) menjelaskan Gabedit sebagai GUI freeware yang menyediakan preprocessing dan postprocessing untuk beberapa paket software kimia komputasi. (PubMed) Dalam praktiknya, Gabedit dapat digunakan untuk membangun molekul, menyiapkan input, menjalankan atau menghubungkan job tertentu, serta memvisualisasikan hasil seperti orbital, kerapatan elektron, dan mode vibrasi.

Kekuatan Gabedit dalam praktikum terletak pada posisinya sebagai penghubung. Mahasiswa yang belum nyaman menulis input secara penuh dapat menggunakan Gabedit untuk memahami struktur input. Namun, seperti GaussView dan Avogadro, Gabedit tidak boleh menjadikan mahasiswa pasif. Setiap input yang dihasilkan harus dibaca kembali. Setiap output yang divisualisasikan harus ditafsirkan secara konseptual. Gabedit sangat berguna untuk membantu mahasiswa memahami bahwa satu software dapat berfungsi sebagai jembatan antara struktur visual, mesin hitung, dan analisis hasil.

Dalam peta software buku ini, Gabedit ditempatkan sebagai alat pendukung yang sangat berguna untuk kampus yang ingin mengurangi hambatan teknis tanpa menghilangkan pemahaman input-output. Gabedit juga relevan untuk memvisualisasikan orbital dan mode vibrasi dari output tertentu. Dengan demikian, Gabedit dapat memperkaya praktikum pada topik optimasi geometri, frekuensi IR, dan visualisasi orbital.

7. Psi4: Quantum Chemistry Open-Source dengan Integrasi Python

Psi4 adalah software kimia kuantum open-source yang kuat dan relevan untuk praktikum modern. Artikel Smith dkk. (2020) menjelaskan Psi4 sebagai program struktur elektronik ab initio gratis dan open-source yang menyediakan implementasi Hartree–Fock, DFT, perturbasi banyak-badan, configuration interaction, coupled-cluster, dan metode lain. (PMC) Situs resmi Psi4 juga menekankan bahwa Psi4 merupakan inti C++/Python yang dapat diintegrasikan dan diperluas oleh proyek komunitas dalam ekosistem software. (PsiCode)

Keunggulan pedagogis Psi4 adalah integrasinya dengan Python. Dalam praktikum yang menekankan programming ilmiah, Psi4 memungkinkan mahasiswa menjalankan perhitungan kuantum langsung dari skrip Python atau Jupyter Notebook. Ini membuka jalan bagi workflow yang lebih otomatis. Misalnya, mahasiswa dapat menulis Python untuk menghitung energi H_2 pada berbagai panjang ikatan, memanggil Psi4 untuk setiap jarak, lalu membuat grafik kurva energi potensial. Dengan software tradisional berbasis input-output manual, tugas ini lebih sulit. Dengan Psi4, hubungan antara perhitungan dan analisis menjadi lebih rapat.

Namun, Psi4 membutuhkan kesiapan teknis yang lebih tinggi dibanding ChemCompute. Instalasi, konfigurasi lingkungan Python, dan pemahaman API dapat menjadi hambatan bagi mahasiswa pemula. Oleh sebab itu, dalam buku ini, Psi4 lebih cocok ditempatkan sebagai jalur lanjutan atau alternatif bagi kelas yang sudah siap dengan Python. Pada tingkat dasar, mahasiswa dapat mengenal konsepnya terlebih dahulu: perhitungan kuantum dapat dipanggil dari kode, bukan hanya dari file input statis.

8. PySCF: Kerangka Simulasi Kimia Berbasis Python

PySCF atau *Python-based Simulations of Chemistry Framework* adalah kerangka kerja kimia komputasi berbasis Python yang dirancang untuk fleksibilitas pengembangan metode dan workflow. Artikel Sun dkk. menggambarkan PySCF sebagai platform struktur elektronik umum yang menekankan kesederhanaan kode, fleksibilitas workflow komputasi, serta dukungan untuk sistem terbatas, sistem periodik, Hamiltonian khusus, metode mean-field, dan post-mean-field. (arXiv)

Dalam peta software buku ini, PySCF ditempatkan sebagai jembatan antara praktikum kimia kuantum dan pengembangan metode komputasi. Jika ORCA, GAMESS, dan Gaussian lebih sering digunakan sebagai paket siap pakai, PySCF mengajak mahasiswa melihat kimia kuantum sebagai sesuatu yang dapat diprogram, dimodifikasi, dan diperluas. Ini sangat relevan untuk mahasiswa tingkat lanjut, terutama yang tertarik pada kimia komputasi, machine learning kimia, material komputasi, atau pengembangan algoritma.

Namun, PySCF tidak harus menjadi software utama untuk seluruh mahasiswa pemula. Ia lebih tepat diperkenalkan bertahap. Pada praktikum dasar, mahasiswa dapat menggunakan Python biasa untuk menghitung partikel dalam kotak, menganalisis koordinat, membuat grafik HOMO-LUMO, atau memplot spektrum IR. Setelah itu, PySCF dapat diperkenalkan sebagai contoh bahwa Python juga dapat menjadi mesin perhitungan struktur elektronik sungguhan. Dengan cara ini, mahasiswa melihat kontinuitas antara programming sederhana dan komputasi kuantum profesional.

9. Jmol: Visualisasi Struktur Molekul Tiga Dimensi

Jmol adalah viewer molekul open-source untuk menampilkan struktur kimia tiga dimensi. Situs Jmol menjelaskan bahwa Jmol merupakan viewer struktur molekul 3D yang gratis dan open-source, berguna untuk siswa, pendidik, dan peneliti dalam kimia, biokimia, serta bidang lain yang berhubungan dengan struktur molekul. (Jmol) Jmol juga memiliki versi JavaScript, JSmol, yang banyak digunakan untuk visualisasi struktur molekul di peramban web.

Dalam praktikum kimia kuantum, Jmol sangat berguna untuk menampilkan struktur molekul, orbital, atau mode vibrasi dari file yang didukung. Keunggulannya adalah ringan, terbuka, dan cocok untuk pembelajaran. Mahasiswa dapat memutar molekul, mengubah mode tampilan, mengukur jarak dan sudut, serta memahami bentuk 3D. Dalam pembelajaran kimia, kemampuan memanipulasi struktur 3D sangat penting karena banyak konsep seperti simetri, stereokimia, polaritas, dan orbital tidak dapat dipahami secara baik hanya dari gambar 2D.

Jmol bukan mesin perhitungan kuantum. Ia tidak digunakan untuk menghitung energi ab initio atau DFT. Perannya adalah visualisasi. Namun, visualisasi adalah bagian penting dari interpretasi. Output kuantum yang berupa koordinat, orbital, dan mode vibrasi perlu dilihat agar mahasiswa dapat menghubungkan angka dengan bentuk molekul. Oleh karena itu, Jmol ditempatkan sebagai alat pendukung pembelajaran konseptual.

10. Molden: Visualisasi Output dan Orbital Molekul

Molden adalah program visualisasi yang banyak digunakan dalam kimia komputasi untuk membaca output berbagai paket, memvisualisasikan struktur, orbital molekul, densitas, dan mode vibrasi. Dalam praktikum, Molden berguna untuk membantu mahasiswa membaca hasil perhitungan yang dihasilkan oleh paket struktur elektronik. Meskipun antarmukanya tidak selalu semodern beberapa software baru, Molden tetap dikenal luas dalam komunitas kimia komputasi karena kemampuannya menangani berbagai format output.

Peran Molden dalam buku ini terutama pada tahap postprocessing. Setelah mahasiswa menjalankan perhitungan dengan ORCA, GAMESS, Gaussian, atau paket lain, mereka dapat

menggunakan Molden untuk melihat struktur hasil optimasi, orbital, dan mode vibrasi. Molden membantu mahasiswa memahami bahwa output komputasi tidak berhenti pada teks panjang. Data dalam output dapat diterjemahkan menjadi objek visual yang mendukung interpretasi kimia.

Namun, mahasiswa harus memahami keterbatasan visualisasi. Gambar orbital dari Molden atau software lain bergantung pada isovalue, basis set, dan data orbital yang dibaca. Jika isovalue diubah, tampilan orbital dapat tampak lebih besar atau lebih kecil. Karena itu, gambar orbital harus dilaporkan dengan konteks, bukan diperlakukan sebagai bukti visual mutlak. Molden adalah alat membaca data, bukan pengganti analisis kimia.

11. Multiwfn: Analisis Fungsi Gelombang dan Rapat Elektron

Multiwfn adalah software analisis fungsi gelombang yang banyak digunakan untuk menganalisis rapat elektron, orbital, muatan, potensial elektrostatik, indeks reaktivitas, topologi, dan berbagai properti molekul dari file hasil perhitungan kuantum. Dalam praktikum dasar, Multiwfn dapat diperkenalkan sebagai alat analisis lanjutan, terutama setelah mahasiswa memahami output dasar seperti energi, geometri, frekuensi, dan HOMO-LUMO.

Multiwfn penting karena menunjukkan bahwa data kuantum tidak hanya berupa energi total. Dari fungsi gelombang atau kerapatan elektron, berbagai informasi dapat diekstrak: distribusi elektron, daerah elektrofilik-nukleofilik, potensial elektrostatik molekul, muatan atom, dan indikator reaktivitas. Hal ini sejalan dengan sumber kimia kuantum yang menegaskan bahwa rapat elektron menghubungkan fungsi gelombang banyak elektron dengan sifat kimia yang lebih mudah divisualisasikan, termasuk momen dipol, potensial elektrostatik, dan analisis ikatan.

Namun, Multiwfn sebaiknya tidak diberikan terlalu awal tanpa fondasi. Analisis fungsi gelombang dapat menghasilkan banyak angka dan peta yang tampak meyakinkan, tetapi mahasiswa perlu memahami maknanya. Misalnya, muatan atom bergantung pada skema analisis. Potensial elektrostatik perlu ditafsirkan bersama struktur dan konteks reaksi. Indeks reaktivitas tidak boleh digunakan secara berlebihan. Oleh karena itu, dalam buku ini, Multiwfn ditempatkan sebagai software pendukung untuk proyek mini atau praktikum lanjutan, bukan sebagai alat pertama bagi pemula.

12. Python dan Jupyter Notebook sebagai Lingkungan Analisis

Walaupun Python akan dibahas secara khusus pada Bab 3, peta software Bab 2 tetap harus memasukkannya karena seluruh desain buku ini memadukan software kimia komputasi dengan programming ilmiah. Python digunakan untuk membaca data, menghitung parameter sederhana, membuat grafik, menyusun tabel, memplot fungsi gelombang, menghitung jarak dan sudut dari koordinat XYZ, membuat diagram energi, mensimulasikan spektrum IR, serta mengotomatisasi parsing output. Dalam ChemCompute, Jupyter Notebook juga relevan karena platform tersebut mencantumkan Jupyter Notebooks sebagai bagian dari sumber untuk komputasi yang reprodusibel. (ChemCompute)

Python memperkuat literasi komputasi mahasiswa karena membuat proses analisis menjadi eksplisit. Jika mahasiswa membuat grafik energi secara manual, prosesnya sulit diperiksa. Jika

mahasiswa membuat grafik dengan Python, dosen dapat melihat data input, rumus konversi, dan cara visualisasi. Ini sejalan dengan prinsip reprodusibilitas yang telah dibahas pada Bab 1. Dalam praktikum kimia kuantum, Python bukan pengganti ORCA, GAMESS, Gaussian, atau ChemCompute, melainkan alat untuk memahami dan menganalisis hasilnya.

Python juga memungkinkan inovasi. Mahasiswa dapat memodifikasi kode untuk molekul baru, metode baru, atau parameter baru. Mereka dapat memperluas praktikum dari sekadar mengikuti instruksi menjadi eksplorasi mandiri. Misalnya, setelah diberi kode partikel dalam kotak untuk n = 1 sampai 5, mahasiswa dapat mengubah panjang kotak, mengganti massa partikel, atau membandingkan model dengan sistem elektron π. Setelah diberi kode analisis HOMO-LUMO, mahasiswa dapat memasukkan data berbagai senyawa aromatik tersubstitusi. Inilah alasan buku ini menempatkan Python sebagai mitra wajib dalam setiap mata praktikum.

Sintesis Peta Software

Jika disusun sebagai alur kerja, peta software kimia komputasi dalam buku ini dapat digambarkan sebagai berikut:

```
Avogadro / GaussView / Gabedit
→ pembuatan struktur dan input awal

ChemCompute / ORCA / GAMESS / Gaussian / Psi4 / PySCF
→ perhitungan kimia kuantum

Jmol / Molden / Multiwfn / GaussView / Gabedit
→ visualisasi dan analisis output

Python / Jupyter Notebook
→ pengolahan data, grafik, validasi, dan laporan
```

Alur ini menunjukkan bahwa praktikum kimia kuantum tidak bergantung pada satu perangkat lunak tunggal. Yang lebih penting adalah memahami fungsi tiap alat. Jika satu kampus memiliki Gaussian dan GaussView, jalur tersebut dapat digunakan. Jika tidak, Avogadro, ChemCompute, GAMESS, ORCA, Psi4, PySCF, Jmol, dan Python dapat menjadi alternatif. Jika mahasiswa belum mampu memasang software lokal, ChemCompute dapat digunakan sebagai pintu masuk. Jika mahasiswa sudah kuat dalam Python, Psi4 dan PySCF dapat memperluas workflow. Jika tujuan praktikum hanya visualisasi struktur, Jmol atau Avogadro cukup. Jika tujuan analisis lanjutan rapat elektron, Multiwfn menjadi relevan.

Analisis kritis terhadap peta software ini menghasilkan beberapa prinsip. Pertama, pemilihan software harus berdasarkan tujuan praktikum, bukan popularitas. Kedua, software komersial dan open-source dapat saling melengkapi, tetapi lisensi harus dihormati. Ketiga, antarmuka grafis memudahkan pembelajaran, tetapi file input-output tetap harus dipahami. Keempat, platform web seperti ChemCompute memperluas akses, tetapi tetap memerlukan dokumentasi. Kelima, Python memperkuat reprodusibilitas dan inovasi. Keenam, visualisasi harus selalu dihubungkan dengan konsep kimia, bukan dianggap sebagai gambar dekoratif.

Dalam konteks pendidikan kimia, peta software ini membantu dosen merancang praktikum yang adaptif. Kampus dengan fasilitas terbatas tetap dapat menjalankan praktikum melalui ChemCompute dan Python. Kampus dengan laboratorium komputer dapat memasang Avogadro, ORCA, Jmol, dan Python. Kampus dengan lisensi komersial dapat menggunakan Gaussian dan GaussView sebagai jalur tambahan. Dengan demikian, buku ini tidak bergantung pada satu kondisi institusional, tetapi dapat disesuaikan dengan kemampuan sarana yang tersedia.

Dari sudut pandang mahasiswa, peta software ini membentuk kesadaran profesional. Mereka belajar bahwa kimia komputasi bukan sekadar mengenal satu program, tetapi memahami ekosistem kerja. Mereka belajar membangun struktur, menjalankan job, membaca output, memvisualisasikan data, menulis kode analisis, menyusun laporan, dan menyimpan file secara reprodusibel. Kompetensi semacam ini jauh lebih tahan lama dibanding keterampilan menghafal satu menu software tertentu, karena software dapat berubah, tetapi alur ilmiahnya tetap.

Subbab berikutnya akan membahas ChemCompute sebagai laboratorium komputasi berbasis web. Pembahasan tersebut akan menguraikan lebih rinci bagaimana ChemCompute dapat digunakan dalam praktikum kimia kuantum, apa keunggulannya bagi pembelajaran, apa batasannya, serta bagaimana mahasiswa harus tetap menjaga prinsip input, output, reprodusibilitas, dan analisis kritis ketika menggunakan platform daring.

2.2 ChemCompute sebagai Laboratorium Komputasi Berbasis Web

ChemCompute dapat didefinisikan sebagai platform komputasi kimia berbasis web yang dirancang untuk membantu mahasiswa dan dosen menjalankan praktikum kimia komputasi tanpa harus memasang, mengompilasi, mengonfigurasi, dan memelihara software serta perangkat keras komputasi secara lokal. Definisi ini penting karena ChemCompute bukan sekadar situs web biasa, melainkan sebuah **science gateway** atau gerbang komputasi ilmiah: pengguna berinteraksi melalui antarmuka web, sedangkan perhitungan kimia komputasi dijalankan pada sumber daya komputasi yang tersedia di balik platform tersebut. Halaman resmi ChemCompute menggambarkannya sebagai perangkat lunak kimia komputasi untuk pengajaran dan riset sarjana tanpa kerepotan kompilasi, instalasi, pemeliharaan software, dan pemeliharaan hardware. (ChemCompute)

Dalam konteks buku praktikum ini, ChemCompute dipahami sebagai **laboratorium komputasi berbasis web**. Istilah laboratorium di sini tidak merujuk pada ruang fisik dengan meja, kursi, komputer lokal, dan server kampus, tetapi pada lingkungan digital tempat mahasiswa dapat menyusun input, memilih paket komputasi, mengirim job, menunggu hasil, mengunduh output, lalu menganalisis data. Dengan demikian, ChemCompute memperluas makna laboratorium kimia. Laboratorium tidak lagi harus selalu berada dalam satu ruangan kampus; sebagian aktivitasnya dapat berpindah ke ruang daring yang dapat diakses melalui peramban web. Model seperti ini sangat relevan untuk pembelajaran kimia kuantum karena banyak perhitungan dasar dapat dilakukan secara digital, selama struktur, metode, basis set, charge, multiplicity, dan outputnya terdokumentasi dengan baik.

Secara pedagogis, ChemCompute membantu mengatasi salah satu hambatan utama dalam pembelajaran kimia komputasi, yaitu hambatan instalasi. Dalam banyak kasus, mahasiswa pemula tidak gagal memahami kimia kuantum karena tidak mampu berpikir ilmiah, melainkan karena tersendat pada persoalan teknis: instalasi program, konfigurasi sistem operasi, kompatibilitas pustaka, lisensi software, memori komputer, atau kesalahan menjalankan job melalui terminal. Artikel Perri dan Weber dalam *Journal of Chemical Education* menjelaskan bahwa antarmuka web dibuat untuk memfasilitasi penggunaan GAMESS dan software bebas lain bagi mahasiswa; antarmuka tersebut membantu mengelola bagian yang paling menyita waktu dan sering paling sulit dalam kimia komputasi, yaitu menjalankan kalkulasi itu sendiri. (ACS Publications)

Latar historis ChemCompute dapat ditempatkan dalam perkembangan lebih luas komputasi ilmiah berbasis web. Pada awal perkembangan kimia komputasi, pengguna harus memasang program di komputer lokal atau mengakses server melalui terminal. Model ini sangat efektif bagi peneliti terlatih, tetapi kurang ramah bagi kelas sarjana yang pesertanya banyak dan kemampuan teknisnya beragam. Setelah internet, komputasi awan, dan infrastruktur science gateway berkembang, muncul peluang untuk memindahkan sebagian beban teknis ke platform web. Perri dan Weber mendeskripsikan situs web yang memfasilitasi penggunaan GAMESS dengan tujuan memberi kesempatan kepada mahasiswa sarjana melakukan eksperimen kimia komputasi tanpa harus membeli software mahal. (ACS Publications) Dalam perspektif pendidikan, ini merupakan perubahan penting karena akses terhadap praktikum komputasi tidak lagi sepenuhnya bergantung pada laboratorium komputer khusus atau lisensi komersial.

ChemCompute juga berkembang seiring meningkatnya kebutuhan integrasi komputasi ke dalam kurikulum kimia sarjana. Selama bertahun-tahun, kimia komputasi sering diposisikan sebagai materi lanjutan, hanya untuk mahasiswa tingkat akhir atau kelompok riset tertentu. Padahal, banyak konsep kimia dasar dapat dipahami lebih baik melalui pemodelan: bentuk molekul, polaritas, orbital, energi ikatan, spektrum vibrasi, dan reaktivitas. Platform web seperti ChemCompute memungkinkan dosen memperkenalkan perhitungan kimia komputasi lebih awal karena mahasiswa tidak harus menguasai instalasi lokal terlebih dahulu. Artikel Buragohain tahun 2024 menggambarkan ChemCompute sebagai platform web gratis yang menyediakan akses ke berbagai paket electronic structure dan molecular dynamics open-source untuk mengeksplorasi struktur serta sifat sistem kimia, sekaligus menunjukkan bagaimana platform ini dapat memperkenalkan laboratorium kimia komputasi berkualitas tinggi pada tingkat sarjana tanpa hambatan finansial besar. (Springer)

Dari sudut pandang teori pembelajaran, ChemCompute mendukung pembelajaran berbasis inkuiri. Mahasiswa dapat mengajukan pertanyaan sederhana, misalnya: mengapa air polar sedangkan karbon dioksida nonpolar; bagaimana panjang ikatan H–H berubah terhadap energi; bagaimana perbedaan energi HOMO-LUMO antara etilena dan butadiena; atau bagaimana spektrum IR formaldehida memperlihatkan regangan C=O. Pertanyaan tersebut kemudian diterjemahkan menjadi perhitungan komputasi. Mahasiswa membangun struktur, memilih metode, menjalankan job, mengambil output, dan menafsirkan hasil. Proses ini menempatkan mahasiswa sebagai pelaku eksplorasi, bukan sekadar penerima informasi dari buku teks.

ChemCompute juga sesuai dengan prinsip **learning by doing** dalam kimia komputasi. Banyak konsep kimia kuantum sulit dipahami jika hanya dijelaskan secara verbal. Fungsi gelombang,

orbital, frekuensi vibrasi, energi total, dan permukaan energi potensial akan lebih mudah dipahami ketika mahasiswa melihat bagaimana perubahan struktur atau metode memengaruhi hasil. Dengan ChemCompute, dosen dapat merancang latihan terstruktur: mahasiswa menjalankan optimasi geometri H_2O, NH_3, CH_4, CO_2, dan BF_3; kemudian membandingkan sudut ikatan, momen dipol, dan simetri. Dari pengalaman tersebut, mahasiswa tidak hanya menghafal bentuk molekul, tetapi memahami bahwa geometri dan distribusi elektron dapat dihitung dan dianalisis.

Sebagai laboratorium berbasis web, ChemCompute memiliki beberapa keunggulan utama. Pertama, ia meningkatkan aksesibilitas. Mahasiswa dapat bekerja tanpa harus memiliki komputer dengan spesifikasi tinggi. Kedua, ia mengurangi beban teknis instalasi. Ketiga, ia dapat digunakan dalam pembelajaran tatap muka, daring, maupun hibrida. Keempat, ia membantu dosen memasukkan kimia komputasi ke dalam kelas tanpa harus membangun server sendiri. Kelima, ia dapat digunakan untuk latihan dasar yang terstandar. ChemCompute sendiri menyatakan bahwa GAMESS dan Jupyter Notebooks memiliki eksperimen siap pakai untuk digunakan di kelas. (ChemCompute)

Keunggulan lain adalah fleksibilitas tempat dan waktu. Dalam model laboratorium konvensional, mahasiswa harus berada di ruang tertentu pada waktu tertentu. Dalam model berbasis web, sebagian pekerjaan dapat dilakukan sebelum kelas. Mahasiswa dapat mengirim job lebih awal, lalu menggunakan waktu pertemuan untuk menganalisis output. Artikel tentang Chem Compute Science Gateway menjelaskan bahwa platform ini memungkinkan mahasiswa sarjana menyiapkan job, mengirimnya ke sumber daya komputasi, dan memvisualisasikan output; akses dari luar kampus juga memungkinkan mahasiswa memulai job panjang sebelum kelas sehingga waktu kelas dapat difokuskan pada analisis data. (ACS Publications) Pola ini sangat efektif untuk praktikum kimia kuantum karena waktu komputasi sering tidak dapat diprediksi secara tepat.

Namun, ChemCompute harus digunakan secara kritis. Kemudahan antarmuka web dapat menciptakan kesan bahwa kimia komputasi hanyalah proses mengisi formulir dan menekan tombol submit. Ini harus dihindari. ChemCompute adalah alat, bukan pengganti pemahaman. Mahasiswa tetap harus memahami jenis job, metode, basis set, charge, multiplicity, struktur awal, satuan energi, status konvergensi, dan isi output. Jika mahasiswa hanya mengunggah struktur, memilih opsi, lalu menyalin angka tanpa membaca output, maka praktikum kehilangan makna ilmiahnya. Dengan demikian, keberhasilan penggunaan ChemCompute tidak diukur dari berapa cepat job selesai, tetapi dari sejauh mana mahasiswa mampu menjelaskan hubungan antara input, proses komputasi, output, dan interpretasi kimia.

Kritik ini penting karena dalam kimia komputasi terdapat risiko **black-box computation**. Black-box computation terjadi ketika pengguna memperoleh hasil dari software tanpa memahami model, asumsi, dan batasannya. ChemCompute dapat menjadi sarana pembelajaran yang kuat jika digunakan sebagai *transparent box*, yaitu mahasiswa tetap membaca file input dan output. Sebaliknya, ia dapat menjadi black box jika mahasiswa hanya berinteraksi dengan tombol dan hasil akhir. Buku praktikum ini harus menekankan bahwa setiap penggunaan ChemCompute wajib disertai catatan: molekul apa yang dihitung, metode apa yang digunakan, basis set apa yang dipilih, charge dan multiplicity berapa, jenis job apa yang dijalankan, apakah job konvergen, output mana yang menjadi sumber data, dan bagaimana data dianalisis dengan Python.

Dalam konteks reprodusibilitas, ChemCompute memerlukan disiplin pengarsipan. Karena perhitungan dijalankan di platform web, mahasiswa harus mengunduh dan menyimpan file output. Jika tersedia file input, file tersebut juga harus disimpan. Nama file harus informatif, misalnya `h2o_gamess_b3lyp_631gd_opt.out`, bukan `hasil1.txt`. Mahasiswa juga perlu mencatat tanggal perhitungan, paket yang digunakan, dan opsi perhitungan. Jika hasil hanya dilihat di peramban web tetapi tidak disimpan, maka laporan menjadi lemah karena data tidak dapat diperiksa ulang. Dengan kata lain, ChemCompute memudahkan akses, tetapi tidak boleh mengurangi standar dokumentasi.

ChemCompute juga relevan untuk membangun budaya **computational notebook**. Jika mahasiswa menggunakan Jupyter Notebook yang tersedia melalui ekosistem ChemCompute atau platform lain, mereka dapat menggabungkan teks, rumus, kode, tabel, grafik, dan interpretasi dalam satu dokumen. Model notebook sangat sesuai untuk praktikum karena mahasiswa dapat menulis langkah kerja, menjalankan kode Python, menampilkan grafik, serta menyimpan hasil analisis. Dalam konteks pendidikan komputasi, notebook membantu membuat proses analisis lebih transparan dan reprodusibel. ChemCompute mencantumkan Jupyter Notebook sebagai salah satu komponen yang dapat digunakan dan disitasi bersama platformnya. (ChemCompute)

Walaupun demikian, penggunaan notebook juga memerlukan etika. Mahasiswa tidak boleh hanya menyalin notebook dari sumber lain tanpa memahami isinya. Setiap sel kode harus dapat dijelaskan. Jika notebook menggunakan data output ChemCompute, sumber output harus disebut. Jika grafik dibuat dari data tertentu, data tersebut harus tersedia. Jika kode diubah dari template dosen atau dokumentasi, sumber perlu dicatat sesuai aturan praktikum. ChemCompute dan Jupyter dapat memperkuat reprodusibilitas, tetapi hanya jika digunakan dengan disiplin ilmiah.

Dalam praktikum kimia kuantum, ChemCompute dapat digunakan untuk beberapa jenis perhitungan dasar. Pertama, **optimasi geometri** molekul kecil seperti H_2O, NH_3, CH_4, CO_2, H_2, N_2, etilena, formaldehida, atau etanol. Kedua, **single point energy** untuk membandingkan pengaruh metode atau basis set pada struktur tertentu. Ketiga, **frekuensi vibrasi** untuk memastikan struktur minimum dan memperoleh data spektrum IR. Keempat, **analisis orbital** jika paket dan output yang digunakan mendukung pengambilan energi orbital. Kelima, **energi relatif** untuk membandingkan konformer atau isomer sederhana. Dalam semua kasus ini, ChemCompute menyediakan jalur eksekusi, sedangkan analisis konseptual tetap dilakukan oleh mahasiswa.

Contoh praktikum sederhana adalah optimasi geometri air. Mahasiswa dapat menggambar H_2O, memilih paket komputasi yang tersedia, menetapkan charge 0 dan multiplicity 1, memilih metode dan basis set yang sesuai untuk tingkat praktikum, lalu menjalankan optimasi. Setelah output diperoleh, mahasiswa mengekstrak energi akhir, panjang ikatan O–H, sudut H–O–H, dan momen dipol jika tersedia. Selanjutnya, mahasiswa membandingkan hasil dengan teori VSEPR dan data literatur. Dengan Python, mahasiswa dapat membaca koordinat akhir dan menghitung ulang panjang serta sudut ikatan. Dalam latihan ini, ChemCompute berperan sebagai mesin perhitungan, sedangkan Python berperan sebagai alat validasi dan analisis.

Contoh kedua adalah perbandingan CO_2 dan H_2O. Kedua molekul memiliki ikatan polar, tetapi CO_2 nonpolar sedangkan H_2O polar. Mahasiswa dapat menjalankan optimasi geometri untuk keduanya, mengambil data geometri, dan menghitung atau membaca momen dipol. Dari output,

mahasiswa dapat melihat bahwa CO_2 linear sehingga vektor dipol C=O saling meniadakan, sedangkan H_2O bengkok sehingga memiliki momen dipol bersih. Praktikum seperti ini menghubungkan kimia umum, simetri molekul, struktur elektronik, dan komputasi kuantum dalam satu alur pembelajaran.

Contoh ketiga adalah eksplorasi sistem konjugasi. Mahasiswa dapat membandingkan etilena, butadiena, dan benzena. Setelah struktur dioptimasi melalui ChemCompute atau paket yang tersedia, mahasiswa mengambil energi HOMO dan LUMO bila output mendukung, lalu menghitung gap HOMO-LUMO dengan Python. Hasilnya dapat digunakan untuk membahas pengaruh delokalisasi elektron π terhadap gap energi. Dengan demikian, ChemCompute dapat menjadi gerbang menuju pembelajaran struktur elektronik molekul organik secara visual dan kuantitatif.

Contoh keempat adalah frekuensi vibrasi formaldehida. Mahasiswa menjalankan optimasi dan frekuensi, lalu mengambil daftar frekuensi dan intensitas. Dengan Python, data tersebut diplot menjadi spektrum IR simulasi. Puncak C=O dapat dianalisis dan dibandingkan dengan rentang frekuensi literatur. Pada tahap ini, mahasiswa belajar bahwa output frekuensi bukan hanya daftar angka, tetapi dapat diterjemahkan menjadi spektrum. Mereka juga belajar bahwa frekuensi komputasi harmonik dapat berbeda dari eksperimen sehingga kadang memerlukan faktor skala. ChemCompute menjadi sumber data, sedangkan Python menjadi sarana membangun representasi spektral.

Dari sudut pandang kurikulum, ChemCompute memungkinkan dosen menyusun praktikum bertingkat. Pada tingkat awal, mahasiswa menggunakan eksperimen siap pakai atau template sederhana. Pada tingkat menengah, mahasiswa memodifikasi molekul, metode, atau basis set. Pada tingkat lanjut, mahasiswa merancang proyek mini, misalnya membandingkan efek substituen pada benzena, menghitung energi konformer etanol, atau mempelajari kestabilan dimer air. Pendekatan bertingkat ini penting karena mahasiswa tidak boleh langsung diberikan tugas terlalu kompleks. Mereka perlu memahami struktur input-output secara bertahap.

ChemCompute juga mendukung pembelajaran inklusif. Tidak semua mahasiswa memiliki laptop kuat. Tidak semua kampus memiliki lisensi software komersial. Tidak semua dosen memiliki waktu untuk membantu instalasi program di banyak komputer mahasiswa. Platform web dapat mengurangi kesenjangan akses tersebut. Lehtola dan Marques dalam kajian tentang software bebas dan open-source untuk pendidikan kimia komputasi mencatat bahwa Chem Compute menyediakan sumber daya komputasi bagi pengajaran dan riset sarjana tanpa biaya bagi pengajar, sehingga institusi yang tidak memiliki personel atau sumber daya finansial untuk menyiapkan server fisik atau cloud sendiri tetap dapat menawarkan pendidikan kimia komputasi; mereka juga mengingatkan bahwa platform semacam ini bergantung pada sumber daya komputasi pihak ketiga sehingga keberlanjutan ketersediaannya perlu dipertimbangkan. (Wires Online Library)

Catatan kritis tersebut penting. ChemCompute memang membantu memperluas akses, tetapi buku praktikum tidak boleh sepenuhnya bergantung pada satu platform. Jika akses internet terganggu, server penuh, layanan berubah, atau paket tertentu tidak tersedia, praktikum harus tetap memiliki jalur alternatif. Karena itu, buku ini tetap memasukkan Avogadro, ORCA, GAMESS lokal, Gaussian bagi kampus yang memiliki lisensi, Psi4, PySCF, Jmol, Molden, Multiwfn, dan Python.

ChemCompute menjadi jalur utama yang mudah diakses, tetapi bukan satu-satunya jalan. Prinsipnya: konsep praktikum harus tetap berjalan meskipun alat spesifik dapat berubah.

Keterbatasan lain adalah kontrol teknis. Pada instalasi lokal, pengguna dapat mengatur banyak parameter: versi software, jumlah core, memori, file tambahan, modul eksternal, atau skrip batch. Pada platform web, sebagian opsi mungkin dibatasi demi stabilitas dan keamanan. Untuk praktikum dasar, pembatasan ini tidak menjadi masalah besar. Bahkan, pembatasan dapat membantu mahasiswa fokus pada konsep inti. Namun, untuk riset lanjut, pengguna mungkin memerlukan kontrol yang lebih detail. Oleh karena itu, ChemCompute paling tepat digunakan untuk praktikum sarjana, pengenalan metode, latihan konsep, dan proyek mini yang tidak terlalu berat.

Dalam pembelajaran, dosen perlu menyeimbangkan antara kemudahan dan kedalaman. Jika ChemCompute terlalu memudahkan proses submit job, dosen harus memperkuat tugas analisis. Mahasiswa dapat diminta menjelaskan setiap bagian input, menandai bagian output yang menunjukkan energi, mengidentifikasi status konvergensi, menghitung ulang parameter geometri dengan Python, dan membandingkan hasil dengan teori. Dengan cara ini, waktu yang sebelumnya habis untuk instalasi dapat dialihkan ke pemahaman ilmiah. Inilah nilai utama ChemCompute: bukan membuat praktikum menjadi instan, tetapi memindahkan beban dari persoalan teknis menuju analisis kimia.

Dari sudut pandang etika akademik, penggunaan ChemCompute tetap harus mengikuti prinsip sitasi, kejujuran, dan dokumentasi. Jika mahasiswa menggunakan ChemCompute, platform harus disebut dalam metode. Jika menggunakan paket tertentu di dalam ChemCompute, paket tersebut juga harus disebut. Jika platform menyediakan panduan sitasi, mahasiswa sebaiknya mengikuti panduan tersebut. Halaman sitasi ChemCompute mencantumkan rujukan Chem Compute Science Gateway dan artikel Perri & Weber 2014 untuk penggunaan antarmuka job submission GAMESS. (ChemCompute) Dengan mencantumkan sitasi, mahasiswa belajar menghargai infrastruktur ilmiah yang memungkinkan praktikum dilakukan.

ChemCompute juga dapat memperkuat budaya kerja kolaboratif. Dalam kelas, dosen dapat membagi kelompok mahasiswa berdasarkan molekul atau jenis perhitungan. Setiap kelompok menjalankan job melalui ChemCompute, lalu mengunggah output ke folder bersama. Setelah itu, seluruh kelas menggunakan Python untuk menganalisis data gabungan. Misalnya, satu kelas dapat membangun dataset kecil berisi momen dipol dan gap HOMO-LUMO beberapa molekul organik sederhana. Aktivitas seperti ini mengajarkan mahasiswa tentang data bersama, standardisasi format, dan pentingnya metadata. Namun, kerja kolaboratif harus tetap diatur agar setiap kelompok bertanggung jawab atas datanya.

Dalam kaitannya dengan Python, ChemCompute perlu dilihat sebagai sumber data komputasi aktual, sedangkan Python sebagai alat analisis. ChemCompute menjalankan perhitungan struktur elektronik atau dinamika sesuai paket yang tersedia. Python membaca output, mengekstrak energi, membuat grafik, menghitung selisih energi, memplot spektrum, atau membuat tabel laporan. Pembagian peran ini penting karena mahasiswa sering bertanya: jika sudah ada ChemCompute, mengapa masih perlu Python? Jawabannya: ChemCompute menghasilkan data, tetapi Python membantu memahami, memeriksa, mengolah, dan menyajikan data tersebut secara ilmiah. Tanpa

Python, mahasiswa cenderung bekerja manual dan rentan salah salin. Tanpa ChemCompute atau software kuantum lain, Python dasar tidak cukup untuk menghitung struktur elektronik molekul nyata.

Dalam konteks praktikum tujuh mata praktikum buku ini, ChemCompute dapat ditempatkan secara berbeda. Pada praktikum partikel dalam kotak, ChemCompute dapat digunakan melalui Jupyter Notebook bila tersedia, tetapi Python mandiri juga sudah cukup. Pada praktikum optimasi geometri, ChemCompute sangat berguna sebagai platform eksekusi. Pada praktikum Hartree–Fock dan basis set, ChemCompute dapat digunakan untuk menjalankan variasi metode/basis. Pada praktikum DFT dan HOMO-LUMO, ChemCompute dapat menjadi sumber output energi orbital bila paket yang dipakai mendukung. Pada praktikum simetri dan momen dipol, ChemCompute menghasilkan geometri dan dipol, sedangkan analisis simetri dilakukan manual atau dengan software visualisasi. Pada praktikum IR, ChemCompute menghasilkan frekuensi, sedangkan Python membuat spektrum. Pada praktikum energi reaksi, ChemCompute menghasilkan energi setiap spesies, sedangkan Python menghitung ΔE, ΔH, atau ΔG.

Dengan demikian, ChemCompute bukan hanya tambahan teknis dalam buku ini, tetapi bagian strategis dari desain praktikum. Ia menjawab masalah akses, mengurangi beban instalasi, membuka peluang pembelajaran daring, dan memungkinkan mahasiswa lebih cepat masuk ke analisis kimia. Namun, manfaat tersebut hanya optimal jika penggunaannya dipandu oleh prinsip yang benar: input harus dipahami, metode harus dicatat, output harus disimpan, konvergensi harus diperiksa, software harus disitasi, dan hasil harus dianalisis secara kritis.

Implikasi praktis bagi dosen adalah perlunya menyusun panduan ChemCompute yang tidak hanya berisi “klik di sini, pilih ini, submit”. Panduan harus menjelaskan mengapa memilih metode tertentu, bagaimana menentukan charge dan multiplicity, bagaimana membaca output, bagaimana menyimpan file, dan bagaimana mengolah hasil. Dosen juga perlu menyediakan alternatif jika ChemCompute tidak dapat diakses. Misalnya, praktikum dapat dirancang dalam tiga jalur: jalur ChemCompute, jalur ORCA/GAMESS lokal, dan jalur data output contoh untuk analisis Python. Dengan desain seperti ini, pembelajaran tetap berjalan dalam berbagai kondisi.

Implikasi bagi mahasiswa adalah perubahan sikap. Mahasiswa harus melihat ChemCompute sebagai fasilitas ilmiah, bukan sekadar situs praktikum. Mereka harus disiplin menyimpan file, membaca output, mencatat parameter, dan mengaitkan hasil dengan teori. Mereka juga harus belajar bahwa kemudahan akses membawa tanggung jawab. Perhitungan yang mudah dijalankan tidak otomatis mudah ditafsirkan. Output yang cepat diperoleh tidak otomatis benar. Grafik yang indah tidak otomatis sahih. Setiap hasil tetap harus melewati validasi ilmiah.

Secara konseptual, ChemCompute memperlihatkan bahwa laboratorium kimia masa depan semakin terhubung dengan komputasi awan, web interface, notebook interaktif, dan analisis data. Namun, tujuan akhirnya tetap sama dengan laboratorium ilmiah pada umumnya: menghasilkan pemahaman yang jujur, teliti, dan dapat dipertanggungjawabkan tentang sifat materi. Dalam buku ini, ChemCompute digunakan untuk mendekatkan mahasiswa pada pengalaman tersebut. Ia membuka pintu menuju praktikum kimia kuantum yang lebih inklusif, reprodusibel, dan inovatif, tetapi tetap harus ditopang oleh teori kimia kuantum, etika akademik, dan kemampuan analisis Python.

Subbab berikutnya akan membahas alur kerja ChemCompute secara operasional, mulai dari registrasi dan login, pemilihan paket komputasi, penyusunan input, pengiriman job, pemantauan status job, pengunduhan output, hingga analisis hasil. Pembahasan tersebut akan menerjemahkan posisi ChemCompute sebagai laboratorium berbasis web menjadi langkah praktikum yang konkret dan dapat diikuti mahasiswa.

2.3 Alur Kerja ChemCompute

Alur kerja ChemCompute adalah rangkaian langkah sistematis yang dilakukan mahasiswa untuk menjalankan praktikum kimia komputasi melalui platform berbasis web, mulai dari akses akun, pemilihan paket komputasi, penyusunan input, pengiriman job, pemantauan proses, pengunduhan output, sampai analisis hasil. Dalam konteks praktikum kimia kuantum, alur kerja ini harus dipahami sebagai **prosedur ilmiah digital**, bukan sekadar prosedur teknis menggunakan situs web. Setiap langkah memiliki makna akademik: login berkaitan dengan identitas pengguna dan pengelolaan job, pemilihan paket berkaitan dengan metode komputasi, penyusunan input berkaitan dengan representasi sistem molekul, pengiriman job berkaitan dengan eksekusi perhitungan, pemantauan status berkaitan dengan validasi proses, pengunduhan output berkaitan dengan reprodusibilitas, dan analisis hasil berkaitan dengan interpretasi kimia.

ChemCompute secara resmi digambarkan sebagai platform kimia komputasi untuk pengajaran dan riset sarjana tanpa kerepotan kompilasi, instalasi, serta pemeliharaan software dan hardware lokal. (ChemCompute) Halaman paket ChemCompute juga menjelaskan bahwa pengguna memilih paket kimia, dan paket seperti GAMESS serta Jupyter Notebooks menyediakan eksperimen yang siap digunakan untuk kelas. (ChemCompute) Dengan demikian, alur kerja ChemCompute perlu dibaca dalam dua lapisan. Lapisan pertama adalah lapisan teknis, yaitu bagaimana mahasiswa mengakses dan menjalankan job. Lapisan kedua adalah lapisan ilmiah, yaitu bagaimana mahasiswa memastikan bahwa job yang dijalankan benar secara kimia, terdokumentasi, dan dapat dipertanggungjawabkan.

Dalam praktikum kimia kuantum, alur kerja ChemCompute sebaiknya selalu mengikuti urutan berikut:

1. Registrasi dan login.
2. Pemilihan paket komputasi.
3. Penyusunan input.
4. Pengiriman job.
5. Pemantauan status job.
6. Pengunduhan output.
7. Analisis hasil.

Urutan tersebut tampak sederhana, tetapi setiap tahap memiliki potensi kesalahan yang dapat memengaruhi hasil. Jika mahasiswa salah memilih paket, job tidak sesuai tujuan. Jika input salah, output menjadi tidak valid. Jika status job tidak diperiksa, hasil gagal dapat dianggap berhasil. Jika output tidak diunduh, hasil tidak dapat direproduksi. Jika analisis dilakukan tanpa memahami metode, laporan menjadi dangkal. Oleh karena itu, alur kerja ChemCompute harus dilatih secara disiplin sejak praktikum pertama.

1. Registrasi dan Login

Tahap pertama adalah registrasi dan login. Dalam platform berbasis web, akun pengguna berfungsi untuk mengelola identitas, menyimpan riwayat job, mengakses paket tertentu, serta mengunduh hasil perhitungan. Registrasi juga memungkinkan mahasiswa bekerja secara lebih tertib karena setiap job dapat ditelusuri ke pengguna tertentu. Dalam pembelajaran, hal ini penting untuk akuntabilitas. Dosen dapat meminta mahasiswa menyimpan nama job, tanggal perhitungan, jenis job, dan file output sesuai akun masing-masing.

Secara pedagogis, login bukan sekadar formalitas. Ia menandai bahwa mahasiswa memasuki ruang kerja ilmiah digital. Sama seperti mahasiswa harus mencatat nama, tanggal, dan judul praktikum pada buku catatan laboratorium basah, mahasiswa juga harus mencatat identitas job dalam laboratorium komputasi. Setiap job yang dikirim ke ChemCompute harus memiliki nama yang jelas, misalnya:

```
h2o_gamess_hf_sto3g_opt
nh3_b3lyp_631gd_freq
co2_hf_631g_singlepoint
```

Nama seperti ini lebih baik daripada `coba1`, `praktikum`, atau `hasil_baru`, karena memuat informasi molekul, paket/metode, basis set, dan jenis pekerjaan. Kebiasaan memberi nama job secara sistematis akan sangat membantu ketika mahasiswa menjalankan banyak perhitungan dalam satu mata praktikum.

Dalam tahap login, mahasiswa juga perlu memastikan koneksi internet stabil, menggunakan peramban yang kompatibel, dan menyiapkan folder kerja lokal. Walaupun perhitungan dijalankan di server jarak jauh, hasil tetap harus disimpan di komputer mahasiswa atau penyimpanan awan yang terorganisasi. Sebelum memulai job, mahasiswa sebaiknya membuat folder praktikum, misalnya:

```
praktikum_02_chemcompute/
├── input/
├── output/
├── data/
├── python/
├── gambar/
└── laporan/
```

Struktur ini membuat alur kerja lebih tertib. File input, output, data hasil, kode Python, gambar, dan laporan tidak bercampur. Jika sejak awal file disimpan rapi, proses analisis dan penulisan laporan menjadi lebih mudah.

2. Pemilihan Paket Komputasi

Tahap kedua adalah memilih paket komputasi. ChemCompute menyediakan akses ke paket kimia komputasi tertentu dan eksperimen yang dapat digunakan untuk kelas. Halaman paket ChemCompute menyebut bahwa pengguna memilih paket kimia dan bahwa GAMESS serta Jupyter Notebooks memiliki eksperimen siap pakai untuk pembelajaran. (ChemCompute) Dalam

konteks praktikum kimia kuantum, paket yang dipilih harus sesuai dengan tujuan perhitungan. Jika tujuan praktikum adalah optimasi geometri atau energi elektronik, paket struktur elektronik seperti GAMESS relevan. Jika tujuan praktikum adalah latihan programming, visualisasi data, atau analisis numerik, Jupyter Notebook dapat menjadi pilihan yang tepat.

Pemilihan paket tidak boleh dilakukan secara asal. Mahasiswa harus memahami perbedaan antara paket komputasi dan jenis perhitungan. Paket adalah perangkat lunak atau lingkungan yang digunakan, misalnya GAMESS atau Jupyter Notebook. Jenis perhitungan adalah pekerjaan yang dilakukan, misalnya optimasi geometri, single point energy, frekuensi vibrasi, atau analisis data. Metode adalah model teoretis yang digunakan, misalnya HF, DFT/B3LYP, atau semiempiris. Basis set adalah himpunan fungsi matematika untuk merepresentasikan orbital. Keempat unsur ini harus dibedakan agar laporan praktikum tidak rancu.

Sebagai contoh, jika mahasiswa menulis “menggunakan ChemCompute”, kalimat itu belum cukup untuk menjelaskan metode. ChemCompute adalah platform. Mahasiswa perlu menulis paket yang dijalankan, misalnya GAMESS, jenis job yang dilakukan, misalnya optimasi geometri, metode yang dipilih, misalnya Hartree–Fock, dan basis set, misalnya STO-3G atau 6-31G. Penulisan yang lebih lengkap adalah: “Optimasi geometri H_2O dilakukan melalui ChemCompute menggunakan paket GAMESS pada tingkat teori HF/STO-3G dengan charge 0 dan multiplicity 1.” Kalimat ini lebih reprodusibel karena orang lain dapat memahami konteks perhitungan.

Pemilihan paket juga berkaitan dengan tingkat kesulitan praktikum. Untuk mahasiswa pemula, dosen sebaiknya memilih paket dan template yang paling stabil. Latihan pertama dapat menggunakan molekul kecil seperti H_2O, NH_3, CH_4, atau CO_2. Setelah mahasiswa memahami alur, barulah diberikan variasi metode, basis set, atau molekul. Jika sejak awal mahasiswa diberi sistem besar atau metode rumit, fokus pembelajaran dapat bergeser dari pemahaman konsep ke penyelesaian error teknis. Oleh karena itu, pemilihan paket harus mempertimbangkan tujuan pedagogis, bukan hanya kecanggihan metode.

