УДК 372.851+004

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ МАТЕМАТИЧНОГО ПРИЗНАЧЕННЯ В КУРСІ ФІЗИКИ СЕРЕДНЬОЇ ТА ВИЩОЇ ШКОЛИ

С.В. Шокалюк, С.О. Семеріков
м. Кривий Ріг, Криворізький державний педагогічний університет
ksv_ipm@mail.ru, cc@optima.com.ua

Стаття присвячена застосуванню стандартних функцій системи комп'ютерної математики Maxima для розв'язання задач шкільної фізики та спеціальних функцій вбудованого модуля dynamics для розв'язання задач фізики вищої школи.

Ключові слова: система комп'ютерної математики, комп'ютерне моделювання, динамічні системи, фрактали.

The article is devoted to using of standard functions of CAS Maxima for the decision of problems of school physics, and also special functions of built-in module 'dynamics' for the decision of problems of higher school physics.

Key words: computer aided systems, computer simulation, dynamical systems, fractals.

Постановка проблеми. Розуміння технології навчання фізики як процесуального способу досягнення цілей навчання на основі узгодженого поєднання організаційних форм, методів і засобів навчання дає підстави виділити технології комп'ютерного навчання. Комп'ютерна техніка стає компонентом змісту навчання фізики, засобом оптимізації та підвищення ефективності навчального процесу, вона сприяє реалізації багатьох принципів розвиваючого навчання.

Одним з найбільш перспективних напрямів використання інформаційних технологій у фізичній освіті є комп'ютерне моделювання фізичних процесів і явищ. Комп'ютерні моделі легко вписуються у традиційний урок, дозволяючи вчителю організувати нові нетрадиційні види навчальної діяльності. За умови адекватного використання комп'ютерних моделей можна вирішити багато задач навчання. Окремі напрямки використання комп'ютерного моделювання у

навчальному процесі досліджені в ряді робіт з методики викладання фізики (О.І. Бугайов, М.І. Жалдак, О.М. Желюк, Ю.О. Жук, О.І. Іваницький, В.С. Коваль, А.М. Сільвейстр, В.І. Сумський, І.О. Теплицький та інші).

Високо оцінюючи значення досліджень вітчизняних авторів у визначенні ролі та місця елементів комп'ютерного моделювання в системі дидактичних засобів з фізики, зазначимо, що окремий аспект цієї проблеми, а саме вибір адекватного середовища для побудови моделі, ще не знайшов належного висвітлення в методиці викладання. В навчальному посібнику [4] основним середовищем моделювання обрані електронні таблиці. Автори [3] пропонують зміну середовищ в процесі навчання, зокрема – перехід до мов програмування високого рівня. У посібнику [1] висвітлено можливості пакету GRAN1 в процесі навчання фізики у середній школі. Проте можливості такого потужного засобу, як системи комп'ютерної математики (СКМ), більшість авторів розглядає лише в курсі фізики вищої школи (переважно – в спецкурсах з комп'ютерного моделювання у фізики).

Пам'ятаючи, що СКМ є однією з найбільш поширених інформаційних технологій математичного призначення, *основною метою дослідження* поставимо огляд можливостей СКМ для підтримки шкільного та вузівського курсів фізики.

Основна частина.

І. СКМ у шкільному курсі фізики.

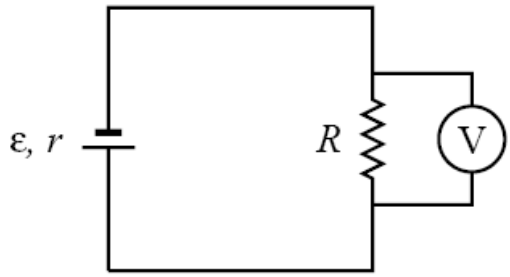
В розділі "Інформаційні технології математичного призначення" шкільного курсу інформатики чільне місце займають системи комп'ютерної математики. Під системами комп'ютерної математики розуміють програмне забезпечення, яке дозволяє не лише виконувати чисельні розрахунки на комп'ютері, а також виконувати аналітичні (символьні) перетворення різних математичних об'єктів. Системи комп'ютерної математики займають визначну роль у сучасній освіті, зокрема дистанційній. Адже можливості графіки таких систем, засоби візуального програмування та мультимедійні технології є потужним інструментарієм для створення електронних посібників, довідників

та дистанційних курсів.

