ДЗ 1. Решение нелинейных алгебраических уравнений

Лебедев Сергей, sergey.a.lebedev@gmail.com

22 февраля 2016 г.

Задание 1: Реализвать метод деления отрезка пополам и метод Ньютона. Найти решение уравнения $x^5 + x - \alpha$ методом деления отрезка пополам и методом Ньютона. Сравнить полученные решения.

Высокоуровневое описание метода деления отрезка пополам алг. 1.

Algorithm 1 Метод деления отрезка пополам

```
function Bisection(xLeft, xRight)
    N ← 0
    while N < N_MAX do
        midPoint ← (xLeft − xRight)/2
        if |f(midPoint)| < epsYorxRight − xLeft < epsY then
   return midPoint
        else
            N ← N + 1
            if sign(f(midPoint)) == sign(f(xLeft)) then
                xLeft ← midPoint
            else
                xRight ← midPoint
```

Высокоуровневое описание метода Ньютона алг. 2.

На рисунке 1 изображены функции решения уравнения $x^5 + x - \alpha = 0$ полученные с использованием метода деления отрезка пополам и метода Ньютона при различных значениях параметра α.

Algorithm 2 Метод Ньютона

function NEWTONS($xStart$)
  $N \leftarrow 0$
  $xOld \leftarrow xStart$
  while $N < N\_MAX$ **do**
    $xNew \leftarrow xOld - f(xOld)/\frac{df}{dt}(xOld)$
    if $|f(xNew)| < epsY or |xOld - xNew| < epsX$ **then**
   return $xNew$
    else
      $N \leftarrow N + 1$
      $xOld \leftarrow xNew$

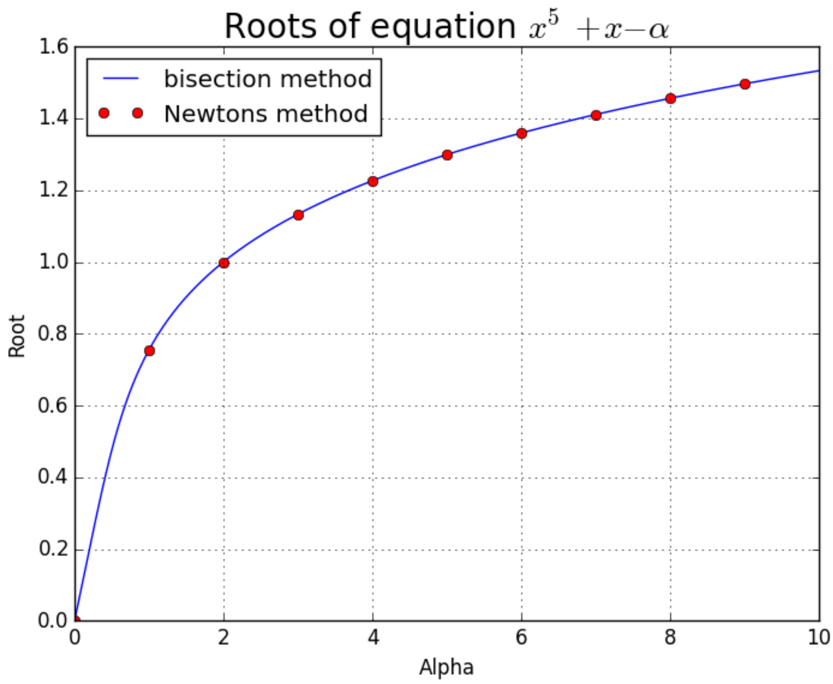

Рис. 1: Корни уравнения $x^5 + x - \alpha$ при различных значениях α

Задание 2: Найти решение уравнения $xk^2 + (x - k)(1 + x^2)^2 = 0$ методом Ньютона и сравнить его с аналитическим.

Уравнение

$$xk^2 + (x - k)(1 + x^2)^2 = 0 \tag{1}$$

может быть представлено в виде

$$(x^2 - kx + 1)(x^3 + x - k) = 0 \tag{2}$$

Таким образом корни исходного уравнения (1) могут быть найдены решением двух независимых уравнений (3, 4).

$$x^2 - kx + 1 = 0 \tag{3}$$

$$x^3 + x - k = 0 \tag{4}$$

Уравнение (3) квадратное, его корни могут быть записаны в виде (5)

$$x_{12} = \frac{k \pm \sqrt{k^2 - 4}}{2} \tag{5}$$

Единственное решение уравнения 4 может быть найдено с использованием формулы Кардано и записано в виде

$$x_3 = \sqrt[3]{\frac{k}{2} + \sqrt{\frac{k^2}{4} + \frac{1}{27}}} + \sqrt[3]{\frac{k}{2} - \sqrt{\frac{k^2}{4} + \frac{1}{27}}} \tag{6}$$

На рисунке 2 изображены функции решения уравнения 1 полученные аналитически и с использованием метода Ньютона при различных значениях параметра k

