به نام خدا

گزارش پروژه اول درس دینامیک سیالات محاسباتی

عنوان پروژه: حل عددی معادله انتقال حرارت دوبعدی گذرا درون یک کانال

دانشجو: علی باقری برمس

شماره دانشجویی: ۴۰۱۷۴۲۲۷۴

استاد: دکتر حسینعلی پور

بهمن ۱۴۰۱

صورت پروژه:

شکل زیر یک سطح مقطع از یک کانال مربعی به ضلع ao را نشان میدهد که درون آن نیز یک فضای خالی مربعی به ضلع ai وجود دارد. سیال گرم در کانال درونی به گونهای حرکت میکند که دمای دیواره ثابت و برابر400 کلوین بماند. کانال بیرونی با هوای اطراف در ارتباط است و به صورت جابجایی با محیط انتقال حرارت دارد. ضریب انتقال حرارت جابجایی محیط 50 وات بر متر مربع بر کلوین و دمای آن 300 کلوین فرض میگردد.

اگر جنس کانال از فولاد کربن و دمای اولیه‌ی آن 300 کلوین باشد، به سوالهای زیر پاسخ دهید:

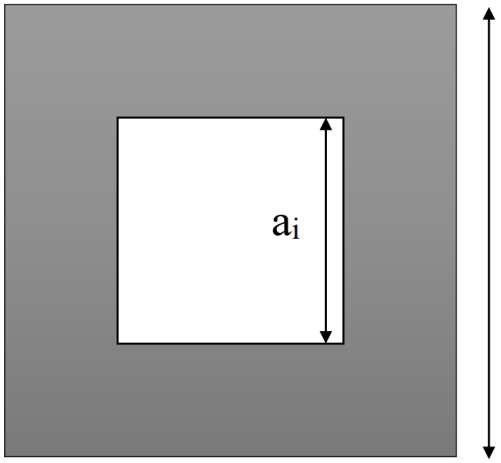
الف) مساله را تا رسیدن به حالت پایا به صورت صریح حل کنید. نمودارهای توزیع نهایی دما را برای مقادیر مختلف عدد نفوذ رسم کنید و شرط پایداری و زمان حل را برای هر کدام بررسی کنید.

ب) مساله را به صورت ضمنی حل کنید. نمودارهای توزیع دما را با فواصل زمانی مناسب تا رسیدن به حالت پایا رسم کنید.

ج) با هدف کاهش زمان محاسباتی و به کمک تقارن موجود در شکل، مایلیم تا فقط یک چهارم از شکل را مورد بررسی قرار دهیم. با این فرض، مساله را تا رسیدن به شرایط پایا و به روش صریح حل کنید. نمودارهای توزیع دما را با فواصل زمانی مناسب تا رسیدن به حالت پایا را نیز رسم کنید.

د) نتایج هر سه بخش بالا را با بیان توضیحات مناسب با هم مقایسه کنید.

توضیحات: گزارش شما میبایست شامل روشهای گسستهسازی معادلات، نتایج و بحث در مورد آنها و نیز کدهای نوشته شده باشد



$$\frac{a_o}{a_i} = 2$$

خلاصه داده های مسئله:

$$a_0 = 1 + 0.02(74) = 2.48m$$

$$\frac{a_i}{a_0} = 2 \Rightarrow a_i = 4.96m0$$

$$h = 50\ {}^{W}/_{m^2K}$$

$$T_\infty = 300K$$

$$Initial\ Temperature = 300\ K$$

جنس ماده: فولاد کربنی

با فرض جنس Plain Carbon Steel مسئله حل خواهد شد. خواص این فولاد به شرح زیر است.[1]

$$Conductivity(k) = 60.5\frac{W}{m.K}$$

$$Diffusivity\ (\alpha) = 17.7 \times 10^{-6}$$

معادله حاکم بر مسئله که انتقال حرارت دوبعدی ناپایا است به شکل زیر خواهد بود.[2]

$$\frac{\partial T}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right)$$

در ادامه به گسسته سازی معادلات به دو روش ضمنی و صریح پرداخته می‌شود.

۱. روش صریح

برای روش صریح از متد FTCS استفاده می‌شود و معادله انتقال حرارت به صورت زیر درمی‌آید.

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} = \alpha\left(\frac{T_{i+1,j}^{n} - 2T_{i,j}^{n} + T_{i-1,j}^{n}}{(\Delta x)^2} + \frac{T_{i,j+1}^{n} - 2T_{i,j}^{n} + T_{i,j-1}^{n}}{(\Delta y)^2}\right)$$

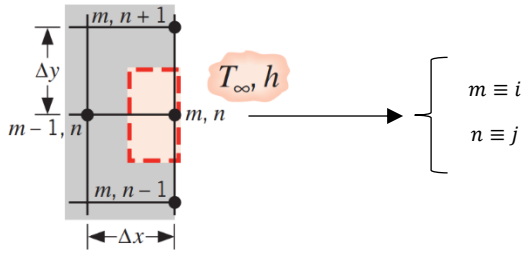
$\Delta x = \Delta y$ در نظر گرفته می‌شود. در نتیجه:

$$T_{i,j}^{n+1} = \alpha\frac{\Delta t}{(\Delta x)^2}\left(T_{i+1,j}^{n} + T_{i-1,j}^{n} + T_{i,j+1}^{n} + T_{i,j-1}^{n}\right) + \left(1 - 4\alpha\frac{\Delta t}{(\Delta x)^2}\right)T_{i,j}^{n}$$

در داخل کانال دما ثابت و برابر با ۴۰۰ کلوین است. اما در خارج از کانال انتقال حرارت به صورت جابه‌جایی وجود دارد و شرایط مرزی با نوشتن بقای انرژی برای یک حجم کنترل دی مرز به صورت زیر حاصل می‌شود.

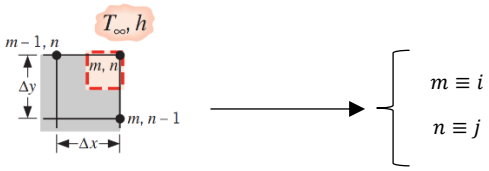
برای نقاط روی دیواره به جز گوشه ها شرایط مرزی از رابطه زیر بدست می‌آید.[1]

$$T_{i,j}^{n+1} = \quad Fo\left(2T_{i-1,j}^{n} + T_{i,j+1}^{n} + T_{i,j-1}^{n} + 2BiT_\infty\right) + (1 - 4Fo - 2BiFo)T_{i,j}^{n}$$



که در آن $Bi = \frac{h\Delta x}{k}$ و $\mathrm{Fo} = \frac{\alpha \Delta t}{(\Delta x)^2}$ می‌باشد و در شرایطس پایداری است که $Fo(2 + Bi) \leq \frac{1}{2}$. برای نقاط بیرونی در گوشه‌ها نیز از رابطه زیر استفاده می‌شود.[1]

$$T_{i,j}^{n+1} = \quad 2Fo\left(T_{i-1,j}^{n} + T_{i,j-1}^{n} + 2BiT_{\infty}\right) + (1 - 4Fo - 4BiFo)T_{i,j}^{n}$$



