

**دانشکده مهندسی مکانیک**

**رشته مهندسی مکانیک**

**گرايش تبدیل انرژی**

پروژه دوم دینامیک سیالات محاسباتی

**حل عددی معادله مشتقات جزئی انتقال حرارت**

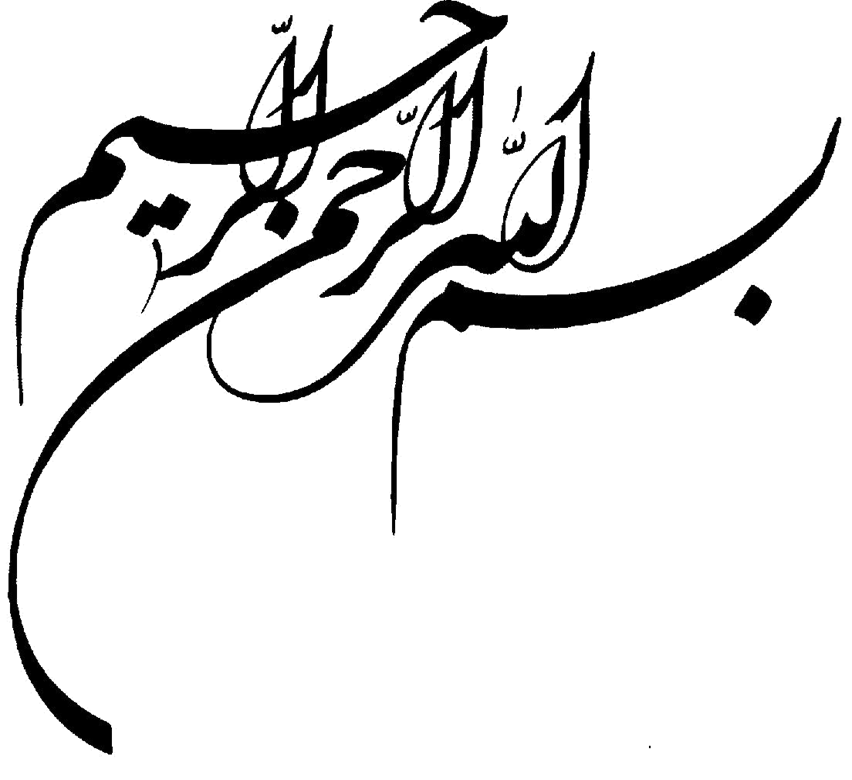
**دکتر خالقی**

**نگارنده**

سجاد خدادادی

9665612002

اردیبهشت 1397



**فهرست مطالب**

[پروژه دوم دینامیک سیالات محاسباتی أ‌](#_Toc514416717)

[حل عددی معادله مشتقات جزئی انتقال حرارت أ‌](#_Toc514416718)

[چکیده ث‌](#_Toc514416719)

[شرح مسئله 1](#_Toc514416720)

[پیشگفتار 1](#_Toc514416721)

[شرح مسئله 2](#_Toc514416722)

[تولید شبکه غیریکنواخت 4](#_Toc514416723)

[روش صریح 5](#_Toc514416724)

[روش ADI 7](#_Toc514416725)

[روش ADE 10](#_Toc514416726)

[کد روش صریح برای مش غیریکنواخت و خواص فیزیکی ثابت 13](#_Toc514416727)

[کد روش صریح برای مش غیریکنواخت و خواص فیزیکی متغییر 14](#_Toc514416728)

[کد روش ADI برای مش غیریکنواخت و خواص فیزیکی متغییر 16](#_Toc514416729)

[کد روش ADE برای مش غیریکنواخت و خواص فیزیکی متغییر 20](#_Toc514416730)

[نتایج 23](#_Toc514416731)

[نتیجه گیری 28](#_Toc514416732)

# چکیده

در این پروژه به حل عددی معادله انتقال حرارت دوبعدی ناپایا با استفاده از روش صریح و ADI و ADE پرداخته شده است. برای دستگاه معادلات تشکیل شده در روش ADI از TDMA استفاده و در نهایت نتایج حاصل از این سه روش با هم مقایسه شده اند. کلیه کدها به زبان c++ نوشته شده‌اند.

