```
function dibujogdl(X,Y,NI,NJ,CG)
%
% Programa para dibujar una estructura plana considerando los gdl
%
% Por: LEONES OLIVES GABRIEL
%          CEINCI-ULEAM
%        JULIO 2019
%-------------------------------------------------------------
% dibujo (X,Y,NI,NJ,CG)
%-------------------------------------------------------------
% X       Vector que contiene coordenadas en X
% Y       Vector que contiene coordenadas en Y
% NI      Vector con los nudos iniciales de los elementos
% NJ      Vector con los nudos finales de los elementos
% CG      Matriz de coordenadas generalizadas
%%
mbr=length(NI);
for i=1:mbr
   dx=X(NJ(i))-X(NI(i));dy=Y(NJ(i))-Y(NI(i));
   L(i)=sqrt(dx*dx+dy*dy);

end
Lo=0;
for i=1:mbr
   Lo=Lo+L(i);
end
prolo=Lo/mbr;

%%
```

```
tfl=6; %Tamaño de la Flecha
hfl=prolo*0.35; %Altura  de la flecha
afl=prolo*0.35; %Largo de la flecha
tlt=8; %Tamaño de la Letra
radio=prolo*0.18;% Radio de la flecha
clr='red';% Color principal de la flecha
clb='b';% Color secundario de la flecha
dfh=0.4;% Defase de la letra horizontal
dfv=0.05;% Defase de la letra vertical
dfv2=prolo*0.07;% Defase de la letra vertical 2
dfh2=tlt*0.025+(radio*1);
tmn=15; %Tamaño del nudo



%% Dibujo de la estructura
x1=min(X)-1;x2=max(X)+1;y1=min(Y)-1;y2=max(Y)+1;mbr=length(NI);
figure (1),title('Esquema Estructural'),xlim([x1 x2]) ,ylim([y1 y2])
if max(Y) > 3
   for i=1:mbr
      line([X(NI(i)) X(NJ(i))], [Y(NI(i)) Y(NJ(i))],'Color','k')
   end
else
   for i=1:mbr
      line([X(NI(i)) X(NJ(i))], [Y(NI(i)) Y(NJ(i))],'Color','k','LineWidth',max(Y))
   end
end
axis equal
```

```
hold on
%% Dibujo de los nudos
plot(X,Y,'.','MarkerEdgeColor',clr,'markersize',tmn)
%% Dibujo de los Grados libertad vertical
for i=1:length(X)
NUG(i)=i;
end
NUG=NUG';
VG=[NUG CG(:,2)];
u=1;
for i=1:length(NUG)
   if CG(i,2)>0
      NVG(u,:)=VG(i,:);
      u=u+1;
   end
end
VNG=NVG(:,1);
YV=Y(VNG);
XV=X(VNG);
for i=1:length(VNG)
ha = annotation('arrow');  % almacenamiento de la flecha en ha
ha.Parent = gca;          % asociar la flecha a los ejes actuales
ha.X = [XV(i) XV(i)];         %ubicacion en unidades de datos
ha.Y = [YV(i) YV(i)+hfl];
ha.Color = clr;
ha.HeadWidth  = tfl;
ha.HeadLength = tfl;
end
x =XV+dfv;
```

```
y =YV+hfl/2;
a = NVG(:,2); b = num2str(a); c = cellstr(b);
h1=text(x, y, c);
set(h1,'Color',clr,'FontSize',tlt);
hold on
% obtencion del limite de abcisado vertical
for i=1:length(VNG)
hy(i,:) = [YV(i) YV(i)+hfl];
end
maha = max(hy(:,2));
miha = min(hy(:,1));
may=max(Y);
miy=min(Y);
if maha>=may
mahy=maha;
else
mahy=may;
end
if miha>=miy
mihy=miy;
else
mihy=miha;
end


%% Dibujo de los Grados libertad horizontal
AG=[NUG CG(:,1)];
u=1;
```

```
for i=1:length(NUG)
   if CG(i,2)>0
      NAG(u,:)=AG(i,:);
      u=u+1;
   end
end
ri=max(NAG(:,2));
for i=1:ri
   rb(i)=i;

end

RESP=histc(NAG(:,2),rb);



if RESP(1)>1 | RESP(length(rb))>1
u=1;
r=max(NAG(:,2));
for i=length(NAG(:,1)):-1:1
   if r==NAG(i,2)
      axd2(u,:)=NAG(i,:);
       else
      r=NAG(i,2);
      u=u+1;
   end
end

u=1;
for i=length(axd2(:,1)):-1:1
```

```
    axd3(u,:)=axd2(i,:);