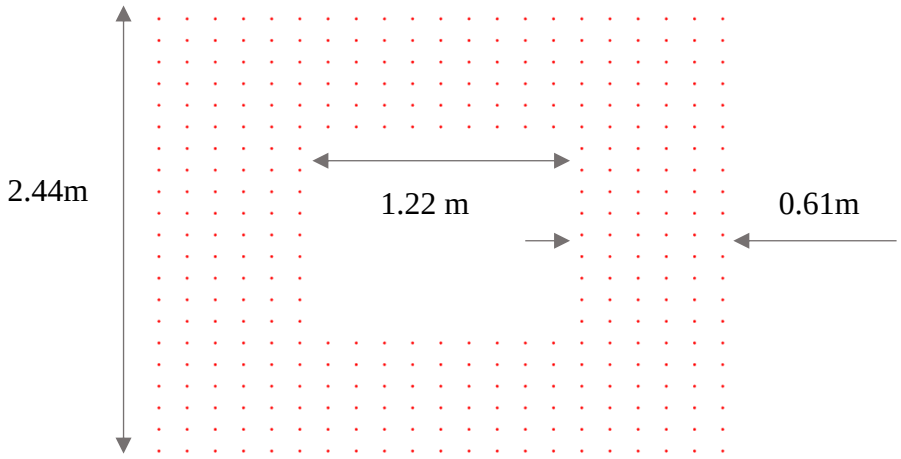
معادله فوق در شرایطی پایدار است که $Fo(1 + Bi) \leq \frac{1}{4}$.

در نتیجه با استفاده از معادلات بدست آمده می‌توان مسئله را به صورت صریح حل کرد.

به این منظور ابتدا کانال را شبکه‌بندی می کنیم. در اینجا برای آنکه شبکه بندی مرتب باشد اندازه شبکه مکانی، ضریبی از پهنای کانال در نظر گرفته می‌شود.

اگر پهنای کانال برابر با $0.61 = \mathrm{w} = \frac{a_i - a_0}{2}$ باشد، برای داشتن شبکه مناسب، اندازه شبکه را ابتدا برابر با $\Delta \mathrm{x} = \Delta \mathrm{y} = \frac{\mathrm{w}}{5}$ در نظر گرفته می‌شود. با بررسی سه شرط پایداری موجود حداکثر اندازه شبکه زمانی $\Delta t =$ 190 بدست خواهد آمد.

در شکل (۱) شبکه بندی هندسه مسئله مشاهده می‌شود.



شکل1 شبکه بندی هندسه مسئله

برای آنکه مسئله پایا شود، یک مقدار کوچک $\varepsilon = 10^{-4}$ در نظر گرفته می‌شود و شرط رسیدن به حل پایا به صورت زیر تعریف می‌شود.

$$\varepsilon > \max(|T^{n+1} - T^{n}|)$$

در ادامه در نرم افزار *MatLab* مسئله به صورت صریح به ازای مقادیر مختلف Δt حل شده و نتایج آورده شده است.

برای مقایسه بهتر نتایج، مسئله یکبار به صورت یک معادله بیضوی پایا بدون در نظر گرفتن عبارت مشتق زمانی به روش Point Successive Over-Relaxation Method (PSOR) حل شده است. معادلات به صورت زیر گسسته می‌شوند. $\beta = \frac{\Delta y}{\Delta x} = 1$ در نظر گرفته شده است.

نقاط میانی:

$$T_{i,j}^{n+1} = \frac{\omega}{4}\left(T_{i+1,j}^{n} + T_{i-1,j}^{n+1} + T_{i,j+1}^{n} + T_{i,j-1}^{n+1}\right) + (1-\omega)T_{i,j}^{n}$$

دیواره سمت راست:

$$T_{i,j}^{n+1} = \frac{\omega}{4+2Bi}\left(2T_{i-1,j}^{n+1} + T_{i,j+1}^{n} + T_{i,j-1}^{n+1} + 2BiT_{\infty}\right) + (1-\omega)T_{i,j}^{n}$$

گوشه سمت راست و پایین:

$$T_{i,j}^{n+1} = \frac{\omega}{2+2Bi}\left(T_{i+1,j}^{n} + T_{i,j-1}^{n+1} + 2BiT_{\infty}\right) + (1-\omega)T_{i,j}^{n}$$

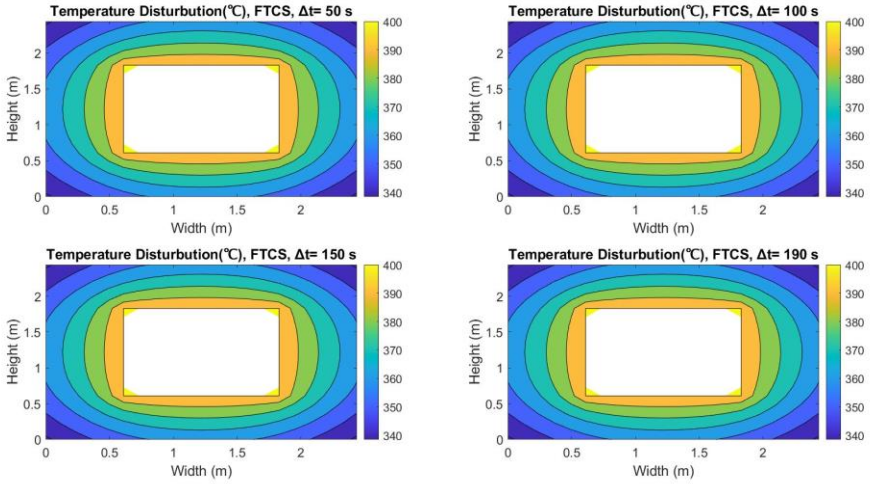
جدول ۱ زمان رسیدن به حل پایا و خطا به ازای مقادیر مختلف Δt

Δt	Steady State time	Error (with respect to PSOR Method)
Second	hour	%
1	10.67	0.2297
10	15.59	0.0221
30	17.27	0.0067
50	17.54	0.0036
70	17.34	0.0023
90	16.92	0.0016
110	16.3	0.0011
130	15.71	0.0008
150	14.96	0.0007

