Численное решение интегрального уравнения Фредгольма 2-го рода методом вырожденных ядер

А. Климовский

13 Мая 2001

1 Постановка задачи

Рассморим уравнение Фредгольма 2-го рода (здесь и далее $y(t),\ f(t) \in \mathcal{L}^2([a;b])$; $K(t,s) \in \mathcal{L}^2([a;b]^2)$):

$$y(t) + \int_a^b K(t,s)y(s)ds = f(t) \tag{1}$$

Тестовая задача:

$$y(t) + \frac{1}{2}\int_0^1 e^{-ts}y(s)ds = t + \frac{1-e^{-t}}{2} \tag{2}$$

2 Краткое изложение метода и вычислительная схема решения

Рассмотрим следующее представление ядра тестовой задачи (2):

$$e^{-ts} = \sum_{i=0}^{n} \frac{(-t)^n}{n!}s^n + O\left((ts)^n\right),\ (t,s) \in \mathbb{R}^2 \tag{3}$$

Обозначим $\phi(t) := \frac{(-t)^n}{n!}$, $\psi(t) := s^n$, $c_j := (\psi_j, y)$, $A_{ij} := (\psi_i, \phi_j)$, $f_i := (\psi_i, f)$. Заменяя ядро в уравнении (2) конечным представлением из (3) и домножая (2) скалярно на $\psi_i(t)$ получаем систему линейных алгебраических уравнений (СЛАУ) для нахождения c_j:

$$c_i + \sum_{j=1}^{n} A_{ij}c_j = f_i,\ i = 1, \ldots, n \tag{4}$$

Решив данную систему можно найти приближенное решение $\tilde{y}(t)$ уравнения (2) из соотношения:

$$\tilde{y}(t) = f(t) - \sum_{j=1}^{n} c_j\phi_j(x) \tag{5}$$

В данной работе СЛАУ (4) решалась методом Гаусса с выбором главного элемента, а скалярные произведения в $L^2([a;b])$ вычислялись с использованием квадратурной формулы трапеций.

3 Результаты счета

Ниже приведен вывод генерируемый программой:

```
Number of points along X axis: 20
y(0.000) = -0.252
y(0.050) = -0.169
y(0.100) = -0.087
y(0.150) = -0.006
y(0.200) =  0.073
y(0.250) =  0.151
y(0.300) =  0.227
y(0.350) =  0.302
y(0.400) =  0.377
y(0.450) =  0.450
y(0.500) =  0.522
y(0.550) =  0.593
y(0.600) =  0.663
y(0.650) =  0.732
y(0.700) =  0.800
y(0.750) =  0.868
y(0.800) =  0.934
y(0.850) =  1.000
y(0.900) =  1.066
y(0.950) =  1.130
```

4 Приложение: исходные тексты программы

```
//-----------------------------------------------
// Solves folowing Fredholm's integral equation
// of the 2nd order:
//
// \begin{equation}
//     \label{eq:testFredholm2}
//     y(t)+\frac{1}{2}\int_0^1{e^{-ts}y(s)ds} = t+\frac{1-e^{-t}}{2}
// \end{equation}
//
// Author: A. Klimovsky, root@ludus.kharkiv.com
//-----------------------------------------------
#include <iostream.h>
#include <math.h>

#include "lin_alg.h"
#include "gauss.h"
#include "integrator.h"

const double a = 0;
const double b = 1;
```

```
const double eps = 1.0e-3;

long int n0;
long int n = 50;

double h;
vector c;

double f(double t)
{
    return t+(1-exp(-t))/2;
}

double psi(int n, double t)
{
    int i;
    double temp;

    temp = 1;
    for (i = 1; i <= n; i++)
        temp *= t;
    return temp;
}

double phi(int n, double t)
{
    int i;
    double temp;

    temp = 0.5;
    for (i = 1; i <= n; i++)
        temp *= (-t)/(i);
    return temp;
}

class psiPhi {
    int i;
    int j;

public:
    psiPhi(int i0, int j0):
        i(i0),
        j(j0)
    {}

    double operator() (double t)
    {
        return psi(i, t)*phi(j, t);
    }
};

class psiF {
    int i;
```

```
public:
    psiF(int i0):
        i(i0)
    {}

    double operator() (double t)
    {
        return psi(i, t)*f(t);
    }
};

void inputMeshGranularity()
{
    cout << "Number of points along X axis: ";
    cin >> n0;
}

void solve()
{
    matrix A(n, n);
    vector f(n);
    long int i;
    long int j;

    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++)
            A[i][j] = integrate(a, b, psiPhi(i, j));
        A[i][i] += 1;
        f[i] = integrate(a, b, psiF(i));
    }
    c = calculate_inverse_matrix(A)*f;
}

double y(double t)
{
    int i;
    double temp;

    temp = 0;
    for (i = 0; i < n; i++)
        temp += c[i]*phi(i, t);

    return f(t)-temp;
}

void initialize()
{
    h = (b-a)/n0;
}

void output()
{
    long int i;
    double t;
```

```
    cout.setf(ios_base::fixed, ios_base::floatfield);
    cout.precision(3);

    for (i = 0, t = a; i < n0; i++, t += h)
        cout << "y(" << t << ") = " << y(t) << endl;
}

void main()
{
    inputMeshGranularity();
    initialize();
    solve();
    output();
}
```