Волинський національний університет імені Лесі Українки

Кафедра теоретичної та комп'ютерної фізики імені А.В.Свідзинського

Дмитро Шваліковський

CAS Maxima: основи роботи

ЛУЦЬК
2022

УДК 004.94, 517.9

Рекомендовано до друку науково-методичною радою
Волинського національного університету імені Лесі Українки
(протокол №3 від 16 листопада 2022 р.)

Рецензенти:
Луньов С.В., кандидат фіз.-мат. наук, доцент кафедри фізики та вищої математики Луцького НТУ;
Новосад О.В., кандидат фіз.-мат. наук, доцент кафедри експериментальної фізики, інформаційних та освітніх технологій ВНУ імені Лесі Українки.

Шваліковський Д. М. `CAS Maxima`: основи роботи. Луцьк: Вежа-Друк, 2022. 106 с.

Навчально-методичний посібник містить опис програмного пакету `CAS Maxima`, що використовується для розв'язання низки задач математичного аналізу та лінійної алгебри, а також набір завдань для проходження студентами обчислювальної практики. Команди та функції цього застосунку докладно описані із використанням різних опцій та областей застосувань і проілюстровані численними прикладами. Посібник містить 8 розділів, які охоплюють найбільш важливі розділи вищої математики для допомоги студентам фізичних та технічних спеціальностей.

Матеріали розроблені у відповідності до навчальних планів підготовки студентів у галузі знань «10 Природничі науки» спеціальностей «Середня освіта (Фізика)», «Фізика та астрономія», «Прикладна фізика та наноматеріали» у ВНУ імені Лесі Українки.

УДК 004.94, 517.9

Зміст

Вступні зауваги

Походження Maxima

Проект **Maxima** є нащадком DOE Macsyma, що був заснований Енергетичним Управлінням США (Departament of Energy, DOE) наприкінці 1960 року в MIT (Массачусетський технологічний інститут). Macsyma перша створила систему комп'ютерної алгебри, вона проклала шлях для таких програм як Maple і Mathematica. Звичайно, спочатку Macsyma була закритим комерційним проектом. Доступність проекту OpenSource-спільноті стала можливою завдяки професору Техаського Університету Вільяму Шелтеру (William Schelter), який у 1998 році домігся від DOE отримання коду Macsyma та його публікації під ліцензією GPL під назвою **Maxima**. Він же довгий час розробляв як саму **Maxima**, і один із діалектів мови Lisp – GCL (GNU Common Lisp) – на якому розроблялася **Maxima** після її «звільнення». Зараз над проектом працює велика кількість математиків і програмістів на чолі з Джеймсом Емундсоном (James Amundson). На сьогоднішній день пакет досить активно розвивається і багато в чому не поступається таким розвиненим системам комп'ютерної математики, як Maple чи Mathematica.

Структура Maxima

Пакет **Maxima** складається з інтерпретатора макромови, написаного на Lisp, і кількох поколінь пакетів розширень, написаних на макромові пакета або безпосередньо на Lisp. **Maxima** дозволяє вирішувати досить широке коло завдань, що відносяться до різних розділів математики.

Області математики, що підтримуються в **Maxima**:

- операції з поліномами (маніпуляція раціональними та степеневими виразами, знаходження коренів тощо);
- обчислення з елементарними функціями, у тому числі з логарифмами, експонентними функціями, тригонометричними функціями;
- обчислення зі спеціальними функціями, зокрема еліптичними, гамма-, дзета-функціями та інтегралами;
- обчислення границь та похідних;
- аналітичне обчислення визначених та невизначених інтегралів;
- вирішення інтегральних рівнянь;
- розв'язання рівнянь алгебри та їх систем;
- операції зі степеневими рядами та рядами Фур'є;

- операції з матрицями та списками, велика бібліотека функцій для вирішення завдань лінійної алгебри;
- операції з тензорами;
- теорія чисел, теорія груп, абстрактна алгебра.

Дослідниками розроблений ряд додаткових пакетів для Maxima, які можна завантажувати перед використанням, та які істотно розширюють її можливості і коло розв'язуваних завдань.

Основними перевагами програми **Maxima** є:

- можливість вільного використання (**Maxima** відноситься до класу вільних програм та поширюється на основі ліцензії GNU);
- можливість функціонування під управлінням різних ОС (зокрема Linux та Windows);
- невеликий розмір програми;
- широкий клас розв'язуваних завдань;
- можливість роботи як у консольній версії програми, так і з використанням одного із графічних інтерфейсів (XMaxima, **wxMaxima** або як плагін (plug-in) до редактора TexMacs);
- розширення **wxMaxima** (що входить до комплекту поставки) надає користувачеві зручний і зрозумілий інтерфейс, позбавляє необхідності вивчати особливості введення команд для вирішення типових завдань;
- інтерфейс програми українською мовою;
- наявність довідки та інструкцій щодо роботи з програмою.

Розділ 1

Основи роботи з Maxima

1.1 Інтерфейс wxMaxima

Після запуску **wxMaxima** маємо вікно введення команд з панелями швидкого доступу до найбільш вживаних операцій **Maxima**.

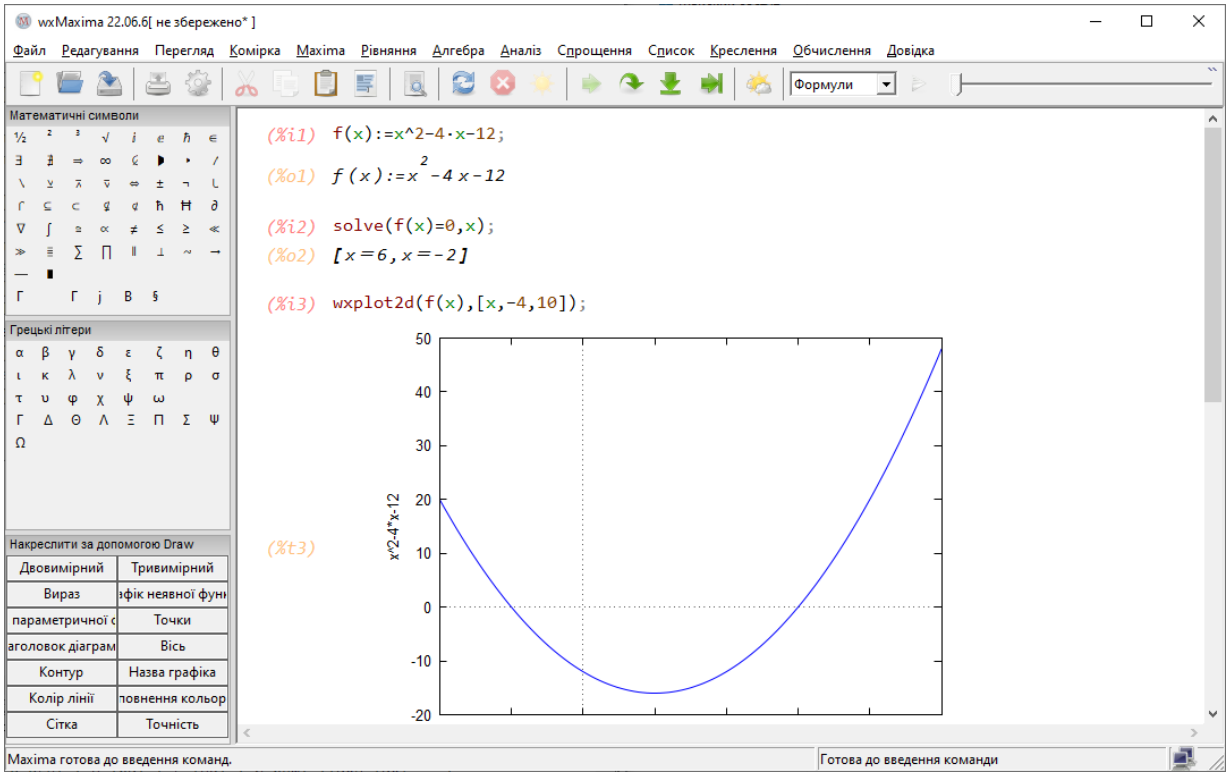

Рис. 1.1: Інтерфейс **wxMaxima**

Здійснимо найпростішу операцію – перемножимо два числа: вводимо команду з клавіатури 111*123 та натиснемо на `Shift+Enter`.

```
(%i1)  111*123;

(%o1)     13653
```

Таким чином, у документі були сформовані дві лінії: `(%i1)` – вхідна команда та `(%o1)` – відповідний їй вихідний результат. Кожна лінія має власну назву, поміщену в дужки. Це дуже зручно, оскільки замість повторного введення довгої команди можна просто вказати дужку `(%oN)`.

Система **wxMaxima** передбачає можливість введення декількох команд одразу в одному рядку. Для цього треба ввести команди, розділені символами «;». У цьому випадку формується одна лінія введення та кілька ліній виведення:

```
(%i4)  a:10; b:5; a*b;
(%o2)  10
(%o3)  5
(%o4)  50
```

Запуск команди на виконання за замовчуванням здійснюється комбінацією `Shift+Enter`, тоді як просто `Enter` переносить курсор на інший рядок. Це необхідно для форматування тексту та розбивки довгих команд на менші, більш зручні для читання та впорядкування. Втім, оптимальним видається запускати виконанням натиском `Enter`, тому при першому запуску рекомендується поставити в Налаштуваннях галочку на пункті «Enter для обчислення в комірках», тоді ці дві опції поміняються місцями. Надалі будемо вважати, що такий вибір зроблено.

Будь-яка команда в **Maxima** повинна закінчуватись знаком «;». Пакет **wxMaxima** автоматично додає його в кінці при натисненні `Enter`, але в інших інтерфейсах (консольний чи **XMaxima**) необхідно записувати його явно. Щоб вказати на кінець командного рядка, можна замість крапки з комою використовувати знак $. Це зручно, якщо виведення проміжних розрахунків на екран не потрібні, тоді їх можна приховати. Прихований результат все одно буде обчислений. Наприклад,

```
(%i5)  R:10$ L:5$ S=%pi*R*(R+L),numer;
```

(%o5) $S = 471.2388980384689$

Як бачимо, для виведення числового значення величини S була додана команда `numer`. Справа в тому, що **Maxima** завжди намагається залишатись у точних математичних рамках, де ірраціональні числа (π, e, $\sqrt{2}$,...) залишаються необчисленими. Без цього уточнення вивід дав би вираз $S = 150\pi$. Схожу функцію має команда `float`, а команда `bfloat` видає результат у нормальній формі числа, тобто після першої значущої цифри йде кома з десятковими значеннями, в кінці bN, що означає $\cdot 10^N$.

За замовчуванням **wxMaxima** відображає 16 цифр після коми. Це значення можна змінити двома способами: задавши команду `fpprec:N`, де `N` – бажана кількість цифр, або використавши меню «Обчислення – Встановити підвищену точність обчислень».

```
(%i6)  bfloat(150*%pi);
```

(%o6) $4.7123889803846 9b2$

Назви функцій та змінних у системі **Maxima** чутливі до реєстру, тобто великі та малі літери відрізняються.

```
(%i7)  sin(%pi/2);
(%o7)  1

(%i8)  Sin(%pi/2);
```

(%o8) $\mathrm{Sin}\left(\frac{\pi}{2}\right)$

Значення імен змінних зберігаються протягом всієї роботи з документом. Якщо необхідно зняти означення з якоїсь змінної `var`, це можна зробити за допомогою команди `kill(var)`, причому це може бути як ім'я змінної, так і комірка введення чи виведення.

Також можна очистити всю пам'ять сеансу і звільнити всі імена, використавши команду `kill(all)`, або вибрати пункт меню «Maxima – Спорожнити пам'ять». Тоді нумерація комірок знову почнеться з одиниці.

Якщо навести курсор миші на верхній лівий кут квадратної дужки та клацнути лівою кнопкою, то ця частина документу згорнеться, повторне клацання миші поверне клітинку до початкового стану. При потребі вставити додаткову комірку між двома, необхідно клікнути мишею на проміжку між клітинами, щоб з'явилась горизонтальна лінія; після цього можна вводити нову команду. Щоб видалити комірку, необхідно виділити її зліва мишкою, та натиснути `Del`. Надзвичайно зручним інструментом є контекстне меню виділеної комірки, при натисненні правої кнопки миші: можна скопіювати цю комірку як команду **Maxima**, як код LaTex або як малюнок. **wxMaxima** має і певні можливості для форматування звичайного тексту. При наборі можна користатись стилями, заголовками різних рівнів, можна також вставляти малюнки просто у тіло документу.

Особливістю графічного інтерфейсу системи **wxMaxima** є те, що коли в робочому вікні відкривається раніше збережений документ, будуть відображаються лише команди введення, тоді як клітини з результатами не відображатимуться. Для їх виконання потрібно натиснути пункт меню «Комірка – Обчислити всі комірки».

1.2 Довідка

При роботі з виразами буває необхідним дізнатись синтаксис команди або пригадати написання додаткових опцій при введенні команд. Для цього існує обширна довідкова система, яка викликається в меню «Довідка – Довідка з Maxima». Ця дія виводить на екран документ з розбитими на розділи можливостями **Maxima** та повним списком всіх команд **Maxima** з їх прикладами.

Однак зручним є також використання довідки у самій системі **wxMaxima** , що здійснюється двома основними способами: командами `example` та `??`. Перша команда дає приклад синтаксису команди з результатами виконання:

```
(%i8)  example(ratsimp);
```

(%i9) $\mathrm{b}*(\mathrm{a}/\mathrm{b}-\mathrm{x})+\mathrm{b}*\mathrm{x}+\mathrm{a}$

(%o9) $b\cdot x+b\cdot\left(\frac{a}{b}-x\right)+a$

```
(%i10) ratsimp(%)
```

(%o10) $2\cdot a$

Друга команда дублює відповідну інформацію з Довідки:

```
(%i11) ?? ratsimp;
0: fullratsimp  (Functions and Variables for Polynomials)
1: ratsimp  (Functions and Variables for Polynomials)
2: ratsimp <1>  (Functions and Variables for Polynomials)
3: ratsimpexpons  (Functions and Variables for Polynomials)};
Enter space-separated numbers, 'all' or 'none': 1
```

1.3 Базові операції, константи, функції

Maxima працює з так званими *атомами* – це число, символ або список (множина чисел чи символів). Основними в роботі з символьними виразами є два оператори: присвоєння «:» та функціональної залежності «:=».

Оператор присвоєння є операцією заміни одного типу атому на інший, наприклад символ на число, або масив чисел.

```
(%i2)  a:2; b:[3,4,5];
```

(%o1) 2

(%o2) $[3, 4, 5]$

```
(%i3)  b^a*c;
```

(%o3) $[9c, 16c, 25c]$

```
(%i4)  [d,e,f]:[-1,-2,-3];
```

(%o4) $[-1, -2, -3]$

```
(%i5)  e;
```

(%o5) -2

Функціональна залежність вказує явно на незалежні змінні функції, по яких її можна диференціювати, інтегрувати, підставляти конкретні значення змінних та ін.

```
(%i6)  g(x,y):=x^2*sin(y);
```

(%o6) $\mathrm{g}(x, y) := x^2 \cdot \sin(y)$

```
(%i7)  diff(g(x,y),x);
```

(%o7) $2 \cdot x \cdot \sin(y)$

```
(%i8)  g(2,3*%pi/2);
```

(%o8) -4

Математичні операції є цілком звичними: додавання (+), віднімання (−), множення (*), ділення (/), піднесення до степеня (^), факторіал (!).

Вбудовані константи: `%e` – число e; `%pi` – число π; `%i` – уявна одиниця $\sqrt{-1}$; `%phi` – золоте відношення $(1+\sqrt{5})/2$; `inf` – додатня дійсна нескінченність $+\infty$; `minf` – від'ємна дійсна нескінченність $-\infty$; `infinity` – комплексна нескінченність.

Зарезервовані слова (їх не можна використовувати як назви функцій чи виразів): `af; else; ic2; plog; and; elseif; if; psi; av; erf; ift; product; args; ev; ilt; put; array; exp; in; rat; at; inf; rk; fix; integrate; step; for; is; sum; col; from; li; then; limit; thru; min; del; get; next; unless; diag; go; not; diff; while; do; ic1; or; zeta`.

Команди вбудованих функцій:

`sin(x)`	$\sin x$	`sinh(x)`	$\mathrm{sh}\ x$
`cos(x)`	$\cos x$	`cosh(x)`	$\mathrm{ch}\ x$
`tan(x)`	$\mathrm{tg}\ x$	`tanh(x)`	$\mathrm{th}\ x$
`cot(x)`	$\mathrm{ctg}\ x$	`coth(x)`	$\mathrm{cth}\ x$
`asin(x)`	$\arcsin x$	`log(x)`	$\ln x$
`acos(x)`	$\arccos x$	`sqrt(x)`	$\sqrt{x}$
`atan(x)`	$\mathrm{arctg}\ x$	`abs(x)`	$\lvert x \rvert$
`acot(x)`	$\mathrm{arcctg}\ x$	`exp(x)`	e^x

Розділ 2

Поліноми, вирази, рівняння

Maxima має широкий вибір інструментів для роботи з виразами, зокрема спрощення та інші перетворення: автоматичне розкриття дужок та винесення за дужки; арифметичні дії над елементами поліномів, а також над виразами із вмістом степеневих, показникових та логарифмічних функцій; обробка тригонометричних виразів тощо. Всі ці дії покликані полегшувати читаність математичних формул і підвищувати простоту їх сприйняття, при правильному використання даних маніпуляцій вони дозволять заощадити в процесі роботи значну кількість часу.

2.1 Перетворення раціональних виразів

Математики раціональним називають вираз, що складається тільки з арифметичних операторів та піднесення до натурального степеня; природно, елементи таких виразів можуть містити і неарифметичні та нестепеневі функції, тоді такі елементи з точки зору раціональних виразів вважаються атомарними, тобто неподільним і неперетворюваними.

Нехай маємо поліноміальний вираз $P(x) = a + bx + cx^2 + dx^3$. **Maxima** дозволяє працювати з кожним елементом цього виразу окремо.

```
(%i1)  P:a+b*x+c*x^2+d*x^3;
```

$$(\%o1) \quad d \cdot x^3 + c \cdot x^2 + b \cdot x + a$$

`first(P)`– повертає перший елемент `P`.

```
(%i2)  first(P);
```

$$(\%o2) \quad d \cdot x^3$$

`last(P)` – повертає останній елемент `P`.

```
(%i3)  last(P);
```

$$(\%o3) \quad a$$

`rest(P)` – повертає `P` з вилученим першим елементом.

```
(%i4)  rest(P);
```

$$(\%o4) \quad c \cdot x^2 + b \cdot x + a$$

`part(P,3)` – виділяє третю частину виразу `P`.

```
(%i5)  part(P,3);
```

$$(\%o5) \quad b \cdot x$$

Необхідно звернути увагу, що **Maxima** нумерує частини виразів саме у виведеному блоці `(%o1)`, а не введеному `(%i1)`, потрібно уважно простежити за цим щоб не було помилок.

`ev(expr,arg1,arg2,...)` – основна функція, що обробляє вирази. Вона обчислює вираз `expr` в оточенні, що визначаються аргументами `argN`. Це можуть бути ключі, булеві вирази, присвоєння, рівняння, функції. На функцію `expr` за замовчуваням діє спрощення виразів. Аргументами `argN` можуть бути і вбудовані команди: `expand, factor, trigexpand, trigreduce, diff`.

```
(%i1) ev((a+b)^2,a=2*x,b=[1,2,3]);
```

$$(\%o1) \quad [(2\cdot x+1)^2,(2\cdot x+2)^2,(2\cdot x+3)^2]$$

```
(%i2) ev((a+b)^2,expand);
```

$$(\%o2) \quad b^2+2\cdot a\cdot b+a^2$$

`factor(expr)` – розбиває вираз на множники, виконуючи одночасно дії спрощення, тобто зведення подібних множників чи членів суми, скорочення, розкладання і пониження степеня.

```
(%i3) factor(2^63+1);
```

$$(\%o3) \quad 3^3\cdot 19\cdot 43\cdot 5419\cdot 77158673929$$

```
(%i4) factor(1+exp(3*x));
```

$$(\%o4) \quad \left(e^x+1\right)\cdot\left(e^{2\cdot x}-e^x+1\right)$$

```
(%i5) factor((x^2-2*x*y+y^2)/(x^2-y^2));
```

$$(\%o5) \quad -\frac{y-x}{y+x}$$

`gfactor(expr)` – функція, аналогічна до `factor`, лише на полі комплексних чисел.

```
(%i6) gfactor(x^4-y^4);
```

$$(\%o6) \quad -(y-x)\cdot(y+x)\cdot(y-i\cdot x)\cdot(y+i\cdot x)$$

`factorsum(expr)` – факторизує окремі доданки у виразі (а `gfactorsum` – на полі комплексних чисел).

```
(%i7) factorsum(a*x*z^2+a*z^2+2*a*w*x*z+2*a*w*z+
      a*w^2*x+v^2*x+2*u*v*x+u^2*x+a*w^2+v^2+2*u*v+u^2);
```

$$(\%o7) \quad (x+1)\cdot\left(a\cdot(z+w)^2+(v+u)^2\right)$$

```
(%i8) gfactorsum(a^3+3*a^2*b+3*a*b^2+b^3+x^2+2*%i*x*y-y^2);
```

$$(\%o8) \quad (b+a)^3-(y-i\cdot x)^2$$

`expand(expr)` – функція, протилежна до `factor`, вона розкриває дужки, перемножуючи та підносячи до степеня.

```
(%i9) expand((a-b)^4);
```

$$(\%o9) \quad b^4-4\cdot a\cdot b^3+6\cdot a^2\cdot b^2-4\cdot a^3\cdot b+a^4$$

```
(%i10) (sqrt(2)+1)^5;
```

$$(\%o10) \quad \left(\sqrt{2}+1\right)^5$$

```
(%i11) expand(%o10);
```
(%o11) $29 \cdot \sqrt{2} + 41$

```
(%i12) (%o10),numer;
```
(%o12) 82.01219330881976

`expandwrt(expr,x,y,...)` – розкриває дужки не скрізь, а лише відносно тих символів, які задані в списку аргументів.

`combine(expr)` – об'єднує доданки з однаковими знаменниками.

```
(%i13) combine(a/(1+a^2)+b/(1+a^2)+b/(1+a));
```
(%o13) $\dfrac{b+a}{a^2+1} + \dfrac{b}{a+1}$

`xthru(expr)` – приводить вираз до спільного знаменника, не розкриваючи дужок і не намагаючись факторизувати доданки.

```
(%i14) xthru( 1/(a+b)^9+1/(a+b)^11);
```
(%o14) $\dfrac{(b+a)^2+1}{(b+a)^{11}}$

`multthru(mult,expr)` – домножує кожен доданок в сумі `expr` на множник `mult`, причому при множенні дужки не розкриваються.

```
(%i15) multthru ((a-b)^3,1/(a-b)^2+b/(a-b)+1);
```
(%o15) $(a-b)^2 \cdot b - b + (a-b)^3 + a$

`divide(expr1,expr2)` – обчислює частку і остачу від ділення многочлену `expr1` на многочлен `expr2`.

```
(%i16) divide(a^3-3,a-1);
```
(%o16) $[a^2+a+1, -2]$

Останній вираз означає, що

$$\frac{a^3-3}{a-1} = a^2+a+1-\frac{2}{a-1}.$$

`gcd(expr1,expr2,...)` – знаходить найбільший спільний знаменник виразів.

```
(%i17) gcd(a/(a+b),b/(a+b)^2);
```
(%o17) $\dfrac{1}{b^2+2\cdot a\cdot b+a^2}$

`num(expr)` та `denom(expr)` – видає чисельник та знаменник виразу expr.

```
(%i18) G:(a-b)/(a^2+b^2);
```
(%o18) $\dfrac{a-b}{b^2+a^2}$

```
(%i19) num(G);
```
(%o19) $a-b$

```
(%i20) denom(G);
```
(%o20) b^2+a^2

`rat(expr)` – перетворює раціональний вираз на вираз канонічної форми, тобто розкриває всі дужки, потім все зводить до спільного знаменника, сумує і скорочує.

```
(%i21) ((x - 2*y)^4/(x^2 - 4*y^2)^2 + 1)*(y + a)*(2*y + x) /
       (4*y^2 + x^2);
```

$$(\%o21)\quad \frac{(y+a)\cdot(2\cdot y+x)\cdot\left(\frac{(x-2\cdot y)^4}{(x^2-4\cdot y^2)^2}+1\right)}{4\cdot y^2+x^2}$$

```
(%i22) rat(%o21);
```

$$(\%o22)/R/\quad \frac{2\cdot y+2\cdot a}{2\cdot y+x}$$

`ratsimp(expr)` – приводить всі частини (в тому числі аргументи функцій) виразів, які не є дробово-раціональною функцією, до канонічного представлення, здійснюючи спрощення, що не виконує команда `rat`. Повторний виклик команди в загальному випадку може змінити результат, тобто не обов'язково спрощення відбувається до кінця.

```
(%i23) ratsimp((x-1)^(3/2)-(x+1)*sqrt(x-1))/sqrt((x-1)*(x+1));
```

$$(\%o23)\quad -\frac{2\cdot\sqrt{x-1}}{\sqrt{(x-1)\cdot(x+1)}}$$

```
(%i24) ratsimp(%o23);
```

$$(\%o24)\quad -\frac{2\cdot\sqrt{x-1}}{\sqrt{x^2-1}}$$

`fullratsimp(expr)` – викликає команду `ratsimp` до тих пір, поки вираз не перестане змінюватися.

```
(%i25) H:(x^(a/2)+1)^2*(x^(a/2)-1)^2/(x^a-1);
```

$$(\%o25)\quad \frac{\left(x^{\frac{a}{2}}-1\right)^2\cdot\left(x^{\frac{a}{2}}+1\right)^2}{x^a-1}$$

```
(%i26) ratsimp(H);
```

$$(\%o26)\quad \frac{x^{2\cdot a}-2\cdot x^a+1}{x^a-1}$$

```
(%i27) fullratsimp(H);
```

$$(\%o27)\quad x^a-1$$

`ratexpand(expr)` – разкриває дужки в виразі `expr`. Відрізняється від `expand` тим, що приводить вираз до канонічної форми.

```
(%i28) J:(x-1)/(x+1)^2+1/(x-1);
```

$$(\%o28)\quad \frac{x-1}{(x+1)^2}+\frac{1}{x-1}$$

```
(%i29) expand(J);
```

$$(\%o29)\quad \frac{x}{x^2+2\cdot x+1}-\frac{1}{x^2+2\cdot x+1}+\frac{1}{x-1}$$

```
(%i30) ratexpand(J);
```

$$(\%o30)\quad \frac{2\cdot x^2}{x^3+x^2-x-1}+\frac{2}{x^3+x^2-x-1}$$

```
(%i31) combine(%o30);
```

$$(\%o31)\quad \frac{2\cdot x^2+2}{x^3+x^2-x-1}$$

`ratsubst(expr1,expr2,expr3)` – команда підстановки виразу `expr1` замість виразу `expr2` у многочлені `expr3`.

```
(%i31) ratsubst (a,x*y^2,x^4*y^3+x^4*y^8);
```

$$(\%o31)\quad a\cdot x^3\cdot y+a^4$$

`partfrac(expr,var)` – перетворює дробово-раціональний вираз `expr` в прості дроби по заданій змінній `var` (це важливо зокрема при інтегруванні складних дробів).

```
(%i32) K:-x/(x^3+4*x^2+5*x+2);
```

$$(\%o32)\quad -\frac{x}{x^3+4\cdot x^2+5\cdot x+2}$$

```
(%i33) partfrac(K,x);
```

$$(\%o33)\quad \frac{2}{x+2}-\frac{2}{x+1}+\frac{1}{(x+1)^2}$$

`at(expr,[x=a,y=b,...])` – підставляє у вираз `expr` замість змінних `x,y,...` вирази `a,b,...`, причому змінні `x,y,...` також можуть бути виразами.

```
(%i34) at(x^2+y^3+z^4,[x^2=a,y^3=b,z^4=c]);
```

$$(\%o34)\quad c+b+a$$

`rhs(expr)`, `lhs(expr)` – відповідно права та ліва частина виразу `expr`.

2.2 Показникові, логарифмічні вирази

Функція `radcan(expr)` – спрощує вирази, що містять експоненти, логарифми та радикали, шляхом перетворення до форми, що є канонічною для широкого класу виразів. Змінні в виразах впорядковуються. Еквівалентні вирази в цьому класі не обов'язково однакові, але їх різниця застосуванням `radcan` зводиться до нуля.

```
(%i1)  K:(log(x+x^2)-log(x))^a/log(1+x)^(a/2);
```

$$(\%o1)\quad \frac{(\log(x^2+x)-\log(x))^a}{\log(x+1)^{\frac{a}{2}}}$$

```
(%i2)  radcan(K);
```

$$(\%o2)\quad \log(x+1)^{\frac{a}{2}}$$

```
(%i3)  L:(%e^x-1)/(1+%e^(x/2));
```

$$(\%o3)\quad \frac{e^x-1}{e^{\frac{x}{2}}+1}$$

```
(%i4)  radcan(L);
```

$$(\%o4)\quad e^{\frac{x}{2}}-1$$

Функція `logcontract(expr)` – рекурсиво сканує вираз `expr`, перетворюючи вирази виду $a\log b+c\log d+C$ до форми $\log(\texttt{ratsimp}(b^a\cdot d^c))+C$ (числа a та c попередньо необхідно оголосити цілими).

```
(%i5)  declare(n,integer);
```
(%o5) *done*

```
(%i6)  logcontract(3*(2*n*log(x)+3*n*log(y)));
```
(%o6) $\log\left(x^{6\cdot n}\cdot y^{9\cdot n}\right)$

2.3 Тригонометричні вирази

Перетворення тригонометричних виразів здійснюється за допомогою схожих команд, як і у випадку раціональних виразів, лише на початку ставиться префікс `trig`. Для найкращого результату їх слід комбінувати з командами `ratsimp`, `fullratsimp`, `radcan` та ін.

`trigexpand(expr)` – розкладає всі тригонометричні та гіперболічні функції від сум і добутків в комбінації відповідних функцій одиничних кутів та аргументів.

```
(%i1)  F:x+sin(3*x)/sin(x);
```
(%o1) $\dfrac{\sin\left(3\cdot x\right)}{\sin\left(x\right)}+x$

```
(%i2)  trigexpand(F);
```
(%o2) $\dfrac{3\cdot\cos\left(x\right)^2\cdot\sin\left(x\right)-\sin\left(x\right)^3}{\sin\left(x\right)}+x$

```
(%i3)  ratsimp(%o2);
```
(%o3) $-\sin\left(x\right)^2+3\cdot\cos\left(x\right)^2+x$

```
(%i4)  trigexpand(sin(2*%alpha+3*%beta));
```
(%o4) $\cos\left(2\cdot\alpha\right)\cdot\sin\left(3\cdot\beta\right)+\sin\left(2\cdot\alpha\right)\cdot\cos\left(3\cdot\beta\right)$

```
(%i5)  trigexpand(sin(3*%alpha)+cos(4*%beta));
```
(%o5) $\sin\left(\beta\right)^4-6\cdot\cos\left(\beta\right)^2\cdot\sin\left(\beta\right)^2+\cos\left(\beta\right)^4-\sin\left(\alpha\right)^3+3\cdot\cos\left(\alpha\right)^2\cdot\sin\left(\alpha\right)$

Зауважимо, що в записі **Maxima** $\sin(\alpha)^4$ позначає $\sin^4\alpha$ тощо.

`trigreduce(expr)` – понижує степінь виразів, згортає всі добутки тригонометричних і гіперболічних функцій в комбінації відповідних функцій від сум. Функція спрацьовує не до кінця, тому повторний виклик може змінити вираз. При записі команди в форматі `trigreduce(expr,var)` – перетворення здійснюються відносно змінних `var`.

```
(%i6)  trigreduce(cos(x)^4+cos(x)^3+cos(x)^2+cos(x)+1);
```
(%o6) $\dfrac{\cos\left(4\cdot x\right)+4\cdot\cos\left(2\cdot x\right)+3}{8}+\dfrac{\cos\left(3\cdot x\right)+3\cdot\cos\left(x\right)}{4}+\dfrac{\cos\left(2\cdot x\right)+1}{2}+\cos\left(x\right)+1$

```
(%i7)  trigreduce(-sin(x)^2+3*cos(x)^2+x);
```
(%o7) $\dfrac{\cos\left(2\cdot x\right)}{2}+3\cdot\left(\dfrac{\cos\left(2\cdot x\right)}{2}+\dfrac{1}{2}\right)+x-\dfrac{1}{2}$

`trigsimp(expr)` – спрощує тригонометричні та гіперболічні вирази, застосовуючи до них тотожності $\sin^2 x+\cos^2 x=1$ та $\mathrm{ch}^2x-\mathrm{sh}^2x=1$.

```
(%i8)  trigsimp(sin(x)^2+3*cos(x)^2);
```

$$(\%o8) \quad 2\cdot\cos(x)^2+1$$

```
(%i9)  trigreduce(cos(x)^4+sin(x)^4);
```

$$(\%o9) \quad \frac{\cos(4\cdot x)+4\cdot\cos(2\cdot x)+3}{8}+\frac{\cos(4\cdot x)-4\cdot\cos(2\cdot x)+3}{8}$$

```
(%i10) trigsimp(%o9);
```

$$(\%o10) \quad \frac{\cos(4\cdot x)+3}{4}$$

Для гіперболічних функцій:

```
(%i11) trigsimp(sinh(x)^2+3*cosh(x)^2);
```

$$(\%o11) \quad 4\cdot\cosh(x)^2-1$$

```
(%i12) trigsimp(sinh(x)^4-cosh(x)^4);
```

$$(\%o12) \quad 1-2\cdot\cosh(x)^2$$

`trigrat(expr)` – приводить заданий тригонометричний вираз `expr` до канонічної спрощеної квазілінійної форми, причому він розглядається як раціональний, що містить sin, cos, tan, аргументи яких є лінійними формами змінних і π/n (n – ціле). Завжди, коли можливо, заданий вираз лінеаризується.

```
(%i13) trigrat((1+sin(2*%beta)-cos(2*%beta))/sin(%beta));
```

$$(\%o13) \quad 2\cdot\sin(\beta)+2\cdot\cos(\beta)$$

```
(%i14) trigrat(sin(x)^3*(1+cot(x))+cos(x)^3*(1+tan(x)));
```

$$(\%o14) \quad \sin(x)+\cos(x)$$

2.4 Розв'язування рівнянь

Розв'язання алгебричних рівнянь та їх систем здійснюється за допомогою команди `solve`. Синтаксис наступний:

`solve(eqn=0,x)` – знаходить розв'язок рівняння відносно змінної `x`.

`solve([eqn1=0,eqn2=0,...eqnN=0],[x1,x2,...xN])` – знаходить розв'язок системи рівнянь відносно змінних `x1,x2,...xN`.

```
(%i1)  solve(x^2-2*x-8=0,x);
```

$$(\%o1) \quad [x=-2, x=4]$$

```
(%i2)  solve(x-a/x+b=0,x);
```

$$(\%o2) \quad [x=-\frac{\sqrt{b^2+4\cdot a}+b}{2}, x=\frac{\sqrt{b^2+4\cdot a}-b}{2}]$$

```
(%i3)  solve([x^2+y^2=10,x-y=4],[x,y]);
```

$$(\%o3) \quad [[x=1,y=-3],[x=3,y=-1]]$$

В останньому прикладі **Maxima** видала розв'язок у вигляді списку, оскільки є дві точки, що задовільняють системі.