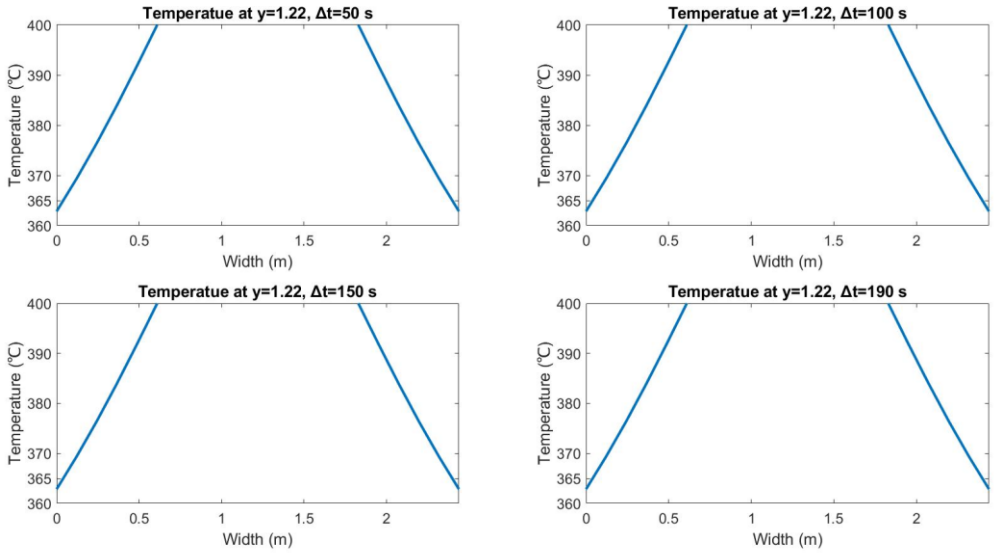
170	14.12	0.0008
190	13.25	0.0010

همانطور که مشاهده می‌شود هنگامی که گام زمانی بسیار کوچک شود، در این مسئله در حدود ۱۰ ثانیه و کمتر، بدلیل آنکه اختلاف دمای بدست آمده در زمان های متوالی بسیار کوچک می‌شود، خطای حل زیاد می شود. در نتیجه گام زمانی بسیار کوچک مناسب نخواهد بود. علت اختلاف زمان رسیدن به حل پایا می‌تواند بدلیل خطای گسسته سازی باشد که برای روش FTCS از مرتبه $(\Delta t\ , (\Delta x)^2, (\Delta y)^2)$ می‌باشد. همچنین روش PSOR نیز خطا دارد و دقیق نیست و صرفا جهت مقایسه استفاده شده است.

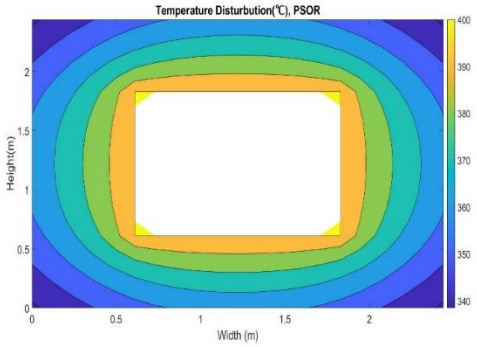
در ادامه نتایج بدست آمده از روش FTCS و PSOR مشاهده می‌شود.



شکل2 توزیع دما بدست آمده از روش FTCS به ازای مقادیر مختلف Δt



شکل ۳ منحنی دما در خط میانی کانال به ازای مقادیر مختلف Δt. مشاهده می‌شود که تغییرات تقریبا به صورت خطی است.



شکل ۴ توزیع دمای بدست آمده از روش PSOR به ازای ω = 1.5

روش PSOR بعد از ۶۴ تکرار به همگرایی می‌رسد.

۲. روش ضمنی

برای حل مسئله به روش ضمنی از روش Lassonen بهره برده شده است و گسسته سازی معادلات به شکل زیر در خواهد آمد. این روش تحت هر شرایط پایدار خواهد بود لذا می‌توان از گام های زمانی بزرگتر برای حل مسئله بهره برد.

نقاط میانی:

$$(1+4Fo)T_{i,j}^{n+1} - Fo\left(T_{i+1,j}^{n+1} + T_{i-1,j}^{n+1} + T_{i,j+1}^{n+1} + T_{i,j-1}^{n+1}\right) = T_{i,j}^{n}$$

نقاط روی دیواره بیرونی در حضور سیال(دیوار سمت راست):

$$(1+2Fo(2+Bi))T_{i,j}^{n+1} - Fo\left(2T_{i-1,j}^{n+1} + T_{i,j+1}^{n+1} + T_{i,j-1}^{n+1}\right) = T_{i,j}^{n} + 2BiFoT_{\infty}$$

گوشه بیرونی در حضور سیال(گوشه سمت راست و پایین):

$$(1+4Fo(1+Bi))T_{i,j}^{n+1} - Fo\left(2T_{i-1,j}^{n+1} + T_{i,j-1}^{n+1}\right) = T_{i,j}^{n} + 4BiFoT_{\infty}$$

در ادامه به مقایسه نتایج بدست آمده از حل ضمنی با نتایج بدست آمده از حل صریح و همچنین روش PSOR پرداخته خواهد شد.

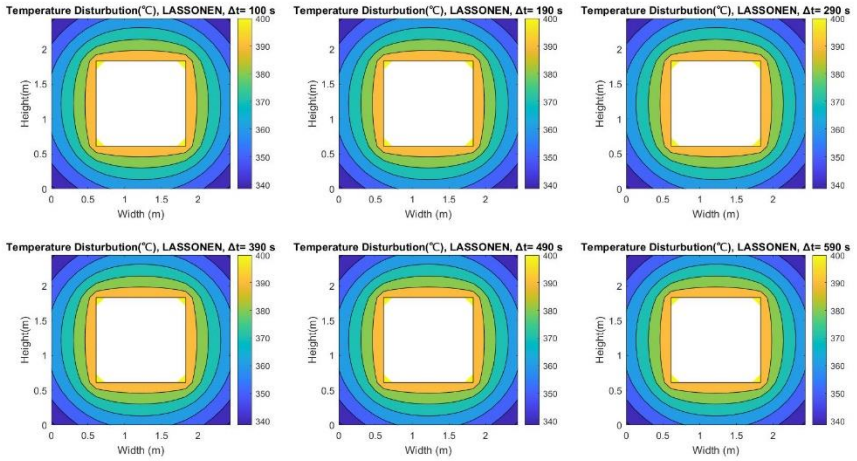
جدول 2 مقایسه روش FTCS و Lassonen

Δt Second		Lassonen	FTCS	Maximum difference of Methods
1	Steady State time (hour)	10.6961	10.677	0.0030
	Error (with respect to PSOR Method) %	0.2306	0.2297	
50	Steady State time (hour)	19.6111	17.542	0.0031
	Error (with respect to PSOR Method) %	0.0045	0.0036	
130	Steady State time (hour)	21.8833	15.708	0.0030

