**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Численные методы математической физики

Лабораторная работа №2

«Решение краевой задачи для уравнения колебаний струны»

Вариант К-3

Крученкова Евгения Андреевича

студента 3 курса,

специальность «Прикладная математика»

Преподаватель:

доцент кафедры ВычМат,

кандидат физико-математических наук

Будник А.М.

Минск, 2022

*ОГЛАВЛЕНИЕ*

*Постановка задачи. 3*

*1. Построение разностной схемы с заданной погрешностью аппрокксимации 4*

*2. Исследование устойчивости полученной разностной схемы с помощью МРП 7*

*3. Реализация разностной схемы 8*

*Реализация метода прогонки. 8*

*Листинг программы. 9*

*Результаты. 12*

*Выводы. 14*

Постановка задачи.

Для решения краевой задачи вида:

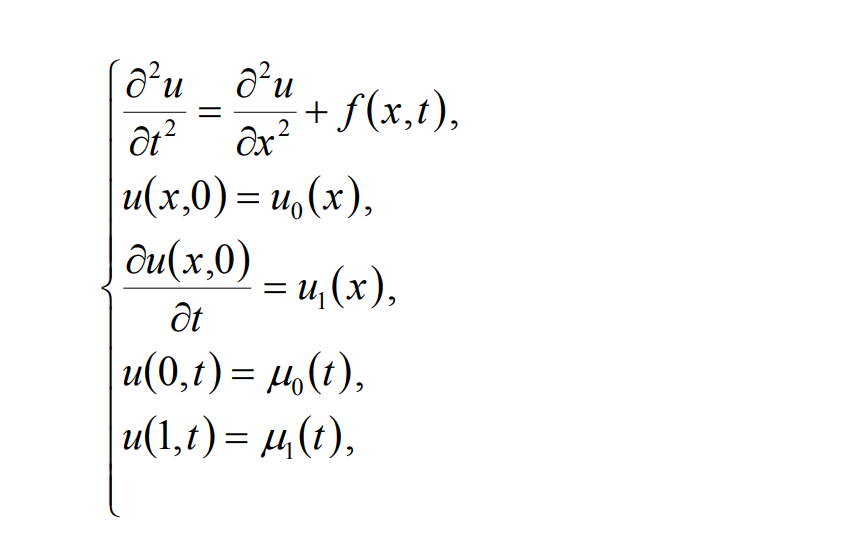
на сетке построить разностную схему с весами при с погрешностью аппроксимации не ниже

Необходимо:

1. Показать, что построенная разностная схема имеет заданный порядок аппроксимации
2. Исследовать устойчивость разностной схемы по начальным данным, используя метод разделения переменных.
3. Реализовать данную разностную схему при и , выбранным из условия устойчивости.
4. Оценить приближённое решение, анализируя погрешность аппроксимации при разных шагах.

# Построение разностной схемы с заданной погрешностью аппрокксимации

Перепишем задачу в общем виде

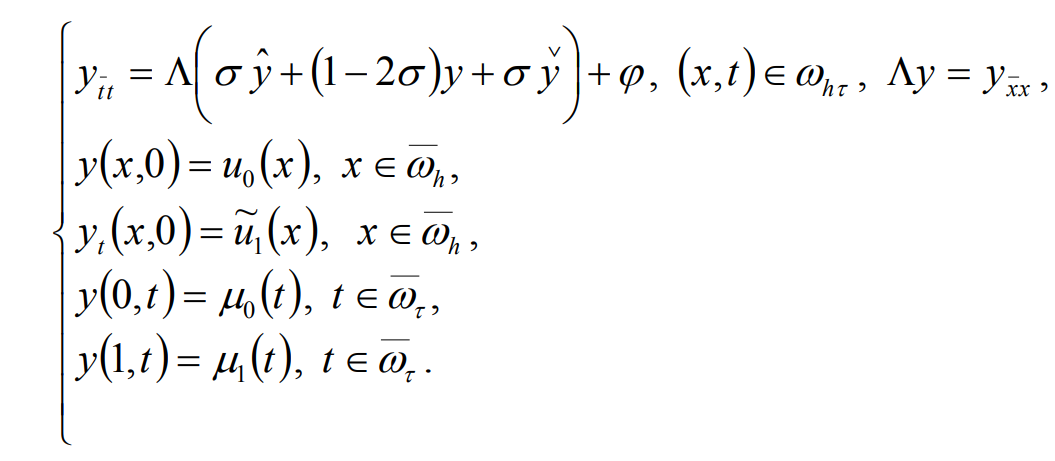


где

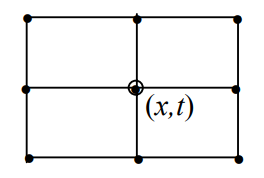
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

Введем в рассмотрение равномерную сетку с количеством разбиений и шагом по x и соответственно, и с количеством разбиений и шагом по t и соответственно. В нашем случае , , количество разбиений и шаг будут выбраны позднее из условий устойчивости.

