МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»
НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Лабораторна робота №9
з курсу «Чисельні методи»
тема: «Диференціальні рівняння у частинних похідних»

Виконав: студент 3 курсу

групи КА-23

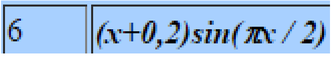
Деундяк О.В.

Прийняла: Кузнєцова Н. В.

Київ – 2014р.

Рівняння коливань струни.

Розв'язати рівняння гіперболічного типу $\boldsymbol{u_{tt} = u_{xx} + F(t,x)}$, $\boldsymbol{0 < x < L = 1}$,   (2) для функції $\boldsymbol{u(t,x)}$ з початковими $\boldsymbol{u(0,x) = u_0(x)}$, $\boldsymbol{u_t(0,x) = 0}$ та крайовими $\boldsymbol{u(t,0) = u_1(t)}$; $\boldsymbol{u(t,L) = u_2(t)}$ умовами.



6	*(x+0,2)sin(πx / 2)*

$$u(t,x) = (x + 0.2)\sin\left(\frac{\pi x}{2}\right)\cos(\pi t)$$

Текст програми:

TMA.h

```
#pragma once

#include <vector>

class TMA
{
   std::vector<double> c, d, r;
   size_t i; // number of iterations
   size_t n; // size of system
public:
   TMA(size_t n) : i(0), c(n), d(n), r(n)
   {
      this->n = n;
   }

   std::vector<double> Result();
   void Iterate(double a, double b, double c, double d);
};
```

TMA.cpp

```
#include "TMA.h"

void TMA::Iterate(double A, double B, double C, double D)
{
   if (i == n)
      return;

   if (i == 0)
   {
      c[i] = C / B;
      d[i] = D / B;
   }
   else
   {
      c[i] = C / (B - A*c[i - 1]);
      d[i] = (D - A*d[i - 1]) / (B - A*c[i - 1]);
   }
   i++;
}

std::vector<double> TMA::Result()
{
   if (i != n)
      return std::vector<double>();

   r[n - 1] = d[n - 1];;
   for (size_t i = n - 1; i > 0; i--)
      r[i - 1] = d[i - 1] - c[i - 1] * r[i];

   return r;
};
```

Hyper.h

```
#pragma once

#define _USE_MATH_DEFINES
#include <cmath>

#include <functional>
#include <vector>

typedef std::function<double(double)> func;
typedef std::function<double(double, double)> func2;

class Hyper
{
public:
    static std::vector<std::vector<double>> Process(double s, func2 F, func u0, func u1, func u2, func v0,
size_t Nt, size_t Nl, double period, double length);
};
```

Hyper.cpp

```
#include "Hyper.h"
#include "TMA.h"

std::vector<std::vector<double>> Hyper::Process(double s, func2 F,
    func u0, func u1, func u2, func v0,
    size_t Nt, size_t Nl, double period, double length)
{
    std::vector<std::vector<double>> u(Nt+1, std::vector<double>(Nl+1));

    double h = length / Nl;
    double dt = period / Nt;

    double q = h / dt*h / dt;

    for (size_t n = 0; n <= Nl; n++)
        u[0][n] = u0(n*h);

    u[1][0] = u1(dt);
    u[1][Nl] = u2(dt);
    for (size_t n = 1; n < Nl; n++)
        u[1][n] = u[0][n] + dt*v0(h*n) + dt*dt / 2 * ((u[0][n + 1] - 2 * u[0][n] + u[0][n - 1]) / h / h +F(0, h*n));

    double err = 0;
    for (size_t k = 1; k < Nt; k++)
    {
        TMA tma(Nl + 1);
        tma.Iterate(0, 1, 0, u1((k + 1)*dt));

        for (size_t n = 1; n < Nl; n++)
            tma.Iterate(-s, q + 2 * s, -s,
                2 * u[k][n]*q - u[k - 1][n]*q + (1 - 2 * s)*(u[k][n + 1] - 2 * u[k][n] + u[k][n - 1]) + s*(u[k - 1][n + 1]
- 2 * u[k - 1][n] + u[k - 1][n - 1])
                + (s*F((k + 1)*dt, n*h) + (1 - 2 * s)*F(k*dt, n*h) + s*F((k - 1)*dt, n*h))*h*h);
        tma.Iterate(0, 1, 0, u2((k + 1)*dt));

        u[k+1] = tma.Result();
    }

    return u;
}
```

