Jaworski Jakub Gr. 22 – Sprawozdanie

Przedmiot: Metody Numeryczne

Prowadzący: dr. Hab. Janusz Chwastowski

Wykład: dr hab. Inż. L.Bieniasz

1. Założenia:

Rozwiązanie równania różniczkowego cząstkowego rzędu 2, mogącego opisywać transport ciepła, w ośrodku wokół kuli o promieniu r, przy współczynniku transportu ciepła D,po raptownym obniżeniu temperatury w chwili t=0;

Równanie :

Warunek początkowy: U(x,0)=1

Warunki brzegowe: U(r, t)=0, U(+∞, t)=1

Rozwiązanie analityczne dane jest wzorem:

Dla obliczeń przyjmujemy:

Warunek brzegowy:

,oraz parametry:

=2, r=1, a=10, D=1, λ=D\*dt/=1

Do obliczeń numerycznych wykorzystywać będziemy: Metodę pośrednią Laasonen’a, do dyskretyzacji, oraz Algorytmu Thomasa, dal rozwiązania układów równań algebraicznych.

2.Dyskretyzacja:

Korzystamy ze wzoru:

Korzystamy z odpowiednich przybliżeń pochodnych.

Grupujemy wyrazy.

3.Wykresy

Wykres zależności maksymalnej wartości bezwzględnej błędu obserwowanej dla , w funkcji kroku przestrzennego h.

Wykres pokazuje zależność błędu od kroku dla kroku z zakresu 1 – 1/150. Z wykresu widać że błąd maleje wraz ze spadkiem wartości kroku przestrzennego, tym szybciej im mniejszy jest krok h. Na wykresie występują pewne odchylenia od normy („schodki”) spowodowane najprawdopodobniej dodatkowymi bledami dzielenia i zaokrągleń dla pewnych wartości h).

Wykresy rozwiązań numerycznych i analitycznych dla kilku wybranych wartości czasu t.

Poniższe wykresy wykonane są dla kroku h=0.1 (w przypadku większego kroku czytelność wykresów znacząco by zmalała).

t=0.02

t=1

t=1.5

Dla wszystkich wykresów w tym punkcie odległość między rozwiązaniem analitycznym a numerycznym jest największa dla x najbardziej zbliżonego 1 (w tym przypadku 1.1). Warto też zauważyć podobieństwo wykresów obu rozwiązań, różnica jest ledwo widoczna , wykresy w większości praktycznie się pokrywają.

Wykres zależności maksymalnej wartości bezwzględnej błędu w funkcji czasu t.

Wartość błędu wyraźnie spada wraz z obliczeniami dla kolejnych t. Spadek ten jest najbardziej zauważalny dla małych wartości t .Otrzymanie wykresu w którym błąd rósłby wraz z krokiem t, najprawdopodobniej świadczyło by o błędnym wykorzystaniu wzorów, lub błędzie maszynowym znacznie przewyższającym rzeczywisty.

4.Wnioski

Obliczanie numeryczne wartości równania różniczkowego przy pomocy dyskretyzacji Laasonen’a, oraz metody Thomasa jest metodą dokładną, nie znaczy to jednak że obliczenia te nie są obarczone błędem. Najważniejszymi składnikami błędu są błąd maszynowy, oraz błędy przybliżeń. Obu z tych błędów nie jesteśmy w stanie całkowicie wykluczyć. Błąd powyższych metod dla kroku h=0.1, już dla t>1 (100 krok) jest rzędu , co jest dokładnością bardziej niż zadowalającą. Warto też zauważyć że jesteśmy w stanie osiągnąć taki błąd także dla mniejszych wartości t zmniejszając krok maszynowy. Oczywiście dalsze zmniejszanie kroku maszynowego w koń przestało by przynosić wzrost dokładności związane jest to z ograniczeniami typów danych zmiennoprzecinkowych których używamy. Zdefiniowanie nowych typów danych mogło by być w tym wypadku pomocne, ale znacząco wydłużyło by obliczenia (operacje na takich liczbach musiały by być symulowane).

Podsumowując co jest najważniejsze otrzymujemy dobrą dokładność rzędu od dla małych t, oraz h do, przy niewielkim czasie obliczeń.