     u=u+1;
end
XAI=X(axd3(:,1));
YAI=Y(axd3(:,1));
XAD=X(axd2(:,1));
YAD=Y(axd2(:,1));
for i=1:length(axd3(:,1))
hb = annotation('arrow');  % almacenamiento de la flecha en ha
hb.Parent = gca;          % asociar la flecha a los ejes actuales
hb.X = [XAI(i)-afl XAI(i)];        % ubicacion en unidades de datos
hb.Y = [YAI(i) YAI(i)];
hb.Color = clb;
hb.HeadWidth  = tfl;
hb.HeadLength = tfl;
end
x2 =XAI-afl;
y2 =YAI+dfv2;
a2 = axd3(:,2); b2 = num2str(a2); c2 = cellstr(b2);
h2=text(x2, y2, c2);
set(h2,'Color',clb,'FontSize',tlt);
hold on

% obtencion del limite de abcisado horizontal
for i=1:length(axd3(:,1))
hx(i,:) = [XAI(i)-afl XAI(i)];
end
mahb = max(hx(:,2));
mihb = min(hx(:,1));
```

```
ma=max(X);
mix=min(X);
if mahb>=ma
mahx=mahb;
else
mahx=ma;
end
if mihb>=mix
mihx=mix;
else
mihx=mihb;
end
axis([mihx-prolo*0.18 mahx+prolo*0.18 mihy-prolo*0.18 mahy+prolo*0.18])

else

ANG=NAG(:,1);
YA=Y(ANG);
XA=X(ANG);
for i=1:length(ANG)
hb = annotation('arrow'); % almacenamiento de la flecha en ha
hb.Parent = gca;          % asociar la flecha a los ejes actuales
hb.X = [XA(i)-afl XA(i)];         % ubicacion en unidades de datos
hb.Y = [YA(i) YA(i)];
hb.Color = clr;
hb.HeadWidth  = tfl;
hb.HeadLength = tfl;
end
x2 =XA-afl;
```

```
y2 =YA+dfv2;
a2 = NAG(:,2); b2 = num2str(a2); c2 = cellstr(b2);
h2=text(x2, y2, c2);
set(h2,'Color',clr,'FontSize',tlt);
hold on
% obtencion del limite de abcisado horizontal
for i=1:length(ANG)
hx(i,:) = [XA(i)-afl XA(i)];
end
mahb = max(hx(:,2));
mihb = min(hx(:,1));
ma=max(X);
mix=min(X);
if mahb>=ma
mahx=mahb;
else
mahx=ma;
end
if mihb>=mix
mihx=mix;
else
mihx=mihb;
end
axis([mihx-prolo*0.18 mahx+prolo*0.18 mihy-prolo*0.18 mahy+prolo*0.18])

end
%% Dibujo de los Grados libertad de Giro
[dim1,dim2]=size(CG);
if dim2==3
```

```
   GG=[NUG CG(:,3)];
   u=1;
for i=1:length(NUG)
   if CG(i,2)>0
      NGG(u,:)=GG(i,:);
      u=u+1;
   end
end
GNG=NGG(:,1);
YG=Y(GNG);
XG=X(GNG);
x3 =XG-dfh2;
y3 =YG-hfl/2;
a3 = NGG(:,2); b3 = num2str(a3); c3 = cellstr(b3);
h3=text(x3, y3, c3);
set(h3,'Color',clr,'FontSize',tlt);
hold on
   for i=1:length(NGG(:,2))
% De los valores de ajuste deseados para la primera flecha
radius = radio; % Altura de arriba a abajo
centre = [XG(i) YG(i)];
arrow_angle = 240; % Ángulo de orientación deseado en grados
angle = -200; % Ángulo entre el inicio y el final de la flec
direction = 1; % para CW ingrese 1, para CCW ingrese 0
colour = clr; % Color de flecha
head_size = tfl; % Tamaño de la cabeza de flecha
   % needs hold on
% El centro de verificación es un vector con dos puntos
[m,n] = size(centre);
```

```
if m*n ~= 2
   error('Centre must be a two element vector');
end
arrow_angle = deg2rad(arrow_angle); % Convertir ángulo a rad
angle = deg2rad(angle); % Convertir ángulo a rad
xc = centre(1);
yc = centre(2);
% Crear valores (x, y) que están en la dirección positiva a lo largo de la x
% eje y la misma altura que el centro
x_temp = centre(1) + radius;
y_temp = centre(2);
% Crear valores X y Y para los puntos de inicio y final del arco
x1 = (x_temp-xc)*cos(arrow_angle+angle/2) - ...