main.cpp

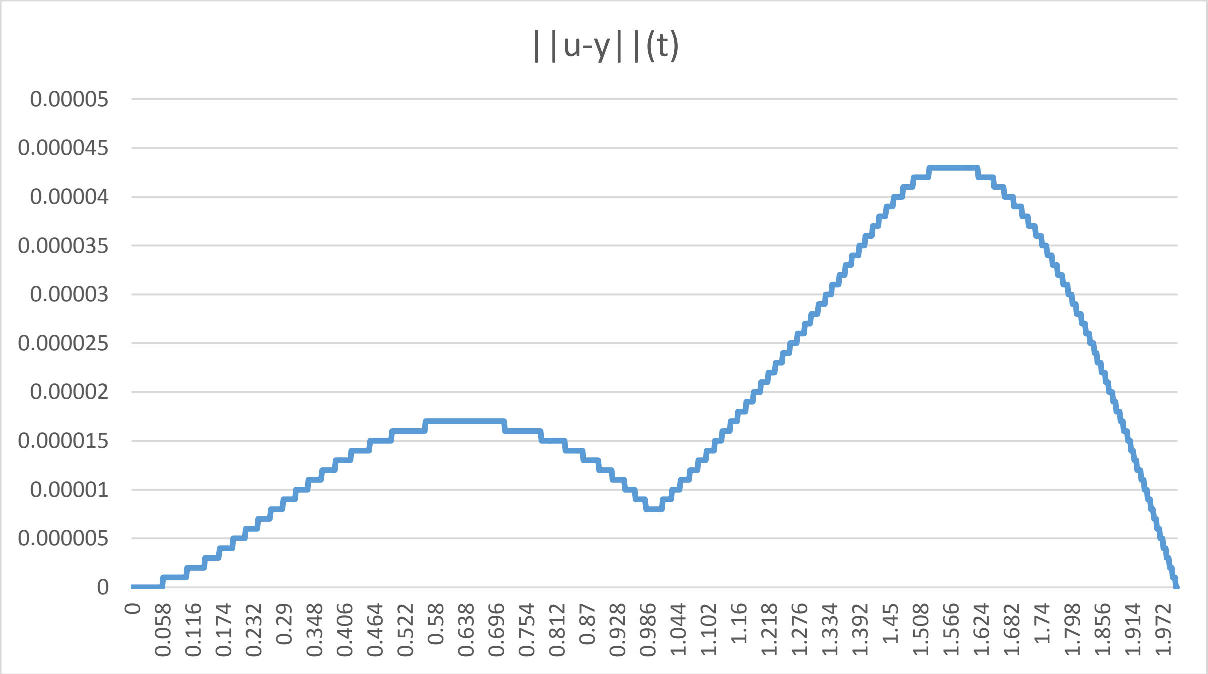
```
#include "Hyper.h"
#include <iostream>
#include <algorithm>
#include <iomanip>
#include <fstream>

int Test()
{
    size_t const Nt = 1000;
    size_t const Nl = 1000;

    func u0 = [](double x)->double{ return (x + 0.2)*sin(M_PI*x / 2); };
    func u1 = [](double t)->double{ return 0.; };
    func u2 = [](double t)->double{ return 1.2*cos(M_PI*t); };
    func v0 = [](double x)->double{ return 0.; };

    func2 utt = [](double t, double x)->double{ return -M_PI*M_PI*(x + 0.2)*sin(M_PI*x / 2)*cos(M_PI*t);
};
    func2 uxx = [](double t, double x)->double{ return (-
M_PI*M_PI/4*(x+0.2)*sin(M_PI*x/2)+M_PI*cos(M_PI*x/2))*cos(M_PI*t); };

    func2 F = [utt, uxx](double t, double x)->double{ return utt(t,x)-uxx(t,x); };

    func2 answ = [](double t, double x)->double{ return (x + 0.2)*sin(M_PI*x / 2)*cos(M_PI*t); };

    double period = 2, length = 1;
    double h = length / Nl;
    double dt = period / Nt;

    std::vector<std::vector<double>> res = Hyper::Process(0.75, F, u0, u1, u2, v0, Nt, Nl, period, length);

    std::ofstream out("output.txt");

    out << std::left << std::fixed;

    {
        for (size_t k = 0; k < Nt + 1; k++)
        {
            double m = 0;
            for (size_t n = 0; n < Nl + 1; n++)
            {
                double a1 = answ(k*dt, n*h);
                double a2 = res[k][n];
                double err = abs(a1-a2);
                m = std::max(m,err);
            }
            out << std::setw(4) << k*dt << '\t'
                << std::setw(12) << m << std::endl;
        }
    }

    /*{
        for (size_t k = 0; k < Nt + 1; k++)
        {
            double m = 0;
            for (size_t n = 0; n < Nl + 1; n++)
            if (abs(res[k][n]) > m)
                m = abs(res[k][n]);

            out << std::setw(4) << k*dt << '\t'
                << std::setw(12) << m << std::endl;
        }
    }*/

    /*{
        for (size_t n = 0; n < Nl + 1; n++)
        out << std::setw(4) << n*h << '\t'
            << std::setw(12) << res[Nt][n] << std::endl;
    }*/

    /*{
        for (size_t n = 0; n < Nl + 1; n++)
            out << std::setw(4) << n*h << '\t'
            << std::setw(12) << answ(Nt*dt, n*h) - res[Nt][n] << std::endl;
    }*/

    /*for (size_t k = 0; k <= Nt; k++)
    {
        out << "t=" << k*dt << std::endl;
        for (size_t n = 0; n <= Nl; n++)
        {
            out << n*h << '\t' << answ(k*dt, n*h) << '\t' << res[k][n] << '\t' << answ(k*dt, n*h) - res[k][n] << std::endl;
        }
        out << std::endl;
    }*/

    out << std::endl;

    return 0;
}

int main()
{
    return Test();
}
```