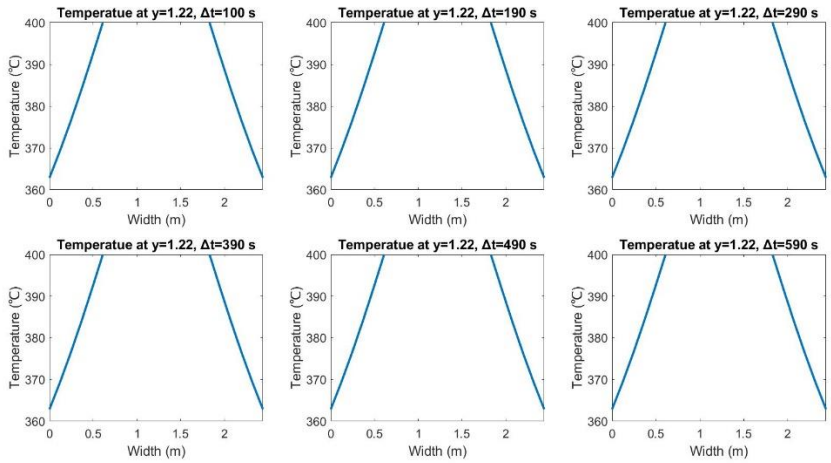
	Error (with respect to PSOR Method) %	0.0016	0.0008	
190	Steady State time (hour)	22.8	13.247	0.0029
	Error (with respect to PSOR Method) %	0.0011	0.001	
240	Steady State time (hour)	23.4	10.867	0.0028
	Error (with respect to PSOR Method) %	0.0008	0.0012	
290	Steady State time (hour)	23.925	8.0556	0.0026
	Error (with respect to PSOR Method) %	0.0009	0.0013	
340	Steady State time (hour)	24.3667	4.4389	0.0023
	Error (with respect to PSOR Method) %	0.001	0.0015	
390	Steady State time (hour)	24.7	12.783	0.0021
	Error (with respect to PSOR Method) %	0.001	0.0015	
440	Steady State time (hour)	25.0556	734.19	-
	Error (with respect to PSOR Method) %	0.0011	0	
490	Steady State time (hour)	25.4528	309.38	-
	Error (with respect to PSOR Method) %	0.0012	0	
540	Steady State time (hour)	25.65	190.95	-
	Error (with respect to PSOR Method) %	0.0012	0	
590	Steady State time (hour)	25.8944	116.53	-
	Error (with respect to PSOR Method) %	0.0012	0	

همانطور که مشاهده می‌شود روش ضمنی Lassonen تحت هر شرایط پایدار است و پاسخ قابل قبولی می‌دهد ولی روش FTCS از یک جا به بعد کاملا ناپایدار می‌شود.

در ادامه نتایج بدست آمده از روش ضمنی مشاهده می‌شود.



شکل 5 توزیع دما بدست آمده از روش Lassonen به ازای مقادیر مختلف Δt

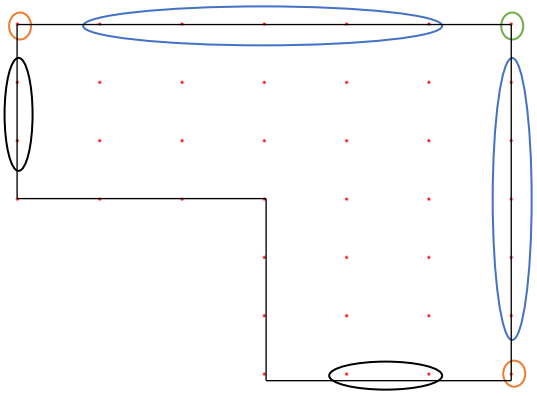


شکل ۶ منحنی دما در خط میانی کانال به ازای مقادیر مختلف Δt . مشاهده می‌شود که تغییرات تقریبا به صورت خطی است.

۳. یک چهارم کانال

برای افزایش سرعت محاسبات، یک چهارم بالا سمت راست شکل، جداگانه به روش صریح حل شده است و در انتها با استفاده از تقارن شکل کامل محاسبه شده است. برای اطمینان از روش حل، نتیجه بدست آمده با نتیجه بدست آمده برای شکل کامل مقایسه شده است.

معادلات در این حالت مشابه روش FTCS است و تنها در محل جدا شدن شکل تفاوت جزئی دارد.



شکل 7 شبکه بندی هندسه برای یک چهارم کانال

نقاط مشخص شده با رنگ مشکی:

$$T_{i,j}^{n+1} = F0\left(T_{i+1,j}^{n} + T_{i-1,j}^{n} + 2\,T_{i,j+1}^{n}\right) + (1 - Fo\,)T_{i,j}^{n}$$

نقاط مشخص شده با رنگ نارنجی:

$$T_{i,j}^{n+1} = \quad Fo\left(2T_{i-1,j}^{n} + 2T_{i,j+1}^{n} + 2BiT_{\infty}\right) + (1 - 4Fo - 2BiFo)T_{i,j}^{n}$$

نقاط مشخص شده با رنگ آبی:

$$T_{i,j}^{n+1} = \quad Fo\left(2T_{i-1,j}^{n} + T_{i,j+1}^{n} + T_{i,j-1}^{n} + 2BiT_{\infty}\right) + (1 - 4Fo - 2BiFo)T_{i,j}^{n}$$

نقطه مشخص شده با رنگ سبز:

$$T_{i,j}^{n+1} = \quad 2Fo\left(T_{i-1,j}^{n} + T_{i,j-1}^{n} + 2BiT_{\infty}\right) + (1 - 4Fo - 4BiFo)T_{i,j}^{n}$$

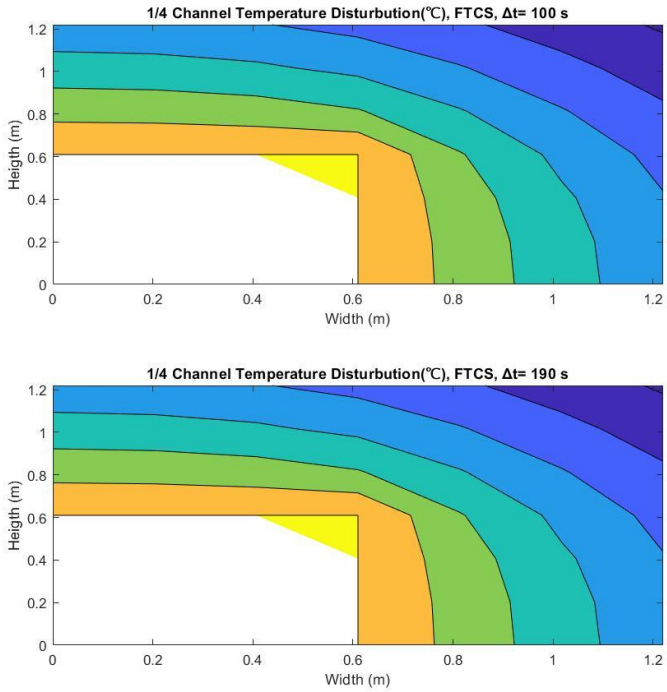
سایر نقاط:

$$T_{i,j}^{n+1} = Fo\ \left(T_{i+1,j}^{n} + T_{i-1,j}^{n} + \ T_{i,j+1}^{n} + T_{i,j-1}^{n}\right) + (1 - 4Fo\ )T_{i,j}^{n}$$