Заменим функции на сеточные и дифференциальные опервторы на разностные, рассмотрим схему с весами:

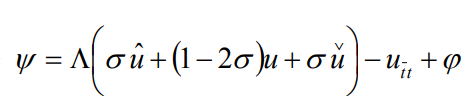


Шаблон для данной схемы будет выглядеть следующим образом:

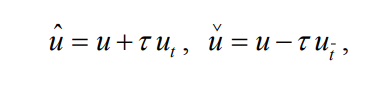
****

Исследуем порядок аппроксимации данной разностной схемы при и подберём и таким образом, чтобы порядок аппроксимации схемы был не ниже

Сначала рассмотрим погрешность аппроксимации разностной схемы на решении исходной задачи:



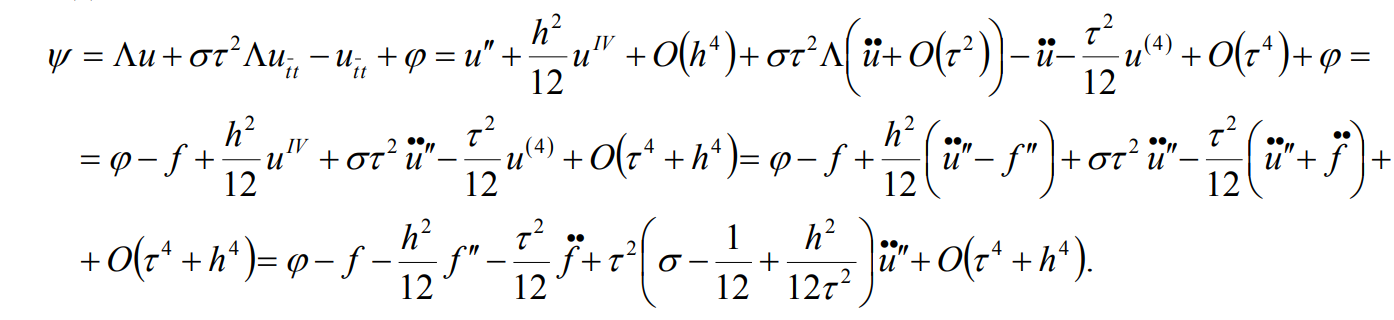
Так как



то



Тогда



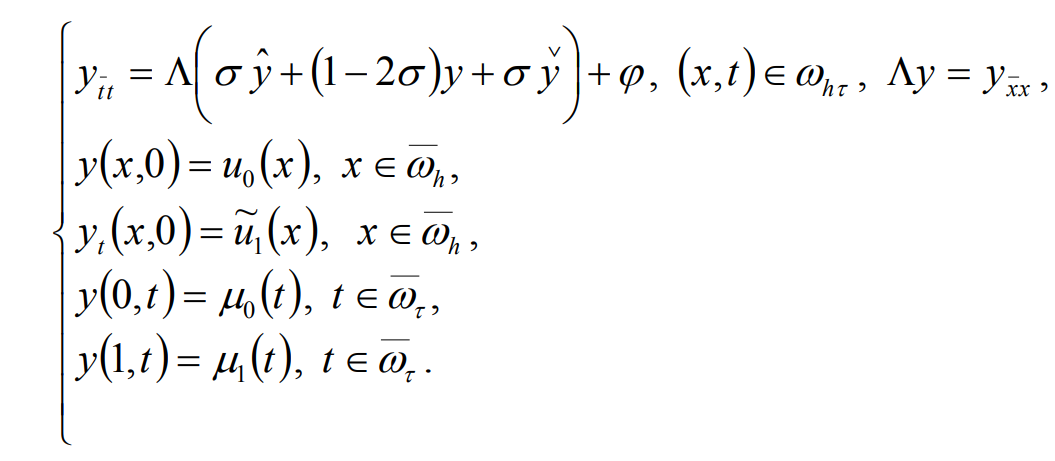
Подставим в данное выражение :

Следовательно, выбрав , получим при выбранном необходимый порядок аппроксимации.

Теперь рассмотрим погрешность аппроксимации второго начального условия:

Следовательно, взяв , получим необходимый порядок аппроксимации.

Итак, мы построили разностную схему с весами, имеющую необходимый порядок аппроксимации:

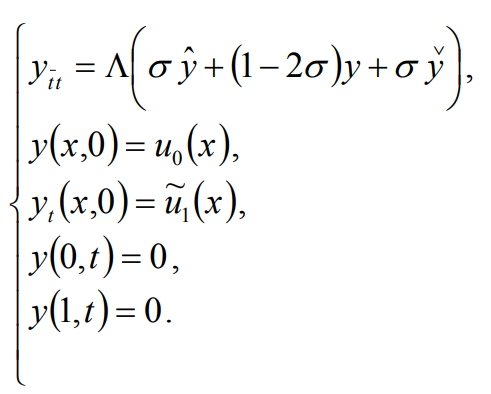


,

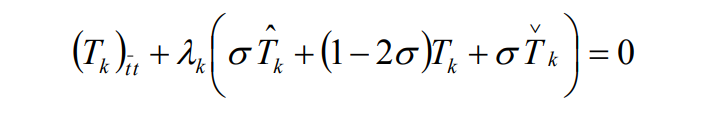
.