Останнім часом все більшого використання набуває безкоштовна, вільно поширювана, локалізована СКМ Maxima, яка містить потужний інструментарій для виконання операцій над математичними функціями, включаючи диференціювання, інтегрування, розкладання у ряди, перетворення Лапласа, розв'язання звичайних диференційних рівнянь, а також набір функцій для виконання побудови графіків і статистичних даних на площині та у просторі.

Розглянемо приклади розв'язання задач із фізичним змістом у СКМ Maxima.

Приклад 1. Батарея з'єднана з зовнішнім резистором опору R, а напруга через резистор виміряна вольтметром V. Для знаходження електрорушійної сили ε та внутрішнього опору r батареї були використані два зовнішні резистори 1.13 кОм та 17.4 кОм. Падіння напруги в обох випадках становило 6.26 В та 6.28 В. Знайти силу току в обох випадках. Отримати значення ε та r. Побудувати графік розсіяння потужності у зовнішньому опорі як функції від R, якщо значення R змінюється у межах від 0 до $5r$.



Розв'язання

Силу току знайдемо за законом Ома:

$$I = \frac{\Delta V}{R}.$$

За даними значеннями різниці потенціалів ΔV та опору резистору R силу току знайдемо у пакеті Maxima:

```
(%i1) 6.26/1.13e3;
(%o1) 0.0055398230088496
```

Силу току у другому випадку знайдемо аналогічним способом:

```
(%i2) 6.28/17.4e3;
```

```
(%o2) 3.609195402298851*10^-4
```

Таким чином, сила току при опорі 1.13 кОм становить 5.54 мА і при опорі 17.4 кОм – 0.361 мА. Для розрахунку електрорушійної сили і внутрішнього опору використаємо залежність:

$$\Delta V=\varepsilon-rI.$$

Підставляючи два набори значень, дані для ΔV і R, отримаємо систему з двох рівнянь з двома невідомими. Збережемо ці два рівняння у двох змінних системи Maxima, позначивши їх `eq1` та `eq2` відповідно:

```
(%i3) eq1: 6.26 = emf - r*%o1;
(%o3) 6.26=emf-0.0055398230088496*r
(%i4) eq2: 6.28 = emf - r*%o2;
(%o4) 6.28=emf-3.609195402298851*10^-4*r
```

Зазначимо, що в Maxima для надання змінній певного значення використовується символ двокрапка, а не знак рівності. Для звернення до значень результатів виконання попередніх команд використовуємо звернення `%o1` та `%o2`.

Останні два рівняння утворюють систему двох лінійних рівнянь з двома невідомими. Для розв'язання системи рівнянь в Maxima використовуємо функцію `solve`:

```
(%i5) solve([eq1,eq2]);
(%o5) [[r=983100/254569, emf=79952407/12728450]]
```

Запис `[eq1,eq2]` був використаний для створення списку з двох елементів, який очікує команда `solve` при розв'язанні системи рівнянь, а не одного рівняння. Функція `solve` надала результати у вигляді раціональних чисел. Для подання отриманих чисел у форматі дійсного числа з плаваючою комою слід виконати наступну команду:

```
(%i6) %,numer;
(%o6) [[r=3.861821352953423,emf=6.281393806787158]]
```

де символ `%` використано для звернення до результату виконання останньої команди (в даному випадку рівносильно зверненню `%o5`). Таким чином, наближені значення електрорушійної сили та внутрішнього опору становлять

6.2814 В та 3.8618 Ом відповідно.

Електрична потужність розсіяння резистору P визначається за формулою

$$P=RI^2.$$

В свою чергу, сила току через електрорушійну силу та опори визначається як:

$$I=\frac{\varepsilon}{R+r}.$$

Отже, потужність розсіяння в зовнішньому резисторі можна обчислити за формулою:

$$P=R\left(\frac{\varepsilon}{R+r}\right)^2$$

Будуємо графік потужності P як функції від R, використовуючи команду:

```
(%i7) plot2d(R*(6.2814/(R+3.8618))^2, [R, 0, 5*3.8618]);
```

Результат виконання останньої команди представлений на рис. 1.