# شرح مسئله

## پیشگفتار

معادله­ای که شامل مشتقات جزئی یک تابع مجهول که خود تابع دو یا چند متغیر مستقل است، معادله مشتق جزئی نامیده می­شود. معادلات دیفرانسیل با مشتقات جزئی به دسته‌ای از معادلات دیفرانسیل گفته می‌شود که در آن‌ها توابع مجهول بر حسب چند متغیر مستقل به همراه مشتق پاره‌ای توابع نسبت به آن متغیرها شرکت داشته ‌باشند. فرم کلی معادلات دیفرانسیل جزئی مرتبه دو به صورت زیر می­باشد:

|  |  |
| --- | --- |
| (1) |  |

برحسب رابطه بین A,B,C انواع مختلف معادلات تشکیل می­شوند که این معادلات به همراه اسامی آن­ها در جدول2-1 آورده شده است.

جدول 2-1-انواع معادلات دیفرانسیل جزئی مرتبه دوم

|  |  |  |  |
| --- | --- | --- | --- |
| نام معادله مدل | معادله مدل | نوع معادله |  |
| معادله لاپلاس |  | بیضوی[[1]](#footnote-2) | <0 |
| معادله انتقال حرارت |  | سهموی[[2]](#footnote-3) | =0 |
| معادله موج |  | هذلولوی[[3]](#footnote-4) | >0 |

معادله­ای که در این پروژه بررسی می شود از نوع parabolic می باشد که معادله انتقال حرارت دوبعدی ناپایا می­باشد. این معادله به صورت زیر است:

|  |  |
| --- | --- |
| (2) |  |

روش های مختلفی برای حل این معادلات وجود دارد. یکی از کارامدترین این روش‌ها حل به کمک رهیافت عددی است. یکی از رایج ترین روش‌های عددی برای حل این معادلات استفاده از روش اختلاف محدود است. در این روش مشتقات موجود در معادله‌ی PDE بر اساس روش اختلاف محدود گسسته‌سازی می‌شوند و معادلات حاکم بر مسئله را می‌توان به صورت صریح و با استفاده از نتایج موجود حل کرد یا تلاشی انجام داد که داده‌ها به صورت یک دستگاه معادلات در‌می‌آید. این دستگاه معادله را می‌توان به روش‌های مختلف حل کرد.

## شرح مسئله

علت وریشه ی انتقال حرارت هدایتی را بایستی در انتقال انرژی بین مولکولها و یا اتمهای یک جسم جستجو نمود. مولکولهای سیال دارای انرژی حرکت انتقالی تصادفی و همچنین انرژی چرخشی و ارتعاشی می باشند که دمای بالاتر نشانه ی انرژی بالاتر ذره است. در گاز ساکن بین دو صفحه در اثر برخورد و تبادل انرژی مولکولها یک جریان انرژی از مولکولهای با انرژی بیشتر به مولکول‌های با انرژی کمتر وجود دارد که همان انتقال حرارت هدایتی می‌باشد و به این پدیده توزیع یا پخش حرارت نیز می‌گویند. در مورد هدایت در جامدات، انتقال حرارت به‌دلیل انتقال انرژی بین الکترون‌های اجسام هادی و یا انتقال انرژی بصورت امواج لاتیک در اجسام عایق صورت می گیرد. در این روش یا مود انتقال حرارت، به دلیل برخوردهای مولکولی انرژی از ذرات پر انرژی به ذرات کم انرژی منتقل می شود . هدف به دست آوردن توزیع دما در یک صفحه مربع 2 بعدی است. انتقال حرارت در این صفحه به صورت دائم و از نوع رسانش است. معادله حاکم بر این مسئله عبارتند از:

|  |  |
| --- | --- |
| (3) |  |

اگر خواص ثابت باشند معادله (3) را می‌توان به فرم زیر نوشت:

|  |  |
| --- | --- |
| (4) |  |

که در آن می‌باشد.

برای شرایط مرزی و اولیه این مساله خواهیم داشت:

|  |  |
| --- | --- |
| موقعیت شرط مرزی | درجه حرارت(کلوین) |
| دیوار سمت راست | 25 |
| دیوار سمت چپ | 200 |
| دیوار بالا | صفر |
| دیوار پایین |  |
| J:\Drawing1.emf  شکل1- هندسه کلی مساله | |

