



$$\psi_k = \sum_{\mu} c_{\mu k} \chi_{\mu} \quad (\text{LCAO})$$

$$\hat{F} \varphi_i = \varepsilon_i \varphi_i$$

$$S_{\mu\nu} = \int \chi_{\mu}^*(\mathbf{r}) \chi_{\nu}(\mathbf{r}) d\mathbf{r}$$

С. М. Пономаренко

Квантово-механічні методи обчислення Використання PySCF

$$\left[-\frac{1}{2} \nabla_i^2 + \sum_{j \neq i} \frac{1}{r_{ij}} + V_{\text{nuc}}(i) \right] \varphi_i = \varepsilon_i \varphi_i$$
$$\Psi = \det \begin{pmatrix} \varphi_1(1) & \varphi_2(1) & \cdots & \varphi_N(1) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(N) & \varphi_2(N) & \cdots & \varphi_N(N) \end{pmatrix}$$

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

С. М. Пономаренко

Квантово-механічні методи обчислення Використання PySCF

*Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як
навчальний посібник для здобувачів ступеня магістра за
спеціальностями Еб «Прикладна фізика та наноматеріали»*

КИЇВ
КПІ ім. Ігоря Сікорського
2025

1	Метод Хартрі-Фока для атомів	4
1.1	Теоретичні основи методу Хартрі-Фока	4
1.1.1	Рівняння Хартрі-Фока	4
1.1.2	Варіанти методу Хартрі-Фока	4
1.1.3	Енергія Хартрі-Фока	5
1.2	Розрахунок атома Гідрогену	5
1.2.1	Особливості одноелектронної системи	6
1.2.2	Залежність від базисного набору	6
1.2.3	Аналіз орбіталей	7
1.2.4	Висновки	8
1.3	Розрахунок атома Гелію	8
1.3.1	Двоелектронна система	8
1.3.2	Порівняння RHF та UHF	9
1.3.3	Збуджені стани Гелію	10
1.4	Атоми другого періоду (Li–Ne)	11
1.4.1	Літій (Li, $Z=3$)	11
1.4.2	Берилій (Be, $Z=4$)	12
1.4.3	Бор–Неон: систематичне дослідження	13
1.4.4	Енергії іонізації	16
1.5	Аналіз енергій та орбіталей	17
1.5.1	Енергетична діаграма орбіталей	17
1.5.2	Розподіл електронної густини	19
1.5.3	Порівняння електронних густин різних атомів	24
1.6	Порівняння методів: RHF vs UHF vs ROHF	27
1.6.1	Тестовий випадок: атом Вуглецю	27
1.6.2	Теоретичний аналіз результатів	29
1.6.3	Систематичне порівняння для всіх атомів	29
1.7	Складні випадки та збіжність	31
1.7.1	Перехідні метали	31
1.7.2	Використання дробових заповнень	33
1.7.3	Стратегії досягнення конвергенції	35
1.8	Практичні завдання	36
1.8.1	Завдання 1: Систематичне дослідження	36
1.8.2	Завдання 2: Енергії іонізації	37

1.8.3	Завдання 3: Спектроскопічні константи	37
1.8.4	Завдання 4: Залежність від базису	37
1.9	Резюме	38
1.9.1	Ключові висновки	38

Література	39
-------------------	-----------

1

Метод Хартрі-Фока для атомів

1.1. Теоретичні основи методу Хартрі-Фока

1.1.1. Рівняння Хартрі-Фока. Метод Хартрі-Фока (HF) є наближеним методом розв'язання рівняння Шредінгера для багатоелектронних систем. Основна ідея полягає у представленні багатоелектронної хвильової функції у вигляді єдиного детермінанта Слейтера:

$$\Psi(\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \psi_1(\mathbf{r}_1) & \psi_2(\mathbf{r}_1) & \cdots & \psi_N(\mathbf{r}_1) \\ \psi_1(\mathbf{r}_2) & \psi_2(\mathbf{r}_2) & \cdots & \psi_N(\mathbf{r}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_1(\mathbf{r}_N) & \psi_2(\mathbf{r}_N) & \cdots & \psi_N(\mathbf{r}_N) \end{vmatrix} \quad (1.1)$$

де $\psi_i(\mathbf{r})$ — спін-орбіталі, які є добутком просторової частини та спінової функції.

Канонічні рівняння Хартрі-Фока мають вигляд:

$$\hat{f}\psi_i = \varepsilon_i\psi_i \quad (1.2)$$

де \hat{f} — оператор Фока:

$$\hat{f} = \hat{h} + \sum_{j=1}^N (\hat{J}_j - \hat{K}_j) \quad (1.3)$$

Тут \hat{h} — одноелектронний оператор (кінетична енергія + притягання до ядра), \hat{J}_j — кулонівський оператор, \hat{K}_j — обмінний оператор.

1.1.2. Варіанти методу Хартрі-Фока.

Restricted Hartree-Fock (RHF)

RHF використовується для систем з замкненими оболонками, де всі електрони спарені:

$$\psi_i^\alpha(\mathbf{r}) = \psi_i^\beta(\mathbf{r}) = \varphi_i(\mathbf{r}) \quad (1.4)$$

Підходить для: He, Be, Ne, Mg, Ar, Ca, Zn (у синглетних станах).

Unrestricted Hartree-Fock (UHF)

UHF дозволяє різні просторові орбіталі для альфа та бета спінів:

$$\psi_i^\alpha(\mathbf{r}) \neq \psi_i^\beta(\mathbf{r}) \quad (1.5)$$

Підходить для: всіх відкритих систем (H, Li, B, C, N, O, F та їх іони).

Restricted Open-shell Hartree-Fock (ROHF)

ROHF — компроміс між RHF та UHF. Спарені електрони описуються однаковими орбіталями, неспарені — різними:

$$\begin{cases} \psi_i^\alpha = \psi_i^\beta & \text{для спарених} \\ \psi_i^\alpha \neq \psi_i^\beta & \text{для неспарених} \end{cases} \quad (1.6)$$

1.1.3. Енергія Хартрі-Фока. Повна енергія системи у методі HF:

$$E_{HF} = \sum_{i=1}^N h_{ii} + \frac{1}{2} \sum_{i,j}^N (J_{ij} - K_{ij}) + V_{NN} \quad (1.7)$$

де:

- h_{ii} — одноелектронні інтеграли
- J_{ij} — кулонівські інтеграли
- K_{ij} — обмінні інтеграли
- V_{NN} — енергія міжядерного відштовхування (= 0 для атомів)

1.2. Розрахунок атома Гідрогену

Атом Гідрогену є найпростішим квантовим об'єктом, який можна описати в рамках не лише хімії, а й фундаментальної квантової механіки. Його Гамільтоніан має вигляд:

$$\hat{H} = -\frac{1}{2}\nabla^2 - \frac{1}{r},$$

де $-\frac{1}{2}\nabla^2$ — оператор кінетичної енергії електрона, а $-\frac{1}{r}$ — потенціал кулонівської взаємодії між електроном і ядром. Ця система має аналітичне рішення, і енергія основного стану дорівнює

$$E_1 = -\frac{1}{2} \text{ Ha.}$$

Для атома Гідрогену це рішення можна отримати навіть чисельно в PySCF, що дозволяє перевірити точність обраного базисного набору та методів квантово-хімічних розрахунків.

1.2.1. Особливості одноелектронної системи. Атом Гідрогену є одноелектронною системою, тому метод Гартрі-Фока (HF) не містить жодних апроксимацій, окрім обмежень базисного набору. У звичайних багатоелектронних атомах HF наближає взаємодію електронів середнім потенціалом, але для Н відсутнє електрон-електронне відштовхування, тож метод дає *точну* хвильову функцію для обраного базису.

```
from pyscf import gto, scf
import numpy as np

# Створення атома Гідрогену
mol = gto.M(
    atom='H 0 0 0',      # координати ядра
    basis='sto-3g',       # мінімальний базис
    spin=1,              # один неспарений електрон
    verbose=4            # рівень деталізації виводу
)

print(f'Кількість електронів: {mol.nelectron}')
print(f'Базисних функцій: {mol.nao_nr()}')

# Розрахунок методом UHF (необмежений Гартрі-Фок)
mf = scf.UHF(mol)
energy = mf.kernel()

print(f'\nЕнергія H (STO-3G): {energy:.8f} Ha')
print(f'Енергія H (STO-3G): {energy * 27.211386:.6f} eV')

# Теоретичне значення
print(f'Теоретична енергія: -0.5 Ha')
print(f'Похибка базису: {abs(energy + 0.5):.6f} Ha')
```

Отримане значення енергії наближається до -0.5 Ha , але точність залежить від базису. STO-3G — це мінімальний базис, де кожна орбіталь апроксимується трьома гаусовими функціями, тому результат має невелику похибку (близько 10^{-3} Ha).

1.2.2. Залежність від базисного набору. Базисний набір визначає якість апроксимації хвильової функції. Чим більший набір, тим

ближче розрахована енергія до аналітичного результату. Для атома Гідрогену ця збіжність особливо показова, бо ми можемо порівняти з точним розв'язком.

