МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н. Э. БАУМАНА

Факультет информатики и систем управления

Кафедра теоретической информатики и компьютерных технологий

Лабораторная работа №2

по курсу «Современные вычислительные методы»

«Численное решение одномерной краевой задачи (локальные базисные функции)»

Выполнил:
студент ИУ9-111
Выборнов А. И.

Руководитель:
Басараб М.А.

Москва 2016

1. Постановка задачи

Необходимо найти приближенное решение краевой задачи на числовой оси $x \in [0, l]$.

$$\frac{d^2u}{dx^2} = f(x)$$

$$\begin{cases} \left(\alpha_1 \dfrac{du}{dx} + \beta_1 u\right)_{x=0} = \psi_1 \\ \left(\alpha_2 \dfrac{du}{dx} + \beta_2 u\right)_{x=l} = \psi_2 \end{cases}$$

Вариант 2, условия задачи:

$$u(x) = x^2 ln(x + l),$$

$$\alpha_1 = 1, \alpha_2 = 0,$$

$$\beta_1 = 0, \beta_2 = 1.$$

Необходимо решить задачу используя метод Бубнова-Галеркина и в качестве базиса использовать кубические B-сплайны.

2. Задача с однородными краевыми условиями

Вычислим производные исходной функции u:

$$u'(x) = 2xln(x + l) + \frac{x^2}{x + l}$$

$$u''(x) = 2ln(x + l) + \frac{4x}{x + l} + \frac{x^2}{(x + l)^2}$$

Подставим функцию u в систему граничных условий для получения значений на границах отрезка $[0, l]$:

$$\begin{cases} \psi_1 = 0 \\ \psi_2 = l^2 * ln(2l) \end{cases}$$

Аппроксимация значений функции u заключается в представлении функции в следующем виде:

$$u(x) = \varphi_0(x) + \sum_{k=1}^{n} a_k \varphi_k(x)$$

Где $\{\varphi_i\}, i = \overline{1, k}$ — система базисных векторов. Базисный вектор φ_0 можно представить в следующем виде:

$$\varphi_0(x) = A + Bx$$

Вычислим коэффициенты A, B:

$$\varphi_0(0) = A$$
$$\varphi_0(l) = A + B * l$$

Для этого, необходимо подставить φ_0 в начальные граничные условия, и получим:

$$\begin{cases} A = 0 \\ B = l * ln(2l) \end{cases}$$

На остальные базисные вектора накладываются следующие условия:

$$\begin{cases} \varphi_k|_{x=0} = 0 \\ \varphi_k|_{x=l} = 0 \\ k = \overline{1, k} \end{cases}$$

В качестве такого базиса рассмотрим следующие вектора:

$$\varphi_k = B_3(i, x)x(x - l)$$

Составим функцию y, которая будет иметь вид:

$$y(x) = u(x) - \varphi_0(x) = \sum_{k=1}^{n} a_k \varphi_k(x)$$

Далее решаем однородную краевую задачу для функции y.

3. Выражения для расчета коэффициентов СЛАУ

Согласно методу Галеркина строится СЛАУ следующим образом:

$$\begin{cases} a_{ij} = (L\varphi_i, \varphi_j)_{L^2}, \\ b_i = (f, \varphi_j)_{L^2}. \end{cases}$$

Вычисляем нормы и получаем:

$$\begin{cases} a_{ij} = \int_0^l (B_3(i,x)x(x-l))'' B_3(j,x)x(x-l)dx \\ b_i = \int_0^l (2ln(x+l) + \dfrac{4x}{x+l} - \dfrac{x^2}{(x+l)^2}) B_3(i,x)x(x-l)dx. \end{cases}$$

Решаем полученную СЛАУ и получаем искомый коэффициент c_i.