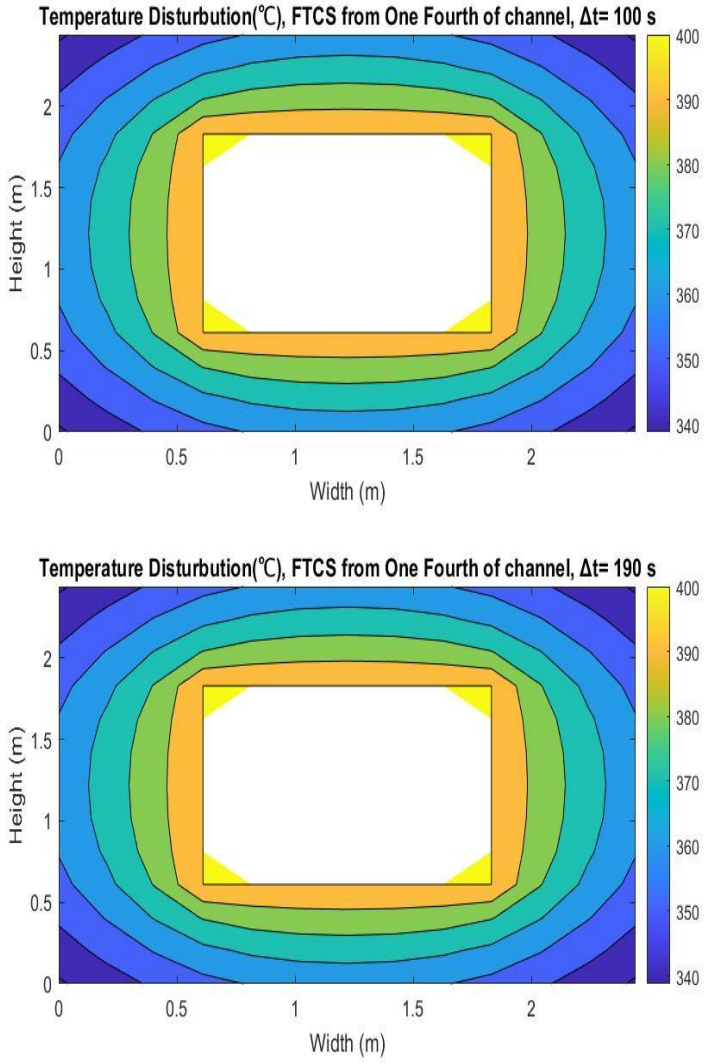
نتایج به شرح زیر است.

جدول ۳ مقایسه حل یک چهارم کانال با کل کانال به روش FTCS

Δt Second	FTCS Method	$\frac{1}{4}$ Channel	Full Channel
50	Steady State time (hour)	19.2778	18.61
	Error %	3.72e-4	
100	Steady State time (hour)	20.7778	19.3
	Error %	3.64e-4	
150	Steady State time (hour)	21.6250	19.2917
	Error %	3.59e-4	
190	Steady State time (hour)	22.1139	19.0528
	Error %	3.50e-4	



شکل ۸ توزیع دما در یک چهارم کانال



شکل ۹ توزیع دمای بدست آمده از تقارن بوسیله دمای یک چهارم کانال

۴. منابع

[1] T. L. Bergman and F. P. Incropera, Eds., *Fundamentals of heat and mass transfer*, 7th ed. Hoboken, NJ: Wiley, 2011.

[2] K. A. Hoffmann and S. T. Chiang, *Computational fluid dynamics. 1*, 4. ed., 2. print. Wichita: Engineering Education System, 2004.