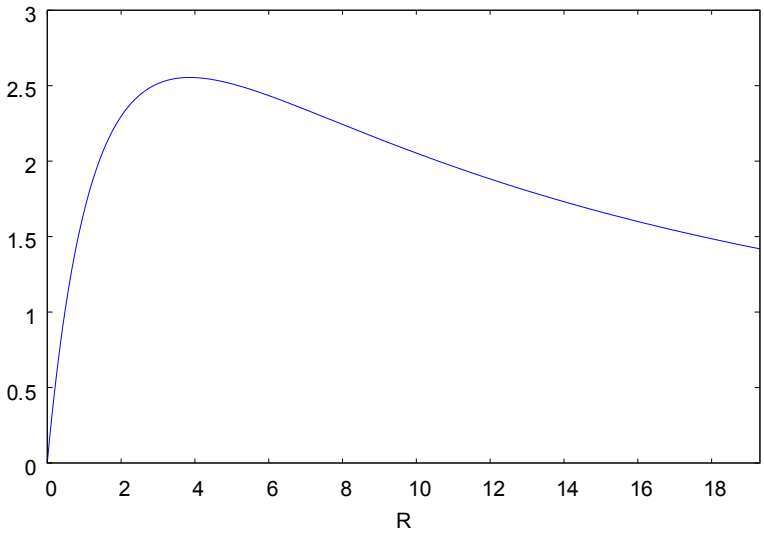


Рис. 1. Потужність розсіяння у зовнішньому резисторі як функція зовнішнього опору

Рухаючи курсором у графічному вікні, ми маємо можливість читати координати точки, на яку вказує курсор миші. А, значить, можемо перевірити, чи приймає потужність розсіяння у зовнішньому резисторі свого максимального значення, коли зовнішній опір дорівнює внутрішньому.

Приклад 2. Координати радіус-вектора, як функції часу, задаються рівнянням:

$$\vec{r}=(5-t^2e^{-t/5})\vec{e}_x+(3-e^{-t/12})\vec{e}_y.$$

Знайти координати вектора, швидкість та прискорення у моменти часу t=0 і t=15 с, та коли час прямує у нескінченність. Побудувати траєкторію руху радіус-вектора упродовж перших 60 секунд руху.

Розв'язання

Почнемо з представлення координат радіус-вектора як списку r, що складається з двох елементів: перший елемент буде визначати абсцису x, другий – ординату y:

```
(%i8) r: [5-t^2*exp(-t/5),3-exp(-t/12)];
(%o8) [5-t^2*%e^(-t/5),3-%e^(-t/12)]
```

Вектор швидкості визначається як похідна радіус-вектора, а вектор прискорення – як похідна вектора швидкості. Для знаходження похідної у пакеті Maxima використовується функція `diff`. Продемонструємо застосування даної функції для знаходження швидкості та прискорення:

```
(%i9) v: diff(r,t);
(%o9) [(t^2*%e^(-t/5))/5-2*t*%e^(-t/5),%e^(-t/12)/12]
(%i10) a: diff(v,t);
(%o10)    [-(t^2*%e^(-t/5))/25+(4*t*%e^(-t/5))/5-2*%e^(-t/5),-%e^(-t/12)/144]
```

Константа `%e` в Maxima представляє число Ейлера e. Для знаходження координат радіус-вектора, швидкості та прискорення слід виконати такі команди:

```
(%i11) r, t=0, numer;
(%o11) [5,2]
(%i12) v, t=0, numer;
(%o12) [0,0.083333333333333]
(%i13) a, t=0, numer;
(%o13) [-2,-0.0069444444444444]
```

Для часу t=15 розрахунки слід виконати аналогічним способом, а саме:

```
(%i14) r, t=15, numer;
(%o14) [-6.202090382769388,2.71349520313981]
(%i15) v, t=15, numer;
(%o15) [0.74680602551796,0.023875399738349]
(%i16) a, t=15, numer;
```

```
(%o16) [0.049787068367864,-0.0019896166448624]
```

Граничні значення, коли час прямує у нескінченність, слід обчислити з використанням функції `limit` (звернення `inf` використовується для позначення математичного символу нескінченності):

```
(%i17) limit(r,t,inf);
(%o17) [5,3]
(%i18) limit(v,t,inf);
(%o18) [0,0]
(%i19) limit(a,t,inf);
(%o19) [0,0]
```

Таким чином, радіус-вектор прямує у точку `[5,3]`.