4. Исходный код программы на языке программирования python

```
import math
import numpy as np
import matplotlib.pyplot as plt
from scipy.misc import derivative

l = 1
k = 8
step = 1e-2

integral = lambda f: sum(f(x)*step for x in np.arange(0,l,step))

delta = l*1.0/k
base = [i*delta for i in range(k+1)]
t = [None] + [base[0]-delta] + base + [base[-1]+delta]

def B_in(i,n,x):
    if n == 1:
        return 1 if t[i] <= x and x < t[i+1] else 0
    else:
        return (x-t[i])*1.0 / (t[i+n-1]-t[i]) * B_in(i,n-1,x) + \
            (t[i+n]-x)*1.0 / (t[i+n]-t[i+1]) * B_in(i+1,n-1,x)

u = lambda x: x*x*math.log(x+l)
f = lambda x: 2*math.log(x+l) + 4*x*1.0/(x+l) - x**2*1.0/(x+l)**2

phi_0 = lambda x: l*math.log(2*l)*1.0*x
phi_i = lambda i: lambda x: B_in(i,3,x)*(x-l)*x
phi = [phi_0] + [phi_i(i) for i in range(1,k+1)]

a_ij = lambda i,j: integral(lambda x: derivative(phi_i(i), x, 1e-6,n =2) * phi_i(j)(x))
a = [[a_ij(i,j) for j in range(1, k+1)] for i in range(1,k+1)]
b_f = lambda x, i: f(x) * phi_i(i)(x)
b = lambda i: integral(lambda x: b_f(x, i))

a = np.array(a)
b  = np.array([b(i) for i in range(1, k+1)])
c = np.linalg.solve(a,b)
c = [1] + list(c)

um = lambda x: sum(c[i]*phi[i](x) for i in range(0,k+1))

N = sum((u(x) - um(x))**2 for x in np.arange(0,l,step))

print 'residual: ', N

def draw(f, label, draw_type):
    x = np.arange(0,l,step)
    y = map(f, x)
    plt.plot(x, y, draw_type,label=label)

draw(u, 'u', '')
draw(um, 'um', 'p')

plt.legend()
plt.show()

# for i in range(1, k+1):
#     X = np.arange(0,l,step)
#     Y = [phi_i(i)(x) for x in X]
#     plt.plot(X, Y,label=i)
# plt.legend()
# plt.show()
```

5. Результат применения метода

Для решения использовалась вышеприведённая программа на python. В качестве правой границы, коэффициент l рассматривается равным 1. Увеличивая количество базисных векторов, можно получить повышение качества аппроксимации. Анализировалась следующая мера погрешности:

$$\Delta = \sum_{i=0}^{points} (u(x) - u_m(x))^2$$

При длине шага равной 10^{-2} варьировалось количество базисных векторов, полученный результат изображён на рисунке 1.

