УДК 514.18

ОСОБЛИВОСТІ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ТА ФУНКЦІОНАЛЬНОГО ПРОГРАМУВАННЯ МЕТОДІВ ДИФЕРЕНЦІАЛЬНОЇ ГЕОМЕТРІЇ КРИВИХ ЛІНІЙ ТА ПОВЕРХОНЬ

Несвідомін А.В., к.т.н., докторант,
a.nesvidomin@gmail.com
Захарова Т.М., к.т.н., доцент
Національний університет біоресурсів і природокористування України,
(Україна, м. Київ)

Анотація *– будь-які сучасні дослідження, в тому числі геометричне моделювання технічних форм, процесів та явищ методами диференціальної геометрії кривих ліній та поверхонь, потребують розробки відповідного програмного забезпечення в існуючих пакетах символьної математики. На прикладах побудови тригранника Френе показано особливості його програмування в CAS MapleSoft та функцій Wolfram Mathematica.*

Ключові слова *– диференціальна геометрія, криві лінії, поверхні, тригранник Френе, об'єктно-орієнтоване та, функціональне програмування.*

Постановка проблеми. До характерних ознак диференціальної геометрії кривих ліній, як предметної області програмування, віднесемо: 1) необмежену кількість їх назв та різноманітні форми їх аналітичного запису, що потребує формування баз даних, наприклад, за допомогою асоціативних списків; 2) методи дослідження їх геометричних властивостей нараховують десятки назв, наприклад для кривої, знаходження її довжини, дотичної, нормалі, бінормалі, кривини, скруту тощо; 3) подібні дослідження виконуються через виконання ланцюга аналітичних перетворень з використанням диференціально-інтегральних числень, результатом яких є досить громіздкі формули; 4) суттєву роль у дослідженні кривих ліній та поверхонь відіграє необхідність різноманітної візуалізації цих багатовидів для практичного їх використання та інтерактивного діалогу впливу змінних параметрів форм та положень.

Враховуючи вищевикладене, потрібно визначитися у виборі найбільш зручної та ефективної парадигми програмування методів диференціальної геометрії в існуючих пакетах комп'ютерної алгебри (символьних перетворень). Для середнього класу ПК на сьогодні таким CAS-пакетами є взаємно конкуруючі *Maple* [1] і *Mathematica* [2], які різняться стилями програмування, можливостями розширення користувацького функціоналу, засобами інтеграції з іншими середовищами програмування.

Аналіз останніх досліджень. Аналізуючи існуючі дослідження з комп'ютеризації диференціальної геометрії кривих ліній та поверхонь з використанням різноманітних сервісів штучного інтелекту (*LLM*) та кодування (*vibe coding*) на сьогодні можна стверджувати про необхідність органічного поєднання науковця з цими невпинно розвиваючими технологіями. Вибір парадигми програмування залежить від науковця, оскільки впливає не тільки на структуру коду та загальний підхід до розв'язання задачі, але і на спосіб мислення розробника програмного забезпечення, його бачення кінцевого результату.

Формулювання цілей. Розкрити особливості застосування *Mathematica* та *Maple* до розробки відповідно функціонального та об'єктно-орієнтованого програмування методів диференціальної геометрії кривих ліній та поверхонь, їх представлення та оперування.

Основна частина. Виходячи з основних принципів об'єктно-орієнтованого програмування (ООП), комп'ютеризацію диференціальної геометрії кривих ліній та поверхонь в Maple будемо здійснювати як створення системи взаємодіючих багатовидів:

1. *CURVes* - крива лінія, як вектор-функція *r(u)* одної змінної *u*;
2. *SURFace* - поверхня, як вектор-функція *r(u,v)* двох змінних *u* і *v*;
3. *SOLiD* - тіло, як вектор-фунція *r(u,v,w)* трьох змінних *u*, *v* і *w*;
4. *SURF_CURV* - крива лінія *r(t)* на поверхні *r(u,v)* і т.д.