# Исследование устойчивости полученной разностной схемы с помощью МРП

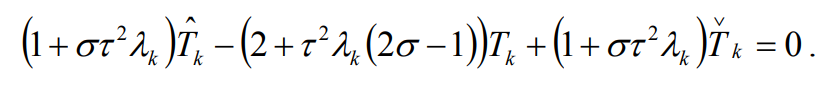
Для исследования услойчивости по начальным данным рассмотрим одногродную задачу с неоднородными начальными условниями:



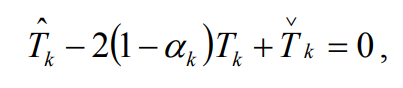
Выполнив разделение переменных, получим задачу:

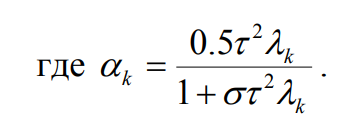


или

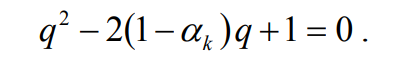


Последнее равенство, разделив почленно на , перепишем в виде:



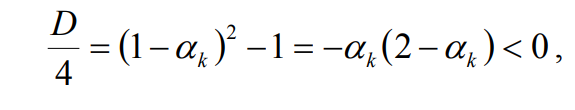


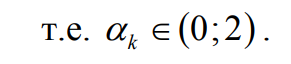
Запишем характеристическое уравнение:

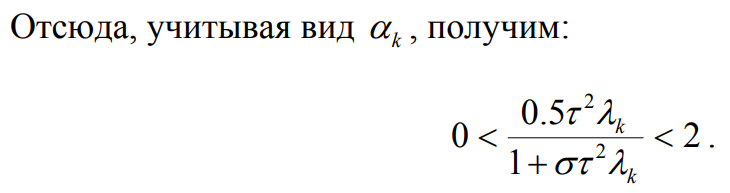


Тогда условия устойчивости

Так как , то условие устойчивости будет выполняться только в том случае, когда комплексно-сопряжённые, а тогда







Решая это неравенство оносительно , найдём условия устойчивости:

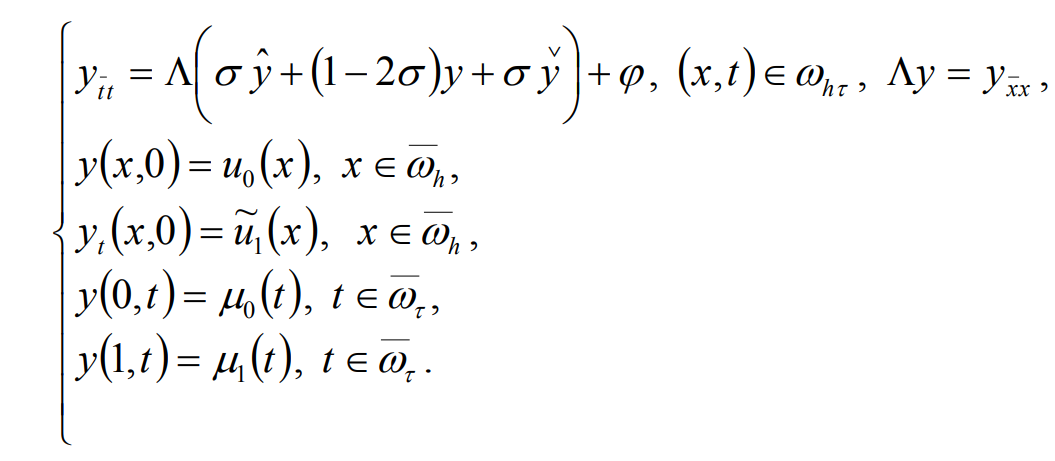
Подставляя сюда , найдём ограничения на шаг при заданном шаге :

Данное неравенство будет выполняться при любых

# Реализация разностной схемы

## Реализация метода прогонки.

Имеем разностную схему следующего вида:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

,

Схема устойчива при любом шаге разбиения. Выберем

Запишем схему в индексной форме:

Будем решать данную задачу, применяя метод прогонки по для каждого временного слоя, последовательно заполняя матрицу значений.

Коэффициенты прогонки в данном случае будут выглядеть следующим образом:

Итак, алгоритм:

1. Заполняем все значения , используя граничные условия.
2. Заполняем два верхних временных слоя , используя начальные условия.
3. Применяем при метод прогонки с записанными выше коэффициентами, последовательно заполняя временные слои.