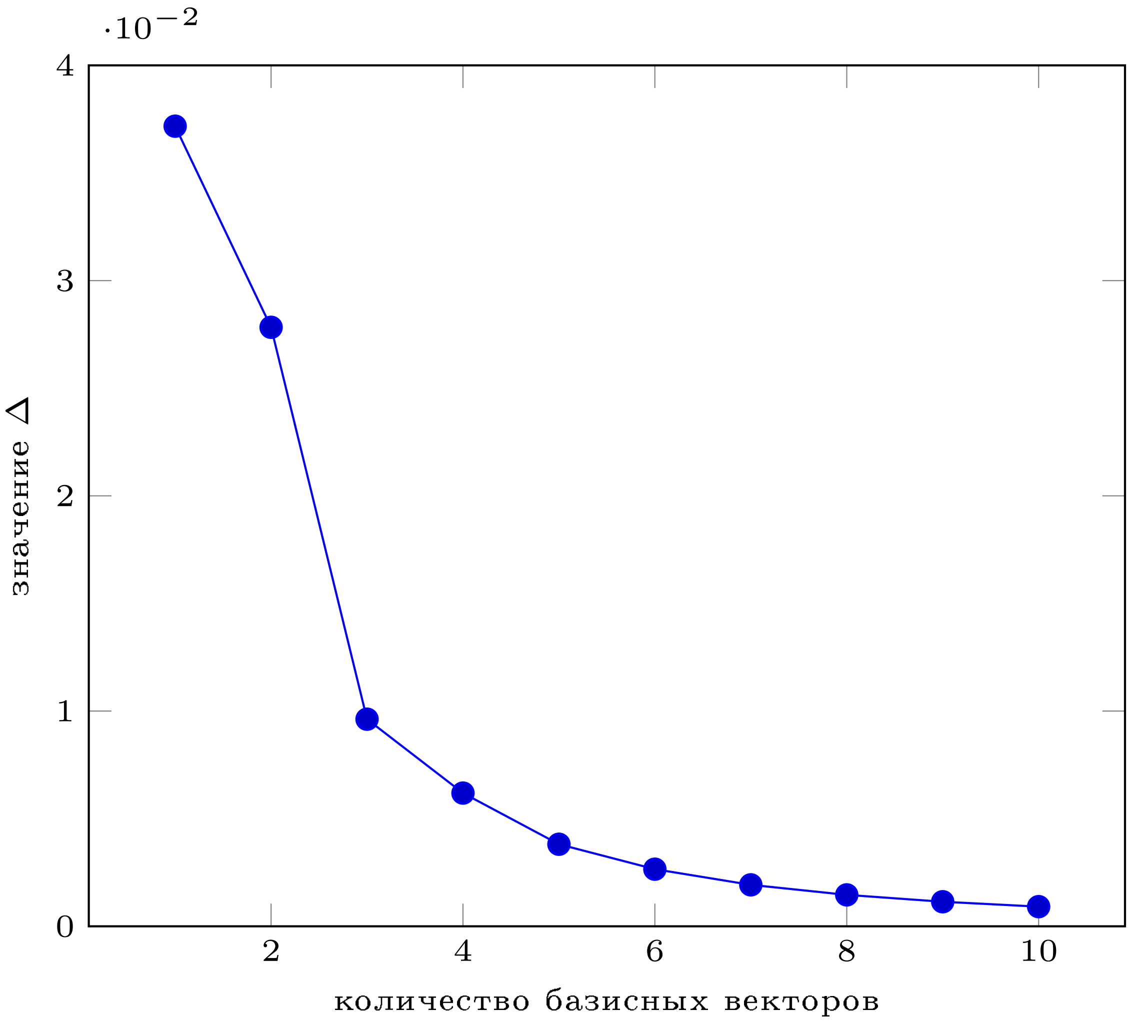

Рисунок 1 — Зависимость меры Δ от количества базисных векторов

На рисунках 2, 3, 4 показаны графики демонстрирующие полученное приближение функции для разных k.

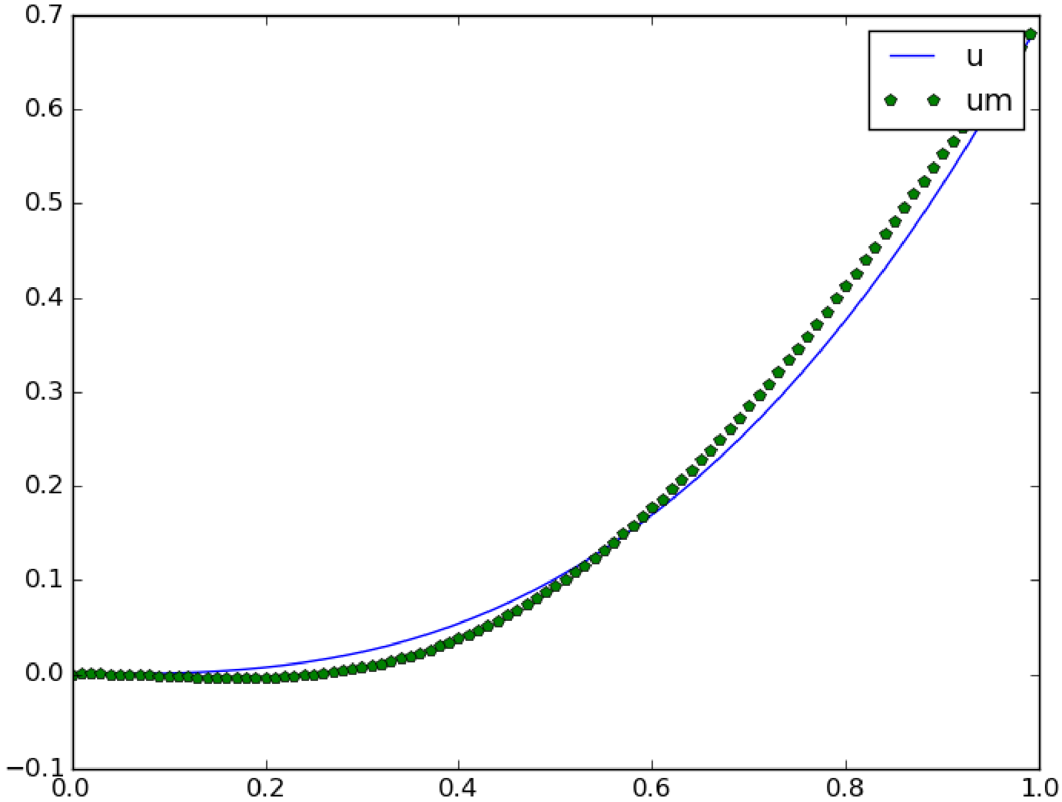

Рисунок 2 — Приближение функции u методом аппроксимации Бубнова-Галеркина, при $k = 1$.