Результати роботи програми

```
t=0.000000
0.000000        0.000000        0.000000        0.000000
0.100000        0.046930        0.046930        0.000000
0.200000        0.123607        0.123607        0.000000
0.300000        0.226995        0.226995        0.000000
0.400000        0.352671        0.352671        0.000000
0.500000        0.494975        0.494975        0.000000
0.600000        0.647214        0.647214        0.000000
0.700000        0.801906        0.801906        0.000000
0.800000        0.951057        0.951057        0.000000
0.900000        1.086457        1.086457        0.000000
1.000000        1.200000        1.200000        0.000000

t=0.200000
0.000000        0.000000        0.000000        0.000000
0.100000        0.037967        0.037417        0.000551
0.200000        0.100000        0.098975        0.001025
0.300000        0.183643        0.181981        0.001662
0.400000        0.285317        0.282884        0.002433
0.500000        0.400443        0.397138        0.003305
0.600000        0.523607        0.519373        0.004234
0.700000        0.648755        0.643580        0.005175
0.800000        0.769421        0.763342        0.006079
0.900000        0.878962        0.872069        0.006893
1.000000        0.970820        0.970820        0.000000

t=0.400000
0.000000        0.000000        0.000000        0.000000
0.100000        0.014502        0.019239        -0.004736
0.200000        0.038197        0.048493        -0.010297
0.300000        0.070145        0.086757        -0.016612
0.400000        0.108981        0.132405        -0.023423
0.500000        0.152956        0.183083        -0.030128
0.600000        0.200000        0.235705        -0.035705
0.700000        0.247803        0.286332        -0.038529
0.800000        0.293893        0.329894        -0.036001
0.900000        0.335734        0.359594        -0.023860
1.000000        0.370820        0.370820        0.000000

t=0.600000
0.000000        -0.000000       0.000000        -0.000000
0.100000        -0.014502       -0.001710       -0.012792
0.200000        -0.038197       -0.011665       -0.026531
0.300000        -0.070145       -0.028814       -0.041331
0.400000        -0.108981       -0.052550       -0.056431
0.500000        -0.152956       -0.082629       -0.070326
0.600000        -0.200000       -0.119310       -0.080690
0.700000        -0.247803       -0.163562       -0.084241
0.800000        -0.293893       -0.217317       -0.076576
0.900000        -0.335734       -0.283648       -0.052086
1.000000        -0.370820       -0.370820       0.000000

t=0.800000
0.000000        -0.000000       0.000000        -0.000000
0.100000        -0.037967       -0.019477       -0.018491
0.200000        -0.100000       -0.063549       -0.036451
0.300000        -0.183643       -0.130141       -0.053502
0.400000        -0.285317       -0.216929       -0.068388
0.500000        -0.400443       -0.321266       -0.079177
0.600000        -0.523607       -0.440222       -0.083384
0.700000        -0.648755       -0.570379       -0.078377
```