Команда `solve` застосовується і до розв'язання тригонометричних рівнянь. При цьому виникає попередження, що видається лише перший додатній розв'язок, без періодичного продовження, яке пошукачу треба буде знайти самостійно.

```
(%i4)  solve(2*sin(x-%pi/4)=sqrt(3));
```

solve: using arc-trig functions to get a solution.
Some solutions will be lost.

$$(\%o4) \qquad [x = \frac{7 \cdot \pi}{12}]$$

```
(%i5)  solve(sin(x)^2+2*sin(x)+1=0,x);
```

solve: using arc-trig functions to get a solution.
Some solutions will be lost.

$$(\%o5) \qquad [x = -\frac{\pi}{2}]$$

Команда `solve` у **Maxima** шукає корені і на полі комплексних чисел.

```
(%i6)  solve(x^2+x+1=0,x);
```

$$(\%o6) \qquad [x = -\frac{\sqrt{3} \cdot i + 1}{2}, x = \frac{\sqrt{3} \cdot i - 1}{2}]$$

Якщо є потреба знайти корені в чисельному вигляді, в кінці необхідно додати команду `numer`.

```
(%i7)  solve(x^2+3*x+1=0,x),numer;
```

rat: replaced 2.23606797749979 by 16692641/7465176 = 2.236067977499794
rat: replaced 2.23606797749979 by 16692641/7465176 = 2.236067977499794

$$(\%o7) \qquad [x = -2.618033988749896, x = -0.3819660112501031]$$

`allroots(eqn=0,x)` обчислює всі дійсні і комплексні корені поліноміальних рівнянь.

```
(%i8)  allroots((1+2*x)^3=10*(1+x^5),x);
```

$$(\%o8) \qquad [x = 0.6057379117337299, x = 0.9658759245461883 \cdot i - 0.4382771080081766,$$

$$x = -0.9658759245461883 \cdot i - 0.4382771080081766, x = -1.021758604074856,$$

$$x = 1.292574908357479]$$

`realroots(eqn=0,num)` шукає всі дійсні корені (комплексні не беруться до уваги) поліноміального виразу з точністю до числа `num`.

```
(%i10) realroots(-1-x+x^5=0,6.0e-6);
```

$$(\%o10) \qquad [x = \frac{612003}{524288}]$$

Хоч `solve` і розв'язує широкий клас рівнянь, ця команда все ж пасує перед трансцендентними рівняннями, тобто такими, де присутні як поліноміальні вирази, так і експоненціальні чи тригонометричні. Для прикладу, рівняння $\sin x = x/2$ та $e^x = x^2$ дадуть такі результати:

```
(%i11) solve(sin(x)=x/2,x);
```

$$(\%o11) \qquad [x = 2 \cdot \sin(x)]$$

```
(%i12) solve(exp(x)=x^2,x);
```

(%o12) $[x = -e^{\frac{x}{2}}, x = e^{\frac{x}{2}}]$

На такий випадок існує дуже корисна команда, яка чисельно шукає корені в певних межах змінної, використовуючи процедуру Ньютона наближеного обчислення. Для використання цього методу потрібно, щоб функція, яка описує даний вираз (`eqn1-eqn2`), на краях проміжка `[a,b]` приймала протилежні за знаком значення. Якщо ця умова не буде дотримуватись, система видасть попередження. Щоб дізнатись, в яких межах приблизно знаходиться корінь, корисно спочатку побудувати графіки обох виразів `eqn1` та `eqn2` та подивитись на точку їх перетину (про побудову графіків докладніше у Розділі 4).

`find_root(eqn1=eqn2,var,a,b)` – шукає наближені корені рівняння `eqn1=eqn2` за змінною `var` на інтервалі `[a,b]`.

Накреслимо графіки виразів на різних сторонах наших рівнянь. Бачимо, що корінь

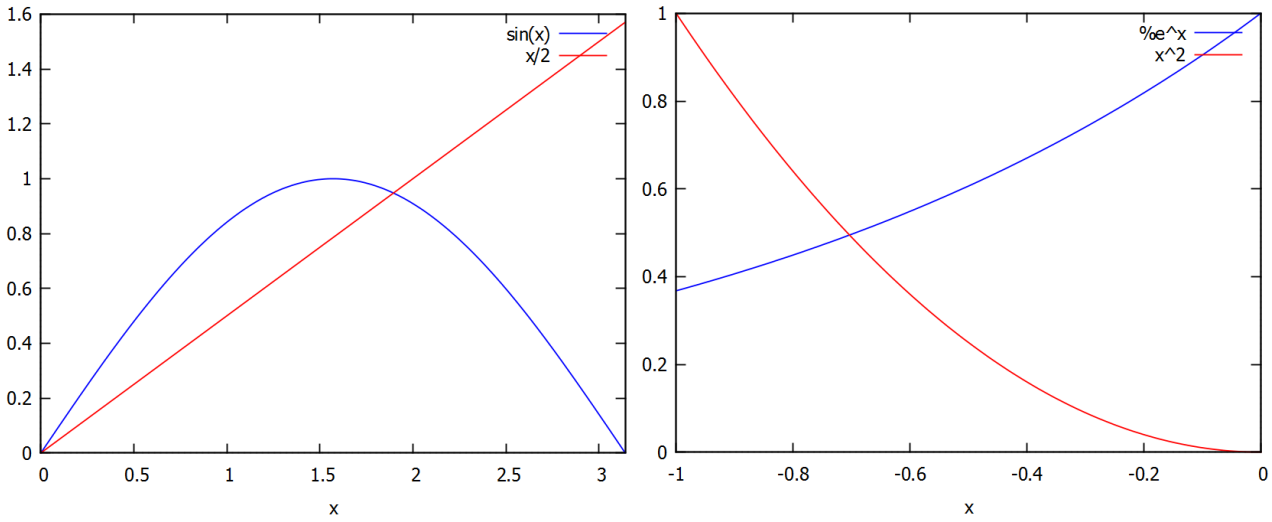

Рис. 2.1: Графіки лівих та правих частин трансцендентних рівнянь.

першого рівняння лежить у проміжку $[0, \pi]$, другого у проміжку $[-1, 0]$. Але якщо ми впишемо обидва ці проміжки в команду `find_root`, вона обидва рази дасть тривіальний корінь 0, тому будемо обмежувати проміжки з 0.1.

```
(%i13) find_root(sin(x)=x/2,x,0.1,%pi);
```

(%o13) 1.895494267033981

```
(%i14) find_root(exp(x)=x^2,x,-1,-0.1);
```

(%o14) − 0.7034674224983917

2.5 Нерівності

Для розв'язання нерівностей у **Maxima** існує популярний та досить добре розроблений пакет `solve_rat_ineq`, який працює з раціональними та дробово-раціональними виразами. Системи нерівностей він не розглядає, лише одиночні вирази. Пакет відразу включений до дистрибутиву **Maxima** і його потрібно завантажити командою `load`.

```
(%i1)  load(solve_rat_ineq);
```

```
(%i2)  solve_rat_ineq(x^2-3*x-4<0);
```

(%o2) $[[x > -1, x < 4]]$

```
(%i3)  solve_rat_ineq((x^2-1)/(x^2+x-6)>0);
```

(%o3) $[[x < -3], [x > -1, x < 1], [x > 2]]$

Для розв'язання нерівностей корисною також є команда `factor`, що розбиває вираз на множники, які вже можна аналізувати на зміну знаку функції при застосуванні методу інтервалів.

Якщо у нерівності не існує раціональних коренів, `solve_rat_ineq` шукає їх наближеними чисельними методами.

```
(%i4)  solve_rat_ineq(x^2>x^3+1);
```

(%o4) $[[x < -0.754877616175949]]$

```
(%i5)  solve_rat_ineq(x^5+x^3<x^2+x);
```

(%o5) $[[x < -0.5698403072569234], [x > 0, x < 1]]$

2.6 Завдання до Розділу 2

Числові задачі

2.1. Що більше: π^e чи e^π? Знайти на скільки процентів одне з цих чисел більше за інше з точністю до 19-го знаку після коми.

2.2. П'єр Ферма вважав, що всі числа виду $2^{2^n}+1$, де n натуральне, є простими. Знайти n, починаючи з якого це твердження перестає справджуватись.

2.3. Записати поліном $G(x) = 8x^5 - 45x^3 + 56x^2 - 7x + 11$. Вивести на екран: перший доданок, останній доданок, третій доданок, всі доданки крім першого.

2.4. Знайти найменше n, при якому число $10^{10}+n$ буде простим.

2.5. Тіло падає вниз з висоти 30 м, початкова швидкість 2 м/с. Записати формулу руху тіла, знайти час польоту.

2.6. Встановити, що вираз $6^{2n} + 3^{n+2} + 3^n$ кратний до 11 для перших перших п'яти натуральних чисел.

2.7. Знайти остачу від ділення $6x^4 - 7x^3 + 2x + 11$ на $3x^2 - 1$. Піднести цю остачу до 4-го степеня та розкрити дужки.

2.8. Знайти точне та наближене значення виразу $(\sqrt{11}+5)^2$ з точністю до 23 знаку після коми.

2.9. Обчислити довжину дуги кола радіуса 6 см та розхилом 23° з точністю до 45-ої цифри після коми.

2.10. Показати, що вираз $11^{n+2} + 12^{2n+1}$ ділиться на 133 без остачі для перших п'яти натуральних чисел.

Спростити вирази (алгебра)

2.11. $\dfrac{x^4-x^3-11x^2+9x+18}{x^4-3x^3-7x^2+27x-18} \div \dfrac{x^3-9x^2+26x-24}{x^3-8x^2+19x-12}$;

2.12. $\dfrac{3x^4-24x^3-3x^2+204x-252}{-x^5+10x^4-15x^3-70x^2+220x-168} \cdot \dfrac{2-x}{x+1}$;

2.13. $\dfrac{x^3+2x^2+4x+8}{x^5+5x^4-16x-80} \cdot \dfrac{2x^4+10x^3-16x-80}{x^2+2x+4}$;

2.14. $\dfrac{2x^4+10x^3-2x-10}{x^2+x+1} \cdot \dfrac{x^3+x^2+x+1}{x^5+5x^4-x-5}$;

2.15. $\dfrac{x^5+4x^4-81x-324}{3x^4+10x^3-81x-270} \cdot \dfrac{3x^3+19x^2+57x+90}{x^4+7x^3+21x^2+63x+108}$;

2.16. $\dfrac{x^3+3x^2-9x-27}{x^3-5x^2-15x-72} \cdot \dfrac{x^4-8x^3-27x+216}{49x^4-882x^2+3969}$;

2.17. $\dfrac{7x^4-126x^2+567}{x^5-8x^4-27x^2+216x} \cdot \dfrac{x^3-5x^2-15x-72}{x^3+3x^2-9x-27}$;

2.18. $\dfrac{4x^4+4x^3-48x^2-112x-64}{2x^3+4x^2-32x-64} \div \dfrac{x^2+3x+2}{x+4}$;

2.19. $\dfrac{x^4+x^3-7x^2-x+6}{5x^4+10x^3-100x^2-330x-225} \cdot \dfrac{x^3-2x^2-15x}{x^2-3x+2}$;

2.20. $\dfrac{x^3+2x^2+4x+8}{x^5+5x^4-16x-80} \cdot \dfrac{3x^4+10x^3-16x-80}{x^2+2x+4}$.

Спростити вирази (тригонометрія)

2.21. $\operatorname{ctg}^2 x - \dfrac{1+\operatorname{ctg} x+\operatorname{ctg}^2 x}{1+\operatorname{tg} x+\operatorname{tg}^2 x}$;

2.22. $\dfrac{1-2\sin 3x\cos(-3x)}{\cos 3x+\sin(-3x)}$;

2.23. $\cos^4 x-\cos^2 x+\sin^2 x$;

2.24. $\dfrac{\cos x}{1-\sin x}+\dfrac{1-\sin x}{\cos x}$;

2.25. $\dfrac{\sin^2 x\operatorname{tg}^2 x}{\operatorname{tg}^2 x+\cos^2 x-1}$;

2.26. $\operatorname{tg} x+\dfrac{\cos x}{1+\sin x}$;

2.27. $\dfrac{\cos^3 x-\sin^3 x}{1+\sin x\cos x}$;

2.28. $\sin^6 x+\cos^6 x+3\sin^2 x\cos^2 x$;

2.29. $\dfrac{\sin x+2\sin 2x+\sin 3x}{\cos x+2\cos 2x+\cos 3x}$;

2.30. $\dfrac{\sin x+\operatorname{tg} x}{1+\cos x}$.

Розв'язати рівняння (тригонометрія)

2.31. $\cos(2x+30^\circ)=-1$;

2.32. $\dfrac{\sqrt{3}}{\operatorname{tg}(4x+\pi/6)}=3$;

2.33. $\cos^2(x+\pi/6)=1/2$;

2.34. $\sin(\pi/4-x)=-\sqrt{3}/2$;

2.35. $\cos(\pi/9-4x)=1$;

2.36. $\operatorname{tg}(x/4-15^\circ)=\sqrt{3}$;

2.37. $2\cos(x/2+\pi/6)-\sqrt{3}=0$;

2.38. $\operatorname{ctg}(x/4-\pi/4)-1=0$;

2.39. $\sin(x/2+\pi/6)=1/2$;

2.40. $\cos(4x-\pi/6)=-\sqrt{3}/2$.

Розв'язати рівняння чисельно (скористатись відомостями з Розділу 4)

2.41. $\ln^2(x-1) = 3\cos 2x + 1;$

2.42. $\dfrac{10}{1+x^2} = 2\sin 2x + x;$

2.43. $\dfrac{10x-2}{3+x^2} = \sqrt[4]{x};$

2.44. $\dfrac{10}{1+x^2} = 2\cos 2x + x;$

2.45. $\sin x \cdot \sqrt{(81-x^2)} = 5\,\mathrm{arctg}\, x;$

2.46. $\sqrt{(25-x^2)} = \mathrm{arctg}\, 2x;$

2.47. $\sin^2 x \cdot \sqrt{(81-x^2)} = 5e^{-x^2};$

2.48. $\dfrac{x^2-9}{x^2+4} = \sqrt{(x^2+1)} \cdot e^{-x};$

2.49. $x + \sqrt{x} = e^{-x^2};$

2.50. $\sqrt{x} \cdot \cos x = \ln\left(x^2+1\right);$

Розділ 3

Робота з числовими масивами

3.1 Списки

Списки – базові будівельні блоки для Maxima та Lisp. Щоб задати список, достатньо записати його елементи через кому і обрамити квадратними дужками. Список може бути порожнім або складатись із одного елементу.

```
(%i1)  list1:[1,2,3,x,a+b];
```

(%o1) $[1, 2, 3, x, b + a]$

```
(%i2)  list2:[];
```

(%o2) []

Елементом списку може бути й інший список.

```
(%i3)  list3:[1,2,[3,4],[5,6,7]];
```

(%o3) $[1, 2, [3, 4], [5, 6, 7]]$

Посилання на елемент списку здійснюється за номером цього елементу у списку.

```
(%i4)  list3[3];
```

(%o4) $[3, 4]$

```
(%i5)  list3[4][2];
```

(%o5) 6

`length(list)` – повертає число елементів списку (при цьому самі елементу можуть бути достатньо складними конструкціями).

```
(%i6)  length(list3);
```

(%o6) 4

`copylist(list)` – дає копію списку `list`.

```
(%i7)  list4:copylist(list3);
```

(%o7) $[1, 2, [3, 4], [5, 6, 7]]$

Команда makelist створює список, кожний елемент якого генеруеться з деякого виразу. Можливі два варіанти виклику команди:

`makelist(expr,i,i1,iN)` – повертає список, `j`-й елемент якого рівний `ev(expr,i=j)`, при цьому індекс `j` міняється від `i1` до `iN`.

```
(%i8)  makelist(concat(x,i),i,1,6);
```

(%o8) $[x1, x2, x3, x4, x5, x6]$

```
(%i9)  makelist(a*i^2,i,1,5);
```

(%o9) $[a, 4 \cdot a, 9 \cdot a, 16 \cdot a, 25 \cdot a]$

`makelist(expr,x,list)` – повертає список, `j`-й елемент якого рівний `ev(expr,x=list[j])`, при цьому індекс `j` міняється від `1` до `length(list)`.

```
(%i10) list5:[1,2,3,4,5,6,7];
```

(%o10) $[1, 2, 3, 4, 5, 6, 7]$

```
(%i11) makelist(exp(i),i,list5);
```

(%o11) $[e, e^2, e^3, e^4, e^5, e^6, e^7]$

Аналогічні дії виконує команда `create_list(expr,x1,list1,...,xn,listN)` – вона будує список шляхом обчислення виразу `expr`, що залежить від `x1`, до кожного елементу списку `list1` (аналогічно `expr`, що залежить від `x2`, застосовується до `list2` і т.д.)

```
(%i12) create_list(x^i,i,[1,3,7]);
```

(%o12) $[x, x^3, x^7]$

`append(list1,list2,...listN)` – послідовно склеює списки.

```
(%i13) append([1],[2,3],[4,5,6,7]);
```

(%o13) $[1, 2, 3, 4, 5, 6, 7]$

`join(list1,list2)` – створює новий список з двох, компонуючи їх елементи почергово в порядку слідування. Новий список містить `list1_1`, потім `list2_1`, потім `list1_2`, `list2_2` і т.д. Якщо кількість елементів у списках не рівна, створений список обривається на останньому спільному номері елементу.

```
(%i14) join([1,2,3],[10,20,30,40]);
```

(%o14) $[1, 10, 2, 20, 3, 30]$

`reverse(list)` – міняє порядок елементів списку на зворотній.

```
(%i15) list6:[11,12,x-y,a+b,[15,16]];
```

(%o15) $[11, 12, x - y, b + a, [15, 16]]$

```
(%i16) reverse(list6);
```

(%o16) $[[15, 16], b + a, x - y, 12, 11]$

`member(list1,list2)` – повертає `«true»` якщо `list1` є елементом списку `list2`, або `«false»` в протилежному випадку.

```
(%i17) member(b,[a,b,[b,c]]);
```

(%o17) true

```
(%i18) member(b,[[a,b],[b,c]]);
```

(%o18) false

`first(list)` – виводить перший елемент списку, `last(list)` – останній елемент, `rest(list)` – видає залишок списку після видалення першого елементу, `rest(list,n)` – залишок списку після видалення перших n елементів.

```
(%i19) list7:[1,2,3,4,a,b];
```

(%o19) $[1, 2, 3, 4, a, b]$

```
(%i20) first(list7);
```

(%o20) 1

```
(%i21) last(list7);
```

(%o21) b

```
(%i22) rest(list7);
```

(%o22) $[2, 3, 4, a, b]$

```
(%i23) rest(list7,2);
```

(%o23) $[3, 4, a, b]$

`sum(list,i,i1,iN)` – сумує значення списку `list` при зміні индексу `i` від `i1` до `iN`.

```
(%i24) sum(i^2,i,1,5);
```

(%o24) 55

```
(%i25) sum(x+i*(i+1)/2,i,1,4);
```

(%o25) $4 \cdot x + 20$

`product(list,i,i1,iN)` – перемножує значення списку `list` при зміні индексу `i` від `i1` до `iN`.

```
(%i26) product(i^2,i,1,5);
```

(%o26) 14400

```
(%i27) product(x+i*(i+1)/2,i,1,4);
```

(%o27) $(x + 1) \cdot (x + 3) \cdot (x + 6) \cdot (x + 10)$

Існує корисна команда `map(F,expr1,...,exprN)`, яка дозволяє застосувати функцію (оператор, символ операції) `F` до виразів `expr1,...,exprN`. При її застосуванні до списків F діє на кожен елемент списку. Необхідно звернути увагу, що F – це саме ім'я функції (без вказування змінних, від яких вона залежить). Функція F може бути і заданою користувачем.

```
(%i28) map(ratsimp,x/(x^2+x)+(y^2+y)/y);
```

(%o28) $y + \dfrac{1}{x+1} + 1$

```
(%i29) map("=",[a,b],[-2,3]);
```
(%o29) $[a = -2, b = 3]$

```
(%i30) map(exp,[0,1,2,3,4,5]);
```
(%o30) $[1, e, e^2, e^3, e^4, e^5]$

```
(%i31) f(x):=x^3;
```
(%o21) $\mathrm{f}(x) := x^3$

```
(%i32) map(f,[1,2,3,4,5]);
```
(%o32) $[1, 8, 27, 64, 125]$

3.2 Масиви

Масиви в **Maxima** – сукупності однотипних об'єктів з індексами. Число індексів не повинно перевищувати п'яти. У **Maxima** існують і функції з індексами (функції масиву). Можливе створення та використання змінних з індексами до оголошення відповідного масиву, тоді такі змінні розглядаються як елементи масивів з невизначеними розмірами (так звані хеш-масиви). Розміри невизначених масивів зростають динамічно в міру надання значень елементам. Індекси масивів з невизначеними кордонами не обов'язково мають бути числами. Для підвищення ефективності обчислень рекомендується перетворювати масиви з невизначеними межами на звичайні масиви (для цього використовується команда `array`).

Створення масиву виконується функцією `array`. Синтаксис звернення до команди:

```
array(name,dim_1,...,dim_N)
array(name, type, dim_1,...,dim_N)
array([name_1,...,name_M],dim_1,...,dim_N)
```

```
(%i1)  array(a,1,1);
```
(%o1) a

```
(%i5)  a[0,0]:0; a[0,1]:1; a[1,0]:2; a[1,1]:3;
```
(%o2) 0

(%o3) 1

(%o4) 2

(%o5) 3

```
(%i6)  listarray(a);
```
(%o6) $[0, 1, 2, 3]$

Команда `listarray`, використана в прикладі, перетворює масив у список. Синтаксис виклику: `listarray (A)`. Аргумент `A` може бути визначеним або невизначеним масивом, функцією масиву або функцією з індексами. Порядок включення елементів масиву до списку - рядками.

`arrayinfo (A)` – виводить інформацію про масив `A`. Аргумент `A`, як і у випадку `listarray`, може бути визначеним або невизначеним масивом, функцією масиву чи функцією з індексами.

```
(%i8)  array(AR,2,3);
```

(%o8) AR

```
(%i9)  AR[2,3]:%pi;
```

(%o9) π

```
(%i10) AR[1,2]:%e;
```

(%o10) e

```
(%i11) arrayinfo(AR);
```

(%o11) $[declared, 2, [2, 3]]$

```
(%i12) ARR[foo]:(a+b)^2;
```

(%o12) $(b + a)^2$

```
(%i13) ARR[bar]:(a-b)^3;
```

(%o13) $(a - b)^3$

```
(%i14) arrayinfo(ARR);
```

(%o14) $[hashed, 1, [bar], [foo]]$

```
(%i15) listarray(ARR);
```

(%o15) $[(a - b)^3, (b + a)^2]$

Команда `make_array(type,dim_1,...,dim_N)` створює та повертає масив Lisp. Тип масиву може бути `any, flonum, fixnum, hashed, functional`. Індекс `i` може змінюватися в межах від `0` до `dim_i-1`. Переваги `make_array` в порівнянні з `array` – можливість динамічно керувати розподілом пам'яті для масивів. Присвоювання `y:make_array(...)` створює посилання на масив. Коли масив більше не потрібний, посилання знищується присвоюванням `y:false`, пам'ять звільняється потім очищувачем сміття.

```
(%i16) A1:make_array(fixnum,10);
```

(%o16) $Lisparray[10]$

```
(%i17) A1[1]:8;
```

(%o17) 8

```
(%i18) A2:make_array(any,10);
```

(%o18) $Lisparray[10]$

```
(%i19) arrayinfo(A2);
```

(%o19) $[declared, 1, [9]]$

Команда `fillarray(array1,array2)` дозволяє заповнювати масиви значеннями з іншого масиву чи списку. Заповнення провадиться по рядках.

```
(%i20) array(a,1,1);
```
(%o20) a

```
(%i21) fillarray(a,[1,2,3,4]);
```
(%o21) a

```
(%i22) a[1,1];
```
(%o22) 4

```
(%i23) a2:make_array(fixnum,8);
```
(%o23) $Lisparray[8]$

```
(%i24) fillarray(a2,[1,2,3,4,5]);
```
(%o24) $Lisparray[8]$

Як видно з розглянутих прикладів, довжина списку може і не співпадати з розмірністю масиву. Якщо вказано тип масиву, він повинен заповнюватись елементами того самого типу. Вилучення масивів з пам'яті здійснюється функцією `remarray(array)`.

3.3 Матриці

У **Maxima** визначені прямокутні матриці. Основний спосіб створення матриць – використання функції `matrix`. Синтаксис виклику:

`matrix(row1,...,rowN)`.

Кожен рядок – список виразів, усі рядки однакової довжини. На множині матриць визначені операції додавання, віднімання, множення та ділення. Ці операції виконуються поелементно, якщо операнди – дві матриці, скаляр та матриця або матриця та скаляр. Піднесення до степеня можливе, якщо один із операндів – скаляр. Перемноження матриць (загалом некомутативна операція) позначається символом «`.`». Операція множення матриці на себе може розглядатися як зведення в степінь. Піднесення до степеня «-1» означає отримати обернену матриці (якщо це можливо).

```
(%i1)  M:matrix([4,3],[-2,1]);
```
(%o1) $\begin{pmatrix} 4 & 3 \\ -2 & 1 \end{pmatrix}$

```
(%i2)  N:matrix([1,-2],[2,-3]);
```
(%o2) $\begin{pmatrix} 1 & -2 \\ 2 & -3 \end{pmatrix}$

Виконання арифметичних операцій з матрицями.

```
(%i3)  M+N;
```
(%o3) $\begin{pmatrix} 5 & 1 \\ 0 & -2 \end{pmatrix}$

```
(%i4)  M-N;
```

$$(\%o4) \quad \begin{pmatrix} 3 & 5 \\ -4 & 4 \end{pmatrix}$$

```
(%i5)  M*N;
```

$$(\%o5) \quad \begin{pmatrix} 4 & -6 \\ -4 & -3 \end{pmatrix}$$

```
(%i6)  M/N;
```

$$(\%o6) \quad \begin{pmatrix} 4 & -\frac{3}{2} \\ -1 & -\frac{1}{3} \end{pmatrix}$$

Необхідно звернути увагу – ці операції здійснюються поелементно. При спробі виконати ці арифметичні операції над матрицями різних розмірів видається помилка.

Приклад операцій з матрицями і скалярами:

```
(%i7)  M^2;
```

$$(\%o7) \quad \begin{pmatrix} 16 & 9 \\ 4 & 1 \end{pmatrix}$$

```
(%i8)  2^M;
```

$$(\%o8) \quad \begin{pmatrix} 16 & 8 \\ \frac{1}{4} & 2 \end{pmatrix}$$

Знову ж таки, при роботі з матрицями в **Maxima** необхідно бути уважними: вираз `M^2` зовсім не означає матричне помноження $M \cdot M$, а лише піднесення кожного елементу матриці до квадрата.

Для здійснення матричного множення використовується інший оператор – нижня крапка «`.`»:

```
(%i9)  M.N;
```

$$(\%o9) \quad \begin{pmatrix} 10 & -17 \\ 0 & 1 \end{pmatrix}$$

```
(%i10) N.M;
```

$$(\%o10) \quad \begin{pmatrix} 8 & 1 \\ 14 & 3 \end{pmatrix}$$

Очевидно, що для успішного перемноження матриці повинні бути узгоджені за розмірами (кількість стовпців першої рівна кількості рядків другої).

Обернену матрицю дозволяє знайти оператор «`^^`». Його не слід плутати з оператором «`^`», який видає матрицю з оберненими відповідними елементами.

```
(%i11) M^^(-1);
```

$$(\%o11) \quad \begin{pmatrix} \frac{1}{10} & -\frac{3}{10} \\ \frac{1}{5} & \frac{2}{5} \end{pmatrix}$$

```
(%i12) M^(-1);
```

$$(\%o12) \quad \begin{pmatrix} \frac{1}{4} & \frac{1}{3} \\ -\frac{1}{2} & 1 \end{pmatrix}$$

```
(%i13) M.(%o11);
```

$$(\%o13) \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

```
(%i14) M*(%o12);
```

$$(\%o14)\quad \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Команда `genmatrix` повертає матрицю заданої розмірності, що складена з елементів двоіндексного масиву. Синтаксис виклику:

```
genmatrix (a, i2, j2, i1, j1)
genmatrix (a, i2, j2, i1)
genmatrix (a, i2, j2)
```

Індекси `i1`, `j1` и `i2`, `j2` вказують на лівий та правий нижній елементи матриці в вихідному масиві.

```
(%i15) h[i,j]:=1/(i+j-1);
```

$$(\%o15)\quad h_{i,j} := \frac{1}{i+j-1}$$

```
(%i16) genmatrix(h,3,3);
```

$$(\%o16)\quad \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{pmatrix}$$

```
(%i17) array(A,fixnum,2,2);
```

$$(\%o17)\quad A$$

```
(%i18) A[1,1]:%e;
```

$$(\%o18)\quad e$$

```
(%i19) A[2,2]:%pi;
```

$$(\%o19)\quad \pi$$

```
(%i20) genmatrix(A,2,2);
```

$$(\%o20)\quad \begin{pmatrix} e & 0 \\ 0 & \pi \end{pmatrix}$$

`zeromatrix(M,N)` – повертає матрицю заданої размірності, складену з нулів.

```
(%i21) zeromatrix(2,2);
```

$$(\%o21)\quad \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

`ident(N)` – повертає одиничну матрицю заданоє розмірності N × N.

```
(%i22) ident(2);
```

$$(\%o22)\quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

`copymatrix(M)` – створює копію матриці `M`.

Функції `row` і `col` дозволяють отримати відповідно рядок і стовпець заданої матриці, отримуючи список. Синтаксис виклику:

`row(M,i)` – повертає `i`-й рядок;

`col(M,i)` – повертає `i`-й стовпець.

Функції `addrow` та `addcol` додають до матриці рядок або стовпець відповідно. Синтаксис виклику:

`addrow(M,list1,...,listN)` – додає рядки;
`addcol(M,list1,...,listN)` – додає стовпці.

```
(%i24) K:matrix([1,2],[3,4]);
```

$$(\%o24) \quad \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
(%i25) L:addrow(K,[10,20]);
```

$$(\%o25) \quad \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 10 & 20 \end{pmatrix}$$

```
(%i26) addcol(L,[x,y,z]);
```

$$(\%o26) \quad \begin{pmatrix} 1 & 2 & x \\ 3 & 4 & y \\ 10 & 20 & z \end{pmatrix}$$

Функція `submatrix` повертає нову матрицю, що складається з заданої підматриці. Синтаксис виклику:

`submatrix(i1,...,iN, M,j1,...,jM)`
`submatrix(i1,...,iN,M)`
`submatrix(M,j1,...,jM)`

Підматриця будується так: з матриці `M` видаляються рядки `i1,...,iN` і `j1,...,jM`. Приклад (використовуємо останній результат з попереднього прикладу, видаляємо перший рядок та другий стовпець):

```
(%i27) submatrix(1,(%o26),2);
```

$$(\%o27) \quad \begin{pmatrix} 3 & y \\ 10 & z \end{pmatrix}$$

Для заповнення матриці значеннями деякої функції використовується команда `matrixmap` (аналог `map, apply, fullmap`). Синтаксис виклику:

`matrixmap(f,M)` – повертає матрицю з елементами `i,j`, рівними `f(M[i,j])`.

```
(%i28) D:matrix([1,2],[3,4]);
```

$$(\%o28) \quad \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
(%i29) f(x):=x^3;
```

$$(\%o29) \quad \mathrm{f}\,(x) := x^3$$

```
(%i30) matrixmap(f,D);
```

$$(\%o30) \quad \begin{pmatrix} 1 & 8 \\ 27 & 64 \end{pmatrix}$$

Існує ще багато команд для роботи з матрицями, їх детальний розгляд здійснено в Розділі 5, при розв'язанні задач лінійної алгебри.

3.4 Завдання до Розділу 3

Протабулювати функцію для будь-яких 30 значень x, результати вивести у вигляді списку числових значень y

3.1. $f(x) = \dfrac{10}{x}$;

3.2. $f(x) = \cos x^3$;

3.3. $f(x) = e^{\frac{1}{x}}$;

3.4. $f(x) = x^2 \ln x$;

3.5. $f(x) = \dfrac{1}{\sin x}$;

3.6. $f(x) = \operatorname{ctg} x + \cos x$;

3.7. $f(x) = (10 - x^2)^2$;

3.8. $f(x) = \operatorname{tg} \dfrac{1}{\sqrt{x}}$;

3.9. $f(x) = x^2 \sin x \cos x$;

3.10. $f(x) = \dfrac{5}{x} + \dfrac{x}{5}$.

Зґенерувати матриці 4×4 за поданими формулами, до них додати рядки та стовпці будь-яких чисел, щоб утворились матриці 6×6

3.11. $h_{i,j} = \dfrac{1}{i^2 + j^2}$;

3.12. $h_{i,j} = i + \dfrac{1}{j}$;

3.13. $h_{i,j} = \cos i + \sin j$;

3.14. $h_{i,j} = e^{i-j}$;

3.15. $h_{i,j} = \operatorname{arctg} i + \operatorname{arcctg} j$;

3.16. $h_{i,j} = \dfrac{1}{i^2 + j^2}$;

3.17. $h_{i,j} = \dfrac{1}{2 - i + j}$;

3.18. $h_{i,j} = (i + j)^2$;

3.19. $h_{i,j} = \ln(i \cdot j)$;

3.20. $h_{i,j} = 0.18i^3 - 1.16j$.

Розділ 4

Графічні можливості Maxima

Для побудови графіків **Maxima** використовує **Gnuplot**, який викликається автоматично при заданні відповідних команд. Взагалі в **Maxima** можливі два методи для побудови графіків:

1. Задання стандартних графічних команд `plot2d, plot3d` викликає спливаюче вікно **Gnuplot**, що містить необхідний графік. Продовжувати роботу з **Maxima** в цей час неможливо, поки вікно не закриється. Вихідне вікно **Gnuplot** та відповідний графік можна масштабувати, використовуючи мишку. Також за допомогою мишки у 2d-графіках можна робити вимірювання, а 3d-графіки можливо обертати у всіх напрямках.

2. Добавляючи вираз `«wx»` до імен графічних команд (`plot2d, plot3d`), створюється малюнок формату .png, який вбудований прямо у вікно **wxMaxima** . Оскільки графік залишається видимим протягом усього сеансу Maxima, цей метод корисний для інтерактивної роботи. Права кнопка мишки на графіку дозволяє копіювати в буфер обміну або зберігати як файл. Тим не менш, з причини своєї низької роздільної здатності, подальше використання графіків, створених таким чином, не є доцільним.

4.1 Двовимірні графіки

Команда `plot` – це стандартний інтерфейс **Gnuplot** від **Maxima** , зручний у застосуванні, але не дуже гнучкий щодо зовнішнього вигляду графіки. При використанні цього інтерфейсу неможливо змінити зовнішній вигляд, формат виводу або вихідний графік на консолі **Gnuplot** після того, як графік було побудовано.

Основні команди графічного інтерфейсу `Plot`:

`plot2d(f(x),[x,a,b],opts)` – будує графік функції $f(x)$ на проміжку $x \in [a,b]$ за допомогою опцій `opts`.

`plot2d([discrete,[x_list],[y_list]],opts)` – будує точки вказаних значень `[x_list],[y_list]` двох окремих списків, що позначають відповідно x- та y-координати точок.

`plot2d([discrete,[list]],opts)` – інший спосіб побудови точок, коли вони задані у вигляді послідовного списку `[x1,y1],[x2,y2]...`

`plot2d([parametric,x(t),y(t),[t,t1,t2]],opts)` – будує параметричну криву з параметром $t \in [t_1, t_2]$.

Побудуємо простий графік функції з заданими межами по x; зміна по осі y обчислюється автоматично (Рис. 4.1).

```
(%i1)  plot2d(sin(x),[x,0,2*%pi]);
```

(%o1) $[C : /Users/orreg/maxout.gnuplot]$

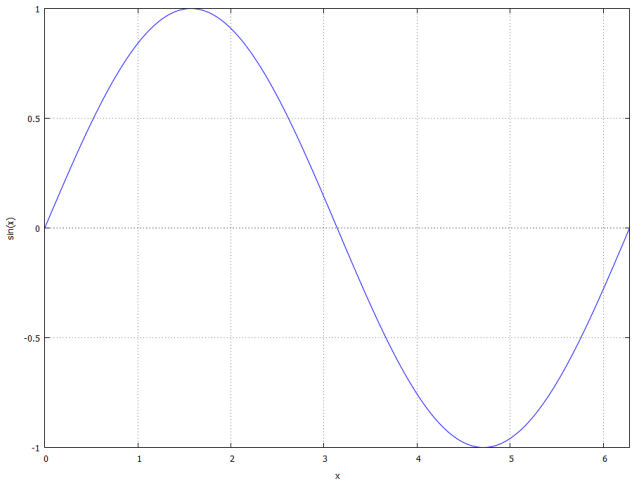

Рис. 4.1: Простий графік функції.