اگر k را متغییر خطی از دما به صورت زیر فرض شود، خواهیم داشت:

|  |  |
| --- | --- |
| (5) |  |

و با توجه به این تعریف می‌توان نوشت:

|  |  |
| --- | --- |
| (6) |  |
| (7) |  |
| (8) |  |
| (9) |  |

## تولید شبکه غیریکنواخت

به دلیل اینکه اصولا در نزدیک مرز گرادیان‌ها شدیدتر می‌باشند برای دنبال کردن تغیرات در این نواحی و همچنین توجه به بالا نرفتن سلول‌های محاسباتی اقدام به تولید شبکه‌های غیریکنواخت می‌شود. به این صورت که سلول‌ها در نزدیکی مرز ریز شده و با فاصله گرفتن از آن سایز آنها بزرگتر می‌شود. این کار با کمک تصاعد هندسی صورت می‌گیرد به این صورت که مجموع n جمله اول این تصاعد برابر طول مورد نظر می‌باشد. لذا با مشخص بودن طول و همچنین تعداد گره‌های محاسباتی می‌توان قدر نسبت را بدست آورد. همچنین می‌توان با معلوم فرض کردن قدر نسبت و طول مورد نظر تعداد گره ها را محاسبه نمود. در این هندسه از ابتدای حفره تا وسط آن فرض شده که با نسبت q بزرگ شده و از نصف تا انتهای صفحه این قدر نسبت به میزان 1/q تقلیل یافته است.

|  |  |
| --- | --- |
| (10) |  |

و لذا n با استفاده از فرمول زیر محاسبه می‌شود:

|  |  |
| --- | --- |
| (11) |  |

در مش غیر یک‌نواخت تفاضل مرکز سلول تا سلول‌های مجاورش یکسان نیست و در گسسته‌سازی این موضوع اهمیت پیدا خواهد کرد.

|  |  |
| --- | --- |
| (12) |  |
| (13) |  |
| (14) |  |
| (15) |  |

## روش صریح

همانطور که از اسم این روش پیداست، در این روش برای محاسبه دمای یک نقطه در سطح زمانی دوم، فقط از دماهای نقاط مجاور در زمان قبلی استفاده می‌شود و کاری با دمای نقاط در همان سطح نداریم. این روند بدین صورت است که با استفاده از تقریب رایج مرکزی برای مشتقات درجه دوم مکانی در سطح زمانی قبل و تقریب پسرو مرتبه یک برای ترم مشتق درجه یک زمانی معادلات را گسسته می­‌کنیم، آنگاه دمای هر نقطه را تماماً با استفاده از داده‌های قبلی در سطح زمانی قبل برای نقاط مجاور محاسبه می­کنیم. از ویژگی­های این روش عدم تشکیل دستگاه معادلات و در نتیجه راحتی در محاسبات است، اما بدلیل شرط پایداری که برای این روش نیاز است باید برای همگرا شدن به پاسخ درست گام زمانی کوچک در نظر گرفته شود تا واگرایی رخ ندهد. گسسته‌سازی ترم زمانی به صورت پسرو و گسسته‌سازی ترم دیفیوژن به صورت مرکزی انجام شده است. در ادامه گسسته­سازی معادلات برای این روش آورده شده و شرط پایداری برای حالت دوبعدی ذکر شده که بصورت زیر است:

|  |  |
| --- | --- |
| (16) |  |

شرط پایداری نیز برابر است با:

|  |  |
| --- | --- |
| (17) |  |

در صورتی که مش‌بندی غیر یک‌نواخت و خواص با دما متغییر باشند این گسسته سازی به شیوه زیر تغییر می‌نماید.

گسسته‌سازی ترم دیفیوژن با خواص متغییر به صورت زیر خواهد بود:

|  |  |
| --- | --- |
| (18) |  |
| (19) |  |

در نتیجه معادله (3) به صورت زیر گسسته‌سازی می‌شود:

|  |  |
| --- | --- |
| (20) |  |

فلوچارت مربوط به روش صریح

|  |
| --- |
| C:\Users\sajad\Desktop\Drawing5.emf |

## روش ADI

این روش یک روش ضمنی است و تفاوت آن با ضمنی کامل در این است که برای جلوگیری از 5 قطری شدن ماتریس ضرایب و مشکلات ناشی از آن یک جهت را صریح در نظر می‌گیرد. این روش دومرحله‌ای است. در مرحله اول در یک جهت(مثلا x) معادلات نوشته شده و یک دستگاه معادله تشکیل می‌شود و مقادیر در جهت دیگر معلوم فرض می‌شود(مثلا y). در مرحله بعد این رویه برعکس شده و جهت‌ها تعویض می‌شوند.

|  |  |
| --- | --- |
| C:\Users\sajad\Desktop\Drawing1.jpg | C:\Users\sajad\Desktop\Drawing2.jpg |
| شکل2: مرحله اول روش ADI | شکل3: مرحله دوم روش ADI |

گسسته‌سازی در مرحله اول به فرم زیر است:

|  |  |
| --- | --- |
| (21) |  |

و اما برای مرحله دوم گسسته‌سازی مشاهده می‌شود که :

|  |  |
| --- | --- |
| (22) |  |

اگر k ثابت باشد گسسته‌سازی به فرم زیر خلاصه می‌شود:

|  |  |
| --- | --- |
| (23) |  |
| (24) |  |

مزیت این روش این است که بدون قید و شرط پایدار است و می‌توان گام زمانی بزرگ تری را انتخاب نمود.