5.Program

#include <cstdio>

#include <iostream>

#include <cstdlib>

#include "calerf.cpp"

#include <cmath>

using namespace std;

// stale wspolczynniki

double dx=0.1;//dx=0.005;

double dt=0.01;//=0.000025;

const double D=1;

const double tmax=2;

const double r=1;

const double a=10;

//double lambda=dt/dx/dx\*D;

const double lambda=1;

//ustalanie wielkosci tablic

int SIZE=(int)(a/dx-1);

int SIZE\_T=(int)(tmax/dt +1);

//ustawienia wydrukow

bool analit=false;//rozwiazanie analityczne

bool oblicz=false;//obliczone rozwiazanie

bool errors=false;//blad

bool errorodt=true;//zaleznosc bledu od t

bool errormax=false;//maxymalny blad

bool tloopa=false;//krok h

bool wybran=false;///wybrane wartosci

int index=1;

double Ts(int nr,double x)

{//mozliwe wyniki dyskretyzacji Laasonenem z uzyciem roznych przyblizen pochodnych

switch(nr){

case 1 : return lambda\*(1-dx/x);

case 2 : return -(1+2\*lambda);

case 3 : return lambda\*(1+dx/x);

case 4 : return 1;

}

/\*switch(nr){

case 1 : return lambda;

case 2 : return -1-2\*lambda\*(1+dx/x);

case 3 : return lambda\*(1+2\*dx/x);

case 4 : return 1;

}\*/

/\*switch(nr){

case 1 : return lambda\*(1-2\*dx/x);

case 2 : return -1-2\*lambda\*(1-dx/x);

case 3 : return lambda;

case 4 : return 1;}\*/

}

//warunki brzegowe i poczatkowy

const double lewa=0;

const double poczatek=1;

double prawy(double t){return 1.0-r/(r+a)\*erfc(a/2.0/sqrt(D\*t));}

double analityczne(double x,double t){return 1.0-r/x\*erfc((x-r)/2.0/sqrt(D\*t));}

//dunkcje omocnicze do wypisywania macierzy

void wypisz(double \*\*a,int m)

{

for(int i=0;i<m;i++){for(int j=0;j<m;j++)cout<<a[i][j]<<" ";cout<<"\n";}

cout<<endl;

} ;

void wypisz(double \*\*a,double \* b,int m)

{

for(int i=0;i<m;i++){for(int j=0;j<m;j++)cout<<a[i][j]<<" ";cout<<" "<<b[i];cout<<"\n";}

cout<<endl;

} ;

//macierz trojprzekatniowa jest realizowana przez macierz m^2 moglibyśmy użyć trzech wektorw o wielkości m, zmniejszyło by to zapotrzebowanie pamięci operacyjnej, ale zaciemniło by to rónież przykłąd. Dla celów edukacyjnych zostajemy przy macirzy m^2

void thomas(double \*\* a,double \* b,int n,double \* x=0){

if(x==0)x=new double [n];

for(int i=1;i<n;i++)a[i][i]=a[i][i]-a[i-1+1][i-1]/a[i-1][i-1]\*a[i-1][i-1+1];;

for(int i=1;i<n;i++)b[i]=b[i]-a[i-1+1][i-1]/a[i-1][i-1]\*b[i-1];

x[n-1]=b[n-1]/a[n-1][n-1];

for(int i=n-2;i>=0;i--)x[i]=(b[i]-a[i][i+1]\*x[i+1])/a[i][i];

//cout<<"\n";

//for(int j=0;j<n;j++)cout<<x[j]<<" ";cout<<"\n";

};

int main(int argc,char \*argv[])

{

FILE \* pFile;

pFile = fopen ("dane.csv","w");

if(argc>=2){tloopa=true;analit=false;oblicz=false;errors=false;}

do{

if(argc>=2){dx=index/150.0;

index++;

if(index==151)tloopa=false;

dt=dx\*dx;

//ustalanie wielkosci tablic

SIZE=(int)(a/dx-1);

SIZE\_T=(int)(tmax/dt +1);}

//allokacja danych

cout<<SIZE<<" "<<SIZE\_T<<endl;

double \*\* A=new double\* [SIZE];

double \* b=new double [SIZE];

for(int i=0;i<SIZE;i++)A[i]=new double[SIZE];

double \*\* Ans=new double\* [SIZE\_T];

for(int i=0;i<SIZE\_T;i++)Ans[i]=new double[SIZE];

double \*\* Err=new double\* [SIZE\_T];

for(int i=0;i<SIZE\_T;i++)Err[i]=new double[SIZE];

double x;