3. Penyusunan Input

Tahap ketiga adalah penyusunan input. Inilah tahap paling penting dalam alur kerja ChemCompute karena input menentukan sistem apa yang dihitung dan bagaimana perhitungan dilakukan. Input kimia komputasi minimal memuat informasi struktur molekul, charge, multiplicity, jenis perhitungan, metode, basis set, dan kadang parameter tambahan. Artikel Perri dan Weber menjelaskan bahwa antarmuka web ChemCompute dikembangkan untuk memfasilitasi penggunaan GAMESS oleh mahasiswa, sehingga mahasiswa dapat melakukan eksperimen kimia komputasi tanpa perlu membeli software mahal; dalam praktiknya, mahasiswa dapat memulai dari software gratis seperti Avogadro atau MacMolPlt untuk menggambar molekul dan menghasilkan input GAMESS. (ACS Publications)

Dalam praktikum, input dapat disusun melalui beberapa cara. Pertama, mahasiswa dapat menggunakan struktur dari editor molekul seperti Avogadro, kemudian mengekspor atau mengadaptasi input. Kedua, mahasiswa dapat menggunakan template input yang disediakan dosen. Ketiga, mahasiswa dapat menulis input secara manual. Keempat, mahasiswa dapat

menggunakan eksperimen siap pakai jika tersedia. Apa pun jalurnya, input harus dibaca dan dipahami. Mahasiswa tidak boleh mengirim input tanpa mengetahui arti bagian-bagiannya.

Input untuk perhitungan air, misalnya, harus menyatakan bahwa molekul terdiri atas satu atom O dan dua atom H, charge total 0, multiplicity 1, serta koordinat atom. Jika menggunakan GAMESS, input biasanya memuat blok kontrol dan data. Contoh konseptual sederhana untuk optimasi air dapat berbentuk:

```
 $CONTRL SCFTYP=RHF RUNTYP=OPTIMIZE $END
 $BASIS GBASIS=STO NGAUSS=3 $END
 $DATA
Water molecule
C1
O  8.0   0.000000   0.000000   0.000000
H  1.0   0.758602   0.000000   0.504284
H  1.0  -0.758602   0.000000   0.504284
 $END
```

Input seperti ini harus dijelaskan kepada mahasiswa. `RUNTYP=OPTIMIZE` menunjukkan bahwa job berupa optimasi geometri. `SCFTYP=RHF` menunjukkan tipe perhitungan Hartree–Fock tertutup untuk sistem singlet. `GBASIS=STO NGAUSS=3` berkaitan dengan basis set STO-3G. Blok `$DATA` memuat nama molekul, simetri, atom, nomor atom, dan koordinat. Jika mahasiswa memahami input, mereka akan lebih mudah memperbaiki error dan menafsirkan output.

Penyusunan input juga harus memperhatikan **charge** dan **multiplicity**. Molekul air netral menggunakan charge 0 dan multiplicity 1. Amonium NH_4^+ menggunakan charge +1 dan multiplicity 1. Radikal metil $CH_3\bullet$ menggunakan charge 0 dan multiplicity 2. Oksigen O_2 keadaan dasar menggunakan multiplicity 3. Kesalahan charge dan multiplicity dapat menghasilkan struktur elektronik yang salah meskipun job selesai. Karena itu, sebelum mengirim job, mahasiswa harus memeriksa apakah jumlah elektron dan keadaan spin sesuai dengan sistem kimia yang dimaksud.

Selain charge dan multiplicity, struktur awal harus diperiksa. Atom tidak boleh terlalu berdekatan secara tidak wajar. Hidrogen harus lengkap. Molekul harus sesuai bentuk kimia yang dimaksud. Jika struktur dibuat dengan Avogadro, mahasiswa sebaiknya melakukan pembersihan geometri awal menggunakan force field sebelum mengirim job kuantum. Namun, perlu ditegaskan bahwa pembersihan force field bukan pengganti optimasi kuantum. Ia hanya membantu menghasilkan struktur awal yang lebih wajar.

Input yang baik juga harus sederhana pada tahap awal. Mahasiswa pemula tidak perlu langsung memakai banyak opsi teknis. Misalnya, untuk latihan pertama cukup gunakan HF/STO-3G atau metode dasar lain yang stabil. Setelah mahasiswa memahami alur, dosen dapat memperkenalkan DFT, basis set lebih besar, frekuensi, atau model pelarut. Prinsip pedagogisnya adalah **mulai dari input minimal yang benar, kemudian tingkatkan kompleksitas secara bertahap**.

4. Pengiriman Job

Tahap keempat adalah pengiriman job. Pada tahap ini, input yang sudah disiapkan dikirim melalui ChemCompute untuk dijalankan pada server. Halaman GAMESS ChemCompute menyatakan

bahwa situs tersebut memungkinkan pengguna mengirim job GAMESS ke server mereka dengan harapan memudahkan proses kalkulasi kimia kuantum. (ChemCompute) Secara teknis, mahasiswa mungkin hanya melihat proses ini sebagai klik tombol submit. Secara ilmiah, pengiriman job adalah saat ketika hipotesis praktikum diterjemahkan menjadi perhitungan numerik.

Sebelum mengirim job, mahasiswa harus melakukan pemeriksaan akhir. Pemeriksaan ini sebaiknya menggunakan checklist:

1. Apakah nama job sudah jelas?
2. Apakah molekul yang dihitung benar?
3. Apakah koordinat atom lengkap?
4. Apakah charge benar?
5. Apakah multiplicity benar?
6. Apakah jenis job sesuai tujuan?
7. Apakah metode dan basis set sesuai instruksi?
8. Apakah ukuran molekul sesuai untuk sumber daya yang tersedia?
9. Apakah input sudah disimpan?
10. Apakah folder output sudah disiapkan?

Checklist ini penting karena banyak kesalahan praktikum terjadi sebelum job dikirim. Setelah job berjalan, mahasiswa baru menyadari bahwa molekul salah, hidrogen kurang, atau metode tidak sesuai. Dalam kelas besar, kesalahan semacam ini dapat membuang waktu server dan waktu belajar. Dengan checklist, mahasiswa belajar disiplin seperti di laboratorium basah: sebelum menyalakan alat, pastikan sampel dan metode benar.

Pengiriman job juga harus memperhatikan tanggung jawab penggunaan sumber daya. Mahasiswa tidak boleh mengirim banyak job besar tanpa kebutuhan. Untuk praktikum dasar, molekul kecil dan metode sederhana sudah memadai. Job yang terlalu besar dapat lama berjalan, gagal, atau mengganggu pengguna lain. Etika penggunaan ChemCompute menuntut mahasiswa memilih perhitungan yang proporsional dengan tujuan pembelajaran. Jika ingin melakukan eksplorasi lebih besar, mahasiswa perlu berkonsultasi dengan dosen.

Pada tahap ini, dosen dapat memberikan aturan: satu mahasiswa atau satu kelompok hanya mengirim job tertentu sesuai jadwal; job uji coba dilakukan dengan molekul kecil; job besar harus disetujui. Aturan ini bukan untuk membatasi kreativitas, tetapi untuk menjaga keberlanjutan sumber daya bersama. Laboratorium komputasi berbasis web adalah fasilitas kolektif, sehingga penggunaannya harus bertanggung jawab.

5. Pemantauan Status Job

Tahap kelima adalah pemantauan status job. Setelah job dikirim, mahasiswa perlu memeriksa apakah job sedang menunggu, sedang berjalan, selesai normal, atau gagal. Pemantauan status bukan sekadar menunggu hasil, tetapi bagian dari validasi proses. Dalam perhitungan kimia kuantum, job dapat gagal karena berbagai alasan: input salah format, charge tidak sesuai, SCF tidak konvergen, struktur awal buruk, memori tidak cukup, waktu komputasi habis, atau metode tidak cocok.

Mahasiswa perlu membedakan antara **job selesai** dan **job berhasil secara ilmiah**. Job dapat selesai karena program berhenti akibat error. Job juga dapat menghasilkan output tetapi optimasi tidak konvergen. Job dapat memberikan energi tetapi struktur bukan minimum. Karena itu, status web hanya langkah awal. Validasi sebenarnya dilakukan dengan membaca output. Jika output menunjukkan error, mahasiswa tidak boleh mengambil angka terakhir sebagai hasil final. Jika optimasi tidak konvergen, mahasiswa harus melaporkan kegagalan atau memperbaiki input.

Pemantauan status juga memberikan pelajaran tentang sifat iteratif kimia komputasi. Perhitungan tidak selalu berhasil pada percobaan pertama. Mahasiswa mungkin perlu memperbaiki struktur awal, mengganti metode, menggunakan basis set lebih kecil, memperbaiki charge, atau mengubah jenis job. Ini bukan kegagalan belajar. Justru proses memperbaiki job adalah bagian penting dari praktikum. Dalam riset komputasi nyata, iterasi semacam ini sangat umum. Mahasiswa perlu belajar menganalisis error, bukan panik atau langsung mengganti angka.

Dalam laporan, mahasiswa sebaiknya mencatat jika ada job gagal dan bagaimana perbaikannya. Misalnya:

```
Job pertama H2O dengan struktur awal yang terlalu rapat gagal mencapai
konvergensi optimasi. Struktur diperbaiki menggunakan Avogadro, kemudian job
kedua dengan metode yang sama berhasil konvergen.
```

Catatan seperti ini menunjukkan kejujuran dan pemahaman proses. Laporan yang hanya menampilkan hasil akhir tanpa mencatat kendala sering kehilangan nilai pembelajaran. Dosen sebaiknya memberi apresiasi pada analisis kegagalan yang jujur, selama mahasiswa mampu menjelaskan penyebab dan langkah perbaikan.

6. Pengunduhan Output

Tahap keenam adalah pengunduhan output. Output adalah dokumen utama yang berisi hasil perhitungan. Dalam praktikum kimia kuantum, output harus diperlakukan sebagai data primer. Jika mahasiswa tidak mengunduh output, hasil praktikum tidak dapat diverifikasi. Jika output hanya disalin sebagian tanpa file asli, dosen tidak dapat memeriksa apakah data berasal dari job yang benar. Karena itu, setiap job yang selesai harus diunduh dan disimpan pada folder `output/` dengan nama file sistematis.

Output harus disimpan bersama input. Idealnya, satu job memiliki pasangan file seperti:

```
input/h2o_gamess_hf_sto3g_opt.inp
output/h2o_gamess_hf_sto3g_opt.out
```

Jika struktur akhir diekstrak, simpan juga:

```
data/h2o_gamess_hf_sto3g_final.xyz
```

Jika energi, frekuensi, atau momen dipol dimasukkan ke tabel, simpan tabelnya:

```
data/h2o_gamess_hf_sto3g_summary.csv
```

Jika grafik dibuat dengan Python, simpan gambar:

```
gambar/h2o_energy_plot.png
```

Pengunduhan output juga perlu dilakukan segera setelah job selesai. Platform web dapat mengalami perubahan sesi, job lama dapat sulit ditemukan, atau mahasiswa dapat lupa nama job. Karena itu, setelah job selesai, langkah pertama adalah mengunduh file output. Langkah kedua adalah memeriksa output. Langkah ketiga adalah mencatat metadata. Langkah keempat adalah menganalisis data. Urutan ini mencegah hilangnya data.

Output ChemCompute atau paket yang dijalankan melalui ChemCompute biasanya memuat banyak informasi. Mahasiswa tidak perlu membaca semuanya dari awal sampai akhir pada tahap dasar, tetapi harus tahu bagian penting yang dicari. Untuk optimasi geometri, cari status konvergensi, energi akhir, dan koordinat akhir. Untuk frekuensi, cari daftar frekuensi dan periksa apakah ada frekuensi imajiner. Untuk energi orbital, cari bagian orbital atau eigenvalue jika tersedia. Untuk momen dipol, cari bagian dipole moment. Untuk energi reaksi, catat energi masing-masing spesies dengan satuan yang benar. Pada tahap lanjut, Python dapat digunakan untuk mengekstrak informasi ini secara otomatis.

Pengunduhan output juga berkaitan dengan integritas akademik. Mahasiswa harus menyerahkan output asli sebagai lampiran atau bukti. Jika output diedit, harus jelas bagian mana yang diringkas. Tidak diperbolehkan mengubah angka pada output. Tidak diperbolehkan mengambil output kelompok lain lalu mengganti nama. Tidak diperbolehkan membuat output palsu. Dalam praktikum komputasi, file output adalah analog dari lembar data mentah di laboratorium basah. Mengubah output sama seriusnya dengan memalsukan data eksperimen.

7. Analisis Hasil

Tahap ketujuh adalah analisis hasil. Inilah inti pembelajaran. ChemCompute menjalankan perhitungan, tetapi mahasiswa yang harus menafsirkan. Analisis hasil meliputi pembacaan output, validasi konvergensi, ekstraksi data, konversi satuan, pembuatan tabel, visualisasi grafik, perbandingan dengan teori atau literatur, dan penyusunan kesimpulan. Jika tahap analisis lemah, seluruh perhitungan menjadi kurang bermakna.

Analisis harus dimulai dari validasi. Pertanyaan pertama bukan “berapa energinya?”, tetapi “apakah perhitungannya valid?”. Untuk optimasi geometri, periksa apakah optimasi konvergen. Untuk struktur minimum, periksa apakah tidak ada frekuensi imajiner jika frekuensi dihitung. Untuk SCF, periksa apakah konvergen. Untuk perhitungan energi reaksi, pastikan semua spesies dihitung dengan metode dan basis set yang sama. Untuk perhitungan HOMO-LUMO, pastikan energi orbital berasal dari output yang benar. Untuk momen dipol, pastikan struktur yang dianalisis adalah struktur final, bukan struktur awal.

Setelah validasi, data diekstrak. Misalnya, pada praktikum optimasi H_2O, mahasiswa mengambil energi akhir, koordinat final, panjang ikatan O–H, sudut H–O–H, dan momen dipol. Koordinat final dapat dianalisis dengan Python untuk menghitung ulang jarak dan sudut. Pada praktikum frekuensi formaldehida, mahasiswa mengambil frekuensi dan intensitas IR, lalu membuat

spektrum simulasi dengan Python. Pada praktikum energi reaksi, mahasiswa mengambil energi reaktan dan produk, mengonversi Hartree ke kJ/mol, lalu menghitung ΔE.

Analisis hasil sebaiknya selalu menghubungkan angka dengan konsep kimia. Misalnya, jika sudut H–O–H sekitar 104–105°, mahasiswa harus menghubungkannya dengan bentuk bengkok dan pasangan elektron bebas. Jika CO_2 memiliki momen dipol mendekati nol, mahasiswa harus menghubungkannya dengan bentuk linear dan pembatalan vektor dipol. Jika butadiena memiliki gap HOMO-LUMO lebih kecil daripada etilena, mahasiswa harus menghubungkannya dengan delokalisasi elektron π. Jika frekuensi C=O formaldehida tinggi dan kuat, mahasiswa harus menghubungkannya dengan kekuatan ikatan dan perubahan momen dipol selama vibrasi.

Python memiliki peran penting pada tahap analisis. Data dari output dapat dimasukkan ke CSV, lalu diproses dengan pandas dan matplotlib. Misalnya, mahasiswa dapat membuat tabel:

```
molekul, metode, basis_set, energi_hartree, dipol_debye
H2O, HF, STO-3G, -74.96, 1.8
NH3, HF, STO-3G, -55.45, 1.5
CO2, HF, STO-3G, -187.6, 0.0
```

Kemudian Python digunakan untuk membuat grafik momen dipol atau tabel perbandingan. Dengan cara ini, hasil ChemCompute tidak berhenti sebagai output teks, tetapi diolah menjadi data ilmiah yang komunikatif. Python juga membantu mengurangi kesalahan salin dan memperkuat reprodusibilitas.

Analisis hasil harus diakhiri dengan kesimpulan yang dibatasi oleh metode. Mahasiswa tidak boleh menulis klaim terlalu luas. Kalimat yang baik adalah:

```
Pada tingkat teori HF/STO-3G melalui perhitungan GAMESS di ChemCompute, hasil optimasi menunjukkan bahwa H2O memiliki struktur bengkok dengan momen dipol tidak nol, sedangkan CO2 memiliki struktur linear dengan momen dipol mendekati nol. Hasil ini konsisten secara kualitatif dengan teori VSEPR dan konsep simetri molekul.
```

Kalimat ini baik karena menyebut tingkat teori, platform, hasil utama, dan batas klaim. Kalimat yang kurang baik adalah:

```
Perhitungan membuktikan bahwa air selalu polar dan karbon dioksida tidak pernah polar.
```

Kalimat kedua terlalu absolut dan tidak menyebut metode. Dalam kimia komputasi, kesimpulan harus selalu dikaitkan dengan model dan konteks.

Contoh Alur Kerja Lengkap: Optimasi Geometri H_2O

Untuk memperjelas alur kerja ChemCompute, berikut contoh praktikum sederhana optimasi geometri air.

Pertanyaan praktikum:
Bagaimana struktur geometri molekul H_2O setelah dioptimasi secara komputasi, dan bagaimana hasilnya dikaitkan dengan bentuk bengkok molekul air?

Langkah 1 — Login:
Mahasiswa masuk ke akun ChemCompute dan menyiapkan folder lokal `praktikum_h2o/`.

Langkah 2 — Pilih paket:
Mahasiswa memilih paket komputasi yang tersedia, misalnya GAMESS, karena ChemCompute mendukung pengiriman job GAMESS melalui server. (ChemCompute)

Langkah 3 — Susun input:
Mahasiswa menyiapkan struktur H_2O dengan charge 0 dan multiplicity 1. Jika menggunakan input template, mahasiswa memeriksa koordinat, metode, basis set, dan jenis job.

Langkah 4 — Kirim job:
Mahasiswa mengirim job optimasi geometri.

Langkah 5 — Pantau status:
Mahasiswa menunggu sampai job selesai, kemudian memeriksa apakah job berhenti normal.

Langkah 6 — Unduh output:
Mahasiswa mengunduh output dan menyimpannya sebagai:

```
output/h2o_gamess_hf_sto3g_opt.out
```

Langkah 7 — Analisis:
Mahasiswa mencari energi akhir, koordinat final, panjang ikatan O–H, sudut H–O–H, dan status konvergensi. Jika diperlukan, koordinat final dianalisis dengan Python untuk menghitung jarak dan sudut. Kesimpulan ditulis dengan menyebut metode dan batas interpretasi.

Contoh ini sederhana, tetapi mencerminkan seluruh prinsip alur kerja ChemCompute. Jika mahasiswa menguasai alur untuk H_2O, mereka dapat menerapkannya pada NH_3, CH_4, CO_2, formaldehida, etanol, dan molekul lain.

Contoh Alur Kerja Lengkap: Frekuensi IR Formaldehida

Contoh kedua adalah perhitungan frekuensi IR formaldehida.

Pertanyaan praktikum:
Bagaimana spektrum IR simulasi formaldehida, dan puncak mana yang berkaitan dengan regangan C=O?

Langkah 1 — Login dan folder:
Mahasiswa masuk ke ChemCompute dan membuat folder `praktikum_ir_formaldehida/`.

Langkah 2 — Pilih paket:
Mahasiswa memilih paket yang mendukung optimasi dan frekuensi.

Langkah 3 — Susun input:
Mahasiswa menyiapkan struktur formaldehida, charge 0, multiplicity 1, dan job `Opt + Freq` jika tersedia.

Langkah 4 — Submit:
Job dikirim untuk optimasi dan frekuensi.

Langkah 5 — Cek status:
Mahasiswa memastikan job selesai dan output tersedia.

Langkah 6 — Unduh output:
File output disimpan dalam folder `output/`.

Langkah 7 — Validasi:
Mahasiswa memeriksa apakah struktur minimum memiliki frekuensi nyata. Jika ada frekuensi imajiner, struktur belum valid sebagai minimum.

Langkah 8 — Analisis Python:
Mahasiswa mengambil daftar frekuensi dan intensitas, lalu membuat spektrum IR simulasi dengan Python.

Langkah 9 — Interpretasi:
Mahasiswa mengidentifikasi puncak C=O, membandingkannya dengan literatur, dan menjelaskan kemungkinan perbedaan akibat metode harmonik dan basis set.

Contoh ini menunjukkan bahwa ChemCompute bukan hanya untuk memperoleh output, tetapi untuk membangun alur praktikum lengkap dari input sampai interpretasi spektrum.

Checklist Alur Kerja ChemCompute

Agar setiap praktikum berjalan tertib, mahasiswa dapat menggunakan checklist berikut.

Sebelum submit job:

1. Nama job sudah jelas.
2. Molekul sudah benar.
3. Struktur awal sudah diperiksa.
4. Hidrogen lengkap.
5. Charge benar.
6. Multiplicity benar.
7. Paket komputasi sesuai.
8. Jenis job sesuai.
9. Metode dan basis set sesuai.

10. Input disimpan.

Setelah job selesai:

1. Output diunduh.
2. Status selesai normal diperiksa.
3. SCF konvergen.
4. Optimasi konvergen jika job optimasi.
5. Frekuensi imajiner diperiksa jika job frekuensi.
6. Energi akhir dicatat.
7. Koordinat final disimpan.
8. Data penting diekstrak.
9. Analisis Python dilakukan.
10. Hasil disimpan dalam tabel dan grafik.
11. Kesimpulan menyebut metode dan batas klaim.

Checklist ini perlu dilampirkan dalam buku praktikum karena membantu mahasiswa mengembangkan kebiasaan kerja ilmiah. Praktikum komputasi yang baik bukan yang paling cepat selesai, melainkan yang paling jelas alurnya, paling lengkap dokumentasinya, dan paling tepat interpretasinya.

Analisis Kritis terhadap Alur Kerja ChemCompute

Alur kerja ChemCompute memiliki keunggulan besar, tetapi juga perlu dikritisi. Keunggulannya adalah aksesibilitas, kemudahan, fleksibilitas, dan kesesuaian untuk pendidikan sarjana. Platform ini memungkinkan kampus dengan sumber daya terbatas tetap memperkenalkan kimia komputasi. Ia juga memungkinkan pembelajaran daring dan hibrida. Selain itu, ChemCompute dapat mengurangi hambatan instalasi sehingga dosen dapat lebih fokus pada konsep.

Namun, ada beberapa keterbatasan. Pertama, pengguna mungkin memiliki kontrol teknis lebih terbatas dibanding instalasi lokal. Kedua, ketersediaan paket atau eksperimen dapat berubah. Ketiga, akses bergantung pada internet dan server. Keempat, mahasiswa dapat terjebak pada penggunaan antarmuka tanpa memahami input. Kelima, job yang terlalu besar mungkin tidak sesuai untuk platform pembelajaran. Karena itu, buku ini menempatkan ChemCompute sebagai salah satu jalur utama, tetapi tetap menyediakan jalur Avogadro, ORCA, GAMESS lokal, Gaussian jika tersedia, Psi4, PySCF, Jmol, Molden, Multiwfn, dan Python.

Secara pedagogis, keterbatasan tersebut dapat diatasi dengan desain praktikum yang baik. Dosen harus memberi template input yang jelas, meminta mahasiswa menjelaskan setiap bagian input, mewajibkan output asli, dan menugaskan analisis Python. Jika ChemCompute tidak dapat diakses pada suatu waktu, dosen dapat menyediakan output contoh untuk dianalisis. Dengan demikian, pembelajaran tetap dapat berlangsung meskipun platform mengalami kendala.

Alur kerja ChemCompute membawa beberapa implikasi penting. Pertama, mahasiswa perlu belajar bekerja secara sistematis. Mereka tidak boleh langsung mengirim job tanpa memeriksa input. Kedua, mahasiswa perlu memahami bahwa hasil komputasi adalah data ilmiah yang harus disimpan. Ketiga, mahasiswa perlu mengembangkan literasi output. Keempat, mahasiswa perlu menggunakan Python untuk analisis agar hasil lebih reprodusibel. Kelima, mahasiswa harus menyusun kesimpulan berdasarkan data, bukan berdasarkan dugaan awal.

Dalam buku praktikum ini, setiap mata praktikum yang menggunakan ChemCompute sebaiknya memiliki struktur kerja sebagai berikut:

```
Tujuan praktikum
→ molekul model
→ persiapan struktur
→ input ChemCompute
→ pengiriman job
→ pengunduhan output
→ validasi output
→ analisis Python
→ interpretasi kimia
→ laporan
```

Struktur tersebut akan dipakai berulang pada praktikum optimasi geometri, Hartree–Fock dan basis set, DFT dan HOMO-LUMO, simetri dan momen dipol, frekuensi IR, serta energi reaksi. Dengan pengulangan alur, mahasiswa akan membangun kebiasaan kerja komputasi yang kuat.

Pada akhirnya, alur kerja ChemCompute harus dipahami sebagai proses ilmiah yang menyatukan kemudahan teknologi dengan ketelitian akademik. ChemCompute memudahkan mahasiswa menjalankan perhitungan, tetapi tidak menggantikan tugas mahasiswa untuk berpikir. Platform web menyediakan akses, tetapi mahasiswa tetap bertanggung jawab atas input, output, analisis, dan kesimpulan. Dengan alur kerja yang benar, ChemCompute dapat menjadi laboratorium komputasi berbasis web yang efektif, inklusif, dan kuat untuk praktikum kimia kuantum.

Subbab berikutnya akan membahas kapan menggunakan ChemCompute dan kapan menggunakan Python. Pembahasan tersebut penting karena ChemCompute dan Python memiliki fungsi berbeda: ChemCompute digunakan untuk menjalankan perhitungan kimia kuantum aktual berbasis paket komputasi, sedangkan Python digunakan untuk memahami model, membuat grafik, menghitung parameter sederhana, dan menganalisis output.

2.4 Kapan Menggunakan ChemCompute dan Kapan Menggunakan Python

ChemCompute dan Python memiliki fungsi yang berbeda dalam praktikum kimia kuantum, tetapi keduanya saling melengkapi. ChemCompute digunakan terutama untuk menjalankan **perhitungan kimia kuantum aktual** berbasis paket komputasi, misalnya perhitungan energi elektronik, optimasi geometri, frekuensi vibrasi, dan sifat molekul melalui mesin komputasi seperti

GAMESS atau paket lain yang tersedia pada platform tersebut. Python digunakan terutama untuk **memahami model, membuat simulasi sederhana, mengolah data, membuat grafik, menghitung parameter turunan, membaca output, dan menyusun analisis yang reprodusibel**. ChemCompute menjawab kebutuhan "menjalankan perhitungan molekul nyata", sedangkan Python menjawab kebutuhan "memahami, memeriksa, mengolah, dan memvisualisasikan hasil perhitungan".

Perbedaan fungsi ini harus dipahami sejak awal agar mahasiswa tidak keliru menempatkan keduanya. ChemCompute bukan pengganti seluruh teori kimia kuantum, dan Python bukan pengganti otomatis semua paket struktur elektronik profesional. ChemCompute adalah gerbang berbasis web menuju perhitungan kimia komputasi tanpa instalasi lokal yang rumit; halaman resminya menyebut ChemCompute sebagai perangkat lunak kimia komputasi untuk pengajaran dan riset sarjana tanpa kerepotan kompilasi, instalasi, dan pemeliharaan software maupun hardware. (ChemCompute) Python adalah bahasa pemrograman umum yang banyak digunakan dalam scientific computing karena sintaksnya relatif mudah, fleksibel, dan mampu diintegrasikan dengan pustaka numerik, visualisasi, serta paket kimia komputasi; situs resmi Python menekankan bahwa Python memungkinkan pengguna bekerja cepat dan mengintegrasikan sistem secara efektif. (Python.org)

Dalam desain buku praktikum ini, ChemCompute dan Python tidak ditempatkan sebagai dua jalur yang bersaing, tetapi sebagai dua lapisan kerja. **Lapisan pertama** adalah lapisan komputasi struktur elektronik, yaitu perhitungan molekul berbasis metode kuantum atau semiempiris. Pada lapisan ini, ChemCompute sangat berguna karena mahasiswa dapat menjalankan job melalui web. **Lapisan kedua** adalah lapisan analisis, validasi, dan representasi data. Pada lapisan ini, Python sangat berguna karena mahasiswa dapat menulis kode untuk membaca data, menghitung parameter, membuat tabel, dan memvisualisasikan grafik. Kombinasi keduanya menghasilkan praktikum yang tidak hanya mudah diakses, tetapi juga analitis dan reprodusibel.

Secara historis, pembagian fungsi ini muncul karena kimia komputasi berkembang dari dua tradisi yang saling bertemu. Tradisi pertama adalah tradisi **paket kimia kuantum**, seperti GAMESS, Gaussian, ORCA, Psi4, dan PySCF, yang dirancang untuk menyelesaikan persoalan struktur elektronik molekul. Tradisi kedua adalah tradisi **komputasi ilmiah umum**, yaitu penggunaan bahasa pemrograman dan pustaka numerik untuk menyelesaikan persamaan, mengolah data, membuat grafik, dan membangun workflow analisis. ChemCompute mewakili perkembangan science gateway berbasis web yang menurunkan hambatan akses terhadap paket kimia komputasi. Python mewakili perkembangan scientific programming yang membuat analisis data menjadi eksplisit, terbuka, dan dapat diulang.

ChemCompute sebaiknya digunakan ketika mahasiswa membutuhkan hasil perhitungan yang secara langsung memerlukan mesin kimia komputasi. Misalnya, untuk mengoptimasi geometri H_2O dengan metode Hartree–Fock atau DFT, mahasiswa memerlukan paket struktur elektronik. Untuk menghitung energi total formaldehida, mahasiswa memerlukan algoritma SCF dan basis set. Untuk memperoleh frekuensi vibrasi, mahasiswa memerlukan perhitungan turunan energi terhadap koordinat inti. Untuk menghitung momen dipol dari distribusi elektron, mahasiswa memerlukan hasil struktur elektronik. Dalam situasi seperti ini, ChemCompute berfungsi sebagai sarana praktis untuk menjalankan paket komputasi tanpa harus menginstal sendiri.

Sebaliknya, Python sebaiknya digunakan ketika mahasiswa perlu memahami model, mengolah data, dan membangun representasi hasil. Misalnya, untuk menghitung energi partikel dalam kotak, Python sudah cukup karena model matematisnya sederhana dan dapat diprogram langsung. Untuk membuat grafik fungsi gelombang ψ dan rapat probabilitas ψ^2, Python sangat tepat. Untuk menghitung panjang ikatan dari koordinat XYZ, Python cukup. Untuk membuat grafik energi terhadap panjang ikatan, Python sangat berguna. Untuk menghitung gap HOMO-LUMO dari data output, Python efisien. Untuk membuat spektrum IR simulasi dari daftar frekuensi dan intensitas, Python lebih fleksibel daripada menyalin grafik dari software. SciPy, misalnya, menyediakan algoritma untuk optimasi, integrasi, interpolasi, nilai eigen, persamaan aljabar, persamaan diferensial, statistik, dan berbagai kelas masalah scientific computing lain. (SciPy)

Perbedaan paling mendasar antara ChemCompute dan Python dapat dijelaskan melalui konsep **tingkat abstraksi**. ChemCompute bekerja pada tingkat paket komputasi: pengguna memberikan input kimia, lalu sistem menjalankan program yang kompleks di belakang layar. Python bekerja pada tingkat algoritma dan analisis: pengguna menulis instruksi eksplisit untuk menghitung, membaca, memfilter, mengonversi, dan memvisualisasikan data. Dalam ChemCompute, mahasiswa mungkin tidak melihat seluruh detail algoritma SCF, integral basis set, atau prosedur optimasi. Dalam Python, mahasiswa dapat melihat secara langsung bagaimana suatu grafik dibuat, bagaimana satuan dikonversi, bagaimana jarak ikatan dihitung, atau bagaimana nilai ΔG diperoleh dari energi reaktan dan produk.

Namun, perbedaan ini tidak berarti Python selalu lebih "mendasar" dan ChemCompute selalu lebih "praktis". Keduanya memiliki kedalaman masing-masing. ChemCompute memberi akses pada perhitungan yang secara teoritis dan numerik jauh lebih kompleks daripada kode Python sederhana. Sebaliknya, Python memberi kesempatan kepada mahasiswa untuk membuka sebagian proses analisis yang sering tersembunyi dalam software. Karena itu, pertanyaan "mana yang lebih baik, ChemCompute atau Python?" adalah pertanyaan yang keliru. Pertanyaan yang benar adalah: **untuk tujuan praktikum tertentu, bagian mana yang sebaiknya dijalankan dengan ChemCompute, dan bagian mana yang sebaiknya dianalisis dengan Python?**

Dalam praktikum optimasi geometri, misalnya, ChemCompute digunakan untuk menghitung struktur minimum molekul. Mahasiswa dapat menggambar molekul, memilih metode, menjalankan optimasi, dan mengunduh output. Python kemudian digunakan untuk membaca koordinat akhir dan menghitung panjang ikatan, sudut ikatan, atau perubahan geometri. Dengan demikian, ChemCompute menjawab "di mana geometri minimum menurut metode tertentu?", sedangkan Python menjawab "bagaimana kita menganalisis dan menyajikan geometri itu secara kuantitatif?". Sumber buku kimia kuantum yang digunakan juga menegaskan bahwa praktikum komputasi modern sebaiknya menyertakan struktur awal, metode, basis set, kriteria konvergensi, output, dan interpretasi; hasil komputasi harus ditafsirkan secara kimia, bukan hanya dilaporkan sebagai angka.

Dalam praktikum Hartree–Fock dan basis set, ChemCompute dapat digunakan untuk menjalankan energi molekul dengan beberapa basis set. Mahasiswa dapat menghitung H_2O pada HF/STO-3G, HF/3-21G, HF/6-31G, dan HF/6-31G(d), jika paket yang digunakan mendukung. Python kemudian digunakan untuk menyusun tabel energi, menghitung selisih energi antar basis set, membuat grafik energi terhadap basis set, dan menampilkan waktu komputasi jika datanya

tersedia. ChemCompute menghasilkan data energi, sedangkan Python membangun pola dan interpretasi. Tanpa Python, mahasiswa mungkin hanya memiliki daftar angka. Dengan Python, mahasiswa dapat melihat kecenderungan bahwa basis set lebih besar biasanya menurunkan energi total dan meningkatkan biaya komputasi.

Dalam praktikum DFT dan HOMO-LUMO, ChemCompute digunakan untuk menjalankan perhitungan struktur elektronik molekul. Python digunakan untuk menghitung gap HOMO-LUMO dan membuat diagram tingkat energi. Misalnya, mahasiswa mengambil energi HOMO dan LUMO etilena, butadiena, benzena, formaldehida, dan nitrobenzena. Python kemudian menghitung gap, membuat grafik perbandingan, dan membantu mahasiswa menafsirkan hubungan antara konjugasi, substituen, stabilitas elektronik, dan reaktivitas. Dengan demikian, ChemCompute menghasilkan data orbital, sedangkan Python membantu mengubah data itu menjadi argumentasi kimia.

Dalam praktikum frekuensi IR, ChemCompute digunakan untuk menjalankan optimasi dan frekuensi. Python digunakan untuk membuat spektrum IR simulasi dari daftar frekuensi dan intensitas. Jika mahasiswa hanya membaca daftar frekuensi dalam output, mereka mungkin kesulitan membayangkan bentuk spektrum. Dengan Python, setiap puncak dapat diberi pelebaran Gaussian atau Lorentzian, lalu diplot sebagai spektrum. Mahasiswa dapat melihat posisi puncak O–H, C=O, N–H, C–H, atau C–O secara visual. Di sinilah Python berperan sebagai alat representasi ilmiah. Sumber buku kimia kuantum yang digunakan menjelaskan bahwa frekuensi vibrasi, energi titik nol, dan koreksi termal merupakan bagian penting dalam kimia komputasi; laporan juga perlu membandingkan hasil dengan data rujukan dan menganalisis perbedaan dari sisi metode, basis set, korelasi elektron, efek pelarut, dan keterbatasan model.

Dalam praktikum energi reaksi, ChemCompute digunakan untuk menghitung energi reaktan dan produk. Python digunakan untuk menghitung ΔE, ΔH, atau ΔG, mengonversi Hartree ke kJ/mol, menyusun diagram energi reaksi, dan memeriksa konsistensi stoikiometri. Misalnya, jika mahasiswa menghitung isomerisasi cis-2-butena menjadi trans-2-butena, ChemCompute menghasilkan energi masing-masing isomer. Python menghitung perbedaan energi dan membuat diagram yang menunjukkan isomer mana lebih stabil pada tingkat teori yang digunakan. Tanpa Python, mahasiswa mudah salah konversi satuan atau salah menjumlahkan koefisien reaksi. Dengan Python, prosedur perhitungan menjadi eksplisit dan dapat diperiksa.

Secara praktis, pedoman penggunaan ChemCompute dan Python dapat diringkas dalam tabel berikut.

Tujuan Praktikum	**Gunakan ChemCompute Ketika**	**Gunakan Python Ketika**
Partikel dalam kotak	Tidak wajib, kecuali memakai Jupyter di platform web	Menghitung E_n, ψ, ψ^2, membuat grafik
Optimasi geometri	Menjalankan optimasi molekul nyata	Menghitung jarak, sudut, membandingkan geometri
Single point energy	Menghitung energi elektronik dengan metode/basis tertentu	Membuat tabel energi dan grafik perbandingan

Tujuan Praktikum	Gunakan ChemCompute Ketika	Gunakan Python Ketika
Hartree–Fock dan basis set	Menjalankan variasi basis set	Menganalisis tren energi dan biaya komputasi
DFT dan HOMO-LUMO	Menghasilkan energi orbital dan output elektronik	Menghitung gap, hardness, softness, diagram orbital
Simetri dan momen dipol	Menghitung momen dipol dan geometri	Mengolah vektor dipol sederhana dan grafik polaritas
Frekuensi IR	Menghitung frekuensi dan intensitas	Membuat spektrum IR simulasi
Energi reaksi	Menghitung energi reaktan dan produk	Menghitung ΔE, ΔH, ΔG, diagram energi
Analisis output	Menghasilkan file output	Parsing output, ekstraksi data, tabel otomatis

Tabel tersebut menunjukkan bahwa ChemCompute lebih dekat dengan **produksi data komputasi molekuler**, sedangkan Python lebih dekat dengan **pengolahan dan pemaknaan data**. Dalam pembelajaran, kedua fungsi ini harus disatukan. Mahasiswa tidak cukup menjalankan job; mereka harus menganalisisnya. Mahasiswa tidak cukup membuat grafik; mereka harus memahami dari mana data grafik berasal. ChemCompute tanpa Python berisiko menghasilkan praktikum klik-otomatis. Python tanpa data komputasi aktual berisiko menjadi latihan matematika yang terpisah dari molekul nyata. Integrasi keduanya menghasilkan pembelajaran yang lebih utuh.

Dari sudut pandang metodologi, ChemCompute sebaiknya digunakan untuk perhitungan yang membutuhkan algoritma struktur elektronik yang tidak realistis jika ditulis sendiri oleh mahasiswa pemula. Algoritma Hartree–Fock, DFT, integrasi numerik, optimasi geometri kuantum, dan analisis frekuensi merupakan pekerjaan komputasi yang kompleks. Walaupun secara teori dapat dipelajari, implementasi lengkapnya terlalu berat untuk praktikum dasar. Karena itu, mahasiswa menggunakan paket profesional melalui ChemCompute. Sebaliknya, Python digunakan untuk pekerjaan yang secara pedagogis berguna jika ditulis sendiri: menghitung fungsi sederhana, membaca koordinat, membuat grafik, menghitung selisih energi, dan menyusun tabel.

Dalam bahasa pendidikan, ChemCompute mendukung **access to computation**, sedangkan Python mendukung **computational thinking**. Access to computation berarti mahasiswa memiliki akses ke mesin dan software yang dapat menghitung sistem molekuler. Computational thinking berarti mahasiswa memahami bagaimana masalah dipecah menjadi data, variabel, fungsi, algoritma, dan visualisasi. Keduanya penting. Mahasiswa yang memiliki akses tetapi tidak memiliki computational thinking akan menjadi pengguna pasif. Mahasiswa yang memiliki computational thinking tetapi tidak memiliki akses ke mesin perhitungan akan sulit mempelajari molekul nyata. Buku ini menggabungkan keduanya agar mahasiswa menjadi pengguna yang kritis dan kreatif.

Secara kritis, ada beberapa situasi ketika ChemCompute sebaiknya tidak menjadi pilihan utama. Pertama, jika tujuan pembelajaran adalah memahami model matematis sederhana seperti partikel dalam kotak, osilator harmonik dasar, atau distribusi probabilitas, Python lebih tepat karena mahasiswa perlu melihat persamaan bekerja secara langsung. Kedua, jika hanya diperlukan pengolahan data output yang sudah tersedia, ChemCompute tidak diperlukan. Ketiga, jika job

terlalu besar atau terlalu kompleks untuk platform pembelajaran, lebih baik menggunakan server lokal atau mengurangi kompleksitas sistem. Keempat, jika mahasiswa ingin belajar otomasi workflow intensif, Python dengan Psi4 atau PySCF mungkin lebih sesuai untuk tahap lanjut. Psi4 sendiri menyediakan tutorial penggunaan sebagai executable dan sebagai modul Python, sehingga dapat menjembatani input tradisional dan workflow programatik. (PsiCode)

Sebaliknya, ada situasi ketika Python biasa tidak cukup. Jika mahasiswa ingin memperoleh energi elektronik formaldehida pada tingkat DFT, Python dasar dengan numpy dan scipy saja tidak cukup, kecuali mahasiswa mengimplementasikan metode struktur elektronik sendiri, yang tidak realistis untuk praktikum dasar. Jika mahasiswa ingin menghitung frekuensi vibrasi kuantum dari molekul nyata, ia membutuhkan output dari paket kimia komputasi. Jika mahasiswa ingin memperoleh momen dipol dari distribusi elektron, ia membutuhkan perhitungan struktur elektronik. Dalam situasi ini, ChemCompute atau paket kuantum lain diperlukan. Python kemudian masuk setelah data dihasilkan.

Keterbatasan Python juga harus dipahami. Python memang mudah dan kuat, tetapi kode yang salah akan menghasilkan hasil salah. Grafik yang dibuat dengan Python dapat tampak profesional meskipun datanya keliru. Konversi satuan yang salah dapat menghasilkan kesimpulan energi reaksi yang menyesatkan. Parsing output yang tidak teliti dapat mengambil angka dari bagian output yang salah. Oleh sebab itu, penggunaan Python harus disertai validasi. Kode harus diuji dengan data sederhana. Hasil harus diperiksa secara manual untuk beberapa contoh. Variabel harus diberi nama jelas. Satuan harus ditulis. File input data harus disimpan. Ini sejalan dengan prinsip reprodusibilitas yang telah dibahas pada Bab 1.

Keterbatasan ChemCompute juga harus dipahami. ChemCompute memudahkan akses, tetapi pengguna mungkin memiliki kontrol teknis lebih terbatas dibanding instalasi lokal. Job bergantung pada koneksi internet dan ketersediaan server. Paket dan opsi yang tersedia dapat berbeda dari kebutuhan riset lanjut. Karena itu, ChemCompute paling tepat digunakan untuk praktikum dasar dan menengah, pengenalan konsep, serta proyek mini sarjana. Untuk riset lanjut, mahasiswa dapat beralih ke ORCA, Gaussian, GAMESS lokal, Psi4, PySCF, NWChem, atau server HPC kampus. Dokumen rujukan software kimia komputasi juga menegaskan bahwa komputasi awan mengalihkan beban pemeliharaan server kepada penyedia layanan dan memungkinkan mahasiswa menjalankan pekerjaan komputasi dari perangkat sederhana, tetapi manajemen hasil dan data output tetap menjadi tantangan penting.

Dalam pembelajaran, dosen dapat menggunakan prinsip **urutan tiga tahap**. Tahap pertama adalah **Python konseptual**, yaitu mahasiswa memahami model sederhana seperti partikel dalam kotak melalui kode. Tahap kedua adalah **ChemCompute aktual**, yaitu mahasiswa menjalankan perhitungan molekul nyata. Tahap ketiga adalah **Python analitis**, yaitu mahasiswa mengolah output ChemCompute. Urutan ini sangat kuat secara pedagogis karena mahasiswa memulai dari teori yang dapat dikodekan, lalu masuk ke perhitungan nyata, lalu kembali ke Python untuk analisis. Pola ini membentuk siklus: **model → perhitungan → data → analisis → interpretasi**.

Contoh urutan tiga tahap dapat diterapkan pada praktikum HOMO-LUMO. Pada tahap Python konseptual, mahasiswa membuat diagram tingkat energi sederhana untuk dua atau tiga orbital. Pada tahap ChemCompute, mahasiswa menjalankan perhitungan etilena dan butadiena. Pada tahap

Python analitis, mahasiswa mengambil energi HOMO-LUMO dari output, menghitung gap, dan membuat diagram energi. Dengan demikian, mahasiswa tidak hanya melihat output, tetapi memahami representasi tingkat energi dari awal.

Contoh lain adalah praktikum IR. Pada tahap Python konseptual, mahasiswa belajar membuat spektrum dari daftar frekuensi dan intensitas buatan. Pada tahap ChemCompute, mahasiswa menjalankan perhitungan frekuensi formaldehida. Pada tahap Python analitis, mahasiswa membuat spektrum IR simulasi dari data output. Pola ini membantu mahasiswa memahami bahwa spektrum bukan gambar misterius, tetapi representasi grafis dari puncak vibrasi yang memiliki frekuensi dan intensitas.

Dalam konteks proyek mini, mahasiswa dapat diberi kebebasan memilih proporsi ChemCompute dan Python sesuai pertanyaan riset. Jika proyeknya adalah “perbandingan gap HOMO-LUMO turunan benzena”, ChemCompute digunakan untuk menghasilkan energi orbital, sedangkan Python digunakan untuk membuat tabel, grafik, dan analisis substituen. Jika proyeknya adalah “simulasi model partikel dalam kotak untuk sistem poliena”, Python mungkin menjadi alat utama, sedangkan ChemCompute digunakan hanya sebagai pembanding dengan molekul nyata. Jika proyeknya adalah “efek pelarut terhadap energi relatif konformer”, ChemCompute digunakan untuk menghitung energi fase gas dan pelarut, sedangkan Python digunakan untuk menghitung selisih energi dan membuat diagram. Dengan cara ini, mahasiswa belajar memilih alat berdasarkan pertanyaan ilmiah.

Integrasi ChemCompute dan Python juga memperkuat reprodusibilitas laporan. ChemCompute menyediakan output yang dapat disimpan, sedangkan Python menyediakan kode analisis yang dapat dijalankan ulang. Jika laporan hanya memuat output, pembaca dapat melihat hasil, tetapi tidak selalu mengetahui bagaimana data diolah. Jika laporan hanya memuat grafik, pembaca tidak mengetahui asal data. Jika laporan memuat output ChemCompute, file data, dan kode Python, maka proses ilmiah menjadi jauh lebih transparan. Inilah standar yang perlu ditanamkan sejak praktikum awal.

Dari sudut pandang kompetensi mahasiswa, penggunaan ChemCompute dan Python menghasilkan dua jenis literasi. Pertama, **literasi software kimia komputasi**, yaitu kemampuan membangun struktur, memilih metode, menjalankan job, membaca output, dan memvalidasi hasil. Kedua, **literasi data dan programming**, yaitu kemampuan mengolah data, membuat grafik, menulis kode, dan menyusun workflow. Mahasiswa kimia masa kini membutuhkan keduanya. Riset kimia modern tidak hanya menuntut pemahaman teori, tetapi juga kemampuan bekerja dengan data digital dan alat komputasi.

Dalam perspektif lebih luas, Python juga membuka jalur menuju software kimia komputasi yang lebih programatik seperti Psi4 dan PySCF. Psi4 disebut sebagai paket quantum chemistry open-source dengan inti C++/Python yang dapat diintegrasikan dan diperluas melalui ekosistem software. (PsiCode) PySCF dirancang sebagai platform struktur elektronik berbasis Python yang menekankan kesederhanaan kode, fleksibilitas workflow, serta dukungan untuk metode mean-field dan post-mean-field. (arXiv) Ini menunjukkan bahwa Python tidak hanya berguna untuk grafik sederhana, tetapi juga menjadi bahasa penting dalam pengembangan kimia komputasi

modern. Namun, untuk buku praktikum dasar, penggunaan Python dimulai dari hal yang paling pedagogis: analisis data dan simulasi konsep.

Analisis kritis perlu diberikan agar mahasiswa tidak terjebak pada dua ekstrem. Ekstrem pertama adalah **software-dependence**, yaitu ketergantungan penuh pada ChemCompute atau software lain tanpa memahami data. Mahasiswa dalam ekstrem ini dapat menjalankan job, tetapi tidak mampu menjelaskan hasil. Ekstrem kedua adalah **coding-formalism**, yaitu terlalu fokus menulis kode tetapi kehilangan konteks kimia. Mahasiswa dalam ekstrem ini dapat membuat grafik, tetapi tidak memahami makna kimia dari grafik tersebut. Buku ini menghindari kedua ekstrem dengan menempatkan ChemCompute dan Python dalam satu alur praktikum yang seimbang.

Secara operasional, aturan sederhana berikut dapat digunakan dalam setiap mata praktikum.

Gunakan ChemCompute jika:

1. Perhitungan memerlukan paket kimia kuantum aktual.
2. Mahasiswa perlu memperoleh energi elektronik molekul nyata.
3. Mahasiswa perlu melakukan optimasi geometri.
4. Mahasiswa perlu menghitung frekuensi vibrasi.
5. Mahasiswa perlu memperoleh output HOMO-LUMO, momen dipol, atau muatan atom.
6. Mahasiswa tidak memiliki instalasi software lokal.
7. Praktikum membutuhkan akses cepat dan seragam untuk kelas.