۵. کد متلب

کد متلب نوشته شده شامل ۱ فایل اصلی کد به همراه ۳ فایل نوشته شده به صورت تابع برای روش های FTCS، PSOR و Lassonen و هم چنین یک فایل تابه برای یک چهارم کانال است.

```
%CFD Project Number 1
%Unsteady heat transfer in a square channel with convection in outside and
                                        %constant temperature inside
%the problem is solved by 3 Methods: FTCS - PSOR - LASSONEN
%different values of time grid has been set and plotted
%Dt is optional - If it is changed Pay attention to plots and stability



%%
close all
clear
clc
%Student No. :401742274

a0=1+0.02*74;                           %Channel inner width
ai=2*a0;                                %Channel outer width

W=abs(ai-a0)/2;                         %Channel Wall
w=3;                                    %number of partitions in Channel Wall
dx=W/w;                                 %delta x
dy=dx;                                  %delta y

alpha=17.7e-6;                          %Diffusion Coefficient
h=50;                                   %Convection Coefficient
k=60.5;                                 %Conductivity

tf=3600*20;                             %Final Time (if needed-not used here)
x=0:dx:ai;                              %x vector
y=0:dy:ai;                              %y vector
nx=length(x);                           %Number of nodes in x direction
ny=length(y);                           %Number of nodes in x direction
Bi=h*dx/k;                              %finite-difference form of the Biot number

T0=300;                                 %Initial Temperature
T=ones(nx,ny)*T0;                       %Initializing Temperature
Tin=400;                                %inner Temperature
Ta=300;                                 %outer Fluid Temperature

x1=find(x==W); x2=find(x==(ai-W)); %Position Of Inner Corners
y1=find(x==W);y2=find(x==(ai-W));

%% Initializing Temperature
for i=1:nx
    for j=1:ny
        if i>=x1 && i<=x2 && j==y1
            T(i,j)=Tin;
        elseif i>=x1 && i<=x2 && j==y2
            T(i,j)=Tin;
        elseif j>=y1 && j<=y2 && i==x1
            T(i,j)=Tin;
        elseif j>=y1 && j<=y2 && i==x2
            T(i,j)=Tin;
        elseif (i>x1 && i<x2) && (j>y1 && j<y2)
            T(i,j)=nan;
        end

    end
end
%% PSOR Method
[T_PSOR, it_PSOR] = PSOR(T,nx,ny,x1,x2,y1,y2,Ta,Bi);
%%
q=1;                                        %counter
Dt=[50 100 150 190  ];                        %delta t
T_FTCS=cell(1,length(Dt));                  %%Initializing a cell to reserve Temperatures With
Diiferent delta ta
T_LASSONEN=cell(1,length(Dt));
```

```
%% FTCS and LASSONEN Method solution
for dt=Dt
t=0:dt:tf;                                          %time vector
nt=length(t);                                       %number of time nodes

Fo=alpha*dt/(dx)^2;                                 %finite-difference form of the Fourier number
St1=Fo*(1+Bi);                                      %Stability Condtion
St2=Fo*(2+Bi);                                      %Stability Condtion

while Fo>1/4 || St1>1/4 || St2>1/2                  %Checking Stability
    disp('Not Stable')
    dt=input('Insert new dt:\n');
    Fo=alpha*dt/(dx)^2;
    St1=Fo*(1+Bi);
    St2=Fo*(2+Bi);
    t=0:dt:tf;
    nt=length(t);
end


eps=10^-4;                                          %epsilon
T_FC_n=T;                                           %new FTCS Temp. matrix

E=1000;                                             %initial diffrence of two adjacent Temp.
Matrixes
it_f=0;                                             %number of iteration Until Steady State is
rechead

while E>eps
    T_FC=T_FC_n;
    T_FC_n = FTCS(T_FC,nx,ny,x1,x2,y1,y2,Fo,Bi,Ta);      %FTCS solution
    E=max(max(abs (T_FC_n-T_FC  )));                    %Maximum diffrence of two adjacent
Temp. Matrixes
    it_f=it_f+1;
end
T_FC=T_FC_n;

T_FTCS{q}=T_FC;                                          %saving Temp. in a cell


FT_F=(it_f-1)*dt;                                        %time when steady state is reached
in second
FT_FTCS_hour(q)=FT_F/3600;


%{
for i=1:nx                                          plotting Mesh
    for j=1:ny
        if isnan(T(i,j))
            MESH(i,j)=nan;
        else
            MESH(i,j)=1;
            plot(i,j,'r.')
            hold on
        end
    end
end
%}


Tm_N=T;
E2=100;                                               %initial diffrence of two adjacent
Temp. Matrixes
it_l=0;                                               %number of iteration Until Steady
State is rechead
while E2>eps
    Tm=Tm_N;
    Tm_N = Lassonnen(Tm,nx,ny,x1,x2,y1,y2,Ta,Tin,w,Fo,Bi);  %LASSONEN solution
    E2=max(max(abs(Tm_N-Tm)));                              %Maximum diffrence of two adjacent
Temp. Matrixes
    it_l=it_l+1;
end
Tm=Tm_N;
T_LASSONEN{q}=Tm;                                            %saving Temp. in a cell
FT_L=(it_l-1)*dt;                                            %time when steady state is
reached in second
```

```
FT_LASSONEN_hour(q)=FT_L/3600;
Error_Lassonen(q)=max(max(abs((Tm-T_PSOR)./T_PSOR)))*100;       %maximum error for FTCS with
respect to PSOR
Error_FTCS(q)=max(max(abs((T_FC-T_PSOR)./T_PSOR)))*100;         %maximum error for LASSONEN
with respect to PSOR
DIFF(q)= max(max(abs((T_FC-Tm))));

disp(['Time until steady state is reached in FTCS Method in hour:
',num2str(FT_FTCS_hour(q))]);
disp(['Time until steady state is reached in Lassonen Method in hour:
',num2str(FT_LASSONEN_hour(q))]);
q=q+1;
end

FT_FTCS_hour=FT_FTCS_hour';
FT_LASSONEN_hour=FT_LASSONEN_hour';
Error_Lassonen=Error_Lassonen';
Error_FTCS=Error_FTCS';
DIFF=DIFF';

%% PLOTTING
[X,Y]=meshgrid(x,y);

figure(1)
for i=1:length(Dt)
    subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i)
    contourf(X,Y,cell2mat(T_FTCS(i)))
    title(['Temperature Disturbution(?), FTCS, ?t= ', num2str(Dt(i)),' s'])
    ylabel('Height (m)');xlabel('Width (m)');
    colorbar
    set(gca,'fontsize',11);
    hold on
end

figure(2)
for i=1:length(Dt)
    subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i);
    TT=cell2mat(T_FTCS(i));
    plot(x,TT(round(ny/2),:),'Linewidth',2.5)
    title(['Temperatue at y=',num2str(y(round(ny/2))),', ?t=',num2str(Dt(i)),' s'])
    xlabel('Width (m)');ylabel('Temperature (?)');xlim([0 ai]);
    set(gca,'fontsize',14);
    hold on
end

figure (3)
for i=1:length(Dt)
    subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i)
    contourf(X,Y,cell2mat(T_LASSONEN(i)))
    title(['Temperature Disturbution(?), LASSONEN, ?t= ', num2str(Dt(i)),' s'])
    ylabel('Height(m)');xlabel('Width (m)')
    set(gca,'fontsize',11);
    colorbar
    hold on
end

figure(4)
for i=1:length(Dt)
    subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i);
    TT2=cell2mat(T_LASSONEN(i));
    plot(x,TT2(round(ny/2),:),'Linewidth',2.5)
    title(['Temperatue at y=',num2str(y(round(ny/2))),', ?t=',num2str(Dt(i)),' s'])
    xlabel('Width (m)');ylabel('Temperature (?)'); xlim([0 ai]);
    set(gca,'fontsize',14);
    hold on
end

figure(5)
 contourf(X,Y,T_PSOR)
    title(['Temperature Disturbution(?), PSOR'])
    ylabel('Height(m)');xlabel('Width (m)')
    set(gca,'fontsize',18);colorbar;


    %% One Fourth of the channel                  ?? ????? ?????
one_fourth
```

```
    i=1;
    T4=cell(1,length(Dt));