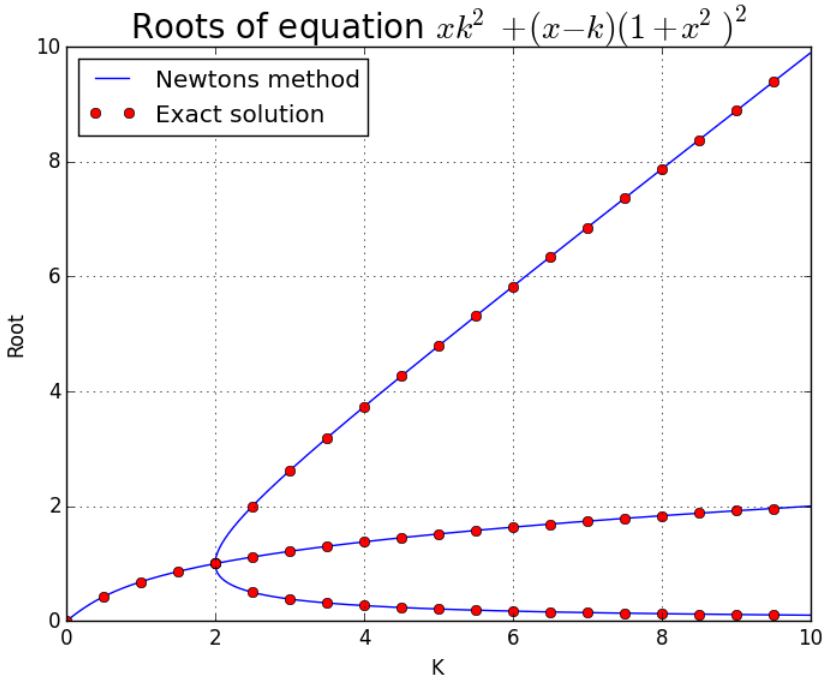

Рис. 2: Корни уравнения $xk^2 + (x - k)(1 + x^2)^2 = 0$ при различных значениях k

1 Приложени 1. Исходный код.

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.special import cbrt
def f1(x, alpha):
return x**5+x-alpha

def f1_der(x, alpha):
return 5*(x**4) + 1

def f2(x, k):
return x*k*k+(x-k)*((1+x*x)**2)
def f2_der(x, k):
return 5*(x**4) - 4*k*(x**3)+6*(x**2)-4*k*x+k**2+1

def f3(x, k):
return x**3+x-k

def f3_der(x, k):
return 3*x**2+1

eps_x = 0.00001
eps_y = 0.00001
N_MAX = 1000

def solution1(alpha):
p = np.sqrt(alpha*alpha/4.0 + 1.0/27.0)
return cbrt(alpha/2 +p)+cbrt(alpha/2-p)

def solution2(alpha):
d = np.sqrt(alpha**2 -4.0)
return (alpha+d)/2, (alpha-d)/2


def bisection(alpha, x_left, x_right):
N = 0
while N < N_MAX:
midpoint = (x_left + x_right) / 2
if abs(f1(midpoint, alpha)) < eps_y and (x_right-x_left) < eps_x:
print N, "iterations performed"
print "The answer is ", "(", midpoint, "," ,f1(midpoint,alpha), ")"
return midpoint
```

```
else:
N = N + 1
if np.sign(f1(midpoint, alpha)) == np.sign(f1(x_left, alpha)):
x_left = midpoint
else:
x_right = midpoint
print "Too many iterations"

def newtons(f, f_der, alpha, x_start):
N = 0
x_old = x_start
while N < N_MAX:
x_new = x_old - f(x_old, alpha)/f_der(x_old, alpha)
if abs(f(x_new,alpha)) < eps_y and abs(x_old - x_new) < eps_x:
print N, "iterations performed"
print "The answer is ", "(", x_new, "," ,f(x_new,alpha), ")"
return x_new
else:
N = N + 1
x_old = x_new
print "Too many iterations"

alpha = 1
leftBorder  = 0.0
rightBorder = 100.0
step = 0.1

alpha_bis = np.arange(0, 10, 0.01)
alpha_new = np.arange(0, 10, 1.0)

y1 = [bisection(a, leftBorder, rightBorder) for a in alpha_bis]
y2 = [newtons(f1,f1_der, a, 100) for a in alpha_new]


plt.plot(alpha_bis, y1, label="bisection method")
plt.plot(alpha_new, y2, 'ro', label="Newtons method")
plt.ylabel("Root")
plt.xlabel("Alpha")
plt.title(r"Roots of equation $x^5+x-\alpha$", fontsize=20)
plt.legend(loc='uppper left')
plt.grid(True)

plt.figure(2)
```

```
k_new_1 = np.arange(0, 10, 0.01)
k_new_2 = np.arange(2, 10, 0.01)
k_sol_1 = np.arange(0, 10, 0.5)
k_sol_2 = np.arange(2, 10, 0.5)
y3_1 = [newtons(f3,f3_der, k, 10) for k in k_new_1]
y3_2 = [newtons(f2,f2_der, k, -10) for k in k_new_2]
y3_3 = [newtons(f2,f2_der, k, 10) for k in k_new_2]

y4_1 = [solution1(k) for k in k_sol_1]
y4_2 = [solution2(k)[0] for k in k_sol_2]
y4_3 = [solution2(k)[1] for k in k_sol_2]

plt.plot(k_new_1, y3_1, label="Newtons method")
plt.plot(k_new_2, y3_2, 'b-')
plt.plot(k_new_2, y3_3, 'b-')
plt.plot(k_sol_1, y4_1,'ro', label="Exact solution")
plt.plot(k_sol_2, y4_2,'ro')
plt.plot(k_sol_2, y4_3,'ro')
plt.ylabel("Root")
plt.xlabel("K")
plt.title(r"Roots of equation $xk^2+(x-k)(1+x^2)^2$", fontsize=20)
plt.legend(loc='upper left')
plt.grid(True)

plt.show()
```