Gunakan Python jika:

1. Tujuan utama adalah memahami model matematis.
2. Mahasiswa perlu membuat grafik fungsi gelombang atau probabilitas.
3. Mahasiswa perlu membaca koordinat dan menghitung geometri.
4. Mahasiswa perlu mengolah output menjadi tabel.
5. Mahasiswa perlu mengonversi satuan.
6. Mahasiswa perlu menghitung ΔE, ΔH, atau ΔG dari data energi.
7. Mahasiswa perlu membuat spektrum IR simulasi.
8. Mahasiswa perlu membuat laporan yang reprodusibel.

Gunakan keduanya jika:

1. Praktikum menghasilkan data komputasi yang perlu dianalisis.
2. Mahasiswa perlu memahami hubungan antara output software dan konsep kimia.
3. Dosen ingin menilai proses secara reprodusibel.
4. Praktikum diarahkan pada proyek mini atau pembelajaran berbasis inkuiri.

Dalam laporan praktikum, penggunaan ChemCompute dan Python harus ditulis secara jelas. Contoh penulisan yang baik adalah:

“Perhitungan optimasi geometri formaldehida dilakukan melalui ChemCompute menggunakan paket GAMESS pada tingkat teori HF/STO-3G dengan charge 0 dan multiplicity 1. File output

diunduh dan dianalisis menggunakan Python untuk mengekstrak energi akhir, koordinat final, panjang ikatan C=O, sudut H–C–H, serta membuat tabel hasil."

Kalimat tersebut menunjukkan pembagian peran yang jelas. ChemCompute digunakan untuk menjalankan perhitungan, Python digunakan untuk analisis. Ini lebih baik daripada kalimat "hasil dihitung dengan komputer" atau "data dianalisis secara komputasi", karena kalimat tersebut terlalu umum dan tidak reprodusibel.

Implikasi praktis bagi dosen adalah perlunya menyusun tugas yang memaksa mahasiswa menggunakan kedua alat secara bermakna. Misalnya, dosen tidak cukup meminta mahasiswa "jalankan optimasi H_2O". Tugas yang lebih baik adalah: "Jalankan optimasi H_2O melalui ChemCompute, unduh output, ekstrak koordinat final, hitung panjang ikatan dan sudut ikatan menggunakan Python, lalu bandingkan dengan data literatur." Tugas kedua lebih kaya karena melibatkan perhitungan, dokumentasi, analisis, validasi, dan interpretasi.

Implikasi bagi mahasiswa adalah perlunya mengembangkan sikap adaptif. Mahasiswa harus memahami bahwa alat dapat berubah. Hari ini mereka menggunakan ChemCompute, besok mungkin ORCA lokal, minggu depan Psi4, dan pada proyek akhir mungkin PySCF atau Gaussian. Namun, prinsip alur kerja tetap sama: struktur awal, input, metode, basis set, charge, multiplicity, perhitungan, output, validasi, analisis, dan laporan. Python membantu menjaga kontinuitas karena analisis data dapat dilakukan lintas software selama data dapat diekstrak.

ChemCompute dan Python juga memiliki implikasi sosial dalam pendidikan kimia. ChemCompute memperluas akses bagi mahasiswa yang tidak memiliki komputer kuat atau kampus yang belum memiliki server. Python memperluas kemandirian karena mahasiswa dapat membuat alat analisis sendiri tanpa bergantung penuh pada software komersial. Kombinasi keduanya mendukung pembelajaran yang lebih demokratis, terbuka, dan inovatif. Dokumen rujukan software kimia komputasi menekankan bahwa ChemCompute membantu mendemokratisasi akses terhadap perangkat lunak kimia kuantum tingkat tinggi dan memungkinkan institusi dengan anggaran terbatas tetap menyelenggarakan praktikum kimia fisik berstandar baik.

Pada akhirnya, keputusan kapan menggunakan ChemCompute dan kapan menggunakan Python harus selalu kembali pada pertanyaan ilmiah. Jika pertanyaannya membutuhkan penyelesaian struktur elektronik molekul nyata, gunakan ChemCompute atau paket komputasi lain. Jika pertanyaannya membutuhkan pemahaman model, analisis data, grafik, konversi satuan, atau pengolahan output, gunakan Python. Jika pertanyaannya menuntut keduanya, gunakan keduanya secara terintegrasi. Dengan prinsip ini, mahasiswa akan belajar bahwa teknologi bukan tujuan akhir, melainkan alat untuk memahami kimia secara lebih dalam.

Subbab berikutnya akan membahas kombinasi ChemCompute, Avogadro, dan Python. Pembahasan tersebut akan memperlihatkan alur kerja yang lebih konkret: Avogadro digunakan untuk membangun struktur, ChemCompute digunakan untuk menjalankan job, dan Python digunakan untuk menganalisis hasil.

2.5 Kombinasi ChemCompute, Avogadro, dan Python

Kombinasi ChemCompute, Avogadro, dan Python merupakan alur kerja praktikum kimia kuantum yang menyatukan tiga fungsi utama: **pembuatan struktur molekul**, **pelaksanaan perhitungan kimia komputasi**, dan **analisis hasil secara programatik**. Avogadro digunakan untuk membangun dan memeriksa struktur molekul awal. ChemCompute digunakan untuk menjalankan perhitungan kimia kuantum melalui platform berbasis web tanpa instalasi lokal yang rumit. Python digunakan untuk membaca, mengolah, menghitung ulang parameter, membuat grafik, menyusun tabel, dan memvalidasi hasil output. Dengan demikian, ketiganya membentuk satu sistem kerja yang utuh: **Avogadro sebagai ruang konstruksi molekul, ChemCompute sebagai mesin perhitungan, dan Python sebagai laboratorium analisis data**.

Alur kerja ini penting karena praktikum kimia kuantum tidak cukup hanya mengandalkan satu perangkat lunak. Jika mahasiswa hanya menggunakan Avogadro, mereka dapat membuat struktur molekul tetapi belum memperoleh energi kuantum yang sahih. Jika mahasiswa hanya menggunakan ChemCompute, mereka dapat menjalankan perhitungan tetapi berisiko menjadi pengguna pasif yang hanya mengunduh output. Jika mahasiswa hanya menggunakan Python, mereka dapat membuat grafik dan menghitung parameter sederhana, tetapi tidak otomatis dapat menghasilkan data struktur elektronik molekul nyata tanpa mesin kimia kuantum. Karena itu, integrasi ketiganya menjawab tiga kebutuhan sekaligus: struktur yang benar, perhitungan yang dapat dijalankan, dan analisis yang reprodusibel.

Secara fungsional, alur kombinasi tersebut dapat digambarkan sebagai berikut:

```
Avogadro
→ membangun struktur molekul awal
→ menambah hidrogen
→ membersihkan geometri awal
→ menyimpan XYZ/MOL/SDF

ChemCompute
→ memilih paket komputasi
→ menetapkan metode dan basis set
→ menentukan charge dan multiplicity
→ menjalankan optimasi, energi, frekuensi, atau job lain
→ mengunduh output

Python
→ membaca koordinat dan output
→ menghitung jarak, sudut, torsi, gap energi, ΔE, ΔG
→ membuat grafik dan tabel
→ memvalidasi serta menyusun laporan
```

Dalam kerangka pendidikan kimia, alur ini sangat kuat karena menempatkan mahasiswa sebagai pelaku ilmiah aktif. Mahasiswa tidak hanya melihat struktur molekul di layar, tetapi membangun struktur itu sendiri. Mahasiswa tidak hanya menerima data dari buku, tetapi menghasilkan data melalui perhitungan. Mahasiswa tidak hanya menyalin angka dari output, tetapi mengolah dan memeriksa angka tersebut melalui kode. Dengan cara ini, praktikum kimia kuantum menjadi latihan berpikir ilmiah: membangun model, menjalankan perhitungan, menguji hasil, menafsirkan data, dan menyusun kesimpulan.

Secara historis, kombinasi tiga alat ini mencerminkan perkembangan kimia komputasi modern. Pada masa awal, perhitungan kimia kuantum dilakukan melalui input teks manual dan terminal. Pengguna harus menulis koordinat atom, metode, basis set, serta parameter komputasi secara teliti. Seiring perkembangan perangkat lunak visualisasi, editor molekul seperti Avogadro memungkinkan struktur dibangun secara grafis dan diekspor ke berbagai format. Avogadro sendiri dideskripsikan sebagai editor molekul dan alat visualisasi bebas serta open-source yang digunakan dalam kimia komputasi, pemodelan molekul, pendidikan kimia, bioinformatika, dan ilmu material. (Avogadro) Artikel Hanwell dkk. juga menggambarkan Avogadro sebagai editor dan visualizer molekul lintas platform yang dapat digunakan untuk membangun struktur, memformat file input, dan menganalisis output berbagai paket kimia komputasi. (PMC)

Perkembangan berikutnya adalah munculnya platform komputasi berbasis web seperti ChemCompute. Platform ini mengurangi hambatan instalasi dan pemeliharaan software. ChemCompute menyebut dirinya sebagai software kimia komputasi untuk pengajaran dan riset sarjana tanpa kerepotan kompilasi, instalasi, dan pemeliharaan software maupun hardware. (ChemCompute) Dalam pendidikan, hal ini sangat signifikan karena tidak semua mahasiswa memiliki komputer kuat, tidak semua kampus memiliki server, dan tidak semua dosen memiliki waktu untuk membantu instalasi perangkat lunak di banyak komputer mahasiswa. ChemCompute membantu memindahkan beban teknis tersebut ke platform web sehingga fokus praktikum dapat diarahkan pada pemahaman input, output, dan interpretasi kimia.

Sementara itu, Python berkembang menjadi bahasa penting dalam komputasi ilmiah karena ekosistem pustakanya mendukung analisis numerik, visualisasi, dan pemrosesan data. NumPy menyediakan fungsi matematika, aljabar linear, dan struktur array untuk scientific computing; Matplotlib menyediakan pustaka visualisasi untuk membuat grafik statis, animasi, dan interaktif; SciPy menyediakan algoritma untuk optimasi, integrasi, interpolasi, persamaan nilai eigen, statistik, dan berbagai persoalan scientific computing. (NumPy) Dengan demikian, Python tidak hanya menjadi bahasa pemrograman, tetapi menjadi ruang analisis ilmiah yang dapat menjembatani output kimia komputasi dengan tabel, grafik, dan kesimpulan.

Dalam praktikum kimia kuantum, Avogadro berperan pada tahap **preprocessing**. Preprocessing adalah tahap sebelum perhitungan dijalankan. Pada tahap ini, mahasiswa membangun struktur molekul, menambahkan atom hidrogen, memeriksa valensi, mengatur konformasi, membersihkan geometri awal dengan force field sederhana, serta menyimpan struktur dalam format yang dapat digunakan lebih lanjut. Tahap ini tidak boleh dianggap remeh. Banyak perhitungan gagal atau menghasilkan kesimpulan keliru karena struktur awal salah. Atom terlalu dekat, hidrogen kurang, muatan tidak sesuai, atau konformer tidak tepat dapat mengubah hasil perhitungan.

Misalnya, ketika mahasiswa ingin menghitung etanol, ia harus memastikan bahwa struktur CH_3CH_2OH memiliki semua atom H yang lengkap, ikatan C–C, C–O, dan O–H benar, serta torsi awal tidak terlalu tidak wajar. Avogadro dapat membantu membersihkan struktur awal sehingga tidak ada atom yang saling bertumpuk. Namun, mahasiswa harus memahami bahwa optimasi force field di Avogadro bukan optimasi kuantum. Hasil dari Avogadro masih merupakan struktur awal yang perlu dioptimasi lebih lanjut melalui perhitungan kimia kuantum. Kesalahan umum mahasiswa adalah menganggap struktur yang sudah “rapi” di Avogadro sebagai struktur final.

Padahal, struktur final dalam praktikum kimia kuantum harus diperoleh dari job optimasi geometri pada tingkat teori tertentu.

Setelah struktur dibangun di Avogadro, mahasiswa menyimpan file, misalnya dalam format XYZ atau MOL. Format XYZ sangat mudah digunakan untuk koordinat Cartesian, sedangkan MOL menyimpan informasi konektivitas yang lebih eksplisit. Struktur tersebut kemudian dapat diunggah atau diadaptasi untuk ChemCompute. Pada tahap ini, mahasiswa harus memastikan bahwa file struktur sesuai dengan molekul yang dimaksud. Jika file XYZ berisi jumlah atom yang salah, simbol atom yang keliru, atau koordinat yang rusak, perhitungan dapat gagal. Oleh karena itu, mahasiswa sebaiknya membuka file struktur dengan editor teks sebelum menggunakannya.

ChemCompute kemudian berperan pada tahap **perhitungan utama**. Setelah struktur tersedia, mahasiswa memilih paket komputasi, menentukan jenis job, metode, basis set, charge, dan multiplicity. ChemCompute menyediakan akses berbasis web untuk menjalankan perhitungan kimia komputasi tanpa instalasi lokal yang kompleks. Halaman GAMESS ChemCompute, misalnya, menyatakan bahwa pengguna perlu membuat akun atau login agar dapat mengakses fungsi penuh termasuk menyimpan submission eksperimen. (ChemCompute) Dengan demikian, ChemCompute berfungsi sebagai ruang eksekusi job, sedangkan input yang dikirim tetap harus dipahami dan didokumentasikan.

Dalam alur Avogadro–ChemCompute–Python, ChemCompute bukan hanya tempat “menekan tombol submit”. ChemCompute adalah titik di mana keputusan ilmiah dibuat. Mahasiswa harus menjawab pertanyaan: metode apa yang dipilih? Basis set apa yang digunakan? Apakah sistem netral atau bermuatan? Apakah sistem singlet, doublet, atau triplet? Apakah job yang dijalankan optimasi geometri, single point energy, atau frekuensi vibrasi? Apakah molekul terlalu besar untuk metode yang dipilih? Apakah struktur awal sudah layak? Semua pertanyaan ini harus dijawab sebelum job dikirim.

Misalnya, untuk air, mahasiswa dapat memilih charge 0 dan multiplicity 1. Untuk amonium, charge harus +1 dan multiplicity umumnya 1. Untuk radikal metil, multiplicity harus 2. Untuk oksigen molekuler keadaan dasar, multiplicity harus 3. Jika mahasiswa salah memasukkan charge atau multiplicity, hasil perhitungan dapat keliru walaupun output tampak selesai normal. Oleh karena itu, penggunaan ChemCompute harus selalu disertai pemahaman kimia, bukan sekadar mengikuti antarmuka.

Setelah job selesai, mahasiswa mengunduh output. Output inilah yang kemudian masuk ke tahap Python. Python berperan pada tahap **postprocessing** dan **analisis reprodusibel**. Postprocessing berarti mengambil data dari output, memilih bagian yang relevan, mengubahnya menjadi tabel, menghitung parameter turunan, dan menyajikannya dalam bentuk grafik atau narasi ilmiah. Dalam praktikum kimia kuantum, postprocessing sama pentingnya dengan perhitungan utama. Output yang panjang tidak otomatis menjadi pengetahuan. Ia harus dibaca, disaring, dihitung, dibandingkan, dan ditafsirkan.

Contoh sederhana adalah menghitung panjang ikatan dan sudut ikatan dari file XYZ hasil optimasi. ChemCompute menghasilkan koordinat akhir. Python membaca koordinat tersebut, lalu menghitung jarak antaratom menggunakan rumus jarak Euclidean:

$$d_{AB} = \sqrt{[(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2]}$$

Untuk sudut ikatan, Python dapat menggunakan operasi vektor dan dot product. Dengan demikian, mahasiswa tidak hanya menyalin panjang ikatan dari software, tetapi memahami bagaimana panjang dan sudut dihitung dari koordinat. Ini memperkuat literasi spasial dan matematis mahasiswa.

Dalam praktikum optimasi geometri air, alur lengkapnya dapat dijelaskan sebagai berikut. Pertama, mahasiswa menggambar H_2O di Avogadro. Kedua, mahasiswa memastikan atom H lengkap dan geometri awal masuk akal. Ketiga, file disimpan sebagai `h2o_initial.xyz`. Keempat, struktur digunakan di ChemCompute untuk menjalankan optimasi geometri dengan metode dan basis set tertentu, misalnya HF/STO-3G untuk latihan awal atau DFT/B3LYP dengan basis set yang sesuai jika tersedia. Kelima, output diunduh sebagai `h2o_hf_sto3g_opt.out`. Keenam, koordinat akhir diekstrak sebagai `h2o_final.xyz`. Ketujuh, Python digunakan untuk menghitung panjang ikatan O–H, sudut H–O–H, dan membuat tabel hasil. Kedelapan, mahasiswa membandingkan hasil dengan teori VSEPR dan data literatur.

Pada contoh ini, setiap alat memiliki fungsi yang jelas. Avogadro membangun struktur. ChemCompute menghitung struktur minimum. Python memeriksa dan menyajikan hasil. Jika salah satu tahap dihilangkan, pembelajaran menjadi kurang utuh. Tanpa Avogadro, mahasiswa mungkin kesulitan membuat struktur awal. Tanpa ChemCompute, mahasiswa tidak memperoleh hasil kuantum aktual. Tanpa Python, mahasiswa cenderung hanya menyalin angka dan tidak membangun analisis reprodusibel.

Contoh kedua adalah praktikum formaldehida untuk spektrum IR. Avogadro digunakan untuk membangun struktur H_2CO. ChemCompute digunakan untuk menjalankan optimasi dan frekuensi. Output frekuensi kemudian diunduh. Python digunakan untuk membaca daftar frekuensi dan intensitas, lalu membuat spektrum IR simulasi. Mahasiswa dapat mengidentifikasi puncak regangan C=O, membandingkan dengan rentang literatur, dan menjelaskan mengapa frekuensi komputasi harmonik dapat berbeda dari eksperimen. Dengan alur ini, spektrum tidak hanya menjadi gambar, tetapi hasil dari proses komputasi yang dapat dilacak dari struktur awal sampai grafik akhir.

Contoh ketiga adalah analisis HOMO-LUMO pada etilena, butadiena, dan benzena. Avogadro digunakan untuk membangun ketiga struktur. ChemCompute digunakan untuk menjalankan perhitungan struktur elektronik. Output digunakan untuk mengambil energi HOMO dan LUMO jika tersedia. Python digunakan untuk menghitung gap HOMO-LUMO dan membuat diagram perbandingan tingkat energi. Dari hasil tersebut, mahasiswa dapat membahas pengaruh konjugasi terhadap gap energi. Etilena memiliki sistem π lebih pendek, butadiena memiliki delokalisasi lebih panjang, sedangkan benzena memiliki sistem aromatik. Dengan alur terpadu, mahasiswa belajar bahwa konsep konjugasi dapat dianalisis secara kuantitatif melalui data komputasi.

Contoh keempat adalah energi reaksi sederhana. Misalnya mahasiswa membandingkan energi relatif konformer anti dan gauche butana. Avogadro digunakan untuk membangun dua konformer awal. ChemCompute digunakan untuk mengoptimasi keduanya dengan metode dan basis set yang sama. Python digunakan untuk mengambil energi akhir, mengonversi satuan dari Hartree ke

kJ/mol, menghitung selisih energi, dan membuat diagram energi. Mahasiswa kemudian menafsirkan konformer mana yang lebih stabil pada tingkat teori yang digunakan. Alur ini mengajarkan bahwa perbandingan energi harus konsisten: metode sama, basis set sama, charge sama, multiplicity sama, dan status optimasi sama.

Integrasi Avogadro, ChemCompute, dan Python juga memperkuat prinsip reprodusibilitas. Setiap tahap meninggalkan jejak file. Dari Avogadro ada file struktur awal. Dari ChemCompute ada file input dan output. Dari Python ada file kode, data, tabel, dan grafik. Jika semua file disimpan dengan benar, orang lain dapat mengulangi atau memeriksa praktikum tersebut. Ini sejalan dengan prinsip pada Bab 1 bahwa setiap angka harus memiliki jejak. Energi harus dapat ditelusuri ke output. Output harus dapat ditelusuri ke input. Input harus dapat ditelusuri ke struktur awal. Grafik harus dapat ditelusuri ke data dan kode.

Struktur folder untuk alur kombinasi ini sebaiknya dibuat konsisten. Misalnya:

```
praktikum_optimasi_h2o/
├── input/
│   └── h2o_hf_sto3g_opt.inp
├── output/
│   └── h2o_hf_sto3g_opt.out
├── data/
│   ├── h2o_initial.xyz
│   ├── h2o_final.xyz
│   └── h2o_summary.csv
├── python/
│   └── analisis_geometri_h2o.py
├── gambar/
│   └── struktur_h2o.png
└── laporan/
    └── laporan_h2o.docx
```

Struktur folder ini bukan sekadar kerapian administratif. Ia adalah bagian dari metode ilmiah. Dengan struktur semacam ini, mahasiswa dapat menemukan file dengan cepat, dosen dapat memeriksa output, dan laporan dapat ditelusuri. Praktikum yang tidak memiliki struktur file rapi akan mudah kehilangan data, salah mengambil output, atau mencampur hasil dari metode berbeda.

Dari sisi teori pembelajaran, kombinasi tiga alat ini mendukung pendekatan **representasi multipel**. Avogadro menghadirkan representasi visual struktur molekul. ChemCompute menghadirkan representasi numerik melalui energi, koordinat, frekuensi, orbital, dan sifat molekul. Python menghadirkan representasi analitik melalui grafik, tabel, dan perhitungan ulang. Dalam kimia, pemahaman sering muncul ketika mahasiswa mampu menghubungkan representasi makroskopik, submikroskopik, simbolik, dan matematis. Praktikum kimia kuantum berbasis tiga alat ini membantu mahasiswa bergerak di antara representasi tersebut.

Misalnya, pada molekul air, Avogadro menampilkan bentuk bengkok secara visual. ChemCompute menghasilkan energi dan geometri teroptimasi. Python menghitung sudut H–O–H dan membuat tabel perbandingan. Konsep VSEPR, polaritas, dan momen dipol kemudian dibahas berdasarkan struktur dan angka. Dengan demikian, mahasiswa tidak hanya “melihat” air sebagai

gambar, tetapi memahami air sebagai sistem molekul yang dapat dimodelkan, dihitung, dan dianalisis.

Secara kritis, integrasi ini juga memiliki risiko. Risiko pertama adalah mahasiswa terlalu bergantung pada Avogadro dan tidak memeriksa struktur. Struktur yang tampak benar belum tentu benar secara kimia. Hidrogen dapat kurang, muatan dapat salah, atau konformer tidak sesuai. Risiko kedua adalah mahasiswa menganggap ChemCompute sebagai black box. Mereka menjalankan job tanpa memahami input. Risiko ketiga adalah mahasiswa menggunakan Python hanya untuk membuat grafik yang tampak indah, tetapi tidak memeriksa apakah data valid. Risiko keempat adalah file dari tiga tahap tidak dikelola dengan baik sehingga hasil tidak dapat direproduksi.

Untuk menghindari risiko tersebut, setiap praktikum harus memiliki **titik kontrol**. Titik kontrol pertama berada setelah Avogadro: mahasiswa harus memeriksa struktur awal, jumlah atom, valensi, hidrogen, dan format file. Titik kontrol kedua berada sebelum ChemCompute: mahasiswa harus memeriksa metode, basis set, charge, multiplicity, dan jenis job. Titik kontrol ketiga berada setelah ChemCompute: mahasiswa harus memeriksa status job, konvergensi, energi akhir, dan frekuensi imajiner jika ada. Titik kontrol keempat berada pada Python: mahasiswa harus memeriksa satuan, rumus, data input, dan hasil grafik. Jika titik kontrol ini diterapkan, alur praktikum menjadi jauh lebih kuat.

Berikut contoh checklist integrasi Avogadro–ChemCompute–Python.

Tahap Avogadro

1. Molekul sudah sesuai rumus kimia.
2. Semua atom H lengkap.
3. Valensi masuk akal.
4. Struktur awal tidak memiliki atom bertumpuk.
5. File disimpan sebagai XYZ/MOL/SDF.
6. Nama file jelas dan sistematis.

Tahap ChemCompute

1. Paket komputasi dipilih sesuai tujuan.
2. Job type sesuai: optimasi, energi, frekuensi, atau lainnya.
3. Metode ditulis lengkap.
4. Basis set ditulis lengkap.
5. Charge benar.
6. Multiplicity benar.
7. Input disimpan.
8. Output diunduh setelah selesai.
9. Status konvergensi diperiksa.

Tahap Python

1. File data yang dibaca jelas.
2. Satuan ditulis eksplisit.
3. Rumus perhitungan benar.
4. Tabel hasil tersimpan.
5. Grafik memiliki label sumbu dan satuan.
6. Kode disimpan dan dapat dijalankan ulang.
7. Kesimpulan sesuai data.

Checklist ini perlu dipahami sebagai alat pembelajaran, bukan beban tambahan. Mahasiswa yang terbiasa menggunakan checklist akan lebih jarang melakukan kesalahan dasar dan lebih siap melakukan penelitian komputasi yang serius.

Dari perspektif sintesis literatur, kombinasi Avogadro, ChemCompute, dan Python mencerminkan tiga arah penting dalam pendidikan kimia komputasi. Pertama, penggunaan editor molekul visual seperti Avogadro membantu mahasiswa membangun representasi tiga dimensi dan menyiapkan input. Kedua, platform web seperti ChemCompute membantu memperluas akses terhadap perhitungan kimia komputasi tanpa instalasi lokal. Ketiga, Python memperkuat kemampuan data science dan scientific computing yang semakin penting dalam sains modern. Dokumen rujukan software kimia komputasi juga menegaskan bahwa integrasi komputasi ke dalam kurikulum bukan sekadar mengganti alat bantu, tetapi merupakan pergeseran epistemologis dalam cara pengetahuan kimia diproduksi, divalidasi, dan ditransmisikan.

Implikasi pedagogisnya adalah dosen sebaiknya tidak mengajarkan software secara terpisah tanpa alur. Mengajarkan Avogadro saja akan menghasilkan kemampuan menggambar struktur. Mengajarkan ChemCompute saja akan menghasilkan kemampuan mengirim job. Mengajarkan Python saja akan menghasilkan kemampuan membuat grafik. Namun, menggabungkan ketiganya akan menghasilkan kompetensi ilmiah yang lebih lengkap: mahasiswa mampu membangun molekul, menjalankan perhitungan, menganalisis output, dan menyusun kesimpulan berbasis data. Inilah kompetensi yang dibutuhkan dalam praktikum kimia kuantum modern.

Implikasi praktisnya adalah setiap praktikum dalam buku ini sebaiknya dirancang dengan pola tetap. Pertama, mahasiswa membangun struktur di Avogadro. Kedua, mahasiswa menyimpan struktur awal. Ketiga, mahasiswa menjalankan perhitungan melalui ChemCompute atau paket lain yang tersedia. Keempat, mahasiswa mengunduh output. Kelima, mahasiswa menganalisis data dengan Python. Keenam, mahasiswa menulis laporan dengan melampirkan input, output, kode, data, dan grafik. Dengan pola berulang, mahasiswa akan terbiasa pada alur kerja profesional.

Alur ini juga fleksibel. Jika kampus memiliki ORCA lokal, ChemCompute dapat diganti dengan ORCA. Jika kampus memiliki lisensi Gaussian, ChemCompute dapat diganti atau dilengkapi dengan Gaussian. Jika mahasiswa sudah mahir Python, analisis dapat diperluas menggunakan Psi4 atau PySCF. Namun, prinsip dasarnya tetap sama: **struktur dibuat dengan benar, perhitungan dijalankan secara terdokumentasi, dan hasil dianalisis secara reprodusibel**. Dengan demikian, buku ini tidak bergantung secara eksklusif pada satu platform, tetapi membangun pola berpikir komputasi yang dapat dipindahkan ke berbagai software.

Contoh konkret lain dapat diberikan pada praktikum asam asetat. Mahasiswa menggambar asam asetat di Avogadro, memastikan bentuk gugus karboksilat benar, dan menyimpan struktur awal. Melalui ChemCompute, mahasiswa menjalankan optimasi geometri dan frekuensi. Output digunakan untuk mengambil panjang ikatan C=O, C–O, O–H, serta frekuensi regangan C=O. Python kemudian digunakan untuk membuat tabel geometri dan spektrum IR simulasi. Mahasiswa dapat membandingkan hasil dengan konsep resonansi gugus karboksil dan data literatur. Dalam satu praktikum ini, mahasiswa menghubungkan struktur, ikatan, vibrasi, dan analisis data.

Contoh pada senyawa aromatik juga sangat berguna. Mahasiswa membangun benzena, toluena, anilina, dan nitrobenzena di Avogadro. ChemCompute digunakan untuk memperoleh energi HOMO-LUMO atau sifat elektronik lain pada tingkat teori yang sama. Python digunakan untuk membuat grafik gap energi dan membandingkan pengaruh substituen donor atau akseptor elektron. Mahasiswa kemudian menafsirkan hasil dalam konteks efek resonansi dan induksi. Praktikum semacam ini menghubungkan kimia organik dan kimia kuantum secara konkret.

Dalam praktikum energi reaksi, alur tiga alat ini juga mengajarkan konsistensi. Misalnya, mahasiswa ingin menghitung energi reaksi protonasi amonia:

$$NH_3 + H^+ \rightarrow NH_4^+$$

Avogadro digunakan untuk membangun NH_3 dan NH_4^+. ChemCompute digunakan untuk menghitung energi masing-masing spesies dengan metode yang sama. Python digunakan untuk menghitung selisih energi. Namun, mahasiswa juga harus menyadari bahwa perhitungan proton bebas dan efek pelarut memerlukan perhatian khusus. Ini membuka ruang diskusi kritis: tidak semua reaksi sederhana secara persamaan kimia sederhana pula secara komputasi. Dengan demikian, kombinasi alat ini juga melatih mahasiswa memahami batas model.

Kombinasi Avogadro, ChemCompute, dan Python juga mendukung asesmen yang lebih baik. Dosen dapat menilai struktur awal dari file Avogadro/XYZ, menilai perhitungan dari output ChemCompute, dan menilai analisis dari kode Python. Penilaian tidak hanya berdasarkan jawaban akhir, tetapi berdasarkan proses. Jika hasil mahasiswa salah, dosen dapat melihat apakah kesalahan terjadi pada struktur awal, input, output, atau kode. Ini membuat umpan balik lebih spesifik dan mendidik.

Dari sudut pandang mahasiswa, alur ini memberi rasa kepemilikan terhadap data. Mereka tidak hanya diberi data oleh dosen, tetapi menghasilkan data sendiri. Mereka tidak hanya mengunduh output, tetapi mengubahnya menjadi tabel dan grafik. Mereka tidak hanya mengikuti instruksi, tetapi belajar mengontrol proses ilmiah. Rasa kepemilikan ini penting untuk membangun motivasi dan kemandirian.

Namun, alur ini harus diperkenalkan secara bertahap. Pada awal Bab 2, mahasiswa cukup memahami peta software dan fungsi masing-masing. Pada subbab ini, mahasiswa memahami integrasi Avogadro–ChemCompute–Python. Pada Bab 3, mahasiswa akan belajar Python dasar yang diperlukan untuk analisis. Pada bab-bab praktikum, alur tersebut diterapkan secara konkret pada partikel dalam kotak, optimasi geometri, Hartree–Fock, DFT, simetri, IR, dan energi reaksi. Dengan demikian, pembelajaran bergerak dari konsep umum menuju keterampilan operasional.

Dalam konteks profesional, alur ini juga mencerminkan praktik riset kimia komputasi. Peneliti membangun struktur dengan editor molekul, menjalankan perhitungan dengan paket kuantum, menganalisis output dengan script, membuat grafik, dan menyusun artikel. Walaupun software yang digunakan dalam riset dapat lebih kompleks, pola dasarnya sama. Karena itu, mahasiswa yang menguasai alur Avogadro–ChemCompute–Python akan lebih siap memasuki riset lanjut.

Pada akhirnya, kombinasi ChemCompute, Avogadro, dan Python harus dipahami sebagai strategi pembelajaran yang menyatukan akses, visualisasi, komputasi, analisis, dan reprodusibilitas. Avogadro membuat molekul dapat dibangun dan dilihat. ChemCompute membuat perhitungan dapat dijalankan tanpa hambatan instalasi besar. Python membuat hasil dapat diperiksa, dihitung ulang, divisualisasikan, dan dilaporkan secara ilmiah. Ketiganya membentuk fondasi praktikum kimia kuantum yang modern, inklusif, dan bertanggung jawab.

Subbab berikutnya akan membahas standar file praktikum. Pembahasan tersebut penting karena alur Avogadro–ChemCompute–Python hanya dapat berjalan baik jika setiap praktikum memiliki folder input, output, python, gambar, laporan, dan data yang tersusun rapi, konsisten, serta mendukung reprodusibilitas.

2.6 Standar File Praktikum

Standar file praktikum adalah seperangkat aturan penataan folder, penamaan file, penyimpanan data, dokumentasi input-output, pengarsipan kode, dan penyusunan bukti komputasi yang digunakan agar seluruh kegiatan praktikum kimia kuantum dapat dilacak, diperiksa, diulang, dan dinilai secara akademik. Dalam praktikum komputasi, file bukan sekadar “lampiran teknis”, melainkan bagian inti dari metode ilmiah. Struktur molekul, file input, file output, kode Python, gambar orbital, tabel hasil, dan laporan adalah rantai bukti yang menghubungkan pertanyaan praktikum dengan kesimpulan akhir.

Dalam laboratorium basah, mahasiswa wajib mencatat bahan, alat, konsentrasi, suhu, volume, waktu reaksi, warna larutan, endapan, massa, atau absorbansi. Dalam laboratorium komputasi, bentuk catatan itu berubah menjadi file digital: struktur awal, koordinat atom, metode, basis set, charge, multiplicity, parameter konvergensi, output energi, frekuensi, orbital, data analisis, script Python, grafik, dan laporan. Karena itu, standar file praktikum berfungsi sebagai “buku catatan laboratorium digital”. Tanpa standar file, praktikum mudah berubah menjadi kumpulan file acak yang sulit ditelusuri, sulit diperiksa, dan rawan kesalahan akademik.

Sumber kimia kuantum yang menjadi dasar buku ini menegaskan bahwa setiap praktikum komputasi sebaiknya menyertakan struktur awal, muatan, multiplicity, metode, basis set, kriteria konvergensi, serta file output yang dapat diperiksa ulang; tanpa dokumentasi input dan output, hasil komputasi sulit diverifikasi. Prinsip pelaporan yang sama juga menegaskan bahwa data wajib bergerak dari struktur awal menuju metode/basis, kriteria konvergensi, output, dan interpretasi. Dengan demikian, standar file bukan tambahan administratif, tetapi syarat dasar agar praktikum kimia kuantum memiliki validitas ilmiah.

Secara konseptual, standar file praktikum memiliki empat fungsi utama. Pertama, **fungsi reprodusibilitas**, yaitu memastikan orang lain dapat mengulang perhitungan berdasarkan file

yang tersedia. Kedua, **fungsi audit akademik**, yaitu memungkinkan dosen memeriksa apakah hasil benar-benar berasal dari output yang sah. Ketiga, **fungsi efisiensi kerja**, yaitu membuat mahasiswa mudah menemukan input, output, data, kode, gambar, dan laporan. Keempat, **fungsi pembelajaran ilmiah**, yaitu membiasakan mahasiswa bekerja seperti peneliti yang bertanggung jawab terhadap jejak data.

Dalam buku ini, setiap praktikum wajib menggunakan struktur folder standar sebagai berikut:

```
praktikum_nama_topik/
├── input/
├── output/
├── python/
├── gambar/
├── laporan/
└── data/
```

Keenam folder tersebut memiliki peran berbeda. Folder `input/` menyimpan file instruksi perhitungan. Folder `output/` menyimpan hasil mentah dari software. Folder `python/` menyimpan kode analisis. Folder `gambar/` menyimpan visualisasi struktur, orbital, spektrum, diagram energi, atau grafik. Folder `laporan/` menyimpan dokumen laporan praktikum. Folder `data/` menyimpan file data terstruktur seperti CSV, XYZ final, tabel ringkasan, atau hasil ekstraksi dari output. Pemisahan ini membuat alur kerja praktikum menjadi tertib dan mudah diperiksa.

1. Folder `input/`: Tempat Menyimpan Instruksi Perhitungan

Folder `input/` adalah tempat menyimpan semua file yang digunakan untuk menjalankan job komputasi. File input dapat berupa input GAMESS, ORCA, Gaussian, Psi4, NWChem, atau file struktur awal yang dikirim ke ChemCompute. Folder ini penting karena input adalah dokumen metodologis utama. Dari input, dosen dapat mengetahui molekul apa yang dihitung, metode apa yang dipakai, basis set apa yang dipilih, charge dan multiplicity berapa, serta jenis job apa yang dijalankan.

Contoh isi folder `input/`:

```
input/
├── h2o_hf_sto3g_opt.inp
├── h2o_b3lyp_631gd_optfreq.inp
├── formaldehyde_b3lyp_631gd_freq.inp
└── butane_anti_b3lyp_def2svp_opt.inp
```

Nama file harus informatif. Nama `h2o_hf_sto3g_opt.inp` lebih baik daripada `input1.inp` karena langsung menyatakan molekul, metode, basis set, dan jenis job. Penamaan file yang baik memperkuat keterlacakan. Ketika mahasiswa memiliki banyak file, nama yang tidak jelas akan menyebabkan kebingungan. Kesalahan mengambil input dapat menghasilkan laporan yang salah, terutama ketika mahasiswa membandingkan beberapa metode atau beberapa molekul.

Folder `input/` juga harus menyimpan struktur awal jika struktur itu digunakan sebagai dasar input. Misalnya, jika molekul dibangun dengan Avogadro dan diekspor sebagai XYZ sebelum dikirim

ke ChemCompute, file tersebut juga dapat disimpan di `input/` atau `data/struktur_awal/`. Yang penting, struktur awal tidak boleh hilang. Dalam optimasi geometri, struktur awal memengaruhi jalur optimasi dan kemungkinan minimum yang ditemukan. Molekul fleksibel seperti butana, etanol, asam asetat, atau sikloheksana dapat memiliki beberapa konformer. Tanpa struktur awal, sulit mengetahui apakah hasil akhir berasal dari konformer yang dimaksud.

Secara teoritis, input adalah pernyataan eksplisit dari model ilmiah. Jika mahasiswa memilih HF/STO-3G, ia sedang memilih pendekatan Hartree–Fock dengan basis set minimal. Jika memilih B3LYP/6-31G(d), ia sedang memilih DFT dengan fungsional hybrid dan basis set split-valence terpolarisasi. Jika menulis charge 0 dan multiplicity 1, ia sedang menyatakan bahwa sistem diperlakukan sebagai molekul netral singlet. Jadi, input bukan sekadar file teknis. Input adalah bentuk tertulis dari asumsi, model, dan keputusan ilmiah.

2. Folder `output/`: Tempat Menyimpan Hasil Mentah Komputasi

Folder `output/` menyimpan file hasil perhitungan dari ChemCompute atau software lokal. File output dapat berupa `.out`, `.log`, `.dat`, atau format lain tergantung program. Output adalah data primer dalam praktikum komputasi. Sama seperti lembar hasil pengukuran dalam praktikum spektrofotometri, output harus disimpan apa adanya. Mahasiswa tidak boleh mengubah angka dalam output, menghapus bagian error, atau mengganti identitas job.

Contoh isi folder `output/`:

```
output/
├── h2o_hf_sto3g_opt.out
├── h2o_b3lyp_631gd_optfreq.log
├── formaldehyde_b3lyp_631gd_freq.out
└── butane_anti_b3lyp_def2svp_opt.out
```

Output perlu disimpan lengkap karena laporan hanya memuat ringkasan. Laporan mungkin hanya menuliskan energi akhir, panjang ikatan, atau frekuensi utama. Namun, jika dosen ingin memeriksa apakah perhitungan benar-benar konvergen, file output lengkap diperlukan. Sumber kimia kuantum yang digunakan menekankan bahwa hasil komputasi perlu ditafsirkan secara kimia, bukan sekadar dilaporkan sebagai angka; laporan juga perlu memuat perbandingan dengan data eksperimen atau literatur bila tersedia, serta menganalisis perbedaan dari sisi metode, basis set, korelasi elektron, efek pelarut, dan keterbatasan model. Analisis seperti itu hanya dapat dilakukan jika output tersedia dan dapat diperiksa.

Dalam folder `output/`, mahasiswa sebaiknya tidak hanya menyimpan output yang berhasil. Output yang gagal juga dapat disimpan dalam subfolder khusus, misalnya:

```
output/
├── berhasil/
└── gagal/
```

File gagal dapat menjadi bahan pembelajaran. Jika SCF tidak konvergen, jika optimasi berhenti sebelum selesai, atau jika input salah, mahasiswa dapat menganalisis penyebabnya. Dalam

pendidikan komputasi, kegagalan bukan sekadar kesalahan, tetapi sumber pemahaman. Dengan menyimpan output gagal, mahasiswa belajar memperbaiki input, memahami pesan error, dan bekerja lebih jujur. Menghapus seluruh bukti kegagalan dapat membuat proses belajar kehilangan jejak.

Output juga perlu diperiksa sebelum digunakan. File yang berada di folder `output/` belum otomatis valid. Mahasiswa harus membaca bagian penting: apakah program berhenti normal, apakah SCF konvergen, apakah optimasi geometri konvergen, apakah frekuensi imajiner muncul, dan apakah energi akhir benar-benar berasal dari geometri final. Jika job gagal tetapi file tetap terbentuk, mahasiswa tidak boleh mengambil angka terakhir sebagai hasil. Standar file harus diikuti dengan standar validasi.

3. Folder `python/`: Tempat Menyimpan Kode Analisis

Folder `python/` menyimpan seluruh script atau notebook yang digunakan untuk mengolah data. File dapat berupa `.py` atau `.ipynb`. Folder ini penting karena buku praktikum ini tidak hanya menggunakan software komputasi, tetapi juga programming ilmiah. Python digunakan untuk membaca koordinat, menghitung jarak dan sudut, mengonversi satuan, menghitung gap HOMO-LUMO, membuat spektrum IR simulasi, menghitung ΔE atau ΔG, serta membuat grafik. Dengan menyimpan kode, proses analisis menjadi transparan.

Contoh isi folder `python/`:

```
python/
├── analisis_geometri.py
├── plot_homo_lumo.py
├── spektrum_ir.py
├── hitung_energi_reaksi.py
└── lingkungan_python.txt
```

File `lingkungan_python.txt` dapat berisi versi Python dan pustaka yang digunakan, misalnya:

```
Python 3.11
numpy 1.26
pandas 2.2
matplotlib 3.8
scipy 1.12
```

Pencatatan ini mendukung reprodusibilitas. Walaupun praktikum dasar biasanya tidak terlalu sensitif terhadap versi pustaka, kebiasaan mencatat lingkungan komputasi sangat baik untuk penelitian lanjut. Dalam scientific computing, hasil analisis dapat bergantung pada pustaka, fungsi, atau cara eksekusi tertentu. Karena itu, kode dan lingkungan perlu disimpan.

Kode Python harus ditulis dengan nama variabel yang jelas. Misalnya, `energi_hartree`, `energi_kjmol`, `jarak_oh`, `sudut_hoh`, `frekuensi_cm1`, dan `intensitas_ir` lebih baik daripada `a`, `b`, `x1`, atau `hasil`. Kode juga harus memiliki komentar singkat untuk menjelaskan langkah penting, terutama konversi satuan. Jika mahasiswa mengonversi Hartree ke kJ/mol, konstanta

konversi harus ditulis eksplisit. Kesalahan konversi satuan adalah salah satu kesalahan umum dalam praktikum energi reaksi.

Folder `python/` juga menjadi bukti bahwa grafik dan tabel dalam laporan dibuat secara ilmiah. Jika laporan memuat diagram energi tetapi kode tidak disertakan, dosen sulit memeriksa asal grafik tersebut. Jika kode disertakan, dosen dapat melihat data yang dibaca, rumus yang dipakai, dan cara grafik dibuat. Dengan demikian, folder `python/` memperkuat integritas akademik.

4. Folder `gambar/`: Tempat Menyimpan Visualisasi

Folder `gambar/` menyimpan semua gambar yang digunakan dalam laporan. Gambar dapat berupa struktur molekul, orbital HOMO-LUMO, mode vibrasi, spektrum IR, diagram energi, grafik konvergensi, peta potensial elektrostatik, atau tangkapan layar antarmuka ChemCompute. Gambar penting karena kimia kuantum sering melibatkan konsep abstrak yang lebih mudah dipahami melalui visualisasi. Namun, gambar harus dikelola dengan standar ilmiah.

Contoh isi folder `gambar/`:

```
gambar/
├── h2o_struktur_awal.png
├── h2o_struktur_final.png
├── formaldehyde_ir_spectrum.png
├── butane_energy_diagram.png
├── benzene_homo.png
└── benzene_lumo.png
```

Setiap gambar harus memiliki sumber data yang jelas. Jika gambar struktur berasal dari Avogadro, file struktur yang digunakan harus tersedia. Jika gambar orbital berasal dari output komputasi, file output atau file orbitalnya harus tersedia. Jika grafik dibuat dengan Python, script dan data grafik harus tersedia. Gambar tidak boleh berdiri sendiri tanpa jejak data. Visualisasi adalah representasi data, bukan dekorasi laporan.

Dalam kimia kuantum, gambar orbital harus ditafsirkan dengan hati-hati. Warna lobus orbital biasanya menunjukkan fase fungsi gelombang, bukan muatan positif dan negatif dalam arti elektrostatis langsung. Ukuran orbital bergantung pada nilai isosurface. Mode vibrasi menunjukkan pola perpindahan atom, bukan gerak klasik yang diamati langsung. Peta potensial elektrostatik menunjukkan distribusi potensial relatif pada permukaan tertentu, bukan satu-satunya bukti reaktivitas. Oleh karena itu, folder `gambar/` harus didampingi narasi analitis dalam laporan.

Format gambar juga perlu diperhatikan. Untuk laporan, format `.png` biasanya cukup baik untuk grafik dan tangkapan layar. Untuk publikasi atau kualitas cetak, `.svg` atau `.pdf` lebih baik karena berbasis vektor. Mahasiswa sebaiknya menghindari gambar buram, terpotong, atau tanpa label. Grafik harus memiliki judul, label sumbu, satuan, dan legenda jika diperlukan. Spektrum IR harus menampilkan sumbu bilangan gelombang dalam cm^{-1} dan intensitas relatif. Diagram energi harus mencantumkan satuan energi, misalnya kJ/mol atau eV.

5. Folder `laporan/`: Tempat Menyimpan Dokumen Praktikum

Folder `laporan/` menyimpan dokumen laporan akhir praktikum. File dapat berupa `.docx`, `.pdf`, atau format lain sesuai ketentuan dosen. Folder ini tidak boleh menjadi tempat menyimpan semua file acak. Laporan adalah hasil sintesis, sedangkan input, output, data, kode, dan gambar berada di folder masing-masing. Pemisahan ini membuat laporan lebih bersih dan memudahkan pemeriksaan.

Contoh isi folder `laporan/`:

```
laporan/
├── laporan_praktikum_02_optimasi_geometri.docx
└── laporan_praktikum_02_optimasi_geometri.pdf
```

Laporan yang baik harus merujuk pada file pendukung. Misalnya, pada bagian metode, mahasiswa dapat menulis:

```
File input disimpan pada input/h2o_hf_sto3g_opt.inp, sedangkan output lengkap disimpan pada output/h2o_hf_sto3g_opt.out. Analisis geometri dilakukan dengan script python/analisis_geometri.py dan data koordinat final disimpan pada data/h2o_final.xyz.
```

Kalimat semacam ini membuat laporan menjadi reprodusibel. Dosen tidak harus menebak dari mana angka berasal. Jika ada kesalahan, dosen dapat menelusuri apakah kesalahan terjadi pada input, output, data, kode, atau interpretasi.

Dalam laporan, mahasiswa juga harus menyebut software yang digunakan. Misalnya, “struktur awal dibuat dengan Avogadro, perhitungan dilakukan melalui ChemCompute menggunakan GAMESS, visualisasi dilakukan dengan Jmol, dan analisis data dilakukan dengan Python.” Dokumen rujukan software kimia komputasi menegaskan bahwa ChemCompute bertujuan mendemokratisasi akses terhadap perangkat lunak kimia kuantum tingkat tinggi melalui antarmuka web, sehingga mahasiswa dapat menjalankan eksperimen tanpa instalasi Linux yang rumit. Jika platform atau software berperan dalam menghasilkan data, software tersebut harus dicatat sebagai bagian metode.

6. Folder `data/`: Tempat Menyimpan Data Terstruktur

Folder `data/` menyimpan data yang telah diekstrak atau disusun ulang dari input dan output. Data dapat berupa koordinat final `.xyz`, tabel energi `.csv`, tabel frekuensi `.csv`, file ringkasan `.txt`, data grafik `.csv`, atau metadata praktikum `.json` atau `.yaml`. Folder ini berbeda dari `output/`. Folder `output/` menyimpan file mentah dari software, sedangkan `data/` menyimpan data yang sudah dipilih, dibersihkan, atau diolah untuk analisis.

