APÉNDICE B: PROGRAMA:

```
function matrizdesalida=const_mat_sal(datosdsal,tipo)

%Se trata de pasar a una matriz la informacion
c=1;
mn=size(datosdsal{1});
if mn(1)>1,
  matrizdesalida(:,c)=datosdsal{1};
else,
  matrizdesalida(:,c)=datosdsal{1}';
end
c=c+1;
mn=size(datosdsal{2}{3});
if mn(1)>1,
  matrizdesalida(:,c)=datosdsal{2}{3};
else,
  matrizdesalida(:,c)=datosdsal{2}{3}';
end
c=c+1;
if length(datosdsal{2})>4,
  mn=size(datosdsal{2}{5});
  if mn(1)>1,
    matrizdesalida(:,c)=datosdsal{2}{5};
  else,
    matrizdesalida(:,c)=datosdsal{2}{5}';
  end
  c=c+1;
  mn=size(datosdsal{3}{1});
  if mn(1)>1,
    matrizdesalida(:,c)=datosdsal{3}{1};
  else,
    matrizdesalida(:,c)=datosdsal{3}{1}';
  end
  c=c+1;
  mn=size(datosdsal{3}{2});
  if mn(1)>1,
    matrizdesalida(:,c)=datosdsal{3}{2};
  else,
    matrizdesalida(:,c)=datosdsal{3}{2}';
  end;
  c=c+1;
  if length(datosdsal{3})>2,
    mn=size(datosdsal{3}{3});
    if mn(1)>1,
      matrizdesalida(:,c)=datosdsal{3}{3};
    else,
      matrizdesalida(:,c)=datosdsal{3}{3}';
    end
    c=c+1;
    mn=size(datosdsal{3}{4});
    if mn(1)>1,
      matrizdesalida(:,c)=datosdsal{3}{4};
    else,
```

```
      matrizdesalida(:,c)=datosdsal{3}{4}';
    end
    c=c+1;
  end
  mn=size(datosdsal{4});
  if mn(1)>1,
    matrizdesalida(:,c)=datosdsal{4};
  else,
    matrizdesalida(:,c)=datosdsal{4}';
  end
  c=c+1;
  mn=size(datosdsal{5});
  if mn(1)>1,
    matrizdesalida(:,c)=datosdsal{5};
  else,
    matrizdesalida(:,c)=datosdsal{5}';
  end
  c=c+1;
  for i=1:length(datosdsal{6}),
    mn=size(datosdsal{6}{i});
    if mn(1)>1,
      matrizdesalida(:,c)=datosdsal{6}{i};
    else,
      matrizdesalida(:,c)=datosdsal{6}{i}';
    end
    c=c+1;
  end
  if tipo>=5,
     for i=1:length(datosdsal{7}),
       mn=size(datosdsal{7}{i});
       if mn(1)>1,
         matrizdesalida(:,c)=datosdsal{7}{i};
       else,
         matrizdesalida(:,c)=datosdsal{7}{i}';
       end
       c=c+1;
     end
  end,
  matrizdesalida(1,c)=tipo;
  matrizdesalida(2,c)=datosdsal{8}*10000;
  if tipo>=5;
     matrizdesalida(3,c)=datosdsal{9};
     for i=4:length(datosdsal{1}),
        matrizdesalida(i,c)=-1;
     end
  else,
     for i=3:length(datosdsal{1}),
        matrizdesalida(i,c)=-1;
     end
  end
end



function pp=construyepp(DAT)
```

```
%Se trata de cumplir las condiciones en posición (2ª columna de DAT),
%en velocidad (3ªcolumna),y en aceleración (4ªcolumna),en los puntos
%psi,de la 1ªcolumna.La matriz estará llena.
%Utilizaremos polinomios de quinto orden.

n=size(DAT);
n=n(1);

for i=1:(n-1),
   d=DAT(i+1,1)-DAT(i,1);  %distancia entre puntos consecutivos
   A=[0 0 0 0 0 1
      0 0 0 0 1 0
      0 0 0 2 0 0
      d^5 d^4 d^3 d^2 d 1
      5*d^4 4*d^3 3*d^2 2*d 1 0
      20*d^3 12*d^2 6*d 2 0 0];
   S=[DAT(i,2) DAT(i,3) DAT(i,4) DAT(i+1,2) DAT(i+1,3) DAT(i+1,4)]';
   P=A\S;
   for j=1:6,
      coef(i,j)=P(j);
   end
end

pp=mkpp(DAT(:,1),coef);
```

```
function deriv=pderiv(fun)
%se trata de hacer la derivada numericamente,de una función%
%periódica entre cero y 2*pi%
dim=size(fun);
if(dim(1)~=1),
   dim=dim(1);
else,
   dim=dim(2);
end

deriv(1)=(fun(2)-fun(dim-1))/(2*2*pi/(dim-1));
for i=2:(dim-1),
   deriv(i)=(fun(i+1)-fun(i-1))/(2*2*pi/(dim-1));
end
deriv(dim)=deriv(1);
```

```
function deriv=pderivtheta(fun)
%se trata de hacer la derivada numericamente,de una función%
%que no es periódica pero que si para psi0 correpontde theta0%
%,para psi0+2*pi,correspnde theta0-2*pi%
dim=size(fun);
if(dim(1)~=1),
   dim=dim(1);
else,
   dim=dim(2);
end
```

```
deriv(1)=(fun(2)-(fun(dim-1)+2*pi))/(2*2*pi/(dim-1));
for i=2:(dim-1),
   deriv(i)=(fun(i+1)-fun(i-1))/(2*2*pi/(dim-1));
end
deriv(dim)=deriv(1);

function z=ecs_tipos89(x,k1,k2,ac,ss,db2,l,sigma)
%Se trata de definir este sistema de ecuaciones, para en los tipos 8 y
9,poder calcular
%alpha y beta conociendo ss.
%alpha es x(1),beta x(2),y z(1) y z(2) representan las ecuaciones.
z(1)=-k1+db2*sin(x(1)-sigma)+l*sin(x(2)-sigma);
z(2)=-k2+ac+ss+db2*cos(x(1)-sigma)+l*cos(x(2)-sigma);



function
x=evaluamatriz(vect,vect1,vect2,w_pas,maxim,minim,PONDERA1,PONDERA2)


%hay que evaluar la aceleración que produce este movimiento. Se utiza la
raiz de
%la media del cuadrado de la aceleración en todo el intervalo.
%En caso de que pueda haber cúspides también se maximiza el máximo radio
de curvatura.
%En este caso también se minimiza la integral de la sobreaceleración,para
tratar de
%suavizar la curva de la aceleración.


%Reconstruimos la matriz e identificamos si estamos en un punto posible.
param1=reshape(vect1,length(vect1)/4,4);
param2=reshape(vect2,length(vect2)/4,4);
param=param1;      %para que queden grabados los valores constantes,
p=1;
x=0;
for j=1:4,
   for i=1:length(vect1)/4,
      if param1(i,j)~=param2(i,j),   %Distintos límites, es un parámetro
variable.
         param(i,j)=vect(p);
         p=p+1;                 %Avanzo un puesto en el vector de
variables.
         if param(i,j)<param1(i,j) | param(i,j)>param2(i,j),
            x=1e150;
         end
      end
   end
end

param(:,4)=param(:,4)*1000;  %Lo reescalo para el cálculo
if x==0;    %No se ha tocado el valor de x, estamos en un punto posible.
   if size(w_pas)==[1 1];
      w=w_pas;
   else,
      w_pp=w_pas;
      dw_pp=fnder(w_pp);
```

```
      ddw_pp=fnder(dw_pp);
   end,

   param_mod(:,1)=param(:,1);
   param_mod(:,2)=param(:,2);

   %La primera derivada:
   if size (w_pas)==[1 1],
      param_mod(:,3)=param(:,3)/w;
   else,       %Usamos el spline
      wpp_con=ppval(w_pp,param(:,1));     %wpp_concreto
     param_mod(:,3)=param(:,3)./wpp_con;
   end

   %La segunda derivada:
   if size(w_pas)==[1 1],
      param_mod(:,4)=param(:,4)/(w^2);
   else,
      dwpp_con=ppval(dw_pp,param(:,1));
      %ds/dt=wpp_con*ds/dpsi,lo uso a continuación.
      param_mod(:,4)=(1./wpp_con.^2).*(param(:,4)-
param_mod(:,3).*wpp_con.*dwpp_con);
   end

   s_pp=construyepp(param_mod);   %construyo la curva

   psi=0:2*pi/1000:2*pi;
   s=ppval(s_pp,psi);
   ds_pp=fnder(s_pp);
   dds_pp=fnder(ds_pp);
   ddds_pp=fnder(dds_pp);
   ds=ppval(ds_pp,psi);
   dds=ppval(dds_pp,psi);
   ddds=ppval(ddds_pp,psi);

   %Si se sale de los límites penalizo.Introduzco una tolerancia
   if (maxim+1e-10)<max(s) | (minim-1e-10)>min(s), x=1e152;end

   %El cálculo de x que se hace a continuación trata de suavizar la
curva.
   if size(w_pas)==[1 1],
      acel=dds*w.^2;
      sobreacel=ddds*w.^3;
   else,
      w_val=ppval(w_pp,psi);        %Los valores de w, dw, ddw en los
puntos psi.
      dw_val=ppval(dw_pp,psi);
      ddw_val=ppval(ddw_pp,psi);
      acel=dds.*(w_val).^2+ds.*w_val.*dw_val;

sobreacel=((dds.*w_val+ds.*dw_val).*dw_val+(ddds.*w_val+2*dds.*dw_val+ds.
*ddw_val).*w_val).*w_val;
   end,
   delta=(2*pi)/1000;  %el intervalo
   acel(length(acel))=0;    %el último lo hago valer cero para no
repetir.
```

```
   x1=.5*sqrt(sum(acel.^2)*delta/(2*pi))+.5*max(abs(acel));
   x2=sqrt(sum(sobreacel.^2)*delta/(2*pi));
   numpaponderar1=abs(x1)/(x2*PONDERA1);

   x=x1+numpaponderar1*x2;

   if PONDERA2,    %hay que minimizar el radio de curvatura mínima, para
que no haya cúspides.
      %El radio de curvatura es realmente ss+ddss, aproximadamente
ss=cte1+cte2*s.
      aprcur=s+dds;

      aprcurmed=median(aprcur)*ones(size(aprcur));
      aprcurmin=min(aprcur);

      valorlim=aprcurmin+.1*(aprcurmed-aprcurmin);  %También me fijo en
los cercanos al min.

      aprcurinf=aprcur.*(aprcur<valorlim);
      aprcurinfmed=sum(aprcurinf)/sum(aprcur<valorlim);

      xx1=-.95*aprcurmin;
      xx2=-.05*aprcurinfmed;
      numpaponderar2=abs(xx1+xx2)/(x*PONDERA2);
      x=xx1+xx2+numpaponderar2*x;
   end

end,



function [psi_max,psi_min]=extremospp(s_pp)

%se trata de buscar los extremos, máximos y mínimos,de una función
%polinómica a trozos.
ds_pp=fnder(s_pp);
[breaks,coef]=unmkpp(ds_pp);

s_max=ppval(s_pp,breaks(1));   %inicializo
s_min=s_max;
psi_max=breaks(1);
psi_min=psi_max;

for i=1:(length(breaks)-1)     %Para todos los intervalos.
   pol=coef(i,:);
   raices=roots(pol);
   for j=1:length(raices),
      r=real(raices(j)+breaks(i));    %Lo traslado y me quedo con la
parte real
      if (r>=(breaks(i)-1.e-4)) & (r<=(breaks(i+1)+1.e-4)),
         aux=ppval(s_pp,r);
         if aux > s_max,
            s_max=aux;
            psi_max=r;
         end
         if aux < s_min,
```

```
            s_min=aux;
            psi_min=r;
         end
      end
   end
end
%Como antes se amplió el intervalo entre nodos,para captar extremos en
%el límite,si hay errores numéricos,ahora forzamos a que psi_max y
%psi_min estén entre 0 y 2*pi.
if psi_max<0, psi_max=0;end
if psi_min<0, psi_min=0;end
if psi_max>2*pi, psi_max=2*pi;end
if psi_min>2*pi, psi_min=2*pi;end



function datosdsal=general(datosnenes)

%Los nombres de las variables están en minúscula,pero son las mismas.
%Esta función pedirá los datos por pantalla, y dará los resultados.
%Llamará a las otras funciones específicas de cada tipo de leva.
%Si se quieren ver los resultados según se calculan, poner un uno en la
variable 'ver'
ver=0;
N=5000;  %intervalos en los que divido [0,2*pi].

if ver,
   datosnenes,pause,
end,

%Organizo lo que se reciven por pantanlla:
tipo=datosnenes{1}(1);      %el tipo de leva
dmax=datosnenes{1}(2);  dmin=datosnenes{1}(4);
igmax=datosnenes{1}(3);   igmin=datosnenes{1}(5);

param=datosnenes{2};

param(:,1)=param(:,1)*(2*pi/360);    %Lo paso a radianes.


