КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ВИСОКИХ ТЕХНОЛОГІЙ

Завідувач кафедри молекулярної біотехнології та біоінформатики
доцент, к.б.н. Нипорко Олексій Юрійович
Протокол № __ засідання кафедри
від «__» травня 2023 р.

# РОЗРАХУНОК ДИПОЛЬНОГО МОМЕНТУ ТУБУЛІНУ ШЛЯХОМ КОМП'ЮТЕРНОГО МОДЕЛЮВАННЯ

Випускна кваліфікаційна робота магістра
студента спеціальності
091 Біологія
ОП «Біоінформатика і структурна біологія»
**Дідошика Дмитра Юрійовича**

Науковий керівник:
Ассистент кафедри молекулярної
біотехнології та біоінформатики
**Рибалка Ігор Євгенович**

Оцінка захисту роботи

_____

Київ - 2023 р.

# АНОТАЦІЯ

Дідошик Д. Ю. Розрахунок дипольного моменту тубуліну шляхом комп’ютерного моделювання. – Випускна кваліфікаційна робота магістра за спеціальністю 091 Біологія ОП «Біоінформатика та структурна біологія».

Дипольний момент відіграє важливу роль у збиранні мікротрубочок та створенні їх структури. Тубулін, головний компонент мікротрубочок, має дипольний момент, який сприяє електростатичним взаємодіям між ним та іншими молекулами. Ці взаємодії допомагають утворювати відносно стійкі зв'язки між тубуліном та іншими молекулами, які відповідають за збирання та структурування мікротрубочок.

В ході роботи було проаналізовано методи обчислення дипольного моменту тубуліну. Розроблено програмне забезпечення для обчислення дипольних моментів α- та β-мономерів,  та димеру тубуліну. Тестування програмного забезпечення дало задовільні результати.

Магістерська робота викладена на 45 сторінках, ілюстрована 1 таблицею та 19 рисунками. Список використаних джерел включає 10 робіт.

**Ключові слова:** тубуліни, мікротрубочки, цитоскелет, дипольний момент, електричне поле

ЗМІСТ

**ВСТУП** ..... **4**

**РОЗДІЛ 1. РОЛЬ ТУБУЛІНУ ТА ЙОГО ДИПОЛЬНОГО МОМЕНТУ** ..... **5**

1.1 Функції та будова мікротрубочок ..... 5

1.2 Роль дипольного моменту в збірці та дисоціації мікротрубочок .. 9

**РОЗДІЛ 2. МЕТОДИЧНІ ДАНІ** ..... **14**

2.1 Експериментальне визначення дипольного моменту ..... 14

2.2 Комп’ютерне обчислення дипольного моменту ..... 19

2.3 Мова Python та її бібліотеки ..... 23

2.4 Робота з файлами в Python ..... 27

2.5 Робота з файлами формату PDB ..... 29

2.6 Середовище розробки PyScripter ..... 30

**РОЗДІЛ 3. РОЗРОБКА ПРОГРАМИ ДЛЯ ОБЧИСЛЕННЯ ДИПОЛЬНОГО МОМЕНТУ** ..... **32**

3.1 Структура програми та опис її роботи ..... 32

3.2 Результати виконання розрахунків за допомогою розробленої програми ..... 38

**ВИСНОВКИ** ..... **41**

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** ..... **42**

**ДОДАТОК А** ..... **43**

# ВСТУП

Дипольний момент відіграє важливу роль у формуванні мікротрубочок та їх структурної організації. Головний компонент мікротрубочок, яким є тубулін, володіє дипольним моментом, який створює електростатичні взаємодії з іншими молекулами. Ці взаємодії сприяють утворенню стійких зв'язків між тубуліном та іншими молекулами, що забезпечують збирання та структурування мікротрубочок.

Крім того, дипольний момент тубуліну також взаємодіє з електричним полем, що може відтворюватися в мікротрубочках внаслідок протікання іонних потоків, наприклад, під час транспорту різних молекул та іонів. Ця взаємодія може також впливати на форму та структуру мікротрубочок.

**Мета** роботи точно та актуально визначити параметри моделі за якими можна більш точно отримати значення дипольних моментів α- та β-мономерів та димеру тубуліну.

Виходячи з мети були поставлені наступні **завдання**:

1. Проаналізувати роботи попередніх дослідників і їх методи визначення дипольного моменту тубуліну.
2. Створити і проаналізувати модель на основі якої можна визначати дипольний момент макромолекул.
3. Створити програмне забезпечення, що здатне вираховувати дипольний момент макромолекул на основі визначеної нами моделі.

# РОЗДІЛ 1. РОЛЬ ТУБУЛІНУ ТА ЙОГО ДИПОЛЬНОГО МОМЕНТУ

## 1.1 Функції та будова мікротрубочок

Мікротрубочки білкові структури циліндричної форми з порожниною всередині, діаметр складає приблизно 25 нм, в довжину вони можуть досягати кількох мікрон (рис. 1.1). Мікротрубочки є одним із основних типів структурних елементів цитоскелету.

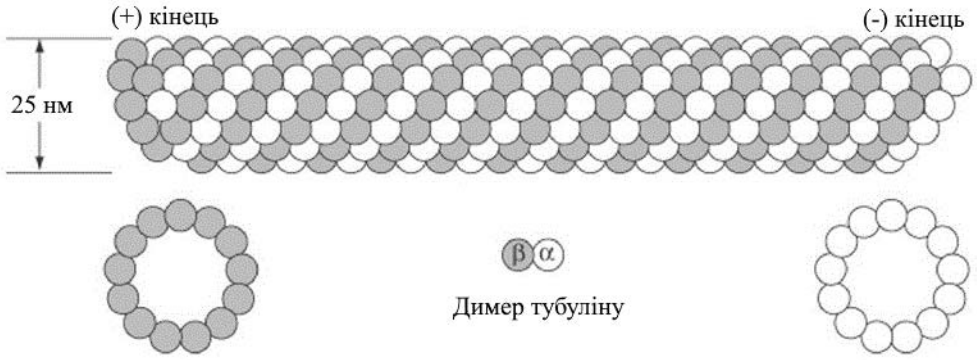


Рис. 1.1 Структура мікротрубочки.

Являючись найбільш міцними елементами цитоскелету мікротрубочки надають клітині механічну опору та підтримують її форму. Вони впливають на розташування та орієнтацію органел клітини, таких як ядро, мітохондрії та лізосоми.[1] Мікротрубочки грають важливу роль у процесі поділу клітини. Вони утворюють мітотичний вусик, який допомагає розподілити хромосоми під час мітозу та мейозу.

Однією з основних функцій мікротрубочок є транспортна функція, мікротрубочки являють собою шляхи для транспортування елементів всередині клітини. Транспорт у мікротрубочках здійснюють два основні класи моторних білків - це кінезини та динеїни.[1] Ці моторні білки

взаємодіють з мікротрубочками, здійснюючи кроковий рух по їх структурі, використовуючи енергію, яка вивільняється під час гідролізу АТФ.

Існує два типи транспорту, які відбуваються за допомогою мікротрубочок, від центру до периферії та від периферії до центру клітини. Антеградний транспорт, коли мікротрубочки утворюють дороги, по яких рухаються моторні білки, переносячи різні везикули, мембрани, білки та інші компоненти в клітині в напрямку від центру до периферії. За антеградний транспорт відповідають кінезини. Також мікротрубочки беруть участь у ретроградному транспорті структур, які потрібно повернути назад до центру клітини, таких як відпрацьовані органели або мембрани, що повинні бути перероблені. За такий тип транспорту відповідають динеїни.

Цей транспортний процес, який залежить від мікротрубочок, важливий для правильної організації та функціонування клітини. Він дозволяє рухати важливі компоненти в клітині до необхідних місць, забезпечує розподіл молекул та структур у клітині, а також впливає на розвиток, диференціацію та відповідь клітин на зовнішні подразники.

Мікротрубочки є динамічними структурами, які постійно полімеризуються та деполімеризуються в клітині.[2] Центром організації мікротрубочок є клітинний центр, звідки вони ростуть до периферії клітини. Але мікротрубочки можуть раптово припинити свій ріст, після чого вони зазвичай швидко скорочуються назад до центру клітини до повного розчинення, а потім знову виростають. Динамічна нестабільність мікротрубочок має важливу роль в фізіології.[3] Наприклад, під час поділу клітини мікротрубочки швидко ростуть і допомагають правильно орієнтувати хромосоми, формуючи веретено поділу.

Основною структурною одиницею мікротрубочок є тубулін. Тубуліни – це родина білків, представниками якої є зокрема α-тубулін та β-тубулін. А- та β-тубулін мають на 40% однакову первинну структуру і ніколи не виявляються поодинці. Вони завжди пов’язані між собою і утворюють димер, що для зручності називають тубуліном. В димері тубуліну α та β мономери

розташовані лінійно, так, що передня частина одного мономера зв’язана з задньою частиною іншого. Кожна молекула α та β тубуліну зв’язує молекулу ГТФ.

