Математические пакеты

Вычислительные методы в GNU Octave

Сучков Андрей Игоревич

Санкт-Петербургский государственный электротехнический университет "ЛЭТИ"

3 октября 2020 г.

- Численное решение уравнения $f(x) = 0$ проводят в два этапа:
  1. отделяют корни уравнения, т.е. находят достаточно тесные промежутки, в которых содержится только один корень;
  2. проводят уточнение отделённых корней, т.е. находят корни с заданной точностью.
- Большинство вычислительных методов знакомо из курса «Вычислительная математика».

- Всякое алгебраическое уравнение относительно x можно записать в виде

$$a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = 0, \quad a_0 \neq 0, x \geqslant 1$$

- Всякий полином можно задать в виде вектора. e.g.: $[a_n, a_{n-1}, \ldots a_1, a_0]$
- Произведение двух многочленов вычисляет функция `q = conv (p1, p2)`
- Частное и остаток от деления двух многочленов находит функция `[q, r] = deconv (p1, p2)`

- Выполнить разложение частного двух многочленов вида $P_1(x)/P_2(y)$ на простейшие рациональные дроби вида $a/(x-b)^k$ можно с помощью функции `[a, b, c, k] = residue  (p1, p2)`
- Однако, данная функция **не раскладывает** на дроби вида $\dfrac{Mx+N}{(x^2+px+q)^k}$, $p^2-4q<0$
- Вычислить значение многочлена в заданной точке можно с помощью функции `y = polyval (p, x)`

- Вычислить производную от многочлена позволяет функция `polyder (p)`
- Производную произведения двух векторов можно с помощью `polyder (p1, p2)`
- Вызов функции в общем виде `[q, r] = polyder (p1, p2)` приведёт к вычислению производной от частного
- Взять интеграл от многочлена позволяет функция `polyint (p, k)`

Нелинейные уравнения и системы

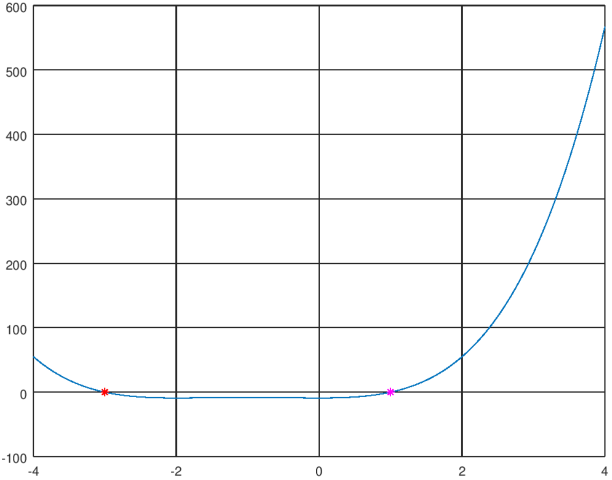
Вычисление корней в полиноме

Листинг 1: Файл froot.m

```
p = [1, 4, 4, 0, -9];
x = -4:0.01:4;
y = polyval (p, x);

plot (x, y), grid on

r = roots (p);
r = r(imag (r) == 0);

hold on
plot (r(1), 0, "r*", r(2), 0, "m*")

saveas (figure (1), "fig1.PNG")
```

Нелинейные уравнения и системы

Результат работы программы



Решение трансцендентных уравнений

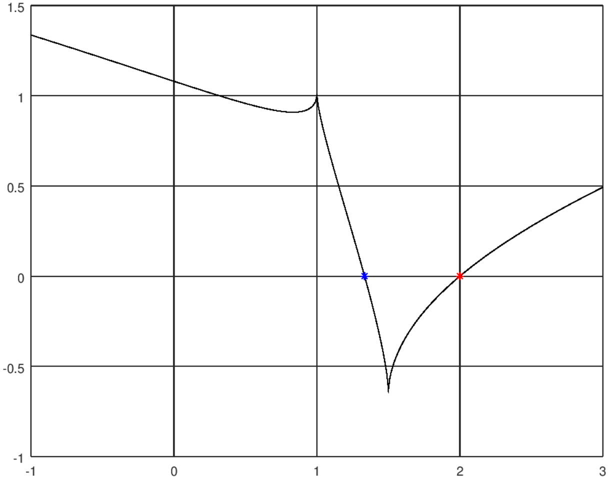
Функция fzero

Листинг 2: Файл fzero_eg.m

```
## script file
1;

function y = f (x)
  y = cbrt ((2 * x - 3) .^ 2) - cbrt ((x - 1) .^ 2);
endfunction

x = -1:0.01:3;
plot (x, f (x), "k"), grid on
x1 = fzero (@f, 1.5);   # return 1.3333
x2 = fzero (@f, [1.5, 2.5]);  # return 2

hold on
plot (x1, 0, "b*", x2, 0, "r*")

saveas (figure (1), "fig2.PNG")
```

Нелинейные уравнения и системы

Результат работы программы



Решение нелинейных систем уравнений

Функция fsolve

Листинг 3: Файл fsolve_eg1.m

```
## script file

f1 = @(x) sqrt (x .^ 2 - 3);
f2 = @(x) -sqrt (x .^ 2 - 3);
f3 = @(x) 1 - cos (x) / 2;
x1 = -5:0.001:-sqrt(3);
x2 = sqrt(3):0.001:5;
x3 = -5:0.001:5;

plot (x1, f1 (x1), "b",
      x1, f2 (x1), "b",
      x2, f1 (x2), "b",
      x2, f2 (x2), "b",
      x3, f3 (x3), "r"), grid on
```