      (y_temp-yc)*sin(arrow_angle+angle/2) + xc;
x2 = (x_temp-xc)*cos(arrow_angle-angle/2) - ...
      (y_temp-yc)*sin(arrow_angle-angle/2) + xc;
x0 = (x_temp-xc)*cos(arrow_angle) - ...
      (y_temp-yc)*sin(arrow_angle) + xc;
y1 = (x_temp-xc)*sin(arrow_angle+angle/2) + ...
      (y_temp-yc)*cos(arrow_angle+angle/2) + yc;
y2 = (x_temp-xc)*sin(arrow_angle-angle/2) + ...
      (y_temp-yc)*cos(arrow_angle-angle/2) + yc;
y0 = (x_temp-xc)*sin(arrow_angle) + ...
      (y_temp-yc)*cos(arrow_angle) + yc;
% Trazar dos veces para obtener ángulos mayores de 180
i = 1;
%Creacion de puntos
P1 = struct([]);
P2 = struct([]);
```

```
P1{1} = [x1;y1]; % Point 1 - 1
P1{2} = [x2;y2]; % Point 1 - 2
P2{1} = [x0;y0]; % Point 2 - 1
P2{2} = [x0;y0]; % Point 2 - 1
centre = [xc;yc]; % centro de garantía es la dimensión correcta
n = 1000; % El número de puntos en el arco
v = struct([]);

while i < 3
   v1 = P1{i}-centre;
   v2 = P2{i}-centre;
   c = det([v1,v2]); % "producto cruzado" de v1 y v2
   a = linspace(0,atan2(abs(c),dot(v1,v2)),n); % Rango de ángulo
   v3 = [0,-c;c,0]*v1; % v3 se encuentra en el plano de v1 y v2 y es ortog. a v1
   v{i} = v1*cos(a)+((norm(v1)/norm(v3))*v3)*sin(a); % Arco, centro en (0,0)
   plot(v{i}(1,:)+xc,v{i}(2,:)+yc,'Color', colour) %Trazar arco, centrado en P0
   i = i + 1;
end
position = struct([]);
% Configuración de x e y para flechas CW y CCW
if direction == 1
   position{1} = [x2 y2 x2-(v{2}(1,2)+xc) y2-(v{2}(2,2)+yc)];
elseif direction == -1
   position{1} = [x1 y1 x1-(v{1}(1,2)+xc) y1-(v{1}(2,2)+yc)];
elseif direction == 2
   position{1} = [x2 y2 x2-(v{2}(1,2)+xc) y2-(v{2}(2,2)+yc)];
   position{2} = [x1 y1 x1-(v{1}(1,2)+xc) y1-(v{1}(2,2)+yc)];
elseif direction == 0
   % no hacer nada
```

```
else
   error('direction flag not 1, -1, 2 or 0.');
end
% Bucle para cada punta de flecha
i = 1;
while i < abs(direction) + 1
   h=annotation('arrow'); %punta de flecha
   set(h,'parent', gca, 'position', position{i}, ...
      'HeadLength', head_size, 'HeadWidth', head_size,...
       'linestyle','none','Color', colour);
   i = i + 1;
end
   end
end
return
% ---end---
```