The solution of statically determinant and indeterminant trusses with the method of Finite Elements and MATLAB
example problem

$E = 210 \text{ GPa} = 210 \times 10^6 \text{ kN/m}^2$

$A = 10^{-4} \text{ m}^2$ for all the bars

Note: this truss is statically determinant
%truss_script3
Nn = 3; % number of nodes
Ne = 3; % number of elements (bars)

% need to give all area (A), Youngs modulus (E), length (L)
% and angle (theta) values for each bar
A = (1.0E-4)*ones(1,Ne);
E = (210.0E6)*ones(1,Ne);
L = [4 sqrt(13) sqrt(13)];
ang2 = (atan(3/2))*180/pi;
ang3 = 180 - (atan(3/2))*180/pi;
theta = [0 ang2 ang3];

% compute all the stiffnesses of the bars
kconstant = A.*E./L;
% give the connectivity matrix, where each row has the values
% [start_node_number  end_node_number ]
connect = [1 2; 1 3; 2 3]; % must be matrix of dimensions Ne x 2

note: if we had changed the order to 3 2 for the third element then that would change the angle (it would be negative)

similarly, if we had changed the order to 3 1 for the second element the angle would be:
% give boundary conditions where each row has the values
% [global_unknown_number value type]
% where type == 1 indicates a displacement b.c.
% and type == 2 indicates a force b.c.
bc = [1 0 1; 2 0 1; 3 0 2; 4 0 1; 5 5 2; 6 -10 2];

\[
\begin{align*}
F_{xn} &= F_{2n-1} = 5 & \text{node number n = 3} \\
F_{yn} &= F_{2n} = -10
\end{align*}
\]

node number n=1
\[
\begin{align*}
U_{xn} &= U_{2n-1} = 0 \\
U_{yn} &= U_{2n} = 0
\end{align*}
\]

node number n=2
\[
\begin{align*}
F_{xn} &= F_{2n-1} = 0 \\
F_{yn} &= F_{2n} = 0 \\
U_{yn} &= U_{2n} = 0
\end{align*}
\]
Ke = zeros(4,4,Ne); % element stiffness matrix initialization
K = zeros(2*Nn,2*Nn); % global stiffness matrix initialization

% generate element stiffness matrices and assemble into global stiffness matrix
for nn=1:Ne
    Ke = truss_stiffness(kconstant(nn), theta(nn));
    K = truss_assemble(K, Ke, connect(nn, 1), connect(nn,2));
end
function Ke = truss_stiffness( k, theta)
    x=theta*pi/180;
    C=cos(x);
    S =sin(x);
    Ke = k*[ C*C  C*S -C*C  -C*S;  C*S  S*S  -C*S  -S*S; ...
             -C*C  -C*S  C*C  C*S ;  -C*S  -S*S  C*S  S*S];
function y = truss_assemble( K, k, i, j)
K(2*i -1, 2*i -1) = K(2*i-1,2*i-1) + k(1,1);
K(2*i -1, 2*i ) = K(2*i-1,2*i) + k(1,2);
K(2*i -1, 2*j -1) = K(2*i-1,2*j-1) + k(1,3);
K(2*i -1, 2*j ) = K(2*i-1,2*j) + k(1,4);
K(2*i , 2*i -1) = K(2*i,2*i-1) + k(2,1);
K(2*i , 2*i ) = K(2*i,2*i) + k(2,2);
K(2*i , 2*j -1) = K(2*i,2*j-1) + k(2,3);
K(2*i , 2*j ) = K(2*i,2*j) + k(2,4);
K(2*j -1, 2*i -1) = K(2*j-1,2*i-1) + k(3,1);
K(2*j -1, 2*i ) = K(2*j-1,2*i) + k(3,2);
K(2*j -1, 2*j -1) = K(2*j-1,2*j-1) + k(3,3);
K(2*j -1, 2*j ) = K(2*j-1,2*j) + k(3,4);
K(2*j , 2*i -1) = K(2*j,2*i-1) + k(4,1);
K(2*j , 2*i ) = K(2*j,2*i) + k(4,2);
K(2*j , 2*j -1) = K(2*j,2*j-1) + k(4,3);
K(2*j , 2*j ) = K(2*j,2*j) + k(4,4);
y = K;
% initialize force vector on right hand side of global equations
f=zeros(1,2*Nn);

% apply boundary conditions, modifying the stiffness matrix
% and force vector on the right hand side

[Km, fm]=Boundary_Conditions(K, f, bc);
function [K, f]=Boundary_Conditions(K,f,bc)
bcl =size(bc); % size of the boundary condition matrix
nbc =bcl(1); % number of boundary conditions
[maxK, i] = max(K(:)); % largest element in global stiffness matrix
% apply boundary conditions and modify stiffness matrix
for ii =1:nbc
    nval=bc(ii,1);
    type =bc(ii,3);
    if type == 2
        f(nval) =bc(ii,2);
    elseif type ==1
        K(nval,nval) =maxK*1.0E9;
        f(nval) = bc(ii,2)*maxK*1.0E9;
    end
end
% solve system of equations
\[ u = K m \backslash f m \]

\[
\begin{align*}
U_1 &= 0 \\
U_2 &= 0 \\
U_3 &= 0.0011 \\
U_4 &= 0 \\
U_5 &= 0.0020 \\
U_6 &= -0.0016
\end{align*}
\]
% evaluate the nodal forces
fn = Nodal_Forces(K, u)

function y = Nodal_Forces(Kg, u)
  y = Kg*u;

F_1 = -5
F_2 = 1.25
F_3 = 0
F_4 = 8.75
F_5 = 5
F_6 = -10

fn =
-5.0000
1.2500
0.0000
8.7500
5.0000
-10.0000
% determine the forces in each bar
fb = zeros(1,Ne);  
for jj = 1:Ne
    fb(jj) = bar_forces(u, kconstant(jj), theta(jj), connect(jj,1), connect(jj,2));
end

fb

function y = bar_forces(u, k, theta, start, finish)
c = cos(theta*pi/180);
s = sin(theta*pi/180);
y = k*[-c -s c s]*[u(2*start-1);u(2*start);u(2*finish-1); u(2*finish)];
\[ fb = \\
5.8333 \\
-1.5023 \\
-10.5162 \]