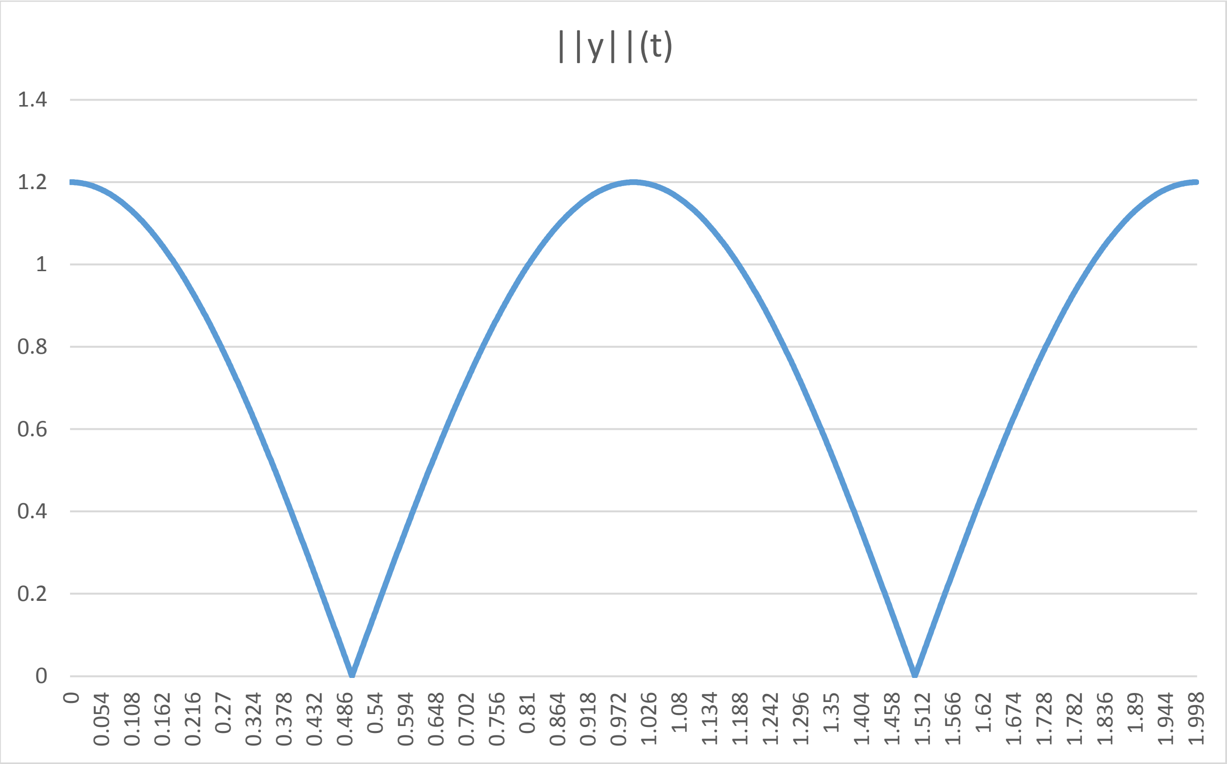
```
0.800000        -0.769421       -0.707090       -0.062331
0.900000        -0.878962       -0.842728       -0.036235
1.000000        -0.970820       -0.970820       0.000000

t=1.000000
0.000000        -0.000000       0.000000        -0.000000
0.100000        -0.046930       -0.031818       -0.015112
0.200000        -0.123607       -0.096536       -0.027071
0.300000        -0.226995       -0.192813       -0.034182
0.400000        -0.352671       -0.317559       -0.035112
0.500000        -0.494975       -0.466066       -0.028908
0.600000        -0.647214       -0.631739       -0.015475
0.700000        -0.801906       -0.805259       0.003353
0.800000        -0.951057       -0.972999       0.021943
0.900000        -1.086457       -1.114482       0.028025
1.000000        -1.200000       -1.200000       0.000000

t=1.200000
0.000000        -0.000000       0.000000        -0.000000
0.100000        -0.037967       -0.040829       0.002862
0.200000        -0.100000       -0.110712       0.010712
0.300000        -0.183643       -0.209974       0.026331
0.400000        -0.285317       -0.335465       0.050148
0.500000        -0.400443       -0.480689       0.080246
0.600000        -0.523607       -0.635335       0.111728
0.700000        -0.648755       -0.784520       0.135764
0.800000        -0.769421       -0.908130       0.138709
0.900000        -0.878962       -0.981261       0.102298
1.000000        -0.970820       -0.970820       0.000000

t=1.400000
0.000000        -0.000000       0.000000        -0.000000
0.100000        -0.014502       -0.049540       0.035038
0.200000        -0.038197       -0.112005       0.073808
0.300000        -0.070145       -0.188241       0.118095
0.400000        -0.108981       -0.275006       0.166024
0.500000        -0.152956       -0.364698       0.211742
0.600000        -0.200000       -0.445266       0.245266
0.700000        -0.247803       -0.500876       0.253074
0.800000        -0.293893       -0.514247       0.220354
0.900000        -0.335734       -0.472206       0.136472
1.000000        -0.370820       -0.370820       0.000000

t=1.600000
0.000000        0.000000        0.000000        0.000000
0.100000        0.014502        -0.056749       0.071251
0.200000        0.038197        -0.102704       0.140900
0.300000        0.070145        -0.136402       0.206548
0.400000        0.108981        -0.154025       0.263007
0.500000        0.152956        -0.148840       0.301796
0.600000        0.200000        -0.112164       0.312164
0.700000        0.247803        -0.036146       0.283948
0.800000        0.293893        0.081176        0.212716
0.900000        0.335734        0.228589        0.107144
1.000000        0.370820        0.370820        0.000000

t=1.800000
0.000000        0.000000        0.000000        0.000000
0.100000        0.037967        -0.054388       0.092356
0.200000        0.100000        -0.075323       0.175323
0.300000        0.183643        -0.056816       0.240459
0.400000        0.285317        0.005356        0.279961
0.500000        0.400443        0.113608        0.286835
```

```
0.600000        0.523607        0.266561        0.257046
0.700000        0.648755        0.455506        0.193250
0.800000        0.769421        0.660364        0.109057
0.900000        0.878962        0.848011        0.030952
1.000000        0.970820        0.970820        0.000000

t=2.000000
0.000000        0.000000        0.000000        0.000000
0.100000        0.046930        -0.030891       0.077822
0.200000        0.123607        -0.016673       0.140280
0.300000        0.226995        0.052046        0.174950
0.400000        0.352671        0.178034        0.174637
0.500000        0.494975        0.356021        0.138953
0.600000        0.647214        0.570939        0.076275
0.700000        0.801906        0.796811        0.005095
0.800000        0.951057        0.998539        -0.047482
0.900000        1.086457        1.140019        -0.053562
1.000000        1.200000        1.200000        0.000000
```