Команда `(wx)plot2d` малює криву або відрізки лінії між парами точок у двовимірній декартовій системі координат. Для цього є три можливості, які можуть бути об'єднані в одній діаграмі разом:

1. Функція $y = f(x)$; відображення діапазону вздовж осі x оголошується як `[x,a,b]`, оголошення y-діапазону необов'язкове.

2. Крива в параметричній формі $x(t)$, $y(t)$, що залежить від довільно обраного параметра t. Якщо цей параметр насправді має назву «t», оголошення діапазону можна опустити. У цьому випадку t отримує значення за замовчуванням, як це зазначено за допомогою `set_plot_option`.

3. Окремі точки, які можна з'єднати (застосовуючи відповідні параметри) за допомогою лінійних сегментів. Існує дві можливості для позначення точок: або в двох окремих списках, що містять значення x і y відповідно, або в одному вкладеному списку, який містить точки, де кожна точка представляє список, що містить його x-значення та y-значення.

Якщо необхідно побудувати два графіки на одній системі координат, із відповідних функцій необхідно сформувати список в квадратних дужках. Проміжок змінних буде спільним.

```
(%i2)  wxplot2d([sin(x),cos(x)],[x,0,2*%pi]);
```

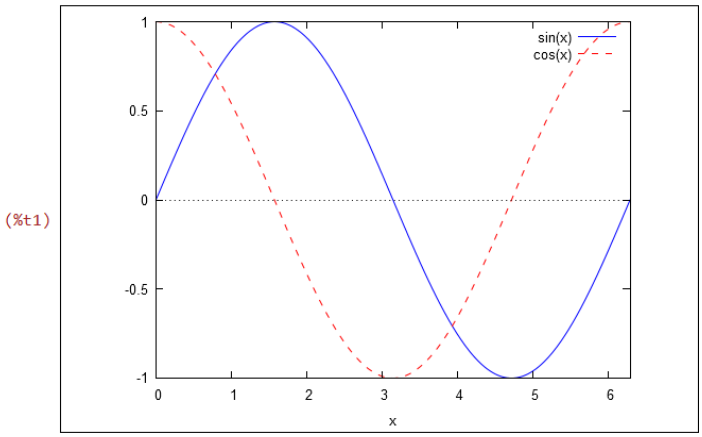

Рис. 4.2: Два графіки на одній системі координат.

Побудуємо графік множини точок, з'єднаних відрізками. Для цього необхідно задати

x- та y-значення списку (які є координатами точок):

```
(%i9)  x_list:[1,1,3,3,1];
```

(%o9) $[1, 1, 3, 3, 1]$

```
(%i10) y_list:[1,3,3,1,1];
```

(%o10) $[1, 3, 3, 1, 1]$

Зобразимо два точкових графіка в одній діаграмі з обома способами задання точок. Точки з'єднані відрізками лінії за замовчуванням.

```
(%i11) plot2d([[discrete,x_list,y_list],
               [discrete,[[3,1],[5,3],[3,5],[1,3],[3,1]]]],
               [x,0,6],[y,0,6]);
```

(%o11) $[C : /Users/orreg/maxout.gnuplot]$

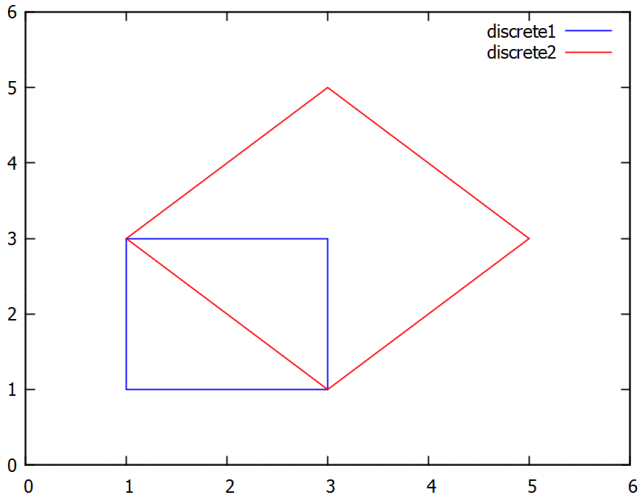

Рис. 4.3: Точкові графіки.

Розглянемо додаткові опції `opts` докладніше. Це необов'язкові параметри, що дозволяють адаптувати графіку до потрібних вимог стосовно кольорів, типів ліній, розмірів, міток, форматів виводу тощо. Опції – це списки, які в основному містять два елементи: перший елемент – це завжди ім'я параметра, другим елементом є пов'язані з цим параметром значення. Опції можуть виступати додатковими параметрами в кожній команді графіка; вони також можуть бути зазначні як значення за замовчуванням за допомогою команди `set_plot_option`. Команда `plot_options` показує всі значення за замовчуванням в опціях.

Якщо необхідно, щоб були побудовані просто точки, без з'єднання їх лініями, у команді `plot2d` потрібно додати опцію `[style,[points,m,n,k]]`, де `m` – її розмір, `n` – її колір, `k` – тип точки. Типів точок можна задати 6.

Команда `[style,[lines,m,n]]` керує виглядом ліній, тут `m` – її товщина, `n` – її колір. Кольорів можна задати 6 (0 = cyan, 1 = blue, 2 = red, 3 = green, 4 = majenta, 5 = black).

`[legend, "leg_1 "leg_2 ...]` – підписує лінії на графіку.

`[xlabel,"name_x"], [ylabel,"name_y"])` – підписує осі координат.

`[nticks,N]` – кількість точок, по яким будується графік.

`[axes,solid]` – виділяє осі виразнішими лініями.

`[same_xy]` – встановлює однаковий масштаб по осях.

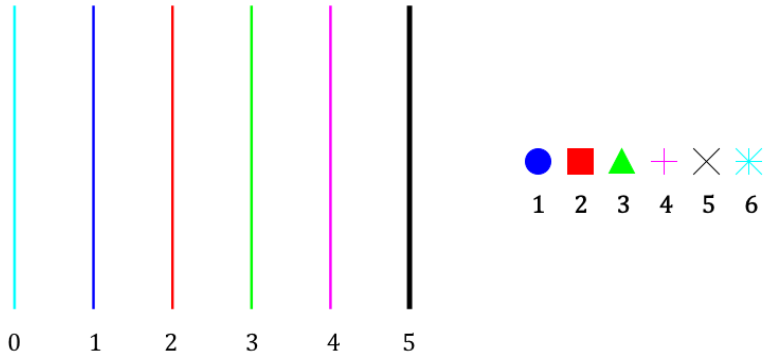

Рис. 4.4: Колір ліній та типи точок.

Значення `false` після коми відключає дану опцію, наприклад при заданні команди `[legend,false]` легенди кривих взагалі не будуть відображатись.

Нехай маємо наступну задачу: методом натурних досліджень проведено збір експериментальних даних, в яких деяка величина y залежить від змінної x. Отримані результати: $x = [1.389, 4.722, 5.000, 6.944, 8.611]$, $y = [30, 34, 38, 44, 46]$. Можна провести регресійний аналіз даних та встановити функціональну залежність:

$y = 25.713 + 2.379x$ – лінійна апроксимація;

$y = 27.463 + 1.443x + 0.0945x^2$ – квадратична апроксимація.

Необхідно зробити графіки трьох видів на одній системі координат. Код буде наступний:

```
(%i12) wxplot2d([25.713+2.379*x,27.463+1.443*x+0.0945*x^2,
       [discrete, [1.389,4.722,5.000,6.944,8.611], [30,34,38,44,46]]],
       [x,-3,13],[style,[lines,3,1],[lines,3,3],[points,4,2,14]],
       [box,false],[legend,"Лінійне наближення","Квадратичне наближення",
       "Вихідні дані"]);
```

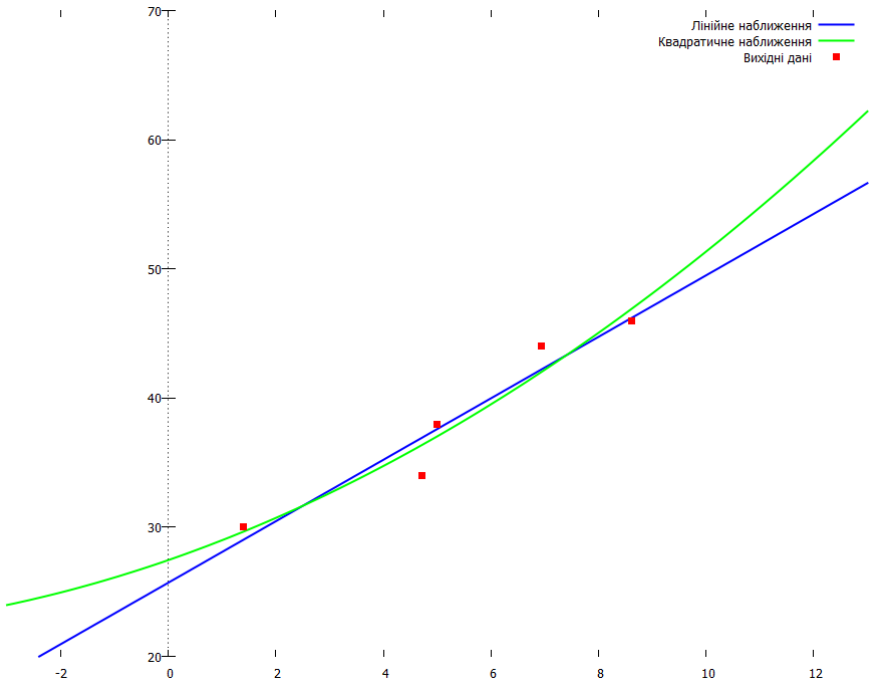

Рис. 4.5: Графіки результатів регресійного аналізу.

4.2 Параметричне та полярне задання функцій

При параметричному представленні функція задається у вигляді $x = \varphi(t)$; $y = \psi(t)$, $t \in [t_1, t_2]$. Команда **Maxima** побудови графіка в такому разі:

`plot2d([parametric,%phi(t),%psi(t),[t,t1,t2],[nticks,N]])`.

Додаткову опцію `nticks` в цій команді краще дописувати і приймати рівною 80-100, це збільшить гладкість кривої.

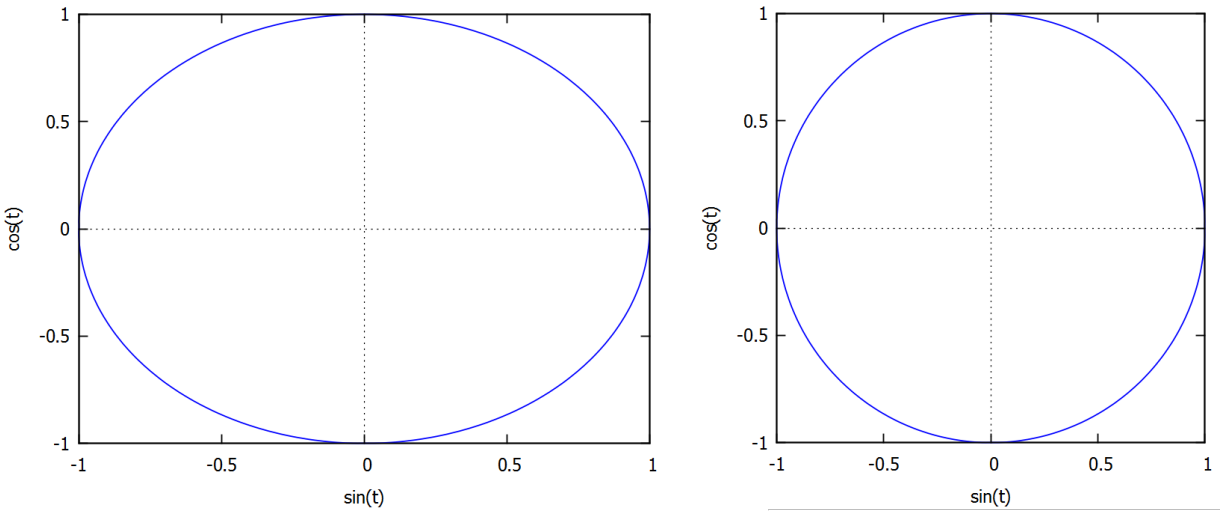

Рис. 4.6: «Коло» та коло з використанням `[same_xy]`.

Намалюємо, наприклад, коло введенням команди

```
(%i1)  plot2d([parametric,sin(t),cos(t),[t,0,2*%pi],[nticks,100]]);
(%o1)      [C : /Users/orreg/maxout.gnuplot]
```

Виведений на екран графік виявиться зовсім не колом, а еліпсом (Рис. 4.6). Причина в тому, що **Maxima** самовільно змінює масштаб осей графіка, для кращої читаності. Якщо потрібне са́ме намальоване коло, необхідно додати опцію `[same_xy]`.

```
(%i2)  plot2d([parametric,sin(t),cos(t),[t,0,2*%pi],[nticks,100]],
              [same_xy]);
(%o2)      [C : /Users/orreg/maxout.gnuplot]
```

Якщо обрамлення зовнішніми межами не потрібне, в опціях потрібно додати `[box,false]`. Щоб отриманий графік не дотикався меж бокса, необхідно дописати явно межі змін x та y, наприклад, `[x,-1.5,1.5]`, `[y,-1.5,1.5]`.

Ще приклади: фігура Лісажу та кардіоїда Паскаля (Рис. 4.7).

```
(%i3)  wxplot2d([parametric,sin(3*t),sin(4*t),
       [t,0,2*%pi]],[nticks,100],[same_xy])$
(%i4)  wxplot2d([parametric,2*cos(t)*(1+cos(t)),2*sin(t)*(1+cos(t)),
       [t,0,2*%pi]],[nticks,100],[same_xy])$
```

Для побудови графику в полярних координатах потрібно задати залежність полярного радіуса від полярного кута. Нехай $\rho = \rho(\theta)$ $(a \le \theta \le b)$ – така залежність. Тогда графік цієї функції в полярних координатах можна побудувати, задавши для функції `plot2d` опцію `[gnuplot_preamble,set polar;set zeroaxis]`. Ця опція буде діяти лише за умови, що вибраний формат графіку **Gnuplot**.

Є, однак, спосіб, який спрощує побудову графіка в полярних координатах, і зводить його до звичної форми написання. Це запровадження нової команди `plot_polar`:

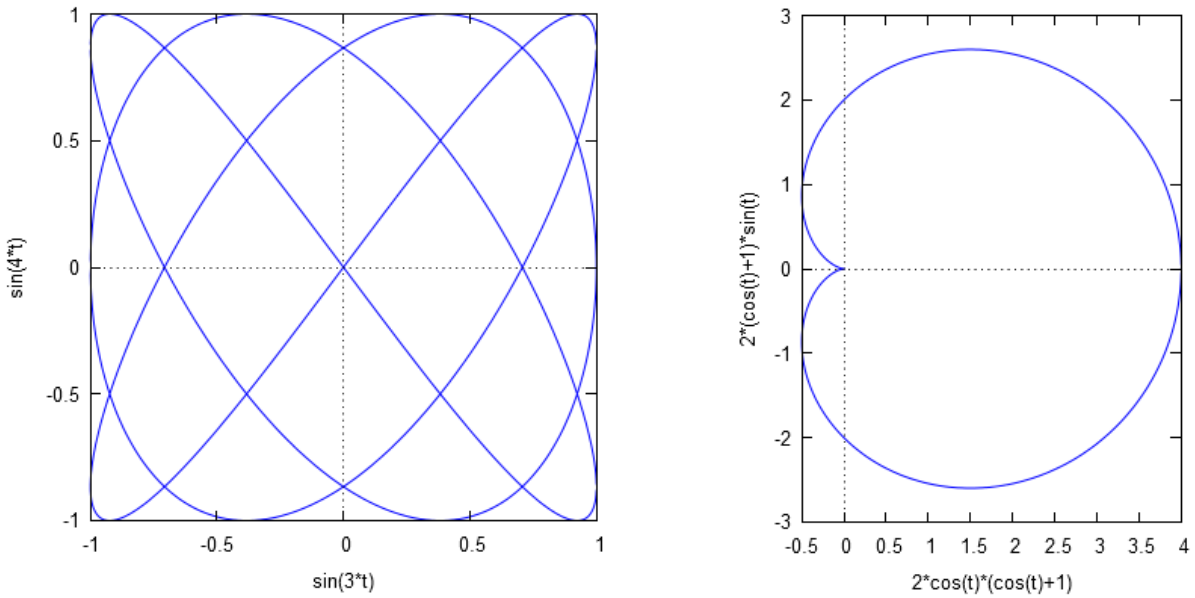

Рис. 4.7: Фігура Лісажу, кардіоїда.

`plot_polar(expr,range):=block([theta_var: range[1]],plot2d(['parametric,cos(theta_var)*expr,sin(theta_var)*expr,range,[nticks,100]]));`

Один раз ввівши таку функцію в **Maxima**, протягом сесії можна користуватись таким синтаксисом: `plot_polar(f(t),[t,t1,t2])`.

Накреслимо для прикладу графіки функцій $\rho = \sin\theta + 2\cos 2\theta$; $\rho = 3\theta^2$ (Рис. 4.8).

```
(%i5)  plot_polar(sin(t)+2*cos(2*t),[t,0,%pi]);
```

(%o5) $[C : /Users/orreg/maxout.gnuplot]$

```
(%i6)  plot_polar(3*t^2,[t,0,5*%pi]);
```

(%o6) $[C : /Users/orreg/maxout.gnuplot]$

Знову ж таки, товщину та стиль лінії можна регулювати, використовуючи опцію `style` (наприклад, `[style,[lines,3,3]]` видає товщину 3 і зелений колір).

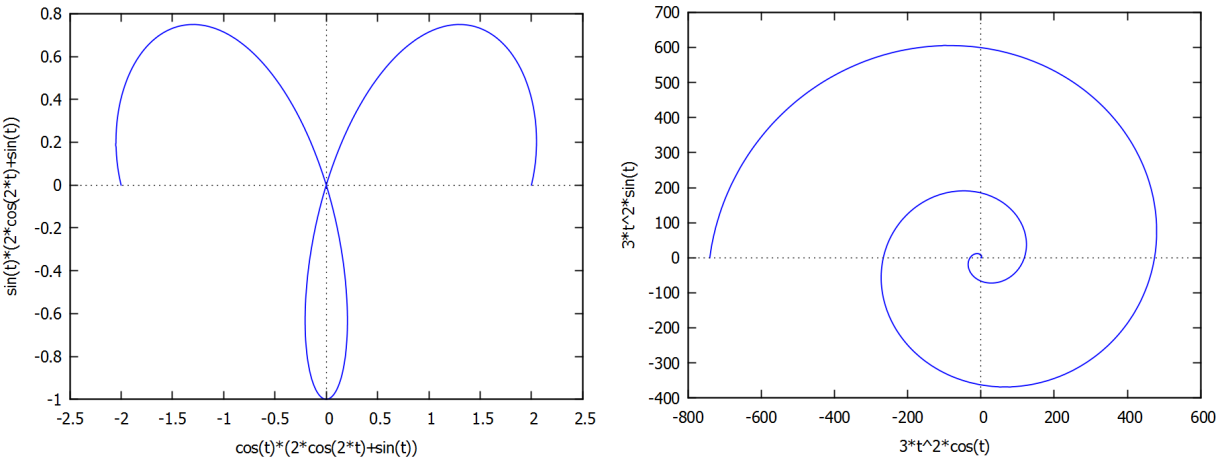

Рис. 4.8: Полярні координати.

4.3 Тривимірні графіки

3d-об'єкти у **Maxima** можуть бути утворені двома різними способами:

1. Задана функція $z = f(x, y)$; діапазони для x-значень та y-значень повинні бути оголошені, z-діапазон при цьому обчислюється автоматично.

2. Задана функція у параметричній формі $f_x(u, v)$, $f_y(u, v)$, $f_z(u, v)$, що залежать від двох (вибір позначень довільний) параметрів u і v. Необхідно оголосити діапазони параметрів u і v, діапазони координат f_x, f_y і f_z не оголошуються.

Синтаксис команди:

`(wx)plot3d(f(x,y),[x,a,b],[y,c,d])` – пряме задання функції $z = f(x, y)$;

`(wx)plot3d([fx,fy,fz],[u,a,b],[v,c,d])` – параметричне задання функції.

Накреслимо дві наступні функції для демонстрації обох випадків (Рис. 4.9):

$$z(x, y) = \frac{1}{1 + x^2 + y^2}; \qquad \begin{cases} f_x(u, v) = \cos u; \\ f_y(u, v) = v; \\ f_z(u, v) = u + \sin v. \end{cases}$$

```
(%i1)  plot3d(1/(1+x^2+y^2),[x,-2,2],[y,-2,2]);
```

(%o1) $[C : /Users/orreg/maxout.gnuplot]$

```
(%i2)  plot3d([cos(u),v,u+sin(v)],[u,0,2*%pi],[v,0,2*%pi],[same_xy])$
```

(%o2) $[C : /Users/orreg/maxout.gnuplot]$

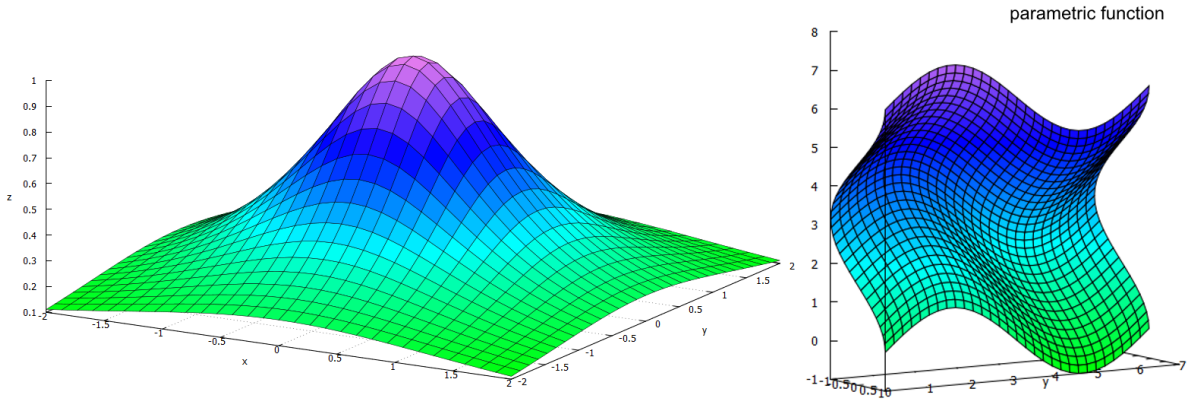

Рис. 4.9: Тривимірні графіки.

Якщо є потреба мати вбудовані в тіло документу **wxMaxima** графіки, необхідно вводити ж самі команди, але у вигляді `wxplot3d`. Але при цьому втрачається дуже корисна властивість побудованих у **Gnuplot** графіків: їх можна обертати мишкою, міняючи огляд та кут зору.

Опція `[palette,false]` відключає кольорове забарвлення поверхні, залишаючи лінії характеристик.

Опція `[same_xyz]` – робить масштаби всіх осей однаковими.

Команда `contour_plot(f(x,y),[x,a,b],[y,c,d])` – зображає лінії рівнів функції двох змінних, їх число та колір можуть бути зазначені використавши опцію `gnuplot_preamble`.

```
(%i3)  plot3d(cos(x)*sin(y),[x,0,2*%pi],[y,0,2*%pi],[palette,false]);
```

(%o3) $[C : /Users/orreg/maxout.gnuplot]$

```
(%i4)  contour_plot(cos(x)*sin(y),[x,0,2*%pi],[y,0,2*%pi]);
```

(%o4) $[C : /Users/orreg/maxout.gnuplot]$

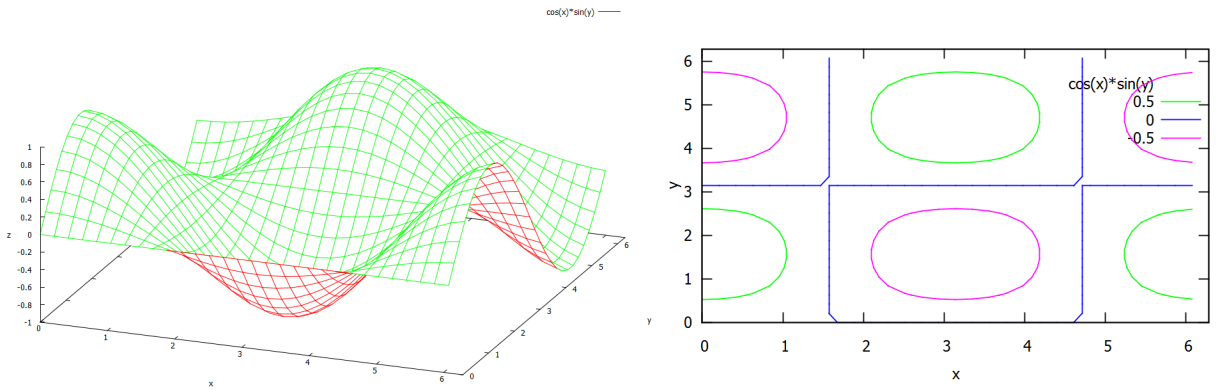

Рис. 4.10: Функція $z = \sin x \cdot \cos y$ та її лінії рівнів.

Опції `[nticks,N]` та `[adapt_depth,N]` задають число та максимальну множину початкових точок для побудови кривої. Вони потрібні, щоб зображена поверхня була достатньо «гладкою».

Опція `[grid,Nx,Ny]` – вказує, скількома точками полігону буде апроксимуватись крива характеристик. Ця команда з «плавної» кривої робить відрізки прямих. Наприклад, побудована вище поверхня $z(x,y) = \frac{1}{1+x^2+y^2}$ з додаванням `[grid,5,5]` буде виглядати так, як зображено на Рис 4.11. З допомогою `grid` можна зобразити просторову криву (ґвинтову лінію), без її застосування ця крива була б занадто «ламаною».

```
(%i5)  plot3d(1/(1+x^2+y^2),[x,-1,1],[y,-1,1],[grid,5,5]);
```

(%o5) $[C : /Users/orreg/maxout.gnuplot]$

```
(%i6)  plot3d([sin(u),cos(u),3*u],[u,0,6*%pi],[v,0,1],[grid,100,100]);
```

(%o6) $[C : /Users/orreg/maxout.gnuplot]$

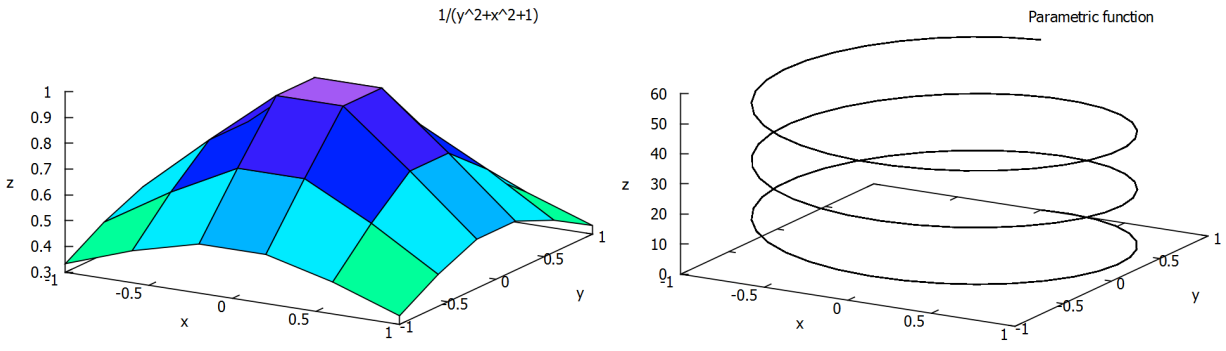

Рис. 4.11: Застосування опції `grid`.

Ще деякі приклади просторових поверхонь – односторонній лист Мебіуса та одинична сфера (уважний читач одразу впізнає в командах вирази для x, y, z у сферичній системі координат).

```
(%i7)  plot3d([cos(u)*(3+v*cos(u/2)),sin(u)*(3+v*cos(u/2)),v*sin(u/2)],
              [u,-%pi,%pi],[v,-1,1]);
       plot3d([cos(u)*sin(v),sin(u)*sin(v),cos(v)],[u,-%pi/2,%pi/2],
              [v,0,2*%pi],[same_xy],[palette,false],[legend, "Sphere"]);
```

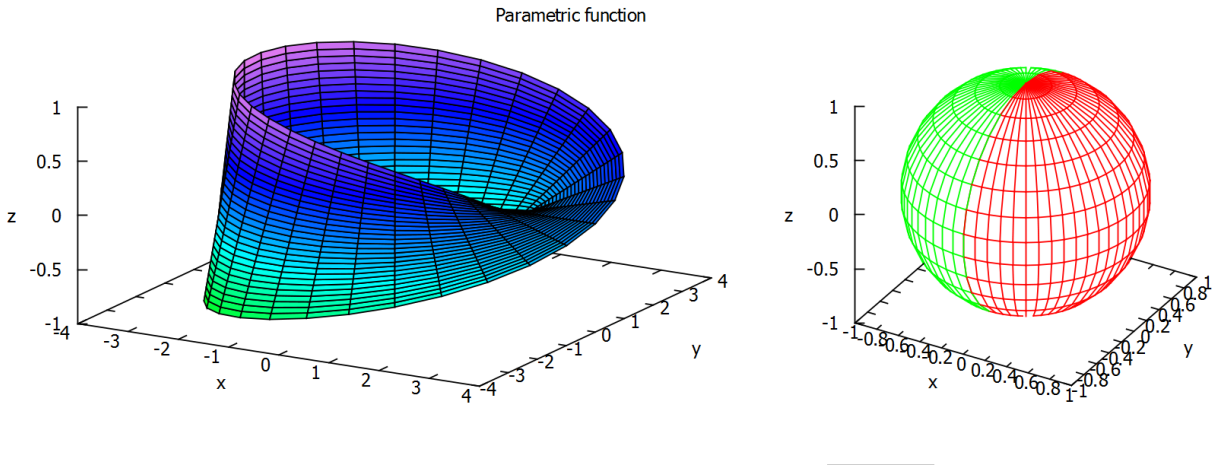

Рис. 4.12: Лист Мебіуса та одинична сфера.

4.4 Функції, задані неявно

Для побудови графіків функцій, заданих неявним чином, використовується команда plot2d, у якій необхідно вказати вираз (або кілька виразів), та обов'язково інтервал змінних як по x, так і по y. Синтаксис в такому випадку:

`plot2d([eqn1,eqn2,...],[x,a,b],[y,c,d],opts).`

► Накреслити графіки неявних функцій: $xy^2 = x - y^3$, $xy = x^2 - y$ (Рис. 4.13(а)).

```
(%i1)  plot2d([x*y^2=x-y^3,x*y=x^2-y],[x,-6,6],[y,-4,4],
              [legend,"expr1","expr2"]);
```

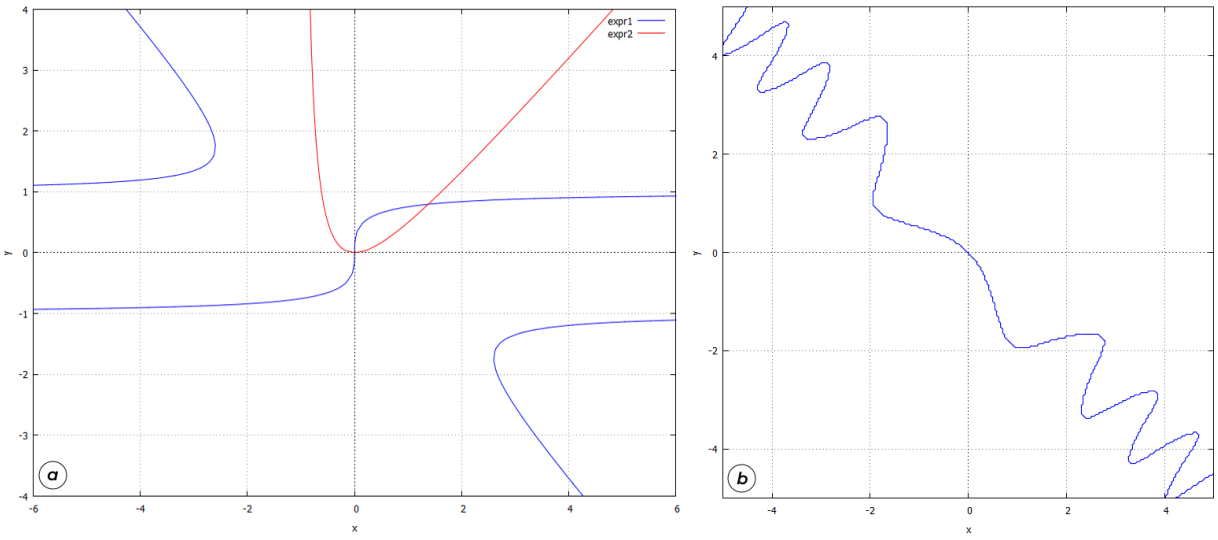

Рис. 4.13: Графіки неявно заданих функцій через `plot` (а) та `implicit_plot` (б).

Поряд з цим можна використовувати додатковий пакет `implicit_plot`, якого треба завантажувати окремо. Синтаксис команди:

`implicit_plot([eqn1(x,y),eqn2(x,y),...],[x,a,b],[y,c,d])`,

тут `eqni(x,y)` – одне співвідношення між x та y або список співвідношень, `[x,a,b]` – задання зміни по x, `[y,c,d]` – задання зміни по y.

▶ Зобразити графік кривої $x + y = \sin xy$.

```
(%i9)  implicit_plot(x+y=sin(x*y),[x,-5,5],[y,-5,5],[same_xy]);
(%o9)  done
```

Графік на Рис. 4.13(б). Видно, що якість та чіткість малюнка з командою `plot` є вищою, оскільки алгоритм побудови графіка є іншим.

4.5 Завдання до Розділу 4

Двовимірні графіки

4.1. Накреслити графіки функцій: $y = x$, $y = -x$, $y = x \sin x$, $x \in [-5\pi, 5\pi]$. Виділити осі, підписати криві «Меркурій», «Венера», «Марс».

4.2. Накреслити графіки функцій: $y = e^{-x^2}$, $y = -\sqrt{x}$, $y = \ln x$, $x \in [-6, 6]$, $y \in [-4, 4]$. Виділити осі, назвати їх «відстань», «час». Підписати криві «Гаусс», «Корінь», «Логарифм».

4.3. Накреслити графік функції: $y = \cos(t^3)$, $t \in [-3, 3]$. Графік не повинен дотикатись до меж бокса. Знайти будь-які три точки, що належать цій кривій, зобразити їх на графіку зірочками чорного кольору.

4.4. Накреслити графік неявної функції: $\operatorname{sh} x = x^2 + y^2$, $x \in [-1, 5], y \in [-5, 5]$. Виділити осі, назвати їх «амплітуда», «координата».

4.5. Накреслити дискретні графіки: точки з'єднані відрізками (0,0), (3,2), (6,0), (4,3), (6,6), (3,4), (0,6), (2,3), (0,0); окремі точки (трикутники) (3,1), (5,3), (3,5), (1,3). Масштаб осей зробити однаковим, підписати набори точок: «Зірка», «Ромб».

4.6. Накреслити дискретні графіки: точки з'єднані відрізками (1,1), (4,5), (9,5), (9,1), (1,1); окремі точки (хрестики) (4,1), (6.5,3), (9,5); $x \in [-2, 8], y \in [-2, 8]$. Масштаб осей зробити однаковим, підписати набори точок: «Трапеція», «Точки».

4.7. Накреслити графік параметричної функції: $x = 20(\cos t + 0.2 \cos 5t)$; $y = 20(\sin t - 0.2 \sin 5t)$, $t \in [0, 2\pi]$. Товщину ліній зробити 4, колір червоний, масштаб осей однаковий. Позначити на графіку такі точки: при $t = 1.5$, $t = 3.5$, $t = 5.5$.