Хоча *Maple* (версія 2024 року) має потужний інструментарій програмування власних процедур (*proc*) та модулів (*module*), але повнофункціональна підтримка принципів об'єктно-орієнтованого програмування (клас, поліморфізм, наслідування) в ньому відсутня. Нижченаведений фрагмент програмного коду демонструє можливість інкапсуляції атрибутів кривої (на прикладі гвинтової лінії, область визначення параметра *u*) та методи її дослідження (кількість яких нараховує більше 60 назв).

```
CURV:=proc(fr::list:=[a cos(u),a sin(u),b u],fu::anything:=u=0..2Pi)
    description "Curve methods":
    module()
      export set,get,..., frene;
frene := proc(uu::anything := NULL, $)
   description "Frene vectors of the curve cr at the point u=ui";
   [tang(uu), norm(uu), binorm(uu)];
end proc;
```

Створення об'єкта "гвинтова лінія" здійснюється викликом процедури CURV:

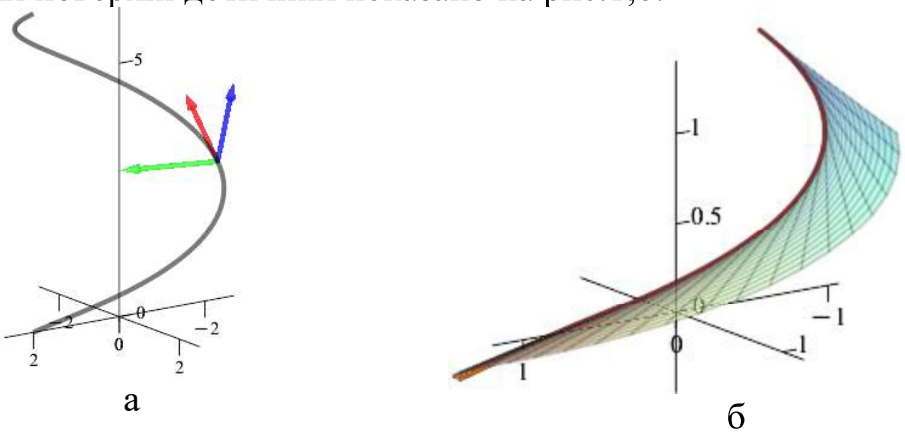
```
o1 := CURV(uabHelixCylinder(u, a, b), u = 0 .. 2 Pi);
```

Визначення ортів тригранника Френе (дотичної, нормалі та бінормалі) в точці $u=\pi$ гвинтової лінії відбувається викликом методу *frene* об'єкта *o1*:

```
o1.-frene(u=Pi);
```

$$\left[\left[-\frac{a\sin(u)}{\sqrt{a^2+b^2}}, \frac{a\cos(u)}{\sqrt{a^2+b^2}}, \frac{b}{\sqrt{a^2+b^2}}\right], [-\cos(u), -\sin(u), 0], \left[\frac{b\sin(u)}{\sqrt{a^2+b^2}}, -\frac{b\cos(u)}{\sqrt{a^2+b^2}}, \frac{a}{\sqrt{a^2+b^2}}\right]\right]$$

Зображення гвинтової кривої з тригранником Френе здійснюється викликом методу *frenePlot* об'єкта *o1* (рис.1,а), а його використання для формування поверхні дотичних показано на рис.1,б.



а б

Рис.1. Гвинтова лінія: а) орти тригранника Френе; б) розгортний гелікоїд

До основних принципів функціонального програмування (ФП) в пакеті *Mathematica* відноситься незмінність даних, декларативний стиль опису задачі, перетворенню даних за допомогою композиції функцій. Вся багатогранність кривих ліній, поверхонь, їх методів дослідження в *Mathematica* реалізується за допомогою пакетів (packages) функцій, як задання багатовидів диференціальної геометрії, так і їх досліджень. Наприклад, створений пакет *curves* включає сотні функцій параметрично заданих кривих ліній у вигляді асоційованого списку – назва кривої, її параметричне рівняння, параметри форми, незалежний параметр, його область визначення тощо. Для гвинтової лінії це буде функція abHelixCylinder$[a, b][u]$, яка повертає асоційований список у вигляді:

$$< |\text{rr} \to \{a\,Cos[u], a\,Sin[u], b\,u\}, \text{uu} \to \{u, 0, 2\pi\}, \text{pp} \to u, \text{ff} \to \text{helixCylinder} > |$$