## Листинг программы.

import math  
class Prog:  
 def \_\_init\_\_(self, N1, N2):  
 self.N1 = N1  
 self.N2 = N2  
 self.res = [.0] \* (N2 + 1)  
 self.h = 1/N1  
 self.tao = 1/N2  
 for i in range(N2+1):  
 self.res[i] = [.0] \* (N1 + 1)  
  
 def sigma(self):  
 return ((1/4)-(self.h\*self.h)/(12\*self.tao\*self.tao))  
 def f(self, x, t):  
 return ((x\*x-t\*t)\*math.exp(-1\*x\*t))  
 def f\_\_(self, x,t):  
 return ((2-4\*t\*x+t\*t\*x\*x-t\*t\*t\*t) \* math.exp(-1 \* x \* t))  
 def u0(self, x):  
 return 1  
 def u1(self, x):  
 return (-x)  
 def miu0(self, t):  
 return(1)  
 def miu1(self, t):  
 return math.exp(-t)  
 def phi(self, x, t):  
 return (self.f(x,t)+((self.h\*self.h)/12)\*self.f\_\_(x,t))  
 def U\_1(self, x):  
 return (self.u1(x)+(self.tao/2)\*self.f(x,0))  
  
 def A(self):  
 return (self.sigma()/(self.h\*self.h))  
 def B(self):  
 return (self.sigma() / (self.h \* self.h))  
 def C(self):  
 return ((2\*self.sigma() / (self.h \* self.h))+(1/(self.tao\*self.tao)))  
 def F(self, i, j):  
 F = self.phi(self.h\*i,self.tao\*j)+(1/(self.tao\*self.tao))\*(2\*self.res[j][i]-self.res[j-1][i])+(1-2\*self.sigma())\*(1/(self.h\*self.h))\*(self.res[j][i+1]-2\*self.res[j][i]+self.res[j][i-1])+self.sigma()\*(1/(self.h\*self.h))\*(self.res[j-1][i+1]-2\*self.res[j-1][i]+self.res[j-1][i-1])  
 return F  
*#####Реализация метода прогонки* def Progonka(self):  
  
 for j in range(self.N2+1):  
 t\_j = self.tao\*j  
 self.res[j][0] = self.miu0(t\_j)  
 self.res[j][self.N1] = self.miu1(t\_j)  
 for i in range(1, self.N1):  
 x\_i = self.h\*i  
 self.res[0][i] = self.u0(x\_i)  
 self.res[1][i] = self.u0(x\_i)+self.U\_1(x\_i)\*self.tao  
  
 alpha = [.0]\*(self.N1+1)  
 beta = [.0]\*(self.N1+1)  
 for j in range(1, self.N2):  
 alpha[1] = 0  
 beta[1] = self.miu0(j\*self.tao)  
 for i in range(1, self.N1):  
 alpha[i+1] = self.B()/(self.C()-self.A()\*alpha[i])  
 beta[i+1] = (self.F(i,j)+beta[i]\*self.A())/(self.C()-self.A()\*alpha[i])  
  
 for i in range((self.N1-1), 0, -1):  
 self.res[j+1][i] = alpha[i+1]\*self.res[j+1][i+1]+beta[i+1]  
 return (self.res)  
  
def print\_res(res, N1, N2):  
 for j in range(N2+1):  
 for i in range(N1+1):  
 print(f"{(res[j][i]):.4f}", end = " ")  
 print("")  
def print\_pogr(res, res\_t, N1, N2):  
 for j in range(N2+1):  
 for i in range(N1+1):  
 print(f"{abs(res[j][i]-res\_t[j][i]):.1E}", end = " ")  
 print("")  
def Test20():  
 p = Prog(20, 20)  
 res = p.Progonka()  
 print("Решение при количестве разбиений 20:")  
 print\_res(res, 20, 20)  
 p\_t = Prog(400, 400)  
 res\_t1000 = p\_t.Progonka()  
 res\_t = [.0] \* 401  
 for i in range(400 + 1):  
 res\_t[i] = [.0] \* (400 + 1)  
 for j in range(21):  
 for i in range(21):  
 res\_t[j][i] = res\_t1000[j \* 20][i \* 20]  
 print("Решение при количестве разбиений 400:")  
 print\_res(res\_t, 20, 20)  
 print("Матрица невязок полученных решений:")  
 print\_pogr(res, res\_t, 20, 20)  
def Test10():  
 p = Prog(10, 10)  
 res = p.Progonka()  
 print("Решение при количестве разбиений 10:")  
 print\_res(res, 10, 10)  
 p\_t = Prog(100, 100)  
 res\_t1000 = p\_t.Progonka()  
 res\_t = [.0] \* 101  
 for i in range(100 + 1):  
 res\_t[i] = [.0] \* (100 + 1)  
 for j in range(11):  
 for i in range(11):  
 res\_t[j][i] = res\_t1000[j \* 10][i \* 10]  
 print("Решение при количестве разбиений 100:")  
 print\_res(res\_t, 10, 10)  
 print("Матрица невязок полученных решений:")  
 print\_pogr(res, res\_t, 10, 10)  
*###################main*Test20()  
Test10()

## Результаты.

*Результаты, полученные при В качестве точного решения рассмотрим решение при*

Решение при количестве разбиений 20:

1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000

1.0000 0.9975 0.9950 0.9925 0.9900 0.9876 0.9851 0.9827 0.9802 0.9778 0.9753 0.9729 0.9705 0.9680 0.9656 0.9632 0.9608 0.9584 0.9560 0.9536 0.9512