4.8. Накреслити графік полярної функції: $\rho = (\theta + 1)^2$, $\theta \in [-2\pi, 2\pi]$. Товщину ліній зробити 3, колір чорний. Позначити на графіку дві точки: при $\theta = -5$ та $\theta = 5$.

4.9. Накреслити графік параметричної функції: $x = \sin 5t \cos t$; $y = \sin 5t \sin t$, $t \in [0, 2\pi]$. Товщину ліній зробити 3, колір чорний, масштаб осей однаковий. Позначити на графіку такі точки: при $t = 0.5\pi$, $t = 1.3\pi$, $t = 1.7\pi$.

4.10. Накреслити графік неявної функції: $x^4 - 96x^2 = y^4 - 100y^2$, $x \in [-22, 22], y \in [-22, 22]$. Виділити осі, назвати їх «час», «відстань».

Тривимірні графіки

4.11. Побудувати поверхню: $\sin(x^2+y^2)$; $x \in [0,3], y \in [0,3]$. Підписати осі «Північ», «Південь», «Захід».

4.12. Побудувати тор: $f_x = (3+\cos v)\cos u$; $f_y = (3+\cos v)\sin u$; $f_z = \sin v$; $u \in [0,2\pi], v \in [-\pi,\pi]$. Змінити межі так, щоб він став зрізаним.

4.13. Побудувати гіперболоїд: $f_x = (5+\operatorname{ch} v)\cos u$; $f_y = (5+\operatorname{ch} v)\sin u$; $f_z = \operatorname{sh} v$; $u \in [0,2\pi], v \in [-\pi,\pi]$. Змінити межі так, щоб він став зрізаним.

4.14. Побудувати гелікоїд: $f_x = u\sin v$; $f_y = u\cos v$; $f_z = v/3$; $u \in [-1,1], v \in [0,10]$.

4.15. Побудувати мушлю равлика: $f_x = 1.16^v \cdot \cos v(1+\cos u)$; $f_y = -1.16^v \cdot \sin v(1+\cos u)$; $f_z = -2{\cdot}1.16^v{\cdot}(1{+}\sin u)$; $u \in [0,2\pi], v \in [-10,6]$. Встановити опцію згладжування [100,100].

4.16. Побудувати конус двома способами: параметричне задання $f_x = \sin u\cos v$; $f_y = \sin u\sin v$; $f_z = |\sin u|$; $u \in [0,5], v \in [0,5]$; пряме задання $z = 3\sqrt{x^2/4+y^2/4}$; $x \in [-5,5], y \in [-5,5]$. Змінити межі так, щоб він став зрізаним.

4.17. Побудувати поверхню: $z = 20e^{-x^2-y^2} - 10$; $x \in [0,2]; y \in [-3,3]$. Зобразити контурні лінії цієї поверхні.

4.18. Побудувати просторову криву «торнадо»: $f_x = u\cos u$; $f_y = u\sin u$; $f_z = u$; $u \in [0,8\pi], v \in [0,3]$. Встановити опцію згладжування [100,100].

4.19. Побудувати циліндр: $f_x = \sin u$; $f_y = \cos u$; $f_z = v$; $u \in [0,2\pi], v \in [0,2]$. Масштаб осей зробити однаковим, розфарбування вимкнути.

4.20. Побудувати поверхню: $f_x = \cos u\cos v$; $f_y = \cos u\sin v$; $f_z = \operatorname{sh} v$; $u \in [0,\pi], v \in [0,\pi]$.

Розділ 5

Задачі лінійної алгебри

Пакет **Maxima** містить велике число функцій для розв'язання різноманітних задач лінійної алгебри. Основні відомості про матриці вже були розглянуті раніше, в Розділі 4. Зараз зупинимось докладніше на застосуванні матриць та операціях з ними.

5.1 Основні операції з матрицями

Інтерфейс **wxMaxima** достатньо зручний, він не потребує вручну вводити дані для виклику команди `matrix` (в численних квадратних дужках можна легко загубитись). Потрібно заповнити допоміжні форми, використовуючи пункти меню «Алгебра – Створити матрицю» або «Створити матрицю за виразом». При цьому з'явиться вікно з комірками для введення чисел.

Нагадаємо основи. В **Maxima** на матрицях означені звичні операції домноження на число, додавання та матричного множення. Останнє реалізується за допомогою бінарної операції «.» (нижня крапка). Розмірності матриць-співмножників мають бути узгоджені.

```
(%i1)  M:matrix([1,2,3],[4,5,6]);
```

$$(\%o1)\quad \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

```
(%i2)  N:matrix([1,-2],[2,-3],[3,-4]);
```

$$(\%o2)\quad \begin{pmatrix} 1 & -2 \\ 2 & -3 \\ 3 & -4 \end{pmatrix}$$

Якщо матриця – лівий співмножник, то правим співмножником може бути вектор-стовпець, вектор-рядок або список. **Maxima** дозволяє також підносити матриці до степеня, але фактично ця операція застосовується до кожного елемента.

```
(%i3)  M.N;
```

$$(\%o3)\quad \begin{pmatrix} 14 & -20 \\ 32 & -47 \end{pmatrix}$$

`transpose(M)` – транспонує матрицю.
`invert(M)` – шукає обернену матрицю.
`determinant(M)` – знаходить детермінант матриці.

```
(%i4)  M2:transpose(M);
```

$$(\%o4)\quad \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
(%i5)  K:(%o3);
```

$$(\%o5)\quad \begin{pmatrix} 14 & -20 \\ 32 & -47 \end{pmatrix}$$

```
(%i6)  invK:invert(K);
```

$$(\%o6)\quad \begin{pmatrix} \frac{47}{18} & -\frac{10}{9} \\ \frac{16}{9} & -\frac{7}{9} \end{pmatrix}$$

```
(%i7)  K.invK;
```

$$(\%o7)\quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

```
(%i8)  determinant(K);
```

$$(\%o8)\quad -18$$

```
(%i9)  determinant(invK);
```

$$(\%o9)\quad -\frac{1}{18}$$

`charpoly(M,%lambda)` – обчислює характеристичний поліном матриці (M – матриця, λ – змінна, відносно якої будується поліном). Корені характеристичного полінома є власними числами матриці.

```
(%i10) charpoly(K,%lambda);
```

$$(\%o10)\quad (-\lambda - 47)\cdot(14-\lambda)+640$$

```
(%i11) ratsimp(%o10);
```

$$(\%o11)\quad \lambda^2+33\cdot\lambda-18$$

```
(%i12) solve((%o11)=0,%lambda);
```

$$(\%o12)\quad [\lambda=-\frac{3\cdot\sqrt{129}+33}{2},\lambda=\frac{3\cdot\sqrt{129}-33}{2}]$$

Однак для обчислення власних чисел та власних векторів матриці зазвичай використовують спеціальні команди: `eigenvalues` та `eigenvectors`.

`eigenvalues(M)` – обчислює власні значення матриці аналітично, якщо це можливо. Для знаходження коренів характеристичного полінома використовується функція `solve`. Результат повертається у вигляді списка, що містить два підсписки. Перший містить власні значення, а другий – їх кратності.

```
(%i13) eigenvalues(K);
```

$$(\%o13)\quad [[-\frac{3\cdot\sqrt{129}+33}{2},\frac{3\cdot\sqrt{129}-33}{2}],[1,1]]$$

`eigenvectors(M)` – аналітично обчислює власні значення та власні вектори матриці, якщо це можливо. Вона повертає список, перший елемент якого – список власних чисел (аналогічно до `eigenvalues`), а далі йдуть власні вектори, кожен з яких представлений як список своїх проекцій.

```
(%i14) eigenvectors(K);
```

$$(\%o14)\quad [[[-\frac{3\cdot\sqrt{129}+33}{2},\frac{3\cdot\sqrt{129}-33}{2}],[1,1]],[[[1,\frac{3\cdot\sqrt{129}+61}{40}]],[[1,-\frac{3\cdot\sqrt{129}-61}{40}]]]]$$

Функція `uniteigenvectors` відрізняється від команди `eigenvectors` тим, що повертає нормовані на одиницю власні вектори.

Maxima включає в себе спеціальну функцію для обчислення ортонормованого набору векторів із заданого. Використовується стандартний алгоритм Грама-Шмідта. Синтаксис виклику:

`gramschmidt(M)` або `gschmidt(M)`.

Аргумент функції – матриця або список. Як компоненти системи векторів, на базі якої будується ортонормована система, розглядаються рядки матриці `M` або підсписки списку `M`. Для використання цієї функції необхідно завантажити пакет «`eigen`».

```
(%i15) load(eigen);
(%i16) L:gramschmidt(M);
```

$$(\%o16) \quad [[1,2,3],[\frac{2^2\cdot 3}{7},\frac{3}{7},-\frac{2\cdot 3}{7}]]$$

```
(%i17) L[1].L[2];
```

$$(\%o17) \quad \frac{2^2\cdot 3}{7}-\frac{12}{7}$$

```
(%i18) ratsimp(%o17);
```

$$(\%o18) \quad 0$$

Перетворення матриці до трикутної форми здійснюється методом виключення Гаусса за допомогою команди `echelon(M)`. Аналогічний результат дає команда `triangularize(M)`, відмінності полягають в тому, що `echelon` нормує діагональний елемент на 1, а `triangularize` – ні.

```
(%i19) triangularize(K);
```

$$(\%o19) \quad \begin{pmatrix} 14 & -20 \\ 0 & -18 \end{pmatrix}$$

```
(%i20) echelon(K);
```

$$(\%o20) \quad \begin{pmatrix} 1 & -\frac{10}{7} \\ 0 & 1 \end{pmatrix}$$

Для розрахунку рангу матриці (порядку найбільшого невиродженого мінора матриці) використовується команда `rank(M)`.

```
(%i21) rank(M);
```

$$(\%o21) \quad 2$$

```
(%i22) rank(N);
```

$$(\%o22) \quad 2$$

```
(%i23) echelon(N);
```

$$(\%o23) \quad \begin{pmatrix} 1 & -2 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

Мінор матриці знаходиться командою `minor(M,i,j)`, де `M` – матриця, `i,j` – індекси елементу, для якого обчислюється мінор.

```
(%i24) minor(M,1,1);
```

$$(\%o24) \quad \begin{pmatrix} 5 & 6 \end{pmatrix}$$

```
(%i25) minor(N,2,2);
```

$$(\%o25) \quad \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

5.2 Розв'язок системи лінійних алгебричних рівнянь

Нехай маємо наступну систему рівнянь:

$$\begin{cases} x + 2y + 10z = -15; \\ 2x + 4y - z = 12; \\ x + y - 3z = 9. \end{cases}$$

Спробуємо розв'язати її, використовуючи для демонстрації різні підходи.

5.2.1 Стандартний метод `solve`

Нагадаємо синтаксис команди `solve` для систем: необхідно сформувати список із рівнянь, та в кінці виписати списком змінні, відносно яких шукати розв'язок.

```
(%i1) solve([x+2*y+10*z=-15,
             2*x+4*y-z=12,
             x+y-3*z=9],
            [x,y,z]);
```

$$(\%o1) \quad [[x = 1, y = 2, z = -2]]$$

5.2.2 Метод Крамера

Знайдемо основний визначник, який складається з коефіцієнтів при змінних.

```
(%i2) D:matrix([1,2,10],[2,4,-1],[1,1,-3]);
```

$$(\%o2) \quad \begin{pmatrix} 1 & 2 & 10 \\ 2 & 4 & -1 \\ 1 & 1 & -3 \end{pmatrix}$$

```
(%i3) %Delta:determinant(D);
```

$$(\%o3) \quad -21$$

Оскільки $\Delta \neq 0$, система має єдиний розв'язок. Знайдемо тепер додаткові визначники, які утворюються підстановкою стовпчика правих частин рівнянь замість відповідного стовпчика коефіцієнтів при кожній змінній.


```
(%i2) D:matrix([1,2,10],[2,4,-1],[1,1,-3]);
```

$$(\%o2) \quad \begin{pmatrix} 1 & 2 & 10 \\ 2 & 4 & -1 \\ 1 & 1 & -3 \end{pmatrix}$$

```
(%i3) %Delta:determinant(D);
```

$$(\%o3) \quad -21$$

```
(%i4)  D1:matrix([-15,2,10],[12,4,-1],[9,1,-3]);
```

$$(\%o4)\quad \begin{pmatrix} -15 & 2 & 10 \\ 12 & 4 & -1 \\ 9 & 1 & -3 \end{pmatrix}$$

```
(%i5)  %Delta1:determinant(D1);
```

$$(\%o5)\quad -21$$

```
(%i6)  D2:matrix([1,-15,10],[2,12,-1],[1,9,-3]);
```

$$(\%o6)\quad \begin{pmatrix} 1 & -15 & 10 \\ 2 & 12 & -1 \\ 1 & 9 & -3 \end{pmatrix}$$

```
(%i7)  %Delta2:determinant(D2);
```

$$(\%o7)\quad -42$$

```
(%i8)  D3:matrix([1,2,-15],[2,4,12],[1,1,9]);
```

$$(\%o8)\quad \begin{pmatrix} 1 & 2 & -15 \\ 2 & 4 & 12 \\ 1 & 1 & 9 \end{pmatrix}$$

```
(%i9)  %Delta3:determinant(D3);
```

$$(\%o9)\quad 42$$

Тоді, за правилом Крамера $x = \frac{\Delta_1}{\Delta}, y = \frac{\Delta_2}{\Delta}, z = \frac{\Delta_3}{\Delta}$, маємо

```
(%i13) x=%Delta1/%Delta; y=%Delta2/%Delta; z=%Delta3/%Delta;
```

$$(\%o11)\quad x = 1$$

$$(\%o12)\quad y = 2$$

$$(\%o13)\quad z = -2$$

5.2.3 Метод Гаусса

Цей метод полягає у послідовному виділенні кожної змінної, виразивши її через інші. Підставляючи отримані невідомі знизу вгору, можна утворювати рівняння зі зменшеною на одиницю кількістю змінних, аж поки не дійдемо до рівняння з однією змінною. В матричному представленні це означає зведення розширеної матриці Гаусса до трикутного вигляду.

Сформуємо розширену основну матрицю `G`, що утворена з усіх коефіцієнтів і правих сторін системи.

```
(%i1)  G:matrix([1,2,10,-15],[2,4,-1,12],[1,1,-3,9]);
```

$$(\%o1)\quad \begin{pmatrix} 1 & 2 & 10 & -15 \\ 2 & 4 & -1 & 12 \\ 1 & 1 & -3 & 9 \end{pmatrix}$$

Зведемо цю матрицю до трикутного виду через команду `echelon`.

```
(%i2)  echelon(G);
```

$$(\%o2)\quad \begin{pmatrix} 1 & 2 & 10 & -15 \\ 0 & 1 & 13 & -24 \\ 0 & 0 & 1 & -2 \end{pmatrix}$$

Отримали систему рівнянь, яку розв'язуємо знизу догори.

```
(%i3)  solve([x+2*y+10*z=-15,y+13*z=-24,z=-2],[x,y,z]);
```

(%o3) $[[x = 1, y = 2, z = -2]]$

5.2.4 Метод оберненої матриці

Виписану раніше систему можна представити в загальному вигляді як

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix},$$

або скорочено $A \cdot X = B$. Домноживши зліва цю рівність на матрицю A^{-1}, можна вивільнити X та знайти невідомі: $X = A^{-1} \cdot B$.

```
(%i1)  A:matrix([1,2,10],[2,4,-1],[1,1,-3]);
```

(%o1) $\begin{pmatrix} 1 & 2 & 10 \\ 2 & 4 & -1 \\ 1 & 1 & -3 \end{pmatrix}$

```
(%i2)  B:matrix([-15],[12],[9]);
```

(%o2) $\begin{pmatrix} -15 \\ 12 \\ 9 \end{pmatrix}$

```
(%i3)  invA:invert(A);
```

(%o3) $\begin{pmatrix} \frac{11}{21} & -\frac{16}{21} & 2 \\ -\frac{5}{21} & \frac{13}{21} & -1 \\ \frac{2}{21} & -\frac{1}{21} & 0 \end{pmatrix}$

```
(%i4)  X=invA.B;
```

(%o4) $X = \begin{pmatrix} 1 \\ 2 \\ -2 \end{pmatrix}$

Як бачимо, всі методи приводять до однакових результатів.

5.3 Спеціальні функції для розв'язання систем рівнянь

Команда `linsolve([expr1,expr2,...,exprN],[x1,x2,...,xN])` – розв'язує список одночасних лінійних рівнянь `[expr1,expr2,...,exprN]` відносно переліку змінних `[x1,x2,...,xN]`. Вирази `expr` можуть бути поліномами зазначених змінних і представлятися в вигляді рівнянь. Кількість змінних не обов'язково повинна бути рівна кількості рівнянь.

Нехай маємо систему з трьох рівнянь, яку розв'яжемо командою `linsolve`.

$$\begin{cases} x + y + z + t = 0; \\ 2x - 2y + z + 3t = 2; \\ 3x - y + 2z - t = 8. \end{cases}$$

```
(%i1)  linsolve([x+y+z+t=0,2*x-2*y+z+3*t=2,3*x-y+2*z-t=8],[x,y,z,t]);
```

(%o1) $[x = -\frac{3 \cdot \%r1 - 8}{4}, y = -\frac{5 \cdot \%r1 + 16}{20}, z = \%r1, t = -\frac{6}{5}]$

Таким чином, загальний розв'язок має вигляд:

$$x = -\frac{3C-8}{4}, \quad y = -\frac{5C+16}{20}, \quad z = C, \quad t = -\frac{6}{5},$$

де C – довільна стала. Їй можна задавати довільні дійсні значення, при кожному з яких виходить частинний розв'язок. Наприклад, при $C = 1$ отримаємо розв'язок

```
(%i2)  ev((%o1),%r1=1);
```

(%o2) $[x = \frac{5}{4}, y = -\frac{21}{20}, z = 1, t = -\frac{6}{5}]$

Спосіб подання розв'язків визначається опцією `linsolve_params` (за умовчанням `true`) Якщо вказаний прапор встановлений у `true`, розв'язання недовизначених систем включає параметри `%r1,%r2` і т.д. Якщо прапор `linsolve_params` встановлений у `false`, пов'язані змінні виражаються через вільні (у попередньому прикладі це було б z).

Багато в чому аналогічний результат дозволяє отримати функція `algsys` (фактично це надбудова над `solve`). Кількість рівнянь може перевищувати кількість невідомих, чи навпаки.

Для обчислення коренів одиничних поліноміальних рівнянь використовується функція `realroots`. Варіанти синтаксису:

`realroots(expr,bound);`
`realroots(eqn,bound);`
`realroots(expr);`
`realroots(eqn).`

Тут `bound` – це межа точності, з якою потрібно знайти корені. Команда знаходить всі корені виразу `expr=0` або рівняння `eqn`. Функція будує послідовність Штурма для ізоляції кожного кореня та використовує алгоритм розподілу навпіл для уточнення кореня з точністю `bound` або точністю, заданою за замовчуванням.

Наприклад, потрібно розв'язати рівняння $x^5 - x + 2 = 0$ з точністю до $5 \cdot 10^{-6}$.

```
(%i3)  realroots(2-x+x^5,5e-6);
```

(%o3) $[x = -\frac{664361}{524288}]$

```
(%i4)  (%o3),numer;
```

(%o4) $[x = -1.267168045043945]$

```
(%i5)  ev(2-x+x^5,x=(%o3)),numer;
```

(%o5) $[x^5 - x + 2 = 3.085850166506532 \cdot 10^{-6}]$

Усі корені полінома (дійсні та комплексні) можна знайти за допомогою команди `allroots`. Спосіб подання розв'язку визначається змінною `polyfactor` (за умовчанням `false`; якщо встановити в `true`, то команде поверне результат факторизації). Алгоритм пошуку коренів напівчисельний.

```
(%i6)  allroots(x^4+1);
```

(%o6) $[x = 0.7071067811865475 \cdot i + 0.7071067811865476,$
$x = 0.7071067811865476 - 0.7071067811865475 \cdot i,$
$x = 0.7071067811865478 \cdot i - 0.7071067811865476,$
$x = -0.7071067811865478 \cdot i - 0.7071067811865476]$

```
(%i7)  polyfactor:true$ allroots(x^4+1);
```

(%o7) $1.0 \cdot \left(x^2 - 1.414213562373095 \cdot x + 0.9999999999999999\right) \cdot \left(x^2 + 1.414213562373095 \cdot x + 1.0\right)$

Спрощення систем рівнянь досягається також функцією `eliminate`, що дозволяє виключити ті чи інші змінні. Виклик `eliminate([eqn1,...,eqnN], [x1,...,xK])` виключає змінні `[x1,...,xK]` із зазначених виразів.

5.4 Комплексні числа

Комплексні вирази визначені в системі **Maxima** двома способами: в алгебричній формі $z = a + i \cdot b$ та експоненційній $z = r \cdot e^{i\phi}$, тригонометрична форма є додатковою та результати розв'язків у ній не відображаються.

```
(%i1)  solve(x^3-1=0,x);
```

(%o1) $[x = \frac{\sqrt{3} \cdot i - 1}{2}, x = -\frac{\sqrt{3} \cdot i + 1}{2}, x = 1]$

```
(%i2)  solve(x^5-1=0,x);
```

(%o2) $[x = e^{\frac{2 \cdot i \cdot \pi}{5}}, x = e^{\frac{4 \cdot i \cdot \pi}{5}}, x = e^{-\frac{4 \cdot i \cdot \pi}{5}}, x = e^{-\frac{2 \cdot i \cdot \pi}{5}}, x = 1]$

Кількість коренів, які повертаються **Maxima**, відповідає основній теоремі алгебри (рівняння третього степеня має три корені, п'ятого – п'ять і т.д.). Перетворення комплексних виразів може здійснюватися звичайними командами для спрощення (`radcan`, `expand` та ін), але передбачено і ряд специфічних функцій, що розраховані на операції саме з комплексними числами.

Команди `realpart(z)` та `imagpart(z)` обчислюють відповідно дійсну та уявну частину числа z.

```
(%i1)  z:sqrt(3)+%i;
```

(%o1) $i + \sqrt{3}$

```
(%i2)  realpart(z);
```

(%o2) $\sqrt{3}$

```
(%i3)  imagpart(z);
```

(%o3) 1

Команди `cabs(z)` та `carg(z)` – повертають відповідно модуль та аргумент числа z.

```
(%i4)  cabs(z);
```

(%o4) 2

```
(%i5)  cabs(exp((%pi*%i)/4));
```

(%o5) 1

```
(%i6)  carg(z);
```

(%o6) $\frac{\pi}{6}$

```
(%i7)  carg(exp((%pi*%i)/4));
```
(%o7) $\frac{\pi}{4}$

Комплексно спряжені числа обчислюються командою `conjugate(z)`.

```
(%i8)  conjugate(z);
```
(%o8) $\sqrt{3}-i$

```
(%i9)  conjugate(exp((%pi*%i)/4));
```
(%o9) $\frac{1}{\sqrt{2}}-\frac{i}{\sqrt{2}}$

Команда `polarform(z)` перетворює алгебричну форму комплексного числа в експоненційну, `rectform(z)` – навпаки, із експоненційної в алгебричну.

```
(%i10) polarform(z);
```
(%o10) $2\cdot e^{\frac{i\cdot\pi}{6}}$

```
(%i11) rectform(exp((%pi*%i)/4));
```
(%o11) $\frac{i}{\sqrt{2}}+\frac{1}{\sqrt{2}}$

Команда `residue(expr,z,z0)` – обчислює рештку у комплексній площині для виразу `expr`, коли змінна z приймає значення z_0. Нагадаємо, що рештка – це коефіцієнт при $(z-z_0)^{-1}$ у розкладі виразу `expr` в ряд Лорана.

```
(%i12) residue(sin(a*z)/z^4,z,0);
```
(%o12) $-\frac{a^3}{6}$

5.5 Завдання до Розділу 5

Операції з матрицями

До поданих матриць: знайти обернену матрицю, детермінант, характеристичний поліном, власні значення і власні вектори, ранг, звести до трикутної.

5.1. $\begin{pmatrix}-2 & 3 & 5\\ 2 & 1 & 4\\ 1 & -3 & -1\end{pmatrix}$ **5.2.** $\begin{pmatrix}3 & -3 & 4\\ 5 & -1 & 2\\ -1 & 3 & -2\end{pmatrix}$ **5.3.** $\begin{pmatrix}-5 & -2 & -4\\ 2 & 2 & 3\\ 4 & -2 & -1\end{pmatrix}$ **5.4.** $\begin{pmatrix}4 & -5 & -4\\ -3 & -2 & 5\\ 2 & 3 & -3\end{pmatrix}$

5.5. $\begin{pmatrix}-3 & 5 & 1\\ -1 & -5 & -2\\ -4 & 3 & 1\end{pmatrix}$ **5.6.** $\begin{pmatrix}4 & -3 & -1\\ -3 & 2 & -2\\ -1 & 3 & 5\end{pmatrix}$ **5.7.** $\begin{pmatrix}-1 & 4 & 4\\ 3 & 1 & -5\\ 5 & -1 & -3\end{pmatrix}$ **5.8.** $\begin{pmatrix}1 & -4 & 4\\ -3 & -3 & 5\\ 2 & -5 & -2\end{pmatrix}$

5.9. $\begin{pmatrix}-3 & 2 & 5\\ 3 & -1 & -4\\ -1 & -2 & 1\end{pmatrix}$ **5.10.** $\begin{pmatrix}-2 & 4 & 2\\ 1 & -3 & 3\\ 5 & -4 & -1\end{pmatrix}$ **5.11.** $\begin{pmatrix}-5 & 0 & 1\\ 2 & 3 & -4\\ -1 & 4 & -1\end{pmatrix}$ **5.12.** $\begin{pmatrix}3 & -2 & 2\\ -4 & 1 & 0\\ -3 & 4 & 5\end{pmatrix}$

Системи лінійних рівнянь

Розв'язати подані системи: прямим методом, методом Крамера, методом Гаусса, матричним методом.

5.13. $\begin{cases} 2x + y - 3z = -12; \\ 3x - 2y - z = 3; \\ -x + 5y + 2z = -3. \end{cases}$

5.14. $\begin{cases} 6x - 5y + z = 7; \\ 5x + 3y + 2z = 0; \\ -2x + y - 3z = 11. \end{cases}$

5.15. $\begin{cases} -3x - 2y + 4z = -15; \\ 2x + 5y - 3z = 3; \\ 4x - y + 7z = 15. \end{cases}$

5.16. $\begin{cases} -3x + y - z = 8; \\ -4x + 2y + 3z = -3; \\ 2x + 3y - 2z = -1. \end{cases}$

5.17. $\begin{cases} x - 3y - 4z = -7; \\ 5x + 2y + 2z = -1; \\ 4x - y - 5z = -6. \end{cases}$

5.18. $\begin{cases} 2x + 3y + 3z = 15; \\ 3x - 6y - 6z = -23; \\ -9x - 3y + 6z = 8. \end{cases}$

5.19. $\begin{cases} x + y + z = 6; \\ -x + y - z = -2; \\ 2x + 3y + z = 11. \end{cases}$

5.20. $\begin{cases} x - y - z = -11; \\ x + y - z = 15; \\ 2x - y + z = -9. \end{cases}$

5.21. $\begin{cases} 2x + y - 3z = -7; \\ 3x - 2y + z = 11; \\ -2x - 3y - 2z = 3. \end{cases}$

5.22. $\begin{cases} x + 2y + z = 1; \\ 3x - y + 2z = 13; \\ 2x + 3y - z = -8. \end{cases}$

5.23. $\begin{cases} -7x + 2y + 4z = 13; \\ 3x - y + 2z = -2; \\ -x + 6y + 5z = 12. \end{cases}$

5.24. $\begin{cases} 2x - 3y + 5z = 10; \\ x - y + 5z = 4; \\ -3x - 4y + 3z = 2. \end{cases}$

Комплексні числа

Для заданих комплексних чисел: вивести дійсну та уявну частину, знайти спряжене число, знайти модуль та аргумент числа, перевести у експоненційну форму.

5.25. $z = 3 + 5i$

5.26. $z = 2 - 10i$

5.27. $z = 2 + 2i$

5.28. $z = 1 - 7i$

5.29. $z = 4 + i$

5.30. $z = 7 - 2i$

5.31. $z = 1 + 4i$

5.32. $z = 6 + 2i$

5.33. $z = 6 - 3i$

Розділ 6

Диференціальне числення

6.1 Границі

Границя виразу $f(x)$ при $x \to a$ обчислюється за допомогою команди
`limit(f(x),x,a,opts)`,
де опція `opts` може набувати два значення: `plus` та `minus`, вони позначають правосторонню та лівосторонню границю. Нагадаємо, що `inf` позначає $+\infty$, `minf` позначає $-\infty$.

```
(%i1)  limit((1+A/x)^x,x,inf);
```

$$(\%o1) \quad e^{A}$$

```
(%i3)  limit(1/x,x,0,plus); limit(1/x,x,0,minus);
```

$$(\%o2) \quad \infty$$

$$(\%o3) \quad -\infty$$

```
(%i4)  y(x):=(x^3-3*x-2)/(x^2-x-2)^2;
```

$$(\%o4) \quad \mathrm{y}\,(x) := \frac{x^3 - 3 \cdot x - 2}{\left(x^2 - x - 2\right)^2}$$

```
(%i5)  limit(y(x),x,-1);
```

$$(\%o5) \quad -\frac{1}{3}$$

```
(%i6)  limit((x/(2+x))^(3*x),x,inf);
```

$$(\%o6) \quad e^{-6}$$

При операціях з раціональними дробами часто буває випадок невизначеності $0/0$, тоді для виділення носіїв нуля доцільно спочатку використати факторизацію виразів, і наочність скорочення «нулів» стане явною. Нехай маємо границю саме такого типу:

► Знайти границю $\lim\limits_{x \to +\infty} \dfrac{x^2 - 4}{x^2 - 3x + 2}$.

```
(%i7)  f(x):=(x^2-4)/(x^2-3*x+2);
```

$$(\%o7) \quad \mathrm{f}\,(x) := \frac{x^2 - 4}{x^2 - 3 \cdot x + 2}$$

```
(%i8)  factor(num(f(x)));
```

$$(\%o8) \quad (x-2) \cdot (x+2)$$

```
(%i9)  factor(denom(f(x)));
```
(%o9) $(x-2)\cdot(x-1)$

```
(%i10) ev((%o8)/(%o9));
```
(%o10) $\dfrac{x+2}{x-1}$

```
(%i11) limit((%o10),x,2);
```
(%o11) 4

При дії з виразами, що містять ірраціональні операції, діють схожим чином, але при цьому використовують команду `radcan`.

► Знайти границю $\lim\limits_{x\to 1}\dfrac{\sqrt{x}-1}{x-1}$.

```
(%i12) radcan((sqrt(x)-1)/(x-1));
```
(%o12) $\dfrac{1}{\sqrt{x}+1}$

```
(%i13) limit((%o12),x,1);
```
(%o13) $\dfrac{1}{2}$

Звісно, пряме обчислення границь дало б ті ж самі результати.

При обчисленнях з використанням **Maxima** доцільно також використовувати для знаходження складних границь розклад чисельника та знаменника в ряд Тейлора (більш детально про степеневі ряди та ряд Тейлова див. нижче). Якщо використовувати меню **wxMaxima**, потрібно у вкладці «Аналіз – Знайти границю» встановити пункт «Ряд Тейлора». Для обчислень використовується функція `tlimit`, работа якої заснована на заміні функцій рядами Тейлора (де це можливо). За замовчуванням прапорець заміни поставлений у `false`, тому для використання `tlimit` його потрібно поставити в `true`:

```
(%i14) tlimswitch=true;
```
(%o16) $true = true$

```
(%i15) f(x):=(tan(x)-sin(x))/(x-sin(x));
```
(%o15) $\mathrm{f}(x) := \dfrac{\tan(x)-\sin(x)}{x-\sin(x)}$

```
(%i16) tlimit(f(x),x,0);
```
(%o16) 3

6.2 Диференціювання

Пакет **Maxima** надає потужні засоби для диференціювання функцій та обчислення диференціалів. Для обчислення похідної слід ввести команду

`diff(f(x),x,N)` – похідна порядку N функції $f(x)$ по одній змінній x;

`diff(f(x,y,...),xi,N)` – похідна порядку N функції багатьох змінних $f(x,y,...)$ по змінній x_i.

```
(%i1)  f(x):=sin(9*x^2);
```
(%o1) $\mathrm{f}(x) := \sin\left(9\cdot x^2\right)$

```
(%i2)  diff(f(x),x);
```
$$(\%o2) \quad 18 \cdot x \cdot \cos\left(9 \cdot x^2\right)$$

```
(%i3)  diff(f(x),x,2);
```
$$(\%o3) \quad 18 \cdot \cos\left(9 \cdot x^2\right) - 324 \cdot x^2 \cdot \sin\left(9 \cdot x^2\right)$$

```
(%i4)  g(x,y):=(x+y^2)/sqrt(x^2+y^4);
```
$$(\%o4) \quad \mathrm{g}\left(x, y\right) := \frac{x + y^2}{\sqrt{x^2 + y^4}}$$

```
(%i5)  diff(g(x,y),x);
```
$$(\%o5) \quad \frac{1}{\sqrt{y^4 + x^2}} - \frac{x \cdot \left(y^2 + x\right)}{\left(y^4 + x^2\right)^{\frac{3}{2}}}$$

Якщо подіяти командою `diff` на функцію, не вказуючи змінну диференціювання, **Maxima** видасть результат із використанням виразів типу `del(x)`, який аналогічний до диференціалу змінної dx. Тобто виведення дасть повний диференціал функції $f(x)$.

```
(%i11) diff(log(x));
```
$$(\%o11) \quad \frac{\mathrm{del}\left(x\right)}{x}$$

```
(%i12) diff(exp(x*y));
```
$$(\%o12) \quad x \cdot e^{x \cdot y} \cdot \mathrm{del}\left(y\right) + y \cdot e^{x \cdot y} \cdot \mathrm{del}\left(x\right)$$

Якщо вказати апостроф перед символом `diff`, то похідна не обчислюється і спрощення, яке зумовлене за замовчуванням, не здійснюється.

```
(%i13) f(x,z):=x^2*z+z^2*x;
```
$$(\%o13) \quad \mathrm{f}\left(x, z\right) := x^2 \cdot z + z^2 \cdot x$$

```
(%i14) diff(f(x,z),x,2)+diff(f(x,z),z,3)+diff(f(x,z),x)*x^2;
```
$$(\%o14) \quad x^2 \cdot \left(z^2 + 2 \cdot x \cdot z\right) + 2 \cdot z$$

```
(%i15) 'diff(f(x,z),x,2)+'diff(f(x,z),z,3)+'diff(f(x,z),x)*x^2;
```
$$(\%o15) \quad \frac{d^3}{d\,z^3} \cdot \left(x \cdot z^2 + x^2 \cdot z\right) + \frac{d^2}{d\,x^2} \cdot \left(x \cdot z^2 + x^2 \cdot z\right) + x^2 \cdot \left(\frac{d}{d\,x} \cdot \left(x \cdot z^2 + x^2 \cdot z\right)\right)$$

6.3 Рівняння дотичної та нормалі

З курсу аналізу відомо, що рівняння дотичної до функції $f(x)$ в точці x_0 має вигляд

$$g(x) = f'(x_0) \cdot (x - x_0) + f(x_0);$$

а рівняння нормалі

$$h(x) = -\frac{1}{f'(x_0)} \cdot (x - x_0) + f(x_0).$$

6.3.1 Пряма функціональна залежність

При явному заданні функції достатньо знайти значення функції в точці та значення похідної функції в точці. Тут будемо користуватись командами

`define(func2,oper(func1))` – означення нової функції `func2` через певні операції над функцією `func1`;

`at(expr,[x=a,y=b,...])` – оператор підстановки.

► Задано функцію $y = x^2$ у точці $x_0 = 1$. Завдання: знайти рівняння дотичної та нормалі та накреслити їх графіки.