Для побудови траєкторії руху слід використати опцію `parametric` функції `plot2d`. Компоненти *x* та *y* радіус-вектору будуть подані окремо; команда `plot2d` не буде звертатися до них у списку. Для отримання першої компоненти списку використовується звернення `r[1]`, а другого елемента – `r[2]`:

```
(%i20) plot2d([parametric,r[1],r[2],[t,0,60],[nticks,100]]).
```

Часовий інтервал від 0 до 60 подано як `[t,0,60]`. Опція `nticks` була використана для збільшення кількості інтервалів, оскільки за замовчанням `10` інтервалів продемонструють пунктирну криву замість неперервної траєкторії. Отриманий графік показано на рис. 2.

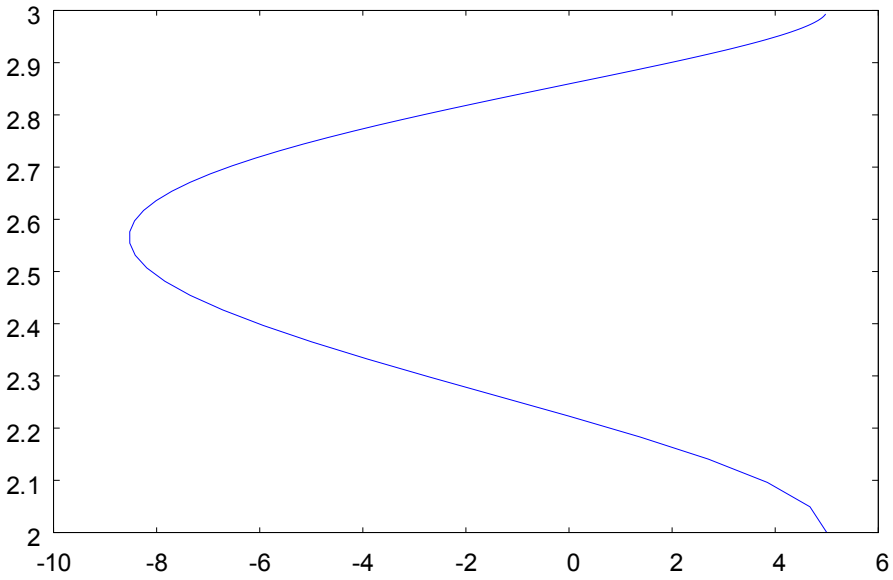


Рис. 2. Траєкторія руху протягом перших 60 секунд,
з початковим моментом часу *t*=5,2

II. Дослідження динаміки систем засобами СКМ.

Бурхливий розвиток теорії динамічних систем у 60-70-ті рр. минулого століття призвели до появи таких понять, як динамічний хаос, та відродження інтересу до фрактальної геометрії. “Динаміка систем” є інтегруючим курсом, що знаходиться на стику математики, фізики та інформатики, і є продовженням курсу механіки, що традиційно викладається студентам перших курсів комп’ютерних спеціальностей. Даний курс був створений на початку 90-х рр. одним з розробників СКМ Maxima – Хайме Віллатом, і успішно впроваджений в ряді університетів Європи [6].

В Криворізькому державному педагогічному університеті значна частина матеріалів цього курсу застосовуються в процесі навчання спецкурсів “Комп’ютерні технології в наукових дослідженнях” та “Автоматика” на спеціальності “Фізика та основи інформатики”.

Застосування СКМ дає можливість швидко та наочно побачити динаміку складних систем, що приводять до утворення фрактальних об’єктів.

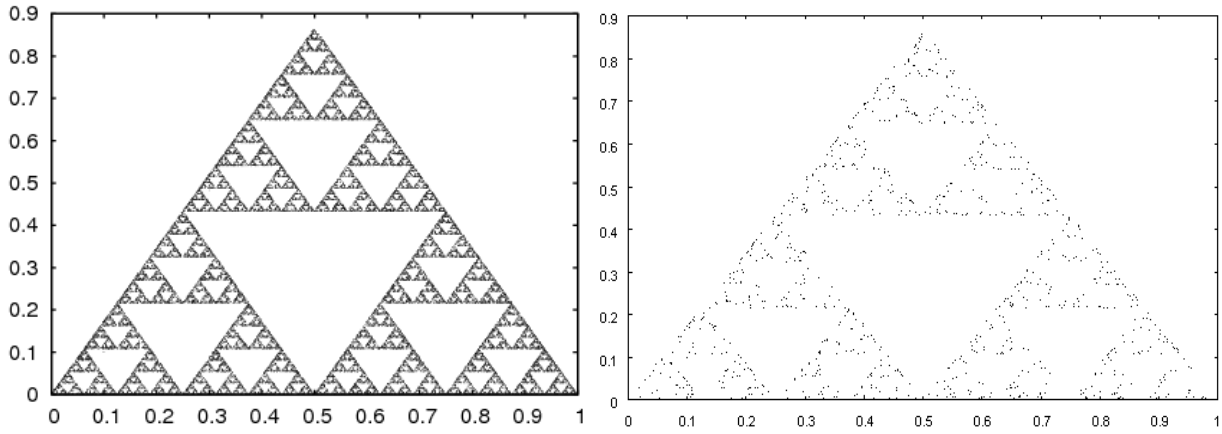
Для візуалізації дискретних динамічних систем та фракталів у СКМ Maxima застосовуються функції пакету `dynamics`, а саме: `chaosgame`, `evolution`, `staircase`, `evolution2d`, `ifs`, `julia`, `mandelbrot`, `orbits` та `rk`. Роботу з функціями пакету `dynamics` слід розпочати з його завантаження командою

```
(%i21) load("dynamics")
```