1.0000 0.9950 0.9901 0.9851 0.9802 0.9753 0.9704 0.9656 0.9608 0.9560 0.9512 0.9465 0.9418 0.9371 0.9324 0.9278 0.9231 0.9185 0.9140 0.9094 0.9048

1.0000 0.9925 0.9851 0.9778 0.9705 0.9632 0.9560 0.9489 0.9418 0.9347 0.9278 0.9208 0.9140 0.9071 0.9004 0.8936 0.8870 0.8803 0.8738 0.8672 0.8607

1.0000 0.9901 0.9802 0.9705 0.9608 0.9512 0.9418 0.9324 0.9231 0.9140 0.9049 0.8959 0.8870 0.8781 0.8694 0.8608 0.8522 0.8437 0.8353 0.8270 0.8187

1.0000 0.9876 0.9753 0.9632 0.9513 0.9394 0.9278 0.9163 0.9049 0.8936 0.8825 0.8716 0.8608 0.8501 0.8395 0.8291 0.8188 0.8086 0.7985 0.7886 0.7788

1.0000 0.9851 0.9705 0.9560 0.9418 0.9278 0.9140 0.9004 0.8870 0.8738 0.8608 0.8480 0.8353 0.8229 0.8107 0.7986 0.7867 0.7750 0.7634 0.7520 0.7408

1.0000 0.9827 0.9656 0.9489 0.9324 0.9163 0.9004 0.8848 0.8694 0.8544 0.8395 0.8250 0.8107 0.7966 0.7828 0.7692 0.7558 0.7427 0.7298 0.7171 0.7047

1.0000 0.9802 0.9608 0.9418 0.9232 0.9049 0.8870 0.8694 0.8522 0.8354 0.8188 0.8026 0.7867 0.7712 0.7559 0.7409 0.7262 0.7118 0.6977 0.6839 0.6703

1.0000 0.9778 0.9560 0.9348 0.9140 0.8937 0.8738 0.8544 0.8354 0.8168 0.7986 0.7809 0.7635 0.7465 0.7299 0.7136 0.6977 0.6822 0.6670 0.6522 0.6376

1.0000 0.9753 0.9513 0.9278 0.9049 0.8826 0.8608 0.8396 0.8189 0.7987 0.7789 0.7597 0.7410 0.7226 0.7048 0.6874 0.6704 0.6538 0.6377 0.6219 0.6065

1.0000 0.9729 0.9465 0.9209 0.8959 0.8717 0.8480 0.8250 0.8027 0.7809 0.7597 0.7391 0.7191 0.6995 0.6805 0.6621 0.6441 0.6266 0.6096 0.5931 0.5769

1.0000 0.9705 0.9418 0.9140 0.8870 0.8608 0.8354 0.8108 0.7868 0.7636 0.7410 0.7191 0.6978 0.6772 0.6572 0.6377 0.6189 0.6005 0.5828 0.5655 0.5488

1.0000 0.9681 0.9371 0.9072 0.8782 0.8502 0.8230 0.7967 0.7712 0.7466 0.7227 0.6996 0.6772 0.6555 0.6346 0.6142 0.5946 0.5756 0.5571 0.5393 0.5220

1.0000 0.9656 0.9325 0.9004 0.8695 0.8396 0.8108 0.7829 0.7560 0.7300 0.7049 0.6806 0.6572 0.6346 0.6127 0.5916 0.5713 0.5516 0.5326 0.5143 0.4966

1.0000 0.9632 0.9278 0.8937 0.8609 0.8292 0.7987 0.7693 0.7410 0.7137 0.6875 0.6622 0.6378 0.6143 0.5917 0.5699 0.5489 0.5287 0.5092 0.4904 0.4724

1.0000 0.9608 0.9232 0.8870 0.8523 0.8189 0.7868 0.7560 0.7263 0.6979 0.6705 0.6442 0.6189 0.5947 0.5713 0.5489 0.5274 0.5067 0.4868 0.4677 0.4493

1.0000 0.9584 0.9186 0.8804 0.8438 0.8087 0.7751 0.7428 0.7119 0.6823 0.6539 0.6267 0.6006 0.5756 0.5517 0.5287 0.5067 0.4856 0.4654 0.4460 0.4274

1.0000 0.9560 0.9140 0.8738 0.8354 0.7987 0.7635 0.7299 0.6978 0.6671 0.6378 0.6097 0.5829 0.5572 0.5327 0.5093 0.4868 0.4654 0.4449 0.4253 0.4066

1.0000 0.9537 0.9095 0.8673 0.8271 0.7887 0.7522 0.7173 0.6840 0.6523 0.6220 0.5932 0.5657 0.5394 0.5144 0.4905 0.4678 0.4460 0.4253 0.4056 0.3867

1.0000 0.9513 0.9049 0.8608 0.8188 0.7789 0.7409 0.7048 0.6705 0.6378 0.6067 0.5771 0.5489 0.5222 0.4967 0.4725 0.4494 0.4275 0.4066 0.3868 0.3679