Будова димеру тубуліну показана на рисунку 1.2, видно що, ГТФ зв’язаний з α-тубуліном розташований близько до місця його контакту з β-тубуліном. Цей ГТФ не гідролізується і не обмінюється з нуклеотидами, що знаходяться в розчині. В цей час ГТФ зв’язаний з β-тубуліном є відкритим з іншої сторони димеру і при збірці мікротрубочок гідролізується до ГДФ.[4]

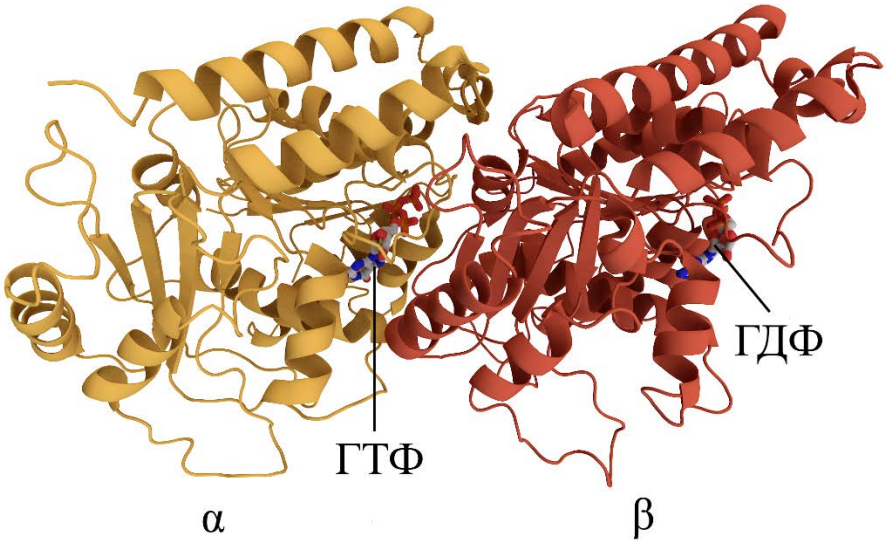


Рис. 1.2 Будова димеру тубуліну.

Мікротрубочки зазвичай складаються із лінійних ланцюгів субодиниць, які називають протофіламенти, їх можна побачити на рисунку 1.3. Протофіламенти з’єднуючись по боках утворюють мікротрубочку діаметром 25 нм, тобто в 5 разів більше ніж товщина її стінок. Хоча можливе утворення мікротрубочки із 11-15 протофіламентів, в основному їх кількість13. В більшості сусідніх протофіламентів α-тубуліни розташовуються ближче до інших α-тубулінів, а β-тубуліни, відповідно до β-тубулінів. Проте протофіламенти по відношенню одне до одного вирівняні не строго і можна побачити шов в якому α-тубуліни приєднуються до β-тубулінів (рис. 1.3).

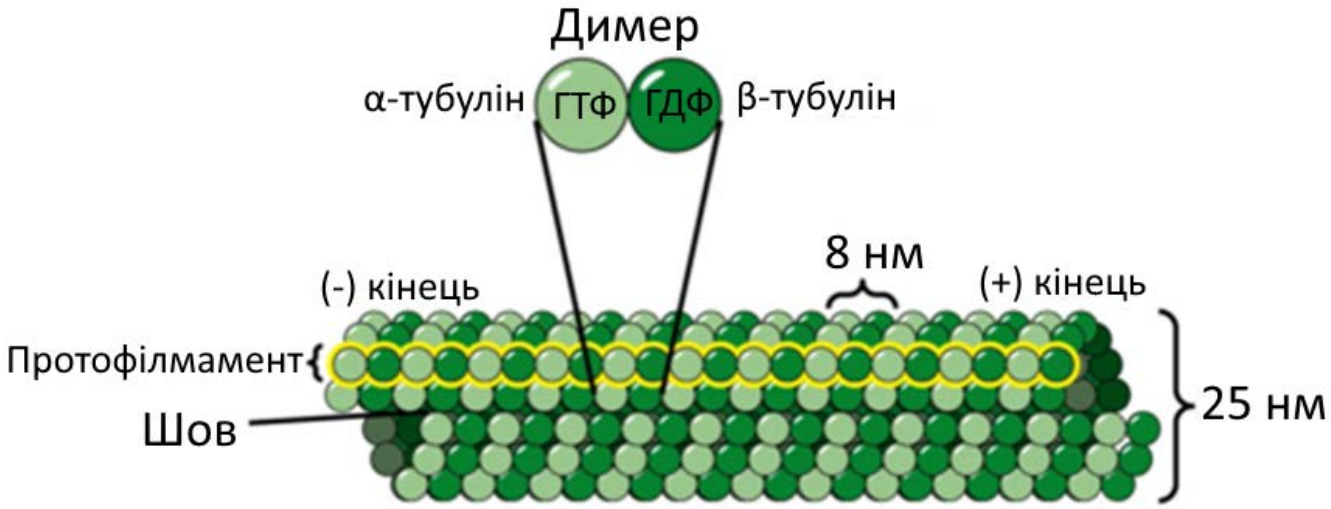


Рис. 1.3 Протофіламенти та шов показані на ділянці мікротрубочки.

Кожен димер тубуліну в мікротрубочці утворює міцні нековалентні зв'язки. Ці зв'язки утворюються як в повздовжньому так і в боковому напрямах.[2,3] Субодиниці в протофіламенті пов'язані повздовжніми зв'язками. Наявність бокових зав'язків між субодиницями робить дисоціацію димерів тубуліну в середині мікротрубочки маловірогідною. Димери тубуліну здатні спонтанно утворювати повздовжні та бокові зв'язки, що призводить до утворення мікротрубочок. Тому збірка мікротрубочок являє собою процес самозбірки, при якому вся необхідна інформація для утворення кінцевої структури міститься в субодиницях.

В кожному протофіламенті всі димери орієнтовані однаково, також всі протофіламенти в мікротрубочці направлені однаково. Таким чином β-тубуліни відкриті з однієї сторони, яку називають плюс-кінцем, а α-тубуліни відкриті відповідно зі сторони мінус-кінця. Таким чином кінці мікротрубочки структурно відрізняються і ведуть себе по різному. Така організація дозволяє незалежну збірку з обох кінців мікротрубочки. Також кожна мікротрубочка являє собою полярну структуру. Така полярність характерна для всієї мікротрубочки, а не тільки для її кінців. Саме полярність дозволяє

мікротрубочкам виконувати транспорту функцію, тобто служити двонаправленим шляхом для білкових молекулярних моторів.[5]

## 1.2 Роль дипольного моменту в збірці та дисоціації мікротрубочок

Було доведено, що електричні сили в мікроструктурах є універсальним інструментом для маніпулювання та вивчення біомолекул. Наприклад, електрокінетичні сили використовувалися для придушення броунівського руху та захоплення окремих білків для оптичного спостереження, для дослідження діаграми сила-швидкість актинових ниток, керованих міозином, і для вивчення кінетики зв’язування мікротрубочок на поверхнях, покритих кінезином.

В дослідженнях полімеризації та деполімеризації мікротрубочок, їх моделюванні як клітинних автоматів, може бути корисним дипольній момент тубуліну. Крім того ці знання можуть допомогти при вивченні самозбірки мікротрубочок. Мікротрубочки демонструють сегнетоелектричні властивості, тобто мають спонтанну орієнтацію диполів під впливом як зовнішнього так і внутрішньоклітинного електричних полів.

Кожний димер тубуліну має дипольний момент, і на стан поляризації димера впливають стани поляризації найближчих сусідів (внаслідок слабких ван-дер-ваальсових зв’язків між ними, які викликають взаємну орієнтацію дипольних моментів молекул тубуліну)[1,3]. Існують цілком конкретні правила, що визначають конформацію кожного димера через конформації його сусідів: якщо коефіцієнт диполь-дипольного зв’язку позитивний, то диполі мають феромагнітну орієнтацію, якщо коефіцієнт зв’язку негативний – антиферомагнітну орієнтацію, якщо коефіцієнт зв’язку випадкова величина – невпорядковану орієнтацію.

Дипольний момент є векторною величиною, яка вказує на асиметрію розподілу позитивного та негативного зарядів у електрично нейтральній

системі. Використання дипольного моменту дозволяє проводити структурні дослідження молекул, встановлювати їх конформацію, конформаційний та ізомерний склад, а також досліджувати залежність дипольного моменту від температури. Значення дипольного моменту молекул дають змогу зробити висновки про розподіл електронної густини у молекулах та залежність цього розподілу від властивостей окремих замісників.

Молекули можуть бути полярними або неполярними в залежності від того, як розподілена електронна густина в молекулі. У неполярних молекулах центри тяжіння позитивних та негативних зарядів знаходяться в одному місці. Полярні молекули складаються з диполів, тобто систем, де заряди (+q та –q) розташовані на певній відстані один від одного. Довжина диполя – це відстань між центрами мас позитивного та негативного зарядів в полярних молекулах. Коли атоми з різною електронегативністю утворюють хімічний зв’язок, електронні хмари перерозподіляються на користь атомів з більшою електронегативністю, зсуваються до атомів з більшою електронегативністю, що призводить до полярного зв’язку.

Значення дипольного моменту має великий вплив на властивості полярних молекул і матеріалів, які складаються з цих молекул. Полярні молекули можуть бути поляризовані в електричному полі, розташовуючись вздовж силових ліній поля, а також вирівнюватись в електричних полях, що створюються іонами у розчинах. Крім того, вони можуть взаємодіяти між собою, взаємодіючи з електричними полями, які вони створюють.

Полярна молекула може змінювати величину свого дипольного моменту під впливом зовнішніх електричних полів або електричних полів інших полярних молекул. Однак, при віддаленні від зовнішніх впливів дипольний момент повертається до своєї попередньої величини. Компоненти диполя які взаємодіють з електричним полем можна побачити на рисунку 1.4.