У коді нижче порівнюються кілька популярних базисів, від найменшого STO-3G до розширених кореляційно-узгоджених наборів cc-pV5Z. Для кожного базису обчислюється енергія та похибка відносно -0.5 Ha.

```
from pyscf import gto, scf
import matplotlib.pyplot as plt

basis_sets = ['sto-3g', '3-21g', '6-31g', '6-311g',
              'cc-pvdz', 'cc-pvtz', 'cc-pvqz', 'cc-pv5z']

energies = []
n_basis = []

print('Базис          N_bas    Енергія (Ha)    Похибка (mHa)')
print('-' * 60)

for basis in basis_sets:
    try:
        mol = gto.M(atom='H 0 0 0', basis=basis, spin=1, verbose=0)
        mf = scf.UHF(mol)
        e = mf.kernel()

        n = mol.nao_nr()
        error = (e + 0.5) * 1000 # похибка в міліГартрі

        energies.append(e)
        n_basis.append(n)

        print(f'{basis:15s} {n:3d}    {e:12.8f}    {error:8.4f}')
    except:
        print(f'{basis:15s} --- недоступний')

# Побудова графіка збіжності
plt.figure(figsize=(10, 6))
plt.plot(n_basis, energies, 'o-', linewidth=2, markersize=8)
plt.axhline(y=-0.5, color='r', linestyle='--', label='Точне значення')
plt.xlabel('Кількість базисних функцій', fontsize=12)
plt.ylabel('Енергія (Ha)', fontsize=12)
plt.title('Збіжність енергії атома H від базисного набору', fontsize=14)
plt.grid(True, alpha=0.3)
plt.legend()
plt.tight_layout()
plt.savefig('h_atom_basis_convergence.pdf')
plt.show()
```

З графіка видно, що зі збільшенням кількості базисних функцій енергія швидко наближається до точного значення -0.5 Ha. Набори типу cc-pVQZ або cc-pV5Z практично дають збіжність до повного базисного ліміту (CBS limit).

1.2.3. Аналіз орбіталей. Хоча в атома Гідрогену існує лише одна заповнена орбіталь (1s), PySCF дозволяє вивести енергетичні рівні

1. Метод Хартрі-Фока для атомів

для всіх функцій базису, а також дослідити матрицю густини. Цей підхід зручний для демонстрації структури HF-розрахунку.

```
from pyscf import gto, scf
import numpy as np

mol = gto.M(atom='H 0 0 0', basis='cc-pvtz', spin=1)
mf = scf.UHF(mol)
energy = mf.kernel()

# Орбітальні енергії
print('\nОрбітальні енергії (альфа-спін):')
print('-' * 50)
for i, (e, label) in enumerate(zip(mf.mo_energy[0], mol.ao_labels())):
    occ = '(occ)' if i < mol.nelec[0] else '(virt)'
    print(f'{i+1:2d}. {label:20s}: {e:10.6f} Ha {occ}')

print(f'\nЕнергія 1s орбіталі: {mf.mo_energy[0][0]:.8f} Ha')
print(f'Теоретична енергія: -0.50000000 Ha')

# Матриця густини
dm = mf.make_rdm1()
print(f'\nМатриця густини (альфа): {dm[0].shape}')
print(f'След dm: {np.trace(dm[0]):.1f} (має дорівнювати 1)')
```

Матриця густини (dm) відображає заповнення орбіталей. Її слід дорівнює кількості електронів (тут — 1). Такі розрахунки є основою для подальшого аналізу електронної густини, побудови орбіталей і візуалізації електронної хмари.

1.2.4. Висновки.

- Для атома Гідрогену метод Гартрі-Фока є **точним**, бо немає взаємодії між електронами.
- Основна похибка виникає через обмеження базисного набору.
- Зі збільшенням кількості базисних функцій енергія швидко збігається до точного значення -0.5 Ha.
- PySCF дозволяє досліджувати вплив базису, аналізувати орбіталі, матриці густини та підготовляє ґрунт для подальшого розгляду багатеелектронних систем.

1.3. Розрахунок атома Гелію

1.3.1. Двоелектронна система. Атом гелію є фундаментальним прикладом для демонстрації методів **Хартрі-Фока (HF)**. Це перша система, де проявляється **електрон-електронна кореляція**, відсутня в атомі водню.

Два електрони гелію мають протилежні спіни й заповнюють одну й ту саму орбіталь $1s$, тому це замкнена оболонка зі спіном $S = 0$ (синглет, $M = 1$).

```

from pyscf import gto, scf

# Атом He (основний стан 1S)
mol = gto.M(
    atom='He 0 0 0',
    basis='cc-pvtz',
    spin=0, # Замкнена оболонка
    verbose=4
)

print(f'Електронна конфігурація: 1s2')
print(f'Кількість електронів: {mol.nelectron}')

# RHF для замкненої оболонки
mf = scf.RHF(mol)
energy = mf.kernel()

print(f'\nЕнергія He (HF): {energy:.8f} Ha')
print(f'Енергія He (HF): {energy * 27.211386:.6f} eV')

# Експериментальна енергія He: -2.90372 Ha
exp_energy = -2.90372
correlation_energy = exp_energy - energy

print(f'\nЕкспериментальна енергія: {exp_energy:.6f} Ha')
print(f'Кореляційна енергія: {correlation_energy:.6f} Ha')
print(f'Відносна похибка HF: {abs(correlation_energy/exp_energy)*100:.2f}%')

```

Коментар. Метод RHF нехтує кореляцією між електронами, тобто вважає, що хвильова функція є добутком одноелектронних орбіталей. Тому енергія HF завжди *вище* (менш від'ємна), ніж точна. Для гелію похибка складає близько $\sim 1.7\%$. Ця різниця — це **кореляційна енергія**:

$$E_{\text{corr}} = E_{\text{exact}} - E_{\text{HF}}$$

1.3.2. Порівняння RHF та UHF. Оскільки гелій має замкнену оболонку ($N_{\alpha} = N_{\beta}$), обидва методи — **RHF** (restricted HF) і **UHF** (unrestricted HF) — дають ідентичні результати. UHF дозволяє α та β орбіталям відрізнятися, але тут ця свобода не використовується.

```

from pyscf import gto, scf

mol = gto.M(atom='He 0 0 0', basis='6-31g', spin=0)

# RHF розрахунок
mf_rhf = scf.RHF(mol)
mf_rhf.verbose = 0
e_rhf = mf_rhf.kernel()

# UHF розрахунок
mf_uhf = scf.UHF(mol)
mf_uhf.verbose = 0
e_uhf = mf_uhf.kernel()

print(f'RHF енергія: {e_rhf:.10f} Ha')
print(f'UHF енергія: {e_uhf:.10f} Ha')

```

```
print(f'Різниця: {abs(e_rhf - e_uhf):.2e} Ha')

# Перевірка симетрії спіну
s2_uhf = mf_uhf.spin_square()
print(f'\n<S²> (UHF): {s2_uhf[0]:.6f}')
print(f'<S²> (точно): 0.000000')
```

Коментар. UHF може порушувати спінову симетрію (*spin contamination*), особливо для відкрито-оболонкових систем. Тут же $S^2 = 0$, тобто спінова симетрія зберігається, і RHF = UHF.

1.3.3. Збуджені стани Гелію. У збуджених станах один електрон переходить на більш високий рівень (наприклад, $2s$), а система може існувати в двох спінових конфігураціях:

- **Синглет:** $S = 0$ — антисиметрична за спіном, симетрична за просторовою координатою.
- **Триплет:** $S = 1$ — симетрична за спіном, антисиметрична за просторовою частиною.

У PySCF ці стани можна моделювати за допомогою відповідного значення параметра `spin` та методу R0HF для частково заповнених оболонок.

```
from pyscf import gto, scf

def he_excited_state(config='1s2s', spin=0, basis='cc-pvdz'):
    """
    Розрахунок збуджених станів He

    config: '1s2s' (синглет або триплет)
    spin: 0 для синглету, 2 для триплету
    """
    mol = gto.M(atom='He 0 0 0', basis=basis, spin=spin)

    if spin == 0:
        # Синглет: 1s² → 1s2s ¹S
        mf = scf.RHF(mol)
    else:
        # Триплет: 1s² → 1s2s ³S
        mf = scf.R0HF(mol)

    # Для збуджених станів можна використати MOM (maximum overlap method)
    mf = scf.addons.mom_occ(mf, mf.make_rdm1(), mf.get_occ())
    mf.verbose = 0
    energy = mf.kernel()

    return energy

# Основний стан
mol_gs = gto.M(atom='He 0 0 0', basis='cc-pvdz', spin=0)
mf_gs = scf.RHF(mol_gs)
mf_gs.verbose = 0
e_gs = mf_gs.kernel()

print('Стани атома Гелію (cc-pVDZ):')
print('-' * 50)
```

```
print(f'1s2 (1S, основний): {e_gs:.6f} Ha')
print(f'    = {e_gs * 27.211386:.2f} eV')

# Примітка: для збуджених станів
# краще використовувати методи типу TD-DFT або CASSCF
```

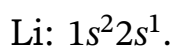
Коментар. Метод MOM (Maximum Overlap Method) дозволяє стабілізувати SCF для збуджених станів, зберігаючи певну орбітальну конфігурацію. Проте результати залишаються наближеними. Для точного опису необхідно застосовувати багатоконфігураційні підходи (наприклад, CASSCF, CI, TD-DFT).

Підсумок:

- Гелій — базовий тест для валідації HF-методу.
- Різниця між RHF і експериментом демонструє значення кореляційної енергії.
- Збуджені стани потребують спеціальних методів (MOM, CASSCF, TD-DFT).

