Математичний модуль

Останнім часом у світі проявляється тенденція використовувати мову програмування `python` для розробки програмних модулів, що мають справу із складними математичними обчисленнями.

Це пов'язано із бажанням дослідників заощадити час на розробці власне програмного забезпечення і зосередитися на математичних проблемах, які вони розв'язують.

Такі заощадження досягаються за рахунок використання великої кількості вже готових програмних модулів, таких як `numpy.linalg` для роботи з лінійною алгеброю та `scipy.integrate` для чисельного інтегрування.

У своєму модулі ми будемо опиратися на ці наробки минулих поколінь дослідників і використовувати їх у повній мірі.

Ми робимо наш програмний модуль відкритим для громадськості. Він доступний онлайн за посиланням. Ми будемо раді будь-яким змістовним доповненням до нього, а також виправленням помилок, якщо такі там раптом присутні.

Власне програмний модуль

Перелік процедур, реалізованих у програмному модулі включно із коротким описом:

- **`solve_system`** – розв'язує (можливо несумісну) систему лінійних алгебраїчних рівнянь, тобто покриває підрозділ 1.1 посібника. Повна документація процедури доступна за посиланням.

- **`solve_summed_system`** – розв'язує лінійну дискретно-підсумовану систему рівнянь. Разом із наступною процедурою покриває підрозділ 1.2 посібника. Повна документація процедури доступна за посиланням.

- **`solve_time_summed_system`** – розв'язує лінійну дискретно-підсумовану систему рівнянь, де окремі значення трактуються як значення певної функції часової змінної у певні дискретні моменти часу. Разом із попередньою процедурою покриває підрозділ 1.2 посібника. Повна документація процедури доступна за посиланням.

- **`solve_distributed_system`** – розв'язує лінійну дискретно-розподілену систему рівнянь. Разом із наступною процедурою покриває підрозділ 1.3 посібника. Повна документація процедури доступна за посиланням.
- **`solve_time_distributed_system`** – розв'язує лінійну дискретно-розподілену систему рівнянь, де окремі значення трактуються як значення певної функції часової змінної у певні дискретні моменти часу. Разом із попередньою процедурою покриває підрозділ 1.3 посібника. Повна документація процедури доступна за посиланням.

- **`solve_integral_system`** – розв'язує лінійну інтегрально-перетворювальну систему рівнянь. Разом із наступною процедурою повністю покриває розділ 1.4 посібника. Повна документація процедури доступна за посиланням.
- **`solve_functional_system`** – розв'язує лінійну функціонально-перетворювальну систему рівнянь. Разом із попередньою процедурою повністю покриває розділ 1.4 посібника. Повна документація процедури доступна за посиланням.

- **`solve_1d_space_distributed_integral_system`** – розв'язує розподілену у просторі інтегрально-перетворювальну систему рівнянь у одновимірному випадку. Повна документація процедури доступна за посиланням.
- **`solve_2d_space_distributed_integral_system`** – розв'язує розподілену у просторі інтегрально-перетворювальну систему

рівнянь у двовимірному випадку. Повна документація процедури доступна за посиланням. Разом із наступними двома процедурами ці дві процедури повністю покривають розділ 1.5 посібника. Не зважаючи на необхідність чисельного обчислення потрійного інтегралу працює відносно швидко, за долі секунди.

- Процедура `solve_3d_space_distributed_integral_system` не була реалізована через необхідність чисельного обчислення чотирикратного інтегралу. Похибка подібних обчислень була б катастрофічною. Окрім цього немає зручного способу візуалізації отриманого розв'язку. І, що найголовніше, подібні задачі не мають фізичного підґрунтя.

- `solve_1d_space_distributed_functional_system` – розв'язує розподілену у просторі функціонально-перетворювальну систему рівнянь у одновимірному випадку. Повна документація процедури доступна за посиланням.
- `solve_2d_space_distributed_functional_system` – розв'язує розподілену у просторі функціонально-перетворювальну систему рівнянь у двовимірному випадку. Повна документація процедури доступна за посиланням. Разом із попередніми двома процедурами ці дві процедури повністю покривають розділ 1.5 посібника. Не зважаючи на необхідність чисельного обчислення потрійного інтегралу працює відносно швидко, за долі секунди.
- Процедура `solve_3d_space_distributed_functional_system` не була реалізована через необхідність чисельного обчислення чотирикратного інтегралу. Похибка подібних обчислень була б катастрофічною. Окрім цього немає зручного способу візуалізації отриманого розв'язку. І, що найголовніше, подібні задачі не мають фізичного підґрунтя.

- `solve_1d_discrete_observations_discrete_modelling` – розв'язує дискретно-спостережувану систему у обмеженій просторово-

часовій області із одновимірною за простором областю. Використовує дискретні моделюючі функції. Разом із наступною процедурою повністю покриває підрозділ 2.2.1 посібника. Повна документація процедури доступна за посиланням.

- `solve_2d_discrete_observations_discrete_modelling` – розв'язує дискретно-спостережувану систему у обмеженій просторово-часовій області із двовимірною за простором областю. Використовує дискретні моделюючі функції. Разом із попередньою процедурою повністю покриває підрозділ 2.2.1 посібника. Повна документація процедури доступна за посиланням.
- Процедура `solve_3d_discrete_observations_discrete_modelling` не була реалізована через надмірну громіздкість коду.