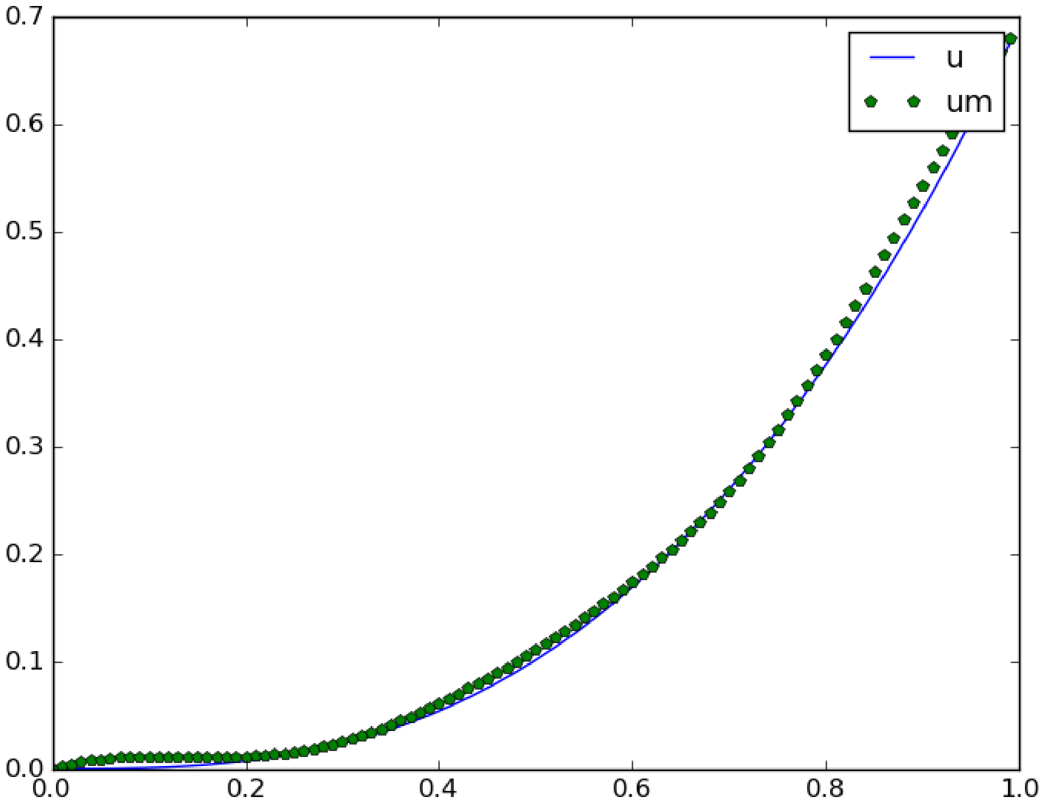

Рисунок 3 — Приближение функции u методом аппроксимации Бубнова-Галеркина, при $k = 3$.

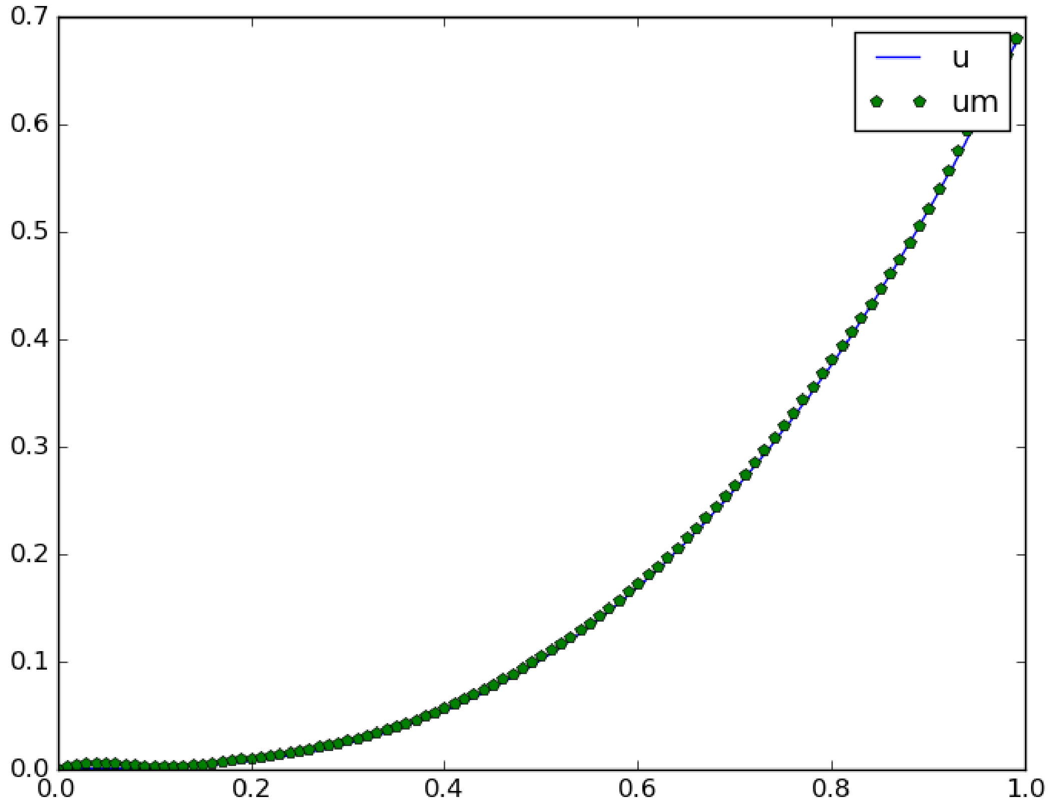

Рисунок 4 — Приближение функции u методом аппроксимации Бубнова-Галеркина, при $k = 8$.