فلوچارت روش **ADI**

|  |
| --- |
| C:\Users\sajad\Desktop\Adi.emf |

## روش ADE

در این روش یک بار به صورت صعودی در جهت های x و y پیش رفته و مقادیر بدست می‌آیند. در مرحله بعد این رویه عوض شده و نزولی به محاسبه اطلاعات پرداخته می‌شود. این روش یک روش صریح است با این تفاوت که در هر سطح زمانی برای مقادیر معلوم از داده‌ها در همان سطح زمانی به جای سطح زمانی قبلی استفاده می‌شود.

|  |  |
| --- | --- |
| (25) |  |
| (26) |  |

اگر k ثابت باشد گسسته‌سازی به فرم زیر خلاصه می‌شود:

|  |  |
| --- | --- |
| (27) |  |
| (28) |  |

فلوچارت روش **ADE**

|  |
| --- |
| C:\Users\sajad\Desktop\Drawing5.emf |

## کد روش صریح برای مش غیریکنواخت و خواص فیزیکی ثابت

|  |
| --- |
| #include<iostream>  #include<math.h>  #define Max 500  #define dt 0.01  #define Fo 0.00199722  #include <fstream>//New  using namespace std;  int main()  {  ofstream file;  file.open("results");  long double T[Max][Max],X[Max],Y[Max],q=1.1,Kipj,Kimj,Kijp,Kijm,hw,he,hn,hs,rho=2700,Cp=897,S=0,F;  int t=0,grid=82;  cout<<"enter time plz:";  cin>>t;  X[0]=0.1;  Y[0]=0.1;  X[1]=0.11;  Y[1]=0.11;  for(int i=2;i<=grid/2;i++)  {  X[i]=X[i-1]+(X[0]\*pow(q,i-2));  Y[i]=Y[i-1]+(Y[0]\*pow(q,i-2));  }  for(int j=grid/2;j<=grid;j++)  {  X[j+1]=X[j]+(X[0]\*pow(q,(grid-j)));  Y[j+1]=Y[j]+(Y[0]\*pow(q,(grid-j)));  }  for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i==0)  {  T[i][j]=200;  }  if(i!=0&&j==0)  {  T[i][j]=10;  }  if(i==grid&&j!=0)  {  T[i][j]=25;  }  if(i!=0&&i!=grid&&j!=0)  {  T[i][j]=0;  }  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  T[i][j]=100;  //cout<<"T["<<i-1<<"]["<<j<<"]:"<<T[i-1][j]<<endl;  }  }  }  cout<<endl<<endl;  }  for(int k=0;k<t;k++)  {  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  hw=X[i]-X[i-1];  cout<<"X["<<i<<"]:"<<hw<<endl;  he=X[i+1]-X[i];  cout<<"he:"<<he<<endl;  hs=1;Y[j]-Y[j-1];  cout<<"hs:"<<hs<<endl;  hn=1;Y[j+1]-Y[j];  T[i][j]=T[i][j]+dt\*Fo\*(((he\*T[i-1][j]-(he+hw)\*T[i][j]+hw\*T[i+1][j])/(he\*hw\*(he+hw)))+((hn\*T[i][j-1]-(hn+hs)\*T[i][j]+hs\*T[i][j+1])/(hn\*hs\*(hn+hs))));  }  }  }  }  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  file<<X[i]<<" ";  file<<Y[j]<<" ";  file<<T[i][j]<<endl;  }  cout<<endl;  }  file.close();  return 0;  } |