prop=ones(size(param));          %Tiene un 1 si hay un dato en ese lugar,
y un 0 si no.
for i=1:length(param(:,1)),
   for j=1:length(param(1,:)),
      if isnan(param(i,j)),
         prop(i,j)=0;
      end
   end
end

if param(1,1)~=0 | param(length(param(:,1)),1)~=2*pi,
   error('el primer dato debe ser en PSI=0 y el último en PSI=360°'),
end

for j=2:length(param(1,:)),
   if isnan(param(i,j))
      if ~isnan(param(length(param(:,1)),j)),
```

```
         error('En  psi 0º y en 360º los datos deben ser los mismos'),
      end,
   else
      if param(1,j)~=param(length(param(:,1)),j),
         error('En  psi 0º y en 360º los datos deben ser los mismos'),
      end
   end
end


%Ahora recojo la velocidad angular (RPM):
RPM=datosnenes{3};

if size(RPM)~=[1 1],
   RPM(:,1)=RPM(:,1)*(2*pi/360);           %Lo paso a radianes.
end

if ver,
   param,
   RPM,
   prop,
   pause,
end,

%Ahora,según el tipo,llenamos los vectores geometría, datosdin, y
requisitos.
switch tipo,
case 1,
   requisitos=datosnenes{4};
case 2,
   geometria=datosnenes{4}(1:2); requisitos=datosnenes{4}(3:4);
   e=geometria(1); l=geometria(2);
case 3,
   geometria=datosnenes{4}(1:2); requisitos=datosnenes{4}(3:4);
   e=geometria(1); rrod=geometria(2);
case 4,
   geometria=datosnenes{4}(1:3); requisitos=datosnenes{4}(4:5);
   l=geometria(1); e=geometria(2); rrod=geometria(3);
case 5,
   geometria=datosnenes{4}(1); datosdin=datosnenes{4}(2:10);
   requisitos=datosnenes{4}(11:13);
   dv=geometria(1);
case 6,
   geometria=datosnenes{4}(1:7); datosdin=datosnenes{4}(8:21);
   requisitos=datosnenes{4}(22:24);
   dv=geometria(1); db=geometria(2); e=geometria(3); l=geometria(4);
omega_b=geometria(5);
   omega_l=geometria(6); rrod=geometria(7);
case 7,
   geometria=datosnenes{4}(1:6); datosdin=datosnenes{4}(7:17);
   requisitos=datosnenes{4}(18:20);
   dv=geometria(1); db=geometria(2); e=geometria(3); l=geometria(4);
omega_b=geometria(5);
   omega_l=geometria(6);
case 8,
   geometria=datosnenes{4}(1:10); datosdin=datosnenes{4}(11:29);
   requisitos=datosnenes{4}(30:32);
```

```
   alpha_mayor_beta=1;
   dv=geometria(1); db1=geometria(2); db2=geometria(3); l=geometria(4);
omega_b=geometria(5);
   sigma=geometria(6); k1=geometria(7); k2=geometria(8); ac=geometria(9);
e=geometria(10);
case 9,
   geometria=datosnenes{4}(1:11); datosdin=datosnenes{4}(12:34);
   requisitos=datosnenes{4}(35:37);
   alpha_mayor_beta=1;
   dv=geometria(1); db1=geometria(2); db2=geometria(3); l=geometria(4);
omega_b=geometria(5);
   sigma=geometria(6); k1=geometria(7); k2=geometria(8); ac=geometria(9);
e=geometria(10);
   rrod=geometria(11);
end

if((dmin<=0)|(dmax<=0)),error('los diametros deben ser >0');end
if(dmax<=dmin),error('dmax debe se > que dmin');end


%Según dmax y dmin,calculamos ssmax, ssmin, smax, y smin.ss se refiere a
l movimiento
%del seguidor,y s, al movimiento de la válvula.
 switch tipo,
 case {1 5 8},  %segidor plano de traslación.
    ssmax=dmax/2; ssmin=dmin/2;
    switch tipo,
    case 1,
       smax=ssmax; smin=ssmin;
    case 5,
       smax=ssmax+dv; smin=ssmin+dv;
    case 8,
       %Cálculo ss en el punto muerto,y alpha (alpha=beta):
       ss_pm=-sqrt((db2+l)^2-k1^2)+k2-ac;
       alpha_pm=asin(k1/(db2+l))+sigma;
       if ssmax<ss_pm |ssmin<ss_pm,
          error('La geometria no es válida:el seguidor no alcanza el
punto muerto'),
       end
       %Se fuerza a que se esté a un lado o a otro del punto muerto.Si se
desea analizar
       %un caso en el que pase por el punto muerto,basta con no usar
'limita'.
       if alpha_mayor_beta==1,       %Para que parta con alpha mayor que
beta la solución
          x0=[pi/2,0];                   %de las ecs,y no dé la posición
equivocada.
          %x=[alpha,beta];
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmax,db2,l,sigma);
          smax=db1*sin(pi/2-(omega_b-x(1)))+dv;

          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmin,db2,l,sigma);
          smin=db1*sin(pi/2-(omega_b-x(1)))+dv;

       else, %No es usual,lo normal es que alpha sea mayor que beta.
```

```
          x0=[0,pi/6];
          %Ahora,smin se alcanza en ssmax,y smax en ssmin.
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmax,db2,l,sigma);
          smin=db1*sin(pi/2-(omega_b-x(1)))+dv;
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmin,db2,l,sigma);
          smax=db1*sin(pi/2-(omega_b-x(1)))+dv;
       end
    end
 case {2 7},
    %hay que forzar que el argumento del arcoseno esté entre -1 y 1.
    if (dmax>2*(l+e)) | (dmin<2*(-l+e)),
       error('dmax es demasiado grande,o dmin demasiado pequeño'),
    end
    %Llamo s,aunque es fi.
    %El arcoseno davalores entre -pi/2 y pi/2.Consideramos este intervalo
suficiente,pues
    %para angulos más grandes se obtienen levas más grandes.De todas
formas se pueden ana-
    %lizar con leva2,imponiendolos.
    ssmax=asin((1/l)*(dmax/2-e));
    ssmin=asin((1/l)*(dmin/2-e));
    switch tipo,
    case 2,
       smax=ssmax; smin=ssmin;
    case 7,
       smax=db*sin(pi/2-(omega_b-(ssmax+omega_l)))+dv;
       smin=db*sin(pi/2-(omega_b-(ssmin+omega_l)))+dv;
    end
 case {3 9},
    if ((dmin/2+rrod)^2-e^2) < 0,
       error('esta geometria no da siempre s>0');
    end
    ssmax=sqrt((dmax/2+rrod)^2-e^2);
    ssmin=sqrt((dmin/2+rrod)^2-e^2);
    switch tipo,
    case 3,
       smax=ssmax; smin=ssmin;
    case 9,
       %Como el 8
       %Cálculo ss en el punto muerto,y alpha (alpha=beta):
       ss_pm=-sqrt((db2+l)^2-k1^2)+k2-ac;
       alpha_pm=asin(k1/(db2+l))+sigma;
       if ssmax<ss_pm |ssmin<ss_pm,
          error('La geometria no es válida:el seguidor no alcanza el
punto muerto'),
       end
       %Se fuerza a que se esté a un lado o a otro del punto muerto.Si se
desea analizar
       %un caso en el que pase por el punto muerto,basta con no usar
'limita'.
       if alpha_mayor_beta==1,       %Para que parta con alpha mayor que
beta la solución
          x0=[pi/2,0]; %de las ecs,y no dé la posición equivocada.
          %x=[alpha,beta];
```

```
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmax,db2,l,sigma);
          smax=db1*sin(pi/2-(omega_b-x(1)))+dv;
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmin,db2,l,sigma);
          smin=db1*sin(pi/2-(omega_b-x(1)))+dv;
       else, %No es usual,lo normal es que alpha sea mayor que beta.
          x0=[0,pi/6];
          %Ahora,smin se alcanza en ssmax,y smax en ssmin.
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmax,db2,l,sigma);
          smin=db1*sin(pi/2-(omega_b-x(1)))+dv;
          x=fsolve('ecs_tipos89',x0,[0,1.e-
14],'',k1,k2,ac,ssmin,db2,l,sigma);
          smax=db1*sin(pi/2-(omega_b-x(1)))+dv;
       end
    end
 case {4 6},
    if ((e^2+l^2-(dmax/2+rrod)^2)/(2*e*l))<(-1) | ((e^2+l^2-
(dmin/2+rrod)^2)/(2*e*l))>1,
       error('Esta geometria no da valores de fi posibles'),
    end
    %El arcocoseno da valores entre cero y pi.No hay que ampliar el
rango,pues si pasamos
    %por cero,o por pi,el angulo de presión vale +-pi/2,lo que no es
admisible.Ángulos entre
    %pi y 2*pi son poco usuales, y dan levas poco adecuadas.Puede
comprobarse usando leva4.
    ssmax=acos((e^2+l^2-(dmax/2+rrod)^2)/(2*e*l));
    ssmin=acos((e^2+l^2-(dmin/2+rrod)^2)/(2*e*l));
    switch tipo,
    case 4,
       smax=ssmax; smin=ssmin;
    case 6,
       smax=db*sin(pi/2-(omega_b-(ssmax+omega_l)))+dv;
       smin=db*sin(pi/2-(omega_b-(ssmin+omega_l)))+dv;
    end
 end

 for i=1:length(param(:,2)),     %Conocidos smax y smin,los pongo en
donde deben.
    if param(i,2)==Inf,
       param(i,2)=smax;
    elseif param(i,2)==-Inf
       param(i,2)=smin;
    else,
       param(i,2)=param(i,2)+smin;   %Pues se dan posciones relativas por
pantalla.
    end

 end


 if ver,
    ssmax,ssmin,
    if tipo==8 | tipo==9, ss_pm, alpha_pm, end
    smax,smin,pause,
```

```
 end,

%La velocidad angular:
if size(RPM)==[1 1],
   w=RPM*2*pi/60;wmed=w;   %si es un escalar.
else
   w=[RPM(:,1),RPM(:,2)*(2*pi/60)]; %en radianes/segundo.
   wmed=median(w(:,2));       %Valor medio de la velocidad angular.
   m=size(w);
   m=m(1);
   if abs(w(1,2)-w(m,2))<1.e-12,     %si son iguales,un spline periódico
      w_pp=spline_per(w(:,1),w(:,2));
   else,
      w_pp=spline(w(:,1),w(:,2));    %si no,un spline normal
   end
   dw_pp=fnder(w_pp);
end


%construyo la solución incial,respetando los datos
psi=param(:,1); s=param(:,2); f=prop(:,2);
j=1;s_in=[];      %por si no hay valores,para que esté definida
n=length(psi);
for i=1:n,
   if f(i)>0,
      psi_in(j)=psi(i);s_in(j)=s(i);j=j+1;
   end
end
%si no hay datos:error.
if length(s_in)==0,error('debe dar valores a s o fi'),end

if length(s_in)~=1,               %más de un punto
   if f(1)>0,                    %en 0 y 2*pi está dado
       s_pp=spline_per(psi_in,s_in);
       s=ppval(s_pp,psi);
    else,                         %ni en 0 ni en 2*pi está dado
       s=spline(psi_in,s_in,psi);     %interpolo en 0 y 2*pi.
       s(1)=(s(1)+s(n))/2;            %Pues deben ser iguales.
       s_in=[s(1),s_in,s(1)];
       psi_in=[0,psi_in,2*pi];
       s_pp=spline_per(psi_in,s_in);
       s=ppval(s_pp,psi);
   end
else,
   s=ones(size(s)).*s_in(1);       %si solo hay un punto,todos igual
   s_pp=spline_per(psi,s);
end
param(:,2)=s;  %Meto los valores en la matriz.
param_mod(:,[1 2])=param(:,[1,2]);  %en param_mod se pondran derivadas
respecto de s,no de t


ds_pp=fnder(s_pp);       %1ª derivada de s respecto de psi,en ppio
dds_pp=fnder(ds_pp);     %1ª derivada de s respecto de psi,en ppio
%En ppio,pondremos las derivadas del spline periódico,luego,cuando haya
datos de
%velocidad y/o aceleración,modificaremos los datos en cuestión.