    T_one_fourth=cell(1,length(Dt));
    for dt=Dt
        [T4{i}, T_one_fourth{i}, FT_onefourth(i), X2,
Y2,m]=one_fourth(a0,ai,w,alpha,h,k,T0,Tin,Ta,eps,dt);
        Error_onefourth(i)=max(max(abs((T4{i} -
cell2mat(T_FTCS(i))./cell2mat(T_FTCS(i)))))*100;
        i=i+1;

    end


  for i=1:length(Dt)

        figure(6)
        subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i)
        contourf(X,Y,T4{i})
        title(['Temperature Disturbution(?), FTCS from One Fourth of channel, ?t= ',
num2str(Dt(i)),' s'])
        ylabel('Height (m)');xlabel('Width (m)');
        colorbar
        set(gca,'fontsize',11);
        hold on
        figure(7)
        subplot(ceil(length(Dt)/2),ceil(length(Dt)/2),i)
        contourf(X2,Y2,T_one_fourth{i})
        title(['1/4 Channel Temperature Disturbution(?), FTCS, ?t= ', num2str(Dt(i)),' s'])
        xlabel('Width (m)');ylabel('Heigth (m)');
        hold on
  end



function [T] = FTCS(T,nx,ny,x1,x2,y1,y2,Fo,Bi,Ta)



    for i=1:nx
        for j=1:ny


            if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                continue

            elseif i==1 && j==1
                T(i,j)=2*Fo*(T(i+1,j)+T(i,j+1)+2*Bi*Ta)+(1-4*Fo-4*Bi*Fo)*T(i,j);
            elseif i==1 && j>1 && j<ny
                T(i,j)=Fo*(2*T(i+1,j)+T(i,j+1)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);

            elseif j==ny && i==1
                T(i,j)=2*Fo*(T(i+1,j)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-4*Bi*Fo)*T(i,j);
            elseif i==nx && j==1
                T(i,j)=2*Fo*(T(i-1,j)+T(i,j+1)+2*Bi*Ta)+(1-4*Fo-4*Bi*Fo)*T(i,j);
            elseif i==nx && j==ny
                T(i,j)=2*Fo*(T(i-1,j)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-4*Bi*Fo)*T(i,j);


            elseif j==1 && i>1 && i<nx
                T(i,j)=Fo*(2*T(i,j+1)+T(i+1,j)+T(i-1,j)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);
            elseif j==ny && i>1 && i<nx
                T(i,j)=Fo*(2*T(i,j-1)+T(i+1,j)+T(i-1,j)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);

            elseif i==nx && j>1 && j<ny
                T(i,j)=Fo*(2*T(i-1,j)+T(i,j+1)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);
            else
                T(i,j)=Fo*(T(i+1,j)+T(i-1,j)+T(i,j-1)+T(i,j+1)-4*T(i,j))+T(i,j);
            end
        end
    end
```

```
end



function [Tm] = Lassonnen(T,nx,ny,x1,x2,y1,y2,Ta,Tin,w,Fo,Bi)




in=(2*(w+1)-1);
io=nx-in;
L=nx*ny-in*in;
CM=zeros(L);                                %Coefficient Matrix
GM=zeros(L,1);                              %Known Values Matrix
Tm=T;



A=1+4*Fo;
B=1+2*Fo*(2+Bi);
C=1+4*Fo*(1+Bi);
D=-Fo;

f=zeros(nx,ny);
% f is a matrix that has number of each array witch will be used to create
% Coefficient Matrix
for j=1:ny
    for i=1:nx
        if (i>x1 && i<x2) && (j>y1 && j<y2)
                f(i,j)=nan;
        elseif j<y1
            f(i,j)=(j-1)*nx+i;
        elseif j>=y1 && j<=y2
            if i<x1
                f(i,j)=w*nx+(j-w-1)*io+( i );
            elseif i>x2
                f(i,j)=w*nx+(j-w-1)*io+( i-in );
            end
        elseif j>y2
            f(i,j)=w*nx+io*in+(j-(w+in)-1)*nx+i;
        end
    end
end



    for j=1:y1-1
        for i=1:nx

         if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                continue

                %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         elseif j==1

            if i==1
                GM(f(i,j))=Tm(i,j)+4*Bi*Fo*Ta;
                CM(f(i,j),f(i,j))=C;
                CM(f(i,j),f(i,j+1))=2*D;
                CM(f(i,j),f(i+1,j))=2*D;


            elseif i==nx
                GM(f(i,j))=Tm(i,j)+4*Bi*Fo*Ta;
                CM(f(i,j),f(i,j))=C;
                CM(f(i,j),f(i-1,j))=2*D;
                CM(f(i,j),f(i,j+1))=2*D;
            else
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j+1))=2*D;
```

```
            end
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

         elseif j==y1-1
             if i==1
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;
                CM(f(i,j),f(i+1,j))=2*D;

             elseif i==nx
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;
                CM(f(i,j),f(i-1,j))=2*D;

             elseif i>=x1 && i<=x2
                GM(f(i,j))=Tm(i,j)-D*Tin;
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i,j-1))=D;
             else
                GM(f(i,j))=Tm(i,j);
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;

             end
             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         else
             if i==1
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i+1,j))=2*D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;
             elseif i==nx
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i-1,j))=2*D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;
             else
                GM(f(i,j))=Tm(i,j);
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i,j-1))=D;

             end


            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%55


        end

    end
end


        for j=y1:y2
            for i=1:nx


                if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                    continue


                elseif i==1
```

```
                    GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                    CM(f(i,j),f(i,j))=B;
                    CM(f(i,j),f(i+1,j))=2*D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;


                elseif i==x1-1
                    GM(f(i,j))=Tm(i,j)-D*Tin;
                    CM(f(i,j),f(i,j))=A;
                    CM(f(i,j),f(i-1,j))=D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;




                elseif i==x2+1
                     CM(f(i,j),f(i,j))=A;
                     CM(f(i,j),f(i,j-1))=D;
                     CM(f(i,j),f(i,j+1))=D;
                     CM(f(i,j),f(i+1,j))=D;
                     GM(f(i,j))=Tm(i,j)-D*Tin;


                elseif i==nx
                     CM(f(i,j),f(i,j))=B;
                     CM(f(i,j),f(i,j-1))=D;
                     CM(f(i,j),f(i-1,j))=2*D;
                     GM(f(i,j))=Tm(i,j)+2*Bi*Fo*Ta;
                     CM(f(i,j),f(i,j+1))=D;


                else
                     GM(f(i,j))=Tm(i,j);
                     CM(f(i,j),f(i,j))=A;
                     CM(f(i,j),f(i-1,j))=D;
                     CM(f(i,j),f(i,j+1))=D;
                     CM(f(i,j),f(i+1,j))=D;
                     CM(f(i,j),f(i,j-1))=D;


                end


            end

        end




for j=y2+1:ny
    for i=1:nx

        if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                continue

        elseif j==y2+1

            if i==1
                    GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                    CM(f(i,j),f(i,j))=B;
                    CM(f(i,j),f(i+1,j))=2*D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;
            elseif i==nx
                    GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                    CM(f(i,j),f(i,j))=B;
                    CM(f(i,j),f(i-1,j))=2*D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;

            elseif i>=x1 && i<=x2
                GM(f(i,j))=Tm(i,j)-D*Tin;
```

```
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i,j+1))=D;