## کد روش صریح برای مش غیریکنواخت و خواص فیزیکی متغییر

|  |
| --- |
| #include<iostream>  #include<math.h>  #define Max 500  #define dt 0.01  #define Fo 0.00199722  #include <fstream>//New  using namespace std;  int main()  {  ofstream file;  file.open("results");  long double T[Max][Max],X[Max],Y[Max],q=1.1,Kipj,Kimj,Kijp,Kijm,hw,he,hn,hs,rho=2700,Cp=897,S=0,F;  int t=0,grid=82;  cout<<"enter time plz:";  cin>>t;  X[0]=0.1;  Y[0]=0.1;  X[1]=0.11;  Y[1]=0.11;  for(int i=2;i<=grid/2;i++)  {  X[i]=X[i-1]+(X[0]\*pow(q,i-2));  Y[i]=Y[i-1]+(Y[0]\*pow(q,i-2));  }  for(int j=grid/2;j<=grid;j++)  {  X[j+1]=X[j]+(X[0]\*pow(q,(grid-j)));  Y[j+1]=Y[j]+(Y[0]\*pow(q,(grid-j)));  }  for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i==0)  {  T[i][j]=200;  }  if(i!=0&&j==0)  {  T[i][j]=10;  }  if(i==grid&&j!=0)  {  T[i][j]=25;  }  if(i!=0&&i!=grid&&j!=0)  {  T[i][j]=0;  }  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  T[i][j]=100;  //cout<<"T["<<i-1<<"]["<<j<<"]:"<<T[i-1][j]<<endl;  }  }  }  cout<<endl<<endl;  }  for(int k=0;k<t;k++)  {  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  cout<<"T["<<i<<"]["<<j<<"]:"<<T[i][j]<<endl;  Kipj=T[i+1][j]+T[i][j];  cout<<"Kipj:"<<Kipj<<endl;  Kimj=T[i][j]+T[i-1][j];  cout<<"Kimj:"<<Kimj<<endl;  cout<<"T["<<i-1<<"]["<<j<<"]:"<<T[i-1][j]<<endl;  Kijp=T[i][j+1]+T[i][j];  cout<<"Kijp:"<<Kijp<<endl;  Kijm=T[i][j]+T[i][j-1];  cout<<"Kimj:"<<Kijm<<endl;  hw=X[i]-X[i-1];  cout<<"X["<<i<<"]:"<<hw<<endl;  he=X[i+1]-X[i];  cout<<"he:"<<he<<endl;  hs=1;Y[j]-Y[j-1];  cout<<"hs:"<<hs<<endl;  hn=1;Y[j+1]-Y[j];  //cout<<"hn:"<<hn<<endl;  S=(dt/(rho\*Cp))\*((2\*Kipj\*(T[i+1][j]-T[i][j])/he)-(2\*Kimj\*(T[i][j]-T[i-1][j])/hw))/(he+hw);  F=(dt/(rho\*Cp))\*((2\*Kijp\*(T[i][j+1]-T[i][j])/hn)-(2\*Kijm\*(T[i][j]-T[i][j-1])/hs))/(hn+hs);  cout<<"F:"<<F<<endl;  cout<<"S:"<<S<<endl;  T[i][j]=T[i][j]+ F+S;  }  }  }  }  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  file<<X[i]<<" ";  file<<Y[j]<<" ";  file<<T[i][j]<<endl;  }  cout<<endl;  }  file.close();  return 0;  } |