1.4. Атоми другого періоду (Li–Ne)

1.4.1. Літій (Li, Z=3). Атом Літію — перший багатоелектронний атом, у якому проявляється **неспарений електрон** та **спінова поляризація**. Електронна конфігурація:



Два перші електрони утворюють замкнену оболонку (1s), а третій — одинокий у підоболонці 2s, що зумовлює мультиплетність 2S (спін $S = \frac{1}{2}$).

Через наявність неспареного електрона метод **UHF** (Unrestricted Hartree–Fock) є природним вибором. UHF дозволяє альфа- та бета-орбіталям відрізнятися, тобто хвильова функція не змушена мати однакову просторову форму для різних спінів.

```
from pyscf import gto, scf

# Li: [He] 2s1, основний стан 2S
mol = gto.M(
    atom='Li 0 0 0',
    basis='cc-pvdz',
    spin=1,      # один неспарений електрон (S=1/2)
    symmetry=True
)

print('Літій (Li):')
print(f'  Конфігурація: [He] 2s1')
print(f'  Основний стан: 2S')
print(f'  Електронів: {mol.nelectron}')
```

```
# UHF розрахунок
mf = scf.UHF(mol)
energy = mf.kernel()

print(f'\nЕнергія Li: {energy:.8f} Ha')
print(f'\nЕнергія Li: {energy * 27.211386:.4f} eV')

# Аналіз заселеностей
print('\nЗаселеності Малікена:')
pop = mf.mulliken_pop()

# Орбітальний аналіз
print('\nАльфа-орбіталі (заповнені):')
for i in range(mol.nelec[0]):
    print(f'  α-MO {i+1}: {mf.mo_energy[0][i]:10.6f} Ha')

print('\nБета-орбіталі (заповнені):')
for i in range(mol.nelec[1]):
    print(f'  β-MO {i+1}: {mf.mo_energy[1][i]:10.6f} Ha')

# Спінова густина
s2 = mf.spin_square()
print(f'\n<S²> = {s2[0]:.6f} (очікується 0.75)')
```

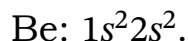
Коментарі.

- Значення $\langle S^2 \rangle \approx 0.75$ свідчить про правильний спіновий стан подвійності (мультиплетність $2S + 1 = 2$).
- Наявність різниці між енергіями альфа- та бета-орбіталей демонструє спінову асиметрію.
- Метод UHF може частково порушувати симетрію (так звана *spin contamination*), але для Li це незначно.

Енергія, отримана методом HF, для Li становить близько -7.432 Ha у базисі cc-pVDZ, тоді як експериментальна енергія основного стану (повна) — близько -7.478 Ha. Таким чином, кореляційна похибка становить близько 0.046 Ha (приблизно 1.25 eV).

Цей приклад є першим, де проявляється *кореляція електронів*, яку HF не враховує. В подальших розділах буде показано, як методи MP2, CI та CC враховують ці ефекти.

1.4.2. Берилій (Be, Z=4). Атом Берилію має електронну конфігурацію



Тут усі орбіталі парні, тому спінова поляризація відсутня, і зручно використовувати **RHF** (Restricted Hartree-Fock). Обидва електрони у підоболонці $2s$ мають протилежні спіни, тож система має мультиплетність $1S$ (синглетний стан).

```
from pyscf import gto, scf

# Be: [He] 2s², основний стан ¹S
```

```

mol = gto.M(
    atom='Be 0 0 0',
    basis='cc-pvtz',
    spin=0,          # замкнена оболонка
    symmetry=True
)

print('Берилій (Be):')
print(f' Конфігурація: [He] 2s2')
print(f' Основний стан: 1S')

# RHF розрахунок
mf = scf.RHF(mol)
energy = mf.kernel()

print(f'\nЕнергія Be: {energy:.8f} Ha')

# Порівняння з експериментом
exp_be = -14.6674 # Ha (експериментальна повна енергія)
print(f'Експериментальна: {exp_be:.4f} Ha')
print(f'Кореляційна енергія: {exp_be - energy:.4f} Ha')

```

Обговорення.

- Для Be HF-енергія виходить приблизно -14.573 Ha (у базисі cc-pVTZ), тоді як експериментальна — -14.667 Ha.
- Різниця близько 0.094 Ha (≈ 2.6 eV) — це **кореляційна енергія**, тобто внесок взаємодії електронів, не врахований у HF.
- У Берилія ефекти електронної кореляції вже досить значні, адже електрони в орбіталі $2s$ взаємодіють один з одним.
- Цей випадок є гарною ілюстрацією межі застосування методу Гартрі-Фока.

Висновки для атомів Li і Be.

- Li демонструє появу неспареного електрона і спінової поляризації (UHF необхідний).
- Be — перший приклад, де **електронна кореляція** дає помітну похибку енергії.
- PySCF дозволяє наочно дослідити вплив базису, спіну, симетрії та методів на точність енергії.

1.4.3. Бор-Неон: систематичне дослідження. Після розгляду окремих атомів Літію та Берилію доцільно перейти до систематичного аналізу всіх атомів другого періоду (від Бору до Неону). У цих атомах поступово заповнюється $2p$ -підрівень, і змінюється як спінова мультиплетність, так і форма електронної густини. Метод Гартрі-Фока дозволяє побачити, як змінюється енергія системи при збільшенні числа електронів, а також як проявляється спінова поляризація у відкритих оболонках.

Електронні конфігурації.

Li:	[He] $2s^1$	2S
Be:	[He] $2s^2$	1S
B:	[He] $2s^2 2p^1$	2P
C:	[He] $2s^2 2p^2$	3P
N:	[He] $2s^2 2p^3$	4S
O:	[He] $2s^2 2p^4$	3P
F:	[He] $2s^2 2p^5$	2P
Ne:	[He] $2s^2 2p^6$	1S

Як видно, кількість неспарених електронів збільшується від Бору до Нітрогену, а потім зменшується до Неону, що зумовлює зміну спінового стану та мультиплетності.

Мета дослідження. Провести розрахунок енергій Гартрі-Фока для атомів другого періоду в однаковому базисі cc-pVDZ, оцінити правильність спінового стану (через $\langle S^2 \rangle$) та проаналізувати систематичні тенденції.

```
from pyscf import gto, scf
import numpy as np

# Дані про атоми другого періоду
atoms_data = [
    ('Li', 1, '^2S', '[He] 2s^1'),
    ('Be', 0, '^1S', '[He] 2s^2'),
    ('B', 1, '^2P', '[He] 2s^2 2p^1'),
    ('C', 2, '^3P', '[He] 2s^2 2p^2'),
    ('N', 3, '^4S', '[He] 2s^2 2p^3'),
    ('O', 2, '^3P', '[He] 2s^2 2p^4'),
    ('F', 1, '^2P', '[He] 2s^2 2p^5'),
    ('Ne', 0, '^1S', '[He] 2s^2 2p^6')
]

basis = 'cc-pvdz'
print(f'Розрахунок атомів 2-го періоду (базис: {basis})')
print('=' * 70)
print(f'{"Атом":4s} {"Спін":4s} {"Терм":4s} {"E(HF), Ha":15s} '
      f'{"E(HF), eV":12s}')
print('-' * 70)

energies = {}

for symbol, spin, term, config in atoms_data:
    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        spin=spin,
        symmetry=True,
        verbose=0
    )
```

```

# Вибір методу SCF
if spin == 0:
    mf = scf.RHF(mol)
else:
    mf = scf.UHF(mol)

mf.verbose = 0
mf.conv_tol = 1e-10
energy = mf.kernel()
energies[symbol] = energy

e_ev = energy * 27.211386
print(f'{symbol:4s} {spin:4d} {term:4s} {energy:15.8f} {e_ev:12.2f}')

# Додаткова інформація для відкритих оболонок
if spin > 0:
    s2 = mf.spin_square()
    expected_s2 = spin * (spin + 2) / 4
    print(f'      <S²> = {s2[0]:.4f} (очікується {expected_s2:.4f})')

print('=' * 70)

# Збереження результатів
np.savez('second_period_hf.npz', **energies)

```

Коментарі до коду.

- Для кожного атома автоматично обирається тип SCF: RHF (для замкнених оболонок, $S = 0$) або UHF (для відкритих).
- Параметр `spin` у PySCF означає різницю між кількістю альфа- та бета-електронів: $\text{spin} = N_\alpha - N_\beta = 2S$.
- Розрахунок $\langle S^2 \rangle$ дозволяє перевірити правильність спінового стану. Для ідеального випадку значення має збігатися з теоретичним $S(S + 1)$.
- У файлі `second_period_hf.npz` зберігаються всі енергії для подальшого аналізу або побудови графіків.

Очікувані тенденції.

1. Повна енергія атома зменшується (стає більш негативною) із зростанням атомного номера Z .
2. Енергія спостерігає стрибки на межі заповнення оболонок: при переходах $\text{Be} \rightarrow \text{B}$, $\text{N} \rightarrow \text{O}$, $\text{F} \rightarrow \text{Ne}$.
3. Спінова мультиплетність відображає заповнення $2p$ -орбіталей згідно з **правилом Гунда** — максимальний спін у середині підоболонки (для N).
4. Значення $\langle S^2 \rangle$ для UHF повинні бути близькі до теоретичних, але можуть мати невеликі відхилення через *spin contamination*.

Фізичне узагальнення. Цей розрахунок демонструє фундаментальну властивість методу Гартрі-Фока: він добре описує загальну

структуру енергетичних рівнів і тенденції в періодичній таблиці, але не враховує електронну кореляцію, через що абсолютні значення енергії мають систематичну похибку.