```
(%i1)  f(x):=x^2;
```

$$(\%o1) \quad \mathrm{f}(x) := x^2$$

```
(%i2)  x0:1;
```

$$(\%o2) \quad 1$$

```
(%i3)  y0:f(x0);
```

$$(\%o3) \quad 1$$

```
(%i4)  define(df(x),diff(f(x),x));
```

$$(\%o4) \quad \mathrm{df}(x) := 2 \cdot x$$

```
(%i5)  k1:at(df(x),x=x0);
```

$$(\%o5) \quad 2$$

```
(%i6)  k2:at(-1/df(x),x=x0);
```

$$(\%o6) \quad -\frac{1}{2}$$

```
(%i7)  g:expand(k1*(x-x0)+f(x0));
```

$$(\%o7) \quad 2 \cdot x - 1$$

```
(%i8)  h:expand(k2*(x-x0)+f(x0));
```

$$(\%o8) \quad -\frac{x-3}{2}$$

```
(%i9)  plot2d([f(x),g,h,[discrete,[[x0,y0]]],[x,-3,3],[y,-1,4],
       [style,[lines,2,1],[lines,2,2],[lines,2,3],[points,3,4,1]],
       [legend,false],[same_xy]);
```

Графік зображений на Рис. 6.1,(а).

6.3.2 Непряма залежність змінних через рівняння

Перейдемо до складнішого випадку. Як знайти рівняння дотичної та нормалі до функції, якщо вона задана не функціональним виглядом, а у вигляді рівняння $f(x, y) = 0$? Тут

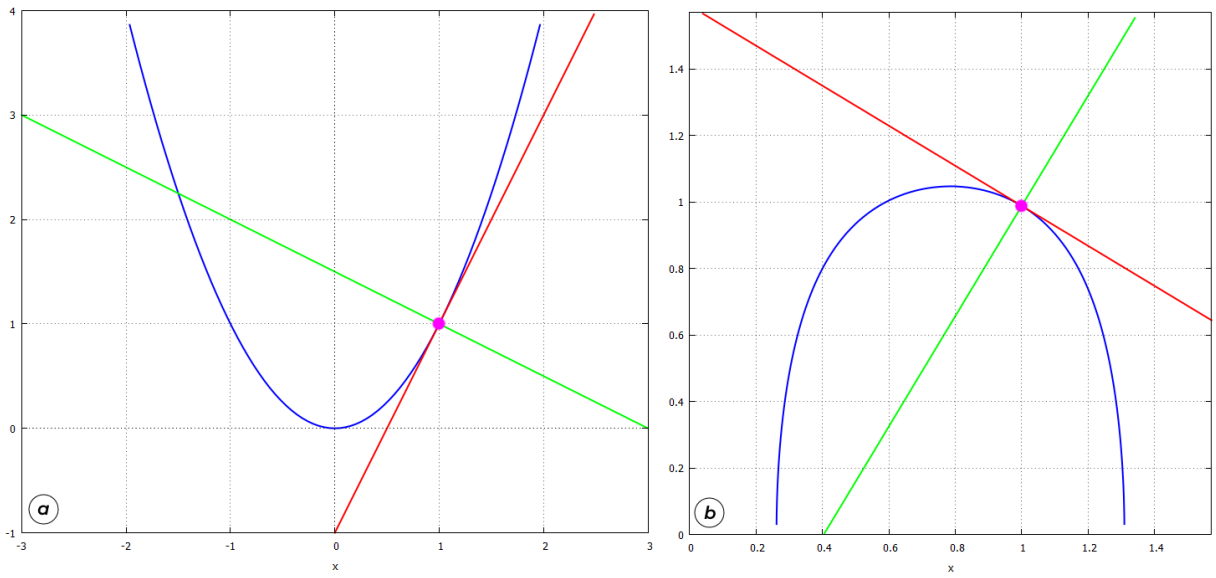

Рис. 6.1: Графіки функції, дотичної, нормалі: пряма (а) та непряма (б) залежності.

нам згодиться наступна рівність для похідної функції, що доводиться в математичному аналізі:

$$\frac{dy}{dx} = -\frac{\dfrac{\partial f(x,y)}{\partial x}}{\dfrac{\partial f(x,y)}{\partial y}}.$$

► Нехай маємо функціональну залежність $\sin 2x \cos y = 0,5$; $x_0 = 1$. Завдання: знайти рівняння дотичної та нормалі та накреслити їх графіки.

```
(%i1)  f(x,y):=sin(2*x)*cos(y)-1/2;
```

$$(\%o1) \quad \mathrm{f}(x,y) := \sin(2\cdot x)\cdot\cos(y) - \frac{1}{2}$$

```
(%i2)  define(df(x,y),-diff(f(x,y),x)/diff(f(x,y),y));
```

$$(\%o2) \quad \mathrm{df}(x,y) := \frac{2\cdot\cos(2\cdot x)\cdot\cos(y)}{\sin(2\cdot x)\cdot\sin(y)}$$

```
(%i3)  x0:1;
```

$$(\%o3) \quad 1$$

```
(%i4)  s1:solve(f(x,y)=0,y);
```

solve: using arc-trig functions to get a solution.
Some solutions will be lost.

$$(\%o4) \quad [y = \operatorname{acos}\left(\frac{1}{2\cdot\sin(2\cdot x)}\right)]$$

```
(%i5)  f:rhs(s1[1]);
```

$$(\%o5) \quad \operatorname{acos}\left(\frac{1}{2\cdot\sin(2\cdot x)}\right)$$

```
(%i6)  y0:at(f,x=x0),numer;
```

$$(\%o6) \quad 0.988581650693521$$

```
(%i7)  k1:at(df(x,y),[x=x0,y=y0]),numer;
```

(%o7) -0.6025870564746433

```
(%i8)  k2:at(-1/df(x,y),[x=x0,y=y0]),numer;
```

(%o8) 1.659511251121737

```
(%i9)  g:expand((k1*(x-x0)+y0));
```

(%o9) $1.591168707168164 - 0.6025870564746433 \cdot x$

```
(%i10) h:expand((k2*(x-x0)+y0));
```

(%o10) $1.659511251121737 \cdot x - 0.6709296004282159$

```
(%i11) plot2d([f,g,h,[discrete,[[x0,y0]]]],[x,0,%pi/2],[y,0,%pi/2],
       [style,[lines,2,1],[lines,2,2],[lines,2,3],[points,3,4,1]],
       [legend,false],[same_xy]);
```

Графік зображений на Рис. 6.1,(б).

6.3.3 Параметричне представлення функції

При параметричному заданні функції маємо наступні співвідношення:

$$\begin{cases} x = \phi(t); \\ y = \psi(t); \end{cases} \qquad \frac{dy}{dx} = \frac{\psi'(t)}{\phi'(t)}.$$

► Нехай маємо функцію $x = \phi(t) = \sin t; \quad y = \psi(t) = \sin 2t; \quad x_0 = 0,8$. Завдання: знайти рівняння дотичної та нормалі та накреслити їх графіки.

```
(%i1)  x(t):=sin(t);
```

(%o1) $\mathrm{x}\,(t) := \sin\,(t)$

```
(%i2)  y(t):=sin(2*t);
```

(%o2) $\mathrm{y}\,(t) := \sin\,(2 \cdot t)$

```
(%i3)  define(df(t),diff(y(t),t)/diff(x(t),t));
```

(%o3) $\mathrm{df}\,(t) := \dfrac{2 \cdot \cos\,(2 \cdot t)}{\cos\,(t)}$

```
(%i4)  x0:0.8;
```

(%o4) 0.8

```
(%i5)  s1:solve(x0=x(t),t),numer;
```

(%o5) $[t = 0.9272952180016114]$

```
(%i6)  t0:rhs(s1[1]);
```

(%o6) 0.9272952180016114

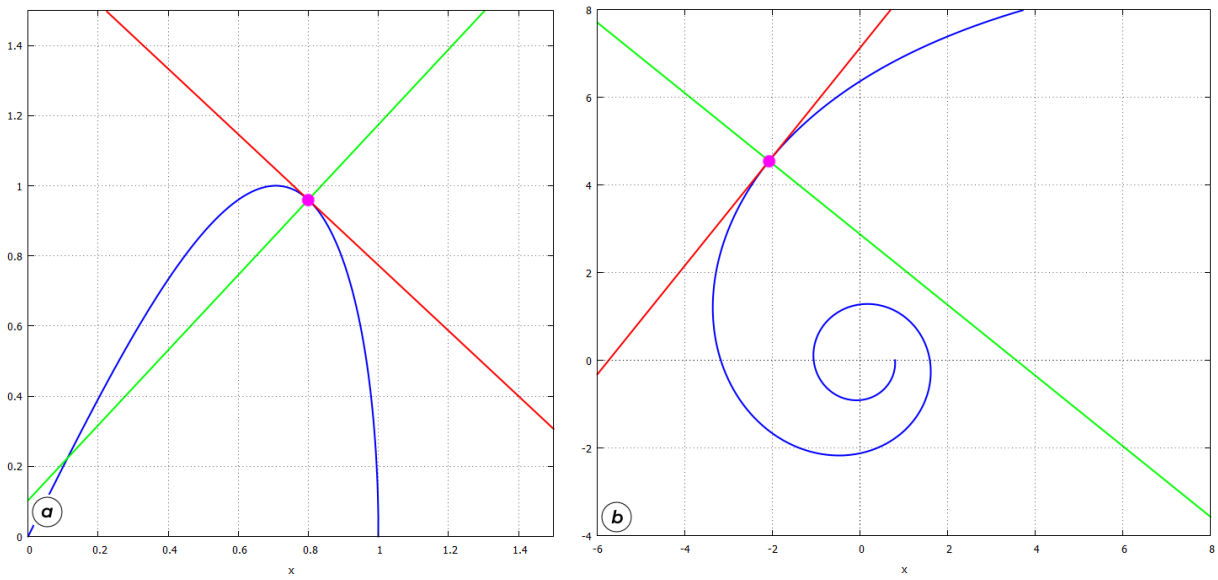

Рис. 6.2: Графіки функції, дотичної, нормалі: параметричне (а) та полярне (б) представлення.

```
(%i7)  k1:at(df(t),t=t0);
```
(%o7) -0.9333333333333271

```
(%i8)  k2:at(-1/df(t),t=t0);
```
(%o8) 1.071428571428578

```
(%i9)  y0:y(t0);
```
(%o9) 0.9600000000000004

```
(%i10) g:expand(k1*(x-x0)+y0);
```
(%o10) $1.706666666666662 - 0.9333333333333271 \cdot x$

```
(%i11) h:expand(k2*(x-x0)+y0);
```
(%o11) $1.071428571428578 \cdot x + 0.1028571428571374$

```
(%i18) plot2d([[parametric,x(t),y(t),[t,0,%pi/2]],g,h,
               [discrete,[[x0,y0]]]],[x,0,1.5],[y,0,1.5],
               [style,[lines,2,1],[lines,2,2],[lines,2,3],[points,3,4,1]],
               [legend,false],[nticks,80],[same_xy]);
```

Графік зображений на Рис. 6.2,(а).

6.3.4 Полярне представлення функції

При полярному заданні функції маємо наступні співвідношення:

$$\begin{cases} x(\theta) = \rho(\theta)\cos\theta; \\ y(\theta) = \rho(\theta)\sin\theta; \end{cases} \qquad \frac{dy}{dx} = \frac{y'(\theta)}{x'(\theta)}.$$

▶ Нехай маємо функціональну залежність $\rho(\theta) = \dfrac{10}{\theta}$, $\theta = 2$ рад. Завдання: знайти рівняння дотичної та нормалі та накреслити їх графіки.

```
(%i1)  x(t):=(10/t)*cos(t);
```

$$(\%o1) \quad \mathrm{x}\,(t) := \frac{10 \cdot \cos\,(t)}{t}$$

```
(%i2)  y(t):=(10/t)*sin(t);
```

$$(\%o2) \quad \mathrm{y}\,(t) := \frac{10 \cdot \sin\,(t)}{t}$$

```
(%i3)  t0:2;
```

$$(\%o3) \quad 2$$

```
(%i4)  define(df(t),diff(y(t),t)/diff(x(t),t));
```

$$(\%o4) \quad \mathrm{df}\,(t) := \frac{\frac{10\cdot\cos(t)}{t} - \frac{10\cdot\sin(t)}{t^2}}{-\frac{10\cdot\sin(t)}{t} - \frac{10\cdot\cos(t)}{t^2}}$$

```
(%i5)  x0:at(x(t),t=t0),numer;
```

$$(\%o5) \quad -2.080734182735712$$

```
(%i6)  y0:at(y(t),t=t0),numer;
```

$$(\%o6) \quad 4.546487134128409$$

```
(%i7)  k1:at(df(t),t=t0),numer;
```

$$(\%o7) \quad 1.241822212787614$$

```
(%i8)  k2:at(-1/df(t),t=t0),numer;
```

$$(\%o8) \quad -0.8052682499173711$$

```
(%i9)  g:expand(k1*(x-x0)+y0);
```

$$(\%o9) \quad 1.241822212787614 \cdot x + 7.1303890611561$$

```
(%i10) h:expand(k2*(x-x0)+y0);
```

$$(\%o10) \quad 2.87093796025357 - 0.8052682499173711 \cdot x$$

```
(%i13) plot2d([[parametric,x(t),y(t),[t,1,4*%pi]],g,h,
              [discrete,[[x0,y0]]]],[x,-6,8],[y,-4,8],
              [style,[lines,2,1],[lines,2,2],[lines,2,3],[points,3,4,1]],
              [legend,false],[nticks,80],[same_xy]);
```

Графік зображений на Рис. 6.2,(б).

6.4 Дослідження функції та побудова її графіка

Схема дослідження функції може включати в себе досить багато пунктів, але ми зупинимось на основних.

1. Точки перетину з осями:
   $f(x) = 0$ – перетин з віссю x;
   $f(0) = y_0$ – перетин з віссю y.
2. Асимптоти функції:
   $\lim_{x\to\infty} f(x) = a$; $\quad y = a$ – горизонтальна асимптота (шукаються окремо $x \to \pm\infty$);
   $\lim_{x\to b} f(x) = \infty$; $\quad x = b$ – вертикальна асимптота (шукаються окремо $x \to b \pm 0$).
3. Критичні точки та точки, підозрілі на екстремум:
   $f'(x) = 0$ або не існує.
4. Області зростання та спадання:
   $f'(x) > 0$ – функція зростає;
   $f'(x) < 0$ – функція спадає;
5. Точки екстремумів:
   $f'(x) = 0$ та $f''(x) > 0$ – у цій точці функція досягає мінімума;
   $f'(x) = 0$ та $f''(x) < 0$ – у цій точці функція досягає максимума.
6. Точки перегину:
   $f''(x) = 0$.
7. Проміжки вгнутості та опуклості:
   $f''(x) > 0$ – функція вгнута;
   $f''(x) < 0$ – функція опукла.
8. Графік функції.

► Нехай задана функція $f(x) = \dfrac{x+1}{x^2+1}$. Завдання: дослідити цю функцію та побудувати її графік.

```
(%i1)  f(x):=(x+1)/(x^2+1);
```

(%o1) $\mathrm{f}(x) := \dfrac{x+1}{x^2+1}$

1. Точки перетину з осями:

```
(%i2)  solve(f(x)=0,x);
```

(%o2) $[x = -1]$

```
(%i3)  y0:at(f(x),x=0);
```

(%o3) 1

З віссю X: $(-1; 0)$, з віссю Y: $(0; 1)$.
2. Асимптоти:

```
(%i4)  limit(f(x),x,inf);
```

(%o4) 0

```
(%i5)  limit(f(x),x,minf);
```

(%o5) 0

```
(%i6)  realroots(denom(f(x)));
```

(%o6) []

Горизонтальна асимптота: $y = 0$. Вертикальних асимптот немає.
3. Критичні точки:

```
(%i7)  define(df(x),diff(f(x),x));
```

(%o7) $$\mathrm{df}\,(x) := \frac{1}{x^2+1} - \frac{2\cdot x\cdot (x+1)}{\left(x^2+1\right)^2}$$

```
(%i8)  s1:solve(df(x)=0,x);
```

(%o8) $[x = -\sqrt{2}-1, x = \sqrt{2}-1]$

```
(%i9)  x1:rhs(s1[1]);
```

(%o9) $-\sqrt{2}-1$

```
(%i10) x2:rhs(s1[2]);
```

(%o10) $\sqrt{2}-1$

```
(%i11) f(x1),numer;
```

(%o11) -0.2071067811865475

```
(%i12) f(x2),numer;
```

(%o12) 1.207106781186547

Дві критичні точки: $\left(-\sqrt{2}-1; -0.2071067811865475\right)$; $\left(\sqrt{2}-1; 1.207106781186547\right)$.
4. Області зростання та спадання:

```
(%i13) load(solve_rat_ineq);
(%i14) solve_rat_ineq(df(x)>0);
```

(%o14) $[[x > -\sqrt{2}-1, x < \sqrt{2}-1]]$

```
(%i15) solve_rat_ineq(df(x)<0);
```

(%o15) $[[x < -\sqrt{2}-1], [x > \sqrt{2}-1]]$

Функція зростає при $x \in \left(-\sqrt{2}-1; \sqrt{2}-1\right)$, спадає при $x \in \left(-\infty; -\sqrt{2}-1\right) \cup \left(\sqrt{2}-1; +\infty\right)$.
5. Точки екстремумів:

```
(%i16) define(ddf(x),diff(f(x),x,2));
```

(%o16) $$\mathrm{ddf}\,(x) := -\frac{2\cdot (x+1)}{\left(x^2+1\right)^2} - \frac{4\cdot x}{\left(x^2+1\right)^2} + \frac{8\cdot x^2\cdot (x+1)}{\left(x^2+1\right)^3}$$

```
(%i17) ddf(x1),numer;
```

(%o17) 0.06066017177982131

```
(%i18) ddf(x2),numer;
```

(%o18) -2.060660171779821

Друга похідна у першій критичній точці x_1 додатня, тому ця точка – мінімум; друга похідна у другій критичній точці x_2 від'ємна, тому ця точка – максимум.

6. Точки перегину:

```
(%i19) s2:solve(ddf(x)=0,x);
```

(%o19) $[x = -\sqrt{3} - 2, x = \sqrt{3} - 2, x = 1]$

```
(%i20) xinf1:rhs(s2[1]);
```

(%o20) $-\sqrt{3} - 2$

```
(%i21) xinf2:rhs(s2[2]);
```

(%o21) $\sqrt{3} - 2$

```
(%i22) xinf3:rhs(s2[3]);
```

(%o22) 1

```
(%i23) f(xinf1),numer;
```

(%o23) -0.1830127018922193

```
(%i24) f(xinf2),numer;
```

(%o24) 0.6830127018922193

```
(%i25) f(xinf3),numer;
```

(%o25) 1

Є три точки перегину: $(-\sqrt{3}-2; -0.1830127018922193)$, $(\sqrt{3}-2; 0.6830127018922193)$, $(1; 1)$.

7. Проміжки вгнутості та опуклості:

```
(%i26) solve_rat_ineq(ddf(x)>0);
```

(%o26) $[[x > -\sqrt{3} - 2, x < \sqrt{3} - 2], [x > 1]]$

```
(%i27) solve_rat_ineq(ddf(x)<0);
```

(%o27) $[[x < -\sqrt{3} - 2], [x > \sqrt{3} - 2, x < 1]]$

Функція вгнута при $x \in \left(-\sqrt{3} - 2; \sqrt{3} - 2\right) \cup (1; +\infty)$, опукла при $x \in (-\infty; -\sqrt{3} - 2) \cup (\sqrt{3} - 2; 1)$.

8. Графік (Рис. 6.3):

```
(%i24) plot2d([f(x),[discrete,[[x1,f(x1)],[x2,f(x2)]]],
       [discrete,[[xinf1,f(xinf1)],[xinf2,f(xinf2)],[xinf3,f(xinf3)]]]],
       [x,-5,5],[y,-0.5,1.5],[style,[lines,3,1],[points,3,2,1],
       [points,3,5,1]],[legend,false],[axes,solid],[same_xy]);
```

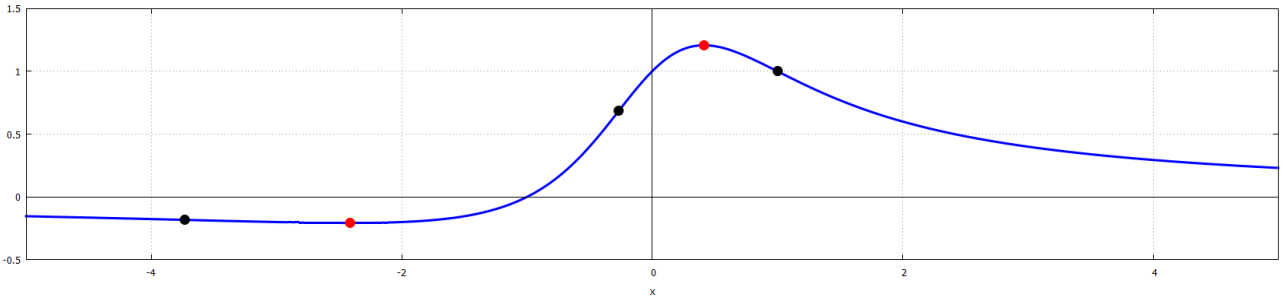

Рис. 6.3: Графік функції, точками позначені екстремуми (●) та точки перегину (●).

6.5 Екстремуми функцій двох змінних

Нехай маємо функцію двох змінних $f(x,y)$. Критичні точки такої функції шукаються з системи рівнянь $\{f'_x(x,y)=0; f'_y(x,y)=0\}$. Тип критичної точки визначається другими похідними функції за такими правилами:

– якщо $f''_{xx}\cdot f''_{yy}-(f''_{xy})^2>0$ та $f''_{xx}>0$ – мінімум;
– якщо $f''_{xx}\cdot f''_{yy}-(f''_{xy})^2>0$ та $f''_{xx}<0$ – максимум;
– якщо $f''_{xx}\cdot f''_{yy}-(f''_{xy})^2<0$ – екстремума немає.

Тут відповідні похідні другого порядку обчислені в отриманих парах (x_0,y_0) (їх може бути кілька).

▶ Дослідити функцію на екстремум: $z=x^3-\frac{9}{2}x^2+6x+y^2-4y-12$.

Задаємо функцію.

```
(%i1)  f(x,y):=x^3-9/2*x^2+6*x+y^2-4*y-12;
```

$$(\%o1)\quad \mathrm{f}(x,y):=x^3-\frac{9\cdot x^2}{2}+6\cdot x+y^2-4\cdot y-12$$

Знаходимо перші похідні та розв'язуємо систему рівнянь.

```
(%i2)  dfx:diff(f(x,y),x);
```

$$(\%o2)\quad 3\cdot x^2-9\cdot x+6$$

```
(%i3)  dfy:diff(f(x,y),y);
```

$$(\%o3)\quad 2\cdot y-4$$

```
(%i4)  s1:solve([dfx,dfy],[x,y]);
```

$$(\%o4)\quad [[x=1,y=2],[x=2,y=2]]$$

Маємо дві критичні точки: $(1,2)$ та $(2,2)$. Тепер шукаємо другі похідні.

```
(%i5)  dfxx:diff(dfx,x);
```

$$(\%o5)\quad 6\cdot x-9$$

```
(%i6)  dfxy:diff(dfx,y);
```

$$(\%o6)\quad 0$$

```
(%i7)  dfyy:diff(dfy,y);
```

$$(\%o7)\quad 2$$

Шукаємо значення других похідних у двох точках та дві комбінації $f''_{xx} \cdot f''_{yy} - (f''_{xy})^2$ у цих точках.

```
(%i8)  [A,B,C]:ev([dfxx,dfxy,dfyy],x=1,y=2);
```

(%o8) $[-3, 0, 2]$

```
(%i9)  A*C-B^2;
```

(%o9) -6

```
(%i10) [A,B,C]:ev([dfxx,dfxy,dfyy],x=2,y=2);
```

(%o10) $[3, 0, 2]$

```
(%i11) A*C-B^2;
```

(%o11) 6

Бачимо, що для першої точки $(1, 2)$ комбінація рівна $-6 < 0$, тому вона не є ні максимумом, ні мінімумом. У другій точці $(2, 2)$ комбінація рівна $6 > 0$ і $f''_{xx} = 3 > 0$, отже це точка мінімуму. Зобразимо графік поверхні (Рис. 6.4).

```
(%i12) plot3d(f(x,y),[x,0,5],[y,0,5],[legend,"f(x,y)"]);
```

(%o12) $[C : /Users/orreg/maxout.gnuplot]$

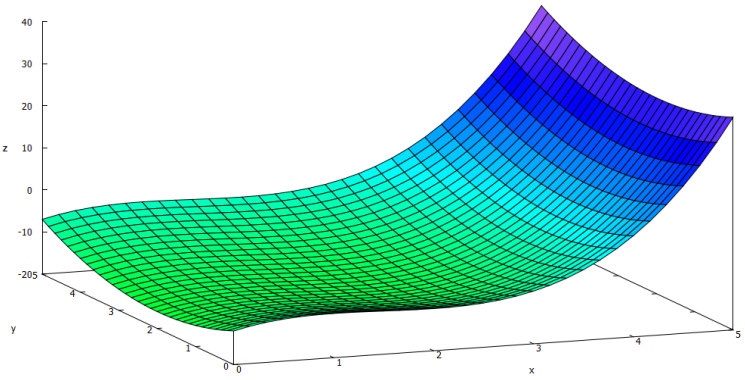

Рис. 6.4: Графік функції двох змінних

6.6 Розклад функції в ряд

Для спеціалістів інженерного профілю дуже важливим є одночасне знаходження розв'язку в замкнутій аналітичній формі та отримання чисельних значень результату. Представлення функції у вигляді степеневого ряду дозволяє звести вивчення властивостей складної функції до вивчення цих властивостей у відповідного апроксимуючого поліноміального розкладу. Заміна функцій на їх степеневі розклади допомагає вивченню границь, аналізу збіжності та розбіжності рядів та інтегралів, наближеному обчисленню інтегралів та розв'язанню диференціальних рівнянь.

Найважливішим таким інструментом є розклад функцій у вигляді рядів Тейлора: це степеневий ряд виду

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!} \cdot (x - x_0)^k,$$

де числова функція f припускається визначеною в деякому околі точки x_0 та має в цій точці похідні всіх порядків, (k) позначає похідну k-го порядку. Якщо точка $x_0 = 0$, розклад називається рядом Маклорена.

Многочленами Тейлора для функції $f(x)$ порядку n називаються частинні суми ряду Тейлора:

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + ... + \frac{f^{(n)}}{n!}(x_0)(x - x_0)^n.$$

В **Maxima** існує спеціальна команда, що дозволяє обчислити ряди та многочлени Тейлора для функції $f(x)$:

`taylor(f(x),x,a,n)`,

де x – змінна, по якій проводиться розклад, a – точка, в околі якої розкладається функція, n – порядок розкладу, він має бути представлений цілим додатнім числом.

Знайдемо роклади в ряд Маклорена важливих функцій:

```
(%i1)  taylor(%e^x,x,0,7);
```

$$(\%o1)/T/\ 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040} + ...$$

```
(%i2)  taylor(sin(x),x,0,7);
```

$$(\%o2)/T/\ x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + ...$$

```
(%i3)  taylor(cos(x),x,0,7);
```

$$(\%o3)/T/\ 1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720} + ...$$

```
(%i4)  taylor(log(1+x),x,0,7);
```

$$(\%o4)/T/\ x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \frac{x^7}{7} + ...$$

Оцінимо точність розкладу в ряд експоненти порівняно з її значенням в точці 1.

```
(%i5)  t:taylor(%e^x,x,0,9);
```

$$(\%o5)/T/\ 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040} + \frac{x^8}{40320} + \frac{x^9}{362880} + ...$$

```
(%i6)  e9:ev(t,x=1);
```

$$(\%o6)/R/\ \frac{98641}{36288}$$

```
(%i7)  e9,numer;
(%o7)     2.718281525573192

(%i8)  bfloat(%e-e9);
(%o8)     3.028858530396942b − 7
```

Бачимо, що дев'ять перших доданків забезпечують точність, більшу за 10^{-6}. Взагалі при використанні розкладу Тейлора для більш високих степенів точність наближення зростає, однак варто зауважити, що цей степінь не можна підвищувати необмежено (у зв'язку з накопиченням погрішності у обчисленнях, що швидко зводить нанівець всі

побудови розкладу, адже накопичена помилка дуже швидко стане набагато більшою за значення чергового члена розкладу в точці).

Разом з командою `taylor` для розкладу функцій та виразів в ряд використовується команда

`powerseries(f(x),x,a)` – будує розклад за змінною x в околі точки a, результатом виконання є побудова ряду Тейлора в загальному вигляді.

```
(%i9)  powerseries(sin(x),x,0);
```

$$(\%o9) \quad \sum_{i1=0}^{\infty} \frac{(-1)^{i1} \cdot x^{2 \cdot i1+1}}{(2 \cdot i1+1)!}$$

```
(%i10) powerseries(x/(1+x),x,0);
```

$$(\%o10) \quad x \cdot \sum_{i2=0}^{\infty} (-1)^{i2} \cdot x^{i2}$$

```
(%i11) powerseries(cos(x^3),x,0);
```

$$(\%o11) \quad \sum_{i3=0}^{\infty} \frac{(-1)^{i3} \cdot x^{6 \cdot i3}}{(2 \cdot i3)!}$$

► Цікавим є також відтворити класичне зображення із підручників матаналізу – порівняння точного графіку функції із розкладами її в ряд різних порядків. Зробимо це, для прикладу, із функцією $\cos x$.

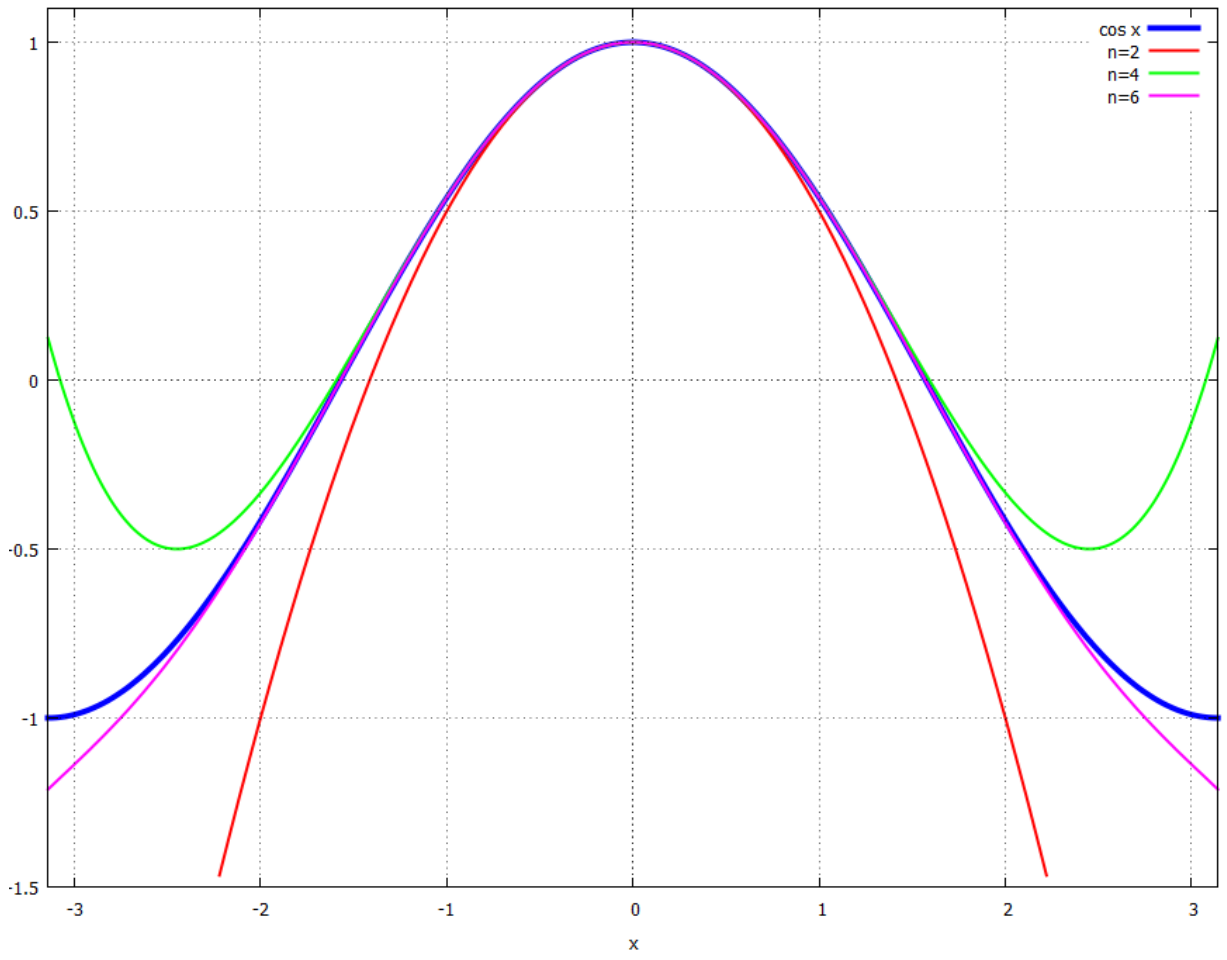

Рис. 6.5: Порівняння $\cos x$ з його поліномами розкладу в точці $x = 0$

```
(%i12) cos2:taylor(cos(x),x,0,2);
```

$$(\%o12)/T/1-\frac{x^2}{2}+\ldots$$

```
(%i13) cos4:taylor(cos(x),x,0,4);
```

$$(\%o13)/T/1-\frac{x^2}{2}+\frac{x^4}{24}+\ldots$$

```
(%i14) cos6:taylor(cos(x),x,0,6);
```

$$(\%o14)/T/1-\frac{x^2}{2}+\frac{x^4}{24}-\frac{x^6}{720}+\ldots$$

```
(%i15) plot2d([cos(x),cos2,cos4,cos6],[x,-%pi,%pi],[y,-1.5,1.1],
              [style,[lines,5,1],[lines,2,2],[lines,2,3],[lines,2,4]],
              [legend,"cos x","n=2","n=4","n=6"]);
```

Графік на Рис. 6.5.

У випадку розкладу функцій кількох змінних ряд Тейлора знаходиться за формулою

$$f(x,y)=\sum_{k=0}^{\infty}\left[(x-a)\frac{\partial}{\partial x}+(y-b)\frac{\partial}{\partial y}\right]^k f(a,b).$$

Синтаксис виклику **Maxima** наступний:

`taylor(f(x,y),[x,y],[a,b],n)` – розклад $f(x,y)$ в ряд Тейлора навколо точки $[a,b]$ n-го порядку.

▶ Розкласти функцію $z=e^{-x^2-y^2}$ в ряд навколо точки $[1,1]$ порядку 2.

```
(%i1)  f(x,y):=exp(-x^2-y^2);
```

$$(\%o1)\qquad \mathrm{f}\,(x,y):=\exp\left(-x^2-y^2\right)$$

```
(%i2)  taylor(f(x,y),[x,y],[1,1],2);
```

$$(\%o2)/T/\frac{1}{e^2}-\frac{2\cdot(x-1)+2\cdot(y-1)}{e^2}+\frac{(x-1)^2+4\cdot(y-1)\cdot(x-1)+(y-1)^2}{e^2}$$

▶ Розкласти функцію $z=\sin(x+y)$ навколо точки $[0,0]$ порядку 5.

```
(%i3)  taylor(sin(x+y),[x,y],[0,0],5);
```

$$(\%o3)/T/\ y+x-\frac{x^3+3\cdot y\cdot x^2+3\cdot y^2\cdot x+y^3}{6}+\frac{x^5+5\cdot y\cdot x^4+10\cdot y^2\cdot x^3+10\cdot y^3\cdot x^2+5\cdot y^4\cdot x+y^5}{120}$$

6.7 Диференціальні операції векторного аналізу

Для прямих обчислень, пов'язаних з операціями векторного аналізу, необхідно завантажити пакет `vect`. Крім того, у застосуванні операторів `div, curl, grad, laplacian` до деякої функції необхідно використовувати команду `express`. Нагадаємо, що оператори `div, curl` застосовуються до векторів (`curl` рівний оператору `rot` у німецькій традиції), а `grad, laplacian` – до скалярів. Процедура обчислень, наприклад градієнту функції, наступна: спочатку діємо оператором `grad` на функцію, потім командою `express` виражаємо отриманий вираз через прямі диференціальні операції, і нарешті командою `ev(...,diff)`

знаходимо результат обчислень у вигляді компонент вектора. Дві останні операції можна провести одночасно.