Решение при количестве разбиений 400:

1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000

1.0000 0.9975 0.9950 0.9925 0.9900 0.9876 0.9851 0.9827 0.9802 0.9778 0.9753 0.9729 0.9704 0.9680 0.9656 0.9632 0.9608 0.9584 0.9560 0.9536 0.9512

1.0000 0.9950 0.9900 0.9851 0.9802 0.9753 0.9704 0.9656 0.9608 0.9560 0.9512 0.9465 0.9418 0.9371 0.9324 0.9277 0.9231 0.9185 0.9139 0.9094 0.9048

1.0000 0.9925 0.9851 0.9778 0.9704 0.9632 0.9560 0.9489 0.9418 0.9347 0.9277 0.9208 0.9139 0.9071 0.9003 0.8936 0.8869 0.8803 0.8737 0.8672 0.8607

1.0000 0.9900 0.9802 0.9704 0.9608 0.9512 0.9418 0.9324 0.9231 0.9139 0.9048 0.8958 0.8869 0.8781 0.8694 0.8607 0.8521 0.8437 0.8353 0.8270 0.8187

1.0000 0.9876 0.9753 0.9632 0.9512 0.9394 0.9277 0.9162 0.9048 0.8936 0.8825 0.8715 0.8607 0.8500 0.8395 0.8290 0.8187 0.8086 0.7985 0.7886 0.7788

1.0000 0.9851 0.9704 0.9560 0.9418 0.9277 0.9139 0.9003 0.8869 0.8737 0.8607 0.8479 0.8353 0.8228 0.8106 0.7985 0.7866 0.7749 0.7634 0.7520 0.7408

1.0000 0.9827 0.9656 0.9489 0.9324 0.9162 0.9003 0.8847 0.8694 0.8543 0.8395 0.8249 0.8106 0.7965 0.7827 0.7691 0.7558 0.7427 0.7298 0.7171 0.7047

1.0000 0.9802 0.9608 0.9418 0.9231 0.9048 0.8869 0.8694 0.8521 0.8353 0.8187 0.8025 0.7866 0.7711 0.7558 0.7408 0.7261 0.7118 0.6977 0.6839 0.6703

1.0000 0.9778 0.9560 0.9347 0.9139 0.8936 0.8737 0.8543 0.8353 0.8167 0.7985 0.7808 0.7634 0.7464 0.7298 0.7136 0.6977 0.6822 0.6670 0.6521 0.6376

1.0000 0.9753 0.9512 0.9277 0.9048 0.8825 0.8607 0.8395 0.8187 0.7985 0.7788 0.7596 0.7408 0.7225 0.7047 0.6873 0.6703 0.6538 0.6376 0.6219 0.6065

1.0000 0.9729 0.9465 0.9208 0.8958 0.8715 0.8479 0.8249 0.8025 0.7808 0.7596 0.7390 0.7189 0.6994 0.6805 0.6620 0.6440 0.6266 0.6096 0.5930 0.5769

1.0000 0.9704 0.9418 0.9139 0.8869 0.8607 0.8353 0.8106 0.7866 0.7634 0.7408 0.7189 0.6977 0.6771 0.6570 0.6376 0.6188 0.6005 0.5827 0.5655 0.5488

1.0000 0.9680 0.9371 0.9071 0.8781 0.8500 0.8228 0.7965 0.7711 0.7464 0.7225 0.6994 0.6771 0.6554 0.6344 0.6142 0.5945 0.5755 0.5571 0.5393 0.5220

1.0000 0.9656 0.9324 0.9003 0.8694 0.8395 0.8106 0.7827 0.7558 0.7298 0.7047 0.6805 0.6570 0.6344 0.6126 0.5916 0.5712 0.5516 0.5326 0.5143 0.4966

1.0000 0.9632 0.9277 0.8936 0.8607 0.8290 0.7985 0.7691 0.7408 0.7136 0.6873 0.6620 0.6376 0.6142 0.5916 0.5698 0.5488 0.5286 0.5092 0.4904 0.4724

1.0000 0.9608 0.9231 0.8869 0.8521 0.8187 0.7866 0.7558 0.7261 0.6977 0.6703 0.6440 0.6188 0.5945 0.5712 0.5488 0.5273 0.5066 0.4868 0.4677 0.4493

1.0000 0.9584 0.9185 0.8803 0.8437 0.8086 0.7749 0.7427 0.7118 0.6822 0.6538 0.6266 0.6005 0.5755 0.5516 0.5286 0.5066 0.4855 0.4653 0.4460 0.4274

1.0000 0.9560 0.9139 0.8737 0.8353 0.7985 0.7634 0.7298 0.6977 0.6670 0.6376 0.6096 0.5827 0.5571 0.5326 0.5092 0.4868 0.4653 0.4449 0.4253 0.4066

1.0000 0.9536 0.9094 0.8672 0.8270 0.7886 0.7520 0.7171 0.6839 0.6521 0.6219 0.5930 0.5655 0.5393 0.5143 0.4904 0.4677 0.4460 0.4253 0.4056 0.3867

