БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Лабораторная работа №3

Решение второй краевой задачи для уравнения теплопроводности

**Преподаватель:** Полевиков Виктор Кузьмич

**Студент:** Ярош Илья

3 курс 11 группа

**2020 г.**

**Постановка задачи**

**Краткая теория**

Наше условие имеет такой вид:

Перепишем ее в виде схемы явного для заданной краевой задачи:

Коэффициенты и находим по соответствующим формулам:

*, ,*

По этим формулам будем искать значения во внутренних узлах сетки

На нижнем уровне, , на левой границе , на правой границе, находится из выражения:

По принципу максимума докажем устойчивость РС

*, , ,*

РС удовлетворяет принципу максимума и является устойчивой при

Найдем порядок аппроксимации во внутренних узлах:

Теперь найдем порядок аппроксимации на границах:

Мы получили первый порядок аппроксимации, поэтому сейчас надо повысить его до второго порядка, обозначим:

Выберем , тогда

**Листинг программы**

**public** **class** MKR {

**static** **double** u (**double** x, **double** t) {

**return** Math.*pow*(Math.*cos*(4 \* x),2) \* Math.*exp*(t \* t);

}

**static** **double** f (**double** x, **double** t) {

**return** 2\*Math.*exp*(t\*t)\*(t\* Math.*pow*(Math.*cos*(4 \* x),2)-(176\*

( Math.*pow*(Math.*cos*(4 \* x),2)- Math.*pow*(Math.*sin*(4 \* x),2))));

}

**static** **double** u0 (**double** x) {

**return** Math.*pow*(Math.*cos*(4 \* x),2);

}

**static** **double** mu0 (**double** t) {

**return** Math.*exp*(t\*t);

}

**static** **double** mu1 (**double** t) {

**return** -8\*Math.*exp*(t \* t)\* Math.*sin*(4) \* Math.*cos*(4);

}

**static** **double** *h* = 0.01;

**static** **double** *tau* = *h*\**h*/(2\*11);

**public** **static** **void** main (String [] args) {

**int** N = (**int**) (1/(*h*));// кол-во разбиений по х

**int** M = (**int**) (1/(*tau*)); // кол-во точек

**double**[][]y = **new** **double** [M][N]; // вся наша сетка

System.***out***.println(M + " " + N);

System.***out***.println(*tau*);

**for** (**int** i = 0; i < N; i++) {

y[0][i] = *u0*(*h*\*i) + *h* \* i;

}

**for** (**int** i = 0; i < M; i++) { //

y[i][0] = *mu0*(i\**tau*) + *tau* \* i;

}

**for** (**int** i = 0; i < M-1; i++){ //

//y[i+1][N-1] = (0.5\*h\*((y[i+1][N-2]- y[i][N-2])/tau)- mu1((i+1)\*tau))\*h + y[i+1][N-2];

**for** (**int** j = 1; j <N-1; j++) {

// y[i][j+1]= tau\*((y[i-1][j]-2\*y[i][j]+y[i+1][j])/(h\*h)+f(i\*h, (j+1)\*tau))+y[i][j];

//y[i][j+1]= tau\*(y[i-1][j]/(h\*h) + y[i][j]\*(1/tau - 2/(h\*h)) +y[i+1][j]/(h\*h)+ f(i\*h, tau\*(j+1)));

//y[i+1][j]= y[i][j] + tau\*(f(j\*h, i\*tau)+ (y[i][j-1] - 2\*y[i][j] + y[i][j+1])/(h\*h));

// y[i+1][j] = tau\* 10/(h\*h)\*y[i][j-1]+ (1-tau\*20/(h\*h))\*y[i][j]+ tau\*10/(h\*h)\*y[i][j+1] + tau\*f(i\*h, tau\*(j+1));

y[i+1][j] = *tau*\*( (11/(*h*\**h*))\*y[i][j-1]+ (1 / *tau* - 22/(*h*\**h*))\*y[i][j]+ (11/(*h*\**h*)\*y[i][j+1] + *f*(j\**h*, *tau*\*(i))));

}

y[i+1][N-1] = -y[i+1][N-2] + *h*\**mu1*((i+1)\**tau*);

}

//for (int i = 1; i < M-1; i++) { y[N-1][i+1] = (0.5\*h\*((y[N-2][i+1]- y[N-2][i])/tau)- mu1((i+1)\*tau))\*h + y[N-2][i+1];}

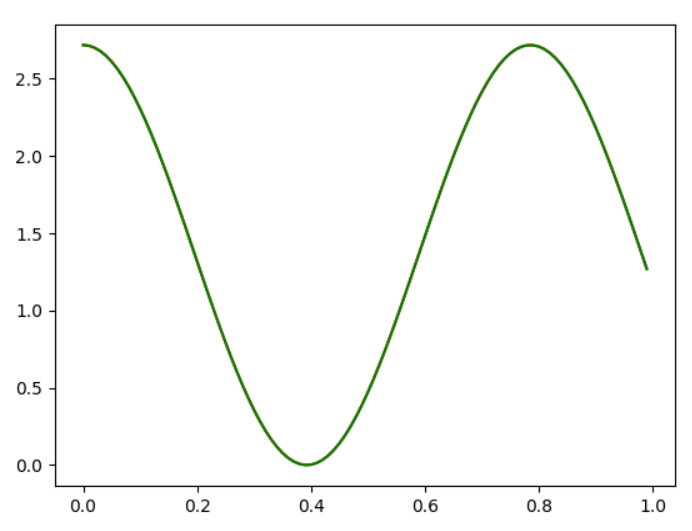
**for** (**int** i = 0; i < N; i++) {

System.***out***.println( *u*(i\**h*, (M-1)\**tau*) + " \t\t " +*u*(i\**h*, 0.99988888888\*(M-1)\*(*tau* )) /\*+ y[M - 1][N-i-1]\*/);