Інший пакет функцій *curveProperties* включає десятки назв методів дослідження будь-якої кривої, як вектор-функції одної змінної *r[u]*. Знаходження вектора дотичної у символьному вигляді або ж у вигляді числових даних в заданій точці кривої потрібно викликати одну і туж функцію crTangent але з різними параметрами, наприклад:

$$\text{crTangent}\,[\text{abHelixCylinder}[a, b][u], \text{u}] \to \{-Sin[u], Cos[u], 0.5\};$$

$$crTangent[abHelixCylinder[1, 0.5][u], \{u, Pi\}] \to \{0, -1, 0.5\}.$$

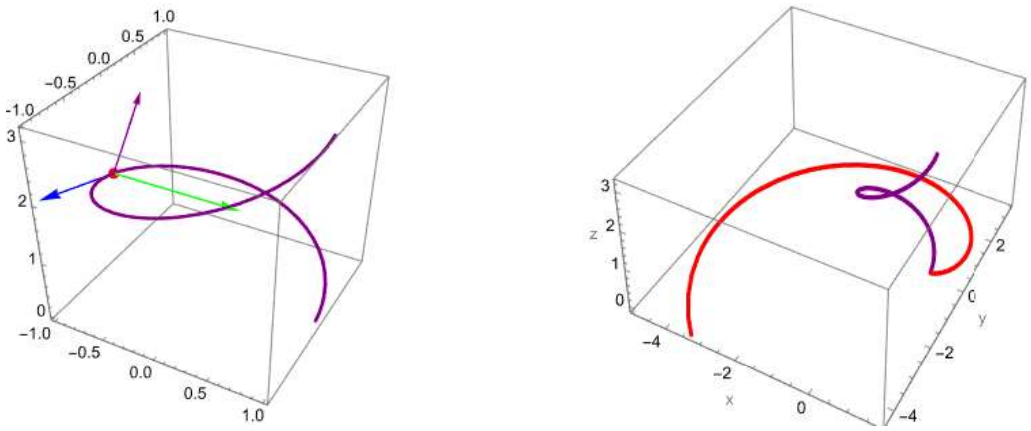
Побудова ортів тригранника Френе в точці $u=Pi$ (рис.2,а) здійснюється викликом функції crTNBfrenetPlot:

$$crTNBfrenetPlot[abHelixCylinder[1, 0.5][u], \{u, Pi\}].$$

Для побудови евольвенти (рис.2,б) гвинтової лінії потрібна функція

$$crEvolventPlot[abHelixCylinder[1, 0.5][u], \{u, 2Pi\}].$$

Звернемо увагу на вкладеність функцій – внутрішня функція визначає задану криву abHelixCylinder$[1, 0.5][u]$ з відповідними параметрами її форми, а зовнішня функція cr$EvolventPlot[\ldots]$ - її властивість (метод). Об'єкт, який зберігає властивості гвинтової лінії, тут відсутній, а дані передаються між функціями у вигляді їх композицій.



а б

Рис.2. Орти тригранника Френе та евольвента гвинтової лінії

Зазначимо, що пакет *Maple* в повній мірі реалізує і функціональне програмування подібно до пакету *Mathematica*.

Таким чином, вибір парадигми програмування тієї чи іншої предметної області залежить не тільки від складності задач, вимог до надійності та зручності супроводу коду, але і від специфіки використовуваної *CAS*, звичками розробника програмного забезпечення. На нашу думку, парадигма об'єктно-орієнтованого програмування більш природно дозволяє моделювати такі геометричні сутності як криві лінії, поверхні, тіла - об'єкти, що об'єднують різнобічні властивості та операції над ними. Цей підхід є інтуїтивно більш зрозумілим при моделюванні конкретних геометричних багатовидів, програмний код є більш компактним. В той же час, функціональне програмування має свої переваги в надійності та передбачуваності кінцевого результату, виразності коду, і що в подальшому кодуванні важливо – це розпаралелювання обчислень (чим *Mathematica* суттєво відрізняється від *Maple*).

Висновки. Оскільки диференціальна геометрія включає як велику кількість досить складних сутностей (криві, поверхні, криві на поверхнях, тіла), так і складні послідовності математичних дій над ними (обчислення кривин, скруту, квадратичних форм тощо), то жодна із парадигм програмування ідеально не покриває обидва аспекти одночасно - ООП організовує код навколо сутностей (об'єктів), тоді як ФП організовує його навколо дій (функцій). Усунення трудомісткості кодування сучасними сервісами штучного інтелекту призведе до можливо\ інтеграції цих двох парадигм програмування диференціальної геометрії кривих ліній та поверхонь.

Бібліографічний список

1. https://www.maplesoft.com/
2. https://www.wolfram.com/mathematica/