//for(int i=0;i<SIZE;i++)for(int j=0;j<SIZE;j++)A[i][j]=0;//////

//przepisywanie elementow poczatkowych do macierzy

for(int i=0;i<SIZE;i++)Ans[0][i]=poczatek;

//właściwa częśc programu obliczająca kolejne przybliżenia

for(int k=0;k<SIZE\_T-1;k++){

x=r+dx;

A[0][0]=Ts(2,x);

A[0][1]=Ts(3,x);

b[0]=-lewa\*Ts(1,x)-Ans[k][0]\*Ts(4,x);

x+=dx;

for(int i=1;i<SIZE-1;i++)

{

A[i][i-1]=Ts(1,x);

A[i][i]=Ts(2,x);

A[i][i+1]=Ts(3,x);

b[i]=-Ans[k][i]\*Ts(4,x);

x+=dx;

}

A[SIZE-1][SIZE-2]=Ts(1,x);

A[SIZE-1][SIZE-1]=Ts(2,x);

b[SIZE-1]=-prawy(dt\*k)\*Ts(3,x)-Ans[k][SIZE-1]\*Ts(4,x);

thomas(A,b,SIZE,Ans[k+1]);

}

//wypisywanie obliczonych wyników

if(oblicz){

fprintf(pFile,"obliczone wyniki;\n");

fprintf(pFile,";");

for(int i=0;i<SIZE;i++)fprintf(pFile,"%f;",(i+1)\*dx+1);

fprintf(pFile,"\n");

for(int k=0;k<SIZE\_T;k++){

if((!wybran)||k==2||k==100||k==150){fprintf(pFile,"%f;",k\*dt);

for(int i=0;i<SIZE;i++)fprintf(pFile,"%f;",Ans[k][i]);

fprintf(pFile,"\n");}

}

fprintf(pFile,"\n");

fflush(pFile);

}

//os x rozwiazania analityczne

if(analit){

fprintf(pFile,"rozwiazania analityczne;\n");

fprintf(pFile,";");

for(int i=0;i<SIZE;i++)fprintf(pFile,"%f;",(i+1)\*dx+1);

fprintf(pFile,"\n");}

//obliczenia bledu+wypis rozwiazan analitycznych

double helpt=dt;double helpx;

for(int i=0;i<SIZE\_T-1;helpt+=dt,i++)

{

if(analit&&((!wybran)||i+1==2||i+1==100||i+1==150))fprintf(pFile,"%f;",helpt);

helpx=r+dx;

for(int j=0;j<SIZE;helpx+=dx,j++){if(analit&&((!wybran)||i+1==2||i+1==100||i+1==150))fprintf(pFile,"%f;",analityczne(helpx,helpt));Err[i][j]=abs(Ans[i+1][j]-analityczne(helpx,helpt))/Ans[i+1][j];}

fflush(pFile);

if(analit&&((!wybran)||i+1==2||i+1==100||i+1==150))fprintf(pFile,"\n");

}

if(analit)fprintf(pFile,"\n");

//wypis bledu

if(errors){

fprintf(pFile,"blad;\n");

fprintf(pFile,";");

for(int i=0;i<SIZE;i++)fprintf(pFile,"%f;",(i+1)\*dx+1);

fprintf(pFile,"\n");

for(int k=0;k<SIZE\_T-1;k++){

fprintf(pFile,"%f;",(k+1)\*dt);

for(int i=0;i<SIZE;i++){fprintf(pFile,"%f;",Err[k][i]);}

fprintf(pFile,"\n");

}

fprintf(pFile,"\n");

}

//wypis max bledu dla tmax

if(index>1)

{double max=0;

for(int i=0;i<SIZE;i++)if(Err[SIZE\_T-2][i]>max)max=Err[SIZE\_T-2][i];

fprintf(pFile,"%f;%f\n",dx,max);}

//wypis maxymalnego t od czasu

if(errorodt){double max;

for(int i=1;i<SIZE\_T;i++)fprintf(pFile,"%f;",i\*dt);

fprintf(pFile,"\n");

for(int i=0;i<SIZE\_T-1;i++){

max=0;

for(int j=0;j<SIZE;j++)if(Err[i][j]>max)max=Err[i][j];

fprintf(pFile,"%f;",max);

}

fprintf(pFile,"\n");

}

//sprzatanie

delete [] A;

delete [] Err;

delete [] b;

delete [] Ans;

}while(tloopa);

system("pause");

if (pFile!=NULL)fclose (pFile);

}