```

```
for i=1:length(psi),
   param_mod(i,3)=ppval(ds_pp,param_mod(i,1));
   param_mod(i,4)=ppval(dds_pp,param_mod(i,1));
end

%Modificamos la pendiente s',atendiendo a la velocidad cuando sea dato:
for i=1:length(psi),
   if prop(i,3)>0,  %Es un dato,o se quiere que sea inicial
      if size (w)==[1 1],
         param_mod(i,3)=param(i,3)/w;
      else,       %Usamos el spline
         wpp_con=ppval(w_pp,param_mod(i,1));     %wpp_concreto
         param_mod(i,3)=param(i,3)/wpp_con;
      end
   end
end

%Modificamos s'',atendiendo a la aceleración,cuando sea dato.
for i=1:length(psi),
   if prop(i,4)>0,       %Es un dato,o se quiere que sea inicial.
      if size(w)==[1 1],
         param_mod(i,4)=param(i,4)/(w^2);
      else,
         wpp_con=ppval(w_pp,param_mod(i,1));     %wpp_concreto.
         dwpp_con=ppval(dw_pp,param_mod(i,1));
         %ds/dt=wpp_con*ds/dpsi,lo uso a continuación.
         param_mod(i,4)=(1/wpp_con^2)*(param(i,4)-
param_mod(i,3)*wpp_con*dwpp_con);
      end
   end
end

if ver,
   param_mod,pause,
end,

%Es posible que en 0 y en 2*pi,las derivadas de s con respecto del tiempo
no sean iguales,
%si la velocidad angular no es una función periódica.En estos casos,las
velocidades del
%seguidor en los extremos deben ser tales que s s' y s''
coincidan.Pondremos entonces siempre
%en 0 y en 2*pi,el valor medio:
if max(abs(param_mod(n,[2 3 4])-param_mod(1,[2 3 4])))>1.e-6,
   disp('En 0 y 2pi no son iguales s y sus derivadas:en 0:'),
   disp(param_mod(1,:)),disp('en 2*pi:'),
   disp(param_mod(n,:)),
   disp('errores en la velocidad angular provocan diferencias en la
primera derivada de s,'),
   disp('errores en la derivada de la velocidad angular (no controlada
por el usuario),'),
   disp('provocan diferencias en la segunda derivada de s'),
   disp('El programa tomará el valor medio y continuará'),pause,
end
param_mod_auxil=(param_mod(n,[2 3 4])+param_mod(1,[2 3 4]))/2;
param_mod(n,[2 3 4])=param_mod_auxil; param_mod(1,[2 3
4])=param_mod_auxil;
```

```
%construimos el polinomio a trozos:
s_pp=construyepp(param_mod);      %Movimiento de la válvula.
%dibujo
psi=0:2*pi/N:2*pi;
datosdsal{1}=psi*(360/(2*pi));
datosdsal{2}{1}=param_mod(:,1)*(360/(2*pi));
datosdsal{2}{2}=param_mod(:,2);
datosdsal{2}{3}=ppval(s_pp,psi);
datosdsal{2}{4}=[smax,smin];
if ver,

plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
   title('candidata a ley de movimiento  vs psi'),pause,
end

%Llamo a una función que modifica la curva para que cumpla las
%restricciones.
[s_pp,param_mod]=limita(s_pp,param,prop,param_mod,igmax,igmin,smax,smin,n
);
%Si los datos de entrada no sirven, directamente se va a la salida
%En este caso solo hay datosdsal {1} y {2}, y hay que cuidarse de no
direcionar luego
%componentes que no existen,por eso empiezan con un if muchos call back
de intsal.
if isnan(s_pp),return,end,

s=ppval(s_pp,psi);
datosdsal{2}{5}=s;

if ver,

plot(psi,s,param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',psi,smin,'r'),
   title('Nueva ley de movimiento  vs psi'),pause,
end,

%Llamamos ahora a las funciones levai (i=1,2...),para que construya la
leva e
%investigue su validez.
if size(w)==[1 1],     %Es un escalar si no una serie de puntos.
   w_paso=w;
else,                  %Le pasamos el spline.
   w_paso=w_pp;
end

switch tipo,
case 1,
   datosdsal_aux=leva1(s,w_paso,dmax,dmin,ver);
   for i=[3 4 5 6 8],
      datosdsal{i}=datosdsal_aux{i};
   end
case 2,
   datosdsal_aux=leva2(s,w_paso,geometria,dmax,dmin,ver);
   for i=[3 4 5 6 8],
      datosdsal{i}=datosdsal_aux{i};
```

```
   end
case 3,
   datosdsal_aux=leva3(s,w_paso,geometria,dmax,dmin,ver);
   for i=[3 4 5 6 8],
      datosdsal{i}=datosdsal_aux{i};
   end
case 4,
   datosdsal_aux=leva4(s,w_paso,geometria,dmax,dmin,ver);
   for i=[3 4 5 6 8],
      datosdsal{i}=datosdsal_aux{i};
   end
case 5,
   datosdsal_aux=leva5(s,w_paso,geometria,dmax,dmin,datosdin,ver);
   for i=[3 4 5 6 7 8 9],
      datosdsal{i}=datosdsal_aux{i};
   end
case 6,
   datosdsal_aux=leva6(s,w_paso,geometria,dmax,dmin,datosdin,ver);
   for i=[3 4 5 6 7 8 9],
      datosdsal{i}=datosdsal_aux{i};
   end
case 7,
   datosdsal_aux=leva7(s,w_paso,geometria,dmax,dmin,datosdin,ver);
   for i=[3 4 5 6 7 8 9],
      datosdsal{i}=datosdsal_aux{i};
   end
case 8,
   datosdsal_aux=leva8(s,w_paso,geometria,dmax,dmin,datosdin,ver);
   for i=[3 4 5 6 7 8 9],
      datosdsal{i}=datosdsal_aux{i};
   end
case 9,
   datosdsal_aux=leva9(s,w_paso,geometria,dmax,dmin,datosdin,ver);
   for i=[3 4 5 6 7 8 9],
      datosdsal{i}=datosdsal_aux{i};
   end
end

datosdsal{10}=requisitos;     %Para luego dibujar los límites en las
gráficas.
if tipo==3 | tipo==4 | tipo==6 | tipo==9,
   datosdsal{10}(1)=datosdsal{10}(1)*(360/(2*pi));  %Paso a °,para
dibujar.
end



function
[datosdsal_aux,ro,theta,excent,vel,acel]=leva1(s,w_paso,dmax,dmin,ver)
%Se analiza la leva tipo 1:con seguidor plano de traslación.
%s es el desplazamiento de la leva%
%N es el numero de veces en q se parte el intervalo 2pi%
%para aproximar%
%psi es el angulo de giro del eje fijo a la leva%
%ro,theta,son las coordenadas polares del punto de contacto%
%partiendo theta del eje fijo a la leva%
N=5000;
```

```
psi=0:(2*pi/N):2*pi;

if ver,
   plot(psi,s),title('s'),pause,
end,

datosdsal_aux{6}{1}=s;
ds=deriv(s);
dds=deriv(ds);

ro=sqrt(s.^2+ds.^2);
datosdsal_aux{3}{1}=ro;
if ver,
   plot(psi,ro),title('ro vs psi'),grid,
   pause,
end,

%aux corresponde con el valor de theta+psi,que está entre 0 y pi,para
valores de s>0.
%No se admitirán valores de s<=0,pues no dejaría espacion físico para el
eje.
%Cogemos los valores de la acotangente entre 0 y pi.
if min(s)<=0,error('no se adminten valores <=0'),end
for i=1:N+1,
   if ds(i)==0,
      aux(i)=pi/2;
   else,
      if ds(i)>0,
         aux(i)=atan(s(i)/ds(i));
      else,
         aux(i)=atan(s(i)/ds(i))+pi;
      end
   end
end
if ver,
  plot(psi,aux),title('aux vs psi');grid,
  pause,
end
theta=aux-psi;
datosdsal_aux{3}{2}=theta;
if ver,
   plot(psi,theta),title('theta vs psi'),grid,
   pause,

   polar(theta,ro),title('LEVA                LEVA'),hold on,    %Los
límites.
   polar(theta,(dmin/2)*diag(ones(N+1))','r'),
   polar(theta(1),ro(1),'*'),        %Señala la tangencia del seguidor en
psi=0.
   polar(theta,(dmax/2)*diag(ones(N+1))','r'),hold off,  %perfil de la
leva%
   pause,
end

%calculo el area%
dtheta=derivtheta(theta);
areaaux=.5*ro.^2.*dtheta*(2*pi/N);
```

```
%como el numero de intervalos es menor que el de nodos en uno,%
%anulo el ùltimo tèrmino,pues seria repetitivo%
%hay que cambiar el signo,pues dtheta es negativo %
areaaux(N+1)=0;
if ver,
   area=-sum(areaaux),
else,
   area=-sum(areaaux);
end
datosdsal_aux{8}=area;


%la excentricidad se define como la proyeccion de la  distancia%
%entre el eje de giro de la leva,y el punto de contacto,%
%según la dirección de la cara plana del seguidor.%
%es un parametro de interés pues representa el brazo de la fuerza%
%que se ejerce en el contacto,con lo que el momento es mayor,%
%además de necesitar un mayor seguidor,por tanto con más inercia%
%.Por esto se representa,y se calcula%
%su máximo.%
excent=ds;
if ver,
   plot(psi,excent),grid,title('excentricidad (m)vs psi');pause,
end
datosdsal_aux{4}=excent;

%el radio de curvatura es otro parametro importante,%
%pues su valor muestra si la trayectoria será no realizable,%
%o si las tensiones en el punto de contacto seran inadmisibles%
rcur=s+dds;     %segùn el libro%
datosdsal_aux{5}=rcur;
if ver,
   plot(psi,rcur),grid,title('rcur(m) vs psi,según libro'),
   pause,

   %lo calculo con la fórmula general para verificar la del libro%
   dro=deriv(ro);
   drorespttheta=dro./dtheta;
   ddrorespttheta=deriv(drorespttheta);
   ddrorespttheta2=ddrorespttheta./dtheta;
   num=(ro.^2+drorespttheta.^2).^(3/2);
   den=ro.^2+2*drorespttheta.^2-ro.*ddrorespttheta2;
   rcurg=num./den;
   plot(psi,rcurg),title('rcur(m) vs psi,ségun fórmula general'),grid,
   pause,
end
%si hay puntos en los que la curvatura cambia de signo,se%
%considerará la leva no válida,pues el seguidor invadiría a %
%la leva,o contactaria en dos puntos.Debe ser siempre convexa la leva.%


%cálculo la velocidad,la aceleración,y la sobreaceleración(cuya
%variación suave es un signo de calidad de la leva)
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
```

```
end

vel=ds.*w;
datosdsal_aux{6}{2}=vel;
dvel=deriv(vel);
acel=dvel.*w;
datosdsal_aux{6}{3}=acel;
dacel=deriv(acel);
sobreacel=dacel.*w;
datosdsal_aux{6}{4}=sobreacel;

if ver
   %Dibujo%
   plot(psi,s),grid,title('movimiento del seguidor (m) vs psi'),
   pause,
   plot(psi,vel),grid,title('velocidad del seguidor(m/seg) vs psi'),
   pause,
   plot(psi,acel),grid,title('aceleración del seguidor(m/seg^2) vs psi'),
   pause,
   plot(psi,sobreacel),grid,title('sobreaceleración del seguidor(m/seg^3)
vs psi'),
   pause,
end
```

```
function
[datosdsal_aux,theta,ro,dfi,vel,acel]=leva2(fi,w_paso,geom,dmax,dmin,ver)

%Se trata de sintetizar y estudiar una leva de tipo 2 (de seguidor plano
oscilante),
%,para una ley dada de movimiento: fi(psi).

N=5000;         %Número de intervalos en que divido,para aproximar.
e=geom(1);      %Ver dibujo de leva tipo 2
l=geom(2);

datosdsal_aux{6}{1}=fi;

psi=0:2*pi/N:2*pi;
dfi=deriv(fi);        %La primera derivada.
ddfi=deriv(dfi);      %La segunda derivada.

%Cálculo el radio en polares ro.
ro=sqrt((l*dfi.*cos(fi)).^2+((1+dfi).*(l*sin(fi)+e)).^2)./(1+dfi);
datosdsal_aux{3}{1}=ro;
if ver,
   plot(psi,ro),title('ro vs psi'),grid,pause,
end

%Para calcular el angulo theta de las coordenadas polares,necesito
%un ángulo auxiliar:aux,que es igual a theta+psi+fi.
%como la funcion atan da un valor entre -pi/2 y pi/2,hay
```

```
%que buscar las ramas de la arcotangente apropiadas para que
%thetha,como función de psi,sea continua

aux=ones(size(psi));
i=1;                  %el primer valor%
if(l*dfi(1)*cos(fi(1)))<1.e-10,   %Considero que es 0,y la atan:+-pi/2
   aux(1)=pi/2;
else
   aux(1)=atan(((1+dfi(1))*(l*sin(fi(1))+e))/(l*dfi(1)*cos(fi(1))));
end

for i=2:(N+1),     %llenamos el resto del vector%
   if(l*dfi(i)*cos(fi(i))==0), aux(i)=pi/2;
   else,
aux(i)=atan(((1+dfi(i))*(l*sin(fi(i))+e))/(l*dfi(i)*cos(fi(i))));
   end
   if(abs(aux(i)-aux(i-1))>(pi/4))          %si hay un salto%
      j=1;
      while((abs(aux(i)+j*pi-aux(i-1))>(pi/4))...