- `solve_1d_discrete_observations_continuous_modelling` – розв'язує дискретно-спостережувану систему у обмеженій просторово-часовій області із одновимірною за простором областю. Використовує неперервні моделюючі функції. Разом із наступною процедурою повністю покриває підрозділ 2.3.1 посібника. Повна документація процедури доступна за посиланням. Не зважаючи на необхідність чисельного обчислення великої кількості інтегралів працює відносно швидко, за долі секунди.
- `solve_2d_discrete_observations_continuous_modelling` – розв'язує дискретно-спостережувану систему у обмеженій просторово-часовій області із двовимірною за простором областю. Використовує неперервні моделюючі функції. Разом із попередньою процедурою повністю покриває підрозділ 2.3.1 посібника. Повна документація процедури доступна за посиланням. Зважаючи на необхідність чисельного обчислення великої кількості подвійних інтегралів працює відносно повільно, може займати до однієї хвилини за великої кількості спостережень.

- Процедура `solve_3d_discrete_observations_continuous_modelling` не була реалізована через необхідність чисельного обчислення великої кількості потрійних інтегралів, що могло б зайняти десятки хвилин, якщо не години обчислень. Окрім цього похибка таких обчислень була б катастрофічною. І, що найголовніше, подібні задачі не мають фізичного підґрунтя.

- `solve_1d_continuous_observations_discrete_modelling` – розв'язує неперервно-спостережувану систему у обмеженій просторово-часовій області із одновимірною за простором областю. Використовує дискретні моделюючі функції. Разом із наступною процедурою повністю покриває підрозділ 2.4.1 посібника. Повна документація процедури доступна за посиланням. Не зважаючи на необхідність чисельного обчислення великої кількості інтегралів працює відносно швидко, за долі секунди.
- `solve_2d_continuous_observations_discrete_modelling` – розв'язує неперервно -спостережувану систему у обмеженій просторово-часовій області із двовимірною за простором областю. Використовує дискретні моделюючі функції. Разом із попередньою процедурою повністю покриває підрозділ 2.4.1 посібника. Повна документація процедури доступна за посиланням. Зважаючи на необхідність чисельного обчислення великої кількості подвійних інтегралів працює відносно повільно, може займати до однієї хвилини за великої кількості спостережень.
- Процедура `solve_3d_continuous_observations_discrete_modelling` не була реалізована через необхідність чисельного обчислення великої кількості потрійних інтегралів, що могло б зайняти десятки хвилин, якщо не години обчислень. Окрім цього похибка таких обчислень була б катастрофічною. І, що найголовніше, подібні задачі не мають фізичного підґрунтя.

Тестування

Враховуючи, що повний код програмного модуля займає 1187 рядків коду, було б неприпустимим покладатися на відсутність жодної друкарської помилки у ньому.

Як наслідок, було написано велику кількість різноманітних тестів, які перевіряють різні властивості розв'язків, які видають реалізовані нами процедури.

Зокрема тестується:

- відповідність знайденого розв'язку точному аналітичному розв'язку для модельних задач для яких він існує;
- обмеженість похибки розв'язку за умов, що точного розв'язку задача на має;
- швидкодія складних процедур на великих вхідних даних, таких як сотні моделюючих точок і десятки умов, що накладаються на задачу.

У зв'язку із таким строгим і всебічним тестування, сам тестовий модуль займає 818 рядків. Але на нього ми вже можемо покластися, адже він не містить у собі ніякої складної логіки.

Наприклад тестування задачі із дискретними спостереженнями і дискретними моделюючими функціями у одновимірному випадку виглядає так:

```
# задаємо точки для початкових
cond_x0s_list = [0.0, 1.0]
# і крайових умов
cond_xtGammas_list = [
    (0.0, 0.5), (0.0, 1.0),
    (1.0, 0.5), (1.0, 1.0),
]

# задаємо значення умов у початкових
cond_f0s_list = [1.0, 1.0]
# і крайових точках
cond_fGammas_list = [
    1.0, 1.0,
```

```
    1.0, 1.0,
]

# моделюючі точки для гладкої частини розв’язку
model_xtInfs_list = [
    (0.0, 0.0), (0.5, 0.0), (1.0, 0.0),
    (0.0, 0.5), (0.5, 0.5), (1.0, 0.5),
    (0.0, 1.0), (0.5, 1.0), (1.0, 1.0),
]
# початкові моделюючі точки
model_x0s_list = [0.0, 0.5, 1.0]
# крайові моделюючі точки
model_xtGammas_list = [
    (0.0, 0.0), (1.0, 0.0),
    (0.0, 0.5), (1.0, 0.5),
    (0.0, 1.0), (1.0, 1.0),
]

# функція правої частини рівняння
def f(x: float, t: float) -> float:
    return 1.0

# функція «Гріна» оператора (сам оператор можна не задавати)
def g(x: float, t: float) -> float:
    return 1.0

# аналітичний розв’язок
def desired(x: float, t: float) -> float:
    return 1.0

# отриманий нами розв’язок
actual = lib.solve_1d_discrete_observations_discrete_modelling(
    cond_x0s_list, cond_xtGammas_list, cond_f0s_list, cond_fGammas_list,
    model_xtInfs_list, model_x0s_list, model_xtGammas_list, f, g)

# точки для звірки аналітичного і нашого розв’язків
xts_list = [
    (0.0, 0.0), (0.5, 0.0), (1.0, 0.0),
    (0.0, 0.5), (0.5, 0.5), (1.0, 0.5),
    (0.0, 1.0), (0.5, 1.0), (1.0, 1.0),
]

# Звіряємо рівність аналітичного і обчисленого нами розв’язків
for x_i, t_i in xts_list:
    np.testing.assert_almost_equal(desired(x_i, t_i), actual(x_i, t_i))
```

Тестовий модуль містить багато прикладів використання реалізованих нами процедур і доступний за посиланням.