```
(%i1)  load(vect);
(%i2)  F(x,y,z):=x^2*y+exp(y*z);
```

$$(\%o2) \quad \mathrm{F}\,(x,y,z) := x^2 \cdot y + \exp\,(y \cdot z)$$

```
(%i3)  gradF:grad(F(x,y,z));
```

$$(\%o3) \quad \mathrm{grad}\left(e^{y \cdot z} + x^2 \cdot y\right)$$

```
(%i4)  express(gradF);
```

$$(\%o4) \quad \left[\frac{d}{d\,x} \cdot \left(e^{y \cdot z} + x^2 \cdot y\right), \frac{d}{d\,y} \cdot \left(e^{y \cdot z} + x^2 \cdot y\right), \frac{d}{d\,z} \cdot \left(e^{y \cdot z} + x^2 \cdot y\right)\right]$$

```
(%i5)  ev(%o4,diff);
```

$$(\%o5) \quad [2 \cdot x \cdot y, z \cdot e^{y \cdot z} + x^2, y \cdot e^{y \cdot z}]$$

Ротор векторної функції:

```
(%i6)  G(x,y,z):=[x^2,2*y^2*z,%e^(x*z)];
```

$$(\%o6) \quad \mathrm{G}\,(x,y,z) := [x^2, 2 \cdot y^2 \cdot z, e^{x \cdot z}]$$

```
(%i7)  curlG:curl(G(x,y,z));
```

$$(\%o7) \quad \mathrm{curl}\left([x^2, 2 \cdot y^2 \cdot z, e^{x \cdot z}]\right)$$

```
(%i8)  ev(express(curlG),diff);
```

$$(\%o8) \quad [-2 \cdot y^2, -z \cdot e^{x \cdot z}, 0]$$

Оператор Лапласа:

```
(%i9)  laplacian(x^2+2*y^2+3*z^2);
```

$$(\%o9) \quad \mathrm{laplacian}\left(3 \cdot z^2 + 2 \cdot y^2 + x^2\right)$$

```
(%i10) ev(express(%o9),diff);
```

$$(\%o10) \quad 12$$

Перевіримо відомі формули: rot grad $F(\vec{r}) = 0$, div rot $\vec{G}(\vec{r}) = 0$.

```
(%i16) div(curlG);
```

$$(\%o16) \quad 0$$

```
(%i17) curl(gradF);
```

$$(\%o17) \quad 0$$

6.8 Сумація рядів

Maxima реалізує числові ряди як один з видів списків, їх ми розглядали в Розділі 3. Нагадаємо синтаксис команди для пошуку суми ряду:

`sum(expr,k,k1,kN)` – сумує ряд `expr` по індексу `k`, `k1` – номер першого члену сумації, `kN` – останнього (він може бути рівний нескінченності).

```
(%i1)  sum(1/3^k,k,1,7);
```

$$(\%o1) \quad \frac{1093}{2187}$$

```
(%i2)  sum(1/k,k,1,10);
```

$$(\%o2) \quad \frac{7381}{2520}$$

Якщо спрямувати кількість членів сумації до нескінченності, **Maxima** просто поверне введене значення у вигляді символьного виразу.

```
(%i3)  sum(1/3^k,k,1,inf);
```

$$(\%o3) \quad \sum_{k=1}^{\infty} \frac{1}{3^k}$$

Щоб програма здійснила сумацію, необхідно після введення команди `sum` долучити опцію `simpsum` через кому.

```
(%i4)  sum(1/3^k,k,1,inf),simpsum;
```

$$(\%o4) \quad \frac{1}{2}$$

▶ Таким чином можна знаходити суму широкого класу рядів. Розглянемо, наприклад, ряди виду $\sum_{k=0}^{\infty} \frac{1}{k^n}$ для різних n.

```
(%i5)  sum(1/k,k,1,inf),simpsum;
```

sum: sum is divergent.

– an error. To debug this try: debugmode(true);

```
(%i6)  sum(1/k^2,k,1,inf),simpsum;
```

$$(\%o6) \quad \frac{\pi^2}{6}$$

```
(%i7)  sum(1/k^3,k,1,inf),simpsum;
```

$$(\%o7) \quad \zeta\,(3)$$

```
(%i8)  sum(1/k^4,k,1,inf),simpsum;
```

$$(\%o8) \quad \frac{\pi^4}{90}$$

```
(%i9)  sum(1/k^5,k,1,inf),simpsum;
```

$$(\%o9) \quad \zeta\,(5)$$

Бачимо, що перший ряд розбіжний, як і має бути за інтегральною ознакою, для $n = 3; 5$ виникає ζ-функція Рімана.

Знакозмінні ряди **Maxima** залишає без змін і не обчислює їх, якщо ці ряди не можна звести до знакосталих:

```
(%i10) sum((-1)^k*(2*k+1)/2^k,k,0,inf),simpsum;
```

$$(\%o10)\quad \sum_{k=0}^{\infty} \frac{(2\cdot k+1)\cdot(-1)^k}{2^k}$$

```
(%i11) sum((-1)^k/2^k,k,0,inf),simpsum;
```

$$(\%o11)\quad \frac{2}{3}$$

Якщо сума ряду не зводиться до аналітичного вигляду, **Maxima** здійснює сумацію з якою завгодно наперед заданою точністю, але інтерфейс **wxMaxima** виводить на екран тільки перші 16 цифр після коми. Це обмеження можна розширити, задавши значення опції `fpprec:N`, де `N` – кількість цифр після коми, після цього записавши команду `bfloat(expr)`.

```
(%i12) sum(1/k^k,k,1,10),numer;
(%o12)  1.291285997059043

(%i13) sum(1/k^k,k,1,100),numer;
(%o13)  1.291285997062663

(%i14) sum(1/k^k,k,1,1000),numer;
(%o14)  1.291285997062663

(%i15) fpprec:200;
(%o15)  200

(%i16) bfloat(sum(1/k^k,k,1,1000),numer);
(%o16)  1.291285997062663540407282590[143 цифр]799729177948273009025649230 6b0
```

Щоб побачити всі 143 цифри, зазначені у квадратних дужках, необхідно команду задавати у консольній версії **Maxima** або у інтерфейсі **XMaxima**.

6.9 Завдання до Розділу 6

Знайти рівняння дотичної та нормалі до функції в точці, побудувати їх графіки

Функції, задані явно:

6.1. $y = x^{e^x}$; $x_0 = 1$;

6.2. $y = \ln x - x^2$; $x_0 = 4$;

6.3. $y = \operatorname{tg} x + \cos x$; $x_0 = \pi/2$;

6.4. $y = 1/x + x^2$; $x_0 = 2$;

6.5. $y = e^{\sqrt{x}}$; $x_0 = 2$;

6.6. $y = x^x$; $x_0 = 4$;

6.7. $y = (x^2-1)^3$; $x_0 = 2$;

6.8. $y = x \ln x$; $x_0 = 2$;

6.9. $y = \frac{3}{x^2+1}$; $x_0 = 3$;

6.10. $y = \frac{1}{\cos x}$; $x_0 = \pi/3$.

Функції, задані неявно:

6.11. $x^2+y^2=x^4+y^4;\quad x_0=1/2;$

6.12. $3x^2y^2-5x+\sin y=3y-1;\quad x_0=0;$

6.13. $\ln y=\operatorname{arctg}\frac{y}{x};\quad x_0=2;$

6.14. $3x^2-5y^2+9x=25-15y;\quad x_0=2.$

6.15. $\cos x^2=xe^y;\quad x_0=0.3.$

6.16. $x^3y^3-2y=x;\quad x_0=1.$

6.17. $(x+y)^2=y-x;\quad x_0=-2.$

6.18. $x^4+y^3-3x^2y=0;\quad x_0=1.$

6.19. $y^2+x^3-y^3+6=3y;\quad x_0=-1.$

6.20. $\sin y+x^2y^3-\cos x=2y;\quad x_0=2.$

Функції, задані параметрично:

6.21. $\begin{cases}x(t)=4-2t;\\ y(t)=3+6t-4t^2;\end{cases}\quad t_0=4.$

6.22. $\begin{cases}x(t)=\sqrt{t+1};\\ y(t)=\frac{1}{t+1};\end{cases}\quad t_0=1.8.$

6.23. $\begin{cases}x(t)=3\sin t;\\ y(t)=-4\cos t;\end{cases}\quad t_0=\pi.$

6.24. $\begin{cases}x(t)=3\sin\left(\frac{t}{3}\right);\\ y(t)=-4\cos\left(\frac{t}{3}\right);\end{cases}\quad t_0=5.$

6.25. $\begin{cases}x(t)=3-2\cos 3t;\\ y(t)=1+4\sin 3t;\end{cases}\quad t_0=1.$

6.26. $\begin{cases}x(t)=4\sin\left(\frac{t}{4}\right);\\ y(t)=1-2\cos^2\left(\frac{t}{4}\right);\end{cases}\quad t_0=\pi.$

6.27. $\begin{cases}x(t)=\sqrt{4+\cos\left(\frac{5t}{2}\right)};\\ y(t)=1+\frac{1}{3}\cos\left(\frac{5t}{2}\right)\end{cases}\quad t_0=1/5.$

6.28. $\begin{cases}x(t)=2e^t;\\ y(t)=\cos\left(1+e^{3t}\right);\end{cases}\quad t_0=1.$

6.29. $\begin{cases}x(t)=\frac{1}{2}e^{-3t};\\ y(t)=e^{-6t}+2e^{-3t}-8;\end{cases}\quad t_0=2.$

6.30. $\begin{cases}x(t)=\sin 3t;\\ y(t)=e^{-3t};\end{cases}\quad t_0=2.$

Функції, задані полярно:

6.31. $\rho=\dfrac{2}{\sin\theta-3\cos\theta};\quad \theta_0=2.$

6.32. $\rho=1+\cos\theta;\quad \theta_0=3.$

6.33. $\rho=\sin^2\theta;\quad \theta_0=1.$

6.34. $\rho=1+\dfrac{\pi^2}{\theta};\quad \theta_0=2.$

6.35. $\rho=\dfrac{1}{\sqrt[3]{\cos\theta\sin^2\theta}};\quad \theta_0=1.$

6.36. $\rho=e^{2\theta};\quad \theta_0=-1.$

6.37. $\rho=2\cos 3\theta;\quad \theta_0=\pi/6.$

6.38. $\rho=\dfrac{2}{\theta^2};\quad \theta_0=1/2.$

6.39. $\rho=3\theta^2;\quad \theta_0=1.$

6.40. $\rho=\sqrt{1+\theta^2};\quad \theta_0=\pi/4.$

Дослідити функцію та побудувати її графік. Вказати на графіку екстремуми та точки перегину

6.41. $y=\dfrac{1}{4}x^4-\dfrac{1}{3}x^3-x^2;$

6.42. $y=\dfrac{x}{(1-x^2)^2};$

6.43. $y=\dfrac{x-2}{\sqrt{x^2+1}};$

6.44. $y=3x-x^3;$

6.45. $y=\sqrt{(x-1)(x-2)(x-3)};$

6.46. $y=\sin x\sin 3x; x\in[0,\pi];$

6.47. $y=\dfrac{e^x}{1+x};$

6.48. $y=\dfrac{\ln x}{\sqrt{x}};$

6.49. $y=\sqrt{1-e^{-x^2}};$

6.50. $y=\ln(x+\sqrt{1+x^2});$

Розділ 7

Диференціальні рівняння

Система **Maxima** може успішно розв'язувати такі типи диференціальних рівнянь першого порядку: з відокремленими змінними, лінійні, нелінійні рівняння, однорідні, неоднорідні; типи рівнянь другого порядку: зі сталими коефіцієнтами, лінійні однорідні з несталими коефіцієнтами, які можуть бути перетворені на рівняння зі сталими коефіцієнтами, рівнянням Ейлера, рівняння, що розв'язуються методом варіації сталої та рівняння, що допускають пониження порядку.

Розглянемо команди системи **Maxima** для пошуку розв'язків диференціальних рівнянь та їх систем у символьному вигляді.

`desolve(deqn,y(x))` – шукає частинні розв'язки лінійних диференціальних рівнянь першого та другого порядку.

`desolve([deqn1,...,deqnN],[y1,...,yN])` – шукає частинні розв'язки систем лінійних диференціальних рівнянь першого та другого порядку.

Робота цієї команди заснована на перетворенні Лапласа заданих дифрівнянь. Вона приймає два аргументи, перший з яких – рівняння або перелік рівнянь, другий – одна залежна змінна або список залежних змінних відповідно.

`desolve` знаходить розв'язок задачі Коші для відповідного рівняння, тому якщо значення залежної змінної або її похідної не задані, то у знайденому розв'язку вони подаються у вигляді `y(0)` або $\left.\frac{\mathtt{d}}{\mathtt{dx}}\mathtt{y(x)}\right|_{\mathtt{x=0}}$.

Значення функцій та похідних у початковій точці можна задати за допомогою функції `atvalue`, яка має наступний синтаксис:

`atvalue(f(x),x=a,A)` – присвоює значення `A` функції $f(x)$ в точці $x = a$;

`atvalue('diff(f(x),x),x=a,B)` – присвоює значення `B` похідній функції $f'(x)$ в точці $x = a$.

Необхідно підкреслити, що похідні в рівняннях та системах, розв'язуваних за допомогою цієї команди, повинні бути записані у вигляді `'diff(f(x),x)`, тобто оператор похідної у відкладеному вигляді (з апострофом), а аргумент функції виписаний явно.

Якщо команда `desolve` не може знайти розв'язок, вона повертає значення `false`.

`ode2(deqn,y,x)` – призначений для розв'язання звичайних лінійних диференціальних рівнянь першого та другого порядку.

Розв'язок знаходиться у загальній формі, він може повернутись як у явній, так і неявній формі. Тут у списку параметрів явним чином вказується залежна змінна, тому позначення типу $y(x)$ не потрібні: функція та змінна позначаються одиночними літерами.

Як і для звичайних рівнянь, є можливість перевірити розв'язок за допомогою підстановки. Довільна константа для рівнянь першого порядку позначається через `%c`. У рівняннях другого порядку константи позначаються як `%k1` та `%k2`. Якщо функція ode2 не може отримати розв'язок з якоїсь причини, вона повертає значення `false`, при цьому

видаляючи звіт про помилку.

Окрім `ode2` існує ще три команди для пошуку частинних розв'язків на основі отриманих загальних. Тобто ці команди, отримуючи конкретні умови щодо значення функції-розв'язку в заданій точці, знаходять виходячи з цих передумов відповідні значення констант інтегрування. Для розв'язку задач Коші (Intial Value Problems, IVP)) використовуються функції `ic1` і `ic2` у випадку ЗДР першого та другого порядку відповідно. Крайові (граничні) задачі (Boundary Value Problems, BVP) розв'язують з використанням функції `bc2`.

`ic1(solution,x=x0,y=y0)` – обробляє розв'язок диференціального рівняння першого порядку з початковими умовами. Приймає три аргументи: перший – це сам розв'язок, у формі, в якій він знаходиться функцією ode2, другий – це початкове значення незалежної змінної у вигляді $x = x_0$, третій – значення функції в цій точці $y_0 = y(x_0)$. Повертає частинний розв'язок, що проходить через точку із зазначеними координатами (x_0, y_0).

`ic2(solution,x=x0,y=y0,dy=dy0)` – призначений для обробки розв'язку диференціального рівняння другого порядку з початковими умовами. Приймає чотири аргументи: до аргументів команди `ic1` додається `dy0` - початкове значення першої похідної залежної змінної відносно незалежної змінної у формі $y'(x_0) = dy0$.

`bc2(solution,x1,y1,x2,y2)` – розв'язує крайову задачу для дифрівняння другого порядку. Тут `solution` – загальний розв'язок рівняння, який отримується функцією `ode2`; x_1 – задає значення незалежної змінної в першій граничній точці при $x = x_1$, y_1 – визначає значення залежної змінної в першій граничній точці у вигляді $y_1 = y(x_1)$. Вирази x_2 і y_2 задають аналогічні значення для цих змінних у другій граничній точці.

7.1 Диференціальні рівняння першого порядку

7.1.1 Рівняння з розділеними змінними

Це рівняння виду $y'(x) = f(x) \cdot g(y)$, **Maxima** може розв'язати їх різними способами.

▶ Знайти загальний і частинний розв'язок рівняння $y' = \dfrac{\sqrt{1-y^2}}{\sqrt{1-x^2}}$ при $y(0) = 1$.

```
(%i1)  deqn1:'diff(y,x)=sqrt(1-y^2)/sqrt(1-x^2);
```

$$(\%o1) \quad \frac{d}{d\,x} \cdot y = \frac{\sqrt{1-y^2}}{\sqrt{1-x^2}}$$

Загальний розв'язок:

```
(%i2)  sol1:ode2(deqn1,y,x);
```

$$(\%o2) \quad \operatorname{asin}(y) = \operatorname{asin}(x) + \%c$$

Частинний розв'язок при початкових умовах $x_0 = 0, y_0 = 1$:

```
(%i3)  ic1(sol1,x=0,y=1);
```

$$(\%o3) \quad \operatorname{asin}(y) = \frac{2 \cdot \operatorname{asin}(x) + \pi}{2}$$

▶ Ще приклад із використанням `desolve`: $y'' = x^2y^2$, $y(0) = 1, y'(0) = 4$.

```
(%i4)  deqn2:'diff(y(x),x,2)=y^2*x^2;
```

$$(\%o4) \quad \frac{d^2}{d\,x^2} \cdot \mathrm{y}\,(x) = x^2 \cdot y^2$$

```
(%i5)  atvalue(y(x),x=0,1);
```

(%o5) 1

```
(%i6)  atvalue('diff(y(x),x),x=0,4);
```

(%o6) 4

```
(%i7)  desolve(deqn2,y(x));
```

(%o7) $$\mathrm{y}\left(x\right)=\frac{x^4\cdot y^2}{12}+4\cdot x+1$$

Якщо початкові умови, зазначені командою `atvalue`, не задати, система поверне розв'язок у наступному вигляді:

```
(%i1)  deqn2:'diff(y(x),x,2)=y^2*x^2;
```

(%o1) $$\frac{d^2}{d\,x^2}\cdot \mathrm{y}\left(x\right)=x^2\cdot y^2$$

```
(%i2)  desolve(deqn2,y(x));
```

(%o2) $$\mathrm{y}\left(x\right)=\frac{x^4\cdot y^2}{12}+x\cdot\left(\frac{d}{d\,x}\cdot \mathrm{y}\left(x\right)\Big|_{x=0}\right)+\mathrm{y}\left(0\right)$$

▶ Розв'язати один з видів рівняння Бернуллі $y'+2e^xy=2e^x\sqrt{y}$.

```
(%i7)  deqn4:'diff(y,x)+2*%e^x*y=2*%e^x*sqrt(y);
```

(%o7) $$\frac{d}{d\,x}\cdot y+2\cdot e^x\cdot y=2\cdot e^x\cdot\sqrt{y}$$

```
(%i8)  ode2(deqn4,y,x);
```

(%o8) $$-\log\left(\sqrt{y}-1\right)=e^x+\%c$$

```
(%i9)  method;
```

(%o9) *separable*

▶ Розв'яжемо систему рівнянь

$$\begin{cases} f'(x)=g'(x)+\sin x; \\ g'(x)=5f'(x)-\cos x; \end{cases} \qquad f(0)=1; \quad g(0)=7.$$

```
(%i1)  deqn1:'diff(f(x),x)='diff(g(x),x)+sin(x);
```

(%o1) $$\frac{d}{d\,x}\cdot \mathrm{f}\left(x\right)=\frac{d}{d\,x}\cdot \mathrm{g}\left(x\right)+\sin\left(x\right)$$

```
(%i2)  deqn2:'diff(g(x),x)=5*'diff(f(x),x)-cos(x);
```

(%o2) $$\frac{d}{d\,x}\cdot \mathrm{g}\left(x\right)=5\cdot\left(\frac{d}{d\,x}\cdot \mathrm{f}\left(x\right)\right)-\cos\left(x\right)$$

```
(%i4)  atvalue(f(x),x=0,1);atvalue(g(x),x=0,7);
```

(%o3) 1

(%o4) 7

```
(%i5)  desolve([deqn1,deqn2],[f(x),g(x)]);
```

$$(\%o5)\quad [\mathrm{f}\,(x)=\frac{\sin(x)}{4}+\frac{\cos(x)}{4}+\frac{3}{4},\mathrm{g}\,(x)=\frac{\sin(x)}{4}+\frac{5\cdot\cos(x)}{4}+\frac{23}{4}]$$

▶ Розв'язати задачу Коші для рівняння $xy'=y(1+\ln y-\ln x),\quad y(1)=e^2$.

```
(%i1)  eqn5:x*'diff(y,x)=y*(log(y)-log(x)+1);
```

$$(\%o1)\quad x\cdot\left(\frac{d}{d\,x}\cdot y\right)=y\cdot(\log(y)-\log(x)+1)$$

```
(%i3)  sol5:ode2(eqn5,y,x);
```

$$(\%o3)\quad x=\%c\cdot(\log(y)-\log(x))$$

```
(%i6)  ic1(sol5,x=1,y=%e^2);
```

$$(\%o6)\quad x=\frac{\log(y)-\log(x)}{2}$$

7.1.2 Однорідні рівняння

Це рівняння виду $y'(x)=f\left(\frac{y}{x}\right)$.

▶ Розв'язати рівняння $y'=\left(\frac{y}{x}\right)^2+2\frac{y}{x}$.

```
(%i1)  deqn1:'diff(y,x)=(y/x)^2+2*(y/x);
```

$$(\%o1)\quad \frac{d}{d\,x}\cdot y=\frac{y^2}{x^2}+\frac{2\cdot y}{x}$$

```
(%i2)  ode2(deqn1,y,x);
```

$$(\%o2)\quad -\frac{x\cdot y+x^2}{y}=\%c$$

```
(%i3)  ic1(%o2,x=3,y=1);
```

$$(\%o3)\quad -\frac{x\cdot y+x^2}{y}=-12$$

```
(%i4)  solve(%o3,y);
```

$$(\%o4)\quad [y=-\frac{x^2}{x-12}]$$

7.1.3 Рівняння в повних диференціалах

Це рівняння виду $P(x,y)dx+Q(x,y)dy=0$.

Якщо виконується умова $\dfrac{\partial P}{\partial y}=\dfrac{\partial Q}{\partial x}$, то ліва частина являє собою повний диференціал деякої функції. Якщо ж ця умова не виконується, але задовільняється теорема єдиності, то можна підібрати такий інтегруючий множник $\mu(x)$, який зводить ліву частину до повного диференціалу. Цей множник можна викликати командою `intfactor`, а метод розв'язання викликається командою `method`.

Для розв'язку в **Maxima** ліву частину небхідно представити у вигляді

$$P(x,y)+Q(x,y)\frac{dy}{dx}=0.$$

▶ Розв'язати рівняння $2xy + x^2y + \frac{y^3}{3} + (x^2 + y^2)y' = 0$.

```
(%i1)  deqn1:(2*x*y+x^2*y+y^3/3)+(x^2+y^2)*'diff(y,x)=0;
```

$$(\%o1) \quad \left(y^2 + x^2\right) \cdot \left(\frac{d}{d\,x} \cdot y\right) + \frac{y^3}{3} + x^2 \cdot y + 2 \cdot x \cdot y = 0$$

```
(%i2)  ode2(deqn1,y,x);
```

$$(\%o2) \quad \frac{e^x \cdot y^3 + 3 \cdot x^2 \cdot e^x \cdot y}{3} = \%c$$

```
(%i3)  intfactor;
```

$$(\%o3) \quad e^x$$

```
(%i4)  method;
```

$$(\%o4) \quad exact$$

▶ Розв'язати рівняння $3x^2 + 6xy^2 + (6x^2y + 4y^3)y' = 0$.

```
(%i5)  deqn2:(3*x^2+6*x*y^2)+(6*x^2*y+4*y^3)*'diff(y,x)=0;
```

$$(\%o5) \quad \left(4 \cdot y^3 + 6 \cdot x^2 \cdot y\right) \cdot \left(\frac{d}{d\,x} \cdot y\right) + 6 \cdot x \cdot y^2 + 3 \cdot x^2 = 0$$

```
(%i6)  ode2(deqn2,y,x);
```

$$(\%o6) \quad y^4 + 3 \cdot x^2 \cdot y^2 + x^3 = \%c$$

```
(%i7)  intfactor;
```

$$(\%o7) \quad 1$$

```
(%i8)  method;
```

$$(\%o8) \quad exact$$

7.1.4 Рівняння Бернуллі

Це рівняння виду

$$y'(x) = \sum_i Q_i(x) \cdot y^{\alpha_i}, \quad \alpha_i = \text{const}.$$

▶ Розв'язати рівняння $y' = \frac{4}{x}y + x\sqrt{y}$.

```
(%i1)  deqn1:'diff(y,x)=4/x*y+x*sqrt(y);
```

$$(\%o1) \quad \frac{d}{d\,x} \cdot y = \frac{4 \cdot y}{x} + x \cdot \sqrt{y}$$

```
(%i2)  ode2(deqn1,y,x);
```

$$(\%o2) \quad y = x^4 \cdot \left(\frac{\log(x)}{2} + \%c\right)^2$$

```
(%i3)  method;
```

$$(\%o3) \quad bernoulli$$

▶ Розв'язати рівняння $y' + \frac{y}{x} = -xy^2$.

```
(%i4)  deqn2:'diff(y,x)+y/x=-x*y^2;
```

$$(\%o4)\quad \frac{d}{d\,x}\cdot y+\frac{y}{x}=-x\cdot y^2$$

```
(%i5)  ode2(deqn2,y,x);
```

$$(\%o5)\quad y=\frac{1}{x\cdot(x+\%c)}$$

```
(%i6)  method;
```

$$(\%o6)\quad bernoulli$$

▶ Розв'язати задачу Коші для рівняння $y'-8y=x$, $y(0)=2$.

```
(%i8)  deqn3:5*'diff(y(x),x)-8*y=x; atvalue(y(x),x=0,2);
```

$$(\%o7)\quad 5\cdot\left(\frac{d}{d\,x}\cdot \mathrm{y}\left(x\right)\right)-8\cdot y=x$$

$$(\%o8)\quad 2$$

```
(%i9)  desolve(deqn3,y(x));
```

$$(\%o9)\quad \mathrm{y}\left(x\right)=\frac{8\cdot x\cdot y}{5}+\frac{x^2}{10}+2$$

```
(%i10) method;
```

$$(\%o10)\quad bernoulli$$

7.2 Диференціальні рівняння другого порядку

В **Maxima** за допомогою команди `ode2` можливе пряме розв'язання лише лінійних диференціальних рівнянь другого порядку $y''+p(x)y'+q(x)y=r(x)$. Спочатку відшукується розв'язок однорідного рівняння у формі $y=k_1y_1+k_2y_2$ (k_1,k_2 – довільні сталі), потім шукається частинний розв'язок методом варіації сталих.

7.2.1 Рівняння зі сталими коефіцієнтами

Це рівняння виду $y''+ay'+by=r(x)$.

▶ Розв'язати неоднорідне рівняння загального виду $2y''-y'-y=4xe^{2x}$.

```
(%i1)  deqn1:2*'diff(y,x,2)-'diff(y,x)-y=4*x*exp(2*x);
```

$$(\%o1)\quad 2\cdot\left(\frac{d^2}{d\,x^2}\cdot y\right)-\frac{d}{d\,x}\cdot y-y=4\cdot x\cdot e^{2\cdot x}$$

```
(%i2)  ode2(deqn1,y,x);
```

$$(\%o2)\quad y=\frac{(20\cdot x-28)\cdot e^{2\cdot x}}{25}+\%k1\cdot e^{x}+\%k2\cdot e^{-\frac{x}{2}}$$

Командою `yp` можна виділити частинний розв'язок.

```
(%i3)  yp;
```

$$(\%o3)\quad \frac{(20\cdot x-28)\cdot e^{2\cdot x}}{25}$$

▶ Розв'язати неоднорідне рівняння з кратними коренями характеристичного рівняння $y''-2y'+y=xe^x$.

```
(%i4)  deqn2:'diff(y,x,2)-2*'diff(y,x)+y=x*exp(x);
```

$$(\%o4)\quad \frac{d^2}{d\,x^2}\cdot y-2\cdot\left(\frac{d}{d\,x}\cdot y\right)+y=x\cdot e^x$$

```
(%i5)  ode2(deqn2,y,x);
```

$$(\%o5)\quad y=\frac{x^3\cdot e^x}{6}+(\%k2\cdot x+\%k1)\cdot e^x$$

```
(%i6)  yp;
```

$$(\%o6)\quad \frac{x^3\cdot e^x}{6}$$

► Розв'язати неоднорідне рівняння з комплексними коренями $y''+y=x\sin x$.

```
(%i7)  deqn3:'diff(y,x,2)+y=x*sin(x);
```

$$(\%o7)\quad \frac{d^2}{d\,x^2}\cdot y+y=x\cdot\sin(x)$$

```
(%i8)  ode2(deqn3,y,x);
```

$$(\%o8)\quad y=\frac{2\cdot x\cdot\sin(x)+(1-2\cdot x^2)\cdot\cos(x)}{8}+\%k1\cdot\sin(x)+\%k2\cdot\cos(x)$$

```
(%i9)  yp;
```

$$(\%o9)\quad \frac{2\cdot x\cdot\sin(x)+(1-2\cdot x^2)\cdot\cos(x)}{8}$$

7.2.2 Рівняння зі змінними коефіцієнтами

Це рівняння виду $y''+p(x)y'+q(x)y=r(x)$.

► Розв'язати задачу Коші для рівняння $x^2y''-xy'=3x^3, y(1)=1, y'(1)=4$.

```
(%i1)  deqn1:x^2*'diff(y,x,2)-x*'diff(y,x)=3*x^3;
```

$$(\%o1)\quad x^2\cdot\left(\frac{d^2}{d\,x^2}\cdot y\right)-x\cdot\left(\frac{d}{d\,x}\cdot y\right)=3\cdot x^3$$

```
(%i2)  sol1:ode2(deqn1,y,x);
```

$$(\%o2)\quad y=x^3+\%k2\cdot x^2-\frac{\%k1}{2}$$

```
(%i3)  ic2(sol1,x=1,y=1,'diff(y,x)=4);
```

$$(\%o3)\quad y=x^3+\frac{x^2}{2}-\frac{1}{2}$$

► Розв'язати рівняння в загальному вигляді: $xy''+y'=x^2$.

```
(%i4)  deqn2:x*'diff(y,x,2)+'diff(y,x)=x^2;
```

$$(\%o4)\quad x\cdot\left(\frac{d^2}{d\,x^2}\cdot y\right)+\frac{d}{d\,x}\cdot y=x^2$$

```
(%i5)  ode2(deqn2,y,x);
```

$$(\%o5)\quad y=\%k1\cdot\log(x)+\frac{x^3}{9}+\%k2$$

► Розв'язати задачу Коші для рівняння $y''-7y'+6y=(x-2)e^x, y(0)=1, y'(0)=3$.

```
(%i6)  deqn3:'diff(y,x,2)-7*'diff(y,x)+6*y=(x-2)*%e^x;
```

$$(\%o6)\quad \frac{d^2}{d\,x^2}\cdot y-7\cdot\left(\frac{d}{d\,x}\cdot y\right)+6\cdot y=(x-2)\cdot e^x$$

```
(%i7)  sol3:ode2(deqn3,y,x);
```

$$(\%o7)\quad y=\%k1\cdot e^{6\cdot x}-\frac{(25\cdot x^2-90\cdot x-18)\cdot e^x}{250}+\%k2\cdot e^x$$

```
(%i8)  ic2(sol3,x=0,y=1,'diff(y,x)=3);
```

$$(\%o8)\quad y=\frac{41\cdot e^{6\cdot x}}{125}-\frac{(25\cdot x^2-90\cdot x-18)\cdot e^x}{250}+\frac{3\cdot e^x}{5}$$

7.3 Диференціальні рівняння з граничними умовами (крайові задачі)

Для задання граничних умов при інтегруванні ДР другого порядку використовується команда `bc2`. Вона задає значення функції у двох точках, які дозволяють розв'язати систему рівнянь відносно двох невідомих сталих інтегрування `%k1` і `%k2`.

▶ Розв'язати рівняння $y''+y\cdot(y')^3=0$, граничні умови $y(0)=1, y(1)=3$.

```
(%i1)  deqn1:'diff(y,x,2)+y*'diff(y,x)^3=0;
```

$$(\%o1)\quad \frac{d^2}{d\,x^2}\cdot y+y\cdot\left(\frac{d}{d\,x}\cdot y\right)^3=0$$

```
(%i2)  sol1:ode2(deqn1,y,x);
```

$$(\%o2)\quad \frac{y^3+6\cdot\%k1\cdot y}{6}=x+\%k2$$

```
(%i3)  bc2(sol1,x=0,y=1,x=1,y=3);
```

$$(\%o3)\quad \frac{y^3-10\cdot y}{6}=x-\frac{3}{2}$$

▶ Розв'язати рівняння $y''+y=x$, граничні умови $y(0)=0, y(4)=1$.

```
(%i4)  deqn2:'diff(y,x,2)+y=x;
```

$$(\%o4)\quad \frac{d^2}{d\,x^2}\cdot y+y=x$$

```
(%i5)  sol2:ode2(deqn2,y,x);
```

$$(\%o5)\quad y=\%k1\cdot\sin(x)+\%k2\cdot\cos(x)+x$$

```
(%i6)  bc2(sol2,x=0,y=0,x=4,y=1);
```

$$(\%o6)\quad y=x-\frac{3\cdot\sin(x)}{\sin(4)}$$

7.4 Пакет розширення `contrib_ode`

У системі **Maxima** є додатковий пакет `contrib_ode`, який дозволяє знаходити розв'язки нелінійних диференціальних рівнянь першого та другого порядку з розширеними можливостями. З допомогою `contrib_ode` можливо розв'язати рівняння Клеро, Лагранжа, Ріккаті та ін. В загальному випадку результат – список розв'язків. Для деяких

рівнянь (зокрема Ріккаті) розв'язок виводиться у формі іншого ЗДР – результату заміни змінних. Функция `contrib_ode` реалізує методи факторизації (factorization), Клеро (Clairault), Лагранжа (Lagrange), Ріккаті (Riccati), Абеля (Abel) та метод симетрії Лі (Lie symmetry method). Для його використання необхідно підключити пакет за допомогою команди `load(contrib_ode)`.

Синтаксис команди такий же, як і в `ode2`:

`contrib_ode(deqn,y,x)`.

Якщо диференціальне рівняння система може розв'язати, вона повертає розв'язок або список розв'язків, причому кожен з них може бути представлене в одному з таких видів:

– шукана функція виражена у явному вигляді;

– шукана функція виражена у неявному вигляді;

– розв'язок одержано у параметричному вигляді з параметром `%t`;

– диференціальне рівняння перетворюється на інше диференціальне рівняння відносно функції `%u`.

► Розв'язати рівняння $x(y')^2 - \dfrac{y'}{1+xy} + y = 0$.

```
(%i1)  load(contrib_ode);

(%i2)  deqn1:x*'diff(y,x)^2-(1+x*y)*diff(y,x)+y=0;
```

$$(\%o2) \quad x \cdot \left(\frac{d}{d\,x} \cdot y\right)^2 + y = 0$$

```
(%i3)  contrib_ode(deqn1,y,x);
```

$$(\%t3) \quad x \cdot \left(\frac{d}{d\,x} \cdot y\right)^2 + y = 0$$

first order equation not linear in y'

$$(\%o3) \quad [x^2 \cdot \left(y^2 + (2 \cdot x + 2 \cdot \%c) \cdot y + x^2 - 2 \cdot \%c \cdot x + \%c^2\right) = 0]$$

```
(%i4)  method;
```

(%o4) *lagrange*

► Розв'язати рівняння $(y')^2 + xy' - y = 0$.

```
(%i5)  deqn2:'diff(y,x)^2+x*'diff(y,x)-y=0;
```

$$(\%o5) \quad \left(\frac{d}{d\,x} \cdot y\right)^2 + x \cdot \left(\frac{d}{d\,x} \cdot y\right) - y = 0$$

```
(%i6)  contrib_ode(deqn2,y,x);
```

$$(\%t6) \quad \left(\frac{d}{d\,x} \cdot y\right)^2 + x \cdot \left(\frac{d}{d\,x} \cdot y\right) - y = 0$$

first order equation not linear in y'

$$(\%o6) \quad [y = \%c \cdot x + \%c^2, y = -\frac{x^2}{4}]$$

```
(%i7)  method;
```

(%o7) *clairault*

▶ Розв'язати рівняння $(1+y')x+(y')^2-y=0$.

```
(%i8)  deqn3:(1+'diff(y,x))*x+('diff(y,x))^2-y=0;
```

$$(\%o8)\quad \left(\frac{d}{d\,x}\cdot y\right)^2+x\cdot\left(\frac{d}{d\,x}\cdot y+1\right)-y=0$$

```
(%i9)  contrib_ode(deqn3,y,x);
```

$$(\%t9)\quad \left(\frac{d}{d\,x}\cdot y\right)^2+x\cdot\left(\frac{d}{d\,x}\cdot y+1\right)-y=0$$

first order equation not linear in y'

$$(\%o9)\quad [[x=e^{-\%t}\cdot\left(\%c-2\cdot(\%t-1)\cdot e^{\%t}\right),y=(\%t+1)\cdot x+\%t^2]]$$

```
(%i10) method;
```

(%o10) *lagrange*

В деяких випадках можливий розв'язок лише в параметричному вигляді.

▶ Розв'язати рівняння $(1+y')x+(y')^2-y=0$.

```
(%i11) deqn4:'diff(y,x)=(x+y)^2;
```

$$(\%o11)\quad \frac{d}{d\,x}\cdot y=(y+x)^2$$

```
(%i12) contrib_ode(deqn4,y,x);
```

$$(\%o12)\quad [[x=\%c-\operatorname{atan}\left(\sqrt{\%t}\right),y=-x-\sqrt{\%t}],[x=\operatorname{atan}\left(\sqrt{\%t}\right)+\%c,y=\sqrt{\%t}-x]]$$

```
(%i13) method;
```

(%o13) *lagrange*

Пакет `contrib_ode` дозволяє розв'язувати диференціальні рівняння, що не беруться за допомогою `ode2` безпосередньо, наприклад узагальнені однорідні рівняння. В такому випадку цей пакет використовує методи Абеля та симетрії Лі.

▶ Розв'язати рівняння $(2x-y+4)y'+(x-2y+5)=0$.

```
(%i14) deqn5:(2*x-y+4)*'diff(y,x)+(x-2*y+5)=0;
```

$$(\%o14)\quad (-y+2\cdot x+4)\cdot\left(\frac{d}{d\,x}\cdot y\right)-2\cdot y+x+5=0$$

```
(%i15) contrib_ode(deqn5,y,x);
```

$$(\%o15)\quad [\frac{\log\left(3-\frac{2\cdot(2\cdot x+4)-x-5}{-y+2\cdot x+4}\right)-3\cdot\log\left(1-\frac{2\cdot(2\cdot x+4)-x-5}{-y+2\cdot x+4}\right)+2\cdot\log\left(-\frac{2\cdot(2\cdot x+4)-x-5}{4\cdot(-y+2\cdot x+4)}\right)}{2}=$$
$$\log(x+1)+\%c]$$

```
(%i16) method;
```

(%o16) *abel2*

▶ Розв'язати рівняння $y'=\dfrac{1-3x-3y}{1+x+y}$.

```
(%i17) deqn6:'diff(y,x)=(1-3*x-3*y)/(1+x+y);
```

$$(\%o17)\quad \frac{d}{d\,x}\cdot y=\frac{-3\cdot y-3\cdot x+1}{y+x+1}$$

```
(%i18) contrib_ode(deqn6,y,x);
```

$$(\%o18) \quad [\frac{2\cdot\log\left(y+x-1\right)+y+3\cdot x}{2}=\%c]$$

```
(%i19) method;
```

$$(\%o19) \quad lie$$

7.5 Чисельні методи

В ряді випадків відшукати символьний розв'язок ДР в достатньо компактному вигляді неможливо, тому необхідно скористатись чисельними способами. Загалом спроба розв'язати диференціальні рівняння чисельними методами веде давню історію, протягом якої з'явились методи Ейлера (він же Рунге-Кутта 1-го порядку), Ейлера-Коші (він же Рунге-Кутта 2-го порядку), і власне Рунге-Кутта 4-го порядку. Останній вважається достатньо точним для отримання надійного розв'язку. Підвищення порядку традиційно нівелюється набіганням похибки обчислень, яка за розміром починає збігатись з різницевим членом, тому зупиняються на порядку 4. Є, однак, задачі специфічного вигляду, в яких на певних інтервалах інтегрування все ж доцільно використовувати 5-ий порядок розкладу. Принципова схема побудови такого розв'язку була розроблена Фельбергом, тому цей спосіб називається методом Рунге-Кутта-Фельберга 4-5 порядку.

Для побудови чисельного розв'язку диференціального рівняння першого порядку воно має бути записане у вигляді $\frac{dy}{dx}=f(x,y)$. У випадку автономної системи – має бути у вигляді $\begin{cases}\frac{dx}{dt}=G(x,y);\\ \frac{dy}{dt}=H(x,y).\end{cases}$

7.5.1 Метод Рунге-Кутта 4-го порядку `rk`

Maxima включає в себе пакет розширення `dynamics`, що дозволяє проінтегрувати системи ДР методом Рунге-Кутта. Окрім цього, пакет `dynamics` містить ряд функцій для побудови різноманітних фракталів.

Метод Рунге-Кутта 4-го порядку реалізує команда `rk`. Синтаксис виклику:

`rk(right_deqn,y,y0,[x,x0,x1,h])` – для одиночного рівняння;

`rk([right_deqn],[x,y],[x0,y0],[t,t0,t1,h])` – для автономної системи рівнянь.

Тут `right_deqn` – права частина рівняння або їх список, `y` або `[x,y]` – залежна змінна або їх список, `y0` або `[x0,y0]` – початкові значення залежної змінної, `[x,x0,x1,h]` або `[t,t0,t1,h]` – оголошення незалежної змінної, її початкове, кінцеве значення, та крок обчислень.

Для демонстрації методу розв'яжемо наступну задачу.

▶ Розв'язати задачу Коші для рівняння $y'=y-x,\quad y(0)=1.5$. Методом Рунге-Кутта знайти графічний розв'язок на відрізку $x\in[0,1]$, крок $h=0.1$.

Розв'яжемо спочатку це рівняння командою `ode2`:

```
(%i1) deqn1:'diff(y,x)=y-x;
```

$$(\%o1) \quad \frac{d}{d\,x}\cdot y=y-x$$

```
(%i3) ode2(deqn1,y,x); ic1(%,x=0,y=1.5);
```

$$(\%o2) \quad y=\left(\%c-(-x-1)\cdot e^{-x}\right)\cdot e^{x}$$

rat: replaced 0.5 by 1/2 = 0.5

(%o3) $y = \dfrac{e^x + 2\cdot x + 2}{2}$

```
(%i4) sol1:rhs(%o3);
```

(%o4) $\dfrac{e^x + 2\cdot x + 2}{2}$

Тепер скористаємось чисельним розв'язком.

```
(%i5) load(dynamics);
(%i6) sol2:rk(y-x,y,1.5,[x,0,1,0.1]);
```

(%o6) *list*

```
(%i7) plot2d([sol1,[discrete,sol2]],[x,0,1],[style,[lines,3,1],
        [points,3,2,1]],[legend,false]);
```

(%o7) $[C : /Users/orreg/maxout.gnuplot]$

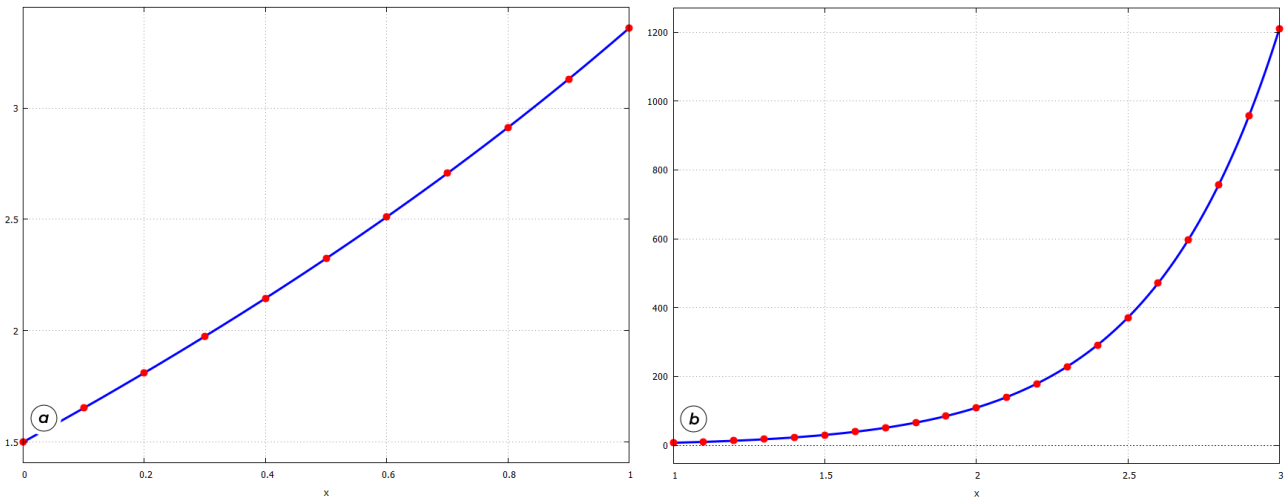

Рис. 7.1: Графіки прямого та чисельного розв'язків ДР за методом Рунге-Кутта.

Графіки розв'язків зображені на Рис. 7.1,(а). Бачимо, що всі точки чисельного розв'язку чітко лежать на кривій, що підкреслює точність обраного методу.

► Розглянемо рівняння, яке ми вже розв'язали раніше: $xy' = y(1+\ln y - \ln x)$, $y(1) = e^2$. Нагадаємо, що тоді ми отримали розв'язок $x = \dfrac{\ln y - \ln x}{2}$, яке зводиться до вигляду $y = xe^{2x}$. Знайдемо графічний розв'язок методом Рунге-Кутта на відрізку $x \in [1, 3]$, крок $h = 0.1$.

```
(%i8) rk6:rk((y/x)*(1+log(y)-log(x)),y,%e^2,[x,1,3,0.1]);
```

(%o8) *list*

```
(%i9) plot2d([x*%e^(2*x),[discrete,rk6]],[x,1,3],[style,[lines,3,1],
        [points,3,2,1]],[legend,false]);
```

(%o9) $[C : /Users/orreg/maxout.gnuplot]$

Графіки розв'язків зображені на Рис. 7.1,(б), і знову ми бачимо, що вони збігаються.

Розглянемо дещо складніший випадок.

► Розв'язати задачу Коші для рівняння $y' = \dfrac{1-3x-3y}{1+x+y}, y(-5) = 8$. Зобразити прямий та чисельний розв'язок цього рівняння методом Рунге-Кутта.

Спочатку розв'яжемо це рівняння з врахуванням початкових умов.

```
(%i1)  load(contrib_ode);
(%i2)  deqn6:'diff(y,x)=(1-3*x-3*y)/(1+x+y);
```

$$(\%o2) \quad \frac{d}{dx} \cdot y = \frac{-3 \cdot y - 3 \cdot x + 1}{y + x + 1}$$

```
(%i3)  contrib_ode(deqn6,y,x);
```

$$(\%o3) \quad [\frac{2 \cdot \log(y + x - 1) + y + 3 \cdot x}{2} = \%c]$$

```
(%i4)  ic1((%o3),x=-5,y=8);
```

$$(\%o4) \quad [\frac{2 \cdot \log(y + x - 1) + y + 3 \cdot x}{2} = \frac{2 \cdot \log(2) - 7}{2}]$$

Як бачимо, розв'язок ми отримали у вигляді співвідношення, яке відносно y розв'язати неможливо. Щоб накреслити графік, необхідно звернутись до команди для неявно заданих функцій.

```
(%i5)  load(implicit_plot);
(%i6)  implicit_plot(%o4,[x,-5,5],[y,-10,10], [nticks,200]);
```

rat: replaced 0.6931471805599453 by 13614799/19642003 = 0.693147180559946

(%o6) *done*

Результат зображений на Рис. 7.2(a).

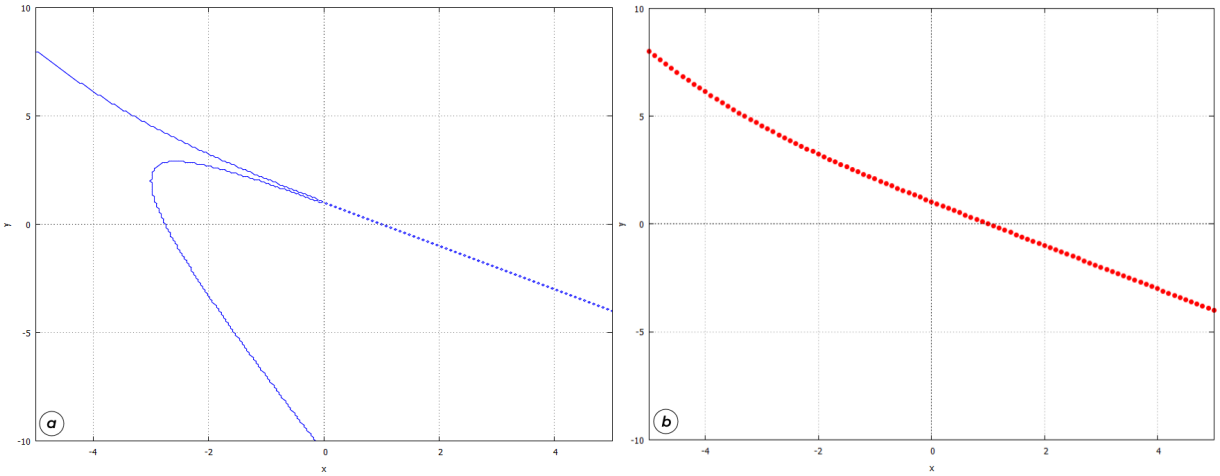

Рис. 7.2: Порівняння аналітичного та чисельного розв'язків.

Звернемось тепер до чисельного методу. Побудуємо інтегральну криву, обчислену за Рунге-Кутта.

```
(%i7)  load(dynamics);
(%i8)  sol2:rk((1-3*x-3*y)/(1+x+y),y,8,[x,-5,5,0.1]);
```

(%o9) *list*

```
(%i9)  plot2d([discrete,sol2],[x,-5,5],[y,-10,10],[style,
              [points,2,2,1]],[legend,false]);
```

(%o9) $[C : /Users/orreg/maxout.gnuplot]$

Результат зображений на Рис. 7.2(б). З порівняння графіків видно, що команда `rk` добре опрацювала верхню частину розв'язків, яка фактично збігається з аналітично визначеною, але нижня частина зовсім не опрацьована, її розв'язки «випали з уваги» Maxima.

▶ Розв'яжемо автономну систему рівнянь

$$\begin{cases} \frac{dx}{dt} = 4 - x^2 - 4y^2; \\ \frac{dy}{dt} = y^2 - x^2 + 1; \end{cases} \qquad t \in [0;4]; x(0) = -1,25; y(0) = 0,75.$$

Виберемо крок інтегрування 0,02.

```
(%i10) sol:rk([4*x^2-4*y^2,y^2-x^2+1],[x,y],[-1.25,0.75],[t,0,4,0.02]);
```

(%o8) *list*

Для побудови графіка перетворимо отриманий список, створивши окремо список значень t (`tlist`), x (`xlist`), y (`ylist`). Потім побудуємо разом два графіки: $x(t)$ та $y(t)$.

```
(%i11) tlist:makelist(sol[k][1],k,1,length(sol))$
       xlist:makelist(sol[k][2],k,1,length(sol))$
       ylist:makelist(sol[k][3],k,1,length(sol))$
(%i12) plot2d([[discrete,tlist,xlist],[discrete,tlist,ylist]],
              [legend,false]);
```

(%o12) $[C : /Users/orreg/maxout.gnuplot]$

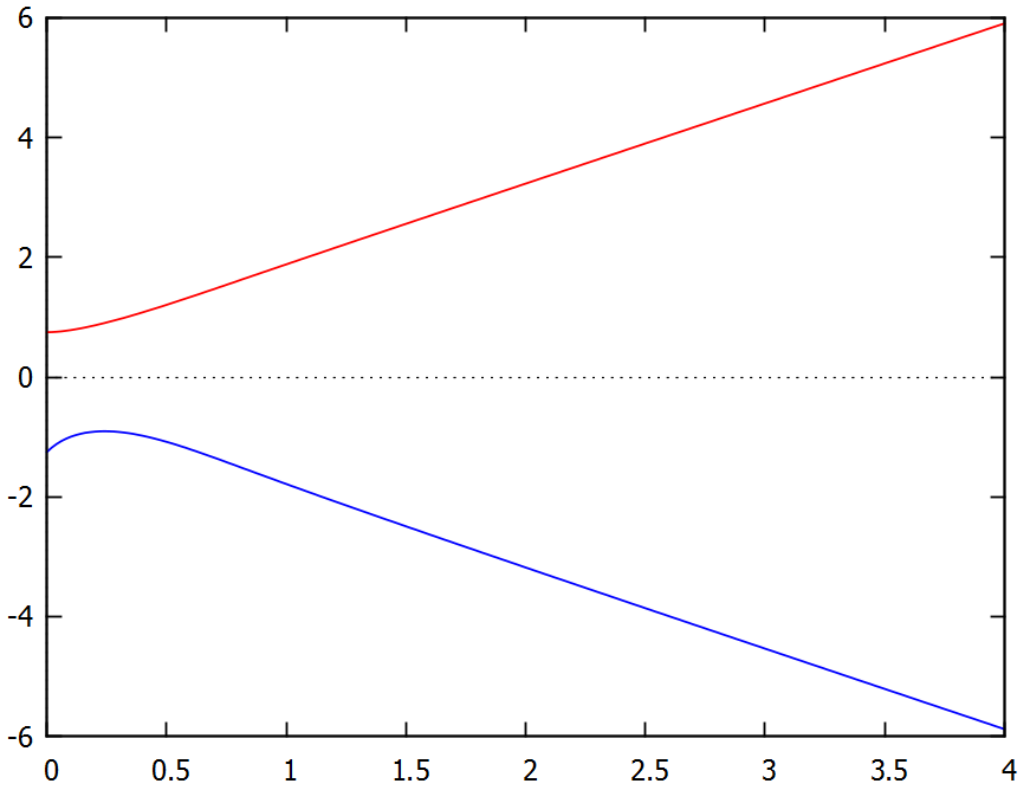

Рис. 7.3: Графіки розв'язку параметричної системи рівнянь.

Графік зображений на Рис. 7.3.

7.5.2 Метод Рунге-Кутта-Фельберга 4-5 порядку `rkf45`

Щоб скористатись цим методом, необхідно завантажити відповідний пакет командою `load(rkf45)`. Синтаксис наступний:

`rkf45(right_deqn,y,y0,[x,x0,x1],opts)` – для одиночного рівняння;

`rkf45([right_deqn],[x,y],[x0,y0],[t,t0,t1],opts)` – для системи рівнянь.

Тут `right_deqn` – права частина ДР або їх список (нагадаємо, що рівняння повинно бути записане у вигляді $\frac{dy}{dx} = f(x,y)$), `y` або `[x,y]` – залежна змінна або їх список, `y0` або `[x0,y0]` – початкові значення залежної змінної, `[x,x0,x1]` або `[t,t0,t1]` – оголошення незалежної змінної, її початкове та кінцеве значення, `opts` – додаткові опції.

Розглянемо опції `opts` докладніше. Їх значення записуються у форматі `opt=flag`.

`full_solution`: значення `true` повертає список всіх точок інтегрування, обраних алгоритмом. За замовчуванням стоїть у положенні `false` – тоді видаються лише фінальні результуючі точки.

`absolute_tolerance`: задає верхню межу абсолютної похибки, за замовчуванням стоїть значення 10^{-6}. Зменшення цього числа, наприклад заданням `absolute_tolerance=10e-8`, збільшить кількість отриманих точок інтегрування.

`max_iterations`: максимальне число ітерацій (за замовчуванням 10^4).

`h_start`: задає початковий крок інтегрування (за замовчуванням 1/100 від заданого інтервалу).

`report`: значення `true` задає повернення звіту про обчислення.

Як видно, на відміну від команди `rk`, у функції `rkf45` відсутнє задання кроку інтегрування. Розмір цього кроку обчислюється автоматично, в залежності від досягненняя заданої абсолютної похибки: це змінна величина, що безпосередньо відображається у різній кількості точок на різних проміжках заданого інтервалу, яка в свою чергу є наслідком звернення до точності 4-го чи 5-го порядку у схемі Рунге-Кутта-Фельберга.

Продемонструємо ці відмінності на прикладі.

▶ Розв'язати диференціальне рівняння безпосередньо, методом `rk`, та методом `rkf45`:

$$\frac{dy}{dx} = -3xy^2 + \frac{1}{x^3+1}; \quad y(0) = 0.$$

```
(%i1)  load(contrib_ode);
(%i2)  deqn2:'diff(y,x)=-3*x*y^2+1/(x^3+1);
```

$$(\%o2) \quad \frac{d}{d\,x} \cdot y = \frac{1}{x^3+1} - 3 \cdot x \cdot y^2$$

```
(%i4)  sol2:contrib_ode(deqn2,y,x)$ sol3:ic1(sol2,x=0,y=0);
```

$$(\%o4) \quad [y = \frac{x}{x^3+1}]$$

```
(%i5)  f:rhs(sol3[1]);
```

$$(\%o5) \quad \frac{x}{x^3+1}$$

Отримали точний розв'язок.

```
(%i6)  right_deqn2:rhs(deqn2);
```

$$(\%o6) \quad \frac{1}{x^3+1} - 3 \cdot x \cdot y^2$$

```
(%i7)  load(dynamics);
(%i8)  sol4:rk(right_deqn2,y,0,[x,0,5,0.1]);
```

$$(\%o8) \quad list$$

Отримали розв'язок за допомогою `rk`.

```
(%i9)  load(rkf45);
(%i10) sol5:rkf45(right_deqn2,y,0,[x,0,5]);
```

(%o10) *list*

Отримали розв'язок за допомогою `rkf45`.
Зобразимо тепер на точній інтегральній кривій ці результати (Рис. 7.4).

```
(%i11) plot2d([f,[discrete,sol4]],[x,0,5],[style,[lines,2,1],
              [points,3,2,1]],[legend,false]);
(%i12) plot2d([f,[discrete,sol5]],[x,0,5],[style,[lines,2,1],
              [points,3,2,1]],[legend,false]);
```

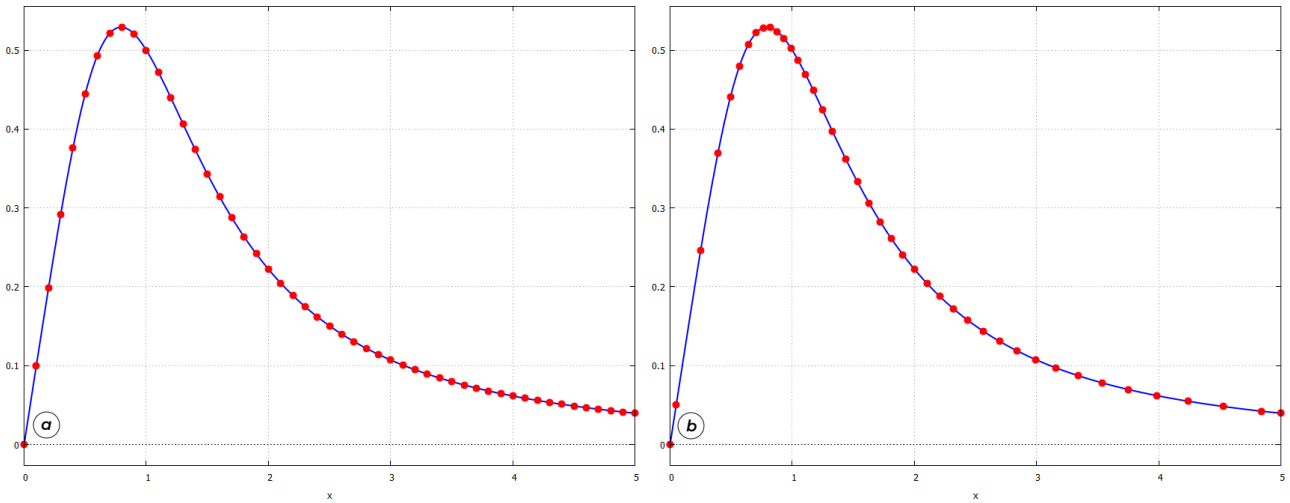

Рис. 7.4: Результати інтегрування ДР за методом `rk` (а) та `rkf45` (б).

З порівняння графіків чітко видно різницю між двома методами: у `rk` всі проміжки по змінній x між точками однакові, тоді як у `rkf45` присутня різна густина точок – поблизу екстремуму їх стає більше, тоді як на пологих місцях менше. Ця різниця відіграє велику роль при дослідженні так званих систем з пороговим ефектом – тобто таких кривих, у яких з'являється значний ступінчастий підйом при незначній зміні малого параметра.

▶ Розглянемо рівняння

$$\frac{dg}{dt} = s - 1.51g + 3.03\frac{g^2}{g^2+1}; \quad g(0) = 0; \quad t \in [0, 100].$$

Це математична модель біохімічного механізму, що зветься «генетичний перемикач». Здавалося б, у цьому рівнянні від сталого параметра s мало що залежить. Однак найменша зміна його від значення $s = 0.202$ до $s = 0.206$ докорінно змінює поведінку системи – з'являється пороговий ефект, який і лежить в природі цього «перемикача». Розв'яжемо задане рівняння з цими значеннями параметрів.

```
(%i1)  right_deqn2:0.202-1.51*g+3.03*(g^2)/(g^2+1);
```

(%o1) $$\frac{3.03 \cdot g^2}{g^2+1} - 1.51 \cdot g + 0.202$$

```
(%i2)  load(rkf45);
(%i3)  sol202:rkf45(right_deqn2,g,0,[t,0,100]);
```

(%o3) *list*

```
(%i4)  right_deqn3:0.206-1.51*g+3.03*(g^2)/(g^2+1);
```

$$(\%o4) \quad \frac{3.03 \cdot g^2}{g^2+1} - 1.51 \cdot g + 0.206$$

```
(%i5)  sol206:rkf45(right_deqn3,g,0,[t,0,100]);
```

$$(\%o5) \quad list$$

```
(%i6)  plot2d([[discrete,sol202],[discrete,sol206]],
              [t,0,100],[style,[lines,2,1],[lines,2,2]],
              [legend,"s=0.202","s=0.206"]);
```

$$(\%o6) \quad [C : /Users/orreg/maxout.gnuplot]$$

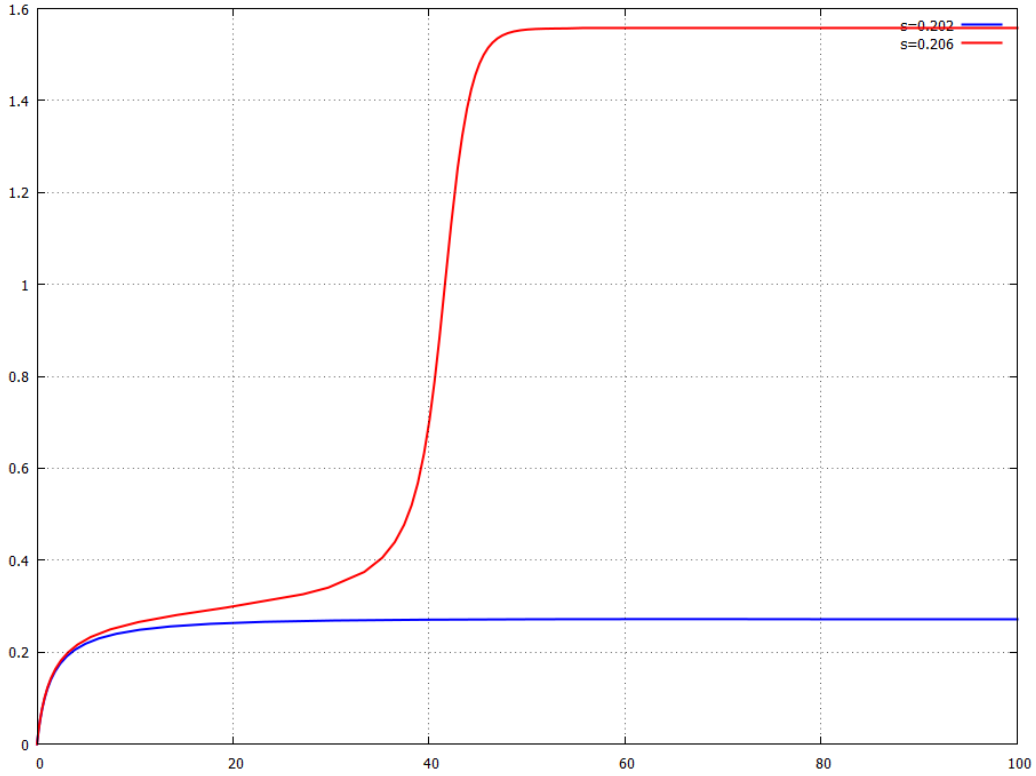

Рис. 7.5: Дослідження порогового ефекту.

При викликанні докладного звіту `report=true` система доповість, що у випадку $s = 0.206$ було змінено та переобчислено 14 поганих кроків (bad steps). Тобто на етапі обрахунку на порогових відрізках **Maxima** виявила, що порядку 4 методу Рунге-Кутта буде недостатньо для заданої точності, та переобрахувала ці точки, збільшивши їх кількість та відповідно звернувшись до 5-го порядку. Цю обставину видно навіть із кількості обчислених точок: для $s = 0.202$ їх 24, для $s = 0.206$ їх 62.

З огляду на таку розумну поведінку алгоритма можна констатувати, що метод `rkf45` є значно надійнішим при розв'язанні ДР чисельними методами, аніж метод `rk`.

Зобразимо наочно кількість та розміщення точок інтегрування обох методів для системи з пороговим ефектом (Рис. 7.6). Видно, що на «цікавий» інтервал сходинки `rk` витратив приблизно 10 точок, тоді як у `rkf45` на це пішло до 50 точок.

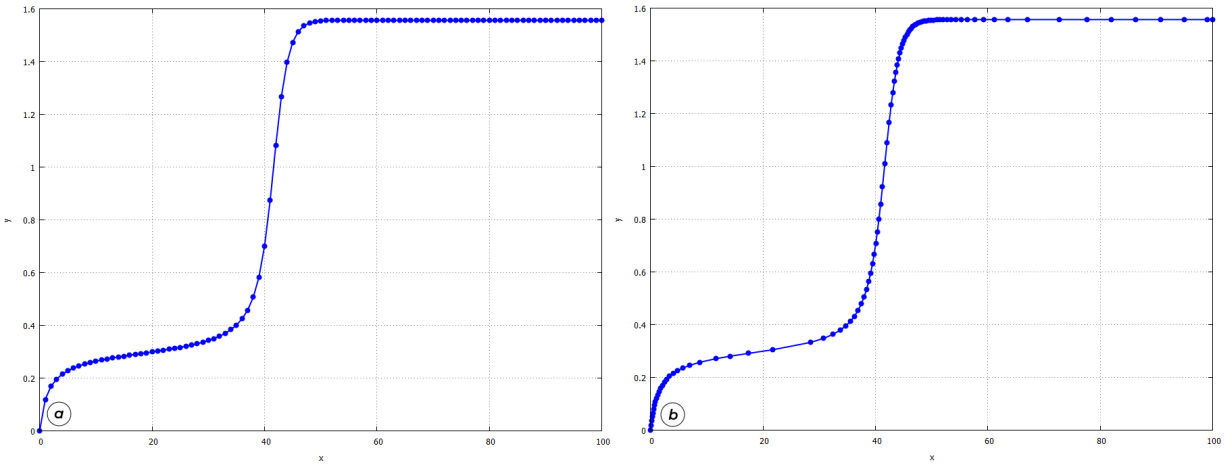

Рис. 7.6: Розміщення точок інтегрування для методу `rk` (а) та `rkf45` (б), загальна кількість точок однакова, $n = 100$.

7.6 Завдання до Розділу 7

Розв'язати задачу Коші для диференціальних рівнянь

7.1. $y'' - y' + x^2 = \sin x; \quad y(0) = 1, y'(0) = 4.$

7.2. $y'' - y' \cdot x^2 = 0; \quad y(0) = 1, y'(0) = 2.$

7.3. $xy'' - 0.3y' + x = 0; \quad y(0) = 2, y'(0) = -2.$

7.4. $y'' - 0.1y' + e^x y = 0; \quad y(0) = 4, y'(0) = 1.$

7.5. $y'' + 3y' - 4y = e^{-4x} + xe^{-x}; \quad y(0) = 6, y'(0) = 2.$

7.6. $y'' + y = x^3; \quad y(0) = 4, y'(0) = -1.$

7.7. $y'' - y \cdot \cos x = x^2; \quad y(0) = 1, y'(0) = 3.$

7.8. $y'' + y' + y = \sin^2 x; \quad y(0) = 0, y'(0) = 6.$

7.9. $y'' + 5y' - y = e^x; \quad y(0) = 1, y'(0) = 1.$

7.10. $y'' + (2x + 3)y' = 0; \quad y(0) = 2, y'(0) = 2.$

Розв'язати рівняння за методом Рунге-Кутта. Розв'язок представити графічно

7.11. $y' + xy \sin 5x = e^{0.1x^2}; \quad x \in [0, 4], y(0) = 1, h = 0.01.$

7.12. $y' + y^4 = 1 - x; \quad x \in [0, 1], y(0) = 0, h = 0.01.$

7.13. $y' + e^{0.1x} y = \sin xy; \quad x \in [0, 10], y(0) = 1, h = 0.01.$

7.14. $y' + xy \cos 3x = e^{0.3x}; \quad x \in [0, 4], y(0) = 2, h = 0.01.$

7.15. $y' + x^y = \sin 2x; \quad x \in [0, 5], y(0) = 1, h = 0.01.$

7.16. $y' + y^4 \ln(x^2 + 5) = 1 - x^5 y; \quad x \in [0, 2.5], y(0) = 3, h = 0.01.$

7.17. $y' \lg y^3 = y \sin(x^2); \quad x \in [-1, 2], y(-1) = 20, h = 0.01.$

7.18. $y' + x^4 \sqrt[4]{y^5} = y \cos(x + 5); \quad x \in [-1, 4], y(-1) = 15, h = 0.01.$

7.19. $y' + xy \cos(4x^2) = e^{0.1x}; \quad x \in [0, 4], y(0) = 1, h = 0.01.$

7.20. $y' + y\sqrt[3]{x^2} = \cos 2x \ln x; \quad x \in [1, 9], y(1) = 2, h = 0.01.$

Знайти розв'язок крайової задачі

7.21. $y'' - 3y' + 2y = x\cos x; \quad y(0) = 1, y(2) = 4.$

7.22. $y'' + 4y' + 3y = x^2 e^x; \quad y(1) = 4, y(4) = 10.$

7.23. $y'' + 5y' - y = x\cos x; \quad y(0) = 1, y(5) = 2.$

7.24. $y'' - 7y' + 12y = e^x \cos x; \quad y(0) = 6, y(4) = 10.$

Знайти розв'язок системи рівнянь

7.25. $\begin{cases} x'(t) = 2x + y; \\ y'(t) = 3x + 4y. \end{cases}$

7.26. $\begin{cases} x'(t) = x - y; \\ y'(t) = y - 4x. \end{cases}$

7.27. $\begin{cases} x'(t) = 8y - x; \\ y'(t) = x + y. \end{cases}$

7.28. $\begin{cases} x'(t) = y + 2e^t; \\ y'(t) = x + t^2. \end{cases}$

7.29. $\begin{cases} x'(t) = y - 5\cos t; \\ y'(t) = 2x + y. \end{cases}$

7.30. $\begin{cases} x'(t) = 3x + 2y + 4e^{5t}; \\ y'(t) = x + 2y. \end{cases}$

Розділ 8

Інтегральне числення

8.1 Невизначені інтеграли

Хоч процес знаходження інтегралу може розглядатись як протилежний до диференціювання, на практиці знаходження інтегралів є набагато складнішим за знаходження похідних. У **Maxima** за інтегрування відповідає основна команда `integrate`:

`integrate(f(x),x)` – шукає невизначений інтеграл за змінною x;

`integrate(f(x),x,a,b)` – шукає визначений інтеграл в межах $x \in [a,b]$.

Для цього оператора присутня також відкладена форма із апострофом `'integrate`, яка буває потрібна, коли підінтегральний вираз залежить від параметрів, що поки не обчислені. Значення параметрів може бути застосоване командою `ev(iexpr,a=A,b=B...)`. `integrate` може задавати питання для уточнення вигляду параметрів. Найперше він звертає увагу на функцію `assume` (припустити), якщо ж таких умов немає, то виникають питання, на які можна відповідати `yes` (так), `no` (ні), `pos` (додатній), `neg` (від'ємний), `zero` (нульовий).

► Знайти інтеграл $\int \sin^3 x dx$.