}

}}

**Результаты**



2.718257116974547 2.7176531659427865

2.713910224672007 2.7133072394466358

2.700897353040897 2.7002972590568284

2.67930174005186 2.6787064442499298

2.6492615239307256 2.648672902559778

2.61096885954527 2.6103887461605684

2.5646686892698125 2.564098863004543

2.510657176189912 2.5100993503748015

2.4492798096693087 2.448735620873156

2.3809291953970115 2.3804001929582324

2.3060425440506984 2.305530180166846

2.2250988746404063 2.2246044950790633

2.1386159504215723 2.1381407859120305

2.0471469669771163 2.046692125337909

1.9512770136535502 1.9508434727061956

1.8516193309858402 1.8512079323001187

1.7488113880507297 1.7484228315615102

1.6435108048400784 1.6431456443701236

1.5363911457371133 1.5360497854545039

1.4281376110029829 1.4278203028358214

1.3194426538334056 1.3191494958583359

1.2110015510212877 1.2107324868361393

1.1035079555579501 1.1032627746425199

0.9976494596211514 0.9974277986838137

0.8941031963316298 0.8939045416331641

0.793531508411917 0.7933551990516851

0.6965777114532302 0.6964229435966848

0.6038619788910696 0.6037278109115526

0.5159773750116388 0.5158627335135784

0.4334860613643065 0.4333897480492857

0.35691570084612595 0.35683640017790313

0.28675608246000034 0.2866923700794519

0.2234559883365198 0.22340634017267758

0.16742032305984159 0.16738312507874617

0.11900752366013166 0.11898108218914168

0.07852726683977529 0.07850981940129236

0.04623848809928642 0.0462282146845972

0.022347725433748366 0.022342760144869352

0.007007798194478909 0.007006241179535954

3.168295666933282E-4 3.1675917249018176E-4

0.0023176189159748607 0.00231710398001579

0.012997368018392316 0.01299448022173178

0.03228776292545783 0.03228058912736115

0.060065410940269126 0.06005206541578436

0.09615262990967843 0.09613126641184086

0.14031858478371406 0.14028740835322628

0.1922807641721512 0.19223804261899707

0.2517067874533094 0.25165086243685886

0.3182165308757472 0.3181458285124772

0.39138455905306063 0.3912975999840384

0.4707428462985107 0.4706382551522426

0.5557837703922334 0.5556602845823488

0.645963359631132 0.6458198374326276

0.7407047723914161 0.7405402002437991

0.8394019869464616 0.8392154859370787

0.9414236779377475 0.941214509423826

1.046117254702687 1.045884825036128

1.1528130356278323 1.1525568999525326

1.2608285318258432 1.2605483969232565

1.369472812735302 1.3691685389000368

1.47805092571843 1.4777225276518868

1.585868341386372 1.5855159881027052

1.6922353962171306 1.691859409962135

1.7964717040485574 1.7960725582383936

1.8979105082279022 1.8974888244208499

1.9959029465790108 1.9954594904996164

2.089822201905945 2.0893578785469975

2.1790675114839555 2.178583359317621

2.2630680098907527 2.262565194225898

2.3412863805970767 2.3407661861252667

2.413222292958826 2.4126861155366837

2.478415602625757 2.4778649403462434

2.5364492948950326 2.5358857385047657

2.586952152182212 2.586377374906115

2.629601128546972 2.629016875385346

2.6641234160846974 2.663531492651173

2.6902981899660876 2.68970045093786

2.7079580209624714 2.7073563582166953

2.7169899464215033 2.716386276933736

2.7173361928426423 2.7167324464247415

2.7089945454303956 2.708392652386309

2.6920183622614515 2.6914202410398596

2.6665162329750793 2.6659237778978704

2.6326512841700125 2.632066353315087

2.590640135950902 2.5900645392668147

2.5407515162988847 2.5401870040273513

2.4833045421295687 2.482752793609891

2.418666678033794 2.41812929096082

2.3472513857582586 2.3467298659625864

2.269515479461326 2.269011231277128

2.1859562036613345 2.185470520943412

2.0971080525685832 2.096642110416109

2.0035393511463986 2.0030941983862935

1.9058486197708426 1.905425172248862

1.8046607457428174 1.8042597804652643

1.7006229861418325 1.7002451363053686

1.594400827589533 1.594046578530881

1.4866737294063557 1.4863434154978004

1.3781307773906148 1.3778245799011617

1.2694662760209643 1.2691842219568361