# problem

## problem 1

1. Write the equivalent governing equation in the domain

the equivalent governing equation is:

1. Programming

get the mesh from triangulation mesh generator:

use matlab to solve problem 1

read mesh information and set b as 10:

clear;clc;close all;  
  
NE = textread('mesh.txt', '%d', 2);  
N = NE(1);  
E = NE(2);  
data = readmatrix('mesh.txt');  
position = data(1:N, 1:2);  
connect = round(data(N+1:N+E, 2:4)) + 1;  
b = 10;  
alpha = 30\*pi/180;

the position of node to solve is as below:

NI = [];  
toler = 1e-5;  
for i = 1: N  
 x = position(i, 1);  
 y = position(i, 2);  
 a1 = abs(x - b);  
 a2 = abs(y - tan(alpha) \* x);  
 a3 = abs(y + tan(alpha) \* x);  
 if a1 > toler && a2 > toler && a3 > toler  
 NI = [NI, i];  
 end  
end

the core codes of FEM:

K = zeros(N, N);  
F = zeros(1, N);  
u = zeros(1, N);  
for k = 1: E  
 node1 = connect(k, 1);  
 node2 = connect(k, 2);  
 node3 = connect(k, 3);  
 x1 = position(node1, 1);  
 y1 = position(node1, 2);  
 x2 = position(node2, 1);  
 y2 = position(node2, 2);  
 x3 = position(node3, 1);  
 y3 = position(node3, 2);  
 A2 = 2\*det([1 x1 y1; 1 x2 y2; 1 x3 y3]);  
 k11 = (x2 - x3)^2 + (y2 - y3)^2;  
 k12 = (x1 - x3) \* (-x2 + x3) + (y1 - y3) \* (-y2 + y3);  
 k13 = (x1 - x2) \* (x2 - x3) + (y1 - y2) \* (y2 - y3);  
 k22 = (x1 - x3)^2 + (y1 - y3)^2;  
 k23 = -(x1^2 + x2 \* x3 - x1 \* (x2 + x3) + (y1 - y2) \* (y1 - y3));  
 k33 = (x1 - x2)^2 + (y1 - y2)^2;  
 F1 = (x3 \* (y1 - y2) + x1 \* (y2 - y3) + x2 \* (-y1 + y3)) / 6;  
 F2 = (x3 \* (y1 - y2) + x1 \* (y2 - y3) + x2 \* (-y1+y3)) / 6;  
 F3 = (x3 \* (y1 - y2) + x1 \* (y2 - y3) + x2 \* (-y1+y3)) / 6;  
 K(node1, node1) = K(node1, node1) + k11 / A2;  
 K(node1, node2) = K(node1, node2) + k12 / A2;  
 K(node1, node3) = K(node1, node3) + k13 / A2;  
 K(node2, node1) = K(node2, node1) + k12 / A2;  
 K(node2, node2) = K(node2, node2) + k22 / A2;  
 K(node2, node3) = K(node2, node3) + k23 / A2;  
 K(node3, node1) = K(node3, node1) + k13 / A2;  
 K(node3, node2) = K(node3, node2) + k23 / A2;  
 K(node3, node3) = K(node3, node3) + k33 / A2;  
 F(node1) = F(node1) + F1;  
 F(node2) = F(node2) + F2;  
 F(node3) = F(node3) + F3;  
end

solve by inverse of matrix and draw the 3D result:

u(NI) = K(NI, NI) \ F(NI).';  
trisurf(connect, position(:, 1), position(:, 2), u)  
  
view([-62.63 75.27])  
exportgraphics(gca, 'res1.png')  
view([-0.44 90.00])  
exportgraphics(gca, 'res2.png')  
view([55.41 66.95])  
exportgraphics(gca, 'res3.png')

the 3 views of ：

## problem 2

Lagrange equation:

for this system, we have

the velocity:

thus we have kietic energy of the system:

and the potential energy is:

thus we will have:

## problem 3

governing equation:

local stiffness relationship:

the matlab code and result are as below:

set the calculation parameters

clear;clc;close all;  
  
q\_ = -200;  
F\_ = -1000;  
M\_ = 2000;  
L = 0.12;  
d1 = 0.03;  
d2 = 0.02;  
Es = 200e9;  
I1 = pi \* d1^4 / 64;  
I2 = pi \* d2^4 / 64;  
N = 51;  
x = linspace(0, 2\*L, N);  
h = x(2) - x(1);

get the stiffness matrix (core code):

K = zeros(2\*N, 2\*N);  
b = zeros(2\*N);  
localk = [12/h^3, 6/h^2, -12/h^3, 6/h^2;  
 6/h^2, 4/h, -6/h^2, 2/h;  
 -12/h^3, -6/h^2, 12/h^3, -6/h^2;  
 6/h^2, 2/h, -6/h^2, 4/h];  
localb = h \* [1/2, h/12, 1/2, -h/12];  
not\_boundary = [3:2\*N-2, 2\*N];  
  
for k = 1:N-1  
 if k <= (N-1)/2  
 K(2\*k-1:2\*k+2, 2\*k-1:2\*k+2) = K(2\*k-1:2\*k+2, 2\*k-1:2\*k+2) + Es \* I1 \* localk;  
 b(2\*k-1:2\*k+2) = b(2\*k-1:2\*k+2) + q\_ \* localb;  
 else  
 K(2\*k-1:2\*k+2, 2\*k-1:2\*k+2) = K(2\*k-1:2\*k+2, 2\*k-1:2\*k+2) + Es \* I2 \* localk;  
 end  
end  
b(N) = b(N) + F\_;  
b(2\*N) = b(2\*N) - M\_;

solve:

bm = b(not\_boundary);  
Km = K(not\_boundary, not\_boundary);  
solution = zeros(2\*N);  
solution(not\_boundary) = Km \ bm.';  
w = solution(1:2:2\*N);  
theta = solution(2:2:2\*N);

draw the deflection picture:

plot(x, w)  
xlabel('x')  
ylabel('w')  
title('w(x)-x')  
exportgraphics(gca, 'w.png')  
hold off;

draw the torsion angle deflection:

plot(x, theta\*180/pi)  
xlabel('x')  
ylabel('\theta')  
title('\theta\circ(x)-x')  
exportgraphics(gca, 'theta.png')

the solution result is as below, where the deflection of the beam is:

the torsion angle is: