Численные методы математической физики

Лабораторная работа 2

Методы решения граничной задачи для ОДУ-2

Крачковский Даниил
5 группа

Преподаватель:
Будник Анатолий Михайлович

7 июня 2018 г.

Постановка задачи

Дана задача ОДУ-2:

$$((2-x)u'(x))' - xu(x) = sin(x) - 2cos(x), \qquad x \in [0,1]$$

$$2u'(0) = u(0) - 1$$

$$-u'(1) = tg(1)u(1)$$

Решить данную граничную задачу ОДУ-2 методом баланса и методом Ритца.

Алгоритм

Метод баланса

Запишем схему, плученную данным методом, аппроксимирующую нашу задачу:

$$\left(\frac{a_1}{h} - 1 + \frac{h}{2}d_0\right) y_0 + \left(\frac{a_1}{h}\right) y_1 = -\left(1 + \frac{h}{2}\varphi_0\right)$$

$$\left(\frac{a_i}{h^2}\right) y_{i-1} - \left(\frac{a_{i+1} + a_i}{h^2} + d_i\right) y_i + \left(\frac{a_{i+1}}{h^2}\right) y_{i+1} = -\varphi_i \qquad i = \overline{1, N-1}$$

$$\left(\frac{a_N}{h}\right) y_{N-1} + \left(-\frac{a_N}{h} + tg(1) + \frac{h}{2}d_N\right) y_N = \frac{h}{2}\varphi_N$$

где d_i, φ_i, a_i вычичляются по следующим формулам:

$$d_0 = \frac{2}{h}\int_0^{h/2} x dx \qquad \varphi_0 = \frac{2}{h}\int_0^{h/2}(2cos(x) - sin(x))dx$$

$$d_i = \frac{1}{h}\int_{x_i-h/2}^{x_i+h/2} x dx \qquad \varphi_i = \frac{1}{h}\int_{x_i-h/2}^{x_i+h/2}(2cos(x) - sin(x))dx \qquad i = \overline{1, N-1}$$

$$a_i = \left(\frac{1}{h}\int_{x_{i-1}}^{x_i}\frac{dx}{2-x}\right)^{-1} \qquad i = \overline{1, N}$$

$$d_N = \frac{2}{h}\int_{1-h/2}^{1} x dx \qquad \varphi_N = \frac{2}{h}\int_{1-h/2}^{1}(2cos(x) - sin(x))dx$$

Решим образовавшуюся систему методом прогонки.

Метод Ритца

Запишем схему, плученную данным методом, аппроксимирующую нашу задачу:

$$\left(\frac{a_i}{h^2}\right) y_{i-1} - \left(\frac{a_i + a_{i+1}}{h^2} + d_i\right) y_i + \left(\frac{a_{i+1}}{h^2}\right) y_{i+1} = -\varphi_i \qquad i = \overline{0, N}$$

где d_i, φ_i, a_i вычичляются по следующим формулам:

$$a_i = \frac{1}{h}\left(\int_{x_{i-1}}^{x_i} k(x)dx - \int_{x_{i-1}}^{x_i} q(x)(x_i - x)(x - x_{i-1})dx\right) \qquad i = \overline{1, N}$$

$$d_i = \frac{1}{h^2}\left(\int_{x_{i-1}}^{x_i} q(x)(x - x_{i-1})dx - \int_{x_i}^{x_{i+1}} q(x)(x_{i+1} - x)dx\right) \qquad i = \overline{1, N-1}$$

$$d_0 = \frac{2}{h^2}\int_0^h q(x)(h - x)dx \qquad d_N = \frac{2}{h^2}\int_{1-h}^1 q(x)(x - 1 + h)dx$$

$$\varphi_i = \frac{1}{h^2}\left(\int_{x_{i-1}}^{x_i} f(x)(x - x_{i-1})dx - \int_{x_i}^{x_{i+1}} f(x)(x_{i+1} - x)dx\right) \qquad i = \overline{1, N-1}$$

$$\varphi_0 = \frac{2}{h^2}\int_0^h f(x)(h - x)dx \qquad \varphi_N = \frac{2}{h^2}\int_{1-h}^1 f(x)(x - 1 + h)dx$$

Решим образовавшуюся систему методом прогонки.