## کد روش ADI برای مش غیریکنواخت و خواص فیزیکی متغییر

|  |
| --- |
| #include<iostream>  #include<math.h>  #define Max 100  #define dt 0.01  #define rho 2700  #define Cp 897  #define Fo 0.001  #include <fstream>//New  using namespace std;  double alpha(double);  int main()  {  ofstream file;  file.open("results");  double T[Max][Max],Tnew1[Max][Max],A[Max],X[Max],U[i],u,B[Max],Y[Max],q=1.1,Kipj,Kimj,Kijp,Kijm,hw,he,hn,hs,a[Max],b[Max],c[Max],d[Max],e[Max],f[Max],H;  int t=0,grid=82,Z=0;  cout<<"enter time plz:";  cin>>t;  X[0]=0.1;  Y[0]=0.1;  X[1]=0.11;  Y[1]=0.11;  H=dt;  for(int i=2;i<=grid/2;i++)  {  X[i]=X[i-1]+(X[0]\*pow(q,i-2));  Y[i]=Y[i-1]+(Y[0]\*pow(q,i-2));  }  for(int j=grid/2;j<=grid;j++)  {  X[j+1]=X[j]+(X[0]\*pow(q,(grid-j)));  Y[j+1]=Y[j]+(Y[0]\*pow(q,(grid-j)));  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i==0)  {  T[i][j]=200;  //200  }  if(i!=0&&j==0)  {  T[i][j]=10;  //10  }  if(i==grid&&j!=0)  {  T[i][j]=25;  //25  }  if(i!=0&&i!=grid&&j==grid)  {  T[i][j]=0;  //0  }  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  T[i][j]=200;  }  }  }  for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  Tnew1[i][j]=T[i][j];  Tnew2[i][j]=T[i][j];  }  }  cout<<endl<<endl;  }  while(H<t)  {  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  hw=X[i]-X[i-1];  he=X[i+1]-X[i];  hs=Y[j]-Y[j-1];  hn=Y[j+1]-Y[j];  a[i-1]=2\*alpha(T[i+1][j])\*(dt/2.)/(he\*(hw+he));  b[i-1]=-2\*alpha(T[i][j])\*(dt/2.)/(hw+he);  c[i-1]=2\*alpha(T[i-1][j])\*(dt/2.)/(hw\*(hw+he));  d[i-1]=2\*alpha(T[i][j+1])\*(dt/2.)/(hn\*(hn+hs));  e[i-1]=-2\*alpha(T[i][j])\*(dt/2.)/(hn+hs);  f[i-1]=2\*alpha(T[i][j-1])\*(dt/2.)/(hs\*(hn+hs));  if(i==1)  {  B[i-1]=d[i-1]\*T[i,j+1]+(e[i-1]\*T[i,j])+(f[i-1]\*T[i,j-1])+T[i,j]+c[i-1]\*T[i-1,j];  c[i-1]=0;  }    else if(i==grid-1)  {  B[i-1]=d[i-1]\*T[i,j+1]+(e[i-1]\*T[i,j])+(f[i-1]\*T[i,j-1])+T[i,j]+a[i-1]\*T[i+1,j];  a[i-1]=0;  }  else  {  B[i-1]=d[i-1]\*T[i,j+1]+(e[i-1]\*T[i,j])+(f[i-1]\*T[i,j-1])+T[i,j];  }  A = diag((1-b)) + diag(-c(2:end),1) + diag(-a(1:(end-1)),-1);  U = A[i]/(B[i]);  B[i+1]=((-A[i+1,i]/A[i,i])\*B[i])+B[i+1];  A[i+1]=((-A[i+1,i]/A[i,i])\*(A[i,j]))+A[i+1,j];  for(int j=grid;j<0;j--)  {  if(j==grid)  {  X[i]=B[i]/A[i][i];  }  else  {  X[i]=B[i]-X[i+1]\*A[i][i+1]/A[i][i];  }    }  for(int j=1;j<=grid-1;j++)  {  u=U;  }  }  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  hw=X[i]-X[i-1];  he=X[i+1]-X[i];  hs=Y[j]-Y[j-1];  hn=Y[j+1]-Y[j];  a[i-1]=2\*alpha(T[i+1][j])\*(dt/2.)/(he\*(hw+he));  b[i-1]=-2\*alpha(T[i][j])\*(dt/2.)/(hw+he);  c[i-1]=2\*alpha(T[i-1][j])\*(dt/2.)/(hw\*(hw+he));  d[i-1]=2\*alpha(T[i][j+1])\*(dt/2.)/(hn\*(hn+hs));  e[i-1]=-2\*alpha(T[i][j])\*(dt/2.)/(hn+hs);  f[i-1]=2\*alpha(T[i][j-1])\*(dt/2.)/(hs\*(hn+hs));  if(j==1)  {  B[j-1]=a[j-1]\*T[i,j+1]+(b[j-1]\*T[i,j])+(c[j-1]\*T[i-1,j])+T[i,j]+f[j-1]\*T[i,j-1];  c[i-1]=0;  }    else if(i==grid-1)  {  B[j-1]=a[j-1]\*T[i,j+1]+(b[j-1]\*T[i,j])+(c[j-1]\*T[i-1,j])+T[i,j]+d[j-1]\*T[i,j-1];  d[i-1]=0;  }  else  {  B[j-1]=a[j-1]\*T[i,j+1]+(b[j-1]\*T[i,j])+(c[j-1]\*T[i-1,j])+T[i,j]+d[j-1]\*T[i,j-1];  }  A = diag((1-b)) + diag(-c(2:end),1) + diag(-a(1:(end-1)),-1);  U[i] = A[i]/(B[i]);  B[i+1]=((-A[i+1,i]/A[i,i])\*B[i])+B[i+1];  A[i+1]=((-A[i+1,i]/A[i,i])\*(A[i,j]))+A[i+1,j];  for(int j=grid;j<0;j--)  {  if(j==grid)  {  X[i]=B[i]/A[i][i];  }  else  {  X[i]=B[i]-X[i+1]\*A[i][i+1]/A[i][i];  }    }  for(int j=1;j<=grid-1;j++)  {  u=U;    }  }  H=H+dt;  cout<<"Time:"<<H<<endl;  }              for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  file<<i<<" ";  file<<j<<" ";  file<<Tnew2[i][j]<<endl;  }  cout<<endl;  }  file.close();  return 0;  }    double alpha(double X)  {  double K =2\*T+50;  double Al=K/(rho\*Cp);  return Al;  } |