Функція `chaosgame ([[x1,y1]...[xm,ym]], [x0,y0], b, n, ..., options, ...)` реалізує так звану гру хаосу: спочатку зображується точка `[x0,y0]`, потім одна з *m* точок `[x1,y1]...[xm,ym]` обирається довільним чином. Наступна точка зображується на відрізку, який з’єднує попередню точку з випадково обраною на відстані, що дорівнює довжині цього відрізка, помноженій на *b*. Процедура повторюється *n* разів.

Для побудови серветки Серпинського, як гри хаосу із 30000 точок, необхідно виконати команду (рис. 3*а*):

```
(%i22) chaosgame([[0, 0], [1, 0], [0.5, sqrt(3)/2]], [0.1, 0.1], 1/2, 30000, [style,dots])
```



а *б*

Рис. 3. Серветка Серпинського як гра хаосу (*а* – 30000 точок, *б* – 1000 точок)

Слід зазначити, що процес виконання побудови досить тривалий у часі і може зайняти кілька десятків хвилин навіть при наявності потужного комп'ютера. Рекомендується при низькій швидкодії системи зменшити кількість точок у грі до 1000 – виконання триватиме 1–2 хвилини (рис. 3*б*).

Функція `evolution (F, y0, n, ..., options, ...)` зображує $n+1$ точку на двовимірному графіку, де горизонтальні координати точок є числа 0, 1, 2, 3 і т.д., а вертикальні координати – відповідні значення послідовності, визначені рекурентним співвідношенням $y_{n+1}=F(y_n)$ з початковим значенням `y0`.

Для зображення послідовності точок 3, cos (3), cos(cos(3)), ... виконаємо команду (рис. 4):

```
(%i23) evolution(cos(y),3,10,[style,[points,0.8]]).
```

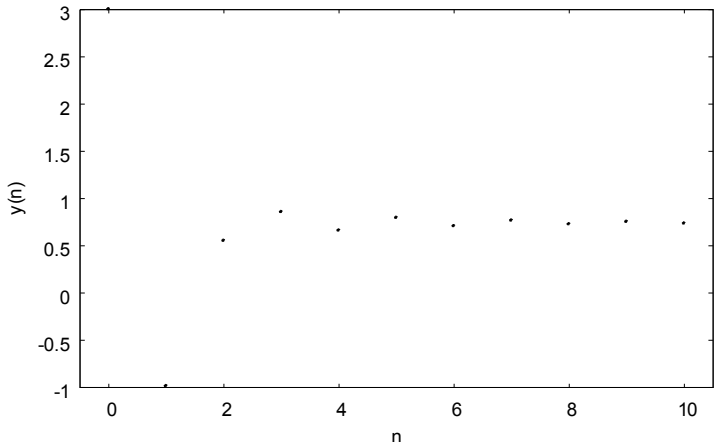


Рис. 4. Графічне подання послідовності точок 3, cos (3), cos(cos(3)), ...

Функція `evolution2d ([F, G], [u, v], [u0, y0], n, ..., options, ...)` зображує на площині першу $n+1$ точку послідовності, що визначається дискретною динамічною системою з рекурентними співвідношеннями $u_{n+1}=F(u_n, v_n)$, $v_{n+1}=G(u_n, v_n)$ з початковими значеннями `u0` и `v0`.