            &(abs(aux(i)-j*pi-aux(i-1))>(pi/4))) %busco el valor%
            j=j+1;                                  %apropiado para%
      end                                           %que haya%
      if(abs(aux(i)+j*pi-aux(i-1))<(pi/4)),       %continuidad%
         aux(i)=aux(i)+j*pi;
      else,
         aux(i)=aux(i)-j*pi;
      end
   end
end
if ver,
   plot(psi,aux),title('aux vs psi'),grid,pause,
end

theta=aux-psi-fi;    %ángulo en polares.
datosdsal_aux{3}{2}=theta;
if ver,
   plot(psi,theta),title('theta vs psi'),grid,pause,

   %dibujo la leva.
   polar(theta,ro),title('LEVA                    LEVA'),hold on,   %Los
límites.
   polar(theta,(dmin/2)*diag(ones(N+1))','r'),
   polar(theta(1),ro(1),'*'),                  %Señala la tangencia del
seguidor con psi=0.
   polar(theta,(dmax/2)*diag(ones(N+1))','r'),,grid,hold off,
   pause,
end

%calculo el area%
dtheta=derivtheta(theta);
areaaux=.5*ro.^2.*dtheta*(2*pi/N);
%como el numero de intervalos es menor que el de nodos en uno,%
%anulo el ùltimo tèrmino,pues seria repetitivo%
%hay que cambiar el signo,pues dtheta es negativo %
areaaux(N+1)=0;
if ver,
   area=-sum(areaaux),
```

```
else,
   area=-sum(areaaux);
end
datosdsal_aux{8}=area;

%la excentricidad se define como la distancia entre el punto de contacto,
%y la proyección del centro del eje de giro de la leva sobre el seguidor.
%Es un parámetro de interés pues representa el brazo de la fuerza%
%que se ejerce en el contacto,con lo que el momento es mayor,%
%además de necesitar un mayor seguidor,por tanto con más inercia%
%.Por esto se representa,y se calcula%
%su máximo.%
excent=(l*dfi.*cos(fi))./(1+dfi);
datosdsal_aux{4}=excent;
if ver,
   plot(psi,excent),grid,title('excentricidad vs psi');pause,
end,

%el radio de curvatura es otro parametro importante,%
%pues su valor muestra si la trayectoria será no realizable,%
%o si las tensiones en el punto de contacto seran inadmisibles%
%hay que multiplicar por l,pues el libro trabaja valores adimensionales.
%La fórmula del libro no funciona bien,dá siempre radios de curvatura
positivos,aunque
%haya puntos de retroceso y la curva tenga después radio de curvatura
negativo.Por eso
%uso la forma general.
if ver,

rcur=l*abs(ddfi.*cos(fi)+(1+dfi).*((1+2*dfi).*sin(fi)+(e/l)*(1+dfi).^2)).
/(1+dfi).^3;
   plot(psi,rcur),grid,title('rcur vs psi,según libro'),
   pause,
end,
%lo calculo con la fórmula general para verificar la del libro%
dro=deriv(ro);
drorespttheta=dro./dtheta;
ddrorespttheta=deriv(drorespttheta);
ddrorespttheta2=ddrorespttheta./dtheta;
num=(ro.^2+drorespttheta.^2).^(3/2);
den=ro.^2+2*drorespttheta.^2-ro.*ddrorespttheta2;
rcurg=num./den;
if ver,
   plot(psi,rcurg),grid,title('rcur vs psi,ségun fórmula general'),
   pause,
end
%Cojo el calculado con la fórmula general, por que la otra fórmula no
coincide cuando el
%radio es negativo.
datosdsal_aux{5}=rcurg;

%si hay puntos en los que la curvatura cambia de signo,se%
%considerará la leva no válida,pues el seguidor invadiría a %
%la leva,o contactaria en dos puntos.Debe ser siempre convexa la leva.%

%cálculo la velocidad,la aceleración,y la sobreaceleración(cuya
%variación suave es un signo de calidad de la leva),angulares
```

```
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(fi));
else,
   w=ppval(w_paso,psi);
end

vel=dfi.*w;
datosdsal_aux{6}{2}=vel;
dvel=deriv(vel);
acel=dvel.*w;
datosdsal_aux{6}{3}=acel;
dacel=deriv(acel);
sobreacel=dacel.*w;
datosdsal_aux{6}{4}=sobreacel;

if ver,
   %Dibujo%
   plot(psi,fi),grid,title('angulo del seguidor vs psi'),
   pause,
   plot(psi,vel),grid,title('velocidad angular del seguidor (1/seg) vs
psi'),
   pause,
   plot(psi,acel),grid,title('aceleración angular del seguidor (1/seg^2)
vs psi'),
   pause,
   plot(psi,sobreacel),grid,title('sobreaceleración angular del seguidror
(1/seg^3) vs psi'),
   pause,
end,
```

```
function
[datosdsal_aux,theta,ro,theta_r,ro_r,vel,acel]=leva3(s,w_paso,geometria,d
max,dmin,ver)

N=5000;    %Número de intervalos en los que divido [0,2*pi].
e=geometria(1); rrod=geometria(2);

psi=0:2*pi/N:2*pi;

datosdsal_aux{6}{1}=s;
ds=deriv(s);
dds=deriv(ds);


%Calculo el radio en polares:
ro=sqrt(s.^2+e^2);
datosdsal_aux{3}{1}=ro;
if ver,
   plot(psi,ro),grid,title('ro vs psi'),pause,
end

%Calculo el ángulo en polares.
%Si e=0,el ángulo theta+psi=pi/2,constante.
```

```
if e==0,
   aux=(pi/2)*diag(ones(N+1))';
else,
   if e>0,                      %Pongo el primer valor.
      aux(1)=atan(s(1)/e);
   else,
      aux(1)=atan(s(1)/e)+pi;
   end
   for i=2:(N+1),     %llenamos el resto del vector%
      aux(i)=atan(s(i)/e);
      if(abs(aux(i)-aux(i-1))>(pi/4))          %si hay un salto%
         j=1;
         while((abs(aux(i)+j*pi-aux(i-1))>(pi/4))...
               &(abs(aux(i)-j*pi-aux(i-1))>(pi/4))) %busco el valor%
               j=j+1;                                  %apropiado para%
         end                                           %que haya%
         if(abs(aux(i)+j*pi-aux(i-1))<(pi/4)),       %continuidad%
         aux(i)=aux(i)+j*pi;
         else,
         aux(i)=aux(i)-j*pi;
         end
      end
   end
end
if ver,
   plot(psi,aux),grid,title('aux vs psi'),
   pause,
end,

theta=aux-psi;
datosdsal_aux{3}{2}=theta;
if ver,
   plot(psi,theta),grid,title('theta vs psi'),
   pause,

   %Dibujo la leva (primitiva).
   polar(theta,ro),title('LEVA PRIMITIVA                    LEVA
PRIMITIVA'),hold on,
   polar(theta(1),ro(1),'*'),       %El punto de contacto en psi=0.
   hold off,pause,
end,

%Veamos ahora el ángulo de presión alpha:
%Según la fórmula que proporciona el libro:
for i=1:N+1,
   if s(i)<=0,
      error('hay ángulos de presión de +-pi/2.s no puede ser <=0'),
   else,
      alpha(i)=atan((ds(i)-e)/s(i));
   end
end
datosdsal_aux{4}=alpha*(360/(2*pi));  %Para dibujar luego el ángulo de
presión en °.
if ver,
   plot(psi,alpha),grid,title('angulo de presión(rad) vs psi,según
libro.'),pause,
end
```

```
%Calculemos ahora de una forma más general:
dtheta=derivtheta(theta);
dro=deriv(ro);
drorespttheta=dro./dtheta;
beta=atan(-drorespttheta./ro);
if ver,
   alphag=aux+(beta+pi/2)-pi;
   plot(psi,alphag),grid,title('angulo de presión(rad),expresión
general.'),pause,
end

%Veamos la curvatura:
%Según el libro:
k=(s.*(s-dds)+(ds-e).*(2*ds-e))./(s.^2+(ds-e).^2).^(3/2);
if ver,
   plot(psi,k),grid,title('curvatura primitiva vs psi,segùn
libro'),pause,
   %Según la fórmula general:
   ddrorespttheta=deriv(drorespttheta);
   ddrorespttheta2=ddrorespttheta./dtheta;
   den=(ro.^2+drorespttheta.^2).^(3/2);
   num=ro.^2+2*drorespttheta.^2-ro.*ddrorespttheta2;
   kg=num./den;
   plot(psi,kg),grid,title('curvatura primitiva vs psi,expresión
general'),pause
end
%si el radio del rodillo es mayor que el radio de curvatura mínimo del
perfil primitivo,
%se produce una cúspide (undercutting).
%La siguiente razon (rrod/rcur) nos dice lo cerca que estamos de que se
produzcan cúspides,
%de que la leva no sea válida.
datosdsal_aux{5}=k*rrod;

%Calculo el perfil real.
[ro_r,theta_r]=prim_a_r(ro,theta,beta,rrod);
datosdsal_aux{3}{3}=ro_r;
datosdsal_aux{3}{4}=theta_r;
if ver,
   plot(psi,ro_r),title('ro_r'),grid,pause,
   plot(psi,theta_r),title('theta_r'),grid,pause,
   %Dibujo:
   polar(theta,ro),hold on,
   title('LEVA PRIMITIVA Y REAL                    LEVA PRIMITIVA
YREAL'),
   polar(theta_r,ro_r),
   polar(theta_r,(dmax/2)*diag(ones(N+1))','r'),
   polar(theta_r(1),ro_r(1),'*'),       %El punto de contacto en psi=0.
   polar(theta_r,(dmin/2)*diag(ones(N+1))','r'),hold off,pause,
end

%Calculo el area de la leva real:
dtheta_r=derivtheta(theta_r);
areaaux=.5*ro_r.^2.*dtheta_r*(2*pi/N);
%como el numero de intervalos es menor que el de nodos en uno,%
%anulo el ùltimo tèrmino,pues seria repetitivo%
%hay que cambiar el signo,pues dtheta es negativo %
```

```
areaaux(N+1)=0;
if ver,
   area=-sum(areaaux),
else,
   area=-sum(areaaux);
end
datosdsal_aux{8}=area;

%cálculo la velocidad,la aceleración,y la sobreaceleración(cuya
%variación suave es un signo de calidad de la leva)
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end

vel=ds.*w;
datosdsal_aux{6}{2}=vel;
dvel=deriv(vel);
acel=dvel.*w;
datosdsal_aux{6}{3}=acel;
dacel=deriv(acel);
sobreacel=dacel.*w;
datosdsal_aux{6}{4}=sobreacel;

if ver,
   %Dibujo%
   plot(psi,s),grid,title('movimiento del seguidor vs psi'),
   pause,
   plot(psi,vel),grid,title('velocidad del seguidor(long/seg) vs psi'),
   pause,
   plot(psi,acel),grid,title('aceleración del seguidor(long/seg^2) vs
psi'),
   pause,
   plot(psi,sobreacel),grid,title('sobreaceleración del
seguidor(long/seg^3) vs psi'),
   pause,
end
```

```
function
[datosdsal_aux,theta_r,ro_r,vel,acel]=leva4(fi,w_paso,geometria,dmax,dmin
,ver)

%Sintetizamos y analizamos el cuarto tipo de leva: seguidor con rodillo,
oscilante.

N=5000;          %Divisiones.

l=geometria(1); e=geometria(2); rrod=geometria(3);


psi=0:2*pi/N:2*pi;
```

```
datosdsal_aux{6}{1}=fi;
dfi=deriv(fi);
ddfi=deriv(dfi);


%Calculo el radio en polares de la leva:
ro=sqrt(l^2+e^2-2*e*l*cos(fi));
datosdsal_aux{3}{1}=ro;
if ver,
   plot(psi,ro),grid,title('ro vs psi'),pause,
end,

%Calculo el angulo theta en polares:
%Primero calculo aux=psi+theta,teniendo en cuenta que debe ser continua
la función:
%Inicializo
if (l-e*cos(fi(1)))==0,          %Si el denominador es cero
   if fi(1)>=0,
      aux(1)=pi/2;
   else,
      aux(1)=-pi/2;
   end
else,
   aux(1)=atan((e*sin(fi(1)))/(l-e*cos(fi(1))));
   if aux(1)<0 & fi(1)>0,      %Para que esté por arriba el contacto,con
fi>0.
      aux(1)=aux(1)+pi;
   end
end
for i=2:(N+1),     %llenamos el resto del vector%
   if(l-e*cos(fi(i)))==0, aux(i)=pi/2;
   else, aux(i)=atan((e*sin(fi(i)))./(l-e*cos(fi(i))));
   end
   if(abs(aux(i)-aux(i-1))>(pi/4))          %si hay un salto%
      j=1;
      while((abs(aux(i)+j*pi-aux(i-1))>(pi/4))...
            &(abs(aux(i)-j*pi-aux(i-1))>(pi/4))) %busco el valor%
            j=j+1;                                  %apropiado para%
      end                                           %que haya%
      if(abs(aux(i)+j*pi-aux(i-1))<(pi/4)),       %continuidad%
         aux(i)=aux(i)+j*pi;
      else,
         aux(i)=aux(i)-j*pi;
      end
   end
end
if ver,
   plot(psi,aux),grid,title('aux vs psi'),pause,
end,

theta=aux-psi;
datosdsal_aux{3}{2}=theta;
if ver,
   plot(psi,theta),grid,title('theta vs psi'),pause,

   %Dibujo la leva:
   %Dibujo la leva (primitiva).
