Monika Pyrek

Równania różniczkowe i różnicowe

Zadanie Obliczeniowe

1. Równanie transportu ciepła

$$-k(x)\frac{d^2u(x)}{dx^2} = 0$$
$$u(2) = 0$$
$$\frac{du(0)}{dx} + u(0) = 20$$
$$k(x) = \begin{cases} x + 1 \; dla \; x \in [0,1] \\ 2x \; dla \; x \in (1,2] \end{cases}$$

Gdzie u to poszukiwana funkcja

$$[0,2] \ni x \rightarrow u(x) \in R$$

2. Z prawej strony mamy warunek Dirchleta, z lewej nie, więc za przestrzeń funkcji V przyjmujemy te, które zerują się na prawym brzegu.

Zauważamy, że funkcja k(x) jest niezerowa na [0,2], więc możemy przez niego podzielić. Równanie ma teraz postać:

$$-u''(x)v(x)dx = 0$$

Mnożymy równanie przez dowolną funkcję v ∈ V

$$\int_0^2 -u''(x)v(x)dx = 0$$

Pozbywamy się drugiej pochodnej całkując przez części

$$-u'(x)v(x)|_0^2 + \int_0^2 u'(x)v'(x)dx = 0$$
$$-u'(2)v(2) + u'(0)v(0) + \int_0^2 u'(x)v'(x)dx = 0$$

Jako że po prawej stronie mamy warunek Dirchleta, a v ∈ V, mamy

v(2) = 0.

Ponadto, z drugiego warunku brzegowego $\frac{du(0)}{dx} + u(0) = 20$, otrzymujemy:

$$u'(0) = 20 - u(0)$$

$Podstawiając\ obie\ te\ rzeczy\ do\ powyższego\ rowania\ otrzymuemy$:

$$(20 - u(0))v(0) + \int_0^2 u'(x)v'(x)dx = 0$$

Sprowadzając równanie do postaci B(u, v) = L(v) chcemy po lewej stronie
mieć tylko wyrazy zawierające zarówno u jak i v, zaś z lewej te zawierające samo v.

$$-u(0)v(0) + \int_0^2 u'(x)v'(x)dx = -20v(0)$$

Kod programu

```
import matplotlib.pyplot as plt
import scipy.integrate
import numpy as np


def x_i(i,n):
    return 2*i / n;

# za funkcje bazową przyjmujemy
#e_i = {x - x_i-1/x_i-x_i-1 , x należy do (x_i-1,x_i),
#        x_i+1- x / x_i+1-x_i , x należy do (x_i,x_i+1) }
# wiemy, że x_i+1-x_i  =  x_i - x_i-1 =  h = 2/n

def e(i,x,n):
    if x >= x_i(i-1,n) and x <= x_i(i,n):
        return (x-x_i(i-1,n))/(2/n)
    elif x >x_i(i,n) and x <= x_i(i+1,n):
        return (x_i(i+1,n)-x)/(2/n)
    else:
        return 0


#pochodna e
def e_prim(i,x,n):
    if x >= x_i(i-1,n) and x <= x_i(i,n):
        return n/2
    elif x >x_i(i,n) and x <= x_i(i+1,n):
           return -n/2
    else:
        return 0


def L(i,n):
    return -20*e(i,0,n)


def B(i,j,n):
    # szukamy przedziałów całkowania
    a = 0
    b = 2

    if abs(i-j) <= 1:
        if i >= j:
            a = max(a,x_i(i-1,n))
            b = min(b,x_i(j+1,n))
        else:
            a = max(a,x_i(j-1,n))
```

```
            b = min(b,x_i(i+1,n))

        return scipy.integrate.quad(lambda x :
e_prim(i,x,n)*e_prim(j,x,n),a,b)[0] - e(i,0,n)*e(j,0,n)
    else:
        return 0



def find_soltion(n):
    #obliczamy odległość pomiędzy kolejnymi punktami
    h = 2 / n

    A = [] #macierz dla B(u,v)
    G = [] #macierz dla L(v)

    for i in range(n):
        A.append([])
        for j in range(n):
            A[i].append(B(j,i,n))

    for i in range(n):
        G.append(L(i,n))

    U = np.linalg.solve((A),(G))

    Results = [0]*200

    #obliczanie wartości dla funkcji
    for i in range(200):
        suma = 0
        for j in range(len(U)):
            suma = suma + U[j]*e(j,x_i(i,200),n)
        Results[i] = suma


    set_X = np.linspace(0,2,200)

    plt.title("Wykres aproksymacji funckji y = u(x)")
    plt.plot(set_X, Results)
    plt.show()




find_soltion(10)
```