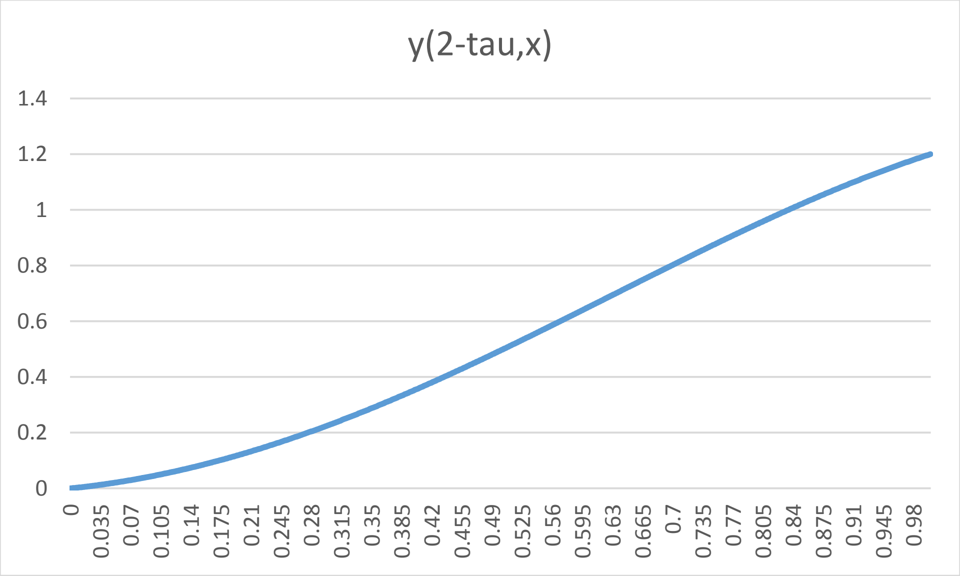
||u-y||(t)

