The Project 2: Linear system solving

Kaiwen Chen[1]

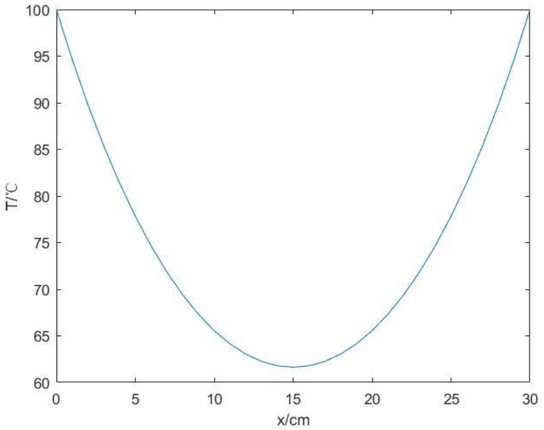
Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong

I. The Results

1. The optimal relaxation factor:

ω_{opt}=1.78

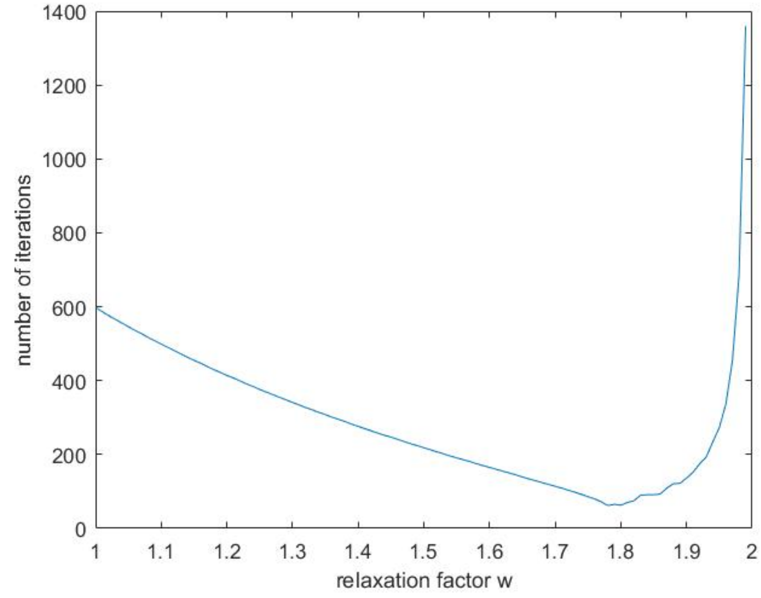
2. The curve of T along x based on ω_{opt}



T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	
100	94.65731	89.79182	85.37899	81.39657	77.82448	74.64472	71.84124	69.39992	67.30844	
T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}	T_{18}	T_{19}	
65.55626	64.134549	63.03614	62.25549	61.78867	61.63333	61.78868	62.25551	63.03617	64.13459	
T_{20}	T_{21}	T_{22}	T_{23}	T_{24}	T_{25}	T_{26}	T_{27}	T_{28}	T_{29}	T_{30}
65.55632	67.308523	69.40003	71.84137	74.64487	77.82465	81.39673	85.37912	89.79194	94.65739	100

3. The relationship between ω and the total number of iteration n before convergence

[1] Master of Science, Department of Mechanical and Aerospace Engineering,
Email:kchenau@connect.ust.hk



The Data

relaxation factor: w	1	1.2	1.3	1.4	1.5	1.6	1.7	1.71	1.72	1.73	1.74	1.75	1.76	1.77	1.78	1.79	1.8	1.9	1.99
the number of iterations needed for convergence: n	598	414	341	276	218	165	114	109	103	98	92	86	80	72	62	65	63	136	1360

II. The Process

A. Mathematical Models

The finite difference approximation form of the ordinary differential equation is:

$$\frac{T_{i+1} - 2T_i + T_{i-1}}{(\Delta x)^2} - \mathrm{G}T_i = 0$$

Since the length of the bar is 30cm and it is divided into 30 parts, the Δx equals 1cm. The constant G is $(0.071)^2$ cm^2. The $\frac{1}{(\Delta x)^2}$ and G have the same unit. So the equation can be written as:

$$T_{i+1} - 2T_i + T_{i-1} - 0.071^2 T_i = 0$$

which is equal to:

$$T_{i+1} = 2.005041T_i - T_{i-1} \qquad i = 1,2,3 \ldots 29$$

with T_0 and T_{30} are given as 100^0C

$$2.005041T_1 - T_2 = T_0 == 100$$
$$-T_1 + 2.005041T_2 - T_3 = 0$$
$$-T_2 + 2.005041T_3 - T_4 = 0$$
$$\vdots$$
$$-T_{27} + 2.005041T_{28} - T_{29} = 0$$
$$-T_{28} + 2.005041T_{29} = T_{30} == 100$$

So the matrix A is

[2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
-1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1 0
0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 2.005041 -1
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1
2.005041 -1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1
2.005041 -1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1
2.005041 -1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
-1 2.005041 -1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 -1 2.005041 -1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 -1 2.005041 -1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 -1 2.005041 -1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 -1 2.005041 -1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 -1 2.005041 -1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 -1 2.005041 -1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 -1 2.005041]

The column vector T is [T_1;T_2;T_3;…;T_{29}]
The column vector b is [100;0;…;0;100]

B. The algorithm

T_i =(T_{i+1} + T_{i-1})/2.005041; i starts from 1 to 29; iteration step by step until the converge condition meets ;

III. The Code

1. Calculate with different relaxation factors

```
% Record the relaxation factor w and number of
iterations k

recordw=0;
recordk=0;
```

```
 % n is the index

n=1;

 % w is the relaxation factor
for w=1:0.01:1.99

 % T is the old temperature differences
 % T2 is the new temperature differences
 % Initial guess£°0

T=[100;zeros(29,1);100];
T2=[100;zeros(29,1);100];

 % k calculate the number of iterations for each
relaxation factor

k=0;

while 1

 % Gauss-Sidel iteration

for i=2:30
    T2(i)=(T2(i+1)+T2(i-1))/2.005041;
    T2(i)=w*T2(i)+(1-w)*T(i);
end

 % Add the number of iteration

k=k+1;

 % Convergence condition

 if  max(abs(T-T2))<0.0001
    break;
 else
    T=T2;
 end

end

 % Record the w and k
```

```
recordw(n)=w;
recordk(n)=k;

 % Change the relaxation factor w and the index n

n=n+1;
end

plot(recordw,recordk);
xlabel('relaxation factor w')
ylabel('number of iterations')
```

2. Calculate with the optimal relaxation factor

```
%woptimal=1.78
w=1.78;
T=[100;zeros(29,1);100];
T2=[100;zeros(29,1);100];
k=0;
%T1=100 T31=100
while 1
for i=2:30
    T2(i)=(T2(i+1)+T2(i-1))/2.005041;
    T2(i)=w*T2(i)+(1-w)*T(i);
end
k=k+1;
if  max(abs(T-T2))<0.0001
    break;
else
    T=T2;
end
end
plot(0:1:30,T2);
xlabel('x/cm')
ylabel('T/¡æ')
```