1.0000 0.9512 0.9048 0.8607 0.8187 0.7788 0.7408 0.7047 0.6703 0.6376 0.6065 0.5770 0.5488 0.5220 0.4966 0.4724 0.4493 0.4274 0.4066 0.3867 0.3679

Матрица невязок полученных решений:

0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00 0.0E+00

0.0E+00 1.1E-09 1.7E-08 6.6E-08 1.6E-07 3.2E-07 5.6E-07 8.8E-07 1.3E-06 1.9E-06 2.6E-06 3.4E-06 4.5E-06 5.7E-06 7.1E-06 8.7E-06 1.1E-05 1.3E-05 1.5E-05 1.8E-05 0.0E+00

0.0E+00 2.7E-06 3.1E-06 3.2E-06 3.4E-06 3.7E-06 4.2E-06 4.8E-06 5.7E-06 6.8E-06 8.2E-06 9.9E-06 1.2E-05 1.4E-05 1.7E-05 2.0E-05 2.4E-05 2.8E-05 3.1E-05 2.2E-05 0.0E+00

0.0E+00 6.3E-06 8.7E-06 9.3E-06 9.7E-06 1.0E-05 1.1E-05 1.2E-05 1.3E-05 1.5E-05 1.7E-05 1.9E-05 2.2E-05 2.6E-05 3.0E-05 3.4E-05 3.9E-05 4.3E-05 3.8E-05 1.7E-05 0.0E+00

0.0E+00 9.4E-06 1.5E-05 1.8E-05 1.9E-05 2.0E-05 2.0E-05 2.2E-05 2.3E-05 2.5E-05 2.8E-05 3.1E-05 3.5E-05 4.0E-05 4.5E-05 5.1E-05 5.5E-05 5.2E-05 3.4E-05 1.5E-05 0.0E+00

0.0E+00 1.2E-05 2.2E-05 2.8E-05 3.0E-05 3.2E-05 3.3E-05 3.5E-05 3.7E-05 3.9E-05 4.2E-05 4.6E-05 5.1E-05 5.7E-05 6.2E-05 6.7E-05 6.5E-05 5.0E-05 2.9E-05 1.7E-05 0.0E+00

0.0E+00 1.5E-05 2.8E-05 3.7E-05 4.3E-05 4.6E-05 4.8E-05 5.0E-05 5.3E-05 5.6E-05 5.9E-05 6.4E-05 6.9E-05 7.5E-05 7.9E-05 7.8E-05 6.5E-05 4.4E-05 3.1E-05 1.7E-05 0.0E+00

0.0E+00 1.8E-05 3.3E-05 4.6E-05 5.6E-05 6.2E-05 6.6E-05 6.8E-05 7.1E-05 7.5E-05 7.9E-05 8.4E-05 9.0E-05 9.3E-05 9.2E-05 8.0E-05 6.0E-05 4.4E-05 3.3E-05 1.5E-05 0.0E+00

0.0E+00 2.1E-05 4.0E-05 5.5E-05 6.8E-05 7.8E-05 8.5E-05 8.9E-05 9.3E-05 9.7E-05 1.0E-04 1.1E-04 1.1E-04 1.1E-04 9.6E-05 7.6E-05 5.9E-05 4.9E-05 3.2E-05 1.6E-05 0.0E+00

0.0E+00 2.4E-05 4.5E-05 6.4E-05 7.9E-05 9.3E-05 1.0E-04 1.1E-04 1.2E-04 1.2E-04 1.2E-04 1.3E-04 1.3E-04 1.1E-04 9.4E-05 7.5E-05 6.4E-05 4.9E-05 3.1E-05 1.7E-05 0.0E+00

0.0E+00 2.7E-05 5.1E-05 7.2E-05 9.1E-05 1.1E-04 1.2E-04 1.3E-04 1.4E-04 1.5E-04 1.5E-04 1.5E-04 1.3E-04 1.1E-04 9.3E-05 8.0E-05 6.7E-05 4.8E-05 3.3E-05 1.6E-05 0.0E+00

0.0E+00 3.0E-05 5.6E-05 8.0E-05 1.0E-04 1.2E-04 1.4E-04 1.5E-04 1.6E-04 1.7E-04 1.7E-04 1.6E-04 1.4E-04 1.1E-04 9.9E-05 8.6E-05 6.7E-05 4.9E-05 3.4E-05 1.6E-05 0.0E+00

0.0E+00 3.2E-05 6.2E-05 8.9E-05 1.1E-04 1.3E-04 1.5E-04 1.7E-04 1.8E-04 1.9E-04 1.8E-04 1.6E-04 1.4E-04 1.2E-04 1.1E-04 8.8E-05 6.8E-05 5.3E-05 3.4E-05 1.7E-05 1.1E-16