            else
                GM(f(i,j))=Tm(i,j);
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i,j-1))=D;
            end
        elseif j==ny

            if i==1
                GM(f(i,j))=Tm(i,j)+4*Bi*Fo*Ta;
                CM(f(i,j),f(i,j))=C;
                CM(f(i,j),f(i,j-1))=2*D;
                CM(f(i,j),f(i+1,j))=2*D;

            elseif i==nx
                GM(f(i,j))=Tm(i,j)+4*Bi*Fo*Ta;
                CM(f(i,j),f(i,j))=C;
                CM(f(i,j),f(i,j-1))=2*D;
                CM(f(i,j),f(i-1,j))=2*D;
            else
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                CM(f(i,j),f(i,j))=B;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j-1))=2*D;
            end
        else
            if i==1
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                    CM(f(i,j),f(i,j))=B;
                    CM(f(i,j),f(i+1,j))=2*D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;
            elseif i==nx
                GM(f(i,j))=Tm(i,j)+2*Fo*Bi*Ta;
                    CM(f(i,j),f(i,j))=B;
                    CM(f(i,j),f(i-1,j))=2*D;
                    CM(f(i,j),f(i,j+1))=D;
                    CM(f(i,j),f(i,j-1))=D;
            else
                GM(f(i,j))=Tm(i,j);
                CM(f(i,j),f(i,j))=A;
                CM(f(i,j),f(i-1,j))=D;
                CM(f(i,j),f(i,j+1))=D;
                CM(f(i,j),f(i+1,j))=D;
                CM(f(i,j),f(i,j-1))=D;
            end

    end
    end

end

ANS=(CM)\GM;
clear i j
    for j=1:ny
        for i=1:nx
            if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                Tm(i,j)=T(i,j);
            else
               Tm(i,j)= ANS(f(i,j));
            end
        end

    end
end



function [T_PSOR,it_PSOR] = PSOR(T,nx,ny,x1,x2,y1,y2,Ta,Bi)
```

```
eps=0.001;
error2=10;
T_PSOR=T;              %gauss-seidel temperature matrix
it_PSOR=0;             %number of iterations until convergence
w=1.5;
while error2>eps

    T_old_PSOR=T_PSOR;       %T_GS at k-1 iteration

    for i=1:nx
        for j=1:ny


            if (i>=x1 && i<=x2) && (j>=y1 && j<=y2)
                continue

            elseif i==1 && j==1
                T_PSOR(i,j)= w/(2*Bi+2) * (T_old_PSOR(i+1,j)+T_old_PSOR(i,j+1)+2*Bi*Ta)+(1-
w)*T_old_PSOR(i,j);
            elseif i==1 && j>1 && j<ny
                T_PSOR(i,j)=w/(2*Bi+4)*(2*T_old_PSOR(i+1,j)+T_old_PSOR(i,j+1)+T_PSOR(i,j-
1)+2*Bi*Ta)+(1-w)*T_old_PSOR(i,j);

            elseif j==ny && i==1
                T_PSOR(i,j)=w/(2*Bi+2) *(T_old_PSOR(i+1,j)+T_PSOR(i,j-1)+2*Bi*Ta)+(1-
w)*T_old_PSOR(i,j);
            elseif i==nx && j==1
                T_PSOR(i,j)=w/(2*Bi+2)*(T_PSOR(i-1,j)+T_old_PSOR(i,j+1)+2*Bi*Ta)+(1-
w)*T_old_PSOR(i,j);
            elseif i==nx && j==ny
                T_PSOR(i,j)=w/(2*Bi+2)*(T_PSOR(i-1,j)+T_PSOR(i,j-1)+2*Bi*Ta)+(1-
w)*T_old_PSOR(i,j);


            elseif j==1 && i>1 && i<nx
                T_PSOR(i,j)=w/(2*Bi+4)*(2*T_old_PSOR(i,j+1)+T_old_PSOR(i+1,j)+T_PSOR(i-
1,j)+2*Bi*Ta)+(1-w)*T_old_PSOR(i,j);
            elseif j==ny && i>1 && i<nx
                T_PSOR(i,j)=w/(2*Bi+4)*(2*T_PSOR(i,j-1)+T_old_PSOR(i+1,j)+T_PSOR(i-
1,j)+2*Bi*Ta)+(1-w)*T_old_PSOR(i,j);

            elseif i==nx && j>1 && j<ny
                T_PSOR(i,j)=w/(2*Bi+4)*(2*T_PSOR(i-1,j)+T_old_PSOR(i,j+1)+T_PSOR(i,j-
1)+2*Bi*Ta)+(1-w)*T_old_PSOR(i,j);
            else
                T_PSOR(i,j)=w/4 *(T_old_PSOR(i+1,j)+T_PSOR(i-1,j)+T_PSOR(i,j-
1)+T_old_PSOR(i,j+1))+(1-w)*T_old_PSOR(i,j);
            end
        end
    end

    error2=max(max(abs(T_PSOR-T_old_PSOR)));
    it_PSOR=it_PSOR+1;
end


end


function [T_full,T1,t_steady,X,Y,MESH] = one_fourth(a0,ai,w,alpha,h,k,T0,Tin,Ta,eps,dt)

aa0=a0/2;
aai=ai/2;

W=abs(aai-aa0);                    %Channel Wall
                                %number of partitions in Channel Wall
dx=W/w;                            %delta x
dy=dx;


tf=3600*50;                        %Final Time (if needed-not used here)
x=0:dx:aai;                         %x vector
y=0:dy:aai;                         %y vector
nx=length(x);                      %Number of nodes in x direction
ny=length(y);                      %Number of nodes in x direction
```

```
Bi=h*dx/k;                          %finite-difference form of the Biot number

                        %Initial Temperature
T=ones(nx,ny)*T0;                   %Initializing Temperature


x1=find(x==W);  %Position Of Inner Corners
y1=find(x==W);


t=0:dt:tf;                                          %time vector
nt=length(t);                                       %number of time nodes

Fo=alpha*dt/(dx)^2;                                 %finite-difference form of the Fourier number
St1=Fo*(1+Bi);                                      %Stability Condtion
St2=Fo*(2+Bi);                                      %Stability Condtion

while Fo>1/4 || St1>1/4 || St2>1/2                  %Checking Stability
    disp('Not Stable')
    dt=input('Insert new dt:\n');
    Fo=alpha*dt/(dx)^2;
    St1=Fo*(1+Bi);
    St2=Fo*(2+Bi);
    t=0:dt:tf;
    nt=length(t);
end



for i=1:nx
    for j=1:ny
        if i<=x1 && j==y1
            T(i,j)=Tin;
        elseif j<=y1 && i==x1
            T(i,j)=Tin;

        elseif (i<x1  && j<y1)
            T(i,j)=nan;
        end

    end
end



E=1000;                                             %initial diffrence of two adjacent Temp.
Matrixes
it_f=0;                                             %number of iteration Until Steady State is
rechead
Tn=T;
while E>eps
    T=Tn;
for j=1:ny
        for i=1:nx
            if (i<=x1 && j<=y1)
                continue


            else
                if j==1
                    if i==nx
                        Tn(i,j)=Fo*(2*T(i-1,j)+2*T(i,j+1)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);
                    elseif i>x1 && i<nx
                        Tn(i,j)=Fo*(T(i+1,j)+T(i-1,j)+2*T(i,j+1)-4*T(i,j))+T(i,j);
                    end

                elseif j==ny
                    if i==nx
                        Tn(i,j)=2*Fo*(T(i-1,j)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-4*Bi*Fo)*T(i,j);
                    elseif i==1
                        Tn(i,j)=Fo*(2*T(i+1,j)+2*T(i,j-1)+2*Bi*Ta)+(1-4*Fo-2*Bi*Fo)*T(i,j);
                    else
                        Tn(i,j)=Fo*(2*T(i,j-1)+T(i+1,j)+T(i-1,j)+2*Bi*Ta)+(1-4*Fo-
2*Bi*Fo)*T(i,j);
                    end
                else
                    if i==1
```

```
                        Tn(i,j)=Fo*(2*T(i+1,j)+T(i,j-1)+T(i,j+1)-4*T(i,j))+T(i,j);
                    elseif i==nx
                        Tn(i,j)=Fo*(2*T(i-1,j)+T(i,j+1)+T(i,j-1)+2*Bi*Ta)+(1-4*Fo-
2*Bi*Fo)*T(i,j);
                    else
                        Tn(i,j)=Fo*(T(i+1,j)+T(i,j-1)+T(i,j+1)+T(i-1,j)-4*T(i,j))+T(i,j);
                    end
                end
            end
        end
end
    E=max(max(abs (Tn-T  )));                    %Maximum diffrence of two adjacent Temp.
Matrixes
    it_f=it_f+1;
end


t_steady=it_f*dt;
T=Tn;

T1=T;
T2=flip(T);
T3=flip(T,2);
T4=flip(flip(T,2));
T_full=[T4 T2;T3 T1];
a=length(T_full)/2;
T_full(a,:)=[];
T_full(:,a)=[];


[X,Y]=meshgrid(x,y);
for i=1:nx
    for j=1:ny
        if isnan(T(i,j))
            MESH(i,j)=nan;
        else
            MESH(i,j)=1;

            hold on
        end
    end
end
end
```