Продемонструємо застосування даної функції для графічного подання послідовності перших 10 чисел Фібоначчі. Виконаємо таку послідовність дій:

```
(%i24) f1:y;
(%i25) f2:x+y;
(%i26) evolution2d([f1,f2],[x,y],[1,1],10).
```

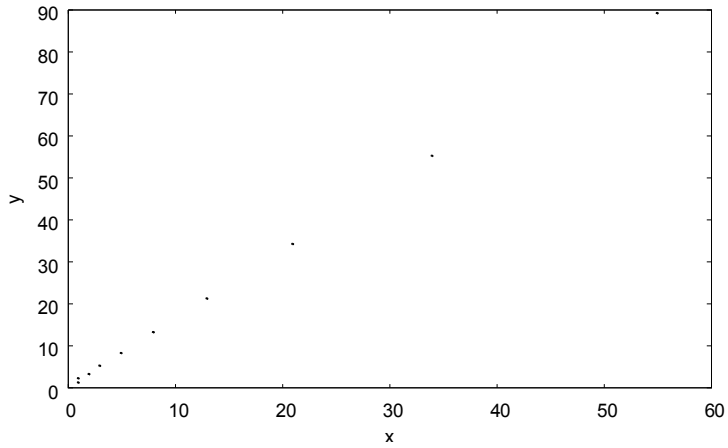


Рис. 5. Графічне подання послідовності чисел Фібоначчі

Функція `staircase (F, y0, n, ...options...)` будує ступінчату діаграму для послідовності з n елементів, що задається рекурентним співвідношенням $y_{n+1}=F(y_n)$ з початковим значенням `y0`.

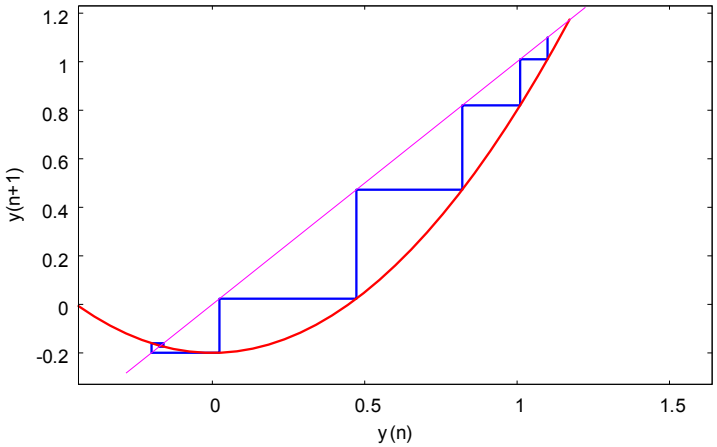


Рис. 6. Графічне подання ступінчатої діаграми для послідовності $y_{n+1}=y_n^2-0.2$

Для побудови ступінчатої діаграми для послідовності $y_{n+1}=y_n^2-0.2$ з

початковим значенням 1.1 слід виконати команду (рис. 6):

```
(%i27) staircase(y^2-0.2,1.1,10)
```

Функція: `ifs ([r1, ..., rm], [A1, ..., Am], [[x1, y1], ..., [xm, ym]], [x0, y0], n, ..., options, ...)` реалізує метод системи ітеративних функцій.

За допомогою системи ітеративних функцій можна побудувати фрактал «Папороть»:

```
(%i28) a1: matrix([0.85,0.04],[-0.04,0.85])$
(%i29) a2: matrix([0.2,-0.26],[0.23,0.22])$
(%i30) a3: matrix([-0.15,0.28],[0.26,0.24])$
(%i31) a4: matrix([0,0],[0,0.16])$
(%i32) p1: [0,1.6]$
(%i33) p2: [0,1.6]$
(%i34) p3: [0,0.44]$
(%i35) p4: [0,0]$
(%i36) prob: [85,92,99,100]
(%i37) ifs(prob, [a1,a2,a3,a4], [p1,p1,p3,p4], [5,0], 50000, [pointsize,0.7]);
```