Подальші кроки. Наступним логічним етапом є додавання пост-Гартрі-Фок методів (MP2, CI, CCSD), щоб показати, як кореляція електронів уточнює енергії і дозволяє досягати експериментальної точності.

1.4.4. Енергії іонізації. Іонізаційна енергія є однією з фундаментальних характеристик атома, що описує енерговитрати на видалення одного електрона. У квантово-хімічних розрахунках її визначають як різницю повних енергій катіона $E(A^+)$ і нейтрального атома $E(A)$:

$$I = E(A^+) - E(A)$$

де енергії E обчислюються в межах методу Гартрі-Фока або його узагальнень. Для ізольованих атомів другого періоду така процедура дозволяє отримати якісно правильну послідовність енергій іонізації, навіть попри те, що метод HF систематично недооцінює абсолютні значення через відсутність кореляційних ефектів.

```
from pyscf import gto, scf

def ionization_energy(symbol, charge_neutral=0, spin_neutral=None,
                      spin_cation=None, basis='cc-pvtz'):
    """
    Розрахунок енергії іонізації атома
    IE = E(A+) - E(A)
    """
    # Нейтральний атом
    mol_neutral = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        charge=charge_neutral,
        spin=spin_neutral,
        verbose=0
    )

    if spin_neutral == 0:
        mf_neutral = scf.RHF(mol_neutral)
    else:
        mf_neutral = scf.UHF(mol_neutral)

    mf_neutral.verbose = 0
    e_neutral = mf_neutral.kernel()

    # Катіон
    mol_cation = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        charge=charge_neutral + 1,
        spin=spin_cation,
        verbose=0
    )
```

```

)

if spin_cation == 0:
    mf_cation = scf.RHF(mol_cation)
else:
    mf_cation = scf.UHF(mol_cation)

mf_cation.verbose = 0
e_cation = mf_cation.kernel()

# Енергія іонізації
ie_ha = e_cation - e_neutral
ie_ev = ie_ha * 27.211386

return ie_ev, e_neutral, e_cation

# Розрахунок ІЕ для атомів 2-го періоду
atoms_ie = [
    ('Li', 1, 0), # Li → Li+
    ('Be', 0, 1), # Be → Be+
    ('B', 1, 0), # B → B+
    ('C', 2, 1), # C → C+
    ('N', 3, 2), # N → N+
    ('O', 2, 3), # O → O+
    ('F', 1, 2), # F → F+
    ('Ne', 0, 1), # Ne → Ne+
]

# Експериментальні значення (eV)
exp_ie = {
    'Li': 5.39, 'Be': 9.32, 'B': 8.30, 'C': 11.26,
    'N': 14.53, 'O': 13.62, 'F': 17.42, 'Ne': 21.56
}

print('Енергії іонізації атомів 2-го періоду')
print('=' * 65)
print(f'{"Атом":4s} {"IE(HF), eV":12s} {"IE(exp), eV":12s} {"Похибка, eV":12s}')
print('-' * 65)

for symbol, spin_n, spin_c in atoms_ie:
    ie_calc, e_n, e_c = ionization_energy(symbol, 0, spin_n, spin_c)
    ie_experimental = exp_ie[symbol]
    error = ie_calc - ie_experimental

    print(f'{symbol:4s} {ie_calc:12.2f} {ie_experimental:12.2f} '
          f'{error:12.2f}')

print('=' * 65)

```

Результатом буде таблиця іонізаційних енергій у електронвольтах. Порівнюючи їх з експериментальними, можна оцінити якість базисного набору та межі методу Гартрі-Фока. Для більш точних результатів використовують після-HF методи, наприклад MP2 або CCSD, які враховують електронну кореляцію.

1.5. Аналіз енергій та орбіталей

1.5.1. Енергетична діаграма орбіталей. Енергетичні діаграми орбіталей є наочним способом представлення енергетичного спектру

1. Метод Хартрі-Фока для атомів

молекулярних орбіталей (МО), що виникають у результаті розв'язання рівнянь Гартрі-Фока.

```
from pyscf import gto, scf
import matplotlib.pyplot as plt
import numpy as np

def plot_orbital_diagram(symbol, spin, basis='cc-pvdz'):
    """Побудова діаграми орбітальних енергій"""

    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        spin=spin,
        verbose=0
    )

    if spin == 0:
        mf = scf.RHF(mol)
    else:
        mf = scf.UHF(mol)

    mf.verbose = 0
    mf.kernel()

    fig, ax = plt.subplots(figsize=(8, 10))

    if spin == 0:
        # RHF: одна колонка орбіталей
        energies = mf.mo_energy * 27.211386 # eV
        n_occ = mol.nelec[0]

        for i, e in enumerate(energies[:n_occ+5]):
            color = 'blue' if i < n_occ else 'red'
            label = mol.ao_labels()[i] if i < len(mol.ao_labels()) else ''
            ax.hlines(e, 0.2, 0.8, colors=color, linewidth=2)
            ax.text(0.85, e, f'{label}', fontsize=10, va='center')

            # Стрілки для електронів
            if i < n_occ:
                ax.arrow(0.35, e+0.3, 0, -0.25, head_width=0.05,
                        head_length=0.1, fc='black', ec='black')
                ax.arrow(0.65, e-0.3, 0, 0.25, head_width=0.05,
                        head_length=0.1, fc='black', ec='black')
    else:
        # UHF: дві колонки (альфа та бета)
        e_alpha = mf.mo_energy[0] * 27.211386
        e_beta = mf.mo_energy[1] * 27.211386
        n_alpha, n_beta = mol.nelec

        # Альфа орбіталі
        for i, e in enumerate(e_alpha[:n_alpha+3]):
            color = 'blue' if i < n_alpha else 'red'
            ax.hlines(e, 0.1, 0.4, colors=color, linewidth=2)
            if i < n_alpha:
                ax.arrow(0.25, e+0.3, 0, -0.25, head_width=0.04,
                        head_length=0.1, fc='black', ec='black')

        # Бета орбіталі
        for i, e in enumerate(e_beta[:n_beta+3]):
            color = 'blue' if i < n_beta else 'red'
            ax.hlines(e, 0.6, 0.9, colors=color, linewidth=2)
            if i < n_beta:
```

```

ax.arrow(0.75, e-0.3, 0, 0.25, head_width=0.04,
         head_length=0.1, fc='black', ec='black')

ax.text(0.25, min(e_alpha[:5])-2, 'α', fontsize=14, ha='center')
ax.text(0.75, min(e_beta[:5])-2, 'β', fontsize=14, ha='center')

ax.set_xlim(0, 1)
ax.set_ylabel('Енергія (eV)', fontsize=12)
ax.set_title(f'Діаграма орбіталей {symbol}', fontsize=14)
ax.set_xticks([])
ax.grid(True, alpha=0.3, axis='y')

plt.tight_layout()
plt.savefig(f'{symbol}_orbital_diagram.pdf')
plt.show()

# Приклади
plot_orbital_diagram('C', spin=2) # Вуглець
plot_orbital_diagram('Ne', spin=0) # Неон

```

1.5.2. Розподіл електронної густини. Розподіл електронної густини $\rho(\mathbf{r})$ є однією з ключових величин у квантовій хімії. Саме ця функція визначає, де саме в просторі «перебувають» електрони атома або молекули, і, отже, пояснює більшість хімічних і спектроскопічних властивостей системи.

Згідно з однодетермінантним наближенням Гартрі-Фока, електронна густина визначається як

$$\rho(\mathbf{r}) = \sum_i^{\text{occ}} |\psi_i(\mathbf{r})|^2,$$

де $\psi_i(\mathbf{r})$ — орбіталі, а сума береться по всіх зайнятих орбіталях.

У програмному пакеті PySCF густина може бути обчислена як на сітці, так і в окремих точках простору, що дозволяє візуалізувати просторовий розподіл електронів.

Обчислення густини вздовж осі z

У наведеному прикладі реалізовано функцію `plot_density_profile`, що обчислює електронну густина вздовж осі z для атома. Це дозволяє побудувати одноосний «профіль густини» та простежити, як електронна густина спадає при віддаленні від ядра.

- Молекулярний об'єкт створюється через `gto.M`, де вказується атом, базис та спінова мультиплічність.
- Виконується розрахунок за методом Гартрі-Фока (RHF або UHF залежно від спіну).
- Для кожної точки z обчислюється матриця базисних функцій $\varphi_i(\mathbf{r})$, і далі густина:

$$\rho(\mathbf{r}) = \sum_{ij} \varphi_i(\mathbf{r}) D_{ij} \varphi_j(\mathbf{r}),$$

де D_{ij} — елементи матриці щільності.