1 Листинг кода

```
import numpy as np
import scipy as sp
import scipy.integrate
import math as ma
import matplotlib.pyplot as plt

q_x = lambda x: x
k_x = lambda x: 2 - x
_1_k_x = lambda x: 1 / k_x(x)
f_x = lambda x: 2*ma.cos(x) - ma.sin(x)

_a, _b = 0, 1

def integrate(func, _from, to):
  return sp.integrate.quad(func, _from, to)[0]


def balance(n):
  h = (_b - _a) / n
  x = [_a + (i * h) for i in range(0, n + 1)]

  d = np.zeros(n + 1)
  d[0] = 2/h * integrate(q_x, 0, h/2)
  d[n] = 2/h * integrate(q_x, 1 - h/2, 1)

  for i in range(1, n):
    d[i] = 1/h * integrate(q_x, x[i] - h/2, x[i] + h/2)


  a = np.zeros(n + 1)

  for i in range(1, n + 1):
    a[i] = 1 / (1/h * integrate(_1_k_x, x[i - 1], x[i]))


  phi = np.zeros(n + 1)
  phi[0] = 2/h * integrate(f_x, 0, h/2)
  phi[n] = 2/h * integrate(f_x, 1 - h/2, 1)

  for i in range(1, n):
    phi[i] = 1/h * integrate(f_x, x[i] - h/2, x[i] + h/2)


  A = np.identity(n + 1)
  A[0, 0] = a[1]/h - 1 + h/2 * d[0]
  A[0, 1] = a[1]/h

  A[n, n - 1] = a[n]/h
  A[n, n] = -a[n]/h + ma.tan(1) + h/2*d[n]
```

```
  for i in range(1, n):
    A[i, i - 1] = a[i] / h**2
    A[i, i] = -((a[i + 1] + a[i]) / h**2 + d[i])
    A[i, i + 1] = a[i + 1] / h**2


  B = np.zeros(n + 1)
  B[0] = -1 - h/2 * phi[0]
  B[n] = h/2 * phi[n]

  for i in range(1, n):
    B[i] = - phi[i]

  return np.linalg.solve(A, B), x


def ritzh(n):
  h = (_b - _a) / n
  x = [_a + (i * h) for i in range(0, n + 1)]

  A = np.identity(n + 1)

  A[0, 0] = 1/h**2 * (integrate(k_x, 0, h) + integrate(lambda _x: q_x(_x) * (_x - h)**2, 0, h
  A[n, n] = 1/h**2 * (integrate(k_x, 1 - h, 1) + integrate(lambda _x: q_x(_x) * (_x - 1 + h)*

  for i in range(1, n):
    res = 1/h**2 * (integrate(lambda _x: q_x(_x) * (x[i + 1] - _x) * (_x - x[i]), x[i], x[i +
    A[i, i + 1] = res
    A[i + 1, i] = res

  for i in range(1, n):
    A[i, i] = 1/h**2 * (integrate(k_x, x[i - 1], x[i + 1]) + integrate(lambda _x: q_x(_x)*(_x


  B = np.zeros(n + 1)

  B[0] = 1/h * integrate(lambda _x: f_x(_x) * (_x - h), 0, h) + 1
  B[n] = 1/h * integrate(lambda _x: f_x(_x) * (_x - 1 + h), 1 - h, 1)

  for i in range(1, n):
    B[i] = 1/h * (integrate(lambda _x: f_x(_x) * (_x - x[i - 1]), x[i - 1], x[i]) + integrate

  for r in A:
    for e in r:
      print('{:15.5}'.format(e), end='')
    print()

  print(B)

  return np.linalg.solve(A, B), x
```

```
if __name__ == "__main__":

  for n in range(1, 2):
    solutionBalance, x = balance(10 * n)
    plt.plot(x, solutionBalance)

  for n in range(1, 10):
    solutionRitzh, x = ritzh(10 * n)
    plt.plot(x, solutionRitzh)

  plt.show()
```