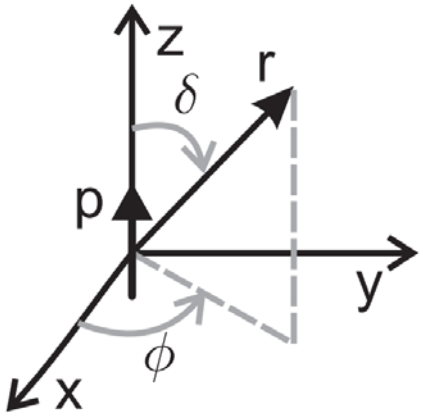


Рис. 1.4 Диполь в декартовій системі координат

Наведений дипольний момент можуть мати з'єднання, що не мають постійного дипольного моменту, але містять у своїй структурі фрагменти, що поляризуються. Додатковий дипольний момент може індукуватися і на молекулах сполук з постійним дипольним моментом; у цьому випадку результуючий дипольний момент залежить від кута між двома диполями, тобто наведений дипольний момент може збільшувати, так і зменшувати постійний дипольний момент.

Без зовнішнього електричного поля, на цю молекулу може діяти електричне поле сусідніх молекул, що володіють постійним дипольним моментом. Напруженість електричного поля, що діє на молекулу, відмінна від нуля, якщо вона має локально несиметричне полярне оточення.

Найважливіша сфера застосування даних про дипольні моменти молекул – структурні дослідження, встановлення конформації молекул, конформаційного та ізомерного складу речовини, її залежності від температури. Величини дипольного моменту молекул дозволяють будувати висновки про розподіл електронної щільності в молекулах і залежність цього розподілу від характеру окремих замінників.

У загальному випадку структурна інтерпретація дипольного моменту вимагає порівняння експериментальних величин зі значеннями, отриманими квантово-механічним розрахунком чи з допомогою адитивної векторної

схеми з допомогою дипольних моментів окремих зв’язків і атомних груп. Останні знаходять або за інтенсивністю коливальних смуг поглинання, або шляхом векторного розкладання дипольний момент деяких симетричних молекул.

Розрахунки з використанням векторної адитивної схеми можуть враховувати різні прояви стереохімічної нежорсткості, наприклад, утруднене або вільне внутрішнє обертання молекули. Високосиметричні молекулярні структури, що мають центр інверсії, двома взаємно перпендикулярними осями обертання або осями, перпендикулярними площині симетрії, не повинні мати дипольний момент. За наявністю чи відсутністю дипольного моменту молекули можна окремих випадках вибрати ту чи іншу структуру без будь-яких теоретичних розрахунків.

Вважається, що електромагнітне поле створюється мікротрубочками, що складаються з ідентичних гетеродимерів тубуліну з періодичною організацією та містять електричні диполі.[2,5] Структура мікротрубочок рис (1.5) зі спіральною та аксіальною періодичністю забезпечує взаємодію поля в часі, зміщеного на один або декілька періодів коливань, і генерацію когерентних сигналів.

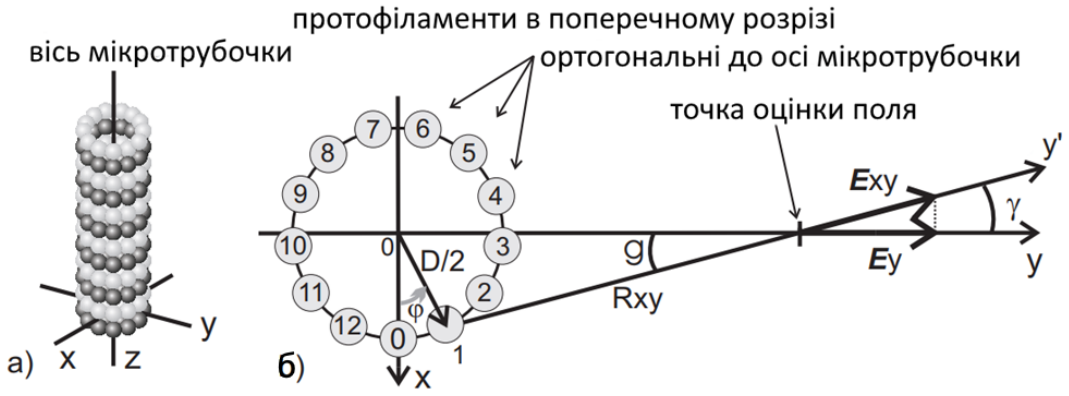


Рис. 1. 5 а) Мікротрубочки в декартовій системі координат, б) переріз на площині xy

Збудження внутрішньої порожнини має забезпечувати рівномірний розподіл енергії в мікротрубочці. Підведена енергія когерентно збуджує осцилятори з високою електричною якістю, внутрішньою порожниною мікротрубочки та електрони на молекулярних орбіталях і в зонах напівпровідності та провідності.

# РОЗДІЛ 2. МЕТОДИЧНІ ДАНІ

## 2.1 Експериментальне визначення дипольного моменту

Дипольний момент - це векторна фізична величина яка характеризує асиметрію розподілу позитивного та негативного зарядів в електрично нейтральній системі. В найпростішому варіанті являє собою добуток величин двох зарядів на відстань між ними. Він направлений вздовж осі диполя від негативного заряду до позитивного:

$$p = ql \qquad (2.1)$$

де q – величина заряду, l – вектор з початком координат в негативному заряді.

У випадку системи N зарядів дипольний момент являє собою, суму добутків величин зарядів на їх радіус-вектори:

$$\mathrm{p} = \sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{q_i r_i} \qquad (2.2)$$

де $\mathrm{q_i}$ – заряди частинки під номером i, $\mathrm{r_i}$ – радіус-вектор частинки під номером i.

У системі СІ для вимірювання дипольного заряду використовується одиниця кулон-метр, проте вона є надто великою для практичних застосувань, тому для цього використовуються інші одиниці. Здебільшого використовується одиниця дебай. 1 дебай = $3{,}33564 \cdot 10^{-30}$ Кл·м.

Як правило, дипольний момент вимірюється експериментально, оскільки це дає точніші та достовірніші результати. Експериментальний підхід дає змогу врахувати різноманітні фактори, що можуть вплинути на значення дипольного моменту, такі як температура, тиск, стан речовини (рідкий, газовий, твердий), та інші. Експериментальне визначення дипольного моменту може бути виконане за допомогою різних методів, залежно від конкретної речовини та умов експерименту.

Експериментальне визначення величин дипольних моментів молекул у збуджених станах не може бути засноване на звичайних методах вимірювання молекулярної поляризації внаслідок дуже малого часу життя цих станів і, отже, нікчемних концентрацій збуджених молекул за звичайних умов збудження[6]. Методи визначення величин засновані на вивченні оптичних властивостей молекул, поміщених у різні умови електричної взаємодії із середовищем (розчинники) або із зовнішнім електричним полем, та пов’язані з дослідженням одного з наступних явищ:

1) зсувів смуг у спектрах поглинання та люмінесценції в різних розчинниках та при різних температурах
2) люмінесценції та поглинання розчинів у сильних електричних полях (поляризація люмінесценції та електричний дихроїзм)
3) спектрів поглинання з’єднань у пароподібному стані в електричному полі (ефект Штарка).

Якщо зовнішнє електричне поле E відомо, перехід з достовірністю ідентифіковано, то визначити дипольний момент молекули можна з вимірювання зсуву частоти лінії, обумовленого ефектом Штарка. При невідомому напрямі дипольного моменту ефект Штарка визначається для кількох ліній. Це дає можливість обчислити значення проекцій дії моменту на осі. У цьому велика перевага даного методу визначення дипольного моменту, оскільки інші методи вимірів дають можливість визначити лише скалярну величину дипольного моменту молекули. Визначення дипольного моменту за ефектом Штарка дає можливість вимірювати невеликі значення дипольного моменту майже з такою самою точністю як і великі.

Ефект Штарка – розщеплення спектральних ліній атомів, молекул, кристалів в електричному полі. Він обумовлений тим, що в полі частка набуває додаткової енергії внаслідок поляризації і виникнення індукованого дипольного моменту. Взаємодія цього моменту з електричним полем призводить до зсуву та розщеплення рівнів енергії частки на підрівні. Залежність розщеплення від напруженості поля може бути лінійною чи

квадратичною. Відповідно, розщеплення ліній, що виникають при переходах між підрівнями, може бути симетричним (лінійний ефект) або несиметричним (квадратичний ефект).

В цьому методі молекула піддається впливу зовнішнього електричного поля, що діє за допомогою електричної сили на диполь молекули. Ця електрична сила зміщує центр мас диполя молекули, тому що вона діє на електричний заряд, який знаходиться віддаленим від центру мас молекули. Це зміщення можна виміряти за допомогою методів, таких як метод інтерференції світла. Для рідин і газів зазвичай використовують конденсатор, в якому знаходиться речовина, яка досліджується, а для твердих тіл – електричні поля, створені між двома електродами, розташованими з протилежних боків досліджуваного зразка.

Для точного вимірювання дипольного моменту за допомогою ефекту Штарка необхідно враховувати температуру та тиск, а також досить точно знати електричне поле, що створюється в зовнішньому конденсаторі або між електродами. Для цього зазвичай використовують калібрувальні зразки з відомими значеннями дипольного моменту, щоб забезпечити точність вимірювань. Отже, метод вимірювання дипольного моменту за допомогою ефекту Штарка є досить складним і вимагає спеціального обладнання та точної калібрування.

Мас-спектрометрія є іншим методом для вимірювання дипольного моменту молекул. В цьому методі молекули розряджаються в газовій фазі і зіткнення з електронами викликає їх іонізацію та фрагментацію. Молекулярні іони можуть бути розділені в магнітному полі залежно від їх маси та заряду. Зі знанням маси іону та його заряду можна визначити його дипольний момент.