## کد روش ADE برای مش غیریکنواخت و خواص فیزیکی متغییر

|  |
| --- |
| #include<iostream>  #include<math.h>  #define Max 100  #define dt 0.1  #define rho 2700  #define Cp 897  #define Fo 0.01  #include <fstream>//New  using namespace std;  double alpha(double);  int main()  {  ofstream file;  file.open("results");  double T[Max][Max],Tnew1[Max][Max],Tnew2[Max][Max],X[Max],Y[Max],q=1.1,Kipj,Kimj,Kijp,Kijm,hw,he,hn,hs,a,b,c,d,e,f,H;  int t=0,grid=82,Z=0;  cout<<"enter time plz:";  cin>>t;  X[0]=0.1;  Y[0]=0.1;  X[1]=0.11;  Y[1]=0.11;  H=dt;  for(int i=2;i<=grid/2;i++)  {  X[i]=X[i-1]+(X[0]\*pow(q,i-2));  Y[i]=Y[i-1]+(Y[0]\*pow(q,i-2));  }  for(int j=grid/2;j<=grid;j++)  {  X[j+1]=X[j]+(X[0]\*pow(q,(grid-j)));  Y[j+1]=Y[j]+(Y[0]\*pow(q,(grid-j)));  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i==0)  {  T[i][j]=800;  }  if(i!=0&&j==0)  {  T[i][j]=800;  }  if(i==grid&&j!=0)  {  T[i][j]=800;  }  if(i!=0&&i!=grid&&j==grid)  {  T[i][j]=800;  }  }    for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  T[i][j]=200;  //cout<<"T["<<i-1<<"]["<<j<<"]:"<<T[i-1][j]<<endl;  }  }  }  for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  Tnew1[i][j]=T[i][j];  Tnew2[i][j]=T[i][j];  }  }  cout<<endl<<endl;  }  while(H<t)  //while(Z<1)  {  for(int i=0;i<grid;i++)  {  for(int j=0;j<grid;j++)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {    hw=X[i]-X[i-1];  cout<<" hw:"<<hw<<endl;  he=X[i+1]-X[i];  cout<<"he:"<<he<<endl;  hs=Y[j]-Y[j-1];  cout<<"hs:"<<hs<<endl;  hn=Y[j+1]-Y[j];  cout<<"hn:"<<hn<<endl;  a=2\*alpha(T[i+1][j])\*(dt/2.)/(he\*(hw+he));  b=-2\*alpha(T[i][j])\*(dt/2.)/(hw+he);  c=2\*alpha(T[i-1][j])\*(dt/2.)/(hw\*(hw+he));  d=2\*alpha(T[i][j+1])\*(dt/2.)/(hn\*(hn+hs));  e=-2\*alpha(T[i][j])\*(dt/2.)/(hn+hs);  f=2\*alpha(T[i][j-1])\*(dt/2.)/(hs\*(hn+hs));  Tnew1[i][j]=a\*T[i+1][j]+b\*T[i][j]+c\*T[i-1][j]+d\*T[i][j+1]+e\*T[i][j]+f\*T[i][j-1];  cout<<"T["<<i<<"]["<<j+1<<"]:"<<T[i][j+1]<<endl;  cout<<"S:"<<S<<endl;  }  }  }    for(int i=grid;i>0;i--)  {  for(int j=grid;j>0;j--)  {  if(i!=0&&i!=grid&&j!=0&&j!=grid)  {  cout<<"Kimj:"<<Kijm<<endl;  hw=X[i]-X[i-1];  cout<<"X["<<i<<"]:"<<hw<<endl;  he=X[i+1]-X[i];  cout<<"he:"<<he<<endl;  hs=1;Y[j]-Y[j-1];  cout<<"hs:"<<hs<<endl;  hn=1;Y[j+1]-Y[j];  cout<<"hn:"<<hn<<endl;  a=2\*alpha(Tnew1[i+1][j])\*(dt/2.)/(he\*(hw+he));  cout<<"a:"<<a<<endl;  cout<<"alpha(Tnew1[i+1][j]):"<<alpha(Tnew1[i+1][j])<<endl;  cout<<"he\*(hw+he):"<<he\*(hw+he)<<endl;  b=-2\*alpha(Tnew1[i][j])\*(dt/2.)/(hw+he);  cout<<"b:"<<b<<endl;  c=2\*alpha(Tnew1[i-1][j])\*(dt/2.)/(hw\*(hw+he));  cout<<"c:"<<c<<endl;  d=2\*alpha(Tnew1[i][j+1])\*(dt/2.)/(hn\*(hn+hs));  cout<<"d:"<<d<<endl;  e=-2\*alpha(Tnew1[i][j])\*(dt/2.)/(hn+hs);  cout<<"e:"<<e<<endl;  f=2\*alpha(Tnew1[i][j-1])\*(dt/2.)/(hs\*(hn+hs));  cout<<"f:"<<f<<endl;  Tnew2[i][j]=a\*Tnew1[i+1][j]+b\*Tnew1[i][j]+c\*Tnew1[i-1][j]+d\*Tnew1[i][j+1]+e\*Tnew1[i][j]+f\*Tnew1[i][j-1];  }  }  }  H=H+dt;  cout<<"Time:"<<H<<endl;  }  for(int i=0;i<=grid;i++)  {  for(int j=0;j<=grid;j++)  {  file<<i<<" ";  file<<j<<" ";  file<<Tnew2[i][j]<<endl;  }  }  file.close();  return 0;  }    double alpha(double X)  {  double K =2\*T+50;  double Al=K/(rho\*Cp);  return Al;  } |