1. Метод Хартрі-Фока для атомів

Результат зображається графічно: густина $\rho(0, 0, z)$ як функція відстані від ядра. Для нейтральних атомів крива має різкий максимум біля ядра, який швидко спадає експоненційно.

```
from pyscf import gto, scf
from pyscf.tools import cubegen
import numpy as np
import matplotlib.pyplot as plt

def plot_density_profile(symbol, spin, basis='cc-pvdz'):
    """Радіальний розподіл густини"""

    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        spin=spin,
        verbose=0
    )

    if spin == 0:
        mf = scf.RHF(mol)
    else:
        mf = scf.UHF(mol)

    mf.verbose = 0
    mf.kernel()

    # Розрахунок густини вздовж осі z
    r_points = np.linspace(0, 5, 200) # Bohr
    density = []

    dm = mf.make_rdm1()

    for r in r_points:
        coords = np.array([[0, 0, r]])
        rho = mol.eval_gto('GTOval_sph', coords)

        if spin == 0:
            dens = np.einsum('pi,ij,pj->p', rho, dm, rho)[0]
        else:
            dens_a = np.einsum('pi,ij,pj->p', rho, dm[0], rho)[0]
            dens_b = np.einsum('pi,ij,pj->p', rho, dm[1], rho)[0]
            dens = dens_a + dens_b

        density.append(dens)

    # Побудова графіка
    plt.figure(figsize=(10, 6))
    plt.plot(r_points, density, linewidth=2)
    plt.xlabel('Відстань від ядра (Bohr)', fontsize=12)
    plt.ylabel('Електронна густина (e/Bohr³)', fontsize=12)
    plt.title(f'Радіальний розподіл густини {symbol}', fontsize=14)
    plt.grid(True, alpha=0.3)
    plt.tight_layout()
    plt.savefig(f'{symbol}_density.pdf')
    plt.show()

    return r_points, density

# Приклад
r, rho = plot_density_profile('Ne', spin=0)
```

Інтерпретація результатів. На побудованому графіку $\rho(r)$ спостерігаються чіткі області локалізації електронів: внутрішні електрони формують центральний пік, тоді як зовнішні оболонки проявляються у вигляді плавних плечей. Для важчих атомів густина стає більш «зосередженою» біля ядра через сильніше притягання кулонівським потенціалом.

Практична порада. При виборі базисного набору (cc-pvdz, 6-31G**, тощо) слід мати на увазі, що форма профілю густини істотно залежить від якості базису: мінімальні базиси дають лише грубу картину, тоді як розширені базиси (cc-pVTZ) відтворюють експоненційний спад більш точно.

Сферично усереднена густина

Для атомів, що мають сферичну симетрію, зручно розглядати усереднену по всіх напрямках густину:

$$\rho(r) = \frac{1}{4\pi} \int \rho(\mathbf{r}) d\Omega,$$

де $r = |\mathbf{r}|$ та $d\Omega$ — елемент тілесного кута.

Ця функція показує, як змінюється густина лише від радіуса r , незалежно від напрямку, і є основою для побудови радіальної функції розподілу

$$P(r) = 4\pi r^2 \rho(r),$$

що описує, яка частка електронів перебуває у сферичному шарі товщини dr на відстані r від ядра.

```
from pyscf import gto, scf
import numpy as np
import matplotlib.pyplot as plt

def spherical_averaged_density(symbol, spin, basis='cc-pvdz'):
    """Сферично усереднена електронна густина"""

    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        spin=spin,
        verbose=0
    )

    if spin == 0:
        mf = scf.RHF(mol)
    else:
        mf = scf.UHF(mol)

    mf.verbose = 0
    mf.kernel()
```

```

dm = mf.make_rdm1()

# Радіальні точки
r_points = np.linspace(0.01, 8, 300)

# Куты для інтегрування (метод Монте-Карло)
n_angles = 100
theta = np.random.uniform(0, np.pi, n_angles)
phi = np.random.uniform(0, 2*np.pi, n_angles)

density_sph = []

for r in r_points:
    rho_avg = 0

    for t, p in zip(theta, phi):
        x = r * np.sin(t) * np.cos(p)
        y = r * np.sin(t) * np.sin(p)
        z = r * np.cos(t)

        coords = np.array([x, y, z])
        ao_value = mol.eval_gto('GTOval_sph', coords)

        if spin == 0:
            rho_point = np.einsum('pi,ij,pj->p',
                                   ao_value, dm, ao_value)[0]
        else:
            rho_a = np.einsum('pi,ij,pj->p',
                               ao_value, dm[0], ao_value)[0]
            rho_b = np.einsum('pi,ij,pj->p',
                               ao_value, dm[1], ao_value)[0]
            rho_point = rho_a + rho_b

        rho_avg += rho_point

    rho_avg /= n_angles
    density_sph.append(rho_avg)

# Радіальна функція розподілу  $4\pi r^2 \rho(r)$ 
radial_dist = 4 * np.pi * r_points**2 * np.array(density_sph)

# Графіки
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 5))

# Густина  $\rho(r)$ 
ax1.plot(r_points, density_sph, linewidth=2)
ax1.set_xlabel('r (Bohr)', fontsize=12)
ax1.set_ylabel('ρ(r) (e/Bohr³)', fontsize=12)
ax1.set_title(f'Електронна густина {symbol}', fontsize=14)
ax1.grid(True, alpha=0.3)
ax1.set_yscale('log')

# Радіальна функція розподілу
ax2.plot(r_points, radial_dist, linewidth=2, color='red')
ax2.set_xlabel('r (Bohr)', fontsize=12)
ax2.set_ylabel('4πr²ρ(r)', fontsize=12)
ax2.set_title(f'Радіальна функція розподілу {symbol}',
              fontsize=14)
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig(f'{symbol}_spherical_density.pdf')
plt.show()

```

```

# Перевірка нормування
dr = r_points[1] - r_points[0]
total_electrons = np.sum(radial_dist) * dr
print(f'\nІнтеграл  $4\pi r^2 \rho(r) dr$  = {total_electrons:.2f}')
print(f'Очікується: {mol.nelectron} електронів')

return r_points, density_sph, radial_dist

# Приклади для різних атомів
r, rho, rdf = spherical_averaged_density('He', spin=0)
r, rho, rdf = spherical_averaged_density('C', spin=2)
r, rho, rdf = spherical_averaged_density('Ne', spin=0)

```

Методика обчислення. Для кожного радіуса r вибираються випадкові напрями (кутові координати θ , φ) за методом Монте-Карло. У цих напрямках обчислюється густина $\rho(x, y, z)$, після чого береться середнє значення:

$$\rho(r) \approx \frac{1}{N} \sum_{k=1}^N \rho(r, \theta_k, \varphi_k).$$

Цей підхід забезпечує добру точність навіть при невеликій кількості напрямів $N \sim 100$.

Радіальна функція розподілу. На другому графіку зображується $4\pi r^2 \rho(r)$ — це функція, інтеграл від якої по r дає повну кількість електронів:

$$\int_0^\infty 4\pi r^2 \rho(r) dr = N_e.$$

Таким чином, можна перевірити нормування хвильової функції та коректність чисельного інтегрування.

Приклад інтерпретації. Для атома гелію максимум $4\pi r^2 \rho(r)$ спостерігається приблизно при $r \approx 0.3$ Bohr, що відповідає найбільш ймовірній відстані електрона від ядра у $1s$ -стані. Для вуглецю або неону виникають додаткові піки, пов'язані з електронами на $2s$ та $2p$ оболонках.

Перевірка нормування. Наприкінці виконання функції виводиться значення

$$\int 4\pi r^2 \rho(r) dr,$$

яке має збігатися з кількістю електронів у системі, що є корисним тестом правильності побудованої густини.

1.5.3. Порівняння електронних густин різних атомів. Ми розглянули, як отримати електронну густину $\rho(\mathbf{r})$ та сферично усереднену густину $\rho(r)$ для окремого атома. Однак для повного розуміння періодичних закономірностей важливо порівняти, як змінюється форма та протяжність електронної хмари при переході від легких до важчих елементів.

Фізична мотивація

Порівняння електронних густин дозволяє:

- побачити, як із зростанням атомного номера Z ядро сильніше притягує електрони;
- проаналізувати зміну розмірів атома та ефективного радіуса;
- виявити закономірності заповнення електронних оболонок;
- візуально оцінити зв'язок між структурою густини та положенням елемента в періодичній системі.

У програмі нижче реалізовано функцію порівняння радіальних профілів густини для кількох атомів одночасно. Вона виконує незалежні розрахунки SCF для кожного атома, будує $\rho(r)$ та радіальну функцію $4\pi r^2 \rho(r)$, і виводить результати на одному графіку для порівняння.

Коментар до реалізації

Ключові кроки програми:

1. Створення об'єкта Mole для кожного атома з базисом cc-pvdz або cc-pvtz.
2. Проведення SCF-розрахунку типу RHF або UHF залежно від спіну.
3. Побудова масиву радіальних точок уздовж осі z :

$$r \in [0, 6] \text{ Bohr}$$

4. Обчислення електронної густини як:

$$\rho(r) = \sum_{i,j} \varphi_i(r) D_{ij} \varphi_j(r),$$

де φ_i — атомні орбіталі, D_{ij} — елементи матриці щільності.