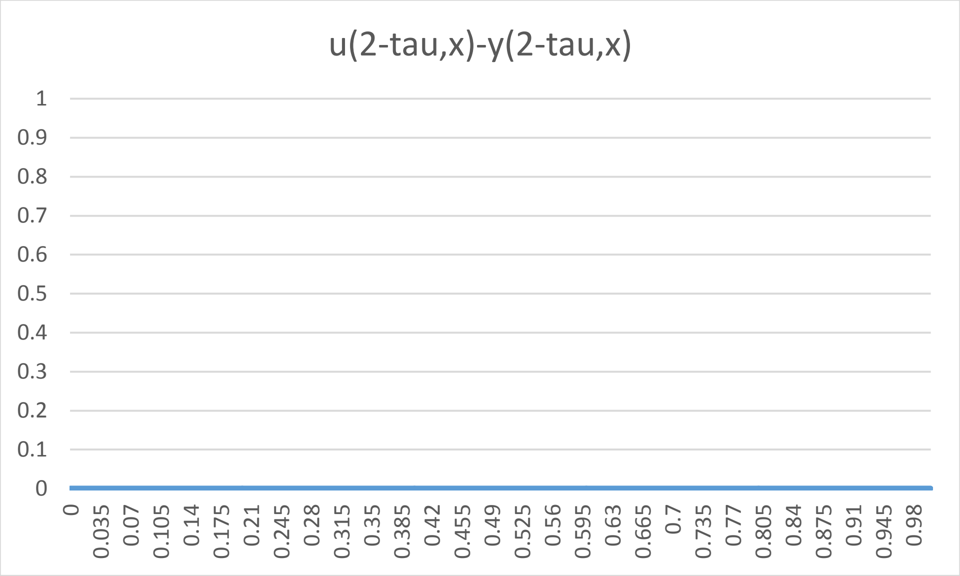
0.00005
0.000045
0.00004
0.000035
0.00003
0.000025
0.00002
0.000015
0.00001
0.000005
0
0
0.058
0.116
0.174
0.232
0.29
0.348
0.406
0.464
0.522
0.58
0.638
0.696
0.754
0.812
0.87
0.928
0.986
1.044
1.102
1.16
1.218
1.276
1.334
1.392
1.45
1.508
1.566
1.624
1.682
1.74
1.798
1.856
1.914
1.972



||y||(t)

1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
0.054
0.108
0.162
0.216
0.27
0.324
0.378
0.432
0.486
0.54
0.594
0.648
0.702
0.756
0.81
0.864
0.918
0.972
1.026
1.08
1.134
1.188
1.242
1.296
1.35
1.404
1.458
1.512
1.566
1.62
1.674
1.728
1.782
1.836
1.89
1.944
1.998


y(2-tau,x)
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
0.035
0.07
0.105
0.14
0.175
0.21
0.245
0.28
0.315
0.35
0.385
0.42
0.455
0.49
0.525
0.56
0.595
0.63
0.665
0.7
0.735
0.77
0.805
0.84
0.875
0.91
0.945
0.98


u(2-tau,x)-y(2-tau,x)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.035
0.07
0.105
0.14
0.175
0.21
0.245
0.28
0.315
0.35
0.385
0.42
0.455
0.49
0.525
0.56
0.595
0.63
0.665
0.7
0.735
0.77
0.805
0.84
0.875
0.91
0.945
0.98