Contoh isi folder `data/`:

```
data/
├── h2o_initial.xyz
├── h2o_final.xyz
```

```
├── energi_basis_set.csv
├── frekuensi_formaldehida.csv
├── homo_lumo_benzena.csv
├── energi_reaksi.csv
└── metadata_praktikum.yaml
```

File CSV sangat berguna karena mudah dibaca oleh Python, spreadsheet, dan banyak software lain. Misalnya, tabel energi basis set dapat ditulis sebagai:

```
molekul,metode,basis_set,energi_hartree,status
H2O,HF,STO-3G,-74.9630,konvergen
H2O,HF,3-21G,-75.5850,konvergen
H2O,HF,6-31G,-75.9830,konvergen
```

File semacam ini memudahkan Python membuat grafik dan tabel. Namun, mahasiswa harus memastikan bahwa angka dalam `data/` benar-benar berasal dari output. Jangan mengetik ulang data tanpa memeriksa. Jika memungkinkan, data diekstrak langsung dari output menggunakan script Python. Jika data dimasukkan manual, mahasiswa harus memeriksa ulang untuk menghindari salah salin.

Folder `data/` juga dapat menyimpan metadata. Metadata adalah data tentang data. Dalam praktikum komputasi, metadata dapat memuat nama molekul, metode, basis set, charge, multiplicity, software, versi program, tanggal perhitungan, nama file input, nama file output, dan status konvergensi. Contoh metadata sederhana:

```
praktikum: optimasi_geometri_h2o
molekul: H2O
software_platform: ChemCompute
paket: GAMESS
metode: HF
basis_set: STO-3G
charge: 0
multiplicity: 1
job_type: optimize
input_file: input/h2o_hf_sto3g_opt.inp
output_file: output/h2o_hf_sto3g_opt.out
status_scf: konvergen
status_optimasi: konvergen
```

Metadata semacam ini mempercepat pemeriksaan. Jika laporan hanya memiliki output tanpa metadata, pembaca harus mencari sendiri informasi penting di file panjang. Jika metadata tersedia, konteks perhitungan langsung terlihat.

Standar Penamaan File

Penamaan file adalah bagian penting dari standar file praktikum. Nama file yang baik harus ringkas, konsisten, informatif, dan bebas dari spasi berlebihan. Disarankan menggunakan huruf kecil, angka, dan garis bawah. Hindari karakter khusus seperti `#`, `%`, `&`, `?`, atau tanda kurung berlebihan karena dapat menyebabkan masalah pada sistem operasi atau script.

Format penamaan yang disarankan:

```
molekul_metode_basis_job.ext
```

Contoh:

```
h2o_hf_sto3g_opt.inp
h2o_hf_sto3g_opt.out
h2o_b3lyp_631gd_optfreq.log
formaldehyde_b3lyp_631gd_freq.csv
benzene_b3lyp_def2svp_homo.png
butane_anti_b3lyp_def2svp_opt.out
```

Jika menggunakan basis set dengan tanda khusus seperti `6-31G(d)`, nama file dapat disederhanakan menjadi `631gd`. Jika menggunakan `def2-SVP`, dapat ditulis `def2svp`. Tujuannya adalah menjaga nama file tetap mudah dibaca dan aman dipakai dalam script.

Penamaan file juga harus mencerminkan variasi sistem. Jika mahasiswa membandingkan konformer, nama konformer harus dimasukkan, misalnya `butane_anti` dan `butane_gauche`. Jika membandingkan fase gas dan pelarut, nama file dapat memuat `gas` atau `water`. Jika membandingkan metode, metode harus tercantum. Contoh:

```
acetone_b3lyp_def2svp_gas_opt.out
acetone_b3lyp_def2svp_water_opt.out
ethanol_anti_b3lyp_631gd_opt.out
ethanol_gauche_b3lyp_631gd_opt.out
```

Standar ini mencegah tercampurnya file. Dalam praktikum energi reaksi, kekeliruan mengambil energi dari file yang salah dapat mengubah kesimpulan. Karena itu, nama file adalah bagian dari validitas data.

Standar Isi Minimal Setiap Praktikum

Setiap folder praktikum minimal harus memuat enam kelompok file:

Folder	Isi Minimal	Fungsi Akademik
`input/`	File input atau struktur awal yang dikirim ke software	Menjelaskan model dan parameter perhitungan
`output/`	File log/out hasil perhitungan	Bukti primer hasil komputasi
`python/`	Script atau notebook analisis	Menunjukkan proses pengolahan data
`gambar/`	Struktur, orbital, grafik, spektrum, diagram	Mendukung visualisasi dan komunikasi hasil
`laporan/`	Laporan akhir DOCX/PDF	Menyajikan sintesis ilmiah

Folder	Isi Minimal	Fungsi Akademik
`data/`	CSV, XYZ final, metadata, tabel ekstraksi	Menyediakan data terstruktur untuk analisis ulang

Tabel ini harus menjadi standar penilaian. Jika salah satu folder kosong tanpa alasan, mahasiswa perlu menjelaskan. Misalnya, pada praktikum partikel dalam kotak, folder `output/` mungkin tidak berisi output software kimia kuantum karena praktikum dilakukan sepenuhnya dengan Python. Namun, folder `python/`, `data/`, `gambar/`, dan `laporan/` tetap harus ada. Pada praktikum optimasi geometri, folder `input/` dan `output/` wajib ada karena perhitungan dijalankan melalui ChemCompute atau software komputasi.

Standar Metadata Praktikum

Selain file, setiap praktikum sebaiknya memiliki file metadata. File ini dapat diberi nama:

`metadata_praktikum.txt`

atau

`metadata_praktikum.yaml`

Isi metadata minimal:

```
nama_praktikum:
nama_mahasiswa:
tanggal:
molekul:
rumus_molekul:
software_visualisasi:
platform_komputasi:
paket_komputasi:
metode:
basis_set:
charge:
multiplicity:
job_type:
model_pelarut:
koreksi_dispersi:
input_file:
output_file:
python_script:
status_scf:
status_optimasi:
frekuensi_imajiner:
catatan:
```

Metadata ini membantu menghubungkan semua file. Dalam praktikum yang melibatkan beberapa molekul, metadata dapat dibuat per molekul atau dibuat dalam bentuk tabel CSV. Prinsipnya, semua informasi penting harus mudah ditemukan. Sumber kimia kuantum yang digunakan

merumuskan reprodusibilitas sebagai gabungan input lengkap, versi program, parameter komputasi, dan file output. Metadata adalah alat praktis untuk menerapkan prinsip tersebut.

Standar Versi dan Revisi File

Dalam praktikum, mahasiswa sering mengulang perhitungan. Misalnya, job pertama gagal, job kedua berhasil, atau struktur awal diperbaiki. Karena itu, file perlu memiliki versi. Penamaan versi dapat dilakukan sederhana:

```
h2o_hf_sto3g_opt_v1.out
h2o_hf_sto3g_opt_v2.out
```

Namun, versi tidak boleh dipakai secara sembarangan. Jika `v1` gagal dan `v2` berhasil, catat perbedaannya dalam metadata atau laporan. Misalnya:

```
v1: struktur awal terlalu rapat, optimasi gagal.
v2: struktur awal diperbaiki dengan Avogadro, optimasi konvergen.
```

Pencatatan revisi seperti ini memperkuat kejujuran ilmiah. Mahasiswa tidak perlu menyembunyikan kegagalan. Yang penting adalah menjelaskan penyebab dan perbaikannya. Dalam riset komputasi, iterasi perhitungan adalah hal biasa. Standar file membantu agar iterasi tersebut tidak menjadi kekacauan data.

Untuk proyek yang lebih besar, version control seperti Git dapat diperkenalkan. Namun, untuk praktikum dasar, penamaan versi dan catatan revisi sudah cukup. Yang tidak boleh dilakukan adalah menimpa file lama tanpa menyimpan salinan atau mencampur output dari beberapa versi tanpa keterangan.

Standar Validasi File Output

Setiap file output yang digunakan dalam laporan harus melewati validasi. Validasi minimal meliputi:

1. Program selesai normal.
2. SCF konvergen.
3. Optimasi geometri konvergen jika job optimasi.
4. Tidak ada frekuensi imajiner untuk struktur minimum jika job frekuensi.
5. Charge dan multiplicity sesuai.
6. Metode dan basis set sesuai dengan laporan.
7. Energi akhir diambil dari bagian yang benar.
8. Satuan energi, frekuensi, dan momen dipol jelas.

Sumber kimia kuantum yang digunakan menegaskan bahwa optimasi geometri harus disertai pengecekan frekuensi: semua frekuensi nyata untuk minimum, dan satu frekuensi imajiner untuk keadaan transisi orde pertama; pemilihan metode, basis set, model pelarut, dan analisis populasi juga harus dilaporkan secara eksplisit agar hasil dapat direplikasi. Dengan demikian, standar file

tidak dapat dipisahkan dari standar validasi. Menyimpan output saja belum cukup; output harus dibaca dan diperiksa.

Dalam folder `data/`, mahasiswa dapat membuat file `validasi.csv`:

```
file_output,status_scf,status_opt,frekuensi_imajiner,keterangan
h2o_hf_sto3g_opt.out,konvergen,konvergen,-,valid
formaldehyde_b3lyp_631gd_freq.out,konvergen,konvergen,0,valid
butane_gauche_b3lyp_def2svp_opt.out,konvergen,tidak_konvergen,-,ulang
```

Tabel semacam ini membuat proses pemeriksaan lebih cepat. Dosen dapat melihat file mana yang valid dan mana yang perlu diperbaiki.

Standar Data untuk Python

Agar Python dapat digunakan secara efektif, data harus disusun rapi. File CSV sebaiknya memiliki header yang jelas, satuan ditulis pada nama kolom, dan angka menggunakan format desimal konsisten. Contoh yang baik:

```
molekul,metode,basis_set,energi_hartree,dipol_debye,status
H2O,HF,STO-3G,-74.9630,1.82,valid
NH3,HF,STO-3G,-55.4550,1.47,valid
CO2,HF,STO-3G,-187.6500,0.00,valid
```

Contoh yang kurang baik:

```
air,-74.963,bagus
amonia,-55.455,ok
karbondioksida,-187.65,tidak polar
```

Contoh kedua kurang baik karena tidak mencantumkan metode, basis set, satuan, dan status validasi. Python memang masih dapat membaca data itu, tetapi secara akademik datanya lemah. Data yang baik harus dapat dipahami oleh manusia dan komputer.

Untuk energi reaksi, data harus menyertakan koefisien stoikiometri:

```
spesies,koefisien,energi_hartree
reaktan_A,-1,-154.123456
reaktan_B,-1,-40.123456
produk_C,1,-194.300000
```

Kemudian Python dapat menghitung:

```
ΔE = Σ(koefisien × energi_spesies)
```

Dengan cara ini, perhitungan energi reaksi menjadi eksplisit. Mahasiswa tidak hanya memasukkan angka akhir, tetapi menunjukkan bagaimana selisih energi dihitung.

Standar Backup dan Penyimpanan

File praktikum harus dicadangkan. Minimal, mahasiswa menyimpan folder praktikum di dua tempat: komputer lokal dan penyimpanan awan atau media eksternal. Ini penting karena file output dapat hilang, komputer dapat rusak, atau sesi ChemCompute dapat tidak lagi mudah diakses. Dokumen rujukan software kimia komputasi menyebut bahwa pada platform awan seperti ChemCompute, manajemen data output menjadi tantangan penting; output dapat diunduh dan dipanggil kembali untuk analisis Python lebih mendalam. Oleh karena itu, mahasiswa harus memiliki kebiasaan mengunduh dan menyimpan file segera setelah job selesai.

Backup sebaiknya dilakukan dengan struktur yang sama. Jangan hanya mengirim beberapa file terpisah ke pesan pribadi atau menyimpan tangkapan layar tanpa output. Backup yang baik mempertahankan folder `input/`, `output/`, `python/`, `gambar/`, `laporan/`, dan `data/`. Jika folder dikompresi, gunakan nama yang jelas, misalnya:

```
praktikum_02_optimasi_geometri_h2o_kasmui.zip
```

Untuk pengumpulan tugas, dosen dapat meminta mahasiswa mengunggah folder terkompresi. Dengan demikian, seluruh bukti praktikum terkumpul dalam satu paket.

Standar README Praktikum

Setiap folder praktikum sebaiknya memiliki file `README.txt` atau `README.md`. File ini menjelaskan isi folder dan cara menjalankan analisis. Contoh:

```
README PRAKTIKUM OPTIMASI GEOMETRI H2O

1. Struktur awal:
   data/h2o_initial.xyz

2. File input:
   input/h2o_hf_sto3g_opt.inp

3. File output:
   output/h2o_hf_sto3g_opt.out

4. Script Python:
   python/analisis_geometri.py

5. Cara menjalankan analisis:
   python python/analisis_geometri.py

6. Output analisis:
   data/h2o_summary.csv
   gambar/h2o_geometri.png

7. Catatan:
   SCF dan optimasi konvergen. Tidak dilakukan perhitungan frekuensi.
```

README membantu dosen dan mahasiswa lain memahami isi folder tanpa membuka satu per satu file. Dalam proyek yang lebih besar, README menjadi dokumen navigasi utama. Kebiasaan menulis README juga sangat baik untuk penelitian ilmiah karena banyak repositori riset modern menggunakan README untuk menjelaskan data, kode, dan workflow.

Standar File untuk Praktikum Berbasis ChemCompute

Jika praktikum menggunakan ChemCompute, file yang wajib disimpan minimal:

```
input/
└── file input atau salinan parameter job

output/
└── file output ChemCompute

data/
├── struktur_awal.xyz
├── struktur_final.xyz
└── ringkasan_hasil.csv

python/
└── script analisis output

gambar/
└── grafik atau visualisasi hasil

laporan/
└── laporan akhir
```

Mahasiswa juga harus mencatat paket yang digunakan melalui ChemCompute. Jangan hanya menulis "dihitung di ChemCompute". Tuliskan paket, metode, basis set, dan jenis job. ChemCompute berfungsi sebagai platform, sedangkan paket seperti GAMESS atau Jupyter Notebook memiliki peran teknis berbeda. Sumber software kimia komputasi menjelaskan bahwa ChemCompute memungkinkan mahasiswa menjalankan eksperimen menggunakan paket komputasi seperti GAMESS dan NWChem tanpa instalasi Linux rumit. Maka, laporan harus menyebut platform dan paketnya.

Standar File untuk Praktikum Berbasis Avogadro

Jika Avogadro digunakan, file yang perlu disimpan:

```
data/
├── molekul_initial.mol
├── molekul_initial.xyz
└── molekul_cleaned.xyz

gambar/
└── struktur_avogadro.png
```

Jika Avogadro digunakan untuk menghasilkan input ORCA atau Gaussian, file input tersebut disimpan di `input/`. Mahasiswa harus mencatat apakah struktur sudah dibersihkan dengan force

field atau belum. Pembersihan geometri awal tidak sama dengan optimasi kuantum, sehingga laporan harus membedakan keduanya.

Sumber rujukan software menyebut bahwa Avogadro dapat berperan sebagai perangkat pembangun input komputasi, termasuk menerjemahkan data struktur menjadi geometri tiga dimensi yang dapat digunakan dalam workflow komputasi. Namun, struktur dari basis data atau editor molekul tetap harus diperiksa, terutama hidrogen, muatan, dan konformasi.

Standar File untuk Praktikum Berbasis Python

Jika praktikum dilakukan terutama dengan Python, misalnya partikel dalam kotak, struktur folder tetap digunakan. Contoh:

```
praktikum_partikel_dalam_kotak/
├── input/
│   └── parameter_model.txt
├── output/
│   └── hasil_eksekusi.txt
├── python/
│   └── partikel_dalam_kotak.py
├── gambar/
│   ├── fungsi_gelombang.png
│   └── probabilitas.png
├── laporan/
│   └── laporan_partikel_dalam_kotak.docx
└── data/
    └── energi_partikel.csv
```

Walaupun tidak ada output dari software kimia kuantum, folder `output/` dapat menyimpan hasil eksekusi atau log kode. Folder `input/` dapat menyimpan parameter model, misalnya panjang kotak, massa partikel, nilai n, dan satuan. Dengan demikian, praktikum Python tetap reprodusibel.

Analisis Kritis terhadap Standar File

Standar file dapat dianggap merepotkan jika mahasiswa belum memahami tujuannya. Namun, tanpa standar file, praktikum komputasi menjadi rapuh. Banyak kesalahan akademik bermula dari manajemen file yang buruk: output tertukar, metode lupa dicatat, file final tidak jelas, data grafik tidak tersedia, input hilang, atau kode tidak dapat dijalankan ulang. Standar file mencegah masalah tersebut.

Kritik lain adalah bahwa standar file dapat terlalu kaku. Untuk praktikum sederhana, membuat banyak folder mungkin tampak berlebihan. Namun, standar ini dapat disesuaikan tanpa menghilangkan prinsipnya. Jika praktikum sangat sederhana, folder boleh berisi sedikit file. Yang penting, kategori input, output, data, kode, gambar, dan laporan tetap dikenali. Standar bukan tujuan akhir, tetapi alat agar kerja ilmiah tertib.

Dari perspektif filsafat ilmu, standar file menunjukkan bahwa pengetahuan ilmiah tidak hanya lahir dari teori dan angka, tetapi juga dari infrastruktur dokumentasi. Sebuah klaim ilmiah menjadi

kuat bukan hanya karena hasilnya masuk akal, tetapi karena prosesnya terbuka untuk diperiksa. Dalam kimia komputasi, keterbukaan itu diwujudkan melalui file input, output, data, dan kode. Dengan demikian, standar file adalah bagian dari epistemologi praktikum komputasi.

Implikasi Akademik dan Praktis

Implikasi akademik standar file adalah meningkatnya kualitas laporan. Laporan tidak lagi berdiri sendiri sebagai narasi, tetapi didukung oleh bukti digital. Mahasiswa dapat menunjukkan input, output, data, kode, dan gambar. Dosen dapat menilai proses, bukan hanya hasil akhir. Jika ada kesalahan, lokasi kesalahan dapat ditelusuri. Apakah kesalahan terjadi pada struktur awal? Apakah charge salah? Apakah output tidak konvergen? Apakah Python salah membaca data? Standar file membuat diagnosis pembelajaran lebih mudah.

Implikasi praktisnya adalah mahasiswa menjadi lebih siap untuk penelitian. Dalam skripsi, tesis, atau publikasi, manajemen file menjadi sangat penting. Peneliti kimia komputasi sering bekerja dengan ratusan output dan banyak variasi metode. Jika sejak praktikum mahasiswa terbiasa dengan struktur file rapi, mereka akan lebih siap menghadapi penelitian nyata. Kebiasaan kecil seperti menamai file dengan benar, menyimpan output, mencatat metadata, dan menulis README akan menjadi modal besar.

Implikasi etisnya juga jelas. Standar file mengurangi peluang fabrikasi dan falsifikasi. Jika input, output, kode, dan data harus disertakan, mahasiswa lebih sulit membuat angka palsu. Jika output dapat diperiksa, klaim konvergensi dapat diuji. Jika kode tersedia, grafik dapat direproduksi. Dengan demikian, standar file mendukung integritas akademik yang telah dibahas pada Bab 1.

Dalam konteks pembelajaran berbasis ChemCompute, standar file juga memperkuat kemandirian. ChemCompute memang menyimpan job pada platform, tetapi mahasiswa tetap harus mengunduh dan mengarsipkan hasil sendiri. Komputasi awan memberikan akses, tetapi tanggung jawab data tetap berada pada pengguna. Hal ini sejalan dengan pembahasan bahwa laboratorium awan membuka akses komputasi, tetapi manajemen hasil output menjadi tantangan yang perlu ditangani secara disiplin.

Contoh Paket Folder Praktikum Lengkap

Berikut contoh paket folder untuk praktikum optimasi geometri formaldehida:

```
praktikum_02_optimasi_formaldehida/
├── input/
│   └── formaldehyde_b3lyp_631gd_optfreq.inp
├── output/
│   └── formaldehyde_b3lyp_631gd_optfreq.out
├── python/
│   ├── analisis_geometri.py
│   └── spektrum_ir.py
├── gambar/
│   ├── formaldehyde_struktur_final.png
│   └── formaldehyde_ir_spectrum.png
├── laporan/
```

```
│   └── laporan_optimasi_formaldehida.docx
└── data/
    ├── formaldehyde_initial.xyz
    ├── formaldehyde_final.xyz
    ├── formaldehyde_frequencies.csv
    ├── formaldehyde_summary.csv
    └── metadata_praktikum.yaml
```

Dalam paket ini, semua bagian praktikum dapat ditelusuri. Struktur awal tersedia. Input tersedia. Output tersedia. Data frekuensi tersedia. Kode analisis tersedia. Gambar tersedia. Laporan tersedia. Metadata tersedia. Jika dosen ingin memeriksa hasil, seluruh jalur ilmiah dapat diikuti dari awal sampai akhir.

Contoh lain untuk praktikum HOMO-LUMO benzena:

```
praktikum_04_homo_lumo_benzena/
├── input/
│   └── benzene_b3lyp_def2svp_sp.inp
├── output/
│   └── benzene_b3lyp_def2svp_sp.out
├── python/
│   └── plot_homo_lumo.py
├── gambar/
│   ├── benzene_homo.png
│   ├── benzene_lumo.png
│   └── benzene_homo_lumo_diagram.png
├── laporan/
│   └── laporan_homo_lumo_benzena.docx
└── data/
    ├── benzene_final.xyz
    ├── benzene_orbital_energy.csv
    └── metadata_praktikum.yaml
```

Pola ini dapat diterapkan pada semua praktikum dalam buku ini. Perbedaan hanya pada jenis file dan data yang dihasilkan.

Sintesis Subbab

Standar file praktikum adalah fondasi teknis bagi seluruh praktikum kimia kuantum. Bab 2 telah memperkenalkan peta software, ChemCompute sebagai laboratorium berbasis web, alur kerja ChemCompute, perbedaan fungsi ChemCompute dan Python, serta kombinasi ChemCompute, Avogadro, dan Python. Semua pembahasan tersebut membutuhkan standar file agar dapat dijalankan secara konsisten. Tanpa standar file, ekosistem software menjadi tidak tertib. Dengan standar file, setiap praktikum memiliki struktur yang jelas: input disiapkan, output disimpan, data diekstrak, kode dianalisis, gambar divisualisasikan, dan laporan disusun.

Standar file juga menghubungkan Bab 2 dengan Bab 3. Bab 3 akan membahas dasar Python untuk praktikum kimia kuantum. Agar Python dapat bekerja baik, data harus rapi. Agar kode dapat membaca file, nama file harus konsisten. Agar grafik dapat dibuat ulang, data dan script harus tersimpan. Dengan demikian, standar file pada subbab ini menjadi jembatan langsung menuju pembelajaran Python pada bab berikutnya.

Penutup Bab 2

Bab 2 telah membahas fondasi operasional yang diperlukan sebelum mahasiswa memasuki praktikum kimia kuantum secara lebih teknis. Jika Bab 1 memberikan orientasi konseptual tentang pentingnya praktikum kimia kuantum in silico, maka Bab 2 menyiapkan mahasiswa pada aspek kerja praktis: lingkungan komputasi, perangkat lunak, struktur folder, format file, pengelolaan data, dan prinsip reprodusibilitas.

Bab ini menegaskan bahwa keberhasilan praktikum kimia komputasi tidak hanya ditentukan oleh kemampuan memahami teori kimia kuantum, tetapi juga oleh kesiapan teknis dalam mengelola perangkat dan data. Mahasiswa perlu mengetahui bagaimana menyiapkan komputer, membuka platform berbasis web seperti ChemCompute, menggunakan editor molekul, menjalankan Python, menyimpan file input-output, mengatur folder kerja, serta memastikan bahwa setiap hasil praktikum dapat ditelusuri kembali ke sumber datanya.

Salah satu gagasan penting dalam Bab 2 adalah bahwa **praktikum kimia komputasi harus dikerjakan secara tertib dan reprodusibel**. Setiap file input, file output, gambar molekul, grafik, tabel CSV, dan kode Python harus diberi nama yang jelas serta disimpan dalam struktur folder yang konsisten. Keteraturan ini bukan hanya untuk kerapian administratif, tetapi merupakan bagian dari etika kerja ilmiah. Hasil komputasi yang tidak dapat ditelusuri ulang akan sulit diverifikasi, sulit diperbaiki, dan sulit dipertanggungjawabkan dalam laporan akademik.

Bab ini juga memperkenalkan pentingnya memahami berbagai format file molekul, seperti XYZ, MOL, SDF, PDB, atau format input spesifik software. Format file bukan sekadar ekstensi, melainkan wadah informasi struktur molekul. Kesalahan kecil dalam koordinat atom, simbol unsur, muatan, multiplicity, atau format baris dapat menyebabkan perhitungan gagal atau menghasilkan output yang salah. Oleh karena itu, mahasiswa harus dilatih membaca dan memeriksa file sebelum menjalankan perhitungan.

Selain itu, Bab 2 menempatkan Python sebagai alat penting dalam praktikum. Python tidak hanya digunakan untuk menulis kode, tetapi juga untuk membangun literasi data. Melalui Python, mahasiswa dapat membaca CSV, membuat tabel, menghitung perubahan energi, membuat grafik, menyimpan output, dan menyusun analisis yang lebih sistematis. Dengan demikian, Python membantu mahasiswa bergerak dari penggunaan software secara pasif menuju analisis data yang aktif dan kritis.

Bab 2 juga memperkenalkan peran ChemCompute dan software kimia komputasi lain sebagai sarana pembelajaran. ChemCompute memudahkan mahasiswa mengakses perhitungan kimia komputasi tanpa harus menginstal seluruh software secara lokal. Sementara itu, software seperti GAMESS, ORCA, Gaussian, Psi4, PySCF, dan Avogadro memperluas kemungkinan praktikum sesuai tingkat kesiapan mahasiswa dan fasilitas institusi. Namun, mahasiswa harus memahami bahwa software hanyalah alat. Validitas hasil tetap bergantung pada input, metode, basis set, parameter perhitungan, status konvergensi, dan interpretasi kimia.

Secara pedagogis, Bab 2 membangun kebiasaan kerja yang sangat penting: teliti sebelum menghitung, rapi saat menyimpan data, kritis ketika membaca output, dan sistematis ketika

menulis laporan. Kebiasaan ini menjadi fondasi bagi seluruh praktikum berikutnya. Tanpa pengelolaan file dan data yang baik, praktikum tentang optimasi geometri, Hartree–Fock, DFT, orbital molekul, momen dipol, dan spektrum IR akan mudah mengalami kekacauan data.

Dengan demikian, Bab 2 dapat disimpulkan sebagai bab persiapan teknis dan metodologis. Bab ini memastikan bahwa mahasiswa tidak hanya siap memahami teori, tetapi juga siap bekerja dalam ekosistem komputasi ilmiah. Praktikum kimia kuantum yang baik memerlukan tiga unsur sekaligus: pemahaman konsep, keterampilan perangkat lunak, dan kedisiplinan pengelolaan data.

Rangkuman Bab 2

1. **Lingkungan kerja praktikum** harus disiapkan sebelum perhitungan dilakukan, meliputi komputer, koneksi internet, browser, editor teks, Python, serta akses ke platform atau software kimia komputasi.
2. **ChemCompute** dapat digunakan sebagai platform berbasis web untuk menjalankan perhitungan kimia komputasi tanpa instalasi lokal yang rumit.
3. **Python** digunakan sebagai alat bantu analisis data, pembuatan tabel, visualisasi grafik, perhitungan numerik sederhana, dan dokumentasi hasil praktikum.
4. **Software kimia komputasi** seperti GAMESS, ORCA, Gaussian, Psi4, PySCF, dan Avogadro memiliki fungsi berbeda dalam workflow praktikum, mulai dari pembuatan struktur molekul, perhitungan struktur elektronik, sampai analisis hasil.
5. **Struktur folder kerja** harus dibuat secara konsisten agar file input, output, data, gambar, kode, dan laporan tidak tercampur.
6. **Format file molekul** seperti XYZ, MOL, SDF, PDB, dan file input spesifik software harus dipahami karena format yang salah dapat menyebabkan perhitungan gagal atau hasil tidak valid.
7. **Nama file harus informatif**, misalnya menunjukkan nama molekul, metode, basis set, jenis job, atau nomor praktikum.
8. **File output harus selalu disimpan** agar hasil perhitungan dapat diperiksa ulang dan dijadikan dasar laporan.
9. **Reprodusibilitas** merupakan prinsip penting dalam praktikum kimia komputasi. Setiap hasil harus dapat ditelusuri dari laporan kembali ke tabel, grafik, kode, data input, dan output asli.
10. **Kedisiplinan teknis** dalam Bab 2 menjadi fondasi bagi praktikum berikutnya yang lebih konseptual dan analitis.

Implikasi Pembelajaran Bab 2

Bab 2 memiliki beberapa implikasi penting bagi pembelajaran kimia kuantum berbasis komputasi.

Pertama, mahasiswa belajar bahwa keterampilan teknis adalah bagian dari kompetensi ilmiah. Penguasaan teori kimia kuantum perlu didukung kemampuan mengoperasikan perangkat lunak, mengelola file, membaca output, dan menyusun data. Tanpa keterampilan teknis, teori sulit diterapkan dalam praktikum.

Kedua, mahasiswa memahami bahwa komputasi ilmiah menuntut keteraturan. Struktur folder, nama file, format data, dan dokumentasi kode bukan hal tambahan, tetapi bagian dari proses penelitian. Praktikum yang tertib akan memudahkan dosen memeriksa hasil, memudahkan mahasiswa memperbaiki kesalahan, dan memudahkan pembaca menelusuri kesimpulan.

Ketiga, mahasiswa mulai membangun literasi data. Mereka belajar bahwa output software perlu diolah, bukan hanya disalin. Data energi, panjang ikatan, sudut ikatan, momen dipol, atau frekuensi vibrasi perlu disusun dalam tabel, dianalisis, divisualisasikan, dan ditafsirkan.

Keempat, mahasiswa belajar bahwa software tidak menggantikan pemikiran ilmiah. Software dapat menghasilkan output, tetapi mahasiswa tetap bertanggung jawab memeriksa input, memilih metode, menentukan basis set, membaca status konvergensi, dan menafsirkan hasil.

Kelima, Bab 2 membentuk kebiasaan reprodusibilitas sejak awal. Kebiasaan ini akan sangat penting ketika mahasiswa masuk ke praktikum yang lebih kompleks, terutama ketika jumlah file, molekul, metode, basis set, dan grafik semakin banyak.

Arah Menuju Bab 3

Setelah Bab 2 menyiapkan lingkungan kerja, struktur file, format data, dan prinsip reprodusibilitas, Bab 3 dapat diarahkan pada pengenalan struktur molekul dan representasi koordinat. Mahasiswa perlu memahami bagaimana molekul dibangun dalam bentuk tiga dimensi, bagaimana koordinat atom ditulis, bagaimana struktur awal dibuat, serta bagaimana file XYZ atau format molekul lain digunakan sebagai input perhitungan.

Dengan demikian, Bab 2 menjadi jembatan antara orientasi umum pada Bab 1 dan praktik teknis pada bab-bab berikutnya. Mahasiswa yang telah memahami Bab 2 akan lebih siap memasuki tahap pembangunan struktur molekul, optimasi geometri, perhitungan energi, dan analisis sifat elektronik.

Daftar Pustaka Bab 2

Allouche, A. R. (2011). Gabedit—A graphical user interface for computational chemistry softwares. *Journal of Computational Chemistry, 32*(1), 174–182.

Cramer, C. J. (2004). *Essentials of Computational Chemistry: Theories and Models* (2nd ed.). John Wiley & Sons.

Foresman, J. B., & Frisch, Æ. (2015). *Exploring Chemistry with Electronic Structure Methods* (3rd ed.). Gaussian, Inc.

Gordon, M. S., & Schmidt, M. W. (2005). Advances in electronic structure theory: GAMESS a decade later. In C. E. Dykstra, G. Frenking, K. S. Kim, & G. E. Scuseria (Eds.), *Theory and Applications of Computational Chemistry: The First Forty Years* (pp. 1167–1189). Elsevier.

Hanwell, M. D., Curtis, D. E., Lonie, D. C., Vandermeersch, T., Zurek, E., & Hutchison, G. R. (2012). Avogadro: An advanced semantic chemical editor, visualization, and analysis platform. *Journal of Cheminformatics, 4*, 17.

Helgaker, T., Jørgensen, P., & Olsen, J. (2000). *Molecular Electronic-Structure Theory*. John Wiley & Sons.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering, 9*(3), 90–95.

Jensen, F. (2017). *Introduction to Computational Chemistry* (3rd ed.). John Wiley & Sons.

Leach, A. R. (2001). *Molecular Modelling: Principles and Applications* (2nd ed.). Prentice Hall.

Lewars, E. G. (2016). *Computational Chemistry: Introduction to the Theory and Applications of Molecular and Quantum Mechanics* (3rd ed.). Springer.

McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61).

Murray-Rust, P., & Rzepa, H. S. (1999). Chemical Markup, XML, and the World Wide Web. 1. Basic principles. *Journal of Chemical Information and Computer Sciences, 39*(6), 928–942.

Neese, F. (2012). The ORCA program system. *WIREs Computational Molecular Science, 2*(1), 73–78.

Neese, F. (2022). Software update: The ORCA program system—Version 5.0. *WIREs Computational Molecular Science, 12*(5), e1606.

Ochterski, J. W. (2000). *Thermochemistry in Gaussian*. Gaussian, Inc.

Oliphant, T. E. (2006). *A Guide to NumPy*. Trelgol Publishing.

Python Software Foundation. (2024). *Python 3 Documentation*. Python Software Foundation.

Roothaan, C. C. J. (1951). New developments in molecular orbital theory. *Reviews of Modern Physics, 23*(2), 69–89.

Schmidt, M. W., Baldridge, K. K., Boatz, J. A., Elbert, S. T., Gordon, M. S., Jensen, J. H., Koseki, S., Matsunaga, N., Nguyen, K. A., Su, S. J., Windus, T. L., Dupuis, M., & Montgomery, J. A. (1993). General atomic and molecular electronic structure system. *Journal of Computational Chemistry, 14*(11), 1347–1363.

Sun, Q., Berkelbach, T. C., Blunt, N. S., Booth, G. H., Guo, S., Li, Z., Liu, J., McClain, J. D., Sayfutyarova, E. R., Sharma, S., Wouters, S., & Chan, G. K.-L. (2018). PySCF: The Python-based simulations of chemistry framework. *WIREs Computational Molecular Science, 8*(1), e1340.

Szabo, A., & Ostlund, N. S. (1996). *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Dover Publications.

Van Rossum, G., & Drake, F. L. Jr. (2009). *Python 3 Reference Manual*. CreateSpace.

Walt, S. van der, Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: A structure for efficient numerical computation. *Computing in Science & Engineering, 13*(2), 22–30.

Young, D. C. (2001). *Computational Chemistry: A Practical Guide for Applying Techniques to Real-World Problems*. John Wiley & Sons.

BAB 3

DASAR PYTHON UNTUK PRAKTIKUM KIMIA KUANTUM

Pengantar Bab 3

Bab 3 membahas dasar Python sebagai fondasi teknis untuk menganalisis data praktikum kimia kuantum. Jika Bab 2 telah memperkenalkan ekosistem software, ChemCompute, Avogadro, dan alur kerja praktikum komputasi, maka Bab 3 mulai menyiapkan mahasiswa agar mampu menggunakan Python secara sistematis. Dalam buku ini, Python tidak ditempatkan sebagai pelengkap kecil, tetapi sebagai instrumen ilmiah yang memungkinkan mahasiswa membaca data, menghitung parameter, membuat grafik, menguji ulang hasil, menyusun tabel, dan membangun laporan yang reprodusibel.

Python diperlukan karena output kimia komputasi sering berupa file teks panjang yang tidak langsung siap menjadi kesimpulan. Dari satu file output, mahasiswa mungkin harus mengambil energi total, koordinat akhir, panjang ikatan, sudut ikatan, frekuensi vibrasi, intensitas IR, momen dipol, energi HOMO-LUMO, atau data termodinamika. Jika semua data diambil secara manual, kesalahan salin mudah terjadi. Dengan Python, proses analisis dapat dibuat eksplisit: data dibaca, dihitung, dikonversi satuan, disusun dalam tabel, divisualisasikan, lalu digunakan dalam laporan. Inilah alasan Bab 3 menjadi jembatan penting antara penggunaan software kimia komputasi dan penyusunan laporan praktikum ilmiah.

Python juga penting karena praktikum kimia kuantum tidak hanya menuntut hasil, tetapi juga proses berpikir. Mahasiswa perlu memahami bagaimana energi partikel dalam kotak dihitung, bagaimana fungsi gelombang divisualisasikan, bagaimana jarak ikatan dihitung dari koordinat XYZ, bagaimana gap HOMO-LUMO diperoleh dari dua nilai energi orbital, bagaimana spektrum IR simulasi dibuat dari frekuensi dan intensitas, serta bagaimana ΔE atau ΔG reaksi dihitung dari energi reaktan dan produk. Dengan kata lain, Python membantu mahasiswa memahami hubungan antara rumus, data, dan interpretasi kimia.

Bab ini terdiri atas lima subbab. Subbab 3.1 membahas instalasi Python, Anaconda, Jupyter Notebook, Google Colab, dan VS Code. Subbab 3.2 membahas paket Python utama yang digunakan dalam praktikum. Subbab 3.3 membahas struktur kode praktikum. Subbab 3.4 membahas standar penulisan kode. Subbab 3.5 menyediakan template kode umum yang akan digunakan ulang pada bab-bab praktikum berikutnya.

3.1 Instalasi Python dan Jupyter Notebook

Instalasi Python dan Jupyter Notebook adalah tahap awal untuk membangun lingkungan kerja praktikum kimia kuantum berbasis programming ilmiah. Dalam konteks buku ini, instalasi tidak boleh dipahami sekadar sebagai memasang aplikasi di komputer. Instalasi harus dipahami sebagai

proses membangun **lingkungan komputasi ilmiah** yang stabil, dapat dijalankan ulang, mudah diperiksa, dan sesuai dengan kebutuhan praktikum. Lingkungan tersebut mencakup interpreter Python, pengelola paket, notebook interaktif, editor kode, serta mekanisme penyimpanan file praktikum secara rapi.

Python adalah bahasa pemrograman tingkat tinggi yang banyak digunakan dalam sains, teknik, pendidikan, analisis data, kecerdasan buatan, dan komputasi ilmiah. Dalam praktikum kimia kuantum, Python digunakan bukan untuk menggantikan sepenuhnya software kimia komputasi seperti GAMESS, ORCA, Gaussian, Psi4, atau PySCF, tetapi untuk mendukung analisis dan pemahaman. Python membantu mahasiswa membuat simulasi sederhana, membaca data output, menghitung parameter turunan, mengonversi satuan, membuat grafik, dan menyusun laporan berbasis data. Situs resmi Python menyediakan installer dan dokumentasi resmi untuk berbagai sistem operasi, termasuk Windows, macOS, dan Linux. (Python.org)

Jupyter Notebook adalah lingkungan kerja interaktif yang memungkinkan mahasiswa menggabungkan teks naratif, persamaan, kode Python, output numerik, tabel, dan grafik dalam satu dokumen. Jupyter sangat cocok untuk praktikum karena mahasiswa dapat menulis tujuan praktikum, menjalankan kode, menampilkan hasil, membuat grafik, dan memberi interpretasi pada tempat yang sama. Situs resmi Jupyter menjelaskan bahwa Jupyter Notebook merupakan aplikasi web asli untuk membuat dan membagikan dokumen komputasional, sedangkan dokumentasi Jupyter menempatkannya dalam ekosistem besar Project Jupyter untuk komputasi interaktif. (jupyter.org)

Dalam praktikum kimia kuantum, instalasi Python dan Jupyter harus diarahkan pada tiga tujuan. Pertama, mahasiswa dapat menjalankan kode praktikum secara mandiri. Kedua, mahasiswa dapat mengulang analisis dari data yang sama dan memperoleh hasil yang sama. Ketiga, mahasiswa dapat menyimpan kode, data, grafik, dan laporan secara tertib. Jika instalasi hanya berhasil menjalankan Python sekali, tetapi tidak jelas lingkungan mana yang digunakan, paket apa yang terpasang, atau di mana file disimpan, maka instalasi tersebut belum memenuhi standar praktikum komputasi yang baik.

Secara historis, penggunaan Python dan notebook interaktif dalam pendidikan sains berkembang karena kebutuhan akan alat yang mudah dibaca manusia, mudah dipelajari, dan kuat untuk analisis numerik. Sebelum notebook populer, banyak analisis ilmiah dilakukan melalui script terpisah, spreadsheet, atau software tertutup. Model notebook mengubah cara mahasiswa belajar komputasi karena kode, teks, rumus, dan grafik berada dalam satu dokumen. Ini memudahkan dosen memeriksa proses berpikir mahasiswa, bukan hanya jawaban akhir. Namun, notebook juga membawa tantangan: jika lingkungan paket tidak dicatat, notebook dapat sulit dijalankan ulang di komputer lain. Kajian tentang notebook menunjukkan bahwa banyak notebook yang dipublikasikan tidak menyatakan dependensi paket secara lengkap, sehingga menghambat reproduksibilitas hasil komputasi. (arXiv)

Oleh karena itu, instalasi Python dalam buku ini harus selalu dikaitkan dengan reprodusibilitas. Mahasiswa tidak hanya diminta “menginstal Python”, tetapi juga memahami versi Python, lokasi lingkungan kerja, paket yang dipasang, folder praktikum, serta cara menjalankan ulang kode. Kode praktikum yang baik harus dapat dijalankan ulang oleh mahasiswa lain atau dosen dengan data

yang sama. Prinsip ini sejalan dengan Bab 1 dan Bab 2: setiap angka dalam laporan harus memiliki jejak, dan setiap grafik harus dapat ditelusuri ke data serta kode.

1. Jalur Instalasi Python: Python Resmi, Anaconda, Google Colab, dan VS Code

Dalam buku ini, ada empat jalur utama yang dapat digunakan mahasiswa untuk bekerja dengan Python:

Jalur	Fungsi Utama	Cocok untuk
Python resmi dari python.org	Instalasi Python dasar dan ringan	Mahasiswa yang ingin lingkungan minimal
Anaconda	Distribusi Python dengan pengelola lingkungan dan paket ilmiah	Praktikum data, notebook, dan komputasi ilmiah
Google Colab	Notebook berbasis cloud tanpa instalasi lokal	Mahasiswa yang perangkatnya terbatas
VS Code	Editor kode dan notebook untuk workflow lebih profesional	Mahasiswa yang ingin menulis script dan notebook secara tertib

Keempat jalur tersebut tidak harus digunakan semuanya secara bersamaan. Mahasiswa dapat memilih sesuai kondisi perangkat, koneksi internet, dan kebutuhan praktikum. Untuk pemula, Anaconda atau Google Colab sering lebih mudah. Untuk mahasiswa yang ingin bekerja lebih profesional, kombinasi Python resmi, virtual environment, VS Code, dan Jupyter dapat digunakan. Untuk kampus dengan fasilitas komputer terbatas, Google Colab dapat menjadi jalur cepat karena tidak memerlukan instalasi lokal. Google menyebut Colab sebagai layanan Jupyter Notebook ter-hosting yang tidak memerlukan setup dan menyediakan runtime siap pakai untuk memulai dengan cepat. (developers.google.com)

2. Instalasi Python Resmi

Instalasi Python resmi dilakukan melalui situs Python. Jalur ini cocok untuk mahasiswa yang ingin memasang Python secara langsung tanpa distribusi besar seperti Anaconda. Pada Windows, mahasiswa dapat mengunduh installer Python, menjalankannya, lalu memastikan opsi penambahan Python ke PATH diaktifkan agar perintah `python` dan `pip` dapat dikenali dari terminal. Dokumentasi resmi Python untuk Windows menjelaskan penggunaan Python pada Windows dan menyediakan informasi tentang installer Python dari python.org maupun Microsoft Store. (Python documentation)

Setelah instalasi, mahasiswa perlu memeriksa apakah Python sudah terpasang dengan benar. Pemeriksaan dilakukan melalui Command Prompt, PowerShell, Terminal, atau shell sejenis:

```
python --version
pip --version
```

Jika perintah tersebut menampilkan versi Python dan pip, instalasi dasar berhasil. Pip adalah pengelola paket bawaan Python yang digunakan untuk memasang pustaka seperti numpy, scipy, matplotlib, pandas, sympy, atau jupyter. Pada tahap praktikum, mahasiswa tidak cukup hanya

melihat bahwa Python terpasang; mereka juga harus memastikan bahwa pip dapat bekerja karena sebagian besar paket praktikum dipasang melalui pip atau conda.

Untuk memasang Jupyter melalui jalur Python resmi, mahasiswa dapat menggunakan perintah:

```
pip install notebook
```

atau untuk JupyterLab:

```
pip install jupyterlab
```

Project Jupyter menyatakan bahwa alat-alat Jupyter tersedia melalui Python Package Index, sehingga dapat dipasang menggunakan ekosistem paket Python. (jupyter.org) Setelah instalasi, Jupyter Notebook dapat dijalankan dengan:

```
jupyter notebook
```

atau JupyterLab dengan:

```
jupyter lab
```

Kelebihan jalur Python resmi adalah ringan dan fleksibel. Mahasiswa dapat memasang hanya paket yang dibutuhkan. Kekurangannya, pemula sering bingung ketika menghadapi PATH, terminal, virtual environment, atau konflik paket. Karena itu, jika kelas terdiri atas mahasiswa dengan kemampuan teknis beragam, dosen dapat merekomendasikan Anaconda atau Google Colab sebagai jalur awal.

3. Instalasi Anaconda

Anaconda adalah distribusi Python yang populer untuk data science dan komputasi ilmiah. Anaconda menyediakan Python, conda sebagai pengelola paket dan lingkungan, serta berbagai paket ilmiah. Halaman unduhan Anaconda menjelaskan bahwa Anaconda menyediakan sistem manajemen paket dan environment open-source yang berjalan pada Windows, macOS, dan Linux, serta mendukung instalasi, pembaruan paket, dan dependensinya. (Anaconda) Dokumentasi instalasi Anaconda juga menyediakan panduan resmi pemasangan Anaconda Distribution. (Anaconda)

Dalam praktikum kimia kuantum, Anaconda berguna karena memudahkan pembuatan lingkungan terpisah. Lingkungan terpisah penting agar paket untuk satu praktikum tidak mengganggu paket lain. Misalnya, mahasiswa dapat membuat environment khusus bernama `kimia-kuantum`:

```
conda create -n kimia-kuantum python=3.11
conda activate kimia-kuantum
```

Setelah environment aktif, mahasiswa dapat memasang paket dasar:

```
conda install numpy scipy matplotlib pandas sympy jupyter
```

atau menggunakan pip jika paket tertentu tidak tersedia melalui conda:

```
pip install periodictable
```

Konsep environment sangat penting. Tanpa environment, mahasiswa dapat memasang banyak paket di sistem utama sehingga rawan konflik. Dengan environment, setiap proyek praktikum dapat memiliki lingkungan yang lebih terkontrol. Dalam laporan atau README, mahasiswa dapat menulis environment yang digunakan, misalnya:

```
Environment: kimia-kuantum
Python: 3.11
Paket utama: numpy, scipy, matplotlib, pandas, sympy, jupyter
```

Kelebihan Anaconda adalah kemudahan bagi pemula dan kelengkapan ekosistem ilmiah. Kekurangannya, ukuran instalasi relatif besar. Pada komputer dengan ruang penyimpanan terbatas, instalasi Anaconda mungkin terasa berat. Selain itu, mahasiswa harus memahami perbedaan antara `conda` dan `pip` agar tidak memasang paket secara sembarangan. Dosen sebaiknya memberikan panduan sederhana: gunakan conda untuk paket utama jika tersedia, gunakan pip untuk paket khusus, dan selalu aktifkan environment sebelum menjalankan praktikum.

4. Jupyter Notebook sebagai Buku Kerja Praktikum

Jupyter Notebook adalah salah satu alat paling sesuai untuk praktikum kimia kuantum karena memungkinkan mahasiswa menggabungkan penjelasan, kode, output, dan grafik. Dalam satu notebook, mahasiswa dapat menulis judul praktikum, tujuan, teori singkat, data input, kode Python, hasil perhitungan, grafik, dan kesimpulan. Ini berbeda dari script `.py` biasa yang hanya memuat kode. Notebook lebih komunikatif untuk pembelajaran karena proses berpikir dapat ditulis berurutan.

Dalam praktikum partikel dalam kotak, misalnya, notebook dapat memuat persamaan energi:

$$E_n = n^2h^2 / 8mL^2$$

Kemudian kode Python menghitung energi untuk beberapa nilai n, lalu grafik fungsi gelombang ditampilkan. Mahasiswa dapat menulis interpretasi di bawah grafik. Dengan demikian, notebook menjadi laporan komputasional sementara yang menghubungkan teori dan hasil.

Dalam praktikum analisis output ChemCompute, notebook dapat digunakan untuk membaca file CSV hasil ekstraksi energi, membuat tabel, menghitung selisih energi, dan menggambar diagram. Jika notebook disimpan bersama data, dosen dapat menjalankan ulang sel kode untuk memeriksa hasil. Namun, mahasiswa harus berhati-hati: notebook dapat menyimpan output lama yang tidak lagi sesuai dengan kode terbaru. Oleh karena itu, sebelum dikumpulkan, notebook sebaiknya dijalankan ulang dari awal sampai akhir melalui menu **Restart Kernel and Run All** atau fitur sejenis, agar hasil yang tampil benar-benar berasal dari kode terbaru.

Kelebihan Jupyter Notebook adalah interaktif, mudah digunakan, dan sangat baik untuk pembelajaran bertahap. Kekurangannya, notebook dapat menjadi tidak rapi jika mahasiswa

mengeksekusi sel secara acak, mengubah variabel di tengah, atau tidak menyusun kode secara berurutan. Karena itu, sejak awal mahasiswa harus dilatih menulis notebook secara teratur: judul, tujuan, import pustaka, definisi konstanta, input data, fungsi, perhitungan, grafik, dan kesimpulan. Struktur kode ini akan dibahas lebih rinci pada subbab 3.3.

5. Google Colab sebagai Alternatif Tanpa Instalasi Lokal

Google Colab atau Colaboratory adalah lingkungan notebook berbasis cloud yang memungkinkan pengguna menulis dan menjalankan Python melalui browser. Colab sangat berguna bagi mahasiswa yang tidak memiliki laptop kuat, tidak ingin melakukan instalasi lokal, atau mengalami kendala konfigurasi. Google menjelaskan bahwa Colab memungkinkan pengguna menulis dan mengeksekusi Python di browser, sedangkan FAQ Colab menyebutnya sebagai layanan Jupyter Notebook ter-hosting yang tidak memerlukan setup dan menyediakan akses ke sumber daya komputasi, termasuk GPU dan TPU, meskipun pengguna praktikum kimia kuantum dasar tidak selalu memerlukan GPU. (Google Colab)

Dalam buku praktikum ini, Google Colab dapat digunakan untuk beberapa jenis kegiatan:

1. Menjalankan kode partikel dalam kotak.
2. Membuat grafik fungsi gelombang dan probabilitas.
3. Membaca file CSV hasil ChemCompute.
4. Menghitung panjang ikatan dari koordinat XYZ.
5. Membuat diagram HOMO-LUMO.
6. Membuat spektrum IR simulasi.
7. Menghitung energi reaksi dari tabel energi.
8. Membagikan notebook praktikum kepada mahasiswa.

Keunggulan utama Colab adalah kemudahan akses. Mahasiswa cukup memiliki akun Google dan koneksi internet. Notebook dapat disimpan di Google Drive, dibagikan kepada dosen, dan dijalankan tanpa instalasi lokal. Ini sangat membantu kelas besar atau pembelajaran daring. Namun, Colab juga memiliki keterbatasan. Runtime dapat terputus, file sementara dapat hilang jika tidak disimpan, paket tertentu mungkin perlu dipasang ulang setiap sesi, dan akses internet menjadi syarat utama. Oleh karena itu, mahasiswa harus menyimpan file data di Google Drive atau mengunduhnya secara lokal setelah praktikum selesai.

Dalam praktikum kimia kuantum, Colab sebaiknya digunakan sebagai jalur analisis, bukan sebagai satu-satunya tempat penyimpanan data. File input, output, data, dan laporan tetap harus diarsipkan sesuai standar Bab 2. Jika mahasiswa membuat grafik di Colab, notebook dan data sumbernya harus disimpan. Jika notebook membaca file dari Google Drive, jalur file harus ditulis jelas. Jika menggunakan paket tambahan, perintah instalasinya harus disertakan di awal notebook, misalnya:

```
!pip install periodictable
```

Namun, penggunaan tanda seru `!` adalah perintah shell dalam notebook, bukan sintaks Python murni. Mahasiswa perlu memahami perbedaan ini agar tidak bingung ketika menjalankan kode yang sama di VS Code atau terminal.

6. VS Code sebagai Editor Kode dan Notebook

Visual Studio Code atau VS Code adalah editor kode yang banyak digunakan untuk pengembangan Python. Dalam konteks praktikum kimia kuantum, VS Code berguna ketika mahasiswa mulai bekerja dengan file `.py`, struktur folder, parsing output, dan proyek yang lebih besar. Dokumentasi VS Code menyatakan bahwa dukungan Python dapat diaktifkan melalui ekstensi Python, sedangkan dukungan Jupyter Notebook dapat digunakan melalui ekstensi Jupyter; VS Code juga mendukung notebook `.ipynb` secara native melalui ekstensi Jupyter. (code.visualstudio.com)

VS Code memiliki keunggulan dibanding Jupyter Notebook murni ketika proyek semakin kompleks. Mahasiswa dapat melihat struktur folder, membuka file input, output, CSV, script Python, dan laporan dalam satu editor. VS Code juga mendukung terminal terintegrasi, pemilihan interpreter Python, debugging, linting, dan pengelolaan environment. Dokumentasi VS Code untuk Python menjelaskan bahwa dengan ekstensi Python, VS Code dapat digunakan untuk membuat, menjalankan, dan melakukan debug aplikasi Python, bekerja dengan virtual environment, serta menggunakan paket. (code.visualstudio.com)

Dalam praktikum dasar, VS Code dapat digunakan untuk tiga jenis file:

```
.py       → script Python
.ipynb    → notebook Jupyter
.csv      → data tabel
.xyz      → koordinat molekul
.out/log → output software komputasi
```

Dengan satu editor, mahasiswa dapat membuka `h2o_final.xyz`, menjalankan `analisis_geometri.py`, melihat `h2o_summary.csv`, dan menyusun grafik. Ini mendukung alur kerja profesional seperti yang dibahas pada Bab 2. Namun, VS Code memerlukan pemahaman interpreter. Mahasiswa harus memastikan bahwa VS Code menggunakan environment yang benar. Jika notebook atau script gagal menemukan numpy, pandas, atau matplotlib, kemungkinan besar interpreter yang dipilih belum sesuai. VS Code menyediakan perintah **Python: Select Interpreter** untuk memilih environment yang digunakan.

7. Virtual Environment dan Manajemen Paket

Salah satu konsep terpenting dalam instalasi Python adalah **virtual environment** atau lingkungan virtual. Lingkungan virtual adalah ruang terpisah untuk memasang paket Python sehingga tidak bercampur dengan sistem utama. Dalam praktikum kimia kuantum, lingkungan virtual membantu menjaga kestabilan. Misalnya, satu environment digunakan untuk praktikum dasar dengan numpy, scipy, matplotlib, pandas, dan sympy. Environment lain dapat digunakan untuk paket lebih khusus seperti ase, rdkit, psi4, atau pyscf jika tersedia.

Dengan Python resmi, environment dapat dibuat menggunakan `venv`:

```
python -m venv kimia-kuantum
```

Pada Windows, environment diaktifkan dengan:

```
kimia-kuantum\Scripts\activate
```

Pada macOS atau Linux:

```
source kimia-kuantum/bin/activate
```

Setelah environment aktif, paket dapat dipasang:

```
pip install numpy scipy matplotlib pandas sympy jupyter
```

Dengan Anaconda, environment dibuat dengan conda:

```
conda create -n kimia-kuantum python=3.11
conda activate kimia-kuantum
conda install numpy scipy matplotlib pandas sympy jupyter
```

Mahasiswa perlu memahami bahwa environment yang berbeda dapat memiliki paket berbeda. Jika kode berjalan di satu komputer tetapi gagal di komputer lain, salah satu penyebabnya adalah perbedaan environment. Oleh karena itu, setiap praktikum sebaiknya menyimpan daftar paket yang digunakan. Cara sederhana adalah membuat file `requirements.txt`:

```
numpy
scipy
matplotlib
pandas
sympy
jupyter
periodictable
```

Paket dapat dipasang ulang dengan:

```
pip install -r requirements.txt
```

Untuk conda, mahasiswa dapat mengekspor environment:

```
conda env export > environment.yml
```

Walaupun ekspor environment mungkin terlalu teknis untuk praktikum awal, prinsipnya perlu diperkenalkan: kode ilmiah yang baik harus menyertakan informasi paket agar dapat dijalankan ulang.

8. Instalasi Minimum untuk Praktikum Kimia Kuantum

Untuk Bab 3 dan bab-bab praktikum berikutnya, instalasi minimum yang disarankan adalah:

```
Python
Jupyter Notebook atau JupyterLab
```

```
numpy
scipy
matplotlib
pandas
sympy
```

Paket tambahan seperti `periodictable`, `ase`, `rdkit`, `psi4`, atau `pyscf` akan dibahas pada subbab 3.2. Tidak semua paket tambahan harus dipasang sejak awal. Paket seperti RDKit, Psi4, dan PySCF dapat memiliki ketergantungan teknis yang lebih kompleks, sehingga penggunaannya bergantung pada kesiapan perangkat dan sistem operasi. Untuk praktikum dasar, numpy, scipy, matplotlib, pandas, dan sympy sudah cukup untuk menjalankan banyak analisis: grafik fungsi gelombang, energi partikel dalam kotak, tabel energi, plot HOMO-LUMO, spektrum IR simulasi, dan energi reaksi.

Mahasiswa dapat menguji instalasi minimum dengan menjalankan kode berikut di Jupyter Notebook, Colab, atau VS Code:

```
import sys
import numpy as np
import pandas as pd
import matplotlib
import scipy
import sympy as sp

print("Python:", sys.version)
print("NumPy:", np.__version__)
print("Pandas:", pd.__version__)
print("Matplotlib:", matplotlib.__version__)
print("SciPy:", scipy.__version__)
print("SymPy:", sp.__version__)
```

Jika kode berjalan tanpa error, lingkungan dasar siap digunakan. Jika muncul pesan seperti `ModuleNotFoundError`, berarti paket yang dimaksud belum terpasang atau interpreter yang digunakan tidak sesuai.

9. Contoh Uji Instalasi: Grafik Fungsi Sederhana

Setelah instalasi, mahasiswa sebaiknya tidak langsung masuk ke perhitungan kimia kuantum yang kompleks. Uji awal dapat dilakukan dengan membuat grafik fungsi sederhana. Misalnya, grafik fungsi gelombang model sinus:

```
import numpy as np
import matplotlib.pyplot as plt

L = 1.0
x = np.linspace(0, L, 500)
n = 1

psi = np.sqrt(2/L) * np.sin(n * np.pi * x / L)
prob = psi**2
```

```
plt.figure()
plt.plot(x, psi, label="ψ(x)")
plt.plot(x, prob, label="ψ²(x)")
plt.xlabel("x")
plt.ylabel("Nilai fungsi")
plt.title("Uji Instalasi: Fungsi Gelombang dan Probabilitas")
plt.legend()
plt.grid(True)
plt.show()
```

Kode ini bukan hanya uji teknis, tetapi juga pengantar konseptual. Jika grafik tampil, berarti numpy dan matplotlib berjalan. Secara kimia kuantum, mahasiswa juga mulai melihat hubungan antara fungsi gelombang dan rapat probabilitas. Dengan demikian, uji instalasi dapat sekaligus menjadi jembatan menuju praktikum partikel dalam kotak.

10. Contoh Uji Instalasi: Membaca Data CSV

Uji kedua adalah membaca data CSV. Misalnya, mahasiswa membuat file `energi.csv`:

```
molekul,energi_hartree
H2O,-74.963
NH3,-55.455
CH4,-40.195
```

Kemudian membaca file tersebut dengan Python:

```
import pandas as pd

data = pd.read_csv("energi.csv")
print(data)
```

Jika tabel tampil, berarti pandas berjalan dan Python dapat membaca file data. Uji ini penting karena pada bab-bab praktikum berikutnya mahasiswa akan banyak membaca tabel energi, frekuensi, dan data HOMO-LUMO.

11. Contoh Uji Instalasi: Konversi Energi

Uji ketiga adalah konversi satuan energi dari Hartree ke kJ/mol. Dalam kimia komputasi, energi output sering muncul dalam Hartree, sedangkan laporan kimia biasanya menggunakan kJ/mol. Kode sederhana:

```
hartree_to_kjmol = 2625.5

delta_e_hartree = -0.0100
delta_e_kjmol = delta_e_hartree * hartree_to_kjmol

print("ΔE =", delta_e_kjmol, "kJ/mol")
```

Uji ini melatih mahasiswa memahami bahwa instalasi Python bukan sekadar untuk menjalankan kode, tetapi untuk membantu perhitungan kimia yang nyata. Jika konversi satuan tidak ditulis jelas, kesalahan energi reaksi dapat sangat besar.

12. Perbandingan Jalur: Mana yang Sebaiknya Dipilih?

Mahasiswa sering bertanya: apakah lebih baik memakai Anaconda, Python resmi, Google Colab, atau VS Code? Jawabannya bergantung pada kebutuhan.

Jika mahasiswa ingin jalan paling cepat tanpa instalasi, gunakan **Google Colab**. Ini cocok untuk pembelajaran daring, komputer terbatas, atau uji kode awal. Kekurangannya, memerlukan internet dan pengelolaan file cloud.

Jika mahasiswa ingin instalasi lokal yang relatif lengkap, gunakan **Anaconda**. Ini cocok untuk kelas praktikum karena paket ilmiah mudah dipasang dan environment dapat dikelola dengan conda. Kekurangannya, ukuran instalasi besar.

Jika mahasiswa ingin lingkungan ringan dan lebih terkontrol, gunakan **Python resmi** dengan `venv`. Ini cocok untuk mahasiswa yang sudah cukup nyaman dengan terminal. Kekurangannya, pemula dapat mengalami kendala PATH dan paket.

Jika mahasiswa ingin bekerja seperti proyek komputasi profesional, gunakan **VS Code** bersama environment Python. Ini cocok untuk script `.py`, folder praktikum, parsing output, dan pengembangan kode yang lebih rapi. Kekurangannya, perlu memahami interpreter dan ekstensi.

Dalam buku ini, jalur yang paling aman untuk kelas pemula adalah **Anaconda atau Google Colab**. Jalur yang paling kuat untuk mahasiswa lanjut adalah **Python resmi atau Anaconda + VS Code + Jupyter**. Dosen dapat memberi kebebasan, tetapi standar akhir tetap sama: kode harus berjalan, data harus tersimpan, dan hasil harus dapat direproduksi.

13. Kesalahan Umum Instalasi dan Cara Memahaminya

Beberapa kesalahan umum sering muncul dalam instalasi Python.

Pertama, `python` tidak dikenali di terminal. Ini biasanya terjadi karena Python belum masuk PATH atau terminal belum dibuka ulang setelah instalasi. Solusinya adalah memeriksa instalasi, menggunakan Python Launcher pada Windows, atau menambahkan PATH sesuai panduan resmi.

Kedua, `pip` memasang paket tetapi Jupyter tidak dapat membacanya. Ini biasanya terjadi karena pip memasang paket di environment berbeda dari kernel Jupyter. Solusinya adalah memastikan environment aktif sebelum memasang paket dan memilih kernel yang benar di Jupyter.

Ketiga, VS Code tidak menemukan paket. Ini biasanya terjadi karena interpreter VS Code belum diarahkan ke environment yang benar. Solusinya adalah menggunakan **Python: Select Interpreter**.

Keempat, notebook Colab kehilangan file setelah runtime terputus. Ini terjadi karena file disimpan di penyimpanan sementara. Solusinya adalah menyimpan data di Google Drive atau mengunduh file setelah selesai.

Kelima, kode berjalan di komputer dosen tetapi tidak berjalan di komputer mahasiswa. Ini biasanya disebabkan perbedaan versi Python, paket, atau file path. Solusinya adalah menggunakan `requirements.txt`, `environment.yml`, dan jalur file relatif.

Kesalahan-kesalahan ini harus dipahami sebagai bagian dari literasi komputasi. Praktikum kimia kuantum modern menuntut mahasiswa tidak hanya memahami molekul, tetapi juga memahami lingkungan digital tempat data molekul diproses.

14. Implikasi Instalasi terhadap Reprodusibilitas Praktikum

Instalasi yang baik berpengaruh langsung terhadap reprodusibilitas. Jika mahasiswa tidak mengetahui environment yang dipakai, sulit mengulang hasil. Jika paket tidak dicatat, dosen sulit menjalankan ulang kode. Jika notebook hanya berjalan di Colab tetapi tidak menyertakan data, laporan tidak lengkap. Jika script bergantung pada file di folder pribadi tanpa jalur yang jelas, kode tidak portabel. Oleh karena itu, setiap praktikum harus menyertakan catatan lingkungan kerja.

Format ringkas catatan lingkungan kerja dapat ditulis sebagai berikut:

```
Lingkungan kerja:
- Sistem operasi: Windows/macOS/Linux/Colab
- Python: 3.x
- Editor: Jupyter Notebook / Google Colab / VS Code
- Paket utama: numpy, scipy, matplotlib, pandas, sympy
- File kode: python/nama_script.py atau notebook/nama_file.ipynb
- File data: data/nama_data.csv
```

Catatan ini sederhana tetapi sangat penting. Ia membuat praktikum dapat diperiksa dan diulang. Dalam konteks penelitian, informasi semacam ini menjadi bagian dari metode komputasi.

15. Implikasi Pedagogis

Instalasi Python dan Jupyter Notebook memiliki implikasi pedagogis yang besar. Pertama, mahasiswa belajar bahwa komputasi ilmiah tidak hanya tentang hasil akhir, tetapi tentang lingkungan kerja yang dapat diulang. Kedua, mahasiswa belajar menggunakan alat modern yang relevan dengan riset kimia saat ini. Ketiga, mahasiswa mulai memahami hubungan antara kode, data, grafik, dan kesimpulan. Keempat, mahasiswa belajar mengelola error teknis secara rasional. Kelima, mahasiswa memperoleh dasar untuk memahami paket-paket Python yang akan digunakan pada subbab berikutnya.

Dalam konteks praktikum kimia kuantum, instalasi yang benar membuat mahasiswa siap masuk ke tahap berikutnya: penggunaan paket Python. Tanpa paket seperti numpy, scipy, matplotlib, pandas, dan sympy, mahasiswa hanya memiliki Python dasar. Python dasar memang penting, tetapi praktikum kimia kuantum memerlukan operasi numerik, grafik, tabel, dan simbolik. Karena

itu, subbab 3.2 akan membahas paket-paket Python utama yang digunakan dalam buku ini, mulai dari numpy sampai kemungkinan penggunaan psi4 atau pyscf jika tersedia.

Pada akhirnya, instalasi Python dan Jupyter Notebook harus dipahami sebagai pembangunan fondasi kerja ilmiah digital. Mahasiswa tidak hanya memasang program, tetapi menyiapkan ruang berpikir komputasional. Ruang ini akan digunakan untuk memodelkan fungsi gelombang, membaca output ChemCompute, menghitung parameter molekul, membuat grafik spektrum, dan menyusun laporan praktikum yang reprodusibel. Dengan fondasi instalasi yang benar, praktikum kimia kuantum akan lebih tertib, transparan, dan bermakna.

3.2 Paket Python yang Digunakan

Paket Python yang digunakan dalam praktikum kimia kuantum adalah himpunan pustaka ilmiah yang membantu mahasiswa melakukan komputasi numerik, manipulasi data, visualisasi grafik, perhitungan simbolik, pengelolaan data unsur, pemrosesan struktur molekul, cheminformatics, serta perhitungan struktur elektronik jika tersedia. Dalam konteks buku ini, paket Python tidak dipahami sebagai tambahan teknis yang berdiri di pinggir praktikum, melainkan sebagai infrastruktur ilmiah yang memungkinkan data kimia kuantum dibaca, dihitung, divisualisasikan, divalidasi, dan dilaporkan secara reprodusibel.

Jika ChemCompute, ORCA, GAMESS, Gaussian, Psi4, atau PySCF berperan menghasilkan data komputasi molekuler, maka paket Python berperan mengubah data tersebut menjadi pengetahuan yang dapat diperiksa. Output kimia komputasi sering berisi ratusan hingga ribuan baris. Di dalamnya terdapat energi total, koordinat akhir, frekuensi vibrasi, momen dipol, energi orbital, muatan atom, dan pesan konvergensi. Tanpa alat analisis, mahasiswa mudah terjebak pada penyalinan manual yang rawan salah. Dengan paket Python yang tepat, mahasiswa dapat membaca data, mengonversi satuan, membuat tabel, membangun grafik, dan menyusun kesimpulan numerik secara lebih teliti.

Dalam buku ini, paket utama yang digunakan adalah **numpy, scipy, matplotlib, pandas, sympy, periodictable, ase, rdkit jika tersedia, serta psi4 atau pyscf jika tersedia**. Setiap paket memiliki fungsi berbeda. NumPy menjadi fondasi array dan komputasi numerik. SciPy menyediakan algoritma ilmiah tingkat lanjut. Matplotlib digunakan untuk visualisasi. Pandas digunakan untuk tabel dan data terstruktur. SymPy digunakan untuk matematika simbolik. Periodictable digunakan untuk data unsur. ASE digunakan untuk struktur atomistik dan manipulasi koordinat. RDKit digunakan untuk cheminformatics jika tersedia. Psi4 atau PySCF digunakan jika praktikum ingin menjalankan perhitungan kimia kuantum langsung dari Python.

Secara historis, perkembangan paket Python ilmiah mencerminkan transformasi besar dalam sains komputasional. Pada masa awal komputasi ilmiah, banyak analisis dilakukan dengan bahasa Fortran, C, atau software khusus. Bahasa-bahasa tersebut kuat, tetapi tidak selalu mudah dipelajari mahasiswa kimia. Python kemudian berkembang sebagai bahasa yang lebih mudah dibaca, tetapi tetap dapat diperkuat dengan pustaka numerik berperforma tinggi. NumPy, SciPy, Matplotlib, pandas, dan ekosistem Jupyter memungkinkan mahasiswa melakukan komputasi ilmiah dengan sintaks yang relatif sederhana. Dalam kimia komputasi modern, Python bukan hanya bahasa bantu; ia telah menjadi bahasa penghubung antara teori, data, visualisasi, dan otomasi workflow.

1. NumPy: Fondasi Komputasi Numerik

NumPy adalah paket paling dasar dalam ekosistem scientific Python. Dokumentasi NumPy menyebutnya sebagai paket fundamental untuk scientific computing dalam Python, menyediakan objek array multidimensi dan berbagai fungsi turunan untuk operasi numerik. (NumPy) Dalam praktikum kimia kuantum, NumPy digunakan untuk menyimpan dan mengolah vektor, matriks, daftar energi, koordinat atom, nilai fungsi gelombang, grid posisi, spektrum, serta data numerik lain.

Peran NumPy sangat penting karena banyak persoalan kimia kuantum dapat direpresentasikan sebagai array. Koordinat molekul adalah array berukuran $N \times 3$, dengan N jumlah atom. Fungsi gelombang pada grid satu dimensi dapat ditulis sebagai array nilai $\psi(x)$. Energi untuk beberapa bilangan kuantum dapat ditulis sebagai array. Data spektrum IR dapat disusun sebagai array frekuensi dan intensitas. Energi HOMO-LUMO beberapa molekul juga dapat ditulis sebagai array. Dengan NumPy, mahasiswa dapat melakukan operasi pada seluruh array sekaligus tanpa harus menulis perulangan yang panjang.

Misalnya, untuk model partikel dalam kotak, mahasiswa dapat membuat grid posisi:

```
import numpy as np

L = 1.0
x = np.linspace(0, L, 500)
n = 1

psi = np.sqrt(2/L) * np.sin(n * np.pi * x / L)
probabilitas = psi**2
```

Kode ini menunjukkan kekuatan NumPy. Mahasiswa tidak perlu menghitung nilai ψ satu per satu. Seluruh nilai pada 500 titik dihitung sekaligus. Secara pedagogis, ini membantu mahasiswa melihat bahwa persamaan matematika dapat diterjemahkan menjadi data numerik dan grafik.

Dalam analisis geometri molekul, NumPy digunakan untuk menghitung jarak antaratom. Jika koordinat atom A dan B ditulis sebagai vektor, jarak dapat dihitung dengan norma vektor:

```
import numpy as np

A = np.array([0.000000, 0.000000, 0.000000])
B = np.array([0.758602, 0.000000, 0.504284])

jarak = np.linalg.norm(B - A)
print(jarak)
```

Contoh ini sangat relevan untuk praktikum optimasi geometri. Setelah ChemCompute atau software lain menghasilkan koordinat akhir, mahasiswa dapat menghitung panjang ikatan secara mandiri. Dengan demikian, mahasiswa tidak hanya menyalin data dari output, tetapi memahami hubungan antara koordinat dan parameter geometri.

Namun, NumPy juga memiliki batas. NumPy kuat untuk operasi numerik, tetapi tidak dirancang khusus untuk kimia. NumPy tidak mengetahui bahwa atom O memiliki valensi tertentu, tidak otomatis mengenali ikatan kimia, dan tidak memahami charge atau multiplicity. Oleh karena itu, NumPy perlu dipadukan dengan paket lain seperti pandas, ASE, atau RDKit jika mahasiswa ingin bekerja dengan data kimia yang lebih kaya. NumPy adalah fondasi numerik, bukan software kimia komputasi lengkap.

2. SciPy: Algoritma Ilmiah untuk Perhitungan Lanjutan

SciPy adalah paket yang memperluas kemampuan NumPy dengan algoritma ilmiah tingkat lanjut. Situs SciPy menyatakan bahwa SciPy memperluas NumPy dengan alat tambahan untuk array computing dan menyediakan struktur data khusus seperti sparse matrices dan k-dimensional trees; dokumentasinya juga menjelaskan bahwa SciPy merupakan kumpulan algoritma matematika dan fungsi kenyamanan yang dibangun di atas NumPy. (SciPy) Dalam praktikum kimia kuantum, SciPy berguna untuk optimasi numerik, integrasi, interpolasi, penyelesaian persamaan, aljabar linear, dan analisis data ilmiah.

Dalam model kimia kuantum sederhana, SciPy dapat digunakan untuk menyelesaikan persoalan nilai eigen, misalnya pendekatan matriks terhadap partikel dalam kotak atau osilator harmonik. SciPy juga dapat digunakan untuk optimasi fungsi energi sederhana, misalnya mencari titik minimum dari fungsi energi model satu dimensi. Dalam analisis spektrum, SciPy dapat membantu melakukan fitting puncak atau interpolasi data. Dalam analisis numerik, SciPy menyediakan alat yang lebih lengkap daripada NumPy.

Contoh sederhana penggunaan SciPy adalah mencari minimum fungsi energi model:

```
import numpy as np
from scipy.optimize import minimize_scalar

def energi_model(r):
    return (r - 0.74)**2 - 1.0

hasil = minimize_scalar(energi_model, bounds=(0.3, 2.0), method="bounded")

print("Jarak minimum:", hasil.x)
print("Energi minimum:", hasil.fun)
```

Kode ini bukan pengganti optimasi geometri kuantum, tetapi membantu mahasiswa memahami ide dasar optimasi: mencari nilai variabel yang membuat energi minimum. Dengan analogi ini, mahasiswa lebih mudah memahami mengapa optimasi geometri dalam software kimia komputasi bekerja secara iteratif.

SciPy juga relevan untuk integrasi probabilitas. Pada model partikel dalam kotak, mahasiswa dapat menghitung probabilitas menemukan partikel dalam interval tertentu dengan integrasi numerik. Misalnya, probabilitas menemukan partikel antara $x = 0$ dan $x = L/2$ dapat dihitung dari integral $\psi^2 dx$. Ini menghubungkan konsep probabilitas kuantum dengan komputasi numerik.

Namun, SciPy harus digunakan secara hati-hati. Algoritma optimasi numerik dapat menemukan minimum lokal, bukan selalu minimum global. Integrasi numerik bergantung pada grid dan metode integrasi. Fitting puncak dapat menghasilkan parameter yang tampak baik tetapi tidak bermakna jika modelnya salah. Oleh karena itu, mahasiswa harus memahami bahwa SciPy menyediakan alat numerik, sedangkan validitas kimia tetap harus ditentukan melalui teori dan interpretasi.

3. Matplotlib: Visualisasi Grafik Ilmiah

Matplotlib adalah paket visualisasi utama dalam Python. Situs resmi Matplotlib menyebutnya sebagai pustaka komprehensif untuk membuat visualisasi statis, animasi, dan interaktif dalam Python. (Matplotlib) Dalam praktikum kimia kuantum, Matplotlib digunakan untuk membuat grafik fungsi gelombang, rapat probabilitas, diagram energi, spektrum IR simulasi, grafik konvergensi energi, grafik hubungan panjang ikatan dengan energi, serta grafik perbandingan gap HOMO-LUMO.

Visualisasi memiliki peran sangat penting dalam kimia kuantum karena banyak konsep bersifat abstrak. Fungsi gelombang tidak langsung dapat dilihat, tetapi dapat diplot. Rapat probabilitas tidak dapat disentuh, tetapi dapat divisualisasikan. Spektrum IR berupa daftar frekuensi dan intensitas akan lebih bermakna jika dibuat menjadi grafik. Diagram energi reaksi membantu mahasiswa memahami perbedaan energi reaktan dan produk. Dengan Matplotlib, mahasiswa dapat mengubah data numerik menjadi representasi visual yang mendukung argumentasi ilmiah.

Contoh sederhana visualisasi fungsi gelombang:

```
import numpy as np
import matplotlib.pyplot as plt

L = 1.0
x = np.linspace(0, L, 500)

plt.figure()
for n in [1, 2, 3]:
    psi = np.sqrt(2/L) * np.sin(n * np.pi * x / L)
    plt.plot(x, psi, label=f"n = {n}")

plt.xlabel("x")
plt.ylabel("ψ(x)")
plt.title("Fungsi Gelombang Partikel dalam Kotak")
plt.legend()
plt.grid(True)
plt.show()
```

Kode ini membantu mahasiswa melihat bahwa semakin besar n, semakin banyak simpul fungsi gelombang. Dengan demikian, grafik bukan sekadar ilustrasi, tetapi alat untuk memahami struktur keadaan kuantum.

Dalam praktikum IR, Matplotlib dapat digunakan untuk membuat spektrum simulasi. Data frekuensi dan intensitas dari output ChemCompute atau ORCA dapat diubah menjadi puncak-

puncak spektrum. Mahasiswa dapat melihat bagaimana daftar angka berubah menjadi spektrum yang menyerupai data eksperimen. Ini sangat penting karena mahasiswa kimia sering lebih akrab dengan spektrum daripada tabel frekuensi mentah.

Matplotlib juga perlu digunakan dengan standar ilmiah. Grafik harus memiliki judul, label sumbu, satuan, legenda bila perlu, dan skala yang tepat. Grafik tanpa satuan dapat menyesatkan. Misalnya, energi harus jelas apakah dalam Hartree, eV, atau kJ/mol. Frekuensi harus jelas apakah dalam cm^{-1}. Panjang ikatan harus jelas apakah dalam Ångström. Dalam laporan praktikum, visualisasi yang baik adalah visualisasi yang dapat dibaca, ditafsirkan, dan ditelusuri ke data sumbernya.

4. Pandas: Tabel dan Data Terstruktur

Pandas adalah paket untuk analisis dan manipulasi data terstruktur. Situs resmi pandas menyebutnya sebagai alat open-source yang cepat, kuat, fleksibel, dan mudah digunakan untuk analisis serta manipulasi data, dibangun di atas bahasa Python. (pandas.pydata.org) Dalam praktikum kimia kuantum, pandas digunakan untuk membaca file CSV, menyusun tabel energi, menggabungkan data beberapa molekul, menghitung kolom baru, memfilter data, dan mengekspor hasil analisis.

Pandas sangat berguna karena laporan praktikum sering membutuhkan tabel. Misalnya, tabel perbandingan energi total beberapa basis set, tabel panjang ikatan beberapa molekul, tabel frekuensi IR utama, tabel energi HOMO-LUMO, atau tabel energi reaksi. Jika tabel dibuat manual di spreadsheet, kesalahan salin dan kesalahan formula mudah terjadi. Dengan pandas, proses penyusunan tabel dapat ditulis dalam kode dan dijalankan ulang.

Contoh penggunaan pandas untuk menghitung gap HOMO-LUMO:

```
import pandas as pd

data = pd.DataFrame({
    "molekul": ["etilena", "butadiena", "benzena"],
    "homo_ev": [-10.5, -8.9, -9.2],
    "lumo_ev": [1.2, 0.3, 0.8]
})

data["gap_ev"] = data["lumo_ev"] - data["homo_ev"]

print(data)
```

Dalam contoh ini, pandas membantu mahasiswa membuat kolom baru secara eksplisit. Jika data diubah, gap akan dihitung ulang. Ini lebih reprodusibel daripada menghitung manual di kalkulator.

Pandas juga dapat membaca file CSV:

```
import pandas as pd

energi = pd.read_csv("data/energi_basis_set.csv")
print(energi)
```

Dengan struktur folder Bab 2, pandas akan banyak digunakan untuk membaca file dalam folder `data/`. Misalnya, `energi_basis_set.csv`, `frekuensi_ir.csv`, `homo_lumo.csv`, atau `energi_reaksi.csv`.

Namun, pandas bukan pengganti pemahaman kimia. Tabel yang rapi tidak otomatis benar. Mahasiswa tetap harus memastikan bahwa data berasal dari output valid, satuan jelas, metode konsisten, dan kolom dihitung dengan rumus yang benar. Pandas membantu mengelola data, tetapi tidak memverifikasi makna kimia secara otomatis.

5. SymPy: Matematika Simbolik untuk Teori Kimia Kuantum

SymPy adalah paket Python untuk matematika simbolik. Situs SymPy menyebutnya sebagai pustaka Python untuk symbolic mathematics yang bertujuan menjadi sistem aljabar komputer lengkap dengan kode yang tetap sederhana dan mudah diperluas. (SymPy) Dalam praktikum kimia kuantum, SymPy berguna untuk menurunkan persamaan, menyederhanakan ekspresi, menghitung turunan, integral, limit, menyelesaikan persamaan, dan bekerja dengan matriks secara simbolik.

Perbedaan NumPy dan SymPy penting. NumPy bekerja secara numerik, sedangkan SymPy bekerja secara simbolik. Jika NumPy menghitung nilai $\sin(\pi/2)$ menjadi angka, SymPy dapat mempertahankan ekspresi matematis dalam bentuk simbolik. Dalam pembelajaran kimia kuantum, SymPy berguna untuk menunjukkan asal-usul rumus. Misalnya, normalisasi fungsi gelombang partikel dalam kotak dapat ditunjukkan secara simbolik.

Contoh penggunaan SymPy untuk integral normalisasi:

```
import sympy as sp

x, L = sp.symbols("x L", positive=True)
n = sp.symbols("n", integer=True, positive=True)

psi = sp.sqrt(2/L) * sp.sin(n * sp.pi * x / L)
integral = sp.integrate(psi**2, (x, 0, L))

print(sp.simplify(integral))
```

Hasilnya menunjukkan bahwa integral ψ^2 dari 0 sampai L bernilai 1. Ini sangat baik untuk pembelajaran karena mahasiswa melihat bahwa syarat normalisasi tidak hanya dinyatakan, tetapi dapat dibuktikan secara simbolik.

SymPy juga dapat digunakan untuk menurunkan fungsi energi sederhana, menghitung turunan pertama dan kedua, atau memperlihatkan hubungan antara titik stasioner dan minimum. Dalam konteks optimasi geometri, SymPy dapat membantu menjelaskan ide bahwa minimum energi terjadi ketika turunan pertama energi terhadap koordinat bernilai nol dan turunan kedua positif.

Namun, SymPy tidak selalu cocok untuk perhitungan numerik besar. Ekspresi simbolik dapat menjadi sangat kompleks dan lambat jika sistem terlalu besar. Oleh karena itu, SymPy paling tepat digunakan untuk teori dan model sederhana, bukan untuk menggantikan paket struktur elektronik.

Dalam buku ini, SymPy digunakan sebagai jembatan antara persamaan matematis dan kode numerik.

6. Periodictable: Data Unsur dan Massa Atom

Periodictable adalah paket Python yang menyediakan data unsur. Dokumentasinya menyatakan bahwa periodictable menyediakan tabel periodik yang dapat diperluas, sudah diisi dengan data penting untuk eksperimen hamburan neutron dan sinar-X; paket ini ditulis sepenuhnya dalam Python dan tidak memerlukan pustaka eksternal. (Tabel Periodik) Dalam praktikum kimia kuantum, periodictable dapat digunakan untuk mengambil massa atom, nomor atom, simbol unsur, dan informasi unsur yang berguna dalam perhitungan sederhana.

Paket ini berguna ketika mahasiswa ingin menghitung massa molekul dari rumus atau daftar atom. Misalnya, dari file XYZ mahasiswa dapat membaca simbol atom, lalu menggunakan periodictable untuk menghitung massa total molekul. Massa juga relevan dalam pembahasan vibrasi, karena frekuensi vibrasi dipengaruhi oleh massa atom. Walaupun software kimia komputasi sudah menghitung massa dan frekuensi secara internal, penggunaan periodictable membantu mahasiswa memahami hubungan antara unsur, massa, dan sifat molekul.

Contoh penggunaan sederhana:

```
import periodictable as pt

massa_O = pt.O.mass
massa_H = pt.H.mass

massa_H2O = massa_O + 2 * massa_H

print("Massa molekul H2O =", massa_H2O)
```

Contoh ini sangat sederhana, tetapi bermanfaat untuk membangun literasi data kimia. Mahasiswa belajar bahwa informasi unsur dapat diambil secara programatik, bukan hanya dilihat di tabel periodik cetak.

Namun, periodictable tidak dirancang sebagai paket cheminformatics lengkap. Ia tidak membaca SMILES, tidak membangun struktur molekul, dan tidak menghitung properti molekul kompleks. Untuk tugas semacam itu, RDKit atau ASE lebih sesuai. Periodictable adalah alat kecil tetapi berguna untuk data unsur dan massa.

7. ASE: Atomic Simulation Environment

ASE atau Atomic Simulation Environment adalah paket Python untuk atomistic simulations. Situs resmi ASE menyatakan bahwa ASE menyediakan platform berbasis Python untuk menyiapkan, memanipulasi, menjalankan, memvisualisasikan, dan menganalisis simulasi atomistik; dokumentasi ASE juga menjelaskan bahwa ASE memiliki objek pusat `Atoms` dan interface ke berbagai kode melalui calculators. (ASE community) Dalam praktikum kimia kuantum, ASE

berguna untuk membaca dan menulis file struktur, memanipulasi koordinat atom, menghitung parameter geometri, serta menghubungkan struktur dengan berbagai kalkulator jika tersedia.

ASE sangat relevan untuk alur Avogadro–ChemCompute–Python. Struktur yang disimpan sebagai XYZ dapat dibaca dengan ASE. Mahasiswa dapat menghitung jarak, sudut, dan torsi menggunakan fungsi ASE. ASE juga dapat menulis file dalam berbagai format. Pada tingkat lanjut, ASE dapat digunakan untuk menjalankan kalkulator tertentu, tetapi dalam buku ini fokus awalnya adalah manipulasi dan analisis struktur.

Contoh penggunaan ASE:

```
from ase.io import read

molekul = read("data/h2o_final.xyz")

jarak_oh1 = molekul.get_distance(0, 1)
jarak_oh2 = molekul.get_distance(0, 2)
sudut_hoh = molekul.get_angle(1, 0, 2)

print(jarak_oh1, jarak_oh2, sudut_hoh)
```

Kode ini jauh lebih mudah daripada menulis sendiri rumus vektor untuk jarak dan sudut. Namun, mahasiswa tetap perlu memahami konsep di baliknya. ASE memudahkan analisis, tetapi tidak boleh membuat mahasiswa lupa bahwa jarak dihitung dari koordinat dan sudut dihitung dari vektor.

ASE juga bermanfaat untuk proyek mini yang melibatkan banyak struktur. Misalnya, mahasiswa dapat membaca beberapa file XYZ, menghitung panjang ikatan utama, menyusun tabel pandas, lalu membuat grafik. Ini sangat berguna untuk praktikum optimasi geometri dan konformer.

Keterbatasan ASE adalah instalasinya kadang memerlukan perhatian lebih daripada paket dasar. Selain itu, ASE lebih kuat untuk atomistic simulations secara umum, sehingga tidak semua fiturnya diperlukan dalam praktikum dasar. Dosen dapat menjadikan ASE sebagai paket tambahan: digunakan jika tersedia, tetapi tidak wajib untuk seluruh mahasiswa pemula.

8. RDKit: Cheminformatics jika Tersedia

RDKit adalah toolkit cheminformatics dan machine learning yang ditulis dalam C++ dan Python. Repositori resmi RDKit menyebutnya sebagai kumpulan software cheminformatics dan machine-learning yang ditulis dalam C++ dan Python. (GitHub) Dalam praktikum kimia kuantum, RDKit berguna untuk membaca SMILES, membangun struktur molekul awal, menghitung descriptor sederhana, menangani dataset molekul, dan melakukan analisis cheminformatics.

RDKit sangat berguna jika praktikum mulai berhubungan dengan banyak molekul. Misalnya, mahasiswa ingin membandingkan gap HOMO-LUMO dari beberapa turunan benzena. Struktur awal dapat berasal dari SMILES, kemudian diubah menjadi molekul 3D awal. RDKit juga dapat menghitung descriptor seperti massa molekul, logP, jumlah donor/akseptor ikatan hidrogen, dan

jumlah rotatable bonds. Walaupun descriptor tersebut bukan perhitungan kimia kuantum, mereka dapat mendukung proyek mini yang menghubungkan kimia komputasi dan kimia informatika.

Contoh konseptual penggunaan RDKit:

```
from rdkit import Chem
from rdkit.Chem import Descriptors

mol = Chem.MolFromSmiles("CCO")
massa = Descriptors.MolWt(mol)

print("Massa molekul etanol =", massa)
```

Namun, RDKit tidak selalu mudah dipasang di semua lingkungan, terutama bagi pemula. Karena itu, dalam silabus buku ini RDKit ditulis "jika tersedia". Artinya, RDKit dapat digunakan pada kelas yang memiliki dukungan instalasi, tetapi praktikum dasar tetap harus dapat berjalan tanpa RDKit. Mahasiswa dapat menggunakan Avogadro untuk membuat struktur jika RDKit tidak tersedia.

Secara kritis, RDKit juga bukan paket struktur elektronik. RDKit tidak menggantikan ChemCompute, ORCA, GAMESS, Gaussian, Psi4, atau PySCF dalam perhitungan energi kuantum. RDKit lebih tepat ditempatkan sebagai alat cheminformatics dan persiapan data molekul. Jika mahasiswa menggunakan RDKit untuk menghasilkan geometri awal, struktur tetap perlu dioptimasi dengan metode kimia komputasi yang sesuai.

9. Psi4: Perhitungan Kimia Kuantum dari Python jika Tersedia

Psi4 adalah software kimia kuantum open-source yang dapat digunakan sebagai executable maupun modul Python. Dokumentasi Psi4 menampilkan tutorial penggunaan Psi4 sebagai executable dan sebagai modul Python, serta memuat konsep input, konstanta fisik, spesifikasi memori, geometri molekul, basis set, variabel hasil, loop, dan tabel hasil. (PsiCode) Artikel Smith dkk. juga menyebut Psi4 sebagai proyek kimia kuantum bebas dan open-source dengan fitur luas serta dukungan paralel multi-core. (PMC)

Dalam praktikum kimia kuantum, Psi4 relevan karena memungkinkan perhitungan struktur elektronik dilakukan langsung dari Python. Mahasiswa dapat menulis molekul, metode, basis set, lalu memanggil fungsi energi atau optimasi. Ini sangat cocok untuk praktikum yang ingin mengotomatisasi banyak perhitungan, misalnya kurva energi potensial H_2 pada berbagai panjang ikatan.

Contoh konseptual penggunaan Psi4:

```
import psi4

psi4.set_memory("1 GB")

molekul = psi4.geometry("""
0 1
H
```

```
H 1 0.74
""")

energi = psi4.energy("hf/sto-3g", molecule=molekul)

print("Energi H2 =", energi)
```

Kode seperti ini menunjukkan integrasi yang kuat antara Python dan kimia kuantum. Mahasiswa dapat membuat loop untuk beberapa panjang ikatan, menyimpan energi, lalu memplot kurva energi. Ini sulit dilakukan jika setiap input harus dibuat manual.

Namun, Psi4 tidak selalu tersedia di semua komputer mahasiswa. Instalasinya dapat lebih kompleks daripada numpy atau matplotlib. Karena itu, dalam buku ini Psi4 digunakan "jika tersedia". Jika tidak tersedia, perhitungan struktur elektronik dapat dilakukan melalui ChemCompute atau software lain, sedangkan Python tetap digunakan untuk analisis output.

10. PySCF: Framework Kimia Kuantum Berbasis Python jika Tersedia

PySCF adalah framework kimia kuantum berbasis Python. Situs resminya menyebut PySCF sebagai quantum chemistry with Python, terutama berbasis Python, mudah dibaca dan ditulis, cepat karena menggunakan NumPy, SciPy, serta kode C/C++ khusus, gratis dan open-source, modular, komprehensif, serta dapat diperluas. (PySCF) Artikel PySCF menjelaskan bahwa PySCF dirancang sebagai platform electronic structure umum yang menekankan kesederhanaan kode dan fleksibilitas workflow, mendukung sistem berhingga maupun periodik, Hamiltonian khusus, serta metode mean-field dan post-mean-field dengan basis Gaussian standar. (arXiv)

Dalam praktikum, PySCF dapat digunakan untuk memperkenalkan mahasiswa pada perhitungan Hartree–Fock dan DFT berbasis Python. Misalnya, mahasiswa dapat menghitung energi H_2O dengan RHF/STO-3G atau DFT sederhana. Keunggulan PySCF adalah keterbukaannya bagi pembelajaran algoritmik. Mahasiswa dapat melihat hubungan antara definisi molekul, basis set, metode SCF, dan energi akhir secara langsung dalam kode Python.

Contoh konseptual PySCF:

```
from pyscf import gto, scf

mol = gto.M(
    atom="""
    O 0.000000 0.000000 0.000000
    H 0.758602 0.000000 0.504284
    H -0.758602 0.000000 0.504284
    """,
    basis="sto-3g",
    charge=0,
    spin=0
)

mf = scf.RHF(mol)
energi = mf.kernel()
```

```
print("Energi RHF H2O =", energi)
```

Kode ini memperlihatkan bahwa PySCF dapat membawa mahasiswa lebih dekat ke struktur algoritmik kimia kuantum. Namun, seperti Psi4, PySCF memerlukan kesiapan instalasi dan pemahaman yang lebih tinggi. Karena itu, PySCF sebaiknya digunakan pada praktikum lanjutan atau proyek mini, bukan sebagai prasyarat awal untuk semua mahasiswa.

Sintesis Fungsi Paket dalam Praktikum

Paket-paket Python dalam buku ini dapat dikelompokkan berdasarkan fungsi pedagogisnya.

Paket	Fungsi Utama	Contoh Penggunaan dalam Praktikum
NumPy	Array dan komputasi numerik	Fungsi gelombang, koordinat, energi, vektor
SciPy	Algoritma ilmiah	Optimasi, integrasi probabilitas, fitting spektrum
Matplotlib	Visualisasi	Grafik ψ, ψ^2, spektrum IR, diagram energi
pandas	Data tabel	CSV energi, tabel HOMO-LUMO, data reaksi
SymPy	Matematika simbolik	Normalisasi, turunan, integral, persamaan
periodictable	Data unsur	Massa atom, massa molekul sederhana
ASE	Struktur atomistik	Membaca XYZ, menghitung jarak dan sudut
RDKit	Cheminformatics	SMILES, descriptor, dataset molekul
Psi4	Quantum chemistry Python	Energi, optimasi, workflow kuantum jika tersedia
PySCF	Quantum chemistry Python	HF, DFT, workflow struktur elektronik jika tersedia

Tabel ini membantu mahasiswa memahami bahwa tidak semua paket digunakan untuk semua praktikum. Pada praktikum awal, NumPy, Matplotlib, pandas, dan SymPy sudah sangat penting. Pada praktikum geometri molekul, ASE mulai berguna. Pada proyek dataset molekul, RDKit berguna jika tersedia. Pada praktikum struktur elektronik berbasis Python, Psi4 atau PySCF dapat digunakan jika lingkungan mendukung.

Urutan Pembelajaran Paket

Urutan pembelajaran paket perlu disusun secara bertahap. Mahasiswa sebaiknya tidak langsung diberi semua paket sekaligus. Urutan yang disarankan adalah:

1. **NumPy** untuk array dan operasi numerik.
2. **Matplotlib** untuk visualisasi grafik.
3. **pandas** untuk tabel data.
4. **SymPy** untuk persamaan simbolik.
5. **SciPy** untuk integrasi, optimasi, dan algoritma ilmiah.
6. **periodictable** untuk data unsur.
7. **ASE** untuk struktur atomistik.
8. **RDKit** jika praktikum masuk ke cheminformatics.
9. **Psi4 atau PySCF** jika praktikum masuk ke perhitungan kuantum berbasis Python.

Urutan ini mengikuti prinsip pedagogis dari sederhana ke kompleks. NumPy dan Matplotlib harus dikuasai lebih dahulu karena hampir semua praktikum membutuhkan array dan grafik. Pandas

diperlukan ketika data mulai berbentuk tabel. SymPy membantu teori. SciPy memperluas numerik. ASE dan RDKit membawa mahasiswa ke dunia struktur molekul dan dataset kimia. Psi4 dan PySCF membawa mahasiswa ke perhitungan struktur elektronik langsung dari Python.

Analisis Kritis: Paket Python Bukan Pengganti Pemahaman Kimia

Walaupun paket Python sangat kuat, mahasiswa harus memahami batasnya. Paket hanya alat. NumPy dapat menghitung array, tetapi tidak mengetahui apakah rumus yang digunakan benar. Pandas dapat membuat tabel, tetapi tidak mengetahui apakah energi berasal dari output konvergen. Matplotlib dapat membuat grafik indah, tetapi tidak mengetahui apakah satuannya salah. SymPy dapat menghitung integral, tetapi tidak memutuskan apakah model fisiknya sesuai. ASE dapat menghitung jarak atom, tetapi tidak otomatis menilai apakah struktur kimia masuk akal. RDKit dapat membaca SMILES, tetapi struktur 3D awal tetap perlu validasi. Psi4 dan PySCF dapat menghitung energi, tetapi hasilnya tetap bergantung pada metode, basis set, charge, multiplicity, dan parameter.

Kesalahan yang sering terjadi adalah mahasiswa menganggap bahwa hasil dari paket Python otomatis benar karena kode berjalan tanpa error. Ini keliru. Kode yang berjalan hanya berarti sintaks dan eksekusi berhasil. Validitas ilmiah membutuhkan pemeriksaan lebih lanjut. Apakah data input benar? Apakah satuan benar? Apakah rumus benar? Apakah metode sesuai? Apakah output asal konvergen? Apakah grafik tidak menyesatkan? Semua pertanyaan ini harus dijawab melalui pemahaman kimia dan reprodusibilitas.

Implikasi Praktis untuk Laporan

Dalam laporan praktikum, paket Python yang digunakan harus disebutkan. Contoh penulisan yang baik:

“Analisis data dilakukan menggunakan Python 3.11 dengan paket NumPy untuk perhitungan numerik, pandas untuk pengelolaan tabel, Matplotlib untuk visualisasi grafik, dan SymPy untuk pemeriksaan simbolik persamaan normalisasi.”

Jika menggunakan ASE, RDKit, Psi4, atau PySCF, sebutkan juga. Misalnya:

“Koordinat XYZ hasil optimasi dibaca menggunakan ASE, kemudian panjang ikatan dan sudut ikatan dihitung dari struktur final.”

Atau:

“Perhitungan energi H_2 pada beberapa panjang ikatan dilakukan menggunakan PySCF dengan metode RHF/STO-3G, kemudian data energi diplot menggunakan Matplotlib.”

Penulisan ini penting karena paket Python adalah bagian dari metode. Jika laporan menyebut software kimia komputasi, ia juga perlu menyebut paket analisis Python yang dipakai.

Implikasi terhadap Reprodusibilitas

Paket Python mendukung reprodusibilitas jika versi dan dependensinya dicatat. Untuk praktikum dasar, pencatatan sederhana sudah cukup. Misalnya:

```
Python 3.11
numpy
scipy
matplotlib
pandas
sympy
periodictable
ase
```

Jika proyek lebih lanjut menggunakan RDKit, Psi4, atau PySCF, pencatatan versi menjadi lebih penting karena paket tersebut lebih kompleks. File `requirements.txt` atau `environment.yml` dapat digunakan untuk menyimpan daftar paket. Dengan cara ini, dosen atau mahasiswa lain dapat memasang lingkungan yang sama dan menjalankan ulang kode.

Reprodusibilitas juga menuntut kode tidak bergantung pada file tersembunyi. Jika script membaca `data/energi.csv`, file tersebut harus tersedia. Jika notebook membaca file output, file output harus disertakan. Jika grafik dibuat dari data, data dan script harus disimpan. Paket Python hanya mendukung reprodusibilitas jika digunakan bersama standar file yang telah dibahas pada Bab 2.

Aplikasi Paket pada Praktikum Buku Ini

Pada praktikum partikel dalam kotak, NumPy digunakan untuk membuat grid x dan menghitung fungsi gelombang, Matplotlib untuk grafik, SymPy untuk normalisasi simbolik, dan SciPy jika diperlukan untuk integrasi numerik.

Pada praktikum optimasi geometri, ChemCompute atau software kimia komputasi menghasilkan koordinat akhir. Python menggunakan NumPy atau ASE untuk menghitung panjang ikatan, sudut ikatan, dan perubahan geometri. Pandas digunakan untuk menyusun tabel hasil.

Pada praktikum Hartree–Fock dan basis set, pandas digunakan untuk menyimpan energi beberapa basis set. Matplotlib digunakan untuk membuat grafik energi. NumPy digunakan untuk menghitung selisih energi.

Pada praktikum DFT dan HOMO-LUMO, pandas digunakan untuk membaca tabel energi orbital. NumPy digunakan untuk menghitung gap. Matplotlib digunakan untuk membuat diagram energi orbital.

Pada praktikum simetri dan momen dipol, NumPy dapat digunakan untuk operasi vektor, ASE untuk membaca koordinat, dan Matplotlib untuk visualisasi sederhana.

Pada praktikum frekuensi IR, pandas digunakan untuk membaca frekuensi dan intensitas, NumPy untuk membangun fungsi puncak, SciPy jika diperlukan untuk fitting, dan Matplotlib untuk membuat spektrum.

Pada praktikum energi reaksi, pandas digunakan untuk tabel energi reaktan dan produk, NumPy untuk perhitungan ΔE, dan Matplotlib untuk diagram energi reaksi.

Sintesis Subbab

Paket Python dalam praktikum kimia kuantum harus dipahami sebagai ekosistem kerja. NumPy menyediakan fondasi numerik. SciPy menyediakan algoritma ilmiah. Matplotlib mengubah data menjadi grafik. Pandas mengelola tabel. SymPy menghubungkan teori dengan matematika simbolik. Periodictable menyediakan data unsur. ASE membantu manipulasi struktur atomistik. RDKit membuka pintu menuju cheminformatics. Psi4 dan PySCF memungkinkan perhitungan struktur elektronik langsung dari Python jika tersedia.

Kekuatan utama ekosistem ini bukan hanya pada kemampuannya menjalankan perhitungan, tetapi pada kemampuannya membuat proses ilmiah menjadi eksplisit. Rumus ditulis dalam kode. Data disimpan dalam tabel. Grafik dibuat dari data yang dapat diperiksa. Kesimpulan ditarik dari hasil yang dapat dijalankan ulang. Dengan demikian, paket Python membantu membentuk mahasiswa menjadi praktikan kimia kuantum yang tidak hanya mampu menggunakan software, tetapi juga mampu berpikir, menganalisis, dan melaporkan hasil secara reprodusibel.

Subbab berikutnya akan membahas struktur kode praktikum. Pembahasan tersebut penting karena paket-paket yang telah dijelaskan pada subbab ini harus digunakan dalam susunan kode yang tertib: import pustaka, definisi konstanta, definisi fungsi, input data, perhitungan, tabel hasil, grafik, ekspor data, dan kesimpulan numerik.

3.3 Struktur Kode Praktikum

Struktur kode praktikum adalah susunan logis bagian-bagian kode Python yang digunakan agar setiap kegiatan komputasi dalam praktikum kimia kuantum dapat dibaca, dijalankan ulang, diperiksa, dimodifikasi, dan dilaporkan secara ilmiah. Dalam konteks buku ini, struktur kode tidak dipahami hanya sebagai urutan teknis penulisan program, tetapi sebagai bentuk tertulis dari cara berpikir ilmiah. Kode yang baik menunjukkan bagaimana data masuk, bagaimana rumus digunakan, bagaimana perhitungan dilakukan, bagaimana hasil disusun, bagaimana grafik dibuat, bagaimana data diekspor, dan bagaimana kesimpulan numerik ditarik.

Struktur kode praktikum sangat penting karena praktikum kimia kuantum tidak boleh berhenti pada aktivitas menyalin angka dari output software. Data hasil ChemCompute, ORCA, GAMESS, Gaussian, Psi4, PySCF, atau program lain harus dianalisis secara eksplisit. Python menjadi alat untuk membuat proses analisis itu tampak, dapat diperiksa, dan dapat diulang. Jupyter Notebook sendiri dirancang sebagai platform komputasi interaktif yang menggabungkan kode langsung, persamaan, teks naratif, dan visualisasi dalam satu dokumen, sehingga sangat sesuai untuk menyusun laporan komputasional praktikum. (jupyter.org)

Dalam buku ini, setiap kode Python praktikum minimal memiliki sembilan bagian utama:

1. **Import pustaka.**
2. **Definisi konstanta.**
3. **Definisi fungsi.**
4. **Input data.**
5. **Perhitungan.**
6. **Tabel hasil.**
7. **Grafik.**
8. **Ekspor data.**
9. **Kesimpulan numerik.**

Urutan ini bukan aturan kaku yang tidak boleh berubah, tetapi standar minimal agar kode praktikum memiliki alur yang jelas. Mahasiswa boleh menambahkan bagian lain, misalnya validasi data, pengecekan error, pembacaan file output, atau dokumentasi metadata. Namun, sembilan bagian tersebut harus tetap tampak karena mewakili tahapan ilmiah: menyiapkan alat, mendefinisikan parameter, merumuskan prosedur, memasukkan data, melakukan perhitungan, menyusun hasil, memvisualisasikan, menyimpan, dan menyimpulkan.

Secara historis, kebutuhan struktur kode muncul dari perkembangan komputasi ilmiah. Pada masa awal, program ilmiah sering ditulis sebagai skrip panjang yang sulit dibaca selain oleh penulisnya sendiri. Dalam pendidikan modern, praktik seperti itu tidak memadai. Kode ilmiah harus dapat dibaca oleh dosen, teman sejawat, dan mahasiswa lain. Prinsip ini selaras dengan tradisi Python yang menekankan keterbacaan. PEP 8, sebagai pedoman gaya resmi Python, menyatakan bahwa pedoman tersebut memberikan konvensi penulisan kode Python untuk pustaka standar; salah satu prinsip umumnya adalah menjaga keterbacaan dan konsistensi kode. (Python Enhancement Proposals (PEPs)) The Zen of Python juga menekankan prinsip seperti “explicit is better than implicit” dan “readability counts”, yang sangat relevan untuk praktikum kimia kuantum karena kode harus menjelaskan proses ilmiah, bukan menyembunyikannya. (Python Enhancement Proposals (PEPs))

1. Import Pustaka

Bagian pertama dalam kode praktikum adalah import pustaka. Import pustaka berarti memanggil paket Python yang diperlukan agar fungsi-fungsi tertentu dapat digunakan. Dalam praktikum kimia kuantum, pustaka yang paling sering digunakan adalah `numpy`, `pandas`, `matplotlib.pyplot`, `scipy`, `sympy`, dan kadang `ase`, `periodictable`, `rdkit`, `psi4`, atau `pyscf` jika tersedia. Bagian import sebaiknya diletakkan di awal kode agar pembaca segera mengetahui paket apa yang digunakan.

Contoh bagian import:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.integrate import simpson
```

Kode tersebut menunjukkan empat pustaka utama. `numpy` digunakan untuk komputasi numerik, `pandas` untuk tabel data, `matplotlib.pyplot` untuk grafik, dan `simpson` dari `scipy.integrate` untuk integrasi numerik. Import seperti ini harus ditulis secara eksplisit. Jangan menyisipkan import di tengah kode tanpa alasan, karena pembaca akan sulit mengetahui dependensi program. Dalam laporan praktikum, pustaka yang digunakan juga perlu disebut pada bagian metode.

Secara pedagogis, bagian import mengajarkan mahasiswa bahwa setiap alat analisis memiliki fungsi. Jika ingin menghitung array, gunakan NumPy. Jika ingin membuat tabel, gunakan pandas. Jika ingin membuat grafik, gunakan Matplotlib. Jika ingin integrasi atau optimasi numerik, gunakan SciPy. Kesadaran ini penting agar mahasiswa tidak memakai paket secara mekanis. Mereka harus memahami mengapa suatu pustaka dipanggil.

Kesalahan umum pada bagian import adalah memanggil terlalu banyak paket yang tidak digunakan. Misalnya, mahasiswa mengimpor `rdkit`, `ase`, `sympy`, dan `scipy` padahal kode hanya menghitung grafik sederhana dengan NumPy dan Matplotlib. Ini membuat kode tampak berat dan membingungkan. Kode praktikum sebaiknya memanggil paket yang diperlukan saja. Kesalahan lain adalah menggunakan nama alias yang tidak umum. Dalam komunitas Python, `numpy as np`, `pandas as pd`, dan `matplotlib.pyplot as plt` sudah menjadi konvensi umum. Mengikuti konvensi membantu kode lebih mudah dibaca.

2. Definisi Konstanta

Bagian kedua adalah definisi konstanta. Konstanta adalah nilai tetap yang digunakan dalam perhitungan, misalnya konstanta Planck, massa elektron, kecepatan cahaya, konstanta Avogadro, faktor konversi Hartree ke kJ/mol, atau panjang kotak dalam model partikel dalam kotak. Dalam praktikum kimia kuantum, konstanta harus ditulis jelas, diberi nama bermakna, dan disertai satuan.

Contoh definisi konstanta:

```
h = 6.62607015e-34        # Konstanta Planck, J s
m_e = 9.1093837015e-31    # Massa elektron, kg
hartree_to_kjmol = 2625.49962  # 1 Hartree = 2625.49962 kJ/mol
```

Penulisan seperti ini jauh lebih baik daripada memasukkan angka langsung ke dalam rumus tanpa penjelasan. Misalnya, menulis `energi * 2625.49962` berulang kali tanpa memberi nama konstanta akan membuat kode sulit dibaca. Dengan memberi nama `hartree_to_kjmol`, pembaca langsung memahami bahwa angka tersebut adalah faktor konversi energi.

Definisi konstanta memiliki nilai epistemik. Ia menunjukkan asumsi satuan yang digunakan. Dalam kimia komputasi, kesalahan satuan merupakan salah satu sumber kesalahan terbesar. Energi output dapat ditulis dalam Hartree, eV, kJ/mol, kcal/mol, atau cm^{-1}. Panjang dapat ditulis dalam Ångström atau Bohr. Jika konstanta konversi tidak ditulis eksplisit, laporan sulit diperiksa. Dengan bagian konstanta, mahasiswa melatih kebiasaan ilmiah: semua angka penting harus memiliki nama, satuan, dan konteks.

Untuk praktikum partikel dalam kotak, definisi konstanta dapat ditulis:

```
h = 6.62607015e-34      # J s
m_e = 9.1093837015e-31  # kg
eV = 1.602176634e-19    # J
L = 1.0e-9              # panjang kotak, meter
```

Dengan konstanta ini, mahasiswa dapat menghitung energi dalam joule dan elektronvolt. Jika panjang kotak diubah, mahasiswa cukup mengubah nilai `L`. Ini membuat kode lebih fleksibel dan menghindari pengulangan angka.

3. Definisi Fungsi

Bagian ketiga adalah definisi fungsi. Fungsi adalah blok kode yang menjalankan tugas tertentu dan dapat digunakan berulang. Dalam praktikum kimia kuantum, fungsi sangat penting karena banyak perhitungan dilakukan berulang pada data berbeda. Misalnya, menghitung energi partikel dalam kotak untuk beberapa nilai n, menghitung panjang ikatan dari dua koordinat atom, menghitung sudut ikatan dari tiga atom, menghitung gap HOMO-LUMO, menghitung ΔE reaksi, atau membuat fungsi spektrum IR.

Contoh fungsi energi partikel dalam kotak:

```
def energi_partikel_kotak(n, L, m):
    """
    Menghitung energi partikel dalam kotak satu dimensi.

    Parameter:
    n : int
        Bilangan kuantum utama.
    L : float
        Panjang kotak dalam meter.
    m : float
        Massa partikel dalam kg.

    Return:
    float
        Energi dalam joule.
    """
    return (n**2 * h**2) / (8 * m * L**2)
```

Fungsi ini membuat kode lebih jelas. Daripada menulis rumus energi berkali-kali, mahasiswa cukup memanggil `energi_partikel_kotak(n, L, m_e)`. Selain itu, docstring menjelaskan parameter dan keluaran. Dokumentasi internal seperti ini penting karena kode praktikum tidak hanya untuk dijalankan, tetapi juga untuk dipelajari.

Contoh fungsi menghitung jarak atom:

```
def hitung_jarak(atom_A, atom_B):
    """
    Menghitung jarak antara dua atom berdasarkan koordinat 3D.
    Koordinat dinyatakan sebagai array [x, y, z].
    """
    return np.linalg.norm(atom_B - atom_A)
```

Contoh fungsi menghitung gap HOMO-LUMO:

```
def hitung_gap_homo_lumo(energi_homo, energi_lumo):
    """
    Menghitung gap HOMO-LUMO.
    Energi HOMO dan LUMO harus memiliki satuan yang sama.
    """
    return energi_lumo - energi_homo
```

Definisi fungsi melatih mahasiswa berpikir modular. Modular berarti persoalan besar dipecah menjadi bagian kecil yang jelas. Dalam kimia kuantum, ini sangat penting. Praktikum energi reaksi, misalnya, dapat dipecah menjadi fungsi membaca data, fungsi konversi satuan, fungsi menghitung energi reaksi, dan fungsi membuat grafik. Dengan struktur seperti ini, kode lebih mudah diperiksa dan diperbaiki.

Kesalahan umum mahasiswa adalah menulis seluruh perhitungan dalam satu blok panjang tanpa fungsi. Kode seperti itu mungkin berjalan, tetapi sulit dipahami. Jika ada kesalahan, sulit menemukan sumbernya. Dengan fungsi, setiap bagian dapat diuji. Misalnya, fungsi `hitung_jarak()` dapat diuji dengan dua titik sederhana sebelum digunakan untuk molekul nyata. Ini sejalan dengan prinsip reprodusibilitas dan validasi.

4. Input Data

Bagian keempat adalah input data. Input data adalah bagian kode yang memasukkan data mentah ke dalam program. Data dapat ditulis langsung dalam kode, dibaca dari file CSV, dibaca dari file XYZ, diambil dari output software, atau dimasukkan sebagai parameter model. Dalam praktikum kimia kuantum, input data harus jelas sumbernya. Data tidak boleh muncul tiba-tiba tanpa keterangan.

Contoh input data langsung:

```
energi_homo = -9.25  # eV
energi_lumo = 0.85   # eV
```

Contoh input data dari tabel CSV:

```
data = pd.read_csv("data/homo_lumo.csv")
```

Pandas menyediakan fungsi `read_csv()` untuk membaca file CSV menjadi DataFrame, sedangkan `DataFrame.to_csv()` digunakan untuk menulis data ke file CSV. Fungsi-fungsi ini menjadi dasar penting pengelolaan data praktikum karena CSV mudah dibaca manusia, spreadsheet, dan Python. (pandas.pydata.org) (pandas.pydata.org)

Contoh input data koordinat secara langsung:

```
koordinat = {
    "O": np.array([0.000000, 0.000000, 0.000000]),
    "H1": np.array([0.758602, 0.000000, 0.504284]),
```

```
    "H2": np.array([-0.758602, 0.000000, 0.504284])
}
```

Dalam praktikum yang lebih tertib, koordinat sebaiknya dibaca dari file XYZ agar tidak terjadi salah salin. Namun, pada tahap awal, input langsung dapat digunakan untuk memperkenalkan konsep.

Input data harus selalu menyertakan satuan. Jika energi dalam eV, tuliskan eV. Jika energi dalam Hartree, tuliskan Hartree. Jika koordinat dalam Ångström, tuliskan Ångström. Jika frekuensi dalam cm^{-1}, tuliskan cm^{-1}. Dalam kode, satuan dapat ditulis pada nama variabel, misalnya `energi_hartree`, `gap_ev`, `frekuensi_cm1`, atau `jarak_angstrom`. Nama variabel seperti ini sangat membantu pembaca.

Kesalahan umum pada input data adalah mencampur satuan. Misalnya, energi HOMO dalam eV dan energi LUMO dalam Hartree dihitung langsung untuk gap. Atau energi reaktan dalam Hartree dan produk dalam kJ/mol dijumlahkan tanpa konversi. Struktur kode yang baik harus mencegah kesalahan ini dengan memberi nama variabel yang jelas dan melakukan konversi satuan secara eksplisit.

5. Perhitungan

Bagian kelima adalah perhitungan. Perhitungan merupakan inti kode praktikum. Pada bagian ini, data input diproses menggunakan rumus, fungsi, algoritma, atau prosedur numerik tertentu. Dalam praktikum kimia kuantum, perhitungan dapat berupa energi partikel dalam kotak, normalisasi fungsi gelombang, panjang ikatan, sudut ikatan, gap HOMO-LUMO, energi reaksi, konversi satuan, atau pembuatan spektrum IR simulasi.

Contoh perhitungan energi partikel dalam kotak:

```
n_values = np.arange(1, 6)
energi_joule = energi_partikel_kotak(n_values, L, m_e)
energi_ev = energi_joule / eV
```

Contoh perhitungan gap HOMO-LUMO dari tabel:

```
data["gap_ev"] = data["lumo_ev"] - data["homo_ev"]
```

Contoh perhitungan energi reaksi:

```
hartree_to_kjmol = 2625.49962

energi_reaktan = -154.123456 + -40.123456
energi_produk = -194.300000

delta_e_hartree = energi_produk - energi_reaktan
delta_e_kjmol = delta_e_hartree * hartree_to_kjmol
```

Perhitungan harus ditulis secara transparan. Jangan menyembunyikan rumus penting dalam angka tunggal. Misalnya, jika menghitung ΔE, tuliskan jelas `energi_produk - energi_reaktan`. Jika menghitung gap HOMO-LUMO, tuliskan jelas `lumo - homo`. Jika mengonversi Hartree ke kJ/mol, tuliskan faktor konversi sebagai konstanta. Dengan cara ini, dosen dapat memeriksa apakah logika perhitungan benar.

Dalam praktikum kimia kuantum, perhitungan juga harus dikaitkan dengan teori. Kode bukan sekadar prosedur numerik. Jika menghitung ψ^2, mahasiswa harus memahami bahwa ψ^2 berkaitan dengan rapat probabilitas. Jika menghitung gap HOMO-LUMO, mahasiswa harus memahami bahwa gap tersebut merupakan selisih energi orbital frontier dalam model yang digunakan. Jika menghitung ΔE, mahasiswa harus memahami bahwa nilai itu bergantung pada metode, basis set, dan konsistensi stoikiometri.

6. Tabel Hasil

Bagian keenam adalah tabel hasil. Tabel hasil menyusun output perhitungan dalam bentuk yang mudah dibaca. Dalam Python, tabel hasil umumnya dibuat dengan pandas DataFrame. Tabel sangat penting karena laporan praktikum membutuhkan data ringkas. Tabel membantu mahasiswa dan dosen melihat pola, membandingkan nilai, dan memeriksa konsistensi.

Contoh tabel energi partikel dalam kotak:

```
hasil = pd.DataFrame({
    "n": n_values,
    "energi_joule": energi_joule,
    "energi_ev": energi_ev
})

print(hasil)
```

Contoh tabel HOMO-LUMO:

```
hasil_homo_lumo = data[["molekul", "homo_ev", "lumo_ev", "gap_ev"]]
print(hasil_homo_lumo)
```

Tabel harus memiliki nama kolom yang jelas. Kolom `energi` kurang baik jika tidak menyebut satuan. Lebih baik `energi_hartree`, `energi_ev`, atau `energi_kjmol`. Kolom `hasil` terlalu umum. Lebih baik `gap_ev`, `jarak_oh_angstrom`, atau `frekuensi_cm1`.

Tabel hasil juga dapat digunakan untuk memeriksa kesalahan. Jika gap HOMO-LUMO negatif, kemungkinan ada kesalahan input atau penamaan kolom. Jika panjang ikatan O–H menjadi 10 Å, kemungkinan koordinat salah. Jika energi reaksi ratusan ribu kJ/mol untuk reaksi sederhana, kemungkinan satuan atau stoikiometri salah. Dengan tabel yang rapi, kesalahan numerik lebih mudah dideteksi.

7. Grafik

Bagian ketujuh adalah grafik. Grafik mengubah data numerik menjadi visualisasi yang membantu interpretasi. Dalam praktikum kimia kuantum, grafik dapat berupa fungsi gelombang, rapat probabilitas, grafik energi terhadap bilangan kuantum, grafik energi terhadap panjang ikatan, diagram HOMO-LUMO, spektrum IR simulasi, atau diagram energi reaksi. Jupyter Notebook mendukung kombinasi kode, teks naratif, persamaan, dan visualisasi, sehingga grafik dapat langsung ditempatkan di dekat analisisnya. (jupyter.org)

Contoh grafik energi partikel dalam kotak:

```
plt.figure()
plt.plot(hasil["n"], hasil["energi_ev"], marker="o")
plt.xlabel("Bilangan kuantum n")
plt.ylabel("Energi (eV)")
plt.title("Energi Partikel dalam Kotak")
plt.grid(True)
plt.show()
```

Contoh diagram HOMO-LUMO sederhana:

```
plt.figure()
plt.bar(data["molekul"], data["gap_ev"])
plt.xlabel("Molekul")
plt.ylabel("Gap HOMO-LUMO (eV)")
plt.title("Perbandingan Gap HOMO-LUMO")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Grafik harus memenuhi standar ilmiah. Sumbu harus diberi label. Satuan harus ditulis. Judul harus informatif. Legenda diberikan jika ada lebih dari satu kurva. Skala grafik harus sesuai. Grafik tidak boleh dibuat hanya agar laporan tampak menarik; grafik harus menjawab pertanyaan praktikum. Jika grafik tidak membantu interpretasi, lebih baik tidak digunakan.

Dalam praktikum spektrum IR, grafik sangat penting. Daftar frekuensi dan intensitas dapat diubah menjadi spektrum yang lebih mudah ditafsirkan. Namun, mahasiswa harus mencatat bagaimana spektrum dibuat: apakah menggunakan pelebaran Gaussian, berapa lebar puncaknya, apakah intensitas dinormalisasi, dan apakah frekuensi diberi faktor skala. Tanpa informasi ini, grafik spektrum sulit direproduksi.

8. Ekspor Data

Bagian kedelapan adalah ekspor data. Ekspor data berarti menyimpan hasil perhitungan, tabel, atau grafik ke file agar dapat digunakan dalam laporan dan diperiksa ulang. Data dapat diekspor dalam format CSV, TXT, PNG, PDF, atau format lain. Dalam praktikum kimia kuantum, ekspor data mendukung reprodusibilitas. Jika hasil hanya tampil di layar tetapi tidak disimpan, proses analisis sulit diperiksa kembali.

Contoh ekspor tabel ke CSV:

```
hasil.to_csv("data/energi_partikel_kotak.csv", index=False)
```

Contoh ekspor grafik:

```
plt.figure()
plt.plot(hasil["n"], hasil["energi_ev"], marker="o")
plt.xlabel("Bilangan kuantum n")
plt.ylabel("Energi (eV)")
plt.title("Energi Partikel dalam Kotak")
plt.grid(True)
plt.savefig("gambar/energi_partikel_kotak.png", dpi=300, bbox_inches="tight")
plt.show()
```

Pandas menyediakan `DataFrame.to_csv()` untuk menulis DataFrame ke file CSV, sehingga tabel hasil dapat disimpan secara terbuka dan dibaca ulang oleh Python, spreadsheet, atau perangkat lunak lain. (pandas.pydata.org)

Ekspor data harus mengikuti standar folder Bab 2. Tabel disimpan di folder `data/`, grafik di `gambar/`, kode di `python/`, dan laporan di `laporan/`. Jangan menyimpan semua file di satu folder tanpa struktur. Struktur folder yang rapi membantu dosen memeriksa hasil dan membantu mahasiswa menemukan file saat menyusun laporan.

Kesalahan umum adalah hanya menyimpan notebook tanpa menyimpan data hasil. Notebook memang dapat memuat output, tetapi data terstruktur tetap perlu diekspor. Jika notebook rusak atau output tidak tampil, file CSV masih dapat digunakan. Jika grafik ingin diperbaiki, data CSV dapat dibaca kembali. Dengan demikian, ekspor data merupakan bagian penting dari praktik ilmiah digital.

9. Kesimpulan Numerik

Bagian kesembilan adalah kesimpulan numerik. Kesimpulan numerik adalah pernyataan ringkas yang dihasilkan dari data dan perhitungan. Dalam kode praktikum, kesimpulan numerik dapat ditampilkan dengan `print()` atau ditulis sebagai Markdown dalam Jupyter Notebook. Kesimpulan ini tidak menggantikan pembahasan laporan, tetapi membantu mahasiswa melihat hasil utama secara langsung.

Contoh kesimpulan numerik untuk gap HOMO-LUMO:

```
molekul_gap_terkecil = data.loc[data["gap_ev"].idxmin(), "molekul"]
gap_terkecil = data["gap_ev"].min()

print(f"Molekul dengan gap HOMO-LUMO terkecil adalah {molekul_gap_terkecil},
yaitu {gap_terkecil:.2f} eV.")
```

Contoh kesimpulan numerik untuk energi reaksi:

```
if delta_e_kjmol < 0:
```

```
    sifat = "eksotermik secara energi elektronik"
else:
    sifat = "endotermik secara energi elektronik"

print(f"ΔE = {delta_e_kjmol:.2f} kJ/mol, sehingga reaksi bersifat {sifat} pada
tingkat teori yang digunakan.")
```

Kalimat “pada tingkat teori yang digunakan” penting. Dalam kimia komputasi, kesimpulan harus selalu dibatasi oleh metode, basis set, dan kondisi perhitungan. Kode boleh menghitung angka, tetapi interpretasi tetap harus ilmiah. Jangan menulis “reaksi pasti spontan” hanya dari ΔE elektronik. Spontanitas membutuhkan ΔG, suhu, entropi, dan konteks lingkungan. Kesimpulan numerik harus teliti dan tidak berlebihan.

Kesimpulan numerik juga membantu validasi. Jika kode menampilkan hasil yang tidak masuk akal, mahasiswa segera menyadari ada masalah. Misalnya, jika kesimpulan menyatakan gap HOMO-LUMO negatif untuk molekul stabil pada data biasa, mahasiswa perlu memeriksa input. Jika ΔE sangat besar tanpa alasan, periksa konversi satuan. Jika sudut ikatan air 360°, periksa rumus sudut.

Contoh Struktur Kode Lengkap: Partikel dalam Kotak

Berikut contoh struktur kode praktikum yang mengikuti sembilan bagian utama:

```
# ============================================================
# Praktikum: Partikel dalam Kotak Satu Dimensi
# Tujuan: Menghitung energi dan fungsi gelombang untuk beberapa n
# ============================================================

# 1. Import pustaka
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# 2. Definisi konstanta
h = 6.62607015e-34       # J s
m_e = 9.1093837015e-31   # kg
eV = 1.602176634e-19     # J
L = 1.0e-9               # meter

# 3. Definisi fungsi
def energi_partikel_kotak(n, L, m):
    """Menghitung energi partikel dalam kotak satu dimensi dalam joule."""
    return (n**2 * h**2) / (8 * m * L**2)

def fungsi_gelombang(n, x, L):
    """Menghitung fungsi gelombang ternormalisasi partikel dalam kotak."""
    return np.sqrt(2/L) * np.sin(n * np.pi * x / L)

# 4. Input data
n_values = np.arange(1, 6)
x = np.linspace(0, L, 500)

# 5. Perhitungan
```

```
energi_joule = energi_partikel_kotak(n_values, L, m_e)
energi_ev = energi_joule / eV

# 6. Tabel hasil
hasil = pd.DataFrame({
    "n": n_values,
    "energi_joule": energi_joule,
    "energi_ev": energi_ev
})

print(hasil)

# 7. Grafik
plt.figure()
for n in [1, 2, 3]:
    psi = fungsi_gelombang(n, x, L)
    plt.plot(x * 1e9, psi, label=f"n = {n}")

plt.xlabel("x (nm)")
plt.ylabel("ψ(x)")
plt.title("Fungsi Gelombang Partikel dalam Kotak")
plt.legend()
plt.grid(True)
plt.tight_layout()

# 8. Ekspor data
hasil.to_csv("data/energi_partikel_kotak.csv", index=False)
plt.savefig("gambar/fungsi_gelombang_partikel_kotak.png", dpi=300)

plt.show()

# 9. Kesimpulan numerik
print(f"Energi n = 1 adalah {energi_ev[0]:.4f} eV.")
print("Energi meningkat sebanding dengan n² pada panjang kotak yang tetap.")
```

Kode ini memperlihatkan struktur yang rapi. Import pustaka berada di awal. Konstanta diberi satuan. Fungsi didefinisikan sebelum digunakan. Input data jelas. Perhitungan dilakukan setelah input. Hasil dibuat sebagai DataFrame. Grafik memiliki label sumbu dan satuan. Data diekspor ke folder `data/`, grafik ke folder `gambar/`. Kesimpulan numerik ditampilkan di akhir.

Contoh ini juga menunjukkan bahwa struktur kode praktikum bukan hanya urutan teknis, tetapi alur pedagogis. Mahasiswa melihat teori partikel dalam kotak diterjemahkan menjadi kode, data, grafik, dan kesimpulan. Inilah tujuan utama penggunaan Python dalam buku ini.

Contoh Struktur Kode Lengkap: Analisis HOMO-LUMO

Contoh berikut menunjukkan struktur kode untuk analisis HOMO-LUMO dari data output yang sudah diringkas ke CSV.

Misalnya file `data/homo_lumo.csv` berisi:

```
molekul,homo_ev,lumo_ev
etilena,-10.5,1.2
```

```
butadiena,-8.9,0.3
benzena,-9.2,0.8
```

Kode analisis:

```
# ============================================================
# Praktikum: Analisis HOMO-LUMO
# Tujuan: Menghitung gap HOMO-LUMO dan membuat grafik perbandingan
# ============================================================

# 1. Import pustaka
import pandas as pd
import matplotlib.pyplot as plt

# 2. Definisi konstanta
# Tidak diperlukan konstanta fisik khusus karena energi sudah dalam eV.

# 3. Definisi fungsi
def hitung_gap_homo_lumo(homo, lumo):
    """Menghitung gap HOMO-LUMO dalam satuan yang sama dengan input."""
    return lumo - homo

# 4. Input data
data = pd.read_csv("data/homo_lumo.csv")

# 5. Perhitungan
data["gap_ev"] = hitung_gap_homo_lumo(data["homo_ev"], data["lumo_ev"])

# 6. Tabel hasil
print(data)

# 7. Grafik
plt.figure()
plt.bar(data["molekul"], data["gap_ev"])
plt.xlabel("Molekul")
plt.ylabel("Gap HOMO-LUMO (eV)")
plt.title("Perbandingan Gap HOMO-LUMO")
plt.grid(axis="y")
plt.tight_layout()

# 8. Ekspor data
data.to_csv("data/homo_lumo_hasil.csv", index=False)
plt.savefig("gambar/gap_homo_lumo.png", dpi=300)

plt.show()

# 9. Kesimpulan numerik
molekul_min = data.loc[data["gap_ev"].idxmin(), "molekul"]
gap_min = data["gap_ev"].min()

print(f"Gap HOMO-LUMO terkecil dimiliki oleh {molekul_min}, yaitu {gap_min:.2f}
eV.")
```

Contoh ini menunjukkan bahwa struktur kode yang sama dapat digunakan untuk praktikum berbeda. Inilah alasan buku ini menekankan template umum. Mahasiswa tidak perlu memulai dari

nol setiap kali. Mereka cukup menyesuaikan fungsi, input data, perhitungan, dan grafik sesuai topik.

Contoh Struktur Kode Lengkap: Energi Reaksi

Untuk praktikum energi reaksi, struktur kode dapat digunakan untuk menghitung ΔE dari energi reaktan dan produk.

Misalnya file `data/energi_reaksi.csv` berisi:

```
spesies,koefisien,energi_hartree
reaktan_A,-1,-154.123456
reaktan_B,-1,-40.123456
produk_C,1,-194.300000
```

Kode:

```
# ============================================================
# Praktikum: Energi Reaksi
# Tujuan: Menghitung ΔE reaksi dari energi elektronik
# ============================================================

# 1. Import pustaka
import pandas as pd
import matplotlib.pyplot as plt

# 2. Definisi konstanta
hartree_to_kjmol = 2625.49962  # kJ/mol

# 3. Definisi fungsi
def hitung_delta_e(data):
    """Menghitung ΔE dalam Hartree berdasarkan koefisien stoikiometri."""
    return (data["koefisien"] * data["energi_hartree"]).sum()

# 4. Input data
data = pd.read_csv("data/energi_reaksi.csv")

# 5. Perhitungan
delta_e_hartree = hitung_delta_e(data)
delta_e_kjmol = delta_e_hartree * hartree_to_kjmol

# 6. Tabel hasil
ringkasan = pd.DataFrame({
    "parameter": ["delta_e_hartree", "delta_e_kjmol"],
    "nilai": [delta_e_hartree, delta_e_kjmol]
})

print(ringkasan)

# 7. Grafik
plt.figure()
plt.bar(["Reaksi"], [delta_e_kjmol])
plt.ylabel("ΔE (kJ/mol)")
plt.title("Energi Reaksi")
```

```
plt.axhline(0)
plt.tight_layout()

# 8. Ekspor data
ringkasan.to_csv("data/ringkasan_energi_reaksi.csv", index=False)
plt.savefig("gambar/energi_reaksi.png", dpi=300)

plt.show()

# 9. Kesimpulan numerik
if delta_e_kjmol < 0:
    sifat = "lebih rendah energi pada sisi produk"
else:
    sifat = "lebih tinggi energi pada sisi produk"

print(f"ΔE = {delta_e_kjmol:.2f} kJ/mol; reaksi {sifat} pada tingkat teori yang
digunakan.")
```

Contoh ini memperlihatkan bahwa struktur kode membantu mencegah kesalahan stoikiometri. Koefisien reaksi ditulis eksplisit dalam tabel, lalu dikalikan dengan energi. Jika koefisien salah, tabel dapat diperiksa. Jika satuan salah, konstanta konversi dapat diperiksa. Jika grafik salah, data sumbernya tersedia.

Analisis Kritis terhadap Struktur Kode

Struktur kode yang rapi bukan berarti kode harus panjang atau rumit. Justru tujuan struktur adalah membuat kode sederhana, terbaca, dan dapat diperiksa. Kode yang terlalu pendek tetapi tidak jelas sering berbahaya. Misalnya:

```
x = (a - b) * 2625.5
print(x)
```

Kode tersebut mungkin menghasilkan angka benar, tetapi tidak jelas apa itu `a`, `b`, dan `x`. Apakah `a` energi produk? Apakah `b` energi reaktan? Apakah satuannya Hartree? Apakah 2625.5 adalah faktor konversi? Kode seperti ini tidak layak untuk praktikum ilmiah karena tidak transparan.

Kode yang lebih baik:

```
hartree_to_kjmol = 2625.49962
energi_produk_hartree = -194.300000
energi_reaktan_hartree = -194.246912

delta_e_hartree = energi_produk_hartree - energi_reaktan_hartree
delta_e_kjmol = delta_e_hartree * hartree_to_kjmol

print(f"ΔE = {delta_e_kjmol:.2f} kJ/mol")
```

Kode kedua lebih panjang, tetapi lebih ilmiah. Nama variabel menjelaskan makna. Konstanta diberi nama. Rumus terlihat. Output memiliki satuan. Inilah prinsip dasar struktur kode praktikum.

Analisis kritis juga perlu menyoroti bahaya notebook yang tidak dijalankan berurutan. Dalam Jupyter Notebook, mahasiswa dapat menjalankan sel ke-10 sebelum sel ke-3, atau mengubah variabel tetapi tidak menjalankan ulang semua sel. Akibatnya, output yang tampil dapat berasal dari keadaan kernel lama. Dokumentasi Jupyter menjelaskan bahwa notebook memiliki tipe sel seperti code, markdown, dan raw, serta setiap sel memiliki perilaku eksekusi sesuai tipenya. (jupyter-notebook.readthedocs.io) Karena itu, sebelum notebook dikumpulkan, mahasiswa harus menjalankan ulang seluruh notebook dari awal agar hasil konsisten dengan kode terakhir.

Struktur kode juga harus memperhatikan batas antara kode dan narasi. Dalam notebook, penjelasan teori singkat dapat ditulis dalam Markdown cell. Jupyter mendukung Markdown cell untuk menambahkan teks dalam notebook. (jupyter-notebook.readthedocs.io) Kode ditulis dalam code cell. Hasil dan grafik muncul sebagai output. Dengan kombinasi ini, notebook praktikum dapat menjadi dokumen ilmiah mini: ada tujuan, teori, kode, hasil, dan interpretasi. Namun, notebook tetap harus menyimpan data dan gambar terpisah sesuai standar file Bab 2.

Implikasi Pedagogis

Struktur kode praktikum memiliki implikasi pedagogis yang kuat. Pertama, struktur ini membantu mahasiswa berpikir sistematis. Mereka belajar bahwa analisis ilmiah memiliki urutan: alat, konstanta, fungsi, data, perhitungan, hasil, grafik, ekspor, kesimpulan. Kedua, struktur ini membantu dosen menilai proses. Jika hasil salah, dosen dapat melihat apakah kesalahan terjadi pada data input, fungsi, rumus, satuan, atau grafik. Ketiga, struktur ini mendorong reprodusibilitas. Kode yang terstruktur lebih mudah dijalankan ulang. Keempat, struktur ini mempersiapkan mahasiswa untuk penelitian ilmiah yang lebih serius.

Dalam praktikum kimia kuantum, struktur kode juga membantu menghubungkan teori dan data. Mahasiswa tidak hanya menulis rumus di laporan, tetapi mengimplementasikan rumus itu dalam kode. Mereka tidak hanya melihat output software, tetapi memproses output tersebut menjadi tabel dan grafik. Mereka tidak hanya menyimpulkan secara verbal, tetapi menyusun kesimpulan numerik berdasarkan hasil. Dengan demikian, struktur kode membentuk literasi komputasi yang menyatu dengan literasi kimia.

Implikasi terhadap Bab-Bab Praktikum Berikutnya

Struktur kode dalam subbab ini akan digunakan berulang pada bab-bab berikutnya. Pada Bab 4, struktur kode digunakan untuk partikel dalam kotak. Pada Bab 5, digunakan untuk analisis optimasi geometri. Pada Bab 6, digunakan untuk Hartree–Fock dan basis set. Pada Bab 7, digunakan untuk DFT dan HOMO-LUMO. Pada Bab 8, digunakan untuk simetri dan momen dipol. Pada Bab 9, digunakan untuk frekuensi IR. Pada Bab 10, digunakan untuk energi reaksi dan termodinamika. Pada Bab 11, struktur kode menjadi dasar otomasi laporan.

Dengan pola yang sama, mahasiswa akan terbiasa membaca kode praktikum. Mereka tidak perlu mempelajari struktur baru setiap bab. Yang berubah adalah konteks kimianya, fungsi yang digunakan, data input, dan bentuk grafik. Kebiasaan ini membuat pembelajaran lebih efisien dan mendalam. Mahasiswa dapat fokus pada makna kimia, bukan bingung dengan format kode yang berubah-ubah.

Struktur kode praktikum merupakan jembatan antara teori, data, dan laporan. Kode yang baik harus dimulai dari import pustaka, dilanjutkan dengan definisi konstanta, definisi fungsi, input data, perhitungan, tabel hasil, grafik, ekspor data, dan kesimpulan numerik. Struktur ini membuat proses analisis lebih jelas, lebih mudah diperiksa, dan lebih reprodusibel. Dalam praktikum kimia kuantum, struktur kode bukan sekadar gaya pemrograman, tetapi bagian dari metodologi ilmiah.

Mahasiswa yang menguasai struktur kode akan lebih siap membaca output ChemCompute, mengolah data energi, menghitung parameter geometri, membuat grafik orbital, menyusun spektrum IR, dan menghitung energi reaksi. Mereka juga akan lebih siap menulis laporan yang berbasis data, bukan sekadar narasi. Dengan demikian, struktur kode menjadi fondasi penting sebelum membahas standar penulisan kode pada subbab berikutnya.

3.4 Standar Penulisan Kode

Standar penulisan kode adalah seperangkat aturan, kebiasaan, dan prinsip teknis yang digunakan agar kode Python dalam praktikum kimia kuantum mudah dibaca, mudah dijalankan ulang, mudah diperiksa, mudah diperbaiki, dan menghasilkan output yang dapat dipakai dalam laporan ilmiah. Dalam konteks buku ini, standar penulisan kode tidak dipahami sebagai urusan estetika pemrograman semata, tetapi sebagai bagian dari metodologi ilmiah. Kode yang tidak jelas akan menghasilkan analisis yang sulit diverifikasi. Sebaliknya, kode yang rapi, terdokumentasi, dan dapat dijalankan ulang akan memperkuat reprodusibilitas praktikum.

Dalam praktikum kimia kuantum, kode Python memiliki fungsi akademik yang sangat penting. Kode digunakan untuk menghitung energi model, membuat grafik fungsi gelombang, membaca koordinat XYZ, menghitung panjang ikatan dan sudut ikatan, membuat tabel energi, menghitung gap HOMO-LUMO, mengonversi satuan, membuat spektrum IR simulasi, serta menghitung energi reaksi. Jika kode ditulis sembarangan, kesalahan kecil dapat berdampak besar pada kesimpulan. Misalnya, salah konversi Hartree ke kJ/mol dapat mengubah interpretasi energi reaksi. Salah membaca kolom HOMO dan LUMO dapat menghasilkan gap energi yang keliru. Salah memberi satuan pada grafik dapat membuat pembaca salah memahami hasil.

Karena itu, standar penulisan kode dalam buku ini bertumpu pada lima prinsip utama:

1. **Kode harus memiliki komentar.**
2. **Kode harus memiliki nama variabel yang jelas.**
3. **Kode tidak boleh bergantung pada file tersembunyi.**
4. **Kode harus dapat dijalankan ulang.**
5. **Kode harus menghasilkan output yang dapat digunakan dalam laporan.**

Kelima prinsip tersebut saling berhubungan. Komentar membantu pembaca memahami maksud kode. Nama variabel yang jelas membantu mencegah salah tafsir. Ketiadaan ketergantungan pada file tersembunyi membuat kode dapat dijalankan di komputer lain. Kemampuan dijalankan ulang mendukung reprodusibilitas. Output yang siap digunakan dalam laporan membuat kode berperan langsung dalam komunikasi ilmiah.

Secara historis, standar penulisan kode dalam sains berkembang seiring meningkatnya penggunaan komputer dalam penelitian. Pada awalnya, banyak program ilmiah ditulis oleh peneliti untuk dirinya sendiri. Kode hanya perlu berjalan di komputer tertentu dan dipahami oleh penulisnya. Namun, ketika komputasi ilmiah menjadi bagian dari publikasi, pendidikan, kolaborasi, dan reprodusibilitas riset, kode tidak lagi boleh menjadi catatan pribadi yang kabur. Kode harus dapat dibaca oleh orang lain, diperiksa, dan dijalankan kembali. Dalam tradisi Python, prinsip ini sejalan dengan PEP 8 yang menekankan konsistensi dan keterbacaan kode, serta *The Zen of Python* yang menegaskan bahwa keterbacaan kode itu penting dan bahwa sesuatu yang eksplisit lebih baik daripada yang implisit. Prinsip-prinsip tersebut sangat relevan untuk praktikum kimia kuantum karena mahasiswa harus belajar menulis kode sebagai bagian dari laporan ilmiah, bukan hanya sebagai alat hitung pribadi.

1. Kode Harus Memiliki Komentar

Komentar adalah keterangan dalam kode yang ditulis untuk menjelaskan maksud, satuan, asumsi, atau langkah penting dalam perhitungan. Dalam Python, komentar satu baris ditulis menggunakan tanda `#`. Komentar tidak dijalankan sebagai perintah, tetapi dibaca oleh manusia. Dalam praktikum kimia kuantum, komentar diperlukan karena kode sering mengandung rumus, konstanta fisika, konversi satuan, dan prosedur analisis data yang perlu dijelaskan.

Contoh kode tanpa komentar:

```
h = 6.62607015e-34
m = 9.1093837015e-31
L = 1e-9
E = h**2 / (8*m*L**2)
print(E)
```

Kode tersebut mungkin benar, tetapi pembaca harus menebak arti `h`, `m`, `L`, dan `E`. Bagi mahasiswa pemula, kode seperti ini kurang mendidik. Bandingkan dengan kode berikut:

```
# Konstanta Planck dalam satuan J s
h = 6.62607015e-34

# Massa elektron dalam satuan kg
m_e = 9.1093837015e-31

# Panjang kotak dalam model partikel dalam kotak, satuan meter
L = 1.0e-9

# Energi keadaan dasar partikel dalam kotak satu dimensi
E1_joule = h**2 / (8 * m_e * L**2)

print(E1_joule)
```

Kode kedua jauh lebih baik karena setiap angka penting diberi keterangan. Pembaca mengetahui satuan, makna variabel, dan tujuan perhitungan. Komentar membantu mahasiswa menghubungkan kode dengan teori kimia kuantum.

Namun, komentar juga tidak boleh berlebihan. Komentar yang baik menjelaskan **mengapa** suatu langkah dilakukan, bukan sekadar mengulang apa yang sudah jelas dari kode. Misalnya:

```
x = x + 1  # menambah x dengan 1
```

Komentar seperti ini tidak banyak membantu. Kode sudah menjelaskan dirinya sendiri. Komentar yang lebih berguna adalah:

```
# Menambah indeks bilangan kuantum untuk menghitung keadaan berikutnya
n = n + 1
```

Dalam praktikum kimia kuantum, komentar sangat diperlukan pada bagian berikut:

- Definisi konstanta fisika.
- Konversi satuan.
- Rumus energi, gap, atau frekuensi.
- Pemilihan kolom data.
- Pembacaan file input.
- Pembuatan grafik.
- Validasi hasil.
- Ekspor data.

Contoh komentar yang baik dalam analisis energi reaksi:

```
# Faktor konversi energi:
# 1 Hartree = 2625.49962 kJ/mol
hartree_to_kjmol = 2625.49962

# ΔE dihitung dari energi produk dikurangi energi reaktan
delta_e_hartree = energi_produk_hartree - energi_reaktan_hartree

# Konversi ΔE dari Hartree ke kJ/mol
delta_e_kjmol = delta_e_hartree * hartree_to_kjmol
```

Komentar semacam ini membuat kode lebih transparan. Jika hasil energi reaksi tampak aneh, dosen dapat memeriksa rumus dan satuan dengan mudah. Mahasiswa juga belajar bahwa setiap angka dalam laporan harus memiliki jejak perhitungan yang jelas.

Selain komentar biasa, fungsi sebaiknya memiliki **docstring**, yaitu keterangan di dalam tanda kutip tiga yang menjelaskan fungsi, parameter, dan keluaran. Contoh:

```
def hitung_gap_homo_lumo(energi_homo_ev, energi_lumo_ev):
    """
    Menghitung gap HOMO-LUMO dalam satuan eV.

    Parameter:
    energi_homo_ev : float
        Energi HOMO dalam eV.
    energi_lumo_ev : float
        Energi LUMO dalam eV.
```

```
    Return:
    float
        Gap HOMO-LUMO dalam eV.
    """
    return energi_lumo_ev - energi_homo_ev
```

Docstring sangat penting jika fungsi akan digunakan berulang pada beberapa praktikum. Mahasiswa yang membuka kode beberapa minggu kemudian tetap dapat memahami tujuan fungsi tersebut. Dalam proyek mini, docstring membantu anggota kelompok lain memahami kode yang ditulis temannya.

2. Nama Variabel Harus Jelas

Nama variabel adalah label yang diberikan pada data, konstanta, atau hasil perhitungan. Dalam kode ilmiah, nama variabel harus menjelaskan makna kimia atau fisik dari nilai yang disimpan. Nama variabel yang buruk membuat kode sulit dipahami dan rawan kesalahan. Nama variabel yang jelas membuat kode hampir dapat dibaca seperti kalimat ilmiah.

Contoh nama variabel yang kurang baik:

```
a = -74.963
b = -75.100
c = b - a
print(c)
```

Kode tersebut tidak menjelaskan apa itu `a`, `b`, dan `c`. Apakah `a` energi reaktan? Apakah `b` energi produk? Apakah satuannya Hartree? Apakah `c` selisih energi? Kode seperti ini tidak layak untuk laporan praktikum.

Contoh yang lebih baik:

```
energi_reaktan_hartree = -74.963
energi_produk_hartree = -75.100

delta_e_hartree = energi_produk_hartree - energi_reaktan_hartree

print(delta_e_hartree)
```

Kode kedua langsung dapat dipahami. Nama variabel menunjukkan makna dan satuan. Dalam praktikum kimia kuantum, satuan sebaiknya dimasukkan ke nama variabel jika nilai numerik dapat memiliki banyak satuan. Misalnya:

```
energi_hartree
energi_kjmol
energi_ev
frekuensi_cm1
jarak_angstrom
sudut_derajat
massa_kg
```

```
panjang_meter
dipol_debye
```

Penamaan seperti ini mencegah kesalahan konversi. Misalnya, jika variabel bernama `energi_hartree`, mahasiswa akan berhati-hati sebelum menuliskannya dalam tabel kJ/mol. Jika variabel bernama `frekuensi_cm1`, pembaca langsung tahu bahwa satuannya adalah cm^{-1}.

Untuk nama variabel dalam Python, gunakan huruf kecil dan garis bawah. Ini sesuai dengan gaya umum Python. Contoh:

```
energi_total_hartree
panjang_ikatan_oh
sudut_hoh_derajat
gap_homo_lumo_ev
delta_g_kjmol
```

Hindari nama variabel yang terlalu pendek kecuali untuk indeks matematika yang sangat umum, seperti `x`, `y`, `n`, atau `i` dalam konteks sederhana. Bahkan untuk `x`, sebaiknya diberi keterangan jika berhubungan dengan satuan:

```
x_meter = np.linspace(0, L_meter, 500)
```

Dalam praktikum geometri, nama atom juga harus jelas. Misalnya:

```
koordinat_O = np.array([0.000000, 0.000000, 0.000000])
koordinat_H1 = np.array([0.758602, 0.000000, 0.504284])
koordinat_H2 = np.array([-0.758602, 0.000000, 0.504284])
```

Ini lebih baik daripada:

```
a = np.array([0.000000, 0.000000, 0.000000])
b = np.array([0.758602, 0.000000, 0.504284])
c = np.array([-0.758602, 0.000000, 0.504284])
```

Nama variabel yang jelas membantu mahasiswa menghindari salah memilih atom ketika menghitung sudut atau jarak. Dalam molekul besar, kesalahan indeks atom sangat sering terjadi. Karena itu, penamaan yang jelas adalah bagian dari validasi.

Dalam tabel pandas, nama kolom juga harus jelas. Contoh baik:

```
data = pd.DataFrame({
    "molekul": ["H2O", "NH3", "CO2"],
    "energi_hartree": [-74.963, -55.455, -187.650],
    "dipol_debye": [1.82, 1.47, 0.00]
})
```

Contoh kurang baik:

```
data = pd.DataFrame({
    "nama": ["H2O", "NH3", "CO2"],
```

```
    "x": [-74.963, -55.455, -187.650],
    "y": [1.82, 1.47, 0.00]
})
```

Kolom `x` dan `y` tidak menjelaskan makna data. Jika tabel ini diekspor ke CSV dan dibaca kembali beberapa hari kemudian, mahasiswa dapat lupa arti kolomnya. Standar penamaan kolom harus sama ketatnya dengan standar penamaan variabel.

3. Kode Tidak Boleh Bergantung pada File Tersembunyi

Kode praktikum tidak boleh bergantung pada file tersembunyi, file pribadi yang tidak disertakan, jalur absolut komputer tertentu, atau data yang hanya ada di perangkat penulis. Prinsip ini sangat penting untuk reprodusibilitas. Jika dosen atau mahasiswa lain tidak dapat menjalankan kode karena file sumber tidak tersedia, maka kode tersebut belum memenuhi standar praktikum.

Contoh kode yang buruk:

```
data = pd.read_csv("C:/Users/Kasmui/Desktop/hasil/energi.csv")
```

Kode ini hanya akan berjalan di komputer tertentu. Jika dijalankan di komputer lain, jalur tersebut tidak ada. Kode yang lebih baik menggunakan jalur relatif sesuai struktur folder praktikum:

```
data = pd.read_csv("data/energi.csv")
```

Dengan jalur relatif, kode dapat dijalankan selama folder `data/` berada dalam folder praktikum yang sama. Ini sesuai dengan standar file Bab 2.

Contoh lain kode yang bermasalah:

```
data = pd.read_csv("hasil_terbaru.csv")
```

Nama file ini tidak cukup jelas. Apakah `hasil_terbaru.csv` adalah hasil energi, frekuensi, HOMO-LUMO, atau geometri? Kode yang lebih baik:

```
data = pd.read_csv("data/homo_lumo_benzena.csv")
```

File tersembunyi juga dapat berupa file yang tidak disebutkan dalam laporan. Misalnya, kode membaca `konstanta.txt`, tetapi file tersebut tidak dikumpulkan. Atau kode membaca `output_asli.log`, tetapi hanya file CSV yang disertakan. Hal ini membuat kode tidak dapat dijalankan ulang. Dalam praktikum, semua file yang dibaca oleh kode harus ada dalam folder praktikum dan disebut dalam README atau laporan.

Kode juga tidak boleh bergantung pada variabel yang sudah ada di notebook karena sel sebelumnya pernah dijalankan, tetapi tidak tampak dalam kode final. Ini sering terjadi pada Jupyter Notebook. Misalnya, sel grafik menggunakan variabel `data`, tetapi sel pembacaan `data` sudah dihapus. Notebook mungkin masih menampilkan grafik karena kernel menyimpan variabel lama, tetapi jika dijalankan dari awal, kode gagal. Oleh karena itu, sebelum dikumpulkan, notebook

harus dijalankan ulang dari awal sampai akhir. Jika berhasil, berarti kode tidak bergantung pada keadaan tersembunyi.

Prinsip ini dapat dirumuskan sebagai berikut: **semua yang dibutuhkan kode harus tampak, tersedia, dan terdokumentasi**. Jika kode membutuhkan file, file harus ada. Jika kode membutuhkan paket, paket harus disebut. Jika kode membutuhkan parameter, parameter harus ditulis. Jika kode membutuhkan output ChemCompute, output harus disertakan. Dengan demikian, kode menjadi portabel dan reprodusibel.

4. Kode Harus Dapat Dijalankan Ulang

Kode yang baik harus dapat dijalankan ulang dari awal dan menghasilkan output yang sama, selama data input dan lingkungan komputasi sama. Ini adalah inti reprodusibilitas. Dalam praktikum kimia kuantum, kemampuan menjalankan ulang kode sangat penting karena dosen dapat memeriksa hasil, mahasiswa dapat memperbaiki analisis, dan laporan dapat dipertanggungjawabkan.

Kode dapat dijalankan ulang jika memenuhi beberapa syarat:

1. Semua paket yang diperlukan sudah diimpor.
2. Semua konstanta didefinisikan.
3. Semua fungsi didefinisikan sebelum dipanggil.
4. Semua file input tersedia.
5. Jalur file relatif dan jelas.
6. Tidak ada variabel tersembunyi.
7. Output tidak ditulis manual tanpa sumber.
8. Kode berjalan dari atas ke bawah tanpa error.

Dalam file `.py`, pengujian sederhana dapat dilakukan dengan menjalankan:

```
python nama_script.py
```

Dalam Jupyter Notebook, pengujian dilakukan dengan menjalankan seluruh sel dari awal. Jika notebook hanya berjalan ketika sel dipilih acak, notebook tersebut belum memenuhi standar. Kode praktikum harus dapat dijalankan oleh dosen tanpa harus menebak urutan eksekusi.

Contoh struktur yang mendukung eksekusi ulang:

```
import pandas as pd
import matplotlib.pyplot as plt

# Membaca data
data = pd.read_csv("data/energi_basis_set.csv")

# Menghitung selisih energi terhadap energi minimum
energi_min = data["energi_hartree"].min()
data["selisih_hartree"] = data["energi_hartree"] - energi_min
```

```
# Menyimpan hasil
data.to_csv("data/energi_basis_set_terolah.csv", index=False)

# Membuat grafik
plt.figure()
plt.plot(data["basis_set"], data["selisih_hartree"], marker="o")
plt.xlabel("Basis set")
plt.ylabel("Selisih energi (Hartree)")
plt.title("Perbandingan Energi terhadap Basis Set")
plt.grid(True)
plt.tight_layout()
plt.savefig("gambar/energi_basis_set.png", dpi=300)
plt.show()
```

Kode tersebut dapat dijalankan ulang selama file `data/energi_basis_set.csv` tersedia. Output tabel dan gambar akan dihasilkan lagi. Ini menunjukkan proses yang jelas.

Kode yang tidak dapat dijalankan ulang sering memiliki ciri-ciri berikut:

```
data2 = data1.copy()
hasil = proses(data2)
```

Jika `data1` dan `proses()` tidak didefinisikan dalam kode, maka kode bergantung pada keadaan tersembunyi. Atau:

```
plt.plot(x, y)
```

Jika `x` dan `y` tidak didefinisikan pada sel sebelumnya yang tersedia, kode tidak reprodusibel.

Dalam praktikum, dosen dapat menerapkan aturan sederhana: **kode dianggap sah jika dapat dijalankan dari awal oleh orang lain tanpa bertanya kepada penulisnya**. Aturan ini membiasakan mahasiswa menulis kode yang jelas dan lengkap.

5. Kode Harus Menghasilkan Output yang Dapat Digunakan dalam Laporan

Kode praktikum harus menghasilkan output yang dapat langsung digunakan atau dirujuk dalam laporan. Output tersebut dapat berupa tabel CSV, grafik PNG/PDF, ringkasan nilai, koordinat final terolah, atau file teks berisi kesimpulan numerik. Jika kode hanya menampilkan angka di layar tanpa menyimpan apa pun, maka hasilnya mudah hilang. Jika grafik hanya muncul di notebook tanpa disimpan, laporan sulit disusun secara rapi. Karena itu, kode harus memiliki bagian ekspor output.

Contoh output yang baik:

```
ringkasan.to_csv("data/ringkasan_energi_reaksi.csv", index=False)
plt.savefig("gambar/diagram_energi_reaksi.png", dpi=300, bbox_inches="tight")
```

Dengan dua baris tersebut, tabel dan grafik tersedia untuk laporan. Mahasiswa dapat memasukkan file gambar ke dokumen laporan dan melampirkan file CSV sebagai data pendukung. Dosen dapat memeriksa apakah gambar sesuai dengan data.

Output laporan harus memiliki standar. Untuk tabel, gunakan format CSV dengan header jelas. Untuk grafik, gunakan resolusi cukup, label sumbu, satuan, dan judul. Untuk ringkasan teks, gunakan kalimat yang menyebut metode dan satuan. Contoh ringkasan yang baik:

```
print(f"Gap HOMO-LUMO benzena = {gap_ev:.2f} eV pada tingkat teori yang digunakan.")
```

Contoh yang kurang baik:

```
print(gap)
```

Output kedua tidak menyebut makna, satuan, atau konteks. Jika dibaca di laporan, angka tersebut tidak informatif.

Untuk praktikum spektrum IR, output yang siap laporan dapat berupa:

```
data_spektrum.to_csv("data/spektrum_ir_formaldehida.csv", index=False)
plt.savefig("gambar/spektrum_ir_formaldehida.png", dpi=300, bbox_inches="tight")
```

Untuk praktikum geometri:

```
hasil_geometri.to_csv("data/geometri_h2o.csv", index=False)
```

Untuk praktikum partikel dalam kotak:

```
hasil_energi.to_csv("data/energi_partikel_dalam_kotak.csv", index=False)
plt.savefig("gambar/fungsi_gelombang_n1_n2_n3.png", dpi=300, bbox_inches="tight")
```

Kode yang menghasilkan output laporan membantu mahasiswa bekerja efisien. Mereka tidak perlu menyalin angka satu per satu. Mereka juga tidak perlu membuat ulang grafik secara manual. Yang lebih penting, output laporan dapat ditelusuri ke kode dan data.

6. Standar Format Kode: Kerapian, Spasi, dan Urutan

Selain lima prinsip utama, kode praktikum perlu mengikuti standar format dasar. Kode harus memiliki indentasi yang benar, baris tidak terlalu panjang, spasi digunakan secara konsisten, dan bagian-bagian kode dipisahkan dengan komentar judul. Python sangat bergantung pada indentasi. Kesalahan indentasi dapat membuat kode gagal atau logika berubah.

Contoh format yang kurang baik:

```
def f(n,L,m):
```

```
 return n**2*h**2/(8*m*L**2)
E=f(1,1e-9,9.109e-31)
print(E)
```

Contoh format yang lebih baik:

```
def energi_partikel_kotak(n, panjang_kotak_meter, massa_partikel_kg):
    """Menghitung energi partikel dalam kotak satu dimensi."""
    return (n**2 * h**2) / (8 * massa_partikel_kg * panjang_kotak_meter**2)


energi_n1_joule = energi_partikel_kotak(
    n=1,
    panjang_kotak_meter=1.0e-9,
    massa_partikel_kg=m_e
)

print(energi_n1_joule)
```

Kode kedua lebih panjang, tetapi jauh lebih mudah dibaca. Parameter fungsi jelas. Nama variabel jelas. Rumus memiliki spasi yang memudahkan pembacaan. Untuk praktikum, keterbacaan lebih penting daripada membuat kode sesingkat mungkin.

Bagian kode juga dapat diberi pemisah:

```
# ============================================================
# 1. Import pustaka
# ============================================================

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Pemisah seperti ini sangat membantu dalam notebook atau script panjang. Mahasiswa dapat langsung menemukan bagian import, konstanta, fungsi, input data, perhitungan, grafik, dan ekspor data.

7. Standar Penanganan Error Sederhana

Kode praktikum tidak harus memiliki sistem error handling yang sangat kompleks, tetapi perlu memiliki pengecekan dasar. Misalnya, sebelum membaca file, kode dapat memeriksa apakah file tersedia. Ini membantu mahasiswa memahami penyebab error.

Contoh:

```
from pathlib import Path
import pandas as pd

file_data = Path("data/energi_basis_set.csv")

if not file_data.exists():
```

```
    raise FileNotFoundError(f"File tidak ditemukan: {file_data}")

data = pd.read_csv(file_data)
```

Kode ini lebih ramah daripada membiarkan error panjang yang membingungkan. Mahasiswa langsung tahu file mana yang hilang. Dalam praktikum, error seperti `FileNotFoundError` sering muncul karena folder salah, nama file salah, atau file belum disimpan.

Untuk validasi kolom data:

```
kolom_wajib = {"molekul", "energi_hartree"}

if not kolom_wajib.issubset(data.columns):
    raise ValueError("Data harus memiliki kolom: molekul dan energi_hartree")
```

Pengecekan seperti ini mencegah kode berjalan dengan data yang salah. Jika file CSV tidak memiliki kolom yang dibutuhkan, program berhenti dengan pesan jelas. Ini penting untuk praktikum yang melibatkan banyak file data.

8. Standar Satuan dan Konversi

Dalam praktikum kimia kuantum, standar satuan harus sangat ketat. Kode harus menyatakan satuan pada variabel, komentar, tabel, dan grafik. Banyak kesalahan interpretasi muncul karena satuan tidak jelas.

Contoh buruk:

```
E = -0.012
print(E)
```

Contoh baik:

```
delta_e_hartree = -0.012
hartree_to_kjmol = 2625.49962

delta_e_kjmol = delta_e_hartree * hartree_to_kjmol

print(f"ΔE = {delta_e_kjmol:.2f} kJ/mol")
```

Dalam tabel:

```
hasil = pd.DataFrame({
    "delta_e_hartree": [delta_e_hartree],
    "delta_e_kjmol": [delta_e_kjmol]
})
```

Dalam grafik:

```
plt.ylabel("Energi relatif (kJ/mol)")
```

Standar ini harus diterapkan secara konsisten. Jika satuan tidak ditulis, grafik dan tabel tidak boleh dianggap lengkap. Dalam kimia kuantum, satuan energi, panjang, frekuensi, dan momen dipol sangat menentukan interpretasi.

9. Standar Reprodusibilitas Notebook

Jupyter Notebook sangat membantu pembelajaran, tetapi rawan masalah jika tidak dikelola dengan baik. Notebook dapat menyimpan output lama, variabel tersembunyi, atau urutan eksekusi yang tidak konsisten. Oleh karena itu, standar notebook harus ditegakkan.

Notebook praktikum harus:

1. Memiliki judul praktikum.
2. Memiliki tujuan singkat.
3. Memiliki bagian import pustaka.
4. Memiliki bagian konstanta dan fungsi.
5. Membaca data dari folder yang jelas.
6. Menjalankan perhitungan secara berurutan.
7. Menampilkan tabel dan grafik.
8. Mengekspor hasil.
9. Menyertakan kesimpulan numerik.
10. Dijalankan ulang dari awal sebelum dikumpulkan.

Jika notebook berisi sel yang tidak dipakai, hapus atau beri keterangan. Jika ada eksperimen gagal, boleh dipertahankan jika diberi catatan. Jika notebook terlalu acak, pisahkan menjadi versi final yang rapi. Dalam laporan, notebook dapat menjadi lampiran atau sumber analisis, tetapi laporan final tetap harus menyajikan narasi ilmiah yang utuh.

10. Contoh Kode Buruk dan Perbaikannya

Berikut contoh kode yang kurang memenuhi standar:

```
import pandas as pd
import matplotlib.pyplot as plt

d = pd.read_csv("C:/Users/Admin/Desktop/x.csv")
d["z"] = d["b"] - d["a"]
print(d)
plt.bar(d["m"], d["z"])
plt.show()
```

Masalah kode tersebut:

1. Jalur file absolut dan hanya berlaku di komputer tertentu.
2. Nama variabel `d`, `z`, `b`, `a`, dan `m` tidak jelas.
3. Tidak ada satuan.
4. Tidak ada komentar.

5. Grafik tidak memiliki label sumbu.
6. Hasil tidak diekspor.
7. Tidak jelas apakah data HOMO-LUMO, energi reaksi, atau data lain.

Perbaikan:

```
# ============================================================
# Analisis Gap HOMO-LUMO
# ============================================================

import pandas as pd
import matplotlib.pyplot as plt

# Membaca data energi HOMO dan LUMO dalam satuan eV
data = pd.read_csv("data/homo_lumo.csv")

# Menghitung gap HOMO-LUMO: gap = energi LUMO - energi HOMO
data["gap_ev"] = data["lumo_ev"] - data["homo_ev"]

# Menyimpan tabel hasil untuk laporan
data.to_csv("data/homo_lumo_hasil.csv", index=False)

# Membuat grafik gap HOMO-LUMO
plt.figure()
plt.bar(data["molekul"], data["gap_ev"])
plt.xlabel("Molekul")
plt.ylabel("Gap HOMO-LUMO (eV)")
plt.title("Perbandingan Gap HOMO-LUMO")
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig("gambar/gap_homo_lumo.png", dpi=300)
plt.show()

# Kesimpulan numerik
molekul_gap_min = data.loc[data["gap_ev"].idxmin(), "molekul"]
gap_min = data["gap_ev"].min()

print(f"Gap HOMO-LUMO terkecil adalah {molekul_gap_min} sebesar {gap_min:.2f}
eV.")
```

Kode perbaikan ini jauh lebih layak untuk praktikum. Jalur file relatif. Nama variabel jelas. Satuan tertulis. Ada komentar. Grafik memiliki label. Hasil diekspor. Kesimpulan numerik ditampilkan.

11. Standar Etika dalam Penulisan Kode

Penulisan kode juga memiliki dimensi etika akademik. Mahasiswa tidak boleh menyalin kode dari teman, internet, AI, atau sumber lain tanpa memahami dan menyebut sumber jika diminta oleh aturan kelas. Penggunaan template dari buku ini diperbolehkan karena memang disediakan sebagai bahan praktikum. Namun, jika mahasiswa memodifikasi kode dari dokumentasi, repositori, atau artikel, sumbernya harus disebut. Kode adalah karya intelektual.

Mahasiswa juga tidak boleh mengubah kode hanya agar hasil tampak sesuai harapan. Misalnya, mengalikan hasil dengan faktor tertentu tanpa alasan ilmiah agar energi mendekati literatur, menghapus data yang tidak sesuai tanpa keterangan, atau mengganti nilai output secara manual. Jika ada data menyimpang, tuliskan dan analisis penyebabnya. Etika kode sama pentingnya dengan etika data.

Dalam praktikum kelompok, pembagian kerja kode harus jelas. Jika satu anggota menulis script Python, anggota lain tetap harus memahami cara kerja kode. Laporan kelompok tidak boleh menjadi kumpulan hasil dari satu orang tanpa pemahaman bersama. Kode yang baik membantu pembagian pemahaman karena strukturnya jelas dan komentarnya memadai.

12. Standar Minimal Kode yang Layak Dikumpulkan

Dalam buku ini, kode praktikum dianggap layak dikumpulkan jika memenuhi kriteria berikut:

Aspek	**Standar Minimal**
Import	Semua pustaka ditulis di awal
Konstanta	Nilai penting diberi nama dan satuan
Variabel	Nama jelas, tidak samar
Fungsi	Fungsi penting memiliki docstring atau komentar
Data	File input jelas dan tersedia
Jalur file	Menggunakan jalur relatif
Perhitungan	Rumus tampak eksplisit
Tabel	Hasil disusun dalam tabel jika relevan
Grafik	Memiliki label sumbu, satuan, dan judul
Ekspor	Data/gambar disimpan ke folder sesuai standar
Re-run	Kode dapat dijalankan dari awal
Kesimpulan	Ada ringkasan numerik dengan satuan

Kriteria ini dapat digunakan dosen sebagai rubrik penilaian kode. Mahasiswa tidak hanya dinilai dari benar-salah hasil, tetapi juga dari kualitas proses komputasi. Kode yang menghasilkan angka benar tetapi tidak dapat dijalankan ulang sebaiknya tidak diberi nilai penuh. Sebaliknya, kode yang rapi tetapi hasilnya belum sempurna dapat diberi umpan balik yang lebih mudah karena prosesnya jelas.

13. Implikasi terhadap Pembelajaran Kimia Kuantum

Standar penulisan kode memiliki implikasi langsung terhadap pembelajaran kimia kuantum. Pertama, mahasiswa belajar menulis kode sebagai bahasa ilmiah. Kode bukan hanya perintah komputer, tetapi representasi prosedur ilmiah. Kedua, mahasiswa belajar bahwa hasil numerik harus memiliki konteks. Energi, panjang ikatan, frekuensi, dan gap orbital tidak bermakna tanpa satuan, metode, dan data sumber. Ketiga, mahasiswa belajar disiplin reprodusibilitas. Keempat, mahasiswa belajar berpikir modular melalui fungsi. Kelima, mahasiswa belajar menyiapkan output laporan secara efisien.

Dalam praktikum partikel dalam kotak, standar kode membantu mahasiswa mengaitkan rumus energi dengan grafik fungsi gelombang. Dalam praktikum optimasi geometri, standar kode membantu mahasiswa menghitung parameter dari koordinat final. Dalam praktikum HOMO-LUMO, standar kode membantu mahasiswa membuat tabel dan diagram energi. Dalam praktikum IR, standar kode membantu mahasiswa membuat spektrum dari data frekuensi. Dalam praktikum energi reaksi, standar kode membantu mahasiswa menghindari kesalahan stoikiometri dan satuan.

Dengan demikian, standar penulisan kode bukan bagian teknis yang terpisah dari kimia. Ia justru memperkuat pemahaman kimia karena membuat perhitungan lebih jelas dan dapat diuji.

14. Sintesis Subbab

Standar penulisan kode dalam praktikum kimia kuantum bertujuan membentuk kebiasaan komputasi ilmiah yang benar. Kode harus memiliki komentar, nama variabel yang jelas, tidak bergantung pada file tersembunyi, dapat dijalankan ulang, dan menghasilkan output yang dapat digunakan dalam laporan. Prinsip-prinsip tersebut membuat kode lebih mudah dibaca, lebih mudah diperiksa, lebih mudah diperbaiki, dan lebih kuat secara akademik.

Kode yang baik tidak harus rumit. Kode yang baik adalah kode yang jelas. Ia menunjukkan data apa yang digunakan, rumus apa yang dipakai, satuan apa yang berlaku, grafik apa yang dibuat, file apa yang disimpan, dan kesimpulan apa yang diperoleh. Dalam praktikum kimia kuantum, kode semacam ini menjadi bagian dari integritas ilmiah. Mahasiswa yang mampu menulis kode rapi akan lebih siap melakukan praktikum, menyusun laporan, dan melanjutkan ke penelitian komputasi yang lebih serius.

Subbab berikutnya akan menyediakan template kode umum praktikum. Template tersebut akan merangkum struktur dan standar penulisan kode yang telah dibahas pada subbab 3.3 dan 3.4 agar dapat digunakan ulang pada semua mata praktikum dalam buku ini.

3.5 Template Kode Umum Praktikum

Template kode umum praktikum adalah kerangka dasar penulisan kode Python yang digunakan secara berulang dalam seluruh mata praktikum kimia kuantum pada buku ini. Template ini berfungsi sebagai pola kerja standar agar setiap kode praktikum memiliki susunan yang konsisten, mudah dibaca, mudah dijalankan ulang, mudah diperiksa, dan menghasilkan output yang dapat dipakai dalam laporan. Dalam konteks praktikum kimia kuantum, template kode bukan sekadar contoh program, melainkan **kerangka metodologi komputasi** yang menghubungkan teori, data, perhitungan, grafik, dan kesimpulan numerik.

Template umum diperlukan karena setiap praktikum dalam buku ini memiliki pola kerja yang mirip meskipun topik kimianya berbeda. Pada praktikum partikel dalam kotak, mahasiswa menghitung energi dan fungsi gelombang. Pada praktikum optimasi geometri, mahasiswa membaca koordinat dan menghitung panjang ikatan. Pada praktikum Hartree–Fock dan basis set, mahasiswa mengolah tabel energi. Pada praktikum DFT dan HOMO-LUMO, mahasiswa menghitung gap orbital. Pada praktikum IR, mahasiswa membuat spektrum dari frekuensi dan intensitas. Pada praktikum energi reaksi, mahasiswa menghitung ΔE, ΔH, atau ΔG dari energi

reaktan dan produk. Semua tugas tersebut membutuhkan alur yang serupa: pustaka diimpor, konstanta didefinisikan, fungsi disiapkan, data dibaca, perhitungan dilakukan, hasil dibuat dalam tabel, grafik disimpan, dan kesimpulan numerik ditulis.

Secara konseptual, template kode praktikum memiliki tiga fungsi. Pertama, **fungsi pedagogis**, yaitu membantu mahasiswa belajar menulis kode secara bertahap tanpa harus memulai dari halaman kosong. Kedua, **fungsi metodologis**, yaitu memastikan setiap praktikum memiliki struktur analisis yang sama sehingga hasilnya mudah diperiksa. Ketiga, **fungsi reprodusibilitas**, yaitu membuat kode, data, grafik, dan output dapat dijalankan ulang oleh dosen atau mahasiswa lain. Dengan demikian, template kode umum adalah jembatan antara pembelajaran Python pada Bab 3 dan penerapan praktikum kimia kuantum pada bab-bab berikutnya.

Template ini juga relevan dengan karakter Jupyter Notebook sebagai dokumen komputasional interaktif. Project Jupyter menjelaskan bahwa Notebook menggabungkan kode hidup, persamaan, teks naratif, dan visualisasi dalam satu dokumen, sehingga cocok untuk kegiatan praktikum yang memerlukan penjelasan, kode, hasil, dan grafik secara terpadu. (jupyter.org) Namun, template ini tidak hanya berlaku untuk Jupyter Notebook. Template yang sama juga dapat digunakan dalam file `.py` di VS Code atau editor Python lain. Perbedaannya hanya pada cara penyajian: Jupyter cocok untuk laporan komputasional interaktif, sedangkan file `.py` cocok untuk script analisis yang lebih ringkas dan otomatis.

1. Prinsip Dasar Template Kode Praktikum

Template kode umum dalam buku ini dibangun atas prinsip bahwa setiap kode praktikum harus dapat menjawab lima pertanyaan ilmiah:

1. **Data apa yang digunakan?**
2. **Rumus atau metode apa yang diterapkan?**
3. **Perhitungan apa yang dilakukan?**
4. **Hasil apa yang diperoleh?**
5. **Bagaimana hasil disimpan dan digunakan dalam laporan?**

Jika kode tidak dapat menjawab pertanyaan tersebut, maka kode belum layak menjadi bagian dari praktikum ilmiah. Kode yang hanya menampilkan angka tanpa sumber data, tanpa satuan, tanpa fungsi, tanpa grafik, dan tanpa ekspor hasil tidak cukup kuat untuk praktikum kimia kuantum. Sebaliknya, kode yang menyatakan data, rumus, satuan, hasil, grafik, dan file output secara jelas dapat menjadi bukti analisis ilmiah.

Template umum juga harus mengikuti struktur folder yang telah ditetapkan pada Bab 2:

```
praktikum_nama_topik/
├── input/
├── output/
├── python/
├── gambar/
├── laporan/
└── data/
```

Dalam template, folder `data/` digunakan untuk membaca dan menyimpan tabel CSV atau koordinat terolah. Folder `gambar/` digunakan untuk menyimpan grafik. Folder `python/` digunakan untuk menyimpan script. Folder `output/` digunakan untuk menyimpan file hasil ChemCompute, ORCA, GAMESS, Gaussian, Psi4, PySCF, atau software lain. Folder `laporan/` digunakan untuk dokumen akhir. Dengan demikian, template kode harus selaras dengan standar file praktikum.

Untuk menjaga agar kode dapat bekerja lintas sistem operasi, template sebaiknya menggunakan `pathlib`. Dokumentasi resmi Python menjelaskan bahwa `pathlib` menyediakan kelas untuk merepresentasikan jalur sistem file dengan semantik yang sesuai untuk berbagai sistem operasi. (Python documentation) Ini penting karena jalur Windows memakai tanda `\`, sedangkan Linux dan macOS memakai `/`. Dengan `pathlib`, kode lebih mudah dipindahkan dari Windows ke Linux, Google Colab, atau sebaliknya.

2. Template Kode Umum Berbasis Script `.py`

Berikut adalah template kode umum yang dapat digunakan sebagai dasar semua praktikum berbasis file `.py`.

```
# ============================================================
# PRAKTIKUM KIMIA KUANTUM
# Judul Praktikum : [tulis judul praktikum]
# Nama File       : [tulis nama file kode]
# Tujuan          : [tulis tujuan singkat praktikum]
# ============================================================

# ============================================================
# 1. Import pustaka
# ============================================================

from pathlib import Path

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


# ============================================================
# 2. Definisi folder kerja
# ============================================================

BASE_DIR = Path(__file__).resolve().parent.parent

DATA_DIR = BASE_DIR / "data"
GAMBAR_DIR = BASE_DIR / "gambar"
OUTPUT_DIR = BASE_DIR / "output"

DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)


# ============================================================
# 3. Definisi konstanta
```

```
# ============================================================

HARTREE_TO_KJMOL = 2625.49962
EV_TO_JOULE = 1.602176634e-19
PLANCK = 6.62607015e-34
MASSA_ELEKTRON = 9.1093837015e-31


# ============================================================
# 4. Definisi fungsi
# ============================================================

def contoh_fungsi(nilai):
    """
    Fungsi contoh.

    Parameter:
    nilai : float
        Nilai input numerik.

    Return:
    float
        Nilai hasil perhitungan.
    """
    return nilai


# ============================================================
# 5. Input data
# ============================================================

# Contoh membaca data CSV:
# data = pd.read_csv(DATA_DIR / "nama_file_data.csv")

# Untuk template awal, gunakan data contoh.
data = pd.DataFrame({
    "parameter": ["contoh"],
    "nilai": [1.0]
})


# ============================================================
# 6. Perhitungan
# ============================================================

data["hasil"] = data["nilai"].apply(contoh_fungsi)


# ============================================================
# 7. Tabel hasil
# ============================================================

print("\nTabel hasil:")
print(data)


# ============================================================
```

```
# 8. Grafik
# ============================================================

plt.figure()
plt.bar(data["parameter"], data["hasil"])
plt.xlabel("Parameter")
plt.ylabel("Hasil")
plt.title("Grafik Hasil Praktikum")
plt.tight_layout()


# ============================================================
# 9. Ekspor data dan gambar
# ============================================================

data.to_csv(DATA_DIR / "hasil_praktikum.csv", index=False)
plt.savefig(GAMBAR_DIR / "grafik_hasil_praktikum.png", dpi=300,
bbox_inches="tight")
plt.show()


# ============================================================
# 10. Kesimpulan numerik
# ============================================================

print("\nKesimpulan numerik:")
print("Kode berhasil dijalankan dan menghasilkan tabel serta grafik.")
```

Template ini sengaja dibuat umum agar dapat disesuaikan untuk semua praktikum. Bagian `Import pustaka` memanggil paket yang diperlukan. Bagian `Definisi folder kerja` menyiapkan jalur file. Bagian `Definisi konstanta` menyimpan konstanta fisika dan faktor konversi. Bagian `Definisi fungsi` memuat rumus atau prosedur. Bagian `Input data` membaca data. Bagian `Perhitungan` melakukan analisis. Bagian `Tabel hasil` menampilkan data. Bagian `Grafik` membuat visualisasi. Bagian `Ekspor data dan gambar` menyimpan output. Bagian `Kesimpulan numerik` menampilkan ringkasan hasil.

Pandas `DataFrame.to_csv()` digunakan untuk menulis objek DataFrame ke file CSV, sehingga hasil analisis dapat disimpan sebagai data terstruktur yang dapat dibaca ulang oleh Python, spreadsheet, atau dosen. (Pandas) Matplotlib `savefig()` digunakan untuk menyimpan figure saat ini sebagai file gambar atau grafik vektor. (Matplotlib) Dua fungsi ini sangat penting dalam template karena laporan praktikum membutuhkan tabel dan gambar yang dapat ditelusuri ke kode.

3. Template Kode Umum Berbasis Jupyter Notebook

Untuk Jupyter Notebook, template tidak selalu harus menggunakan `__file__` karena notebook tidak selalu memiliki variabel tersebut. Template notebook lebih baik menggunakan jalur relatif sederhana dari folder praktikum.

Struktur notebook yang disarankan:

```
Judul Praktikum
```

```
Tujuan Praktikum
Dasar Teori Singkat
Import Pustaka
Definisi Konstanta
Definisi Fungsi
Input Data
Perhitungan
Tabel Hasil
Grafik
Ekspor Data
Kesimpulan Numerik
```

Jupyter mendukung Markdown cell untuk menambahkan teks naratif dalam notebook; dokumentasi Jupyter menjelaskan bahwa teks dapat ditambahkan menggunakan Markdown cell. (jupyter-notebook.readthedocs.io) Karena itu, bagian judul, tujuan, dasar teori, dan interpretasi hasil sebaiknya ditulis sebagai Markdown, sedangkan perhitungan ditulis sebagai code cell. Pola ini menjadikan notebook bukan sekadar kumpulan kode, tetapi dokumen praktikum komputasional.

Contoh bagian awal notebook:

```
# Praktikum Kimia Kuantum
## Judul: Analisis Energi Partikel dalam Kotak

### Tujuan
1. Menghitung energi partikel dalam kotak satu dimensi.
2. Membuat grafik fungsi gelombang.
3. Menyimpan tabel energi dan gambar grafik untuk laporan.
```

Contoh code cell:

```
from pathlib import Path

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

DATA_DIR = Path("data")
GAMBAR_DIR = Path("gambar")

DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)
```

Template notebook harus dijalankan dari atas ke bawah. Inilah salah satu disiplin terpenting dalam penggunaan Jupyter. Notebook yang dijalankan tidak berurutan dapat menyimpan variabel lama di memori, sehingga hasil yang tampil belum tentu berasal dari kode final. Kajian terhadap notebook Jupyter menunjukkan bahwa sifat notebook yang terdiri dari sel-sel kode dan teks dapat menyebabkan efek samping ketika sel dijalankan dalam urutan berbeda, sehingga struktur notebook perlu dikelola dengan baik agar hasil dapat dipercaya. (arXiv)

4. Template untuk Praktikum Partikel dalam Kotak

Template pertama yang langsung relevan dengan Bab 4 adalah template partikel dalam kotak. Template ini digunakan untuk menghitung energi, fungsi gelombang, dan rapat probabilitas.

```
# ============================================================
# Praktikum 1: Partikel dalam Kotak Satu Dimensi
# ============================================================

from pathlib import Path

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Folder kerja
DATA_DIR = Path("data")
GAMBAR_DIR = Path("gambar")
DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)

# Konstanta
h = 6.62607015e-34       # J s
m_e = 9.1093837015e-31   # kg
joule_to_ev = 1 / 1.602176634e-19

# Parameter model
panjang_kotak_meter = 1.0e-9
n_values = np.arange(1, 6)

# Fungsi
def energi_partikel_kotak(n, panjang_kotak, massa):
    """Menghitung energi partikel dalam kotak satu dimensi dalam joule."""
    return (n**2 * h**2) / (8 * massa * panjang_kotak**2)


def fungsi_gelombang(n, x, panjang_kotak):
    """Menghitung fungsi gelombang ternormalisasi."""
    return np.sqrt(2 / panjang_kotak) * np.sin(n * np.pi * x / panjang_kotak)


# Perhitungan energi
energi_joule = energi_partikel_kotak(n_values, panjang_kotak_meter, m_e)
energi_ev = energi_joule * joule_to_ev

hasil_energi = pd.DataFrame({
    "n": n_values,
    "energi_joule": energi_joule,
    "energi_ev": energi_ev
})

print(hasil_energi)

# Grafik fungsi gelombang
x = np.linspace(0, panjang_kotak_meter, 1000)
```

```
plt.figure()
for n in [1, 2, 3]:
    psi = fungsi_gelombang(n, x, panjang_kotak_meter)
    plt.plot(x * 1e9, psi, label=f"n = {n}")

plt.xlabel("x (nm)")
plt.ylabel("ψ(x)")
plt.title("Fungsi Gelombang Partikel dalam Kotak")
plt.legend()
plt.grid(True)
plt.tight_layout()

# Ekspor
hasil_energi.to_csv(DATA_DIR / "energi_partikel_dalam_kotak.csv", index=False)
plt.savefig(GAMBAR_DIR / "fungsi_gelombang_partikel_dalam_kotak.png", dpi=300,
bbox_inches="tight")
plt.show()

# Kesimpulan numerik
print(f"Energi keadaan dasar n = 1 adalah {energi_ev[0]:.4f} eV.")
print("Energi meningkat sebanding dengan n² untuk panjang kotak yang tetap.")
```

Template ini mencerminkan struktur yang akan dipakai pada praktikum pertama. Mahasiswa dapat mengubah panjang kotak, massa partikel, atau rentang nilai n. Dengan perubahan sederhana, mereka dapat mengeksplorasi hubungan antara panjang kotak dan energi. Secara pedagogis, template ini membantu mahasiswa melihat hubungan antara rumus kuantum, kode Python, tabel energi, dan grafik fungsi gelombang.

5. Template untuk Analisis Geometri Molekul

Template kedua digunakan untuk membaca koordinat molekul dan menghitung panjang ikatan serta sudut ikatan. Template ini akan berguna pada praktikum optimasi geometri.

```
# ============================================================
# Template Analisis Geometri Molekul
# ============================================================

from pathlib import Path

import numpy as np
import pandas as pd

DATA_DIR = Path("data")
DATA_DIR.mkdir(exist_ok=True)

def hitung_jarak(atom_A, atom_B):
    """Menghitung jarak antara dua atom berdasarkan koordinat 3D."""
    return np.linalg.norm(atom_B - atom_A)


def hitung_sudut(atom_A, atom_B, atom_C):
    """
    Menghitung sudut A-B-C dalam derajat.
```

```
    Atom B adalah atom pusat.
    """
    v1 = atom_A - atom_B
    v2 = atom_C - atom_B

    cos_theta = np.dot(v1, v2) / (np.linalg.norm(v1) * np.linalg.norm(v2))
    cos_theta = np.clip(cos_theta, -1.0, 1.0)

    sudut_rad = np.arccos(cos_theta)
    return np.degrees(sudut_rad)


# Contoh koordinat H2O hasil optimasi, satuan Å
O = np.array([0.000000, 0.000000, 0.000000])
H1 = np.array([0.758602, 0.000000, 0.504284])
H2 = np.array([-0.758602, 0.000000, 0.504284])

jarak_oh1 = hitung_jarak(O, H1)
jarak_oh2 = hitung_jarak(O, H2)
sudut_hoh = hitung_sudut(H1, O, H2)

hasil = pd.DataFrame({
    "parameter": ["O-H1", "O-H2", "H-O-H"],
    "nilai": [jarak_oh1, jarak_oh2, sudut_hoh],
    "satuan": ["Å", "Å", "derajat"]
})

print(hasil)

hasil.to_csv(DATA_DIR / "geometri_h2o.csv", index=False)

print(f"Sudut H-O-H = {sudut_hoh:.2f} derajat.")
```

Template ini menunjukkan bahwa geometri molekul dapat dianalisis dari koordinat Cartesian. Mahasiswa belajar bahwa panjang ikatan bukan angka misterius dari software, melainkan hasil perhitungan jarak antara dua titik dalam ruang tiga dimensi. Sudut ikatan juga berasal dari operasi vektor. Dengan demikian, Python membantu menghubungkan output koordinat dengan konsep geometri molekul.

6. Template untuk Analisis HOMO-LUMO

Template ketiga digunakan untuk menghitung gap HOMO-LUMO dari tabel energi orbital. Template ini akan digunakan pada praktikum DFT dan reaktivitas molekul.

```
# ============================================================
# Template Analisis HOMO-LUMO
# ============================================================

from pathlib import Path

import pandas as pd
import matplotlib.pyplot as plt

DATA_DIR = Path("data")
```

```
GAMBAR_DIR = Path("gambar")
DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)

# Contoh data energi orbital dalam eV
data = pd.DataFrame({
    "molekul": ["etilena", "butadiena", "benzena"],
    "homo_ev": [-10.5, -8.9, -9.2],
    "lumo_ev": [1.2, 0.3, 0.8]
})

data["gap_ev"] = data["lumo_ev"] - data["homo_ev"]

print(data)

plt.figure()
plt.bar(data["molekul"], data["gap_ev"])
plt.xlabel("Molekul")
plt.ylabel("Gap HOMO-LUMO (eV)")
plt.title("Perbandingan Gap HOMO-LUMO")
plt.grid(axis="y")
plt.tight_layout()

data.to_csv(DATA_DIR / "homo_lumo_hasil.csv", index=False)
plt.savefig(GAMBAR_DIR / "gap_homo_lumo.png", dpi=300, bbox_inches="tight")
plt.show()

molekul_gap_min = data.loc[data["gap_ev"].idxmin(), "molekul"]
gap_min = data["gap_ev"].min()

print(f"Gap HOMO-LUMO terkecil adalah {molekul_gap_min}, yaitu {gap_min:.2f}
eV.")
```

Template ini memperlihatkan pola analisis data elektronik sederhana. Mahasiswa dapat mengganti data contoh dengan hasil output ChemCompute, ORCA, GAMESS, Gaussian, Psi4, atau PySCF. Hal yang penting adalah kolom energi HOMO dan LUMO harus memiliki satuan yang sama. Jika energi berasal dari output dalam Hartree, maka perlu dikonversi ke eV terlebih dahulu atau gap dihitung dalam Hartree lalu dikonversi. Template ini membantu mencegah kesalahan karena nama kolom sudah menyertakan satuan.

7. Template untuk Spektrum IR Simulasi

Template keempat digunakan untuk membuat spektrum IR simulasi dari frekuensi dan intensitas. Template ini akan dipakai pada praktikum frekuensi vibrasi.

```
# ============================================================
# Template Spektrum IR Simulasi
# ============================================================

from pathlib import Path

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
DATA_DIR = Path("data")
GAMBAR_DIR = Path("gambar")
DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)

# Contoh data frekuensi dan intensitas
data_ir = pd.DataFrame({
    "frekuensi_cm1": [1100, 1500, 1720, 2900],
    "intensitas": [20, 35, 100, 40]
})

def gaussian(x, pusat, tinggi, lebar):
    """Membuat puncak Gaussian untuk simulasi spektrum."""
    return tinggi * np.exp(-0.5 * ((x - pusat) / lebar)**2)


x = np.linspace(400, 4000, 4000)
spektrum = np.zeros_like(x)

lebar_puncak = 20.0

for _, baris in data_ir.iterrows():
    spektrum += gaussian(
        x,
        pusat=baris["frekuensi_cm1"],
        tinggi=baris["intensitas"],
        lebar=lebar_puncak
    )

plt.figure()
plt.plot(x, spektrum)
plt.xlabel("Bilangan gelombang (cm⁻¹)")
plt.ylabel("Intensitas relatif")
plt.title("Spektrum IR Simulasi")
plt.gca().invert_xaxis()
plt.tight_layout()

data_ir.to_csv(DATA_DIR / "data_ir.csv", index=False)
plt.savefig(GAMBAR_DIR / "spektrum_ir_simulasi.png", dpi=300,
bbox_inches="tight")
plt.show()

print("Spektrum IR simulasi berhasil dibuat dari data frekuensi dan
intensitas.")
```

Template ini mengajarkan bahwa spektrum IR simulasi dapat dibangun dari daftar puncak. Fungsi Gaussian digunakan untuk memberi pelebaran pada setiap puncak sehingga spektrum tampak kontinu. Mahasiswa perlu memahami bahwa grafik ini adalah simulasi visual, bukan output mentah langsung. Lebar puncak, intensitas, dan faktor skala harus dicatat jika digunakan dalam laporan.

8. Template untuk Energi Reaksi

Template kelima digunakan untuk menghitung energi reaksi dari tabel spesies, koefisien stoikiometri, dan energi elektronik. Template ini akan digunakan pada praktikum energi reaksi dan termodinamika.