```

```
   polar(theta,ro),title('LEVA PRIMITIVA                    LEVA
PRIMITIVA'),hold on,
   polar(theta(1),ro(1),'*'),       %El punto de contacto en psi=0.
   hold off,pause,
end,


%Veamos ahora el ángulo de presión alpha:
%Según la fórmula que proporciona el libro:
if max(l*sin(fi(i)))*min(l*sin(fi(i)))<=0,   %en algún momento se hace
cero.
   error('l*sin(fi) pasa por cero,habrá un angulo de presión de +-pi/2:No
es admisible');
end
alpha=atan((e*(1+dfi)-l*cos(fi))./(l*sin(fi)));
datosdsal_aux{4}=alpha*(360/(2*pi));   %Lo pongo en grados para dibujar.
if ver,
   plot(psi,alpha),grid,title('ángulo de presión vs psi'),pause,
end,

%Lo calculo de una forma más general:
dtheta=derivtheta(theta);
dro=deriv(ro);
drorespttheta=dro./dtheta;
beta=atan(-drorespttheta./ro);
if ver,
   alphag=-pi/2+(aux+fi+beta);
   plot(psi,alphag),grid,title('ángulo de presión vs psi,expresión
general'),pause,
end,


%Veamos ahora la curvatura:
%Según el libro:
num=e^2*(1+dfi).^3-e*l*((1+dfi).*(2+dfi).*cos(fi)+ddfi.*sin(fi))+l^2;
den=(e^2*(1+dfi).^2-2*e*l*(1+dfi).*cos(fi)+l^2).^(3/2);
k=num./den;
datosdsal_aux{5}=k*rrod;   %Para dibujar rrod/rcur que es el parámetro de
interés.
if ver,
   plot(psi,k),grid,title('curvatura vs psi,según libro'),pause,

   %Según la fórmula general:
   ddrorespttheta=deriv(drorespttheta);
   ddrorespttheta2=ddrorespttheta./dtheta;
   den=(ro.^2+drorespttheta.^2).^(3/2);
   num=ro.^2+2*drorespttheta.^2-ro.*ddrorespttheta2;
   kg=num./den;
   plot(psi,kg),grid,title('curvatura primitiva vs psi,expresión
general'),pause
   %si el radio de rodadura es mayor que el radio de curvatura mínimo del
perfil primitivo,
   %se produce una cúspide (undercutting).
   %La razon rrod/rcur nos dice lo cerca que estamos de que se produzcan
cúspides,de que
   %la leva no sea válida.
end,
```

```
%Calculo el perfil real.
[ro_r,theta_r]=prim_a_r(ro,theta,beta,rrod);
datosdsal_aux{3}{3}=ro_r;
datosdsal_aux{3}{4}=theta_r;
if ver,
   plot(psi,ro_r),title('ro_r'),grid,pause,
   plot(psi,theta_r),title('theta_r'),grid,pause,

   %Dibujo:
   polar(theta,ro),hold on,
   title('LEVA PRIMITIVA Y REAL                    LEVA PRIMITIVA
YREAL'),
   polar(theta_r,ro_r),
   polar(theta_r,(dmax/2)*diag(ones(N+1))','r'),
   polar(theta_r(1),ro_r(1),'*'),       %El punto de contacto en psi=0.
   polar(theta_r,(dmin/2)*diag(ones(N+1))','r'),hold off,pause,
end,


%Calculo el area de la leva real:
dtheta_r=derivtheta(theta_r);
areaaux=.5*ro_r.^2.*dtheta_r*(2*pi/N);
%como el numero de intervalos es menor que el de nodos en uno,%
%anulo el ùltimo tèrmino,pues seria repetitivo%
%hay que cambiar el signo,pues dtheta es negativo %
areaaux(N+1)=0;
if ver,
   area=-sum(areaaux),
else,
   area=-sum(areaaux);
end,
datosdsal_aux{8}=area;

%cálculo la velocidad,la aceleración,y la sobreaceleración(cuya
%variación suave es un signo de calidad de la leva),angulares
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(fi));
else,
   w=ppval(w_paso,psi);
end

vel=dfi.*w;
datosdsal_aux{6}{2}=vel;
dvel=deriv(vel);
acel=dvel.*w;
datosdsal_aux{6}{3}=acel;
dacel=deriv(acel);
sobreacel=dacel.*w;
datosdsal_aux{6}{4}=sobreacel;
if ver,
   %Dibujo%
   plot(psi,fi),grid,title('angulo del seguidor vs psi'),
   pause,
   plot(psi,vel),grid,title('velocidad angular del seguidor(1/seg) vs
psi'),
```

```
   pause,
   plot(psi,acel),grid,title('aceleración angular del seguidor(1/seg^2)
vs psi'),
   pause,
   plot(psi,sobreacel),grid,title('sobreaceleración angular del
seguidor(1/seg^3) vs psi'),
   pause,
end,



function datosdsal_aux=leva5(s,w_paso,geometria,dmax,dmin,datosdin,ver)

%Es como en el tipo1,pero ahora hay también que considerar que el
movimiento ss y s (es
%decir: del seguidor,y de la válvula),difieren en una constante (salvo
efectos dinámicos.

N=5000;
dv=geometria(1);

%En el vector datosdinam,están los datos necesarios para el cálculo de
fuerzas, que no están
%en el vector geometria.
e=datosdin(1);mu=datosdin(2);Rint_l=datosdin(3);la=datosdin(4);lb=datosdi
n(5);
m3=datosdin(6);Fmo=datosdin(7);km=datosdin(8);masa_p_u_area=datosdin(9);


psi=0:2*pi/N:2*pi;
ss=s-dv;

%Diseño y analizo la leva llamando a la función leva1:
[datosdsal_aux,ro,theta,excent,vel_ss,acel_ss]=leva1(ss,w_paso,dmax,dmin,
ver);

if ver,
   %Calculo y dibujo la/las otras variables cinemáticas de interés,según
la cadena cinemática:
   plot(psi,s),grid,
   title('movimiento de la válvula (m) vs psi,las derivas son las mismas
que las del seguidor'),
   pause,
end,
datosdsal_aux{6}{5}=s; %El resto de datos cinemáticos los debuelve leva1.

%Analizamos ahora las fuerzas:
%Resolvemos el solido 3 (válvula):
Fin3_y=-m3*acel_ss;
Fm=Fmo+km*(ss-min(ss));
for i=1:N+1,
   %Primero lo hago sin rozamiento en las guias,para ver el signo de
fa_x,y fb_x,
   A=[-mu,1,1;
      1,0,0;
      (excent(i)-e),-(la-ss(i)),-(lb-ss(i))];
   b=[0,-Fin3_y(i)+Fm(i),0]';
```

```
   x=A\b;
   F23_y(i)=x(1);
   Fa_x(i)=x(2);
   Fb_x(i)=x(3);
   %Añado los términos del rozamiento en las guias,teniendo en cuenta los
signos sin
   %rozamiento en las guias.
   A(2,2)=-mu*sign(Fa_x(i))*sign(vel_ss(i));
   A(2,3)=-mu*sign(Fb_x(i))*sign(vel_ss(i));
   x=A\b;
   F23_y(i)=x(1);
   Fa_x(i)=x(2);
   Fb_x(i)=x(3);
end


%Las fuerzas de rozamiento:
F23_x=-mu*F23_y;
Fa_y=-mu*Fa_x.*sign(vel_ss);
Fb_y=-mu*Fb_x.*sign(vel_ss);

%Resolvemos el sólido 2 (la leva):
if size(w_paso)==[1 1],
   acel_psi=0;
else,
   w_pp=w_paso;
   dw_pp=fnder(w_paso);
   acel_psi=ppval(dw_pp,psi).*ppval(w_pp,psi);
end,
%calculo las fuerzas de inercia y el momento de inercia (en el eje):
dtheta=derivtheta(theta);
Io2_aux=(masa_p_u_area/4)*ro.^4.*dtheta.*(2*pi/N);
Io2_aux(N+1)=0;
Io2=-sum(Io2_aux);
Min2=-Io2*acel_psi;

%Calculo el area por las coordenadas x e y del c.d.m,respecto de la
leva.AXG=0; AYG=0;
AXG=0; AYG=0;
for i=1:20,
   if i==1,
      i_inf=1;
   else,
      i_inf=round((i-1)*(N/20));
   end
   i_sup=round(i*(N/20));
   theta_inf=theta(i_inf); theta_sup=theta(i_sup);
   theta_med=(theta_inf+theta_sup)/2;
   delta_theta=theta_inf-theta_sup;   %Pues theta decrece.
   ro_max=(ro(i_inf)+ro(i_sup))/2;
   for j=1:20,
      ro_inf=(j-1)*(ro_max/20); ro_sup=j*(ro_max/20);
      ro_med=(ro_inf+ro_sup)/2;
      delta_ro=ro_sup-ro_inf;
      AXG=AXG+ro_med*cos(theta_med)*ro_med*delta_theta*delta_ro;
      AYG=AYG+ro_med*sin(theta_med)*ro_med*delta_theta*delta_ro;
   end
```

```
end
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
%con minúsculas nos referimos al sistema de referencia de la barra fija
acel_axg2=-AXG*(cos(psi).*w.^2+sin(psi).*acel_psi)-AYG*(-
sin(psi).*w.^2+cos(psi).*acel_psi);
acel_ayg2=AXG*(-sin(psi).*w.^2+cos(psi).*acel_psi)-
AYG*(cos(psi).*w.^2+sin(psi).*acel_psi);
Fin2_x=-masa_p_u_area*acel_axg2;
Fin2_y=-masa_p_u_area*acel_ayg2;

F13_x=F23_x-Fin2_x;
F13_y=F23_y-Fin2_y;
Mroz=-mu*Rint_l*sqrt((F13_x).^2+(F13_y).^2);
Meje=-F23_x.*ss+F23_y.*excent-Min2-Mroz;

if ver,
   %Ahora dibujo:
   plot(psi,F23_y),grid,title('F23_y (N) vs psi');pause,
   plot(psi,F23_x),grid,title('F23_x (N) (rozamiento) vs psi'),pause,
   plot(psi,Fa_x),grid,title('Fa_x (N) vs psi'),pause,
   plot(psi,Fa_y),grid,title('Fa_y (N) (rozamiento) vs psi');pause,
   plot(psi,Fb_x),grid,title('Fb_x (N) vs psi');pause,
   plot(psi,Fb_y),grid,title('Fb_y (N) (rozamiento) vs  psi');pause,
   plot(psi,F13_x),grid,title('F13_x (N) vs psi'),pause,
   plot(psi,F13_y),grid,title('F13_y (N) vs psi'),pause,
   plot(psi,Mroz),grid,title('Mroz (Nm) (rozamiento) vs psi'),pause,
   plot(psi,Meje),grid,title('Meje (Nm) vs psi'),pause;
end,

datosdsal_aux{7}{1}=F23_y;
datosdsal_aux{7}{2}=F23_x;
datosdsal_aux{7}{3}=Fa_x;
datosdsal_aux{7}{4}=Fa_y;
datosdsal_aux{7}{5}=Fb_x;
datosdsal_aux{7}{6}=Fb_y;
datosdsal_aux{7}{7}=F13_x;
datosdsal_aux{7}{8}=F13_y;
datosdsal_aux{7}{9}=sqrt(F13_x.^2+F13_y.^2);
datosdsal_aux{7}{10}=Mroz;
datosdsal_aux{7}{11}=Meje;

%Calculo el trabajo:
trabajo_aux=Meje*(2*pi/N);
trabajo_aux(N+1)=0;  %El último seria repetitivo.
if ver,
   trabajo=sum(trabajo_aux),
else,
   trabajo=sum(trabajo_aux);
end,
datosdsal_aux{9}=trabajo;
```

```
function datosdsal_aux=leva6(s,w_paso,geometria,dmax,dmin,datosdin,ver)

%Usaremos la función leva4, y la cadena cinemática.

N=5000;
psi=0:2*pi/N:2*pi;

dv=geometria(1); db=geometria(2); e=geometria(3); l=geometria(4);
omega_b=geometria(5);
omega_l=geometria(6); rrod=geometria(7);

m5=datosdin(1); Io1=datosdin(2); m4=datosdin(3); Rg4=datosdin(4);
omega_g4=datosdin(5);
mu=datosdin(6); Rint_b=datosdin(7); mrod=datosdin(8); Irod=datosdin(9);
Rint_rod=datosdin(10);
Rint_l=datosdin(11); Fmo=datosdin(12); km=datosdin(13);
masa_p_u_area=datosdin(14);

%Calculo la/las otras variables de interés:
k=pi/2-omega_b+omega_l;
fi=asin((s-dv)/db)-k;


%Diseño y analizo la leva llamando a la función leva4:
[datosdsal_aux,theta_r,ro_r,vel_fi,acel_fi]=leva4(fi,w_paso,[l,e,rrod],dm
ax,dmin,ver);


if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
datosdsal_aux{6}{5}=s;
ds=deriv(s);
vel_s=ds.*w;
datosdsal_aux{6}{6}=vel_s;
dvel_s=deriv(vel_s);
acel_s=dvel_s.*w;
datosdsal_aux{6}{7}=acel_s;
dacel_s=deriv(acel_s);
sobreacel_s=dacel_s.*w;
datosdsal_aux{6}{8}=sobreacel_s;

if ver,
   %Dibujo%
   plot(psi,s),grid,title('movimiento de la válvula (long) vs psi'),
   pause,
   plot(psi,vel_s),grid,title('velocidad de la válvula (long/seg) vs
psi'),
   pause,
   plot(psi,acel_s),grid,title('aceleración de la válvula (long/seg^2) vs
psi'),
```

```
   pause,
   plot(psi,sobreacel_s),grid,title('sobreaceleración de la
válvula(long/seg^3) vs psi'),
   pause,
end,

%Calculamos ahora las fuerzas (suponemos que el rodillo no deliza):
%Sólido 5 (válvula):
Fin5_y=-m5*acel_s;
F45_y=-Fin5_y+Fmo+km*(s-min(s));

if ver,
   plot(psi,F45_y),grid,title('F45_y (N) vs psi'),pause,
end,
datosdsal_aux{7}{1}=F45_y;

%Sólidos 4 (balancín) y 3 (rodillo):
%Las fuerzas y el momento (respecto del eje) de inercia del 4:
Min4=-Io1*(-acel_fi);
ang=omega_b-(fi+omega_l)-omega_g4;
Fin4_x=-m4*(-Rg4*(-sin(ang).*((-vel_fi).^2)+cos(ang).*(-acel_fi)));
Fin4_y=-m4*(-Rg4*(cos(ang).*((-vel_fi).^2)+sin(ang).*(-acel_fi)));

%Los vectores n=(nx,ny), y t=(tx,ty);
nx=(1/rrod)*(e*sin(fi+omega_l)-(l*sin(omega_l)+ro_r.*cos(pi/2-
(theta_r+psi-omega_l))));
ny=(1/rrod)*(e*cos(fi+omega_l)-(l*cos(omega_l)-ro_r.*sin(pi/2-
(theta_r+psi-omega_l))));
tx=-ny;
ty=nx;

%Calulo nu_r (ángulo del radio vector con la normal):
dtheta_r=derivtheta(theta_r);
dro_r=deriv(ro_r);
drorespttheta_r=dro_r./dtheta_r;
nu_r=atan(-drorespttheta_r./ro_r);

%calculamos la velocidad y aceleración del rodillo:
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
if size(w_paso)==[1 1],
   acel_psi=0;
else,
   w_pp=w_paso;
   dw_pp=fnder(w_paso);
   acel_psi=ppval(dw_pp,psi).*ppval(w_pp,psi);
end,

vg3_t=e*cos(fi+omega_l).*vel_fi.*tx-e*sin(fi+omega_l).*vel_fi.*ty;
vel_theta_rod=(vg3_t-w.*ro_r.*cos(nu_r))/(rrod);
acel_theta_rod=deriv(vel_theta_rod).*w;

%Calculamos las fuerzas de inercia del rodillo:
Fin3_x=-mrod*e*(-sin(fi+omega_l).*vel_fi.^2+cos(fi+omega_l).*acel_fi);
```

```
Fin3_y=-mrod*e*(-cos(fi+omega_l).*vel_fi.^2-sin(fi+omega_l).*acel_fi);
Min3=-Irod*acel_theta_rod;

%resuelvo:
for i=1:N+1,
   for ii=1:2;   %iteraciones para aproximar Mroz_b y Mroz_rod,
      if ii==1,
         Mroz_b(i)=0; Mroz_rod(i)=0;
      else,
         Mroz_b(i)=mu*Rint_b*sqrt(x(1)^2+x(2)^2)*sign(vel_fi(i));

Mroz_rod(i)=mu*Rint_rod*sqrt(x(3)^2+x(4)^2)*sign(vel_theta_rod(i)+vel_fi(
i));
      end
      if mu==0,
         ii=2;      %Para que en este caso no itere.
      end
      A=[1 0 1 0 0 0
         0 1 0 1 0 0
         0 0 -e*cos(omega_l+fi(i)) e*sin(omega_l+fi(i)) 0 0
         0 0 0 0 -rrod 0
         0 0 -1 0 tx(i) nx(i)
         0 0 0 -1 ty(i) ny(i)];
      b=[-Fin4_x(i)
         F45_y(i)-Fin4_y(i)
         -F45_y(i)*db*cos((pi/2)-(omega_b-(fi(i)+omega_l)))-Min4(i)-
Mroz_b(i)-Mroz_rod(i)
         Mroz_rod(i)-Min3(i)
         -Fin3_x(i)
         -Fin3_y(i)];
      x=A\b;
   end
   F14_x(i)=x(1); F14_y(i)=x(2); F34_x(i)=x(3); F34_y(i)=x(4);
F23_t(i)=x(5); F23_n(i)=x(6);
end



if ver,
   %Dibujo:
   plot(psi,F14_x),grid,title('F14_x (N) vs psi'),pause,
   plot(psi,F14_y),grid,title('F14_y (N) vs psi'),pause,
   plot(psi,Mroz_b),grid,title('Mroz_b (Nm) (rozamiento) vs psi'),pause,
   plot(psi,F34_x),grid,title('F34_x (N) vs psi'),pause,
   plot(psi,F34_y),grid,title('F34_y (N) vs psi'),pause,
   plot(psi,Mroz_rod),grid,title('Mroz_rod (Nm) (rozamiento) vs
psi'),pause,
   plot(psi,F23_n),grid,title('F23_n (N) vs psi'),pause,
   plot(psi,F23_t),grid,title('F23_t (N) (rozamiento) vs psi'),pause,
end,

datosdsal_aux{7}{2}=F14_x;
datosdsal_aux{7}{3}=F14_y;
datosdsal_aux{7}{4}=sqrt(F14_x.^2+F14_y.^2);
datosdsal_aux{7}{5}=Mroz_b;
datosdsal_aux{7}{6}=F34_x;
datosdsal_aux{7}{7}=F34_y;
```

```
datosdsal_aux{7}{8}=sqrt(F34_x.^2+F34_y.^2);
datosdsal_aux{7}{9}=Mroz_rod;
datosdsal_aux{7}{10}=F23_n;
datosdsal_aux{7}{11}=F23_t;

%Sólido 2 (leva)
%calculo las fuerzas y el momento de inercia(respecto del eje):

dtheta_r=derivtheta(theta_r);
Io2_aux=(masa_p_u_area/4)*ro_r.^4.*dtheta_r.*(2*pi/N);
Io2_aux(N+1)=0;
Io2=-sum(Io2_aux);
Min2=-Io2*acel_psi;

%Calculo el area por las coordenadas x e y del c.d.m,respecto de la leva.
AXG=0; AYG=0;
for i=1:20,
   if i==1,
      i_inf=1;
   else,
      i_inf=round((i-1)*(N/20));
   end
   i_sup=round(i*(N/20));
   theta_r_inf=theta_r(i_inf); theta_r_sup=theta_r(i_sup);
   theta_r_med=(theta_r_inf+theta_r_sup)/2;
   delta_theta_r=theta_r_inf-theta_r_sup;   %Pues theta decrece.
   ro_r_max=(ro_r(i_inf)+ro_r(i_sup))/2;
   for j=1:20,
      ro_r_inf=(j-1)*(ro_r_max/20); ro_r_sup=j*(ro_r_max/20);
      ro_r_med=(ro_r_inf+ro_r_sup)/2;
      delta_ro_r=ro_r_sup-ro_r_inf;

AXG=AXG+ro_r_med*cos(theta_r_med)*ro_r_med*delta_theta_r*delta_ro_r;

AYG=AYG+ro_r_med*sin(theta_r_med)*ro_r_med*delta_theta_r*delta_ro_r;
   end
end
%con minúsculas nos referimos al sistema de referencia de la barra fija
ang=psi+pi/2-omega_l;
acel_axg2=AXG*(cos(ang).*w.^2+sin(ang).*acel_psi)+AYG*(-
sin(ang).*w.^2+...
   cos(ang).*acel_psi);
acel_ayg2=-AXG*(-
sin(ang).*w.^2+cos(ang).*acel_psi)+AYG*(cos(ang).*w.^2+...
   sin(ang).*acel_psi);
Fin2_x=-masa_p_u_area*acel_axg2;
Fin2_y=-masa_p_u_area*acel_ayg2;

F12_x=F23_n.*nx+F23_t.*tx-Fin2_x;
F12_y=F23_n.*ny+F23_t.*ty-Fin2_y;
Mroz_l=-mu*Rint_l*sqrt(F12_x.^2+F12_y.^2);
%u y v son vectores auxiliares:
ang=pi/2-omega_l+psi+theta_r;
ux=-ro_r.*cos(ang);
uy=-ro_r.*sin(ang);
vx=-F23_n.*nx-F23_t.*tx;
vy=-F23_n.*ny-F23_t.*ty;
```

```
Meje=-Min2-Mroz_l-(ux.*vy-uy.*vx);

if ver,
   %Dibujo:
   plot(psi,F12_x),grid,title('F12_x (N) vs psi'),pause,
   plot(psi,F12_y),grid,title('F12_y (N) vs psi'),pause,
   plot(psi,Mroz_l),grid,title('Mroz_l (Nm) (rozamiento) vs psi'),pause,
   plot(psi,Meje),grid,title('Meje (Nm) vs psi'),pause,
end,
datosdsal_aux{7}{12}=F12_x;
datosdsal_aux{7}{13}=F12_y;
datosdsal_aux{7}{14}=sqrt(F12_x.^2+F12_y.^2);
datosdsal_aux{7}{15}=Mroz_l;
datosdsal_aux{7}{16}=Meje;

%Calculo el trabajo:
trabajo_aux=Meje*(2*pi/N);
trabajo_aux(N+1)=0;  %El último seria repetitivo.
if ver,
   trabajo=sum(trabajo_aux),
else,
   trabajo=sum(trabajo_aux);
end,

datosdsal_aux{9}=trabajo;
```

```
function datosdsal_aux=leva7(s,w_paso,geometria,dmax,dmin,datosdin,ver)

%Usaremos la función leva2,y la cadena cinemática.

N=5000;
psi=0:2*pi/N:2*pi;

dv=geometria(1); db=geometria(2); e=geometria(3); l=geometria(4);
omega_b=geometria(5);
omega_l=geometria(6);

m4=datosdin(1); Io1=datosdin(2); m3=datosdin(3); Rg3=datosdin(4);
omega_g3=datosdin(5);
mu=datosdin(6); Rint_b=datosdin(7); Rint_l=datosdin(8);
masa_p_u_area=datosdin(9);
Fmo=datosdin(10); km=datosdin(11);

%Calculo la/las otras variables de interés:
k=pi/2-omega_b+omega_l;
fi=asin((s-dv)/db)-k;

%Diseño y analizo la leva llamando a la función leva2:
[datosdsal_aux,theta,ro,dfi,vel_fi,acel_fi]=leva2(fi,w_paso,[e,l],dmax,dm
in,ver);

%Las otras variables cinemáticas:
```

```
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
datosdsal_aux{6}{5}=s;
ds=deriv(s);
vel_s=ds.*w;
datosdsal_aux{6}{6}=vel_s;
dvel_s=deriv(vel_s);
acel_s=dvel_s.*w;
datosdsal_aux{6}{7}=acel_s;
dacel_s=deriv(acel_s);
sobreacel_s=dacel_s.*w;
datosdsal_aux{6}{8}=sobreacel_s;

if ver,
   %Dibujo%
   plot(psi,s),grid,title('movimiento de la válvula (long) vs psi'),
   pause,
   plot(psi,vel_s),grid,title('velocidad de la válvula (long/seg) vs
psi'),
   pause,
   plot(psi,acel_s),grid,title('aceleración de la válvula (long/seg^2) vs
psi'),
   pause,
   plot(psi,sobreacel_s),grid,title('sobreaceleración de la
válvula(long/seg^3) vs psi'),
   pause,
end,

%Calculo ahora las fuerzas:

%sólido 4 (válvula):
Fin4_y=-m4*acel_s;
F34_y=-Fin4_y+Fmo+km*(s-min(s));
datosdsal_aux{7}{1}=F34_y;

if ver,
   %Dibujo:
   plot(psi,F34_y),grid,title('F34_y (N) vs psi');pause,
end,

%sólido 3 (balancín)
Min3=-Io1*(-acel_fi);
ang=omega_b-(fi+omega_l)-omega_g3;
Fin3_x=-m3*(-Rg3*(-sin(ang).*((-vel_fi).^2)+cos(ang).*(-acel_fi)));
Fin3_y=-m3*(-Rg3*(cos(ang).*((-vel_fi).^2)+sin(ang).*(-acel_fi)));

for i=1:N+1,
   for ii=1:2,
      if ii==1,
         Mroz_b(i)=0;
      else,

Mroz_b(i)=mu*Rint_b*sqrt((F13_x(i)).^2+(F13_y(i)).^2).*sign(fi(i));
      end
```

```
      if mu==0, ii=2; end    %Basta con hacelo una vez

      F23_n(i)=(F34_y(i)*db.*cos((pi/2)-(omega_b-
(fi(i)+omega_l)))+Min3(i)+Mroz_b(i))/...
         (-mu*e+((l*cos(fi(i)))/(1+dfi(i))));
      F13_y(i)=F34_y(i)-mu*F23_n(i)*cos(omega_l+fi(i))+F23_n(i)*cos(pi/2-
(omega_l+fi(i)))...
         -Fin3_y(i);
      F13_x(i)=-mu*F23_n(i)*sin(omega_l+fi(i))-F23_n(i)*sin(pi/2-
(omega_l+fi(i)))-Fin3_x(i);
   end
end


F23_t=mu*F23_n;

if ver,
   %Dibujo:
   plot(psi,F23_n),grid,title('F23_n (N) vs psi'),pause,
   plot(psi,F23_t),grid,title('F23_t (N) (rozamiento) vs psi'),pause,
   plot(psi,F13_x),grid,title('F13_x (N) vs psi'),pause,
   plot(psi,F13_y),grid,title('F13_y (N) vs psi'),pause,
   plot(psi,Mroz_b),grid,title('Mroz_b (Nm) (rozamiento) vs psi'),pause,
end,

datosdsal_aux{7}{2}=F23_n;
datosdsal_aux{7}{3}=F23_t;
datosdsal_aux{7}{4}=F13_x;
datosdsal_aux{7}{5}=F13_y;
datosdsal_aux{7}{6}=sqrt(F13_x.^2+F13_y.^2);
datosdsal_aux{7}{7}=Mroz_b;

%sólido 2 (leva):
if size(w_paso)==[1 1],
   acel_psi=0;
else,
   w_pp=w_paso;
   dw_pp=fnder(w_paso);
   acel_psi=ppval(dw_pp,psi).*ppval(w_pp,psi);
end,
%calculo las fuerzas y el momento de inercia(respecto del eje):
dtheta=derivtheta(theta);
Io2_aux=(masa_p_u_area/4)*ro.^4.*dtheta.*(2*pi/N);
Io2_aux(N+1)=0;
Io2=-sum(Io2_aux);
Min2=-Io2*acel_psi;

%Calculo el area por las coordenadas x e y del c.d.m,respecto de la leva.
AXG=0; AYG=0;
for i=1:20,
   if i==1,
      i_inf=1;
   else,
      i_inf=round((i-1)*(N/20));
   end
   i_sup=round(i*(N/20));
   theta_inf=theta(i_inf); theta_sup=theta(i_sup);
```

```
   theta_med=(theta_inf+theta_sup)/2;
   delta_theta=theta_inf-theta_sup;   %Pues theta decrece.
   ro_max=(ro(i_inf)+ro(i_sup))/2;
   for j=1:20,
      ro_inf=(j-1)*(ro_max/20); ro_sup=j*(ro_max/20);
      ro_med=(ro_inf+ro_sup)/2;
      delta_ro=ro_sup-ro_inf;
      AXG=AXG+ro_med*cos(theta_med)*ro_med*delta_theta*delta_ro;
      AYG=AYG+ro_med*sin(theta_med)*ro_med*delta_theta*delta_ro;
   end
end
%con minúsculas nos referimos al sistema de referencia de la barra fija
ang=psi+pi/2-omega_l;
acel_axg2=AXG*(cos(ang).*w.^2+sin(ang).*acel_psi)+AYG*(-
sin(ang).*w.^2+...
   cos(ang).*acel_psi);
acel_ayg2=-AXG*(-
sin(ang).*w.^2+cos(ang).*acel_psi)+AYG*(cos(ang).*w.^2+...
   sin(ang).*acel_psi);
Fin2_x=-masa_p_u_area*acel_axg2;
Fin2_y=-masa_p_u_area*acel_ayg2;

F12_x=mu*F23_n.*sin(omega_l+fi)+F23_n.*sin(pi/2-(omega_l+fi))-Fin2_x;
F12_y=mu*F23_n.*cos(omega_l+fi)-F23_n.*cos(pi/2-(omega_l+fi))-Fin2_y;
Mroz_l=-mu*Rint_l*sqrt((F12_x).^2+(F12_y).^2);
Meje=-Min2+mu*F23_n.*(l*sin(fi)+e)+F23_n.*(l*dfi.*cos(fi)./(1+dfi))-
Mroz_l;
if ver,
   %Dibujo:
   plot(psi,F12_x),grid,title('F12_x (N) vs psi'),pause,
   plot(psi,F12_y),grid,title('F12_y (N) vs psi'),pause,
   plot(psi,Mroz_l),grid,title('Mroz_l (Nm) (rozamiento) vs psi'),pause,
   plot(psi,Meje),grid,title('Meje (Nm) vs psi'),pause,
end,

datosdsal_aux{7}{8}=F12_x;
datosdsal_aux{7}{9}=F12_y;
datosdsal_aux{7}{10}=sqrt(F12_x.^2+F12_y.^2);
datosdsal_aux{7}{11}=Mroz_l;
datosdsal_aux{7}{12}=Meje;

%Calculo el trabajo:
trabajo_aux=Meje*(2*pi/N);
trabajo_aux(N+1)=0;  %El último seria repetitivo.
if ver,
   trabajo=sum(trabajo_aux),
else,
   trabajo=sum(trabajo_aux);
end,
datosdsal_aux{9}=trabajo;



function  datosdsal_aux=leva8(s,w_paso,geometria,dmax,dmin,datosdin,ver)

%Usaremos la función leva1, y la cadena cinemática.
```

```
N=5000;
psi=0:2*pi/N:2*pi;

dv=geometria(1); db1=geometria(2); db2=geometria(3); l=geometria(4);
omega_b=geometria(5);
sigma=geometria(6); k1=geometria(7); k2=geometria(8); ac=geometria(9);
e=geometria(10);

m6=datosdin(1); Fmo=datosdin(2); km=datosdin(3); Io1=datosdin(4);
m5=datosdin(5);
Rg5=datosdin(6); omega_g5=datosdin(7); mu=datosdin(8);
Rint_b=datosdin(9); m4=datosdin(10);
Ig4=datosdin(11); lg4=datosdin(12); lsup=datosdin(13); linf=datosdin(14);
acs=datosdin(15);
ancho=datosdin(16); m3=datosdin(17); Rint_l=datosdin(18);
masa_p_u_area=datosdin(19);

%Calculo la/las otras variables de interés:
alpha=asin((s-dv)/db1)-pi/2+omega_b;
beta=asin((k1-db2*sin(alpha-sigma))/l)+sigma;
ss=k2-ac-db2*cos(alpha-sigma)-l*cos(beta-sigma);


%Diseño y analizo la leva llamando a la función leva1:
[datosdsal_aux,ro,theta,excent,vel_ss,acel_ss]=leva1(ss,w_paso,dmax,dmin,
ver);

%Dibujo:
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
%La válvula:
ds=deriv(s);
vel_s=ds.*w;
dvel_s=deriv(vel_s);
acel_s=dvel_s.*w;
dacel_s=deriv(acel_s);
sobreacel_s=dacel_s.*w;

if ver,
   plot(psi,s),grid,title('movimiento de la válvula (m) vs psi'),
   pause,
   plot(psi,vel_s),grid,title('velocidad la válvula (m/seg) vs psi'),
   pause,
   plot(psi,acel_s),grid,title('aceleración de la válvula (m/seg^2) vs
psi'),
   pause,
   plot(psi,sobreacel_s),grid,title('sobreaceleración de la
válvula(m/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{5}=s;
datosdsal_aux{6}{6}=vel_s;
```

```
datosdsal_aux{6}{7}=acel_s;
datosdsal_aux{6}{8}=sobreacel_s;

%El ángulo alpha:
dalpha=deriv(alpha);
vel_alpha=dalpha.*w;
dvel_alpha=deriv(vel_alpha);
acel_alpha=dvel_alpha.*w;
dacel_alpha=deriv(acel_alpha);
sobreacel_alpha=dacel_alpha.*w;

if ver,
   plot(psi,alpha),grid,title('angulo alpha vs psi'),
   pause,
   plot(psi,vel_alpha),grid,title('velocidad angular de alpha (1/seg) vs
psi'),
   pause,
   plot(psi,acel_alpha),grid,title('aceleración angular de alpha
(1/seg^2) vs psi'),
   pause,
   plot(psi,sobreacel_alpha),grid,title('sobreaceleración angular de
alpha (1/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{9}=alpha;
datosdsal_aux{6}{10}=vel_alpha;
datosdsal_aux{6}{11}=acel_alpha;
datosdsal_aux{6}{12}=sobreacel_alpha;

%El ángulo beta:
dbeta=deriv(beta);
vel_beta=dbeta.*w;
dvel_beta=deriv(vel_beta);
acel_beta=dvel_beta.*w;
dacel_beta=deriv(acel_beta);
sobreacel_beta=dacel_beta.*w;

if ver,
   plot(psi,beta),grid,title('angulo beta vs psi'),
   pause,
   plot(psi,vel_beta),grid,title('velocidad angular de beta (1/seg) vs
psi'),
   pause,
   plot(psi,acel_beta),grid,title('aceleración angular de beta (1/seg^2)
vs psi'),
   pause,
   plot(psi,sobreacel_beta),grid,title('sobreaceleración angular de beta
(1/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{13}=beta;
datosdsal_aux{6}{14}=vel_beta;
datosdsal_aux{6}{15}=acel_beta;
datosdsal_aux{6}{16}=sobreacel_beta;
```

```
%Resolvemos ahora las fuerzas:

%Primero el sólido 6 (válvula):
Fin6=-m6*acel_s;
F56_y=-Fin6+Fmo+km*(s-min(s));

if ver,
   %Dibujo:
   plot(psi,F56_y),grid,title('F56_y (N) vs psi'),pause,
end,
datosdsal_aux{7}{1}=F56_y;

%Resuelvo los solidos 5 (balancin) y 4 (empujador):
%Calculo fuerzas y momentos de inercia:
Min5=-Io1*(-acel_alpha);
Min4=-Ig4*(-acel_beta);

ang=omega_b-alpha-omega_g5;
Fin5_x=-m5*(-Rg5*(-sin(ang).*((-vel_alpha).^2)+cos(ang).*(-acel_alpha)));
Fin5_y=-m5*(-Rg5*(cos(ang).*((-vel_alpha).^2)+sin(ang).*(-acel_alpha)));

Fin4_x=-m4*(db2*cos(alpha).*acel_alpha-
db2*sin(alpha).*(vel_alpha.^2)+lg4*cos(beta).*acel_beta...
   -lg4*sin(beta).*(vel_beta.^2));
Fin4_y=-m4*(-db2*sin(alpha).*acel_alpha-db2*cos(alpha).*(vel_alpha.^2)-
lg4*sin(beta).*acel_beta...
   -lg4*cos(beta).*(vel_beta.^2));
for i=1:N+1,
   for ii=1:2;   %iteraciones para aproximar Mroz_b,
      if ii==1,
         Mroz_b(i)=0;
      else,
         Mroz_b(i)=mu*Rint_b*sqrt(x(1)^2+x(2)^2)*sign(vel_alpha(i));
      end
      if mu==0,
         ii=2;      %Para que en este caso no itere.