Для вимірювання дипольного моменту за допомогою мас-спектрометрії, молекули спочатку розряджаються в газовій фазі. Далі, молекулярні іони виділяються за допомогою електронного удару або лазерної іонізації. Ці іони потім проходять через ряд магнітних і електричних

полів, які розділяють їх за масою та зарядом. За допомогою мас-спектрометра можна вимірювати масу іонів та їх заряд[7].

Мас-спектрометрія є чутливим та досить точним методом для вимірювання дипольного моменту молекул. Вона також дозволяє досліджувати масу та структуру молекул. Однак, цей метод може бути складним для деяких молекул, зокрема для важких молекул або молекул з низькими дипольними моментами.

Інший метод для вимірювання дипольного моменту молекул – це вимірювання діелектричної проникності речовини в електричному полі. Цей метод базується на тому, що діелектрична проникність речовини залежить від її дипольного моменту.

Діелектрична проникність речовини визначається за формулою:

$$\varepsilon = C/E \quad (2.3)$$

де ε – діелектрична проникність, C – ємність діелектричної речовини, E – напруга електричного поля.

За відсутності електричного поля, внутрішні дипольні моменти молекул розподілені випадковим чином, тому їх відносна сума дорівнює нулю. Однак, в електричному полі, молекули орієнтуються згідно з напрямом поля, утворюючи відносно однакові орієнтації диполів. Це призводить до збільшення сумарного дипольного моменту речовини та її діелектричної проникності.

Для вимірювання діелектричної проникності речовини в електричному полі, речовину поміщають в зону електричного поля, створеного між двома електродами. Напруга електричного поля змінюється, і вимірюється ємність діелектричної речовини при кожній зміні значення напруги. Залежність ємності від напруги дозволяє визначити діелектричну проникність речовини. Далі, діелектрична проникність пов’язана з дипольним моментом речовини за допомогою формули:

$$\varepsilon = 1 + (4\pi N\alpha/3V) \quad (2.4)$$

де N – кількість молекул у об’ємі V, α – полярність молекули.

За допомогою цієї формули можна визначити дипольний момент молекули. Однак, цей метод має свої обмеження, оскільки не всі речовини можна виміряти таким чином. Для того, щоб цей метод був ефективним, речовина має бути діелектриком, тобто мати дуже низьку електричну провідність. Якщо речовина є провідником, то напруга електричного поля в ній буде розсіюватися, що не дозволить виміряти ємність.

Крім того, цей метод може бути складним для використання в деяких випадках, так як вимірювання ємності діелектричної речовини залежить від багатьох чинників, таких як розмір зони електричного поля, форма електродів та інші параметри. В результаті, вимірювання діелектричної проникності може бути дуже чутливим до помилок та вимагати дуже високої точності.

Також використовується спектроскопія електричного дипольного моменту (EDM spectroscopy). У цьому методі використовуються електричні поля, що змінюються за частотою, для індукції різних перехідних станів молекул. Ці перехідні стани можуть бути виміряні за допомогою спектроскопії, яка дозволяє визначити спектральні лінії випромінювання, що виникають при переході молекул з одного стану в інший.

Дипольний момент молекули може бути визначений зі зміни енергії переходу між двома станами, які відрізняються за орієнтацією дипольного моменту молекули відносно зовнішнього електричного поля. Залежність енергії переходу від частоти зовнішнього поля може бути використана для розрахунку дипольного моменту молекули.

Однією з переваг цього методу є те, що він дозволяє вимірювати дипольний момент молекули з високою точністю та при цьому не вимагає зразків великої маси або об’єму. Крім того, EDM спектроскопія може бути застосована для вимірювання дипольних моментів різних молекулярних систем, включаючи молекули, що мають складну будову, такі як біомолекули.

Однак, метод EDM спектроскопії також має свої обмеження, так як вимірювання дипольного моменту за допомогою спектроскопії може бути досить складним, а інформація, яку можна отримати за допомогою цього методу, може бути обмежена. Крім того, EDM спектроскопія може бути чутлива до забруднення та ефектів взаємодії між молекулами, що можуть вплинути на точність вимірювання.

Загалом, EDM спектроскопія є одним із методів, що використовуються для вимірювання дипольного моменту молекул, і має свої переваги та обмеження. Використання цього методу може допомогти в розумінні властивостей молекулярних систем, а також в розробці нових матеріалів, технологій та медичних діагностичних засобів.

Загалом, експериментальне визначення дипольного моменту може бути складним і залежати від багатьох факторів, таких як властивості речовини, умови експерименту та точність вимірювального обладнання. Проте, експериментальні методи вимірювання дипольного моменту є надійними та точними і дають можливість отримати реальні значення дипольного моменту речовини. Крім того, експериментальні методи можуть бути корисними для валідації теоретичних розрахунків дипольного моменту, що можуть бути виконані за допомогою комп’ютерних програм. Отже, якщо є можливість, експериментальне визначення дипольного моменту є надійнішим та точнішим методом, однак, для складних систем, розрахунки за допомогою комп’ютерних програм можуть бути корисними.

## 2.2 Комп’ютерне обчислення дипольного моменту

За допомогою комп’ютерних програм можна розрахувати значення дипольного моменту теоретично, використовуючи різноманітні методи обчислення. Такі розрахунки можуть бути корисними для передбачення

властивостей нових речовин, дослідження їх взаємодії та властивостей у різних умовах, і підтримки експериментальних досліджень.

Один з підходів, що використовується для комп'ютерного визначення дипольного моменту, - це використання методів квантової хімії. У цьому підході використовуються різні квантовохімічні розрахункові методи для визначення енергії та структури молекул в тому числі методи теорії функціонала густини (DFT), методи конфігураційної взаємодії молекул та методи мультипольних розкладів[8].

Теорія функціонала густини (DFT) є чисельним методом для визначення електронної структури молекул та інших систем За методом DFT, електронна щільність визначається як функція координат частинок в системі, а не як окремі електрони, що рухаються в просторі. Це дозволяє зменшити кількість розрахунків та підвищити точність розрахунку.

Для розрахунку дипольного моменту за методом DFT, спочатку потрібно визначити геометрію молекули та електронну щільність в ній. За допомогою відповідних програмних пакетів, таких як Gaussian, NWChem, ORCA та ін., розраховуються електронні властивості молекули, включаючи дипольний момент.

Для отримання найбільш точних результатів, рекомендується використовувати функціонали щільності, які враховують додаткові ефекти, такі як дисперсійні взаємодії, зарядові перенесення та інші. Також, важливо враховувати взаємодію молекул з розчинником, який може впливати на дипольний момент молекули.

Ще одним методом квантової механіки є метод Гартрі-Фока. Він використовується для розрахунку електронної структури молекул та інших хімічних систем. Цей метод базується на понятті взаємодії електронів у молекулі та використовує модель слабкого взаємодії між електронами.

В даному методі, електронна структура молекули описується як суперпозиція одноелектронних хвильових функцій, які називаються «орбітальними». Орбітальні функції залежать від розташування ядер та

інших електронів у системі, і вони використовуються для визначення електронної щільності молекули.

Метод Гартрі-Фока вирішує рівняння Шредінгера для електронної структури молекули, щоб знайти енергетичний стан системи. Це рівняння включає гамильтоніан системи, який включає кінетичну та потенціальну енергії електронів та ядер. Дипольний момент молекули визначається шляхом обчислення середньої величини дипольного моменту для всіх можливих електронних конфігурацій молекули.

Хоча метод Гартрі-Фока є досить точним та широко використовується у квантовій хімії, він має свої обмеження, такі як нездатність враховувати дисперсійні взаємодії та квантові ефекти кореляції електронів. Тому, для додаткової точност

Методи конфігураційної взаємодії молекул є підходом до розрахунку дипольного моменту та інших властивостей молекул, який базується на моделюванні взаємодії між окремими конфігураціями молекул.

В основі методу лежить ідея, що взаємодія між молекулами може бути розглянута як взаємодія між їх окремими конфігураціями, а не як взаємодія між повними молекулами. Таким чином, замість розрахунку взаємодії між двома повними молекулами, можна розрахувати взаємодію між двома окремими конфігураціями цих молекул.

Метод використовує відомі методи розрахунку дипольного моменту, такі як квантово-хімічні розрахунки та експериментальні методи, для визначення дипольного моменту окремих конфігурацій молекул. Потім розраховується взаємодія між цими конфігураціями з використанням класичних методів розрахунку електростатичної взаємодії, таких як метод дискретних диполів або метод мультипольних розкладів.

Однією з головних переваг конфігураційної взаємодії молекул є те, що він дозволяє врахувати взаємодію між молекулами з високою точністю, в тому числі врахування ефектів довкілля. Крім того, метод може бути застосований для розрахунку властивостей великих систем молекул, таких як

біомолекули, що зазвичай не можуть бути обчислені з використанням стандартних методів розрахунку.

Іншим методом квантової хімії є QTAIM (Quantum Theory of Atoms in Molecules) або AIM (Atoms in Molecules). Він являє собою теоретичний метод в квантовій хімії, який дозволяє досліджувати внутрішню будову молекул, зокрема, взаємодії між атомами і їхню електронну структуру.

QTAIM виник як реакція на проблему визначення зарядів атомів в молекулах, яка стала важливим завданням для визначення хімічних властивостей і поведінки молекул. Цей метод базується на квантовій механіці та теорії функціоналу густини (DFT), і він дозволяє розглядати молекулу як систему атомів, пов’язаних електронами.

QTAIM використовує так звані критичні точки електронної густини, де градієнт густини дорівнює нулю, щоб аналізувати хімічні зв’язки між атомами та електронну структуру молекули в цілому. Він дозволяє описувати різні типи зв’язків (координаційні, ковалентні, водневі), визначати заряди атомів, їхню валентну електронну структуру та кількість зв’язків, а також виявляти водневі зв’язки та атоми водню, що належать до різних функціональних груп.

QTAIM можна застосовувати до різних типів молекул, включаючи органічні, неорганічні та металоорганічні сполуки. Він використовується в багатьох областях хімії, таких як каталіз, медицинська хімія, матеріалознавство та біохімія.

Метод молекулярної динаміки (МД) може бути використаний для обчислення дипольного моменту молекули. Дипольний момент є мірою розподілу заряду в молекулі, і він може бути визначений як вектор, що вказує в напрямку від негативно зарядженого до позитивно зарядженого кінця молекули[6,7].

У МД моделюється рух молекул у системі, враховуючи взаємодію між атомами та молекулами. Щоб визначити дипольний момент, можна обчислити середні значення зарядів та координат атомів та молекул у

системі. Також можна використовувати спеціальні програми та алгоритми для обчислення дипольного моменту, які враховують ефект польової поляризації молекули.

Узагалі, метод молекулярної динаміки є потужним інструментом для моделювання та вивчення різних фізичних та хімічних систем, що дозволяє отримувати значення різних параметрів, в тому числі і дипольного моменту.

Комп'ютерні методи можуть бути корисними для визначення дипольного моменту, особливо для складних систем, для яких експериментальне визначення може бути складним або неможливим. Проте, ці розрахунки можуть бути дуже часо- та ресурсоємними і вимагати високої обчислювальної потужності та точності. Важливо зазначити, що комп'ютерне моделювання має свої обмеження, особливо у випадку складних молекул, де може бути складно визначити точну структуру та поведінку молекул у системі.

## 2.3 Мова Python та її бібліотеки

Python – це інтерпретована високорівнева мова програмування. Python має дуже чистий та простий синтаксис, що дозволяє легко читати та розуміти код. Також мова має велику стандартну бібліотеку та велику кількість сторонніх модулів, що дозволяє розширювати функціональність Python та допомагає знижувати час розробки.

Python є крос-платформовою мовою програмування, що дозволяє запускати код на різних операційних системах, таких як Windows, Linux та MacOS. Python має багато інструментів для розробки та налагодження коду, таких як PyScripter, PyCharm, VS Code, Jupyter Notebook та інші.

Одна з головних переваг Python - це можливість використання його як мови скриптів, що дозволяє швидко створювати та виконувати невеликі програми без необхідності компіляції. Крім того, Python має вбудовану

підтримку для обробки текстових файлів, роботи з базами даних, мережевого програмування та багатьох інших завдань.

Python має можливість імпортувати зовнішні модулі (бібліотеки) в поточну програму. Це дозволяє використовувати додаткові функції, які доступні деінде. Модулі можуть бути частиною стандартної бібліотеки, яка автоматично додається до інсталяції Python, можуть бути додатковими бібліотеками, які потрібно окремо встановлювати, або можуть бути іншими програмами Python, написаними користувачем. Незалежно від джерела модуля, вони можуть бути імпортовані в програму за допомогою команди імпорту.

Якщо ми імпортуємо модуль, який не є частиною стандартної бібліотеки, ми повинні переконатися, що він інстальований у нашій комп'ютерній системі (і, як правило, є інструкції на сайтах для завантаження). Якщо його не встановлено, ми отримаємо ImportError.

Наприклад, NumPy - це бібліотека мови Python, яка дозволяє працювати з багатовимірними масивами і матрицями. Вона надає широкі можливості для наукових обчислень, статистичного аналізу, обробки даних і машинного навчання. Бібліотека NumPy забезпечує швидку і ефективну роботу з масивами завдяки використанню векторизованої обробки даних.

NumPy забезпечує підтримку базових числових операцій з наголосом на визначенні обчислень, які працюють над цілим масивом чисел одночасно. Його операції включають функції для випадкових чисел, лінійної алгебри та перетворень Фур'є. Він реалізований на C нижче,1 і, отже, швидко запускається, але, природно, до нього можна отримати доступ на Python [9].

NumPy має власну версію багатовимірних масивів, називається вона N-вимірним масивом, і дозволяє ефективно їх опрацьовувати. Базовий багатовимірний список у Python можна перетворити на масив NumPy і навпаки. Наприклад, у NumPy можна створити матрицю розміром 2x3, використовуючи наступний код: x = numpy.array([[1,2,3],[4,5,6]]). До

елементів у масиві NumPy можна отримати доступ так само, як і до звичайної Python матриці.

NumPy визначатиме тип даних матриці з типів даних її елементів. Наприклад, наведений вище масив x матиме тип int, оскільки його вміст є цілими числами. Якщо натомість ви бажаєте, щоб x мав значення з плаваючою комою, ви можете або зробити одне з чисел явно плаваючим, або ви можете вказати тип даних масиву під час побудови.

Для виконання інших математичних операцій над масивами NumPy має еквівалент більшості функцій, які є в математичному модулі, і вони ефективно виконують операції для кожного елемента масиву. Як і масиви NumPy, функції прийматимуть звичайні списки або кортежі Python як вхідні дані, але повертатиметься масив.

Окрім доступу до масивів із явними номерами індексів, NumPy підтримує нотацію фрагментів, яка дуже схожа на списки та кортежі в стандартному Python. Однак, оскільки масив може мати ранг, вищий за одиницю, як матриця, яка містить і рядки, і стовпці, тоді для кожного виміру масиву можна вказати вираз зрізу, а для розділення виразів для різних вимірів використовується кома.

Масиви NumPy мають низку вбудованих функцій (методів), доступ до яких можна отримати за допомогою крапкової нотації. У відповідних випадках ми часто можемо вказати, на якій осі (наприклад, рядки чи стовпці для матриці) працювати.

Бібліотека NumPy використовується в багатьох областях, таких як наукові дослідження, фізика, математика, інженерія, машинне навчання та обробка даних. Це робить її однією з найбільш важливих бібліотек для наукових обчислень у мові Python.

Бібліотека math в мові Python - це вбудована бібліотека, яка надає функції математичних операцій. Ця бібліотека містить ряд готових функцій для виконання математичних обчислень, таких як обчислення тригонометричних функцій, логарифмів, експонент та інші.

До функцій, які надає бібліотека math, входять: sin(), cos(), tan(), asin(), acos(), atan(), log(), exp(), sqrt(), ceil(), floor(), fabs() та інші.

Ці функції можуть бути корисними в різних областях програмування, включаючи наукові дослідження, фізику, математику, інженерію, статистику та інші. Бібліотека math дозволяє розширити можливості програмування для виконання різних математичних операцій і спрощує роботу з числовими даними в програмах Python.

Для демонстрації результатів математичних операцій часто буде зручно подати числові значення у вигляді графіка чи діаграми. Щоб проілюструвати дані в такому вигляді, ми можемо використовувати популярний модуль Matplotlib.

Matplotlib дозволяє створювати різноманітні типи графіків, такі як лінійні графіки, діаграми розсіювання, кругові діаграми, гістограми, теплові карти та багато інших. Бібліотека також підтримує анімацію, 3D-графіку, роботу з растровими та векторними зображеннями, та надає можливості для налаштування розміру, кольору та інших параметрів графіків.

Модуль pyplot є частиною бібліотеки Matplotlib і надає інтерфейс для створення різноманітних графіків та діаграм. Він містить різноманітні функції для налаштування властивостей графіків, таких як заголовки, мітки осей, легенди та кольорові схеми.

Щоб фактично відобразити лінійний графік на екрані, ми викликаємо функцію .show(), яка має викликати спливаюче вікно. Щоб створити графік із кількома рядками даних, ми можемо неодноразово використовувати plot() із різними списками даних, які всі будуть додані до того самого графіка, перш ніж остаточно викликати show().

За стандартом функція plot() використовуватиме значення за замовчуванням для визначення вигляду графіків. Однак існує багато додаткових параметрів, які можна вказати для керування малюванням ліній, осей, легенд тощо.

Як альтернатива простому показу графіка на екрані, ми також можемо записати зображення у файл. Відповідно тоді, ми використовуємо savefig(), це треба робити перед викликом методу show(), інакше останній очистить поточний графік даних.

Як і прості лінійні графіки, Matplotlib може легко створювати багато інших типів відображення даних. Наприклад, ми можемо створити точкову діаграму за допомогою scatter() (а не plot()), де ми також встановлюємо стиль для символу маркера та його розмір.

## 2.4 Робота з файлами в Python

Python має можливість зчитувати та записувати дані з зовнішніх файлів. Для цього використовуються вбудовані функції для роботи з файлами.

Під час читання файлу в програмі Python вам, очевидно, потрібно вказати розташування та ім’я файлу, що разом називається шляхом до файлу. Потім шлях до файлу використовується вбудованою функцією open() або подібною для отримання доступу до вмісту файлу. Шлях може бути абсолютним, що означає, що він починається у верхній або кореневій частині ієрархії файлів операційної системи, або шлях може бути відносним до поточного робочого каталогу (папки). Спочатку поточний робочий каталог є каталогом, з якого запускається інтерпретатор Python, хоча його можна змінити всередині коду, якщо це необхідно. У Python шлях до файлу є просто рядком.

Функція .open() повертає те, що називається дескриптором файлу, об’єктом Python, який використовується для доступу та представлення відкритого файлу. Тут дескриптор файлу зберігається у змінній під назвою fileObj. Може виникнути спокуса використовувати більш природну назву змінної «файл», але в Python слово «файл» уже використовується для опису типу (класу) таких об’єктів. Тому найкраще уникати перевизначення імені

внутрішньої змінної. Альтернативою описовому, але багатослівному «fileObj» є коротше «fh», що означає «дескриптор файлу».

Існує ймовірність, що програма може не відкрити файл із вказаним шляхом. Наприклад, зазначеного файлу може просто не існувати (наприклад, назва неправильна), або у вас немає дозволу на його читання. За таких обставин, коли ви спробуєте відкрити його, функція видасть помилку, стандартний виняток Python (IOError). Інакше, якщо все піде добре, коли ви закінчите читати файл, ви можете явно сказати, що закінчили, використовуючи функцію close(), метод, який відомий класу файлових об’єктів.

Зазвичай вам не потрібно явно закривати файл, тому що, коли ім’я змінної fileObj виходить за межі видимості та більше не має сенсу, наприклад, у кінці циклу або функції, Python автоматично закриє його. Однак, як правило, корисною звичкою є явно закривати дескриптор файлу, коли ви знаєте, що ви закінчили, на випадок, якщо це важливо. Крім того, це дозволяє тому, хто читає ваш код, знати, що ви закінчили використовувати файл.

Файловий об’єкт має певні функціональні можливості, пов’язані з ним, які дозволяють читати базові дані у файлі. Найпоширенішими функціями є: read(), readline() і readline(). Функція read() використовується, якщо ви хочете прочитати весь файл за один раз, в один довгий рядок.

Функція read() має необов’язковий аргумент, який визначає необхідну кількість байтів (адресних одиниць інформації) для завантаження, але зчитування всього файлу за один раз більш поширене для цієї функції. Звичайно, якщо файл великий, це може бути не гарною ідеєю через обмеження пам’яті. Відповідно, функція readline() зчитує лише один рядок із файлу та часто розміщується в циклі для обробки кількох рядків без Необхідності завантажувати все одночасно.

Що стосується readline() і readlines(), кожен рядок файлу визначається як текстовий рядок, що закінчується символом нового рядка, або рядок, який

закінчується в кінці файлу без нового рядка. Іншими словами, символ нового рядка розділяє рядки. Зверніть увагу, що символ нового рядка в кінці рядка не видаляється під час використання цієї функції, тобто він включається як останній символ повернутого рядка.

## 2.5 Робота з файлами формату PDB

Файли PDB (Protein Data Bank) були винайдені в 1970-х роках для опису тривимірних координат біологічних макромолекул. Як випливає з назви, спочатку вона була розроблена для білків, але зараз ця система широко використовується для представлення ДНК, РНК, вуглеводів, ліпідів, малих молекул та будь-яких інших біологічно важливих молекул. Файли PDB можуть містити опис кількох молекул і багатьох структур, а також можуть містити багато іншої описової інформації. Файл PDB орієнтований як на ключ/значення, так і на рядок, причому ключ на початку кожного рядка надає контекст для даних у решті рядка. Координати, які нас цікавлять, знаходяться в записах, де рядок починається з шести символів «ATOM» (з двома пробілами в кінці), які можна вважати ключем. Координата x наведена в стовпцях з 30 по 37, y в стовпцях з 38 по 45 і z в стовпцях з 46 по 53, припускаючи, що перший стовпець є стовпцем 0 [9].

Спочатку ми відкриваємо файловий об’єкт, читаємо всі рядки, а потім негайно закриваємо його. Далі змінні, що представляють кількість атомів і загальні значення для координат x, y і z, ініціалізуються нулем перед повторенням кожного рядка. Якщо рядок починається з потрібної клавіші «ATOM», кількість атомів збільшується, координати вилучаються, а підсумкові координати збільшуються. Дані координат спочатку є просто текстовими символами з файлу, і їх потрібно перетворити на числа Python (які можна додати чисельно). Python float() виконує перетворення тестового

рядка в число з плаваючою комою. Так, наприклад, рядок «12,572» буде перетворено на число 12,572.

Якщо файл не існує або у вас немає дозволу на його читання, функція викличе стандартний виняток Python (IOError), коли спробує його відкрити [9]. Якщо файл існує, але не є файлом PDB, швидше за все, у ньому не буде рядків, що починаються з тексту «ATOM», і тому функція просто поверне нульові значення. У цьому випадку також можливо, що існує рядок, який починається з «ATOM» (за збігом обставин), але він не має трьох чисел з плаваючою комою в стовпцях з 30 по 53, і в цьому випадку буде створено стандартний виняток Python (ValueError), коли викликається функція float().

## 2.6 Середовище розробки PyScripter

PyScripter - це інтегроване середовище розробки (IDE) для мови програмування Python. Воно доступне для безкоштовного завантаження та використання на платформі Windows.

Основні функції PyScripter включають:

- Підсвічування синтаксису та автодоповнення коду для полегшення написання коду.
- Вікна налагоджування для детального аналізу та налагодження коду.
- Підтримка керування версіями для керування змінами в коді та спільної роботи над проектами.
- Підтримка інтерактивної консолі для швидкої перевірки коду та експериментів.
- Підтримка роботи з проектами та збіркою проектів в один виконуваний файл.

PyScripter також підтримує використання бібліотек Python, таких як NumPy, SciPy, Matplotlib та інших. Крім того, PyScripter має інтерактивну

консоль, що дозволяє виконувати команди Python безпосередньо з програми, що дуже зручно для швидкого тестування фрагментів коду.

Більшість функцій PyScripter можна налаштувати згідно з потребами користувача, що робить його зручним для використання в різних проектах та з різними налаштуваннями.

PyScripter розроблений на мові Delphi і працює на платформах Windows, хоча може бути встановлений і на інші операційні системи, такі як Linux і Mac OS, за допомогою віртуальних машин або інших програм для запуску Windows додатків.

# РОЗДІЛ 3. РОЗРОБКА ПРОГРАМИ ДЛЯ ОБЧИСЛЕННЯ ДИПОЛЬНОГО МОМЕНТУ

## 3.1 Структура програми та опис її роботи

Для виконання дипломної роботи була розроблена програма, яка буде розраховувати значення дипольного моменту молекули тубуліну. Послідовність дій при виконанні програми можна побачити на рисунку 3.1, а її повний код приведено в Додатку А.

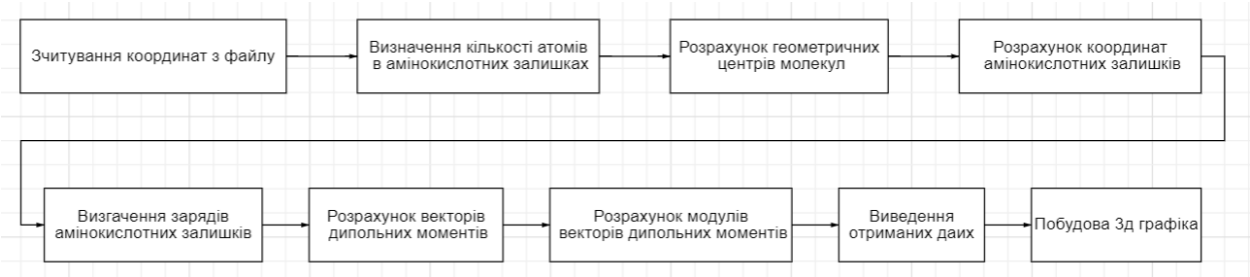


Рис. 3.1 Структурна схема роботи програми

Програма була написана з використанням мови Python та середовища розробки PySctipter,я кі були описані у попередньому розділі. На рисунку 3.2 показано код, що під’єднує до програми необхідні для роботи бібліотеки: math, numpy, matplotlib, які були описані в попередньому розділі.

```
import math
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

Рис. 3.2 Використані бібліотеки

Програма працює з файлом 1tub.pdb [10], що містить дані про структуру молекули тубуліну і на основі цих даних розраховує дипольний момент молекули. На рисунку 3.3 показана функція Atom_Koor яка зчитує координати атомів з файлу формату .pdb та записує їх в масиви NumPy відповідно для координат x, y та z.

```
def Atom_Koor(mono):
    f=open('1tub.pdb','r')
    x=np.array([])
    y=np.array([])
    z=np.array([])
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                r=(line[30:54].split())
                x=np.append(x,float(r[0]))
                y=np.append(y,float(r[1]))
                z=np.append(z,float(r[2]))
    f.close()
    return [x,y,z]
```

Рис. 3.3 Функція Atom_Koor

Далі з файлу визначається кількість атомів в амінокислотних залишках. В файлах формату .pdb всі амінокислотні залишки мають свій порядковий номер за яким програма рахує кількість атомів в тому чи іншому залишку. Так як деякі номери амінокислотних залишків можуть бути пропущені програма перевіряє наявність амінокислотного залишку під тим чи іншим номером, в разі якщо такого номеру в файлі немає програма пропускає його не вносячи ніяких даних в масив. Це зроблено для збільшення точності розрахунків. Отримані дані заносяться в масив. Все це робиться за допомогою функції Atom_N показаної на рисунку 3.4.

```
def Atom_N(mono):
    f=open('1tub.pdb','r')
    Mono=np.array([])
    aaMono=0
    n=0
    tmp=[0,0,0]
    expected_aa = 1
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                if int(line[23:27]) == expected_aa:
                    if aaMono!=0:
                        Mono=np.append(Mono,n)
                    n=0
                    aaMono+=1
                    expected_aa += 1
                    n+=1
                elif int(line[23:27]) < expected_aa:
                    n+=1
                    continue
                else:
                    expected_aa += 1
                    continue
    if n != 0:
        Mono=np.append(Mono,n)
    f.close()
    return(Mono)
```

Рис. 3.4 Функція Atom_N

Наступним кроком йде визначення геометричних центрів мономерів та димеру, що потрібні для подальших розрахунків, функція Center що виконує дану дію показана на рисунку 3.5.

```
def Center(mol):
    mol=np.array(mol)
    mol_center=np.mean(mol, axis=1)
    return mol_center
```

Рис. 3.5 Функція Center

Далі програма проводить розрахунок координат амінокислотних залишків. Функція Residue, показана на рисунку 3.6, розраховує координати амінокислотних залишків за допомогою координат атомів які входять до залишку, їх кількості та центру молекули до якої входить залишок. Розрахунок проводиться окремо для мономерів та димеру, дані заносяться в відповідні масиви.

```
def Residue(mol,n):
    mol=np.array(mol)
    center=Center(mol)
    sub_mean_array = np.empty((mol.shape[0], len(n)))
    for idx, val in enumerate(n):
        start_idx = sum(n[:idx])
        end_idx = start_idx + val
        sub_arr = mol[:,start_idx:end_idx]
        sub_mean = np.mean(sub_arr, axis=1)
        sub_mean[0]=sub_mean[0]-center[0]
        sub_mean[1]=sub_mean[1]-center[1]
        sub_mean[2]=sub_mean[2]-center[2]
        sub_mean_array[:, idx] = sub_mean
    return(sub_mean_array)
```

Рис. 3.6 Функція Residue

Наступним кроком йде визначення зарядів амінокислотних залишків. Визначення зарядів йде за номером амінокислотного залишку та назвою його амінокислоти, які вказані в файлі. Припускаючи, що значення pH дорівнює 7, часткові атомні заряди призначаються всім атомам, так що всі залишки Arg і Lys мають загальний сумарний заряд +1, усі залишки Glu і Asp мають сумарний заряд –1, а всі інші залишки мають нульовий сумарний заряд. Як і у випадку функції Atom_N, функція Res_Charges (рис. 3.7), яка присвоює залишкам заряди, робить перевірку на наявність амінокислотного залишку в залежності від його номеру. Отримані вході виконання функції дані заносяться в відповідні масиви.

```
def Res_Charges(mono):
    f=open('1tub.pdb','r')
    Charges=np.array([])
    aaMono=0
    expected_aa = 1
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                if int(line[23:27]) == expected_aa:
                    if line.find('LYS') > 0 or line.find('ARG') > 0:
                        Charges=np.append(Charges,1)
                    elif line.find('GLU') > 0 or line.find('ASP') > 0:
                        Charges=np.append(Charges,-1)
                    else:
                        Charges=np.append(Charges,0)
                    aaMono+=1
                    expected_aa += 1
                elif int(line[23:27]) < expected_aa:
                    continue
                else:
                    expected_aa += 1
                    continue
    f.close()
    return(Charges)
```

Рис. 3.7 Функція Res_Charges

Далі отримані масиви з координатами амінокислотних залишків та їх зарядами передаються в функцію Dipole, показану на рисунку 3.8. В даній функції за допомогою рівняння (2.2) виконується розрахунок векторів дипольних моментів для мономерів та димеру. Дані заносяться в відповідні змінні.

```
def Dipole(mol,charges):
    p=0
    i=0
    j=0
    k=0
    vector=np.array([0.0,0.0,0.0])
    for elements in mol[0]:
        vector[0]+=elements*charges[i]
        i+=1
    for elements in mol[1]:
        vector[1]+=elements*charges[j]
        j+=1
    for elements in mol[2]:
        vector[2]+=elements*charges[k]
        k+=1
    return vector
```

Рис. 3.8 Функція Dipole

Наступним кроком функція Dip_moment, показана на рисунку3.9, розраховує модулі отриманих з функції Dipole векторів. Далі функція Dip_moment конвертує отримані дані в одиниці вимірювання дебай за допомогою множення даних в ангстремних одиницях заряду на коефіцієнт 4.803.

```
def Dip_moment(mol):
    p=math.sqrt(mol[0]*mol[0]+mol[1]*mol[1]+mol[2]*mol[2])
    return(p*4.803)
```

Рис. 3.9 Функція Dip_moment

На рисунку 3.10 показана черговість виклику описаних вище функцій та змінні в які ці функції записують дані отримані при виконанні.

```
A_a=Atom_Koor('A')
B_a=Atom_Koor('B')
Dim_a=np.concatenate((A_a, B_a), axis=1)

A_n=Atom_N('A').astype(int)
B_n=Atom_N('B').astype(int)
Dim_n=np.concatenate((A_n, B_n))

A_r=Residue(A_a,A_n)
B_r=Residue(B_a,B_n)
Dim_r=Residue(Dim_a,Dim_n)

A_c=Res_Charges('A')
B_c=Res_Charges('B')
Dim_c=np.concatenate((A_c, B_c))

A_dip=Dipole(A_r,A_c)
B_dip=Dipole(B_r,B_c)
Dim_dip=Dipole(Dim_r,Dim_c)

A_p=Dip_moment(A_dip)
B_p=Dip_moment(B_dip)
Dim_p=Dip_moment(Dim_dip)
dim_dip2=A_dip+B_dip
dim_p2=Dip_moment(dim_dip2)
```

Рис. 3.10 Виклик функцій

На рисунку 3.11 показано код що, виводить дані отримані в ході виконання програми. Для α і β мономерів та димеру виводиться кількість

атомів та амінокислотних залишків які в них входять, сумарних заряд молекул та значення їх дипольних моментів.

```
print('A:   ','Atoms',len(A_a[0]),' Residue',len(A_r[0]),' Charge',np.sum(A_c),' Dipole:',A_p)
print('B:   ','Atoms',len(B_a[0]),' Residue',len(B_r[0]),' Charge',np.sum(B_c),' Dipole:',B_p)
print('Dim: ','Atoms',len(Dim_a[0]),' Residue',len(Dim_r[0]),' Charge',np.sum(Dim_c),' Dipole:',Dim_p)
print('векторна сума A та B:',dim_p2)
```

Рис. 3.11 Виведення даних

Останнім етапом роботи програми є виведення 3д графіка, який включає в себе візуалізацію атомів та векторів дипольних моментів мономерів та димеру. Як можна побачити на рисунку 3.12 атоми α мономеру позначені червоним кольором, атоми β мономеру позначені синім кольором. Вектори дипольних моментів для α і β мономерів та димеру позначені синім, червоним та чорним кольорами відповідно.

```
fig = plt.figure(figsize=(10,7.5))
ax = fig.add_subplot(111, projection='3d')
center_A=Center(A_a)
center_B=Center(B_a)
center_Dim=Center(Dim_a)
ax.scatter(A_a[0], A_a[1], A_a[2], c='r', marker='o')
ax.scatter(B_a[0], B_a[1], B_a[2], c='b', marker='o')
ax.quiver(center_A[0],center_A[1],center_A[2],A_dip[0],A_dip[1],A_dip[2],arrow_length_ratio=0.05, color='blue')
ax.quiver(center_B[0],center_B[1],center_B[2],B_dip[0],B_dip[1],B_dip[2],arrow_length_ratio=0.05, color='red')
ax.quiver(center_Dim[0],center_Dim[1],center_Dim[2],Dim_dip[0],Dim_dip[1],Dim_dip[2],arrow_length_ratio=0.05,color='black')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```

Рис. 3.12 Побудова графіка

## 3.2 Результати виконання розрахунків за допомогою розробленої програми

В нижче приведеній таблиці ми показуємо результати розрахунків модулів дипольних моментів для α- β-мономерів та димеру молекули тубуліну. В першому рядку наведено результати наших обчислень, в другому та третьому рядках результати обчислень інших авторів[6,8].

| | α-мономер, дебай | β-мономер, дебай | Димер, дебай |
|---|---|---|---|
| | 845 | 1384 | 2110 |
| Felder | 775 | 1312 | 2037 |
| Mershin | 552 | 1193 | 1740 |

Таблиця 3.1 Порівняння дипольних моментів

Як можна побачити наші результати корелюють з результатами обчислень інших авторів. В першому випадку[8] ми іншим способом розраховували центри мас молекул, наш метод видається нам кроком в бік більшої точності. В другому випадку[6] автори вдавались до допомоги програмного забезпечення TINKER з параметром CHARMM.

На рисунку 3.13 можна побачити візуалізацію молекули тубуліну, атоми α мономеру позначені червоним кольором, атоми β мономеру позначені синім кольором.

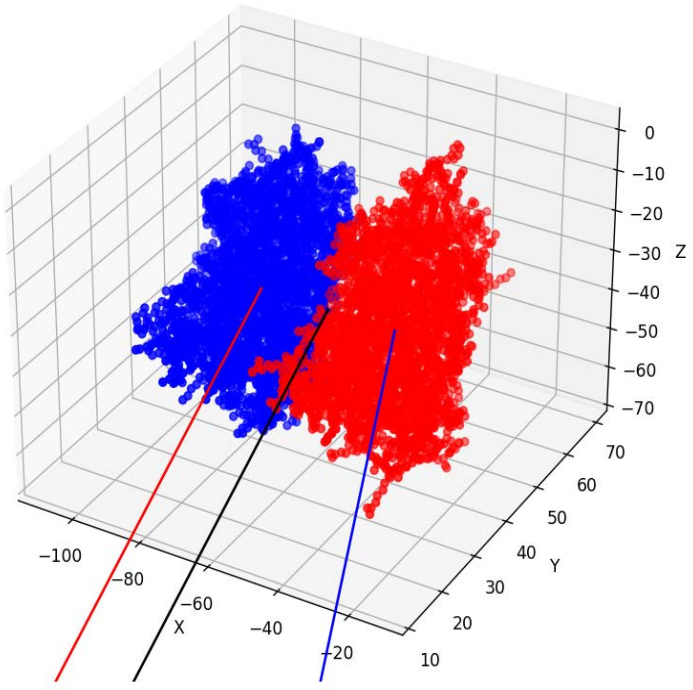


Рис. 3.13 3д графік молекули тубуліна

На рисунку 3.14 можна краще оцінити напрями і довжини векторів дипольних моментів для α- β-мономерів та димеру. Вектори дипольного

моменту для α мономера позначений синім кольором, для β мономера червоним кольором та чорним кольором для дипольного моменту димера.

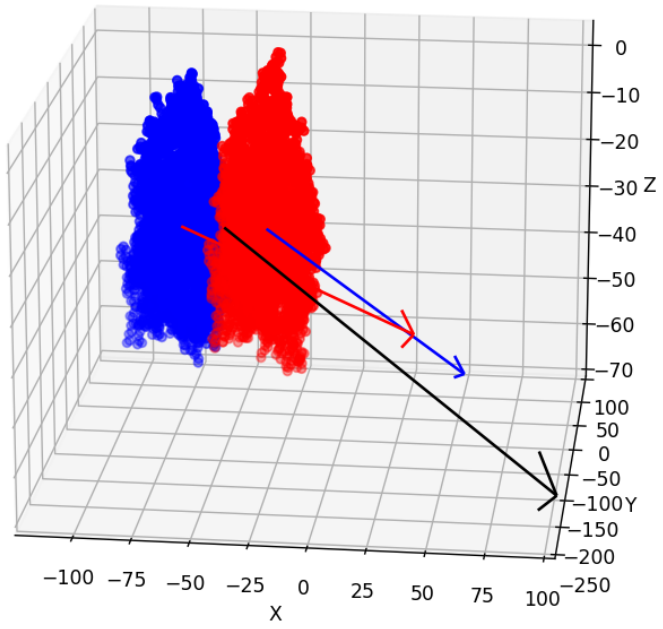


Рис. 3.14 Вектори дипольних моментів α і β мономерів та димеру

# ВИСНОВКИ

1. На основі аналізу літературних даних виявлено, що оцінки значень дипольних моментів α- і β- мономерів та димеру тубуліну сильно залежать від методик і значно відрізняються в роботах різних авторів.

2. Вектор дипольного моменту димеру тубуліну не є геометричною сумою векторів дипольних моментів α- і β- мономерів.

3. Створена модель дозволяє вираховувати дипольні моменти тубуліну з достатньою точністю.

4. На основі створеної розроблено програмне забезпечення для вирахування дипольних моментів моментів α- і β- мономерів та димеру тубуліну.

5. Тестування нашого програмного забезпечення дало задовільний результат.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Benjamin Lewin, Lynne Cassimeris. Cells. 2011
2. Annapurna Vemu, Joseph Atherton, Jeffrey O Spector, Carolyn A Moores, Antonina Roll-Mecak. Tubulin isoform composition tunes microtubule dynamics. 2017
3. Holly V Goodson, Erin M Jonasson. Microtubules and Microtubule-Associated Proteins. 2018
4. Nikita B Gudimchuk, J Richard McIntosh. Regulation of microtubule dynamics, mechanics and function through the growing tip. 2021
5. Pavla Binarová, Jack Tuszynski. Tubulin: Structure, Functions and Roles in Disease. 2019
6. A. Mershin, A.A. Kolomenski, H.A. Schuessler, D.V. Nanopoulos. Tubulin dipole moment, dielectric constant and quantum behavior: computer simulations, experimental results and suggestions. 2004
7. J. A. Tuszynski, J. A. Brown, E. Crawford, E. J. Carpenter. Molecular Dynamics Simulations of Tubulin Structure and Calculations of Electrostatic Properties of Microtubules. 2005
8. Clifford E. Felder, Jaime Prilusky, Israel Silman, and Joel L. Sussman. A server and database for dipole moments of proteins. 2007
9. Tim J Stevens, Wayne Boucher, Python Programming for Biology. 2015
10. Nogales, E., Downing, K.H.Tubulin Alpha-Beta dimer, electorn diffraction. 1998

# ДОДАТОК А

Код програми:

```
import math
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def Atom_Koor(mono):
    f=open('1tub.pdb','r')
    x=np.array([])
    y=np.array([])
    z=np.array([])
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                r=(line[30:54].split())
                x=np.append(x,float(r[0]))
                y=np.append(y,float(r[1]))
                z=np.append(z,float(r[2]))
    f.close()
    return [x,y,z]

def Atom_N(mono):
    f=open('1tub.pdb','r')
    Mono=np.array([])
    aaMono=0
    n=0
    tmp=[0,0,0]
    expected_aa = 1
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                if int(line[23:27]) == expected_aa:
                    if aaMono!=0:
                        Mono=np.append(Mono,n)
                    n=0
                    aaMono+=1
                    expected_aa += 1
                    n+=1
                elif int(line[23:27]) < expected_aa:
                    n+=1
```

```
                continue
            else:
                expected_aa += 1
                continue
    if n != 0:
        Mono=np.append(Mono,n)
    f.close()
    return(Mono)

def Center(mol):
    mol=np.array(mol)
    mol_center=np.mean(mol, axis=1)
    return mol_center

def Residue(mol,n):
    mol=np.array(mol)
    center=Center(mol)
    sub_mean_array = np.empty((mol.shape[0], len(n)))
    for idx, val in enumerate(n):
        start_idx = sum(n[:idx])
        end_idx = start_idx + val
        sub_arr = mol[:,start_idx:end_idx]
        sub_mean = np.mean(sub_arr, axis=1)
        sub_mean[0]=sub_mean[0]-center[0]
        sub_mean[1]=sub_mean[1]-center[1]
        sub_mean[2]=sub_mean[2]-center[2]
        sub_mean_array[:, idx] = sub_mean
    return(sub_mean_array)

def Res_Charges(mono):
    f=open('1tub.pdb','r')
    Charges=np.array([])
    aaMono=0
    expected_aa = 1
    for line in f.readlines():
        if line.find('ATOM') == 0:
            if line.find(mono,20,23) >0:
                if int(line[23:27]) == expected_aa:
                    if line.find('LYS') > 0 or line.find('ARG') > 0:
                        Charges=np.append(Charges,1)
                    elif line.find('GLU') > 0 or line.find('ASP') > 0:
                        Charges=np.append(Charges,-1)
                    else:
                        Charges=np.append(Charges,0)
```

```
                aaMono+=1
                expected_aa += 1
            elif int(line[23:27]) < expected_aa:
                continue
            else:
                expected_aa += 1
                continue
    f.close()
    return(Charges)

def Dipole(mol,charges):
    p=0
    i=0
    j=0
    k=0
    tmp=np.array([0.0,0.0,0.0])
    for elements in mol[0]:
        tmp[0]+=elements*charges[i]
        i+=1
    for elements in mol[1]:
        tmp[1]+=elements*charges[j]
        j+=1
    for elements in mol[2]:
        tmp[2]+=elements*charges[k]
        k+=1
    return tmp

def Dip_moment(mol):
    p=math.sqrt(mol[0]*mol[0]+mol[1]*mol[1]+mol[2]*mol[2])
    return(p*4.803)

A_a=Atom_Koor('A')
B_a=Atom_Koor('B')
Dim_a=np.concatenate((A_a, B_a), axis=1)

A_n=Atom_N('A').astype(int)
B_n=Atom_N('B').astype(int)
Dim_n=np.concatenate((A_n, B_n))

A_r=Residue(A_a,A_n)
B_r=Residue(B_a,B_n)
Dim_r=Residue(Dim_a,Dim_n)

A_c=Res_Charges('A')
```

```
        B_c=Res_Charges('B')
        Dim_c=np.concatenate((A_c, B_c))

        A_dip=Dipole(A_r,A_c)
        B_dip=Dipole(B_r,B_c)
        Dim_dip=Dipole(Dim_r,Dim_c)

        A_p=Dip_moment(A_dip)
        B_p=Dip_moment(B_dip)
        Dim_p=Dip_moment(Dim_dip)
        dim_dip2=A_dip+B_dip
        dim_p2=Dip_moment(dim_dip2)

        print('A:                ','Atoms',len(A_a[0]),'        Residue',len(A_r[0]),'
Charge',np.sum(A_c),' Dipole:',A_p)
        print('B:                ','Atoms',len(B_a[0]),'        Residue',len(B_r[0]),'
Charge',np.sum(B_c),' Dipole:',B_p)
        print('Dim:          ','Atoms',len(Dim_a[0]),'          Residue',len(Dim_r[0]),'
Charge',np.sum(Dim_c),' Dipole:',Dim_p)
        print('векторна сума A та B:',dim_p2)

        fig = plt.figure(figsize=(10,7.5))
        ax = fig.add_subplot(111, projection='3d')
        center_A=Center(A_a)
        center_B=Center(B_a)
        center_Dim=Center(Dim_a)
        ax.scatter(A_a[0], A_a[1], A_a[2], c='r', marker='o')
        ax.scatter(B_a[0], B_a[1], B_a[2], c='b', marker='o')
        ax.quiver(center_A[0],center_A[1],center_A[2],A_dip[0],A_dip[1],A_dip[2]
,arrow_length_ratio=0.05, color='blue')
        ax.quiver(center_B[0],center_B[1],center_B[2],B_dip[0],B_dip[1],B_dip[2],
arrow_length_ratio=0.05, color='red')
        ax.quiver(center_Dim[0],center_Dim[1],center_Dim[2],Dim_dip[0],Dim_di
p[1],Dim_dip[2],arrow_length_ratio=0.05,color='black')
        ax.set_xlabel('X')
        ax.set_ylabel('Y')
        ax.set_zlabel('Z')
        plt.show()
```