پروژه اول درس محاسبات عددی پیشرفته
انواع روش‌های حل معادلات غیرخطی

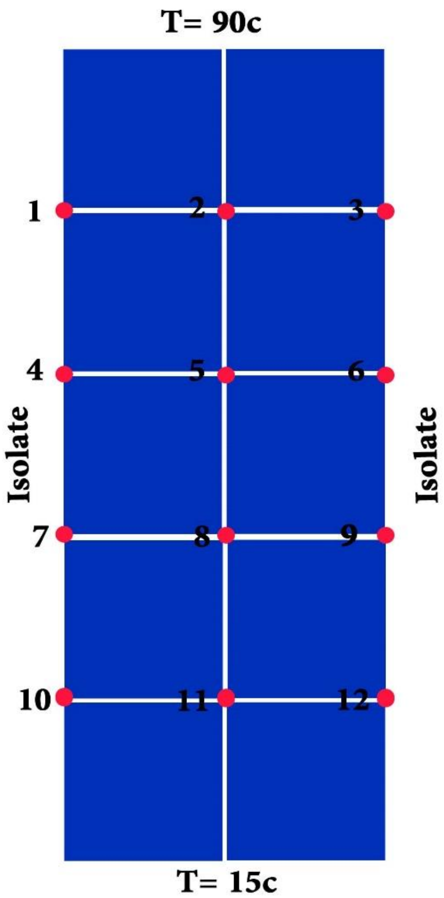
نویسنده : امیرحسین فروزنده نژاد

مقدمه

هدف از انجام این پروژه تفهیم بهتر مباحث تدریس شده در این فصل و همچنین مقایسه انواع روشهای حل دستگاه معادلات خطی از دو منظر همگرایی و سرعت همگرایی است. که در ادامه به حل یک مسئله 12 معادله 12 مجهول با جواب معلوم به 4 روش متفاوت خواهیم پرداخت.

معادله حاکم

برای بررسی دقت و سرعت حل 4 روش از روشهای حل عددی دستگاه معادلات خطی به بررسی معادله انتقال حرارت دو بعدی در یک جسم هادی حرارت پرداخته شده است که شرایط مرزی آن به شرح زیر میباشد:



شکل 1 : شرایط مرزی مسئله

لازم به ذکر است که در این مسئله $\Delta x = \Delta y$ است و از قبل میدانیم بدلیل تقارن شرایط مرزی تغییر دما در طول جسم به صورت یک بعدی و تابع خطی از y است که در y=0 برابر $T_{down} = 15^o$ و در y=l برابر $T_{down} = 90^o$ است.

شماره‌ی گره	دما (درجه سلیسیوس)	شماره‌ی گره	دما (درجه سلیسیوس)
1	75	7	45
2	75	8	45
3	75	9	45
4	60	10	30
5	60	11	30
6	60	12	30

جدول 1 : توضیع دمای گره ها

با نوشتن معادلات حاکم و ساده سازی ب 12 معادله‌ی زیر خواهیم رسید:

$$2 * T_2 + T_4 - 4 * T_1 = -90$$

$$T_1 + T_3 + T_5 - 4 * T_2 = -90$$

$$2 * T_2 + T_6 - 4 * T_3 = -90$$

$$2 * T_5 + T_1 + T_7 - 4 * T_4 = 0$$

$$T_2 + T_4 + T_6 + T_8 - 4 * T_5 = 0$$

$$2 * T_5 + T_3 + T_9 - 4 * T_6 = 0$$

$$2 * T_8 + T_4 + T_{10} - 4 * T_7 = 0$$

$$T_5 + T_7 + T_9 + T_{11} - 4 * T_8 = 0$$

$$2 * T_8 + T_6 + T_{12} - 4 * T_9 = 0$$

$$2 * T_{11} + T_7 - 4 * T_{10} = -15$$

$$T_8 + T_{10} + T_{12} - 4 * T_{11} = -15$$

$$2 * T_{11} + T_9 - 4 * T_{12} = -15$$

معادلات 1 تا 12: معادلات حاکم

روش‌های حل

1 – Gaussian elimination
2 – LU Decomp
3 – Jacobi
4 – Gauss seidle

مراحل قبل حل

از آنجایی که معادلات بدست آمده همگی به نوعی هستند که عدد قطر اصلی بزرگترین عدد آن سطر است (عدد 4) و باقی اعداد در رنج 0 تا 2 هستند پس نیازی به استفاده از pivoting وجود ندارد ولی استفاده از relaxation برای تسریع حل و یا جلوگیری از واگرا شدن جواب ها میتواند مفید باشد که در ادامه به بررسی آن پرداخته خواهد شد که برای این بررسی دقت حل 0.0001 قرار داده شده است و برای حدس اولیه از جدول زیر استفاده شده است