```
# ============================================================
# Template Energi Reaksi
# ============================================================

from pathlib import Path

import pandas as pd
import matplotlib.pyplot as plt

DATA_DIR = Path("data")
GAMBAR_DIR = Path("gambar")
DATA_DIR.mkdir(exist_ok=True)
GAMBAR_DIR.mkdir(exist_ok=True)

HARTREE_TO_KJMOL = 2625.49962

# Contoh data
# Koefisien negatif untuk reaktan, positif untuk produk
data = pd.DataFrame({
    "spesies": ["reaktan_A", "reaktan_B", "produk_C"],
    "koefisien": [-1, -1, 1],
    "energi_hartree": [-154.123456, -40.123456, -194.300000]
})

data["kontribusi_hartree"] = data["koefisien"] * data["energi_hartree"]

delta_e_hartree = data["kontribusi_hartree"].sum()
delta_e_kjmol = delta_e_hartree * HARTREE_TO_KJMOL

ringkasan = pd.DataFrame({
    "parameter": ["delta_e_hartree", "delta_e_kjmol"],
    "nilai": [delta_e_hartree, delta_e_kjmol]
})

print(data)
print(ringkasan)

plt.figure()
plt.bar(["ΔE"], [delta_e_kjmol])
plt.axhline(0)
plt.ylabel("Energi reaksi (kJ/mol)")
plt.title("Energi Reaksi")
plt.tight_layout()

data.to_csv(DATA_DIR / "energi_spesies.csv", index=False)
ringkasan.to_csv(DATA_DIR / "ringkasan_energi_reaksi.csv", index=False)
plt.savefig(GAMBAR_DIR      /      "diagram_energi_reaksi.png",      dpi=300,
bbox_inches="tight")
plt.show()
```

```
if delta_e_kjmol < 0:
    sifat = "produk lebih rendah energi daripada reaktan"
else:
    sifat = "produk lebih tinggi energi daripada reaktan"

print(f"ΔE = {delta_e_kjmol:.2f} kJ/mol; {sifat} pada tingkat teori yang digunakan.")
```

Template ini penting karena perhitungan energi reaksi rawan kesalahan stoikiometri. Dengan kolom `koefisien`, mahasiswa secara eksplisit menulis mana reaktan dan mana produk. Koefisien negatif digunakan untuk reaktan, koefisien positif untuk produk. Perhitungan ΔE kemudian menjadi jumlah koefisien dikalikan energi spesies. Pola ini dapat diperluas untuk reaksi yang lebih kompleks.

9. Template Metadata Praktikum

Selain kode utama, setiap praktikum sebaiknya memiliki metadata. Metadata dapat dibuat langsung dengan Python dan disimpan sebagai file teks sederhana. Metadata membantu dosen mengetahui konteks praktikum tanpa harus membaca seluruh kode.

```
# ============================================================
# Template Metadata Praktikum
# ============================================================

from pathlib import Path

DATA_DIR = Path("data")
DATA_DIR.mkdir(exist_ok=True)

metadata = """
Nama praktikum     : [tulis nama praktikum]
Molekul            : [tulis molekul]
Platform           : ChemCompute / lokal / Google Colab / lainnya
Software           : GAMESS / ORCA / Gaussian / Psi4 / PySCF / lainnya
Metode             : [contoh HF, B3LYP, PM3]
Basis set          : [contoh STO-3G, 6-31G(d), def2-SVP]
Charge             : [tulis charge]
Multiplicity       : [tulis multiplicity]
File input         : input/[nama_file]
File output        : output/[nama_file]
Script Python      : python/[nama_file]
Status validasi    : [konvergen / belum konvergen / perlu dicek]
Catatan            : [tulis catatan penting]
"""

with open(DATA_DIR / "metadata_praktikum.txt", "w", encoding="utf-8") as file:
    file.write(metadata)

print("Metadata praktikum berhasil disimpan.")
```

Template metadata ini mendukung reprodusibilitas. Jika mahasiswa menggunakan ChemCompute, metadata harus menyebut platform, paket, metode, basis set, charge, multiplicity,

dan file output. Jika mahasiswa menggunakan Python murni, metadata harus menyebut parameter model dan paket Python utama. Dengan metadata, laporan praktikum lebih mudah diperiksa dan diarsipkan.

10. Template Fungsi Utilitas Umum

Dalam praktikum yang lebih panjang, beberapa fungsi akan sering dipakai ulang. Misalnya fungsi konversi energi, fungsi membaca CSV, fungsi menyimpan grafik, atau fungsi menghitung jarak. Fungsi-fungsi ini dapat disimpan dalam file utilitas, misalnya `python/fungsi_umum.py`.

Contoh:

```
# ============================================================
# fungsi_umum.py
# Kumpulan fungsi umum untuk praktikum kimia kuantum
# ============================================================

import numpy as np

HARTREE_TO_KJMOL = 2625.49962
HARTREE_TO_EV = 27.211386245988


def hartree_ke_kjmol(energi_hartree):
    """Mengonversi energi dari Hartree ke kJ/mol."""
    return energi_hartree * HARTREE_TO_KJMOL


def hartree_ke_ev(energi_hartree):
    """Mengonversi energi dari Hartree ke eV."""
    return energi_hartree * HARTREE_TO_EV


def hitung_jarak(atom_A, atom_B):
    """Menghitung jarak dua atom dari koordinat 3D."""
    return np.linalg.norm(atom_B - atom_A)


def hitung_sudut(atom_A, atom_B, atom_C):
    """
    Menghitung sudut A-B-C dalam derajat.
    Atom B adalah atom pusat.
    """
    v1 = atom_A - atom_B
    v2 = atom_C - atom_B

    cos_theta = np.dot(v1, v2) / (np.linalg.norm(v1) * np.linalg.norm(v2))
    cos_theta = np.clip(cos_theta, -1.0, 1.0)

    return np.degrees(np.arccos(cos_theta))
```

Kemudian fungsi ini dapat digunakan dalam script lain:

```
from fungsi_umum import hartree_ke_kjmol, hitung_jarak, hitung_sudut
```

Namun, penggunaan file utilitas harus hati-hati. Jika script bergantung pada `fungsi_umum.py`, file tersebut wajib disertakan dalam folder `python/`. Jika tidak disertakan, kode tidak dapat dijalankan ulang. Prinsip Bab 3.4 tetap berlaku: kode tidak boleh bergantung pada file tersembunyi.

11. Analisis Kritis terhadap Template

Template sangat membantu, tetapi memiliki risiko jika digunakan secara mekanis. Mahasiswa dapat menyalin template tanpa memahami isi kode. Jika ini terjadi, template berubah menjadi bentuk baru dari "praktikum klik", hanya saja klik diganti dengan salin-tempel kode. Oleh karena itu, template harus dipahami sebagai **kerangka kerja yang harus dimodifikasi secara sadar**, bukan jawaban final untuk semua masalah.

Setiap kali menggunakan template, mahasiswa harus bertanya:

1. Apakah data input sudah sesuai praktikum?
2. Apakah satuan sudah benar?
3. Apakah fungsi yang digunakan sesuai teori?
4. Apakah grafik menjawab pertanyaan praktikum?
5. Apakah hasil sudah diekspor ke folder yang benar?
6. Apakah kesimpulan numerik tidak berlebihan?
7. Apakah kode dapat dijalankan ulang dari awal?

Template yang baik tidak menghilangkan kebutuhan berpikir. Template justru memberi struktur agar energi berpikir mahasiswa tidak habis pada format dasar, tetapi dapat difokuskan pada pemahaman kimia. Dalam hal ini, template berfungsi seperti format laporan praktikum: ia memberi kerangka, tetapi isi ilmiahnya tetap harus dibangun oleh mahasiswa.

Risiko kedua adalah template menjadi terlalu umum sehingga tidak cukup menjawab kebutuhan spesifik praktikum. Misalnya, template umum grafik tidak selalu cocok untuk spektrum IR, karena spektrum IR lazim ditampilkan dengan sumbu bilangan gelombang yang menurun dari kiri ke kanan. Template energi reaksi tidak langsung cukup untuk termodinamika jika mahasiswa perlu memasukkan koreksi entalpi, entropi, dan energi bebas Gibbs. Karena itu, template umum harus selalu disesuaikan dengan karakter topik.

Risiko ketiga adalah mahasiswa menganggap output template sebagai hasil final. Misalnya, grafik berhasil dibuat, tetapi data berasal dari contoh, bukan dari output ChemCompute. Ini harus dicegah dengan instruksi eksplisit: setiap template harus diganti dengan data praktikum sebenarnya sebelum laporan dikumpulkan. Data contoh hanya untuk uji kode.

12. Implikasi Pedagogis Template Umum

Template kode umum memiliki implikasi pedagogis yang besar. Pertama, template menurunkan hambatan awal belajar Python. Mahasiswa tidak harus memikirkan struktur dasar dari nol. Kedua, template membentuk kebiasaan ilmiah: setiap kode memiliki data, rumus, tabel, grafik, ekspor,

dan kesimpulan. Ketiga, template membantu dosen memberi umpan balik karena semua mahasiswa menggunakan kerangka yang serupa. Keempat, template memudahkan transisi dari praktikum sederhana ke proyek mini. Kelima, template memperkuat keterampilan reprodusibilitas.

Dalam pembelajaran kimia kuantum, template juga membantu mahasiswa menghubungkan teori dengan praktik. Misalnya, pada partikel dalam kotak, template menunjukkan bahwa rumus energi dapat menjadi tabel dan grafik. Pada optimasi geometri, template menunjukkan bahwa koordinat dapat menjadi panjang dan sudut ikatan. Pada HOMO-LUMO, template menunjukkan bahwa dua nilai energi orbital dapat menjadi gap dan diagram. Pada IR, template menunjukkan bahwa frekuensi dan intensitas dapat menjadi spektrum. Pada energi reaksi, template menunjukkan bahwa energi setiap spesies dapat menjadi ΔE.

Dengan demikian, template bukan sekadar alat teknis, tetapi alat konseptual. Template membantu mahasiswa melihat pola umum analisis kimia kuantum: data mentah diubah menjadi parameter kimia, parameter kimia diubah menjadi tabel dan grafik, lalu tabel dan grafik diubah menjadi kesimpulan ilmiah.

13. Integrasi Template dengan Bab-Bab Praktikum

Template dalam subbab ini akan digunakan ulang pada bab-bab berikutnya:

- **Bab 4** menggunakan template partikel dalam kotak.
- **Bab 5** menggunakan template analisis geometri molekul.
- **Bab 6** menggunakan template tabel energi dan basis set.
- **Bab 7** menggunakan template HOMO-LUMO.
- **Bab 8** menggunakan template geometri, vektor, dan momen dipol.
- **Bab 9** menggunakan template spektrum IR simulasi.
- **Bab 10** menggunakan template energi reaksi.
- **Bab 11** mengembangkan template menjadi otomasi analisis output dan laporan.
- **Bab 12** menggunakan template untuk proyek mini dan evaluasi akhir.

Dengan penggunaan berulang, mahasiswa akan semakin terbiasa. Pada awalnya, mereka mungkin hanya mengganti data contoh. Setelah beberapa praktikum, mereka mulai memodifikasi fungsi. Pada proyek mini, mereka dapat menggabungkan beberapa template menjadi workflow mandiri. Inilah perkembangan yang diharapkan: dari pengguna template menjadi perancang analisis komputasi.

14. Template sebagai Jembatan ke Praktikum Nyata

Subbab ini menutup Bab 3 dengan menyediakan jembatan langsung menuju Bab 4. Bab 1 telah membangun fondasi teori dan etika. Bab 2 telah membangun ekosistem software dan standar file. Bab 3 membangun fondasi Python, paket, struktur kode, standar penulisan, dan template umum. Setelah ini, mahasiswa siap masuk ke praktikum pertama: visualisasi fungsi gelombang dan partikel dalam kotak.

Dengan template umum, Bab 4 tidak perlu lagi menjelaskan dari awal bagaimana mengimpor pustaka, membuat folder, mendefinisikan konstanta, membuat grafik, atau menyimpan file. Bab 4 dapat langsung menggunakan struktur yang telah dibangun di sini untuk fokus pada makna fisik dan kimia dari fungsi gelombang, rapat probabilitas, bilangan kuantum, dan kuantisasi energi. Dengan demikian, subbab 3.5 berfungsi sebagai titik transisi dari persiapan teknis menuju pelaksanaan praktikum kimia kuantum yang sesungguhnya.

15. Sintesis Subbab

Template kode umum praktikum adalah fondasi operasional yang akan digunakan berulang dalam seluruh buku. Template ini menyatukan struktur kode, standar penulisan, standar file, dan kebutuhan laporan. Template membantu mahasiswa bekerja lebih sistematis, mengurangi kesalahan format, memperkuat reprodusibilitas, dan memudahkan dosen memeriksa hasil. Namun, template harus digunakan secara kritis. Mahasiswa harus memahami setiap bagian, mengganti data contoh dengan data praktikum sebenarnya, memeriksa satuan, memvalidasi hasil, dan menulis kesimpulan sesuai batas metode.

Dalam praktikum kimia kuantum, template yang baik membuat Python menjadi alat ilmiah yang transparan. Data tidak lagi tersembunyi. Rumus tidak lagi hanya ditulis di teori. Grafik tidak lagi dibuat manual tanpa jejak. Kesimpulan tidak lagi sekadar pernyataan umum. Semuanya tersambung: input, kode, data, grafik, dan laporan. Dengan penguasaan template ini, mahasiswa siap memasuki bab-bab praktikum yang lebih konkret.

Penutup Bab 3

Bab 3 telah membahas dasar-dasar representasi struktur molekul dalam praktikum kimia kuantum. Setelah Bab 1 memberikan orientasi konseptual dan Bab 2 menyiapkan lingkungan kerja komputasi, Bab 3 menempatkan molekul sebagai objek utama praktikum. Molekul tidak lagi hanya dipahami sebagai rumus kimia, gambar dua dimensi, atau model konseptual di buku teks, tetapi sebagai struktur tiga dimensi yang dapat ditulis, disimpan, dihitung, divisualisasikan, dan dianalisis secara komputasional.

Gagasan utama Bab 3 adalah bahwa setiap perhitungan kimia kuantum selalu dimulai dari **struktur molekul**. Struktur molekul menentukan posisi atom, jarak antarinti, sudut ikatan, orientasi ruang, dan bentuk awal sistem yang akan dihitung. Jika struktur awal salah, maka perhitungan berikutnya dapat gagal, menghasilkan energi yang tidak wajar, geometri yang keliru, atau interpretasi kimia yang menyesatkan. Oleh karena itu, kemampuan membangun, memeriksa, menyimpan, dan membaca struktur molekul merupakan keterampilan dasar yang harus dikuasai sebelum mahasiswa menjalankan optimasi geometri, perhitungan energi, analisis orbital, momen dipol, atau spektrum IR.

Bab ini juga menegaskan pentingnya memahami berbagai format representasi molekul, terutama **koordinat Cartesian**, **format XYZ**, dan representasi struktur dalam perangkat lunak visualisasi seperti Avogadro. Dalam format XYZ, setiap atom ditulis dengan simbol unsur dan tiga koordinat ruang. Format ini sederhana, tetapi sangat penting karena banyak software kimia komputasi dapat membaca atau mengubahnya menjadi input perhitungan. Mahasiswa harus memahami bahwa

kesalahan satu simbol atom, satu angka koordinat, atau satu baris format dapat mengubah makna struktur secara keseluruhan.

Selain format XYZ, Bab 3 juga memperkenalkan gagasan tentang struktur dua dimensi dan tiga dimensi. Struktur dua dimensi berguna untuk memperlihatkan konektivitas atom dan rumus struktur, tetapi belum cukup untuk perhitungan kimia kuantum yang memerlukan posisi atom dalam ruang. Struktur tiga dimensi lebih dekat dengan kebutuhan komputasi karena menyertakan informasi geometri. Oleh karena itu, proses membangun molekul dalam Avogadro atau editor molekul lain harus diikuti dengan pemeriksaan geometri, penambahan atom hidrogen, pemeriksaan muatan, pemeriksaan multiplicity, dan penyimpanan file dalam format yang sesuai.

Bab ini juga menggarisbawahi bahwa struktur awal bukan selalu struktur akhir. Struktur awal hanyalah tebakan geometri yang akan digunakan sebagai masukan perhitungan. Dalam optimasi geometri, software akan memperbaiki posisi atom secara bertahap sampai diperoleh struktur dengan energi lebih rendah dan gaya yang memenuhi kriteria konvergensi. Dengan demikian, mahasiswa harus membedakan antara **struktur input** dan **struktur hasil optimasi**. Struktur input dapat dibuat secara manual atau melalui editor molekul, sedangkan struktur hasil optimasi diperoleh dari output perhitungan.

Secara metodologis, Bab 3 menanamkan prinsip bahwa kualitas input menentukan kualitas proses komputasi. Prinsip ini sejalan dengan ungkapan umum dalam komputasi ilmiah: data masuk yang buruk akan menghasilkan keluaran yang buruk. Dalam kimia komputasi, struktur molekul adalah bagian terpenting dari data input. Jika molekul salah dibangun, jika atom hidrogen kurang, jika ikatan tidak wajar, jika koordinat terlalu rapat, atau jika muatan salah, maka hasil perhitungan tidak dapat dipercaya.

Bab 3 juga memiliki peran pedagogis penting. Melalui kegiatan membangun dan memeriksa struktur molekul, mahasiswa belajar menghubungkan teori ikatan kimia dengan geometri molekul. Mereka tidak hanya mengetahui bahwa H_2O berbentuk bengkok atau CH_4 berbentuk tetrahedral, tetapi juga melihat bagaimana bentuk tersebut dinyatakan dalam koordinat tiga dimensi. Mereka tidak hanya mengetahui bahwa molekul memiliki panjang ikatan dan sudut ikatan, tetapi juga memahami bahwa panjang dan sudut tersebut berasal dari posisi atom dalam ruang.

Dengan demikian, Bab 3 menjadi fondasi praktis bagi seluruh perhitungan pada bab berikutnya. Optimasi geometri pada Bab 4 atau Bab 5, perhitungan Hartree–Fock pada Bab 6, DFT, HOMO-LUMO, momen dipol, dan spektrum IR semuanya memerlukan struktur molekul yang benar. Tanpa struktur awal yang baik, seluruh workflow praktikum dapat terganggu. Oleh karena itu, Bab 3 dapat disimpulkan sebagai bab yang membentuk keterampilan dasar mahasiswa dalam menerjemahkan konsep molekul menjadi objek komputasi tiga dimensi yang siap dihitung.

Rangkuman Bab 3

1. **Struktur molekul** merupakan titik awal setiap perhitungan kimia kuantum dan kimia komputasi.

2. Molekul dalam praktikum tidak cukup dipahami sebagai rumus kimia, tetapi harus direpresentasikan sebagai **susunan atom dalam ruang tiga dimensi**.
3. **Koordinat Cartesian** menyatakan posisi setiap atom dengan tiga angka koordinat, yaitu x, y, dan z.
4. **Format XYZ** merupakan format sederhana untuk menyimpan struktur molekul, berisi jumlah atom, baris komentar, dan daftar simbol atom beserta koordinatnya.
5. Struktur dua dimensi berguna untuk memahami konektivitas, tetapi perhitungan kuantum memerlukan struktur tiga dimensi.
6. Software seperti **Avogadro** dapat digunakan untuk membangun struktur awal, menambahkan hidrogen, memperkirakan geometri awal, dan menyimpan file molekul.
7. Struktur awal harus diperiksa sebelum digunakan, terutama simbol atom, jumlah atom, panjang ikatan, orientasi geometri, muatan, dan multiplicity.
8. Struktur awal berbeda dari struktur hasil optimasi. Struktur awal adalah tebakan, sedangkan struktur hasil optimasi diperoleh setelah perhitungan memperbaiki posisi atom.
9. Kesalahan dalam struktur input dapat menyebabkan perhitungan gagal, energi tidak wajar, geometri salah, atau kesimpulan kimia yang tidak valid.
10. Bab 3 menjadi dasar bagi praktikum optimasi geometri, perhitungan energi, orbital molekul, momen dipol, dan spektrum vibrasi pada bab-bab berikutnya.

Implikasi Pembelajaran Bab 3

Bab 3 memberikan beberapa implikasi penting dalam pembelajaran praktikum kimia kuantum.

Pertama, mahasiswa belajar bahwa struktur molekul adalah data ilmiah. Struktur bukan hanya gambar, tetapi representasi numerik yang dapat dihitung. Setiap atom memiliki posisi, setiap posisi memiliki konsekuensi terhadap energi, dan setiap energi bergantung pada geometri.

Kedua, mahasiswa dilatih untuk teliti dalam menyiapkan input. Keterampilan ini sangat penting karena kesalahan kecil dalam struktur awal dapat berdampak besar pada hasil perhitungan. Ketelitian dalam menulis simbol atom, memeriksa koordinat, menyimpan format file, dan memilih muatan merupakan bagian dari keterampilan ilmiah.

Ketiga, mahasiswa belajar menghubungkan bentuk molekul dengan teori kimia dasar. Bentuk linear, bengkok, trigonal planar, tetrahedral, dan bentuk molekul lainnya tidak hanya dipahami sebagai konsep VSEPR, tetapi dapat dinyatakan dalam koordinat tiga dimensi yang digunakan dalam software kimia komputasi.

Keempat, mahasiswa memahami bahwa visualisasi molekul harus dibaca secara kritis. Tampilan molekul yang terlihat indah belum tentu benar secara kimia. Mahasiswa harus memeriksa panjang ikatan, jumlah atom hidrogen, konektivitas, muatan, dan kewajaran struktur.

Kelima, Bab 3 memperkuat prinsip reprodusibilitas. Struktur molekul yang digunakan sebagai input harus disimpan dan diberi nama jelas. Jika struktur awal tidak terdokumentasi, maka hasil perhitungan sulit diulang dan sulit diperiksa.

Arah Menuju Bab 4

Setelah mahasiswa memahami cara membangun dan merepresentasikan struktur molekul, Bab 4 dapat diarahkan pada praktikum awal yang menggunakan struktur tersebut sebagai input perhitungan. Bab berikutnya dapat membahas bagaimana struktur awal dijalankan dalam software, bagaimana job komputasi disiapkan, bagaimana energi awal diperoleh, serta bagaimana mahasiswa mulai membaca output sederhana.

Dengan demikian, Bab 3 berfungsi sebagai jembatan antara persiapan teknis pada Bab 2 dan perhitungan molekul yang lebih operasional pada bab berikutnya. Mahasiswa yang menguasai Bab 3 akan lebih siap menjalankan optimasi geometri, perhitungan energi, dan analisis struktur elektronik karena telah memahami bagaimana molekul dinyatakan sebagai data komputasi.

Daftar Pustaka Bab 3

Allouche, A. R. (2011). Gabedit—A graphical user interface for computational chemistry softwares. *Journal of Computational Chemistry, 32*(1), 174–182.

Cramer, C. J. (2004). *Essentials of Computational Chemistry: Theories and Models* (2nd ed.). John Wiley & Sons.

Foresman, J. B., & Frisch, Æ. (2015). *Exploring Chemistry with Electronic Structure Methods* (3rd ed.). Gaussian, Inc.

Hanwell, M. D., Curtis, D. E., Lonie, D. C., Vandermeersch, T., Zurek, E., & Hutchison, G. R. (2012). Avogadro: An advanced semantic chemical editor, visualization, and analysis platform. *Journal of Cheminformatics, 4*, 17.

Helgaker, T., Jørgensen, P., & Olsen, J. (2000). *Molecular Electronic-Structure Theory*. John Wiley & Sons.

Jensen, F. (2017). *Introduction to Computational Chemistry* (3rd ed.). John Wiley & Sons.

Leach, A. R. (2001). *Molecular Modelling: Principles and Applications* (2nd ed.). Prentice Hall.

Lewars, E. G. (2016). *Computational Chemistry: Introduction to the Theory and Applications of Molecular and Quantum Mechanics* (3rd ed.). Springer.

Murray-Rust, P., & Rzepa, H. S. (1999). Chemical Markup, XML, and the World Wide Web. 1. Basic principles. *Journal of Chemical Information and Computer Sciences, 39*(6), 928–942.

Ochterski, J. W. (2000). *Thermochemistry in Gaussian*. Gaussian, Inc.

Open Babel Development Team. (2024). *Open Babel Documentation*. Open Babel Project.

Ramachandran, K. I., Deepa, G., & Namboori, K. (2008). *Computational Chemistry and Molecular Modeling: Principles and Applications*. Springer.

Schmidt, M. W., Baldridge, K. K., Boatz, J. A., Elbert, S. T., Gordon, M. S., Jensen, J. H., Koseki, S., Matsunaga, N., Nguyen, K. A., Su, S. J., Windus, T. L., Dupuis, M., & Montgomery, J. A. (1993). General atomic and molecular electronic structure system. *Journal of Computational Chemistry, 14*(11), 1347–1363.

Stewart, J. J. P. (1990). MOPAC: A semiempirical molecular orbital program. *Journal of Computer-Aided Molecular Design, 4*, 1–103.

Young, D. C. (2001). *Computational Chemistry: A Practical Guide for Applying Techniques to Real-World Problems*. John Wiley & Sons.

GLOSARIUM

A

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Ab Initio	Metode kimia komputasi yang berangkat dari prinsip mekanika kuantum tanpa parameter empiris utama.	Digunakan untuk menjelaskan metode seperti Hartree–Fock dan metode pasca-Hartree–Fock.	*Ab initio method*
Akurasi	Tingkat kedekatan hasil perhitungan dengan nilai rujukan atau nilai yang dianggap benar.	Dibahas ketika menilai hasil energi, geometri, basis set, dan keterbatasan metode.	*Accuracy*
Algoritma	Langkah-langkah sistematis untuk menyelesaikan masalah komputasi.	Digunakan dalam kode Python untuk membaca data, menghitung selisih energi, dan membuat grafik.	*Algorithm*
Analisis Data	Proses mengolah dan menafsirkan data hasil praktikum.	Menjadi bagian penting dalam jalur Python mandiri untuk membuat tabel, grafik, dan kesimpulan.	*Data analysis*
Avogadro	Software editor dan visualisasi molekul.	Digunakan untuk membangun struktur awal molekul dan menyimpan file struktur.	*Avogadro molecular editor*

B

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Basis Set	Kumpulan fungsi matematika untuk merepresentasikan orbital atom atau orbital molekul.	Menjadi topik utama pada praktikum Hartree–Fock, energi total, dan biaya komputasi.	*Basis set*
Basis Set Minimal	Basis set paling sederhana yang menyediakan fungsi dasar untuk orbital atom utama.	Contohnya STO-3G; digunakan untuk latihan cepat tetapi memiliki keterbatasan akurasi.	*Minimal basis set*
Born–Oppenheimer	Aproksimasi yang memisahkan gerak inti dan elektron.	Menjadi dasar pemahaman energi elektronik pada geometri molekul tertentu.	*Born–Oppenheimer approximation*
Bentuk Molekul	Susunan geometri atom dalam ruang tiga dimensi.	Dibahas pada molekul model seperti H_2O bengkok dan CH_4 tetrahedral.	*Molecular geometry*

C

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
ChemCompute	Platform web untuk menjalankan perhitungan kimia komputasi.	Digunakan sebagai jalur praktikum berbasis web, terutama melalui GAMESS.	*ChemCompute*
CSV	Format file berbasis teks untuk menyimpan data tabel.	Digunakan untuk menyimpan data energi, selisih energi, dan ringkasan basis set efisien.	*Comma-Separated Values*
Convergence	Kondisi ketika perhitungan mencapai kriteria berhenti yang stabil.	Digunakan dalam SCF dan optimasi geometri.	*Convergence*
Correlation Energy	Selisih antara energi eksak dan energi Hartree–Fock limit.	Menjelaskan keterbatasan Hartree–Fock dalam memasukkan korelasi elektron.	*Correlation energy*

D

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Data Input	Data awal yang diberikan kepada program komputasi.	Dapat berupa struktur molekul, metode, basis set, muatan, dan multiplicity.	*Input data*
Data Output	Data hasil yang dihasilkan program.	Meliputi energi total, koordinat akhir, waktu komputasi, dan status konvergensi.	*Output data*
Density Functional Theory	Metode struktur elektronik berbasis kerapatan elektron.	Menjadi metode lanjutan setelah mahasiswa memahami Hartree–Fock dan basis set.	*DFT*
Determinan Slater	Bentuk fungsi gelombang banyak elektron yang memenuhi antisimetri fermion.	Digunakan dalam teori Hartree–Fock.	*Slater determinant*
Difus	Fungsi basis yang menyebar jauh dari inti.	Penting untuk anion, keadaan tereksitasi, dan distribusi elektron yang luas.	*Diffuse function*

E

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Efisiensi Komputasi	Keseimbangan antara kualitas hasil dan biaya komputasi.	Digunakan untuk menilai pilihan basis set yang paling layak.	*Computational efficiency*
Elektron	Partikel bermuatan negatif yang menentukan struktur elektronik molekul.	Menjadi objek utama dalam perhitungan kimia kuantum.	*Electron*
Energi Elektronik	Energi yang berkaitan dengan gerak dan interaksi elektron dalam molekul.	Dihitung pada geometri inti tertentu dalam pendekatan Born–Oppenheimer.	*Electronic energy*
Energi Total	Energi keseluruhan hasil perhitungan pada metode dan basis set tertentu.	Menjadi data utama dalam praktikum Hartree–Fock dan basis set.	*Total energy*
Exchange	Efek kuantum akibat antisimetri fungsi gelombang elektron.	Dimasukkan dalam Hartree–Fock melalui determinan Slater.	*Exchange*

F

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
File Input	File yang berisi instruksi perhitungan.	Dapat memuat koordinat molekul, metode, basis set, muatan, dan jenis job.	*Input file*
File Output	File hasil eksekusi software kimia komputasi.	Harus disimpan untuk validasi, laporan, dan reprodusibilitas.	*Output file*
Fock Matrix	Matriks utama dalam perhitungan Hartree–Fock.	Digunakan dalam prosedur SCF untuk memperoleh orbital molekul.	*Fock matrix*
Fungsi Basis	Fungsi matematika pembentuk orbital.	Dipakai untuk membangun orbital molekul dalam metode Hartree–Fock dan metode lain.	*Basis function*
Fungsi Polarisasi	Fungsi tambahan dengan momentum sudut lebih tinggi.	Membantu menggambarkan distorsi distribusi elektron pada ikatan.	*Polarization function*

G

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
GAMESS	Paket software kimia kuantum ab initio.	Digunakan melalui ChemCompute untuk menjalankan perhitungan Hartree–Fock.	*General Atomic and Molecular Electronic Structure System*
Gaussian	Software komersial untuk perhitungan struktur elektronik.	Diperkenalkan sebagai salah satu software alternatif dalam praktikum.	*Gaussian program*
Geometri Molekul	Susunan atom dalam ruang tiga dimensi.	Menjadi fokus dalam optimasi geometri dan analisis panjang ikatan.	*Molecular geometry*
Grafik Energi	Visualisasi hubungan energi dengan variabel tertentu.	Digunakan untuk menampilkan energi total terhadap ukuran basis set.	*Energy plot*

H

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Hartree	Satuan energi atomik dalam kimia kuantum.	Digunakan untuk menyatakan energi total hasil perhitungan.	*Hartree*
Hartree–Fock	Metode ab initio berbasis medan rata-rata dan determinan Slater.	Menjadi metode utama dalam Praktikum 3.	*Hartree–Fock method*
Hartree–Fock Limit	Batas energi Hartree–Fock ketika basis set mendekati lengkap.	Menjelaskan bahwa basis set besar tidak otomatis menghasilkan energi eksak.	*Hartree–Fock limit*
HOMO	Orbital molekul terisi dengan energi tertinggi.	Digunakan dalam analisis sifat elektronik molekul.	*Highest Occupied Molecular Orbital*

I

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
In Silico	Praktikum atau kajian berbasis simulasi komputer.	Menjadi karakter utama buku ini.	*In silico*
Input Molekul	Struktur molekul yang dimasukkan ke software.	Dapat berupa file XYZ, struktur dari Avogadro, atau input manual.	*Molecular input*

Integral Dua-Elektron	Integral yang menggambarkan interaksi antarelektron.	Berkontribusi besar terhadap biaya komputasi Hartree–Fock.	*Two-electron integral*
Interpretasi Kimia	Penjelasan makna hasil komputasi berdasarkan konsep kimia.	Ditekankan agar mahasiswa tidak hanya menyalin output.	*Chemical interpretation*

J

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Job Komputasi	Satu tugas perhitungan yang dijalankan software.	Contohnya single point energy, optimasi geometri, atau perhitungan frekuensi.	*Computational job*
Jalur ChemCompute	Alur praktikum berbasis platform ChemCompute.	Digunakan untuk menjalankan perhitungan melalui GAMESS secara web.	*ChemCompute workflow*
Jalur Python Mandiri	Alur analisis data menggunakan Python.	Digunakan untuk membuat tabel, grafik, CSV, dan analisis konvergensi sederhana.	*Python workflow*
Jumlah Fungsi Basis	Banyaknya fungsi basis dalam perhitungan.	Dipakai untuk menilai ukuran basis set dan biaya komputasi.	*Number of basis functions*

K

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Kerapatan Elektron	Distribusi probabilitas elektron dalam ruang molekul.	Menjadi dasar pemahaman DFT dan sifat elektronik.	*Electron density*
Kimia Komputasi	Cabang kimia yang menggunakan metode komputasi untuk mempelajari sistem kimia.	Menjadi payung utama seluruh praktikum dalam buku ini.	*Computational chemistry*
Kimia Kuantum	Kajian kimia berdasarkan prinsip mekanika kuantum.	Menjadi dasar teoritis perhitungan struktur elektronik molekul.	*Quantum chemistry*
Konvergensi SCF	Keadaan ketika iterasi SCF mencapai kestabilan energi atau densitas.	Menentukan validitas energi Hartree–Fock.	*SCF convergence*
Koordinat Cartesian	Sistem koordinat x, y, z untuk menyatakan posisi atom.	Digunakan dalam format XYZ dan input molekul.	*Cartesian coordinates*

L

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Laporan Praktikum	Dokumen yang menyajikan prosedur, data, analisis, dan kesimpulan praktikum.	Harus memuat metode, basis set, software, output, dan interpretasi.	*Laboratory report*
LiH	Molekul lithium hidrida.	Digunakan sebagai molekul model diatomik polar.	*Lithium hydride*
LUMO	Orbital molekul kosong dengan energi terendah.	Digunakan dalam analisis sifat elektronik dan reaktivitas molekul.	*Lowest Unoccupied Molecular Orbital*

M

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Matriks Overlap	Matriks yang menggambarkan tumpang tindih fungsi basis.	Muncul dalam persamaan Roothaan–Hall.	*Overlap matrix*
Medan Rata-Rata	Pendekatan yang mengganti interaksi elektron langsung dengan medan efektif.	Menjadi dasar metode Hartree–Fock.	*Mean field*
Metode Komputasi	Cara matematis untuk menghitung sifat molekul.	Contohnya Hartree–Fock, DFT, dan metode semiempiris.	*Computational method*
Molekul Model	Molekul sederhana yang dipilih untuk praktikum.	Contohnya H_2, LiH, HF, H_2O, dan CH_4.	*Model molecule*
Multiplicity	Nilai yang menunjukkan keadaan spin total sistem.	Harus ditentukan dalam input perhitungan, misalnya singlet dengan multiplicity 1.	*Spin multiplicity*

N

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Nama File	Identitas file yang menunjukkan isi atau fungsi file.	Ditekankan agar data, kode, dan output praktikum mudah ditelusuri.	*File name*
Numerik	Berbasis perhitungan angka.	Digunakan untuk menjelaskan sifat perhitungan energi, geometri, dan grafik.	*Numerical*
Nukleus	Inti atom bermuatan positif.	Posisi inti menentukan geometri molekul dalam perhitungan elektronik.	*Nucleus*

Normal Mode	Pola gerak vibrasi molekul.	Relevan dalam praktikum spektrum IR dan frekuensi vibrasi.	*Normal mode*

O

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Optimasi Geometri	Proses mencari struktur molekul dengan energi lebih rendah dan gaya kecil.	Menjadi salah satu praktikum utama sebelum analisis energi lanjutan.	*Geometry optimization*
Orbital Atom	Fungsi gelombang elektron pada atom.	Menjadi dasar pembentukan orbital molekul melalui kombinasi fungsi basis.	*Atomic orbital*
Orbital Molekul	Fungsi matematika yang menggambarkan keadaan elektron dalam molekul.	Digunakan dalam Hartree–Fock, HOMO-LUMO, dan analisis struktur elektronik.	*Molecular orbital*
ORCA	Software kimia kuantum untuk perhitungan struktur elektronik.	Diperkenalkan sebagai alternatif software lokal atau server.	*ORCA program*
Output Kode	Hasil yang dihasilkan oleh kode Python.	Meliputi CSV, tabel, grafik, dan ringkasan basis set efisien.	*Code output*

P

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Panjang Ikatan	Jarak antara dua inti atom yang berikatan.	Dianalisis pada optimasi geometri dan pengaruh basis set.	*Bond length*
Pandas	Pustaka Python untuk analisis data tabel.	Digunakan untuk membaca CSV dan menyusun tabel hasil praktikum.	*pandas*
Persamaan Roothaan–Hall	Bentuk matriks dari persamaan Hartree–Fock berbasis fungsi basis.	Menjelaskan hubungan matriks Fock, koefisien orbital, dan matriks overlap.	*Roothaan–Hall equation*
Polaritas	Ketidakseimbangan distribusi muatan dalam molekul atau ikatan.	Dibahas pada molekul seperti LiH, HF, dan H_2O.	*Polarity*
Python	Bahasa pemrograman untuk analisis data dan otomasi.	Digunakan untuk tabel, grafik, CSV, dan analisis basis set.	*Python*
PySCF	Kerangka kimia kuantum berbasis Python.	Digunakan sebagai alternatif perhitungan dan analisis struktur elektronik.	*Python-based Simulations of Chemistry Framework*

Q

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Quantum Chemistry	Kajian kimia berdasarkan mekanika kuantum.	Padanan asing dari kimia kuantum, menjadi dasar seluruh buku.	*Quantum chemistry*
Quantum Mechanics	Teori fisika yang menjelaskan perilaku partikel mikroskopik.	Menjadi fondasi bagi persamaan Schrödinger dan metode struktur elektronik.	*Quantum mechanics*
Quasi-Newton	Kelompok metode optimasi numerik yang memperkirakan Hessian.	Relevan dalam optimasi geometri molekul.	*Quasi-Newton method*

R

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Reprodusibilitas	Kemampuan hasil praktikum untuk ditelusuri dan diulang.	Ditekankan melalui penyimpanan input, output, CSV, kode, dan grafik.	*Reproducibility*
Restricted Hartree–Fock	Hartree–Fock untuk sistem elektron berpasangan.	Digunakan pada molekul tertutup seperti H_2, H_2O, dan CH_4.	*RHF*
Ringkasan Efisiensi	Ringkasan basis set yang memberi kompromi baik antara energi dan waktu.	Dihasilkan oleh kode Python pada Praktikum 3.	*Efficiency summary*
Rumus Struktur	Representasi konektivitas atom dalam molekul.	Dibedakan dari struktur tiga dimensi yang diperlukan untuk komputasi.	*Structural formula*

S

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
SCF	Prosedur iteratif untuk memperoleh medan dan orbital yang konsisten.	Menjadi inti perhitungan Hartree–Fock.	*Self-Consistent Field*
Semiempiris	Metode komputasi yang menggunakan parameter empiris untuk mempercepat perhitungan.	Dibandingkan dengan ab initio dan DFT dalam konteks metode komputasi.	*Semiempirical method*

Single Point Energy	Perhitungan energi pada geometri tetap.	Digunakan untuk membandingkan pengaruh basis set pada struktur yang sama.	*Single point energy*
Spin	Sifat kuantum elektron yang berkaitan dengan momentum sudut intrinsik.	Berhubungan dengan multiplicity, RHF, dan UHF.	*Spin*
Struktur Molekul	Susunan atom dalam molekul.	Menjadi input dasar seluruh praktikum.	*Molecular structure*

T

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Tabel Energi	Tabel yang memuat energi total hasil perhitungan.	Digunakan untuk membandingkan metode, basis set, dan waktu komputasi.	*Energy table*
Tetrahedral	Bentuk molekul dengan empat substituen mengarah ke sudut tetrahedron.	Digunakan untuk menjelaskan bentuk CH_4.	*Tetrahedral*
Transisi Elektronik	Perpindahan elektron antara tingkat energi.	Relevan dalam pembahasan spektrum UV-Vis dan orbital molekul.	*Electronic transition*
Two-Electron Repulsion	Tolakan antara dua elektron.	Menjadi salah satu sumber kompleksitas dalam kimia kuantum.	*Two-electron repulsion*

U

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Unrestricted Hartree–Fock	Hartree–Fock yang membolehkan orbital alfa dan beta berbeda.	Relevan untuk radikal atau sistem elektron tidak berpasangan.	*UHF*
Unit Atomik	Sistem satuan yang umum dalam kimia kuantum.	Hartree digunakan sebagai satuan energi dalam output perhitungan.	*Atomic units*
Ukuran Basis Set	Besar kecilnya basis set berdasarkan jumlah dan jenis fungsi basis.	Dipakai untuk menilai hubungan energi dan waktu komputasi.	*Basis set size*

V

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Validasi Data	Pemeriksaan kebenaran dan kelayakan data sebelum dianalisis.	Meliputi status SCF, basis set, jenis job, dan satuan energi.	*Data validation*
Variational Principle	Prinsip bahwa energi fungsi percobaan tidak lebih rendah dari energi eksak keadaan dasar.	Menjelaskan mengapa basis set lebih besar umumnya menurunkan energi Hartree–Fock.	*Variational principle*
Vibrasi Molekul	Gerak periodik atom-atom dalam molekul.	Relevan dalam praktikum frekuensi vibrasi dan spektrum IR.	*Molecular vibration*
Visualisasi Molekul	Tampilan grafis struktur molekul.	Digunakan untuk memeriksa bentuk molekul sebelum dan sesudah optimasi.	*Molecular visualization*

W

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Workflow	Alur kerja sistematis dari input sampai output.	Digunakan untuk menjelaskan jalur ChemCompute, jalur Python, dan pelaporan praktikum.	*Workflow*
Waktu Komputasi	Lama waktu yang dibutuhkan komputer untuk menyelesaikan perhitungan.	Dianalisis terhadap ukuran basis set dan jumlah fungsi basis.	*Computational time*
Web-Based Platform	Platform yang dijalankan melalui browser.	ChemCompute digunakan sebagai platform web untuk praktikum.	*Web-based platform*

X

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
XYZ	Format sederhana untuk menyimpan koordinat atom.	Digunakan untuk menulis struktur molekul awal dalam koordinat Cartesian.	*XYZ file format*
xTB	Metode semiempiris modern berbasis tight-binding.	Dapat menjadi pengayaan dalam pembahasan metode komputasi cepat.	*Extended tight-binding*
X-Axis	Sumbu horizontal pada grafik.	Digunakan pada grafik energi atau waktu terhadap jumlah fungsi basis.	*X-axis*

Y

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Y-Axis	Sumbu vertikal pada grafik.	Digunakan untuk menampilkan energi total atau waktu komputasi dalam grafik.	*Y-axis*
Yield Data	Data keluaran yang dihasilkan dari suatu proses komputasi.	Dalam konteks buku ini merujuk pada hasil yang diperoleh dari kode atau software.	*Yield data*

Z

Istilah	Definisi Singkat	Penjelasan Kontekstual dalam Buku	Padanan Istilah Asing
Zero-Point Energy	Energi vibrasi minimum yang tetap dimiliki molekul pada keadaan dasar.	Relevan dalam pembahasan frekuensi vibrasi dan koreksi energi.	*ZPE / Zero-point energy*
Zona Konvergensi	Rentang ketika perubahan hasil mulai kecil dan relatif stabil.	Digunakan secara kontekstual untuk membaca kecenderungan konvergensi basis set sederhana.	*Convergence region*
Z-Matrix	Format koordinat internal molekul berbasis panjang ikatan, sudut, dan dihedral.	Dapat digunakan sebagai alternatif koordinat Cartesian dalam input molekul.	*Z-matrix*

Semarang, Juni 2026

Penulis,

Kasmui

$$\hat{H}\Psi = E\Psi$$

$$\hat{H} = -\frac{\hbar^2}{2m}\nabla^2 + V(r)$$

PRAKTIKUM KIMIA KUANTUM

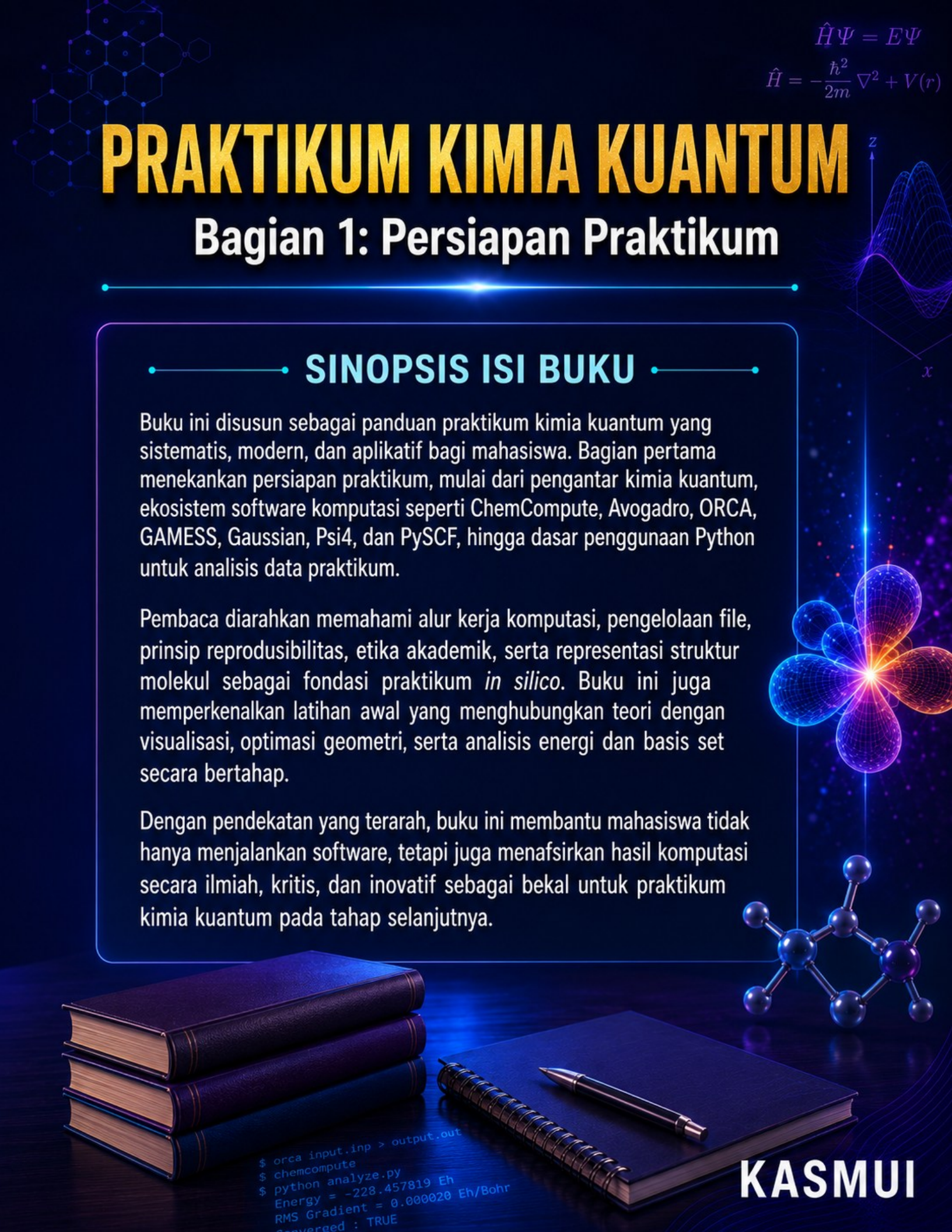
Bagian 1: Persiapan Praktikum

SINOPSIS ISI BUKU

Buku ini disusun sebagai panduan praktikum kimia kuantum yang sistematis, modern, dan aplikatif bagi mahasiswa. Bagian pertama menekankan persiapan praktikum, mulai dari pengantar kimia kuantum, ekosistem software komputasi seperti ChemCompute, Avogadro, ORCA, GAMESS, Gaussian, Psi4, dan PySCF, hingga dasar penggunaan Python untuk analisis data praktikum.

Pembaca diarahkan memahami alur kerja komputasi, pengelolaan file, prinsip reproducibilitas, etika akademik, serta representasi struktur molekul sebagai fondasi praktikum *in silico*. Buku ini juga memperkenalkan latihan awal yang menghubungkan teori dengan visualisasi, optimasi geometri, serta analisis energi dan basis set secara bertahap.

Dengan pendekatan yang terarah, buku ini membantu mahasiswa tidak hanya menjalankan software, tetapi juga menafsirkan hasil komputasi secara ilmiah, kritis, dan inovatif sebagai bekal untuk praktikum kimia kuantum pada tahap selanjutnya.



KASMUI