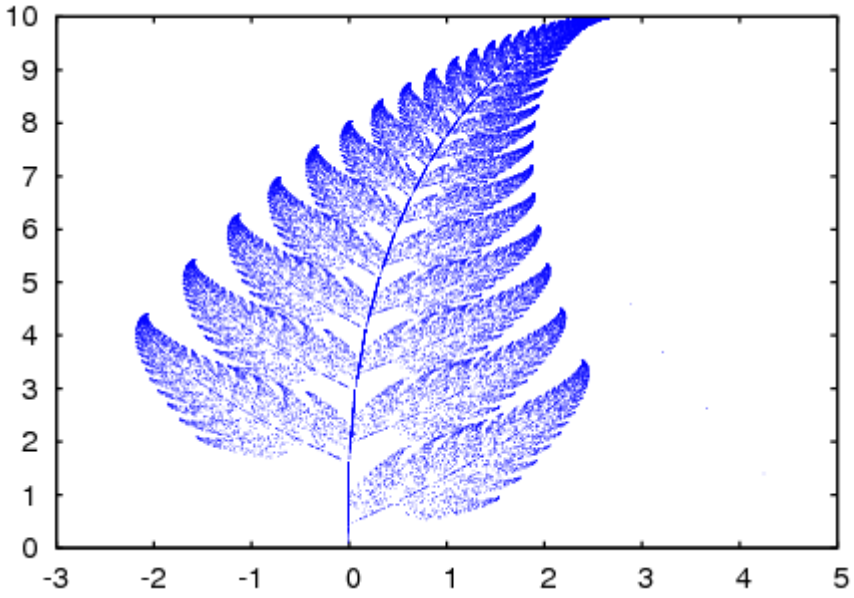


Рис. 7. Фрактал «Папороть»

Функція: `julia(x, y, ...options...)` створює графічний файл із зображенням фракталу Жюліа для комплексного числа $x+iy$. Виконання

програми займає кілька секунд. У результаті користувач отримає повідомлення про створення графічного файлу у форматі XPM. Якщо параметрами функції будуть лише значення x і y, то файл з ім'ям julia.xpm буде створено у поточному каталозі користувача. Перегляд xpm-файлів можна виконати за допомогою Gimp, IrfanView та інших програми для роботи з файлами у бітмап-форматі UNIX.

Приклад фракталу Жюліа для числа $-0.75+0.1i$ при 48 ітераціях наведено на рис. 8 (за замовчуванням виконується 12 ітерацій):

```
(%i38) julia(-0.75,0.1,[levels,48])
```

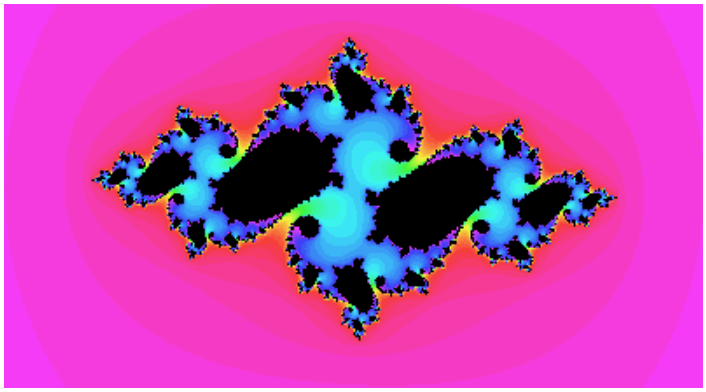


Рис. 8. Фрактал Жюліа для числа $-0.75+0.1i$ (48 ітерацій)

Функція `mandelbrot` створює графічний файл із зображенням фракталу Мандельброта. Так, команда

```
(%i39) mandelbrot([center,0.3,0.5],[radius,0.2],[levels,50])
```

будує фрактал Мандельброта з центром області зображення у точці (0.3, 0.5) і радіусом найбільшого кола 0.2 (рис. 9).

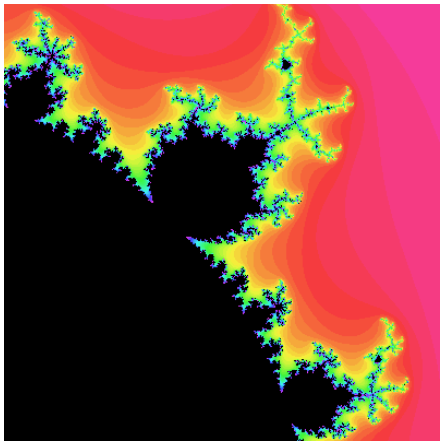


Рис. 9. Фрактал Мандельброта

Функція `orbits (F, y0, n1, n2, [x, x0, xf, xstep], ...options...)` зображує траєкторію сімейства одновимірних дискретних динамічних систем з одним параметром *x*. Даний тип діаграм використовується при вивченні біфуркації одновимірних дискретних систем.