5. Для кожного атома будується:
 - крива $\rho(r)$ — логарифмічний масштаб показує різке спадання густини;
 - радіальний розподіл $4\pi r^2 \rho(r)$ — для аналізу оболонкової структури.

```

import numpy as np
import matplotlib.pyplot as plt

def compare_densities(atoms_list, basis='cc-pvdz'):
    """Порівняння електронної густини різних атомів"""

    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 5))

    colors = plt.cm.tab10(np.linspace(0, 1, len(atoms_list)))

    for (symbol, spin), color in zip(atoms_list, colors):
        mol = gto.M(
            atom=f'{symbol} 0 0 0',
            basis=basis,
            spin=spin,
            verbose=0
        )

        if spin == 0:
            mf = scf.RHF(mol)
        else:
            mf = scf.UHF(mol)

        mf.verbose = 0
        mf.kernel()

        dm = mf.make_rdm1()

        # Густина вздовж осі z
        r_points = np.linspace(0.01, 6, 200)
        density = []

        for r in r_points:
            coords = np.array([[0, 0, r]])
            ao = mol.eval_gto('GT0val_sph', coords)

            if spin == 0:
                rho = np.einsum('pi,ij,pj->p', ao, dm, ao)[0]
            else:
                rho = (np.einsum('pi,ij,pj->p', ao, dm[0], ao)[0] +
                       np.einsum('pi,ij,pj->p', ao, dm[1], ao)[0])

            density.append(rho)

        # Графіки
        ax1.plot(r_points, density, linewidth=2,
                 label=f'{symbol} (Z={mol.atom_charge(0)})',
                 color=color)

        # Радіальна функція
        radial = 4 * np.pi * r_points**2 * np.array(density)
        ax2.plot(r_points, radial, linewidth=2,
                 label=f'{symbol}', color=color)

    ax1.set_xlabel('r (Bohr)', fontsize=12)
    ax1.set_ylabel('ρ(r) (e/Bohr³)', fontsize=12)
    ax1.set_title('Електронна густина', fontsize=14)
    ax1.set_yscale('log')
    ax1.legend()
    ax1.grid(True, alpha=0.3)

    ax2.set_xlabel('r (Bohr)', fontsize=12)
    ax2.set_ylabel('4πr²ρ(r)', fontsize=12)
    ax2.set_title('Радіальний розподіл', fontsize=14)
    ax2.legend()

```

```
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig('atoms_density_comparison.pdf')
plt.show()

# Порівняння атомів благородних газів
noble_gases = [
    ('He', 0), ('Ne', 0), ('Ar', 0)
]
compare_densities(noble_gases, basis='cc-pvtz')

# Порівняння атомів 2-го періоду
second_period = [
    ('Li', 1), ('C', 2), ('N', 3), ('O', 2), ('F', 1)
]
compare_densities(second_period, basis='6-31g*')
```

Аналіз результатів

На графіку для **благородних газів** (He, Ne, Ar) чітко видно:

- з ростом Z максимум густини зміщується ближче до ядра — електрони сильніше притягуються кулонівським полем;
- протяжність хвоста $\rho(r)$ зменшується, атоми стають компактнішими;
- радіальні криві $4\pi r^2 \rho(r)$ демонструють послідовне збільшення кількості максимумів, що відповідає заповненню нових оболонок.

Для **атомів другого періоду** (Li – F) спостерігається:

- різке зростання центральної густини у міру збільшення Z ;
- поява другорядного максимуму в $4\pi r^2 \rho(r)$, що відповідає електронам $2s$ -оболонки;
- звуження зовнішньої частини електронної хмари.

Зв'язок із електронними оболонками

Максимуми радіальної функції розподілу $4\pi r^2 \rho(r)$ відповідають областям найбільшої ймовірності перебування електронів певних орбіталей:

$1s \rightarrow$ перший максимум, $2s, 2p \rightarrow$ другий максимум,
 $3s, 3p, 3d \rightarrow$ третій тощо.

Таким чином, форма $4\pi r^2 \rho(r)$ дає прямий зв'язок між результатами квантово-хімічних розрахунків і традиційною атомною структурою, знайомою студентам з курсу атомної фізики. Візуальний аналіз таких графіків дозволяє простежити закономірності побудови періодичної системи Менделєєва з точки зору електронної густини.

Практичні завдання

1. Побудуйте на одному графіку $\rho(r)$ для Li, Na, K. Як змінюється радіус атома?
2. Знайдіть положення максимумів $4\pi r^2 \rho(r)$ для атомів He, Ne, Ar. До яких оболонок вони належать?
3. Порівняйте радіальні функції для атомів C і O. Як змінюється густина зовнішньої оболонки?

Методичні рекомендації

1. Для легких атомів (H, He, Li) можна порівняти результати з аналітичними розв'язками рівняння Шредінгера.
2. Змінюючи базис (наприклад, STO-3G, 6-31G, cc-pVTZ), можна дослідити, як збільшується точність відтворення радіального профілю.
3. Для відкритих оболонок (атомів з неспареними електронами) потрібно враховувати спінову поляризацію — використовується UHF.

Таким чином, наведені приклади демонструють практичний зв'язок між теоретичними поняттями електронної густини і їх чисельною реалізацією в пакеті PySCF.

1.6. Порівняння методів: RHF vs UHF vs ROHF

У квантово-хімічних розрахунках вибір типу наближення Гартрі-Фока залежить від спінового стану системи. Методи **RHF**, **UHF** та **ROHF** відрізняються тим, як вони поведуться із спіновими функціями електронів — тобто, чи допускають різні орбіталі для α - і β -електронів.

- **RHF (Restricted Hartree-Fock)** — усі електрони спарені, кожна просторова орбіталь заповнена двома електронами з протилежними спінами. Підходить лише для синглетних закритих оболонок.
- **UHF (Unrestricted Hartree-Fock)** — орбіталі для α - і β -електронів дозволено різні. Цей метод придатний для відкритих оболонок (наприклад, атомів з неспареними електронами), однак може страждати від *спінового забруднення*.
- **ROHF (Restricted Open-Shell Hartree-Fock)** — компроміс: спільні орбіталі для парних електронів і різні — лише для неспарених. Це забезпечує правильний спіновий симетрійний стан без забруднення.

1.6.1. Тестовий випадок: атом Вуглецю. Вуглець має конфігурацію $1s^2 2s^2 2p^2$ і основний стан 3P (триплет). Отже, система має два неспарені електрони ($S = 1$), а отже $\langle S^2 \rangle = S(S + 1) = 2.0$.

1. Метод Хартрі-Фока для атомів

Далі наведено приклад прямого порівняння результатів методів UHF і ROHF для атома С у триплетному стані з використанням базису cc-pVTZ.

```
from pyscf import gto, scf

# Атом С: [He] 2s2 2p2, основний стан 3P
basis = 'cc-pvtz'

print('Порівняння методів для атома Вуглецю (3P)')
print('='*70)

# Молекула
mol = gto.M(
    atom='C 0 0 0',
    basis=basis,
    spin=2, # Триплет
    symmetry=True,
    verbose=0
)

# UHF
print('\n1. Unrestricted Hartree-Fock (UHF)')
print('='*70)
mf_uhf = scf.UHF(mol)
mf_uhf.verbose = 4
e_uhf = mf_uhf.kernel()

s2_uhf = mf_uhf.spin_square()
print(f'\nЕнергія (UHF): {e_uhf:.8f} Ha')
print(f'<S2> (UHF): {s2_uhf[0]:.6f} (очікується 2.0)')
print(f'Забруднення спіном: {s2_uhf[0] - 2.0:.6f}')

# ROHF
print('\n2. Restricted Open-shell Hartree-Fock (ROHF)')
print('='*70)
mf_rohf = scf.ROHF(mol)
mf_rohf.verbose = 4
e_rohf = mf_rohf.kernel()

s2_rohf = mf_rohf.spin_square()
print(f'\nЕнергія (ROHF): {e_rohf:.8f} Ha')
print(f'<S2> (ROHF): {s2_rohf[0]:.6f} (очікується 2.0)')

# Порівняння
print('\n' + '='*70)
print('ПОРІВНЯННЯ')
print('='*70)
print(f'ΔE (UHF-ROHF): {(e_uhf - e_rohf)*1000:.4f} mHa')
print(f'ΔE (UHF-ROHF): {(e_uhf - e_rohf)*627.509:.4f} kcal/mol')

print(f'\nЗабруднення спіном:')
print(f'  UHF: {s2_uhf[0] - 2.0:.6f}')
print(f'  ROHF: {s2_rohf[0] - 2.0:.6f}')

# Орбітальні енергії
print(f'\nОрбітальні енергії (Ha):')
print(f'{"Орбіталь":15s} {"UHF (α)":12s} {"UHF (β)":12s} {"ROHF":12s}')
print('='*70)

n_show = 5
for i in range(n_show):
    label = mol.ao_labels()[i] if i < len(mol.ao_labels()) else f'MO{i+1}'
```

```
e_uhf_a = mf_uhf.mo_energy[0][i]
e_uhf_b = mf_uhf.mo_energy[1][i]
e_rohf_i = mf_rohf.mo_energy[i]

print(f'{label:15s} {e_uhf_a:12.6f} {e_uhf_b:12.6f} {e_rohf_i:12.6f}')
```

1.6.2. Теоретичний аналіз результатів.

Спінове забруднення

Для триплетного стану очікується:

$$\langle S^2 \rangle = S(S + 1) = 1(1 + 1) = 2.0.$$

Якщо результат UHF дає $\langle S^2 \rangle > 2.0$, це свідчить про *спінове забруднення* — хвильова функція містить домішку інших спінових станів (наприклад, 5S). Це відбувається через те, що UHF не примушує α - і β -орбіталі бути ортогональними, тому спінова симетрія порушується.

ROHF, на відміну від цього, зберігає точне спінове квантове число, тому $\langle S^2 \rangle$ збігається з теоретичним значенням.

Енергетичне порівняння

UHF зазвичай дає трохи нижчу енергію, ніж ROHF, оскільки має більше варіаційної свободи. Різниця $\Delta E = E_{\text{UHF}} - E_{\text{ROHF}}$ зазвичай незначна (менше кількох мілігаусів), однак у системах з сильним спіновим змішуванням може бути суттєвою.