T1	70
T2	70
T3	70
T4	55
T5	55
T6	55
T7	40
T8	40
T9	40
T10	20
T11	20
T12	20

جدول 2: حدس های اولیه استفاده شد در روش 3 و4

تکنیک Relaxation

Jacobi

آلفا under) (relaxation	تعداد مراحل حل	آلفا (over relaxation)	تعداد مراحل حل
1	89	1	89
0.9	98	1.1	82
0.8	109	1.2	75
0.7	124	1.3	converge
0.6	142	1.4	Converge
0.5	167	1.5	Converge
0.4	204	1.6	Converge
0.3	263	1.7	Converge
0.2	375	1.8	Converge
0.1	679	1.9	converge
		2	51
		2.1	converge

جدول۳ : تاثیر relaxation بر روش Jacobi

Gauss seidle

آلفا (under relaxation)	تعداد مراحل حل	آلفا (over relaxation)	تعداد مراحل حل
1	50	1	50
0.9	60	1.1	41
0.8	72	1.2	34
0.7	88	1.3	26
0.6	107	1.4	18
0.5	134	1.5	19
0.4	172	1.6	25
0.3	232	1.7	36
0.2	346	1.8	53
0.1	654	1.9	110
		2	converge
		2.1	converge

جدول۴ : تاثیر relaxation بر روش Gauss seidle

کد

```
clc;clear;
[num,txt,raw] = xlsread('secondproject.xlsx');
nn = size(num);
n = nn(1,1);
a = zeros(n,n);
for i=1:n
    b(i,1)=num(i,n+1);
    for j=1:n
        a(i,j)=num(i,j);
    end
    xgg(i,1) = num(i,n+2);
end
x = zeros(n,4);



%% Gaussian elimination
A = zeros(n,n*n);
B = zeros(n,n);

for i=1:n
    B(i,1)=num(i,n+1);
    for j=1:n
        A(i,j)=num(i,j);
    end
end
for k=1:(n-1)
    for i=(k+1):n
        for j=(k+1):n
            A(i,k*n+j) = A(i,(k-1)*n+j) - (A(i,(k-1)*n+k)/A(k,(k-
1)*n+k))*A(k,(k-1)*n+j);
            B(i,k+1) = B(i,k) - (A(i,(k-1)*n+k)/A(k,(k-
1)*n+k))*B(k,k);
        end
    end
end

x(n,1) = B(n,n)/A(n,n*n);
for i=(n-1):(-1):1
    s=0;
    for j=i+1:n
        s = s+ A(i,(i-1)*n+j)*x(j,1);
    end
    x(i,1) = (B(i,i) - s)/A(i,(i-1)*n+i);
end
```

```
%% LU Decomp
l = zeros(n,n);
u = zeros(n,n);

for i=1:n
    l(i,1) = a(i,1);
    u(i,i) = 1;
    u(1,i) = a(1,i)/l(1,1);
end

for j = 2:n
    for i = 2:n
        if i<j
            s=0;
            for k=1:(j-1)
                 s = s + l(i,k)*u(k,j);
            end
            u(i,j) = (a(i,j)-s)/l(i,i);
        end

        if j<=i
            s=0;
            for k=1:(j-1)
                s = s + (l(i,k)*u(k,j));
            end
            l(i,j) = a(i,j)- s;
        end


    end
end
c(1,1) = b(1,1)/l(1,1);
for i=2:n
    s=0;
    for k=1:(i-1)
        s=s+l(i,k)*c(k,1);
    end
    c(i,1) = (b(i,1) - s) / l(i,i);
end

x(n,2) = c(n,1);
for i=(n-1):(-1):1
    s=0;
    for k=(i+1):n
        s=s+u(i,k)*x(k,2);
    end
    x(i,2) = c(i,1) - s;
end
```

```
%%
data3 = zeros(20,n);
data4 = zeros(20,n);

%% Jacobi
jj=0;
v = zeros(n,1);
v(1,1)=1;
alpha = 1;
xg = xgg;
while max(v)>0.0001
    jj=jj+1;
    if jj>1000
        disp('the 3th method isnt converge');
        break;
    end

    ss = zeros(n,1);
    for i=1:n
        for j=1:n
            if j~=i
               ss(i,1) = ss(i,1)+ a(i,j)*xg(j,1);

            end
        end

        x(i,3) = xg(i,1) + (alpha/a(i,i)) * ( b(i,1)-ss(i,1)-
a(i,i)*xg(i,1) );

    end
    for i=1:n
            v(i,1) = abs(x(i,3)-xg(i,1));
            xg(i,1) = x(i,3);
    end
end


%% Gauss sidel method
jjj=0;
v = zeros(n,1);
v(1,1)=1;
% alpha = 1;

while max(v)>0.0001
    jjj=jjj+1;
    if jjj>1000
        disp('the 4th method isnt converge');
```

```
        break;
    end

    ss = zeros(n,1);
    for i=1:n
        for j=1:n
            if j~=i
               ss(i,1) = ss(i,1)+ a(i,j)*xgg(j,1);

            end
        end

        x(i,4) = xgg(i,1) + (alpha/a(i,i)) * ( b(i,1)-ss(i,1)-
a(i,i)*xgg(i,1) );
        v(i,1) = abs(x(i,4)-xgg(i,1));
        xgg(i,1) = x(i,4);

    end

end
x
```