Побудуємо траєкторію для квадратичного відображення $x_{n+1}=x_n^2+a$, де *a* – параметр (рис. 10):

```
(%i40) orbits(x^2+a, 0, 50, 200, [a, -2, 0.25], [style, dots]).
```

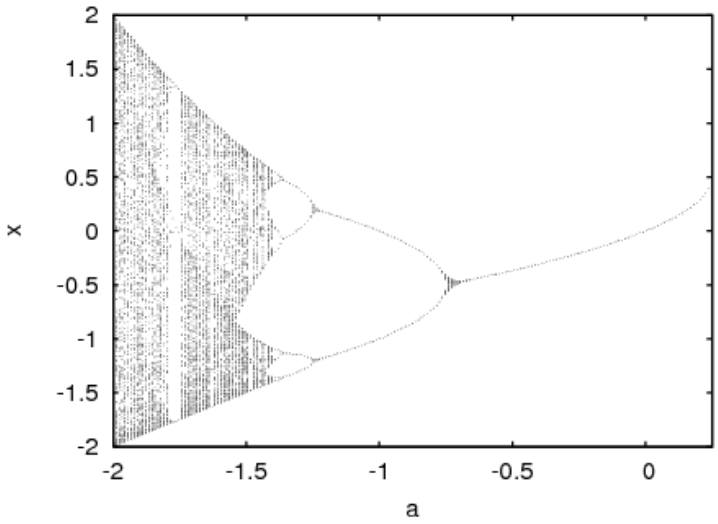


Рис. 10. Траєкторія квадратичного відображення $x_{n+1}=x_n^2+a$

Функція `rk` чисельно розв'язує звичайне диференційне рівняння чи систему рівнянь методом Рунге-Кутта, подаючи результат у вигляді списку значень. Щодо ілюстрації результату розв'язання диференційного рівняння, то більш доцільним є його подання у фазовому просторі [5], для чого застосовується функція `plotdf` однойменного пакету, результатом виконання якої є побудова відповідного поля напрямків.

Висновки:

1. Системи комп'ютерної математики стають незамінними помічниками при вивченні математики, фізики та інформатики, дозволяючи зосередити увагу школярів на змісті задачі та сутності методу її розв'язання.

2. Стандартні можливості СКМ Maxima дозволяють успішно вирішувати шкільні задачі з фізики дослідницького характеру, при цьому немає необхідності відмовлятись від прийнятих у фізиці позначень величин при

побудові математичної моделі задачі.

3. СКМ Maxima має потужний інструментарій для дослідження та графічної інтерпретації динамічних систем різного типу.

4. З метою уникнення залежності часу обчислень від ресурсів комп'ютерної системи перспективним є перехід до Web-систем комп'ютерної математики, застосування яких дає можливість для широкого впровадження технологій дистанційного та мобільного навчання [2].

Література:

1. Жалдак М.І., Набочук Ю.К., Семещук І.Л. Комп'ютер на уроках фізики: Посібник для вчителів. – Рівне: ТЕТІС, 2004. – 230 с.
2. Семеріков С.О., Теплицький І.О., Шокалюк С.В. Нові засоби дистанційного навчання інформаційних технологій математичного призначення // Вісник. Тестування і моніторинг в освіті. – 2008. – №2. – С. 42-50.
3. Соловйов В.М., Семеріков С.О., Теплицький І.О. Інструментальне забезпечення курсу комп'ютерного моделювання // Комп'ютер у школі та сім'ї. – 2000. – №2. – С. 28–32.
4. Теплицький І.О. Елементи комп'ютерного моделювання: Навч. посібник. – Кривий Ріг: КДПУ, 2005. – 208 с.
5. Теплицький І.О., Семеріков С.О. Методика ознайомлення школярів з поняттям фазового простору в курсі фізики // Збірник наукових праць Кам'янець-Подільського державного університету: Серія педагогічна. Випуск 9: Методологічні принципи формування фізичних знань учнів і професійних якостей майбутніх учителів фізики та астрономії. – Кам'янець-Подільський: Кам'янець-Подільський державний університет, інформаційно-видавничий відділ, 2003. – С. 163-165.
6. Villate, J.E. Introdução aos sistemas dinâmicos: uma abordagem prática com Maxima. – Universidade do Porto, 2007. – 222 p.