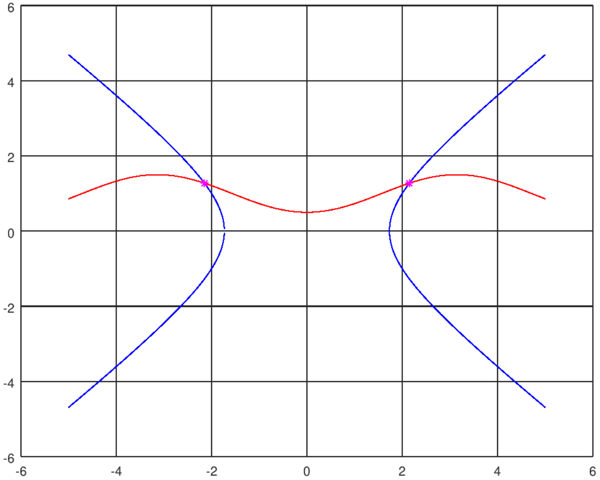
Решение нелинейных систем уравнений

Функция fsolve

Листинг 4: Файл fsolve_eg1.m (продолжение)

```
function y = g (x)
  y(1) = cos (x(1)) + 2 * x(2) - 2;
  y(2) = x(1) ^ 2 / 3 - x(2) ^ 2 / 3 - 1;
endfunction

x1_y1 = fsolve ("g", [-3, -1]);  # return (-2.1499, 1.2736)
x2_y2 = fsolve ("g", [1, 3] );  # return (2.1499, 1.2736)

hold on
plot (x1_y1(1), x1_y1(2), "m*", x2_y2(1), x2_y2(2), "m*")

saveas (figure (1), "fig3.PNG")
```

Результат работы программы



Решение нелинейных систем уравнений

Пример неправильной работы `fsolve`

Листинг 5: Файл fsolve_eg2.m

```
## script file

phi = -2*pi:0.001:2*pi;
x2 = -2:0.001:2;
f2 = @(x) 1 - 2 * sin (x - 1);

plot (cos (phi), sin (phi), "b", x2, f2 (x2), "r")
grid on
```

Решение нелинейных систем уравнений

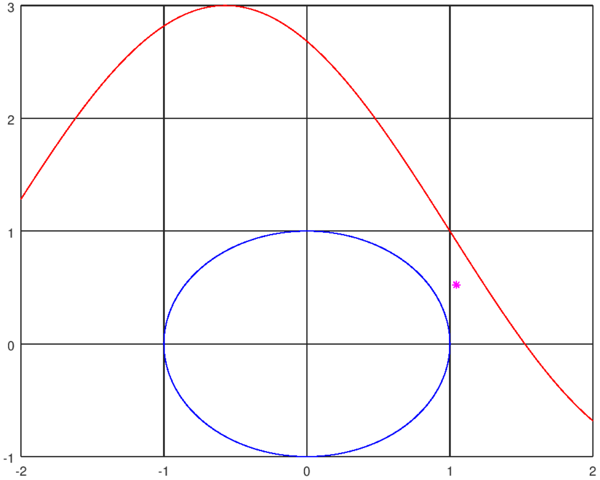
Пример неправильной работы fsolve

Листинг 6: Файл fsolve_eg2.m (продолжение)

```
function y = h (x)
  y(1) = x(1) ^ 2 + x(2) ^ 2 - 1;
  y(2) = 2 * sin (x(1) - 1) + x(2) - 1;
endfunction

x1_y1 = fsolve ("fun", [1, 1]);  # return (1.04584, 0.52342)

hold on
plot (x1_y1(1), x1_y1(2), "m*")

saveas (figure (1), "fig4.PNG")
```

Нелинейные уравнения и системы

Результат работы программы



Решение дифференциальных уравнений

Общие сведения

- Дифференциальное уравнение – уравнение, в которое входят производные функции, и может входить сама функция, независимая переменная и параметры
- В отличие от алгебраических уравнений, в результате решения которых ищется число (несколько чисел), при решении дифференциальных уравнений ищется функция (семейство функций)

Решение дифференциальных уравнений

Функция `lsode`

- Для решения дифференциальных уравнений вида существует функция `lsode`. Синтаксис: `y = lsode("fun", [y0, y1], x)`:
  - `"fun"` – имя функции
  - `[y0, y1]` – начальные условия
  - `x` – массив значений, для которых будет высчитываться решение уравнения

- Функция Струве – неэлементарная функция, которая является частным решением неоднородного уравнения Бесселя:

$$x^2\frac{d^2y}{dx^2}+x\frac{dy}{dx}+(x^2-\alpha^2)y=\frac{4(x/2)^{\alpha+1}}{\sqrt{\pi}\Gamma(\alpha+0.5)}$$

- Для решения данного уравнения введём новую переменную: $g=y'$. Тогда:

$$\begin{cases} y'=g \\ g'=\dfrac{(x/2)^{\alpha-1}}{\sqrt{\pi}\Gamma(\alpha+0.5)}-\dfrac{g}{x}-\left(1-\left(\dfrac{\alpha}{x}\right)^2\right)y \end{cases}$$

Решение дифференциальных уравнений

Функция Струве

Листинг 7: Функция struve.m

```
function struve (alpha = 1, y0 = 1, y1 = 0)
  x = 1:0.01:60;

  function v = df (f, x)
    y = f(1);
    g = f(2);
    C = (x / 2)^(alpha - 1) / (sqrt (pi)*gamma (alpha+0.5));
    v = [g, ... # y' = g
         C - g / x - (1 - (alpha / x) ^ 2) * y];  # g' = ...
  endfunction

  y = lsode ("df", [y0, y1], x);
  plot (x, y(:,1), "r;y = y(x);",
  x, y(:,2), "b;g = y'(x);"), grid on
  title ("Struve function")
  saveas (figure (1), "fig5.PNG")
endfunction
```

Решение дифференциальных уравнений

Результат работы программы