Орбітальні енергії

UHF формує два набори орбіталей — для α - і β -електронів. Тому орбітальні енергії можуть відрізнитись:

$$\varepsilon_i^{(\alpha)} \neq \varepsilon_i^{(\beta)}.$$

У ROHF усі парні орбіталі спільні, тож орбітальні енергії чітко впорядковані відповідно до симетрії та спіну.

1.6.3. Систематичне порівняння для всіх атомів. Щоб узагальнити різницю між методами UHF та ROHF, виконаємо серію розрахунків для кількох відкритооболонкових атомів перших двох періодів. Це дозволить оцінити енергетичну різницю $\Delta E = E_{\text{UHF}} - E_{\text{ROHF}}$ і побачити, чи зберігає ROHF спінову симетрію без втрати точності.

```
from pyscf import gto, scf
import numpy as np

atoms_open_shell = [
    ('H', 1), ('Li', 1), ('B', 1), ('C', 2),
    ('N', 3), ('O', 2), ('F', 1)
]

basis = '6-31g*'

print(f'Порівняння UHF vs ROHF (базис: {basis})')
print('='*80)
print(f'{"Атом":4s} {"Спін":4s} {"E(UHF), Ha":15s} {"E(ROHF), Ha":15s} '
      f'{"ΔE, mHa":10s}')
print('-'*80)

for symbol, spin in atoms_open_shell:
    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
        spin=spin,
        verbose=0
    )

    # UHF
    mf_uhf = scf.UHF(mol)
    mf_uhf.verbose = 0
    mf_uhf.conv_tol = 1e-10
    e_uhf = mf_uhf.kernel()

    # ROHF
    mf_rohf = scf.ROHF(mol)
    mf_rohf.verbose = 0
    mf_rohf.conv_tol = 1e-10
    e_rohf = mf_rohf.kernel()

    delta = (e_uhf - e_rohf) * 1000 # mHa

    print(f'{symbol:4s} {spin:4d} {e_uhf:15.8f} {e_rohf:15.8f} {delta:10.4f}')

print('='*80)
print('\nПримітка: UHF зазвичай дає нижчу енергію, але з')
print('        забрудненням спіну. ROHF зберігає чисту')
print('        спінову симетрію.')
```

Інтерпретація результатів. Як правило, для легких атомів (H, Li, B) енергії UHF і ROHF майже збігаються. Для атомів із більшою кількістю незапарених електронів (C, N, O, F) різниця у кілька мілігаусів відображає більшу гнучкість UHF, що дозволяє частково знизити енергію ціною забруднення спіном ($\langle S^2 \rangle > S(S+1)$).

Таким чином, ROHF — це більш строгий метод із правильною спіноювою симетрією, а UHF — енергетично вигідніший, але менш «фізично чистий». У практичних обчисленнях ROHF часто слугує базовою точкою для подальших кореляційних методів (MP2, CCSD, CASSCF).

Практичне значення

Порівняння методів RHF, UHF і ROHF дозволяє студентам зрозуміти:

- коли можна використовувати RHF (синглети закритих оболонок);
- коли необхідно застосовувати UHF (відкриті оболонки, магнітні системи);
- у яких випадках варто віддати перевагу ROHF (спінова чистота, мінімізація забруднення).

Контрольне завдання

1. Повторіть розрахунок для атомів O (спін = 2) і N (спін = 3). Порівняйте $\langle S^2 \rangle$ для UHF і ROHF.
2. Для кожного атома побудуйте різницю енергій UHF–ROHF у мілігаусах.
3. Проаналізуйте, як спінове забруднення зростає зі збільшенням кількості неспарених електронів.

1.7. Складні випадки та збіжність

Розрахунки у квантовій хімії не завжди проходять гладко. Навіть для, здавалося б, простих атомів або малих молекул можуть виникати проблеми збіжності — коли ітераційний процес самозгодженого поля (SCF) не може досягти стабільного рішення. Особливо це стосується перехідних металів, систем із сильною кореляцією електронів або випадків із виродженими орбіталями. У цьому розділі наведено практичні прийоми, які допомагають стабілізувати розрахунок, а також приклади програмного коду на базі бібліотеки PySCF.

Основні джерела труднощів:

- сильна електронна кореляція в незаповнених d - та f -оболонках;
- мала різниця енергій між орбіталями (виродження);
- вибір невдалого початкового наближення хвильової функції;
- надто жорсткі або надто м'які критерії збіжності.

У наступних підрозділах ми розглянемо конкретні стратегії для подолання таких труднощів.

1.7.1. Перехідні метали. Перехідні метали є одними з найпроблемніших систем для збіжності SCF. Причини — сильна кореляція електронів d -оболонки, множинність близьких за енергією станів спіну та значна роль релятивістичних ефектів. Навіть у межах методу Гартрі-Фока функціонал енергії має кілька локальних мінімумів, і звичайна ітераційна процедура може «застрягнути» в некоректному стані.

У PySCF це часто проявляється у вигляді:

- осциляцій енергії між ітераціями,
- дивергентного DIIS,
- різкої зміни спінового моменту між кроками.

Типовий приклад — атоми Fe, Co, Ni або їхні оксиди, де різниця між конфігураціями високого й низького спіну становить лише декілька сотих гартрі.

```
from pyscf import gto, scf

mol = gto.M(atom='Co 0 0 0', basis='def2-svp', spin=3)
mf = scf.UHF(mol).run()
print('Converged?', mf.converged)
```

Перехідні метали є класичним прикладом систем, де звичайні методи SCF часто не конвергують або дають фізично некоректний результат. Це зумовлено тим, що у таких атомів одночасно заповнюються 3d- та 4s-орбіталі, енергії яких близькі. У результаті електрон може “перестрибувати” між орбіталями під час ітерацій, порушуючи стабільність процесу.

У прикладі нижче подано універсальну функцію для обчислення енергії атомів перехідних металів із урахуванням практичних порад:

- використовується метод UHF (неспарені електрони важливі для коректного опису спіну);
- уведено level shift, що зміщує енергетичний спектр і полегшує збіжність;
- застосовано розширений DIIS-простір (метод прискорення збіжності);
- при необхідності переходять до Ньютона–Рафсона для уточнення рішення;
- результати контролюються через значення $\langle S^2 \rangle$.

Після кожного розрахунку порівнюється очікуване значення спіну зі знайденим, а також обчислюється “забруднення спіном” — показник, наскільки рішення відхиляється від ідеального стану.

```
from pyscf import gto, scf

def transition_metal_calculation(symbol, spin, basis='def2-svp'):
    """
    Розрахунок атома перехідного металу
    Часто потребує спеціальних налаштувань
    """

    print(f'\nРозрахунок {symbol} (2S={spin})')
    print('='*60)

    mol = gto.M(
        atom=f'{symbol} 0 0 0',
        basis=basis,
```

```

    spin=spin,
    symmetry=False, # Іноді краще без симетрії
    verbose=0
)

mf = scf.UHF(mol)

# Налаштування для важких випадків
mf.conv_tol = 1e-8
mf.max_cycle = 200
mf.level_shift = 0.5 # Level shift допомагає конвергенції
mf.diis_space = 12
mf.init_guess = 'atom'

print('Спроба 1: UHF з level shift...')
mf.verbose = 4
energy = mf.kernel()

if not mf.converged:
    print('\nНе конвергувало! Спроба 2: Newton-Raphson...')
    mf = mf.newton()
    mf.max_cycle = 50
    energy = mf.kernel()

if mf.converged:
    s2 = mf.spin_square()
    expected_s2 = spin * (spin + 2) / 4

    print(f'\nРезультати:')
    print(f'  Енергія: {energy:.8f} Ha')
    print(f'  <S²>: {s2[0]:.4f} (очікується {expected_s2:.4f})')
    print(f'  Забруднення: {s2[0] - expected_s2:.4f}')
else:
    print('\nНЕ ВДАЛОСЯ ДОСЯГТИ КОНВЕРГЕНЦІЇ!')

return mf, energy if mf.converged else None

# Приклади перехідних металів
# Sc: [Ar] 3d¹ 4s², ²D
mf_sc, e_sc = transition_metal_calculation('Sc', spin=1)

# Ti: [Ar] 3d² 4s², ³F
mf_ti, e_ti = transition_metal_calculation('Ti', spin=2)

# Cr: [Ar] 3d⁵ 4s¹, ⁷S
mf_cr, e_cr = transition_metal_calculation('Cr', spin=6)

# Mn: [Ar] 3d⁵ 4s², ⁶S
mf_mn, e_mn = transition_metal_calculation('Mn', spin=5)

# Fe: [Ar] 3d⁶ 4s², ⁵D
mf_fe, e_fe = transition_metal_calculation('Fe', spin=4)

```

Такий підхід забезпечує стабільність навіть для важких атомів — наприклад, Cr, Mn або Fe, де стандартний SCF часто не сходиться.

1.7.2. Використання дробових заповнень. Ще одна причина проблем зі збіжністю — наявність вироджених або майже вироджених орбіталей. У таких випадках система “вагається”, яку саме конфігурацію обрати, і стандартний SCF може осцилювати без досягнення

стабільного мінімуму.

Один із ефективних підходів — застосування *дробових заповнень орбіталей* (англ. *fractional occupations*). Ідея полягає у тому, що замість жорсткого заповнення 0 або 1 для кожної орбіталі вводиться часткове заповнення, яке імітує теплове розмазування Фермі-Дірака:

$$f_i = \frac{1}{1 + e^{(\varepsilon_i - \mu)/kT}},$$

де f_i — ефективне заповнення i -ї орбіталі, ε_i — її енергія, μ — хімічний потенціал, а kT — параметр “розмазування”.

Цей підхід згладжує різкі переходи між виродженими орбіталями і дозволяє SCF знайти стабільне рішення.

Нижче наведено приклад для атома ванадію, у якого типово спостерігаються проблеми збіжності через близькість енергетичних рівнів 3d і 4s.

```
from pyscf import gto, scf

# Важкий випадок: атом з близькими за енергією орбіталями
mol = gto.M(
    atom='V 0 0 0',
    basis='cc-pvdz',
    spin=3,
    verbose=0
)

print('Розрахунок V з дробовими заповненнями')
print('='*60)

# Стандартний UHF може не конвергувати
mf = scf.UHF(mol)
mf.verbose = 0
mf.max_cycle = 100

try:
    energy = mf.kernel()
    if not mf.converged:
        raise RuntimeError('Не конвергувало')
except:
    print('Стандартний UHF не конвергував')

# Використання дробових заповнень (smearing)
print('\nСпроба з дробовими заповненнями...')

mf = scf.UHF(mol)
mf = scf.addons.frac_occ(mf)
mf.verbose = 4
energy = mf.kernel()

if mf.converged:
    print(f'\nУспішно! Енергія: {energy:.8f} Ha')
else:
    print('\nВсе одно не конвергувало')
```

Як видно, якщо звичайний UHF не конвергує, можна скористатися `scf.addons.frac_occ(mf)`, що вмикає дробові заповнення.

Методика добре працює для окремих атомів, радикалів і навіть для малих кластерів перехідних металів.

1.7.3. Стратегії досягнення конвергенції. Проблема збіжності — одна з найчастіших у практиці квантово-хімічних розрахунків. Ітераційна процедура самозгодженого поля (SCF) повинна забезпечити стабільне розв’язання рівнянь Гартрі-Фока, однак у реальних системах — особливо для відкритих оболонок, перехідних металів або вироджених станів — енергія може коливатися, DIIS може втрачати стабільність, а розрахунок сходиться до нефізичного мінімуму.

Для таких випадків доцільно застосовувати *послідовну стратегію стабілізації SCF*, у якій різні методи збільшення стійкості перевіряються крок за кроком — від простих до складніших.

Основні прийоми стабілізації:

1. стандартний UHF;
2. зсув рівнів (level_shift), що зменшує вплив віртуальних орбіталей;
3. збільшення DIIS-простору (diis_space);
4. атомне початкове наближення (init_guess='atom');
5. уточнення за методом Ньютона-Рафсона;
6. дробові заповнення (frac_occ) при вироджених станах.

Нижче подано узагальнений алгоритм, який автоматично перебирає ці стратегії для пошуку збіжного рішення.

```
from pyscf import gto, scf

def convergence_strategies(symbol, spin, basis='def2-svp'):
    """
    Покрокова стабілізація SCF для складних систем
    """
    mol = gto.M(atom=f'{symbol} 0 0 0', basis=basis, spin=spin, verbose=0)

    strategies = [
        ('Стандартний UHF', {}),
        ('UHF з level shift', {'level_shift': 0.3}),
        ('UHF з більшим DIIS', {'diis_space': 15}),
        ('UHF з атомним guess', {'init_guess': 'atom'}),
        ('Метод Ньютона-Рафсона', {'newton': True}),
        ('Дробові заповнення', {'frac_occ': True}),
    ]

    print(f'=== Тестування стратегій конвергенції для {symbol} ===')

    for name, params in strategies:
        print(f'\n→ {name}')
        mf = scf.UHF(mol)
        mf.max_cycle = 100
        mf.conv_tol = 1e-9

        for k, v in params.items():
            if k in ('newton', 'frac_occ'):
                continue
```

```
setattr(mf, k, v)

try:
    if params.get('newton'):
        mf = mf.newton()
    if params.get('frac_occ'):
        mf = scf.addons.frac_occ(mf)
    energy = mf.kernel()
    if mf.converged:
        print(f'Успіх: E = {energy:.8f} Ha')
        return mf, energy
    else:
        print('Не конвергувало')
except Exception as e:
    print(f'Помилка: {e}')
print('\nЖодна стратегія не спрацювала.')
return None, None

# Приклад: перехідний метал
mf, e = convergence_strategies('Co', spin=3)
```

Фізичний сенс прийомів.

- **Level shift.** Тимчасово підвищує енергії віртуальних орбіталей, запобігаючи коливанням поля.
- **DIIS-простір.** Збільшення пам'яті історії ітерацій покращує апроксимацію самозгодженого стану.
- **Atom guess.** Використовує атомні орбіталі як старт, що ближче до фізичного рішення.
- **Newton–Raphson.** Прискорює збіжність у зоні поблизу мінімуму функціоналу енергії.
- **Fractional occupations.** Згладжують виродження, дозволяючи системі знайти стабільний середній стан.

Практичні рекомендації.

- Для великих систем зменшіть точність (`conv_tol=1e-6`) для попереднього тесту.
- Якщо спостерігаються осциляції енергії — додайте `level_shift=0.5`.
- Якщо UHF не сходиться — спробуйте `init_guess='atom'` або `symmetry=False`.
- Завжди перевіряйте фізичність розв'язку через $\langle S^2 \rangle$.

Таким чином, навіть якщо стандартний SCF не збігається, послідовне застосування описаних методів практично гарантує знаходження стабільного фізично коректного рішення.

1.8. Практичні завдання

1.8.1. Завдання 1: Систематичне дослідження.

ЗАВДАННЯ 1: Розрахуйте енергії всіх атомів першого періоду

(H, He) з різними базисними наборами. Побудуйте графік збіжності енергії до базисної межі.

Базиси: sto-3g, 6-31g, cc-pvdz, cc-pvtz, cc-pvqz, cc-pv5z

```
from pyscf import gto, scf
import matplotlib.pyplot as plt
```

Ваш код тут

1.8.2. Завдання 2: Енергії іонізації.

ЗАВДАННЯ 2: Обчисліть першу та другу енергії іонізації для атомів Li, Be, B, C, N, O, F, Ne. Порівняйте з експериментальними значеннями.

$$IE_1 = E(A^+) - E(A)$$
$$IE_2 = E(A^{2+}) - E(A^+)$$

Використайте базис cc-pvtz

Ваш код тут

1.8.3. Завдання 3: Спектроскопічні константи.

ЗАВДАННЯ 3: Для атома Карбону розрахуйте енергетичну різницю між основним триплетним станом (3P) та збудженими станами (1D та 1S).

Підказка: використовуйте різні спінові конфігурації

Ваш код тут

1.8.4. Завдання 4: Залежність від базису.

ЗАВДАННЯ 4: Дослідіть, як додавання дифузних функцій впливає на енергію та дипольну поляризованість атома Кисню (O).

Порівняйте: cc-pvdz vs aug-cc-pvdz, cc-pvtz vs aug-cc-pvtz

Ваш код тут

1.9. Резюме

У цьому розділі ми детально вивчили метод Хартрі-Фока для атомних систем:

- **Теоретичні основи** — рівняння HF, детермінант Слейтера, оператор Фока
- **Варіанти методу** — RHF для замкнених оболонок, UHF для відкритих, ROHF як компроміс
- **Одноелектронні системи** — H атом як тестовий випадок
- **Багатоелектронні атоми** — He та атоми другого періоду
- **Аналіз результатів** — орбітальні енергії, заселеності, спінова густина
- **Складні випадки** — перехідні метали, стратегії конвергенції

1.9.1. Ключові висновки.

1. Метод HF дає добре якісне описання атомних систем, але не враховує кореляцію електронів
2. Вибір між RHF/UHF/ROHF залежить від спінової структури системи
3. UHF страждає від забруднення спіном, але зазвичай дає нижчу енергію
4. Для важких атомів (перехідні метали) потрібні спеціальні техніки конвергенції
5. Якість результатів сильно залежить від вибору базисного набору

У наступному розділі ми розглянемо теорію функціоналу густини (DFT), яка часто дає кращі результати для багатоелектронних систем.

Література

Основна література

- 0. *Jensen F.* Introduction to Computational Chemistry. — 3rd ed. — Wiley, 2017. — 661 p. — ISBN 1118825993.
- 0. *Levine I. N.* Quantum Chemistry. — 7th ed. — Pearson, 2014. — 714 p. — ISBN 978-0321803450.

Додаткові посилання

- 0. [Basis Sets](https://gaussian.com/basissets/). — URL: <https://gaussian.com/basissets/>.
- 0. *Ho M., Hernández-Perez J. M.* [Evaluation of Gaussian Molecular Integrals. I Overlap Integrals](#) // The Mathematica Journal. — 2012. — Vol. 14.
- 0. *Ho M., Hernández-Perez J. M.* [Evaluation of Gaussian Molecular Integrals. II. Kinetic-Energy Integrals](#) // The Mathematica Journal. — 2013. — Vol. 15.
- 0. *Ho M., Hernández-Perez J. M.* [Evaluation of Gaussian Molecular Integrals. III. Nuclear-Electron Attraction Integrals](#) // The Mathematica Journal. — 2014. — Vol. 16.
- 0. [Simple Quantum Chemistry: Hartree-Fock in Python](#). — URL: <https://nznano.blogspot.com/2018/03/simple-quantum-chemistry-hartree-fock.html>.