```
(%i1)  integrate(sin(x)^3,x);
```

$$(\%o1) \quad \frac{\cos(x)^3}{3} - \cos(x)$$

Як бачимо, **Maxima** виводить лише змістовну частину інтегрування, константу C необхідно додавати самостійно.

► Знайти інтеграл $\int x(b^2 - x^2)^{-1/2} dx$.

```
(%i1)  integrate(x/sqrt(b^2-x^2),x);
```

$$(\%o1) \quad -\sqrt{b^2 - x^2}$$

У програмі **Maxima** існує спеціальна команда, яка використовує формулу інтегрування частинами

$$\int u(x)v'(x)dx = u(x)v(x) - \int u'(x)v(x)dx.$$

Щоб використати її, необхідно завантажити додатковий пакет `load(antid)`. Синтаксис команди:

`antidiff(expr,x,v(x))` – тут `expr` підінтегральний вираз, x змінна інтегрування, $v(x)$ функція, яка звільняється від диференціювання.

```
(%i1)  load(antid);
(%i2)  A:u(x)*diff(v(x),x);
```

$$(\%o2) \quad \mathrm{u}(x) \cdot \left(\frac{d}{dx} \cdot \mathrm{v}(x) \right)$$

```
(%i3)  antidiff(A,x,v(x));
```

$$(\%o3)\quad \mathrm{u}\left(x\right)\cdot\mathrm{v}\left(x\right)-\int\mathrm{v}\left(x\right)\cdot\left(\frac{d}{d\,x}\cdot\mathrm{u}\left(x\right)\right)dx$$

```
(%i4)  expr1:exp(z(x))*diff(z(x),x)*sin(x);
```

$$(\%o4)\quad e^{\mathrm{z}\left(x\right)}\cdot\sin\left(x\right)\cdot\left(\frac{d}{d\,x}\cdot\mathrm{z}\left(x\right)\right)$$

```
(%i5)  antidiff(expr1,x,z(x));
```

$$(\%o5)\quad e^{\mathrm{z}\left(x\right)}\cdot\sin\left(x\right)-\int e^{\mathrm{z}\left(x\right)}\cdot\cos\left(x\right)dx$$

Ще одна корисна функція – заміна змінних у інтегральному виразі, вона здійснюється за допомогою команди

`changevar(iexpr,f(x,t),t,x)` – де `iexpr` інтеграл у відкладеному вигляді, $f(x,t)$ функція, яка пов'язує стару і нову змінні у форматі $f(x,t)=0$, t нова змінна, x стара змінна.

```
(%i6)  B:'integrate(x*exp(-x^2),x);
```

$$(\%o6)\quad \int x\cdot e^{-x^2}dx$$

```
(%i7)  changevar(B,x^2-z,z,x);
```

$$(\%o7)\quad \frac{\int e^{-z}dz}{2}$$

```
(%i8)  C:'integrate(sin(x)^5*cos(x),x);
```

$$(\%o8)\quad \int\cos\left(x\right)\cdot\sin\left(x\right)^5dx$$

```
(%i9)  changevar(C,sin(x)-t,t,x);
```

solve: using arc-trig functions to get a solution.
Some solutions will be lost.

$$(\%o9)\quad \int t^5dt$$

Інтегрування тригонометричних та логарифмічних виразів:

```
(%i1)  integrate(sin(x)*sin(2*x)*sin(3*x),x);
```

$$(\%o1)\quad \frac{\cos\left(6\cdot x\right)}{24}-\frac{\cos\left(4\cdot x\right)}{16}-\frac{\cos\left(2\cdot x\right)}{8}$$

```
(%i2)  integrate(1/cos(x)^3,x);
```

$$(\%o2)\quad \frac{\log\left(\sin\left(x\right)+1\right)}{4}-\frac{\log\left(\sin\left(x\right)-1\right)}{4}-\frac{\sin\left(x\right)}{2\cdot\sin\left(x\right)^2-2}$$

```
(%i3)  integrate(x^3*log(x),x);
```

$$(\%o3)\quad \frac{x^4\cdot\log\left(x\right)}{4}-\frac{x^4}{16}$$

При інтегруванні дробово-раціональних виразів першою метою є представлення виразу у вигляді елементарних дробів, які утворюються із знаменника розбиттям останнього на множники. Тут корисною є команда `partfrac`, яка здійснює таку операцію.

▶ Знайти інтеграл $\int -\dfrac{xdx}{x^3+4x^2+5x+2}$.

```
(%i4)  K:-x/(x^3+4*x^2+5*x+2);
```

$$(\%o4) \quad -\frac{x}{x^3+4\cdot x^2+5\cdot x+2}$$

```
(%i5)  partfrac(K,x);
```

$$(\%o5) \quad \frac{2}{x+2}-\frac{2}{x+1}+\frac{1}{(x+1)^2}$$

```
(%i6)  integrate(%o5,x);
```

$$(\%o6) \quad 2\cdot\log(x+2)-2\cdot\log(x+1)-\frac{1}{x+1}$$

```
(%i7)  integrate(K,x);
```

$$(\%o7) \quad 2\cdot\log(x+2)-2\cdot\log(x+1)-\frac{1}{x+1}$$

Якщо у знаменнику складний вираз вищих степенів, у нагоді для розкладу його на множники також стануть команди `gfactor`, `allroots` та подібні.

8.2 Визначені інтеграли

У синтаксисі виклику необхідно вказувати межі, виділені комами. Якщо інтеграл невласний, межі позначаються `inf` $(+\infty)$, `minf` $(-\infty)$.

▶ Знайти інтеграл $\int_0^{+\infty}\dfrac{dx}{1+x^2}$.

```
(%i1)  integrate(1/(1+x^2),x,0,inf);
```

$$(\%o1) \quad \frac{\pi}{2}$$

▶ Знайти інтеграл $\int_0^{4\pi} x^2\sin^3 xdx$.

```
(%i2)  integrate(sin(x)^3*x^2,x,0,4*%pi);
```

$$(\%o2) \quad -\frac{288\cdot\pi^2-40}{27}-\frac{40}{27}$$

8.2.1 Спецфункції інтегрування

При інтегруванні деяких функцій часто зустрічалися випадки, коли невизначений інтеграл неможливо знайти в явному вигляді, однак технічні та наукові потреби вимагали роботу саме з такими виразами. Такі інтеграли були добре вивчені, досліджені та протабульовані, а самі вирази отримали назву спеціальних інтегральних функцій.

Знайдемо у **Maxima** наступний інтеграл: $\int_{-\infty}^{+\infty} x^2e^{-x^2}dx$

```
(%i1)  integrate (x^2*exp(-x^2),x,minf,inf);
```

$$(\%o1) \quad \frac{\sqrt{\pi}}{2}$$

Тепер знайдемо такий же, але невизначений:

```
(%i2)  integrate (x^2*exp(-x^2),x);
```

$$(\%o2) \quad \frac{\sqrt{\pi}\cdot\operatorname{erf}(x)}{4}-\frac{x\cdot e^{-x^2}}{2}$$

Бачимо, що у виразі з'явилась функція `erf(x)` – це добре відомий «інтеграл помилок» Гаусса

$$\texttt{erf(t)} = \frac{2}{\sqrt{\pi}} \int_0^t e^{-z^2} dz.$$

Ще декілька спецфункцій, що можуть виникати при інтегруванні:

$$\texttt{gamma(z)} = \int_0^{+\infty} t^{z-1} e^{-t} dt \qquad \Gamma - \text{функція Ейлера;}$$

$$\texttt{gamma\_incomplete(a,z)} = \int_z^{+\infty} t^{a-1} e^{-t} dt \qquad \text{варіант } \Gamma - \text{функції Ейлера;}$$

$$\texttt{beta(r,s)} = \frac{\Gamma(r)\Gamma(s)}{\Gamma(r+s)} \qquad \text{B} - \text{функція;}$$

$$\texttt{zeta(s)} = \sum_{n=1}^{\infty} \frac{1}{n^s} \qquad \zeta - \text{функція Рімана;}$$

$$\texttt{elliptic\_f(phi,m)} = \int_0^{\phi} \frac{dx}{\sqrt{1 - m\sin^2 x}} \qquad \text{еліптичний інтеграл першого роду;}$$

$$\texttt{elliptic\_m(phi,m)} = \int_0^{\phi} \sqrt{1 - m\sin^2 x} dx \qquad \text{еліптичний інтеграл другого роду.}$$

Існує ще багато спецфункцій, які виникають при знаходженні інтегралів чи розв'язанні диференціальних рівнянь, зацікавленому читачеві варто звернутись до досить обширної літератури з цього питання.

8.2.2 Інтегрування з `defint` та `ldefint`

У знаходженні визначених інтегралів існують допоміжні команди `defint` та `ldefint`. Синтаксис задання аналогічний до `integrate`.

Команда `defint(f(x),x,a,b` – шукає інтеграл звичним для нас способом, тобто знаходить спочатку невизначений інтеграл, потім підставляє в нього межі (як не дивно, `integrate` діє іншим, відомим лише розробникам, чином).

Команда `ldefint(f(x),x,a,b` – шукає інтеграл за процедурою `defint`, але при підставлянні меж знаходить границі $\lim_{x\to a}$ та $\lim_{x\to b}$. Якщо проінтегрований вираз «гарний», то результат не буде відрізнятись від `defint` чи `integrate`, однак при розбіжному інтегралі чи необмежених границях виведення результату буде в іншому вигляді. Коли для розбіжного інтегралу `integrate` просто проконстатує цей факт, `ldefint` покаже корисну інформацію, яким саме чином інтеграл буде розбігатись.

► Знайти інтеграл $\int_0^{+\infty} x^{-3} dx$.

```
(%i3)  integrate(1/x^3,x,0,inf);
```

defint: integral is divergent.

– an error. To debug this try: debugmode(true);

```
(%i4)  ldefint(1/x^3,x,0,inf);
```

$$(\%o4) \qquad \frac{\lim_{x\to 0} \frac{1}{x^2}}{2}$$

► Знайти інтеграл $\int_0^{+\infty} \ln x dx$.

```
(%i5)  integrate(log(x),x,0,inf);
```

defint: integral is divergent.

– an error. To debug this try: debugmode(true);

```
(%i6)  ldefint(log(x),x,0,inf);
```

(%o6) $\lim_{x\to\infty} x\cdot\log(x)-x$

8.2.3 Інтеграли, що залежать від параметра

Задамо **Maxima** наступний інтеграл: $\int_{-\infty}^{+\infty}\frac{dx}{b^2+x^2}$.

```
(%i1)  integrate(1/(b^2+x^2),x,0,inf);
```

Is b zero or nonzero? nonzero;
Is b positive or negative? pos;

(%o1) $\frac{\pi}{2\cdot b}$

Як бачимо, нам довелось давати відповіді на питання системи щодо вигляду параметра b, оскільки від цього залежить результат виведення. Наприклад, інтеграл $\int_0^{+\infty} e^{-ax}dx$ при $a>0$ збіжний, при $a\le 0$ розбіжний.

```
(%i2)  integrate(%e^(-a*x),x,0,inf);
```

Is a positive, negative or zero? pos;

(%o2) $\frac{1}{a}$

```
(%i3)  integrate(%e^(-a*x),x,0,inf);
```

Is a positive, negative or zero? neg;
Is `e^a-1` positive, negative or zero? neg;
defint: integral is divergent.

– an error. To debug this try: debugmode(true);

Щоб не відповідати на додаткові питання, на параметри можна накласти умови. Це здійснюється командою `assume(ineq)`, де `ineq` – відповідна нерівність.

```
(%i1)  assume(a>1);
```

(%o1) $[a>1]$

```
(%i4)  integrate(x^a/(x+1)^(5/2),x,0,inf);
```

Is a an integer? no;
Is `2a-1` positive, negative or zero? neg;

(%o4) $\text{beta}\left(\frac{3}{2}-a,a+1\right)$

Скасувати умову дозволяє команда `forget(ineq)`.

8.3 Площа між двома кривими

Площа поверхні між двома кривими $f(x)$ та $g(x)$: $S = \int_a^b (f(x) - g(x))dx$, де a та b – точки їх перетину. Щоб знайти ці точки, потрібно розв'язати рівняння $f(x) = g(x)$. Вимога додатності площі зумовлює те, що $f(x)$ має знаходитись вище за $g(x)$, і за цією обставиною необхідно уважно слідкувати, оскільки на проміжку $[a, b]$ точок перетину може бути кілька. В задачах такого типу найперше потрібно зобразити графіки кривих.

► Знайти площу між кривими $f(x) = \sqrt{x}$ та $g(x) = x^{3/2}$.

Зобразимо ці криві (Рис. 8.1). Видно, що $f(x)$ знаходиться вище за $g(x)$.

```
(%i1)  plot2d([sqrt(x),x^(3/2)], [x,0,1.2],[style,[lines,2,1],
                [lines,2,2]],[plot_format, gnuplot]);
```

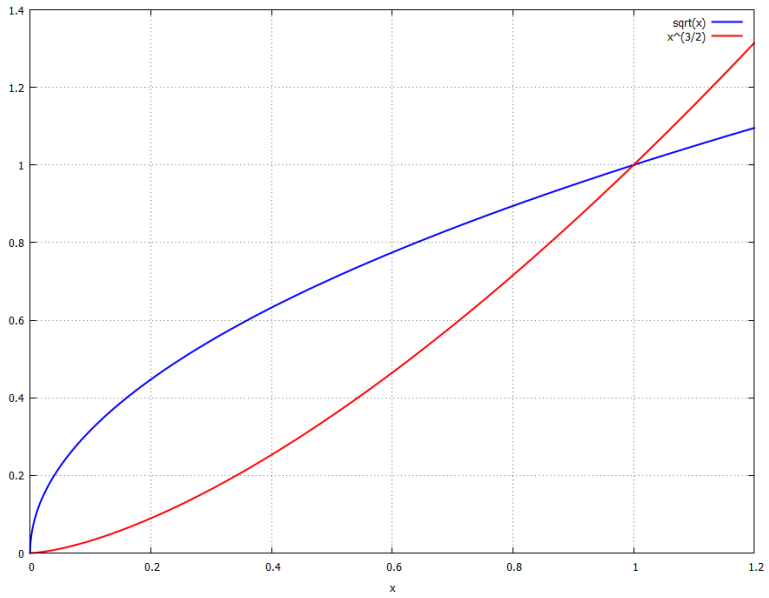

Рис. 8.1: Графіки кривих $f(x)$ та $g(x)$.

Точки перетину:

```
(%i2)  f(x):=sqrt(x);
```

(%o2) $\mathrm{f}(x) := \sqrt{x}$

```
(%i3)  g(x):=x^(3/2);
```

(%o3) $\mathrm{g}(x) := x^{\frac{3}{2}}$

```
(%i4)  solve(f(x)=g(x),x);
```

(%o4) $[x = 0, x = 1]$

Площа поверхні:

```
(%i5)  integrate(f(x)-g(x),x,0,1);
```

(%o5) $\frac{4}{15}$

► Знайти площу між кривими $f(x) = \frac{3}{10}x^5 - 3x^4 + 11x^3 - 18x^2 + 12x + 1$ та $g(x) = -4x^3 + 28x^2 - 56x + 32$.

Зобразимо ці криві (Рис. 8.2). Видно, що $f(x)$ та $g(x)$ мають дві точки перетину, і положення однієї кривої відносно іншої змінюється.

```
(%i1)  f(x):=(3/10)*x^5-3*x^4+11*x^3-18*x^2+12*x+1;
```

$$(\%o1) \quad \mathrm{f}(x) := \frac{3 \cdot x^5}{10} - 3 \cdot x^4 + 11 \cdot x^3 - 18 \cdot x^2 + 12 \cdot x + 1$$

```
(%i2)  g(x):=-4*x^3+28*x^2-56*x+32;
```

$$(\%o2) \quad \mathrm{g}(x) := -4 \cdot x^3 + 28 \cdot x^2 - 56 \cdot x + 32$$

```
(%i3)  plot2d([f(x),g(x)],[x,-1,5],[y,-10,10],
              [style,[lines,2,1],[lines,2,2]],
              [legend,"f(x)","g(x)"],[axes,solid]);
```

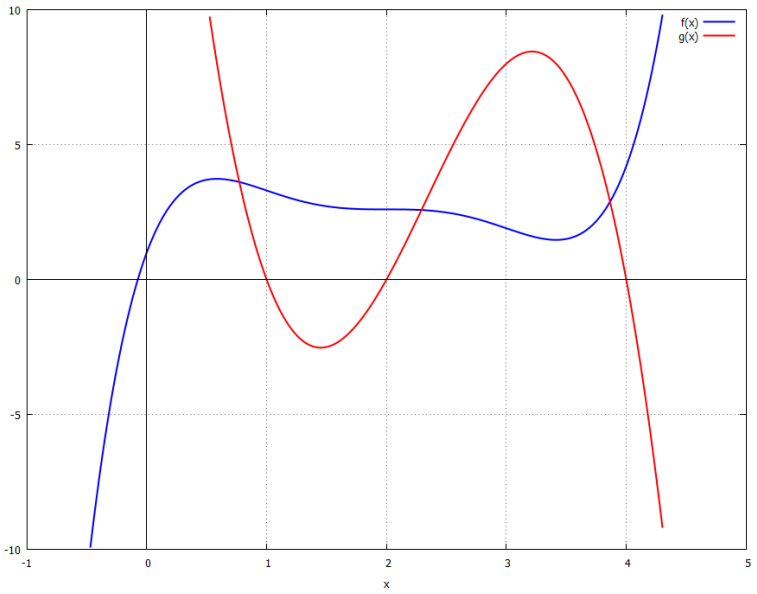

Рис. 8.2: Площа між кривими $f(x)$ та $g(x)$.

Щоб знайти точки перетину, розв'яжемо рівняння $f(x) = g(x)$.

```
(%i4)  solve(f(x)=g(x),x);
```

$$(\%o4) \quad [0 = 3 \cdot x^5 - 30 \cdot x^4 + 150 \cdot x^3 - 460 \cdot x^2 + 680 \cdot x - 310]$$

```
(%i5)  h(x):=3*x^5-30*x^4+150*x^3-460*x^2+680*x-310;
```

$$(\%o5) \quad \mathrm{h}(x) := 3 \cdot x^5 - 30 \cdot x^4 + 150 \cdot x^3 - 460 \cdot x^2 + 680 \cdot x - 310$$

Як бачимо, утворилось рівняння 5-го степеня, яке нерозв'язне в радикалах. Але **Maxima** дозволяє знайти чисельні розв'язки. Із графіка видно, що точки перетину містяться в проміжках $[0, 1]$, $[2, 3]$, та $[3, 4]$. Ця інформація дозволяє нам застосувати команду `find_root`.

```
(%i6)  x1:find_root(h(x),x,0,1);
```

$$(\%o6) \quad 0.772058304527811$$

```
(%i7)  x2:find_root(h(x),x,2,3);
```

$$(\%o7) \quad 2.291819210962955$$

```
(%i8)  x3:find_root(h(x),x,3,4);
```

(%o8) 3.865127100061793

```
(%i9)  [y1,y2,y3]:map(f,[x1,x2,x3]);
```

(%o9) [3.613992056691179, 2.575784006305817, 2.882949345140559]

Тепер, нарешті, можемо обчислити площу S, попередньо виконавши команду `ratprint:false`, щоб **Maxima** не намагалась перетворити вирази з плаваючою комою у раціональні:

```
(%i10) ratprint:false;
```

(%o10) false

```
(%i11) S:integrate(f(x)-g(x),x,x1,x2)+integrate(g(x)-f(x),x,x2,x3);
```

(%o11) $$\frac{169540440064218273521}{13902320084620535442}$$

```
(%i12) S,numer;
```

(%o12) 12.19511844298367

8.4 Повторні інтеграли

У **Maxima** можливо знайти повторні інтеграли якої завгодно розмірності, використовуючи вкладення команди `integrate` одна в одну. Звісно, в першу чергу нас будуть цікавити подвійні та потрійні інтеграли. Припустимо, необхідно знайти інтеграл по області V:

$$\iiint_V F(x,y,z)dV = \int_{x_1}^{x_2}\left[\int_{y_1(x)}^{y_2(x)}\left(\int_{z_1(x,y)}^{z_2(x,y)} F(x,y,z)dz\right)dy\right]dx.$$

На мові **Maxima** реалізація такого інтегралу наступна:
`integrate(integrate(integrate(F(x,y,z),z,z1(x,y),z2(x,y)),y,y1(x),y2(x)),x,x1,x2)`.

Дослідниками також розроблені додаткові пакети для обчислень криволінійних та поверхневих інтегралів різних видів, зокрема `Math214` (докладніше з цим можна ознайомитись на сайті The Maximalist). Можливості цього пакету досить широкі: знаходження площ чи об'ємів складних поверхонь, кривизни та довжини різноманітних кривих, операції векторного аналізу з використанням теорем Гаусса та Стокса.

► Знайти інтеграл
$\iiint_V (x^2+2y^2+3z^2)dV$, де $V:\{-\sqrt{x^2+y^2}\le z\le\sqrt{x^2+y^2};\ -x\le y\le x;\ 1\le x\le 2\}$.

```
(%i1)  integrate(integrate(integrate(x^2+2*y^2+3*z^2,z,
                                     -sqrt(x^2+y^2),sqrt(x^2+y^2)),
                                     y,-(x),(x)),x,1,2);
```

Is x positive, negative or zero? pos;

(%o1) $$\frac{31\cdot\left(5\cdot\operatorname{asinh}(1)+17\cdot\sqrt{2}\right)}{10}$$

8.4.1 Площа еліпса

Рівняння еліпса має вигляд $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$. Покладемо $a>b$ та виразимо x через y у першому квадранті: $x=\frac{a}{b}\sqrt{b^2-y^2}$. Тоді чверть площі еліпса має вигляд

$$\frac{S}{4}=\iint_D dxdy=\int_0^b\left(\int_0^{\frac{a}{b}\sqrt{b^2-y^2}}dx\right)dy.$$

Переведемо задачу на мову **Maxima**, використавши вкладення одного інтегралу в інший для реалізації повторного інтегралу:

```
(%i13) s1:solve((x/a)^2+(y/b)^2=1,x);
```

$$(\%o13)\quad [x=-\frac{a\cdot\sqrt{b^2-y^2}}{b},x=\frac{a\cdot\sqrt{b^2-y^2}}{b}]$$

```
(%i14) x1:rhs(s1[2]);
```

$$(\%o14)\quad \frac{a\cdot\sqrt{b^2-y^2}}{b}$$

```
(%i15) integrate(integrate(1,x,0,x1),y,0,b);
```

$$(\%o15)\quad \frac{\pi\cdot a\cdot b}{4}$$

Звідси загальна площа еліпса $S=\pi ab$.

8.4.2 Момент інерції еліпсоїда

Нехай маємо однорідний еліпсоїд обертання з півосями a,b,c:

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1;\quad -a\le x\le a;\quad -b\le y\le b;\quad -c\le z\le c.$$

Кожен елемент об'єму $dV=dxdydz$ має масу $dm=\rho dV$. Загальна маса об'єкту m, загальний об'єм $V=\frac{4}{3}\pi abc$. Обчислимо момент інерції еліпсоїда відносно осі z, що шукається за формулою

$$I_z=\iiint_V(x^2+y^2)dm=\rho\iiint_V(x^2+y^2)dxdydz.$$

Запишемо межі інтегрування, які знаходяться із проектування поверхні на відповідні осі та площини:

$$\begin{cases}-a\le x\le a;\\ -b\sqrt{1-(x/a)^2}\le y\le b\sqrt{1-(x/a)^2};\\ -c\sqrt{1-(x/a)^2-(x/b)^2}\le z\le c\sqrt{1-(x/a)^2-(x/b)^2}.\end{cases}$$

Тоді задача на мові **Maxima** виглядатиме наступним чином:

```
(%i1)  F(x,y,z):=x^2+y^2;
```

$$(\%o1)\quad \mathrm{F}\left(x,y,z\right):=x^2+y^2$$

```
(%i2)  V:(4/3)*%pi*a*b*c;
```

$$(\%o2)\quad \frac{4\cdot\pi\cdot a\cdot b\cdot c}{3}$$

```
(%i3)  %rho:m/V;
```

$$(\%o3)\quad \frac{3\cdot m}{4\cdot\pi\cdot a\cdot b\cdot c}$$

```
(%i4)  x1:-a;
```

$$(\%o4)\quad -a$$

```
(%i5)  x2:a;
```

$$(\%o5)\quad a$$

```
(%i6)  y1(x):=-b*sqrt(1-(x/a)^2);
```

$$(\%o6)\quad \mathrm{y1}\,(x) := -b\cdot\sqrt{1-\left(\frac{x}{a}\right)^2}$$

```
(%i7)  y2(x):=b*sqrt(1-(x/a)^2);
```

$$(\%o7)\quad \mathrm{y2}\,(x) := b\cdot\sqrt{1-\left(\frac{x}{a}\right)^2}$$

```
(%i8)  z1(x,y):=-c*sqrt(1-(x/a)^2-(y/b)^2);
```

$$(\%o8)\quad \mathrm{z1}\,(x,y) := -c\cdot\sqrt{1-\left(\frac{x}{a}\right)^2-\left(\frac{y}{b}\right)^2}$$

```
(%i9)  z2(x,y):=c*sqrt(1-(x/a)^2-(y/b)^2);
```

$$(\%o9)\quad \mathrm{z2}\,(x,y) := c\cdot\sqrt{1-\left(\frac{x}{a}\right)^2-\left(\frac{y}{b}\right)^2}$$

```
(%i10) Iz:%rho*integrate(integrate(integrate
       (F(x,y,z),z,z1(x,y),z2(x,y)),y,y1(x),y2(x)),x,x1,x2);
```

Is a positive, negative or zero? pos;
Is b positive, negative or zero? pos;
Is `(x-a)(x+a)` positive, negative or zero? neg;

$$(\%o10)\quad \frac{(8\cdot\pi\cdot a^5\cdot b^3+8\cdot\pi\cdot a^7\cdot b)\cdot m}{40\cdot\pi\cdot a^5\cdot b}$$

```
(%i11) rat(Iz);
```

$$(\%o11)/\mathrm{R}/\quad \frac{(b^2+a^2)\cdot m}{5}$$

Бачимо, що отриманий результат збігається із класичним, одержаним аналітично «на папері».

8.5 Чисельне інтегрування

За наявності складних випадків, або при інтегруванні узагальнених чи спеціальних функцій, **Maxima** не може відобразити результат інтегрування у вигляді елементарних функцій або числа. Тоді доводиться інтегрувати чисельними методами. **Maxima** містить два основних способи чисельного інтегрування: команди сімейства `quadpack` та метод `romberg`.

`quadpack` має довгу та шановану історію, інтегрування цим методом було написане на мові FORTRAN ще 1973 року. Згодом код був адаптований та включений до **Maxima**. Команди цього сімейства: `quad_qag`, `quad_qags`, `quad_qagi`, `quad_qawc`, `quad_qawf`, `quad_qawo`, `quad_qagp`. Кожна команда створена для цілком окремого випадку функцій з певними властивостями, тому вміле застосування їх приносить максимальний результат.

Ми розглянемо дві команди (з іншими читач може ознайомитись у Довідці **Maxima**):

`quad_qags(f(x),x,a,b,[opts])` – чисельно шукає інтеграл $f(x)$ на обмеженому проміжку $[a,b]$. Необов'язкові команди `[opts]` регулюють величину відносної помилки апроксимації (за замовчуванням 10^{-8}), число підінтервалів розбиття (за замовчуванням 200).

`quad_qagi(f(x),x,a,b,[opts])` – чисельно шукає інтеграл $f(x)$ на проміжку $[a,b]$, одне зі значень якого (a чи b) може бути нескінченність.

▶ Чисельно обчислити інтеграл $\int_0^3 xe^{\sqrt{x}}dx$.

```
(%i1)  quad_qags(x*exp(sqrt(x)),x,0,3);
```

(%o1) $[18.6521959729504, 1.474278886087793 \cdot 10^{-7}, 147, 0]$

У відповіді з'явився список з чотирьох елементів: власне саме значення інтегралу, відносна похибка обчислення, число проміжних інтегральних значень, код помилки (0 означає проблем в інтегруванні не було).

Цікаво порівняти результат з командою `integrate`:

```
(%i2)  integrate(x*exp(sqrt(x)),x,0,3),numer;
```

(%o2) 18.65219597319475

Різниця між двома значеннями $2.443591995415772 \cdot 10^{-10}$.

▶ Чисельно знайти інтеграл $\int_0^{+\infty} x^3e^{-3x}$.

```
(%i3)  quad_qagi (x^3*exp(-3*x), x, 0, inf);
```

(%o3) $[0.07407407407407407, 1.419030764062393 \cdot 10^{-10}, 105, 0]$

Порівняємо з `integrate`:

```
(%i4)  integrate(x^3*exp(-3*x), x, 0, inf);
```

(%o4) $\frac{2}{27}$

```
(%i5)  2/27,numer;
```

(%o5) 0.07407407407407407

Як бачимо, знову співпадіння двох значень дуже точне.

Ще одна команда для чисельного інтегрування, що використовує метод Ромберга: `romberg(f(x),x,a,b)`.

Точність цього способу задається глобальними змінними `rombergabs` – абсолютна похибка апроксимації, та `rombergtol` – відносна похибка апроксимації. Команда `rombergit` задає кількість поділів навпіл початкового інтервалу під час інтегрування, тобто якщо прийняти `rombergit:10`, кількість проміжків стає $2^{10} = 1024$.

▶ Знайти інтеграл за методом Ромберга $\int_3^{10} \left(\frac{1}{(x-1)^2+0.01} + \frac{1}{(x-2)^3+0.001}\right) dx$.

```
(%i25) f(x):=1/((x-1)^2+1/100)+1/((x-2)^3+1/1000);
```

(%o25) $\mathrm{f}\left(x\right) := \frac{1}{\left(x-1\right)^2+\frac{1}{100}} + \frac{1}{\left(x-2\right)^3+\frac{1}{1000}}$

```
(%i26) rombergtol:1e-6;
```
(%o26) $1.0 \cdot 10^{-6}$

```
(%i27) rombergit:10;
```
(%o27) 10

```
(%i28) int1:romberg(f(x),x,3,10);
```
(%o28) 0.8804650492843361

```
(%i31) int2:integrate(f(x),x,3,10),numer;
```
(%o31) 0.8804650492560029

```
(%i32) int1-int2;
```
(%o32) $2.833322465534138 \cdot 10^{-11}$

Точність цього методу теж дуже висока.

8.6 Завдання до Розділу 8

В поданих інтегралах здійснити заміну змінної, після чого обчислити інтеграл по новій змінній

8.1. $\int \frac{2}{x^2-2x-5}dx; \quad x-1 \to z$

8.2. $\int xe^{-\sqrt{x}}dx; \quad \sqrt{x} \to t$

8.3. $\int x^2 \ln x dx; \quad \ln x \to y$

8.4. $\int \sin^2 x \cos^2 x dx; \quad \sin x \to z$

8.5. $\int \frac{e^x}{x}dx; \quad e^x \to t$

8.6. $\int x^2 \sin\sqrt{x}dx; \quad \sqrt{x} \to y$

8.7. $\int \frac{dx}{5x^3-14x^2+13x-4}; \quad x-1 \to z$

8.8. $\int \frac{dx}{x\sqrt{\ln x}}; \quad 1/x \to t$

8.9. $\int \sqrt{x}e^{-x^2}dx; \quad \sqrt{x} \to y$

8.10. $\int \mathrm{sh}^2 x \,\mathrm{ch}^3 x dx; \quad \mathrm{ch}\, x \to z$

Обчислити площу між заданими кривими

8.11. $f(x)=\frac{2}{x}; \quad g(x)=\frac{4}{x^2}; \quad x \in [1,4]$

8.12. $f(x)=x^5; \quad g(x)=x^3;$

8.13. $f(x)=4\sin x;$
$g(x)=\frac{1}{20}(x+6)(x+2)(x-1)(x+5);$

8.14. $f(x)=x;$
$g(x)=x+\sin^2 x; \quad$ один цикл

8.15. $f(x)=5\cos x;$
$g(x)=x^4-x^3+2x; \quad x \in [-2,2]$

8.16. $f(x)=x^3-x-4; \quad g(x)=x^4-20;$

8.17. $f(x)=3\sin^3 x;$
$g(x)=8\cos^3 x; \quad$ один цикл

8.18. $x-y-1=0; \quad y^2=2x+1;$

8.19. $f(x)=e^{\sqrt{x}}; \quad g(x)=5\sin x;$

8.20. $f(x)=\cos\frac{1}{x}; \quad g(x)=x;$
остання замкнена фігура.

Обчислити чисельно інтеграли за методом `quad_pack` та методом `romberg`, порівняти їх значення

8.21. $\int_0^3 e^{-x^3}dx;$

8.22. $\int_1^2 \sin(x^3)dx;$

8.23. $\int_2^5 \frac{\sin x}{x}dx;$

8.24. $\int_0^1 \sin(\cos x)dx;$

8.25. $\int_0^4 \sqrt{1+\sin^2 x}dx;$

8.26. $\int_1^5 \frac{1}{\sqrt{1+\sin^2 x}}dx;$

8.27. $\int_1^3 \sin\sqrt[3]{x}dx;$

8.28. $\int_1^{10} \sin(\ln x)dx;$

8.29. $\int_1^7 \frac{1}{x^5-3x^3+22}dx;$

8.30. $\int_1^2 \frac{x}{\cos x}dx;$

Бібліоґрафія

[1] Офіційна сторінка документації **wxMaxima**:
https://maxima.sourceforge.io/documentation.html

[2] Короткий вступ до **wxMaxima** українською мовою:
https://wxmaxima-developers.github.io/wxmaxima/wxmaxima.uk.pdf

[3] Maxima 5.44 Manual:
https://www.scribd.com/document/469269094/maxima-5-44-pdf

[4] Wilhelm Haager. Graphics with MAXIMA:
http://www.austromath.at/daten/maxima/zusatz/Graphics_with_Maxima.pdf

[5] Richard H. Rand. Introduction to Maxima:
https://maxima.sourceforge.io/docs/manual/intromax.pdf

[6] Paulo Ney de Souza, Richard J. Fateman, Joel Moses, Cliff Yapp. The Maxima Book:
https://maxima.sourceforge.io/docs/maximabook/maximabook-19-Sept-2004.pdf

[7] Roland Salz. CAS Maxima Workbook:
https://roland-salz.de/Maxima_Workbook.pdf

[8] Gilberto E. Urroz. Introduction to Maxima:
https://www2.palomar.edu/users/cchamberlin/Math%20205%20pages/Maxima/MaximaBook.pdf