0.0E+00 3.5E-05 6.7E-05 9.7E-05 1.2E-04 1.5E-04 1.7E-04 1.8E-04 1.9E-04 1.9E-04 1.8E-04 1.6E-04 1.4E-04 1.3E-04 1.1E-04 8.9E-05 7.2E-05 5.4E-05 3.4E-05 1.8E-05 0.0E+00

0.0E+00 3.8E-05 7.2E-05 1.0E-04 1.3E-04 1.6E-04 1.8E-04 1.9E-04 2.0E-04 1.9E-04 1.8E-04 1.6E-04 1.5E-04 1.4E-04 1.1E-04 9.4E-05 7.6E-05 5.4E-05 3.7E-05 1.8E-05 0.0E+00

0.0E+00 4.0E-05 7.7E-05 1.1E-04 1.4E-04 1.7E-04 1.9E-04 2.0E-04 1.9E-04 1.8E-04 1.7E-04 1.7E-04 1.6E-04 1.4E-04 1.2E-04 1.0E-04 7.7E-05 5.7E-05 3.8E-05 1.8E-05 0.0E+00

0.0E+00 4.2E-05 8.2E-05 1.2E-04 1.5E-04 1.7E-04 1.8E-04 1.9E-04 1.8E-04 1.8E-04 1.7E-04 1.7E-04 1.6E-04 1.4E-04 1.2E-04 1.0E-04 8.0E-05 6.1E-05 3.9E-05 2.0E-05 0.0E+00

0.0E+00 4.4E-05 8.4E-05 1.2E-04 1.5E-04 1.7E-04 1.7E-04 1.7E-04 1.7E-04 1.7E-04 1.7E-04 1.6E-04 1.5E-04 1.4E-04 1.2E-04 1.0E-04 8.4E-05 6.2E-05 4.1E-05 2.0E-05 5.6E-17

0.0E+00 4.5E-05 8.4E-05 1.2E-04 1.4E-04 1.5E-04 1.6E-04 1.6E-04 1.6E-04 1.7E-04 1.6E-04 1.5E-04 1.5E-04 1.4E-04 1.2E-04 1.1E-04 8.6E-05 6.4E-05 4.3E-05 2.1E-05 0.0E+00

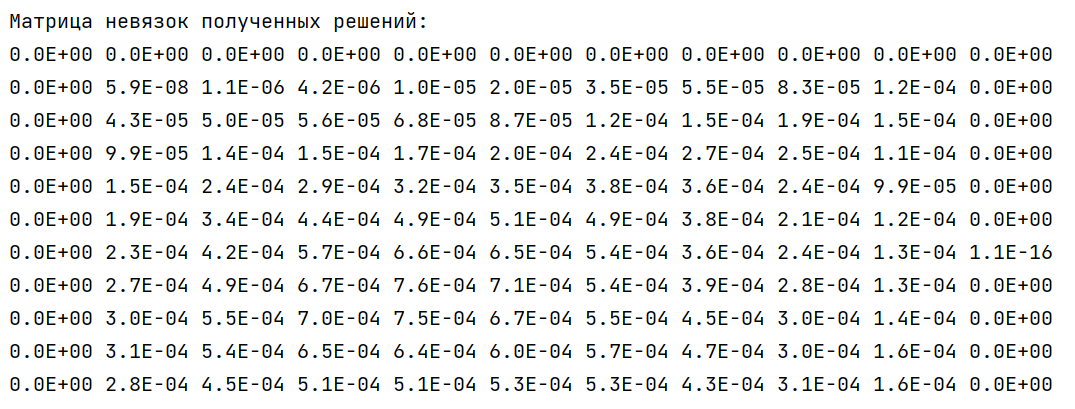
0.0E+00 4.3E-05 8.0E-05 1.1E-04 1.2E-04 1.3E-04 1.4E-04 1.4E-04 1.5E-04 1.5E-04 1.5E-04 1.4E-04 1.4E-04 1.3E-04 1.2E-04 1.0E-04 8.5E-05 6.6E-05 4.4E-05 2.2E-05 0.0E+00

0.0E+00 3.8E-05 6.8E-05 8.9E-05 1.0E-04 1.1E-04 1.2E-04 1.3E-04 1.4E-04 1.4E-04 1.4E-04 1.4E-04 1.3E-04 1.2E-04 1.1E-04 9.7E-05 8.2E-05 6.5E-05 4.3E-05 2.2E-05 0.0E+00

*Также результаты при В качестве точного решения рассмотрим решение при*

**

**

**

## Выводы.

Хоть разностные операторы, используемые в схеме, и обладают вторым порядком, а разностный оператор, применённый для начального условия – первым, была построена схема, имеющая порядок аппроксимации . Такой порядок был достигнут засчёт выбора коэффициентов , и . При данном параметре схема также оказалась устойчива при любом разбиении

Рассмотрим невязку полученных результатов. Так как на первом временном слое и на границах значения были вычислены точно, имеем в этих точках нулевую невязку. И для шагов , и для шагов невязка составляет примерно , что меньше, чем ожидаемая погрешность Это может быть обусловлено величиной коэффициента при главном члене погрешности. Также заметим, что на первых временных слоях погрешность на несколько порядков меньше.