## نتایج

در این بخش نتایج حاصل برای یک حفره با شرایط مرزی دما ثابت و ضریب رسانندگی متغییر با دما و پارامترهای فیزیکی مشخص به ازای روش‌ها و زمان‌های مختلف نشان داده شده است.

|  |
| --- |
| J:\CFD\end\Ex.tif  شکل4: کانتور دمایی در روش صریح |
| J:\CFD\end\ADI.tif  شکل5: کانتور دمایی در روش ADI |
| J:\CFD\end\ADE.tif  شکل6: کانتور دمایی در روش ADE |

از آنجایی که در کارهای علمی کانتور از ارزش کمتری نسبت به نمودار برخوردار است لذا سعی شده نمودار دما روی قطر اصلی حفره در این سه روش با هم مقایسه شود.

|  |
| --- |
| J:\CFD\end\L1.tif  شکل7: دمای قطر اصلی حفره در روش‌های مختلف |

همانطور که از کانتورها و نمودار مشخص است نتایج سه روش به ازای زمان 20 ثانیه با هم همخوانی دارد. و از این به بعد به دلیل یکسان بودن نتایج فقط نتایج روش ADE که زمان اجرای کمتری نسبت به روش ADI دارد گزارش شده است.

|  |
| --- |
| J:\CFD\end\ADI.tif  شکل8: کانتور دمایی برای زمان 20 ثانیه |
| J:\CFD\end\70.tif  شکل9: کانتور دمایی برای زمان 70 ثانیه  J:\CFD\end\150.tif  شکل10: کانتور دمایی برای زمان 150 ثانیه |
| J:\CFD\end\500000.tif  شکل11: کانتور دمایی برای زمان 500 ثانیه |

همچنین نمودار قطر اصلی و وسط حفره در جهت y را می‌توان مشاهده نمود و به نفوذ حرارتی با افزایش زمان اجرای برنامه پی برد.

|  |  |
| --- | --- |
| J:\CFD\end\L2.tif  شکل12: نمودار دمای x=L/2 به ازای زمان‌های مختلف | J:\CFD\end\L3.tif  شکل13: نمودار دما روی قطر اصلی حفره به ازای زمانهای مختلف |

## نتیجه گیری

1. روش ADI یک روش ضمنی است و تفاوت آن با ضمنی کامل در این است که برای جلوگیری از 5 قطری شدن ماتریس ضرایب و مشکلات ناشی از یک جهت را معلوم فرض کرده که با این کار ماتریس 5 قطری به ماتریس سه قطری تبدیل می‌شود و برای حل ماتریس سه قطری از روش TDMA استفاده شده است.
2. روش‌های ضمنی دارای جواب‌هایبهتری هستند اما مشکل آنها زمان اجرای بالاتر می‌باشد. همچنین روش‌های صریح شرط پایداری دارند و ما را محدود به انتخاب گام زمانی کوچک می‌کنند.
3. نکته قابل ذکر در روش ADE این است که تنها زمانی که شرط مرزی ما از نوع دریکه است، قابل استفاده می‌باشد.
4. پیاده‌سازی روش ADI به دلیل ماهیت ضمنی بودن آن از مابقی روش‌های ذکر شده دشوارتر است.

1. Elliptic [↑](#footnote-ref-2)
2. Parabolic [↑](#footnote-ref-3)
3. Hyperbolic [↑](#footnote-ref-4)