      end
      A=[1 0 1 0 0 0
         0 1 0 1 0 0
         0 0 -db2*cos(alpha(i)) db2*sin(alpha(i)) 0 0
         0 0 -1 0 1 0
         0 0 0 -1 0 1
         0 0 -lg4*cos(beta(i)) lg4*sin(beta(i)) -(l-lg4)*cos(beta(i)) (l-
lg4)*sin(beta(i))];
      b=[-Fin5_x(i)
         F56_y(i)-Fin5_y(i)
         -F56_y(i)*db1*cos((pi/2)-(omega_b-alpha(i)))-Min5(i)-Mroz_b(i)
         -Fin4_x(i)
         -Fin4_y(i)
         -Min4(i)];
      x=A\b;
   end
   F15_x(i)=x(1); F15_y(i)=x(2); F45_x(i)=x(3); F45_y(i)=x(4);
F34_x(i)=x(5); F34_y(i)=x(6);
end
```

```
if ver,
   %Dibujo:
   plot(psi,F15_x),grid,title('F15_x (N) vs psi'),pause,
   plot(psi,F15_y),grid,title('F15_y (N) vs psi'),pause,
   plot(psi,Mroz_b),grid,title('Mroz_b (Nm) (rozamiento) vs psi'),pause,
   plot(psi,F45_x),grid,title('F45_x (N) vs psi'),pause,
   plot(psi,F45_y),grid,title('F45_y (N) vs psi'),pause,
   plot(psi,F34_x),grid,title('F34_x (N) vs psi'),pause,
   plot(psi,F34_y),grid,title('F34_y (N) vs psi'),pause,
end,

datosdsal_aux{7}{2}=F15_x;
datosdsal_aux{7}{3}=F15_y;
datosdsal_aux{7}{4}=sqrt(F15_x.^2+F15_y.^2);
datosdsal_aux{7}{5}=Mroz_b;
datosdsal_aux{7}{6}=F45_x;
datosdsal_aux{7}{7}=F45_y;
datosdsal_aux{7}{8}=sqrt(F45_x.^2+F45_y.^2);
datosdsal_aux{7}{9}=F34_x;
datosdsal_aux{7}{10}=F34_y;
datosdsal_aux{7}{11}=sqrt(F34_x.^2+F34_y.^2);

%Resuelvo el sólido 3 (taqué):
%Ahora trabajo en ejes xp yp
F43_yp=(-F34_y)*cos(sigma)+(-F34_x)*sin(sigma);
F43_xp=-(-F34_y)*sin(sigma)+(-F34_x)*cos(sigma);
Fin3_yp=-m3*(-acel_ss);
%Veo el caso sin rozamiento,para ver en que situación estamos,según el
Mdesequilibrio:
F23_n=F43_yp+Fin3_yp;
F23_t=mu*F23_n;
Mdeseq=F23_n.*(excent-e)-F23_t*ac;
Nia=diag(zeros(N+1)); Ndb=Nia; Nda=Nia; Nib=Nia;   %Para que tengan algún
valor
for i=1:N+1,
   if lsup>=ss(i)+ac+acs,
      apa=acs;
   else,
      apa=lsup-ac-ss(i);
   end
   if linf<=ss(i),
      apb=ac;
   else,
      apb=ac+ss(i)-linf;
   end
   if Mdeseq(i)>=0,
      A=[mu -1 1
         -1 mu*sign(vel_ss(i)) mu*sign(vel_ss(i))
       excent(i)-e-mu*ac -apa+mu*sign(vel_ss(i))*ancho/2 -apb-
mu*sign(vel_ss(i))*ancho/2];
      b=[-F43_xp(i)
         -F43_yp(i)-Fin3_yp(i)
         0];
      x=A\b;
      F23_n(i)=x(1); Nia(i)=x(2); Ndb(i)=x(3);
   else,
```

```
      A=[mu 1 -1
         -1 mu*sign(vel_ss(i)) mu*sign(vel_ss(i))
         excent(i)-e-mu*ac apa-mu*sign(vel_ss(i))*ancho/2
apb+mu*sign(vel_ss(i))*ancho/2];
      b=[-F43_xp(i)
         -F43_yp(i)-Fin3_yp(i)
         0];
      x=A\b;
      F23_n(i)=x(1); Nda(i)=x(2); Nib(i)=x(3);
   end
end


F23_t=mu*F23_n;
Tia_yp=mu*Nia.*sign(vel_ss');
Tdb_yp=mu*Ndb.*sign(vel_ss');
Tda_yp=mu*Nda.*sign(vel_ss');
Tib_yp=mu*Nib.*sign(vel_ss');

if ver,
   %dibujo,
   plot(psi,F23_n),grid,title('F23_n (N) vs psi'),pause,
   plot(psi,F23_t),grid,title('F23_t (rozamiento) (N) vs psi'),pause,
   plot(psi,Nia),grid,title('Nia (N) vs psi'),pause,
   plot(psi,Tia_yp),grid,title('Tia_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Ndb),grid,title('Ndb (N) vs psi'),pause,
   plot(psi,Tdb_yp),grid,title('Tdb_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Nda),grid,title('Nda (N) vs psi'),pause,
   plot(psi,Tda_yp),grid,title('Tda_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Nib),grid,title('Nib (N) vs psi'),pause,
   plot(psi,Tib_yp),grid,title('Tib_yp (rozamiento) (N) vs psi'),pause,
end,

datosdsal_aux{7}{12}=F23_n;
datosdsal_aux{7}{13}=F23_t;
datosdsal_aux{7}{14}=Nia;
datosdsal_aux{7}{15}=Tia_yp;
datosdsal_aux{7}{16}=Ndb;
datosdsal_aux{7}{17}=Tdb_yp;
datosdsal_aux{7}{18}=Nda;
datosdsal_aux{7}{19}=Tda_yp;
datosdsal_aux{7}{20}=Nib;
datosdsal_aux{7}{21}=Tib_yp;

%Resuelvo el sólido 2 (leva):
if size(w_paso)==[1 1],
   acel_psi=0;
else,
   w_pp=w_paso;
   dw_pp=fnder(w_paso);
   acel_psi=ppval(dw_pp,psi).*ppval(w_pp,psi);
end,
%calculo la inercia:
dtheta=derivtheta(theta);
Io2_aux=(masa_p_u_area/4)*ro.^4.*dtheta.*(2*pi/N);
Io2_aux(N+1)=0;
Io2=-sum(Io2_aux);
```

```
Min2=-Io2*acel_psi;
%Calculo el area por las coordenadas x e y del c.d.m,respecto de la
leva.AXG=0; AYG=0;
AXG=0; AYG=0;
for i=1:20,
   if i==1,
      i_inf=1;
   else,
      i_inf=round((i-1)*(N/20));
   end
   i_sup=round(i*(N/20));
   theta_inf=theta(i_inf); theta_sup=theta(i_sup);
   theta_med=(theta_inf+theta_sup)/2;
   delta_theta=theta_inf-theta_sup;   %Pues theta decrece.
   ro_max=(ro(i_inf)+ro(i_sup))/2;
   for j=1:20,
      ro_inf=(j-1)*(ro_max/20); ro_sup=j*(ro_max/20);
      ro_med=(ro_inf+ro_sup)/2;
      delta_ro=ro_sup-ro_inf;
      AXG=AXG+ro_med*cos(theta_med)*ro_med*delta_theta*delta_ro;
      AYG=AYG+ro_med*sin(theta_med)*ro_med*delta_theta*delta_ro;
   end
end
%con minúsculas nos referimos al sistema de referencia de la barra fija
%Con respecto al tipo 5,solo hay que cambiarle el signo,pues los ejes en
un caso y en
%otro son en sentido contrario.
acel_axpg2=AXG*(cos(psi).*w.^2+sin(psi).*acel_psi)+AYG*(-
sin(psi).*w.^2+cos(psi).*acel_psi);
acel_aypg2=-AXG*(-
sin(psi).*w.^2+cos(psi).*acel_psi)+AYG*(cos(psi).*w.^2+sin(psi).*acel_psi
);
Fin2_xp=-masa_p_u_area*acel_axpg2;
Fin2_yp=-masa_p_u_area*acel_aypg2;

F12_xp=mu*F23_n-Fin2_xp;
F12_yp=-F23_n-Fin2_yp;
Mroz_l=-mu*Rint_l*sqrt((F12_xp).^2+(F12_yp).^2);
Meje=F23_n.*excent+mu*F23_n.*ss-Min2-Mroz_l;

if ver,
   %Dibujo:
   plot(psi,F12_xp),grid,title('F12_xp (N) vs psi'),pause,
   plot(psi,F12_yp),grid,title('F12_yp (N) vs psi'),pause,
   plot(psi,Mroz_l),grid,title('Mroz_l (Nm) (rozamiento) vs psi'),pause,
   plot(psi,Meje),grid,title('Meje (Nm) vs psi'),pause,
end,

datosdsal_aux{7}{22}=F12_xp;
datosdsal_aux{7}{23}=F12_yp;
datosdsal_aux{7}{24}=sqrt(F12_xp.^2+F12_yp.^2);
datosdsal_aux{7}{25}=Mroz_l;
datosdsal_aux{7}{26}=Meje;

%Calculo el trabajo:
trabajo_aux=Meje*(2*pi/N);
trabajo_aux(N+1)=0;  %El último seria repetitivo.
```

```
if ver,
   trabajo=sum(trabajo_aux),
else,
   trabajo=sum(trabajo_aux);
end,
datosdsal_aux{9}=trabajo;



function datosdsal_aux=leva9(s,w_paso,geometria,dmax,dmin,datosdin,ver)

%Usaremos la función leva3, y la cadena cinemática.

N=5000;
psi=0:2*pi/N:2*pi;

dv=geometria(1); db1=geometria(2); db2=geometria(3); l=geometria(4);
omega_b=geometria(5);
sigma=geometria(6); k1=geometria(7); k2=geometria(8); ac=geometria(9);
e=geometria(10);
rrod=geometria(11);

m6=datosdin(1); Fmo=datosdin(2); km=datosdin(3); Io1=datosdin(4);
m5=datosdin(5);
Rg5=datosdin(6); omega_g5=datosdin(7); mu=datosdin(8);
Rint_b=datosdin(9); m4=datosdin(10);
Ig4=datosdin(11); lg4=datosdin(12); lsup=datosdin(13); linf=datosdin(14);
acs=datosdin(15);
aci=datosdin(16); ancho=datosdin(17); m3=datosdin(18); mrod=datosdin(19);
Irod=datosdin(20);
Rint_rod=datosdin(21); Rint_l=datosdin(22); masa_p_u_area=datosdin(23);

%Calculo la/las otras variables de interés:
alpha=asin((s-dv)/db1)-pi/2+omega_b;
beta=asin((k1-db2*sin(alpha-sigma))/l)+sigma;
ss=k2-ac-db2*cos(alpha-sigma)-l*cos(beta-sigma);

%Diseño y analizo la leva llamando a la función leva3:
[datosdsal_aux,theta,ro,theta_r,ro_r,vel_ss,acel_ss]=leva3(ss,w_paso,[e,r
rod],dmax,dmin,ver);

%Dibujo:
if(size(w_paso)==size(1)),
   w=w_paso*ones(size(s));
else,
   w=ppval(w_paso,psi);
end
%La válvula:
ds=deriv(s);
vel_s=ds.*w;
dvel_s=deriv(vel_s);
acel_s=dvel_s.*w;
dacel_s=deriv(acel_s);
sobreacel_s=dacel_s.*w;

if ver,
```

```
   plot(psi,s),grid,title('movimiento de la válvula (long) vs psi'),
   pause,
   plot(psi,vel_s),grid,title('velocidad la válvula (long/seg) vs psi'),
   pause,
   plot(psi,acel_s),grid,title('aceleración de la válvula (long/seg^2) vs
psi'),
   pause,
   plot(psi,sobreacel_s),grid,title('sobreaceleración de la
válvula(long/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{5}=s;
datosdsal_aux{6}{6}=vel_s;
datosdsal_aux{6}{7}=acel_s;
datosdsal_aux{6}{8}=sobreacel_s;

%El ángulo alpha:
dalpha=deriv(alpha);
vel_alpha=dalpha.*w;
dvel_alpha=deriv(vel_alpha);
acel_alpha=dvel_alpha.*w;
dacel_alpha=deriv(acel_alpha);
sobreacel_alpha=dacel_alpha.*w;

if ver,
   plot(psi,alpha),grid,title('angulo alpha vs psi'),
   pause,
   plot(psi,vel_alpha),grid,title('velocidad angular de alpha (1/seg) vs
psi'),
   pause,
   plot(psi,acel_alpha),grid,title('aceleración angular de alpha
(1/seg^2) vs psi'),
   pause,
   plot(psi,sobreacel_alpha),grid,title('sobreaceleración angular de
alpha (1/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{9}=alpha;
datosdsal_aux{6}{10}=vel_alpha;
datosdsal_aux{6}{11}=acel_alpha;
datosdsal_aux{6}{12}=sobreacel_alpha;

%El ángulo beta:
dbeta=deriv(beta);
vel_beta=dbeta.*w;
dvel_beta=deriv(vel_beta);
acel_beta=dvel_beta.*w;
dacel_beta=deriv(acel_beta);
sobreacel_beta=dacel_beta.*w;

if ver,
   plot(psi,beta),grid,title('angulo beta vs psi'),
   pause,
   plot(psi,vel_beta),grid,title('velocidad angular de beta (1/seg) vs
psi'),
```

```
   pause,
   plot(psi,acel_beta),grid,title('aceleración angular de beta (1/seg^2)
vs psi'),
   pause,
   plot(psi,sobreacel_beta),grid,title('sobreaceleración angular de beta
(1/seg^3) vs psi'),
   pause,
end,

datosdsal_aux{6}{13}=beta;
datosdsal_aux{6}{14}=vel_beta;
datosdsal_aux{6}{15}=acel_beta;
datosdsal_aux{6}{16}=sobreacel_beta;

%Resolvemos ahora las fuerzas:

%Primero el sólido 6 (válvula):
Fin6=-m6*acel_s;
F56_y=-Fin6+Fmo+km*(s-min(s));

if ver,
   %Dibujo:
   plot(psi,F56_y),grid,title('F56_y (N) vs psi'),pause,
end,

datosdsal_aux{7}{1}=F56_y;

%Resuelvo los solidos 5 (balancin) y 4 (empujador):
%Calculo fuerzas y momentos de inercia:
Min5=-Io1*(-acel_alpha);
Min4=-Ig4*(-acel_beta);

ang=omega_b-alpha-omega_g5;
Fin5_x=-m5*(-Rg5*(-sin(ang).*((-vel_alpha).^2)+cos(ang).*(-acel_alpha)));
Fin5_y=-m5*(-Rg5*(cos(ang).*((-vel_alpha).^2)+sin(ang).*(-acel_alpha)));

Fin4_x=-m4*(db2*cos(alpha).*acel_alpha-
db2*sin(alpha).*(vel_alpha.^2)+lg4*cos(beta).*acel_beta...
   -lg4*sin(beta).*(vel_beta.^2));
Fin4_y=-m4*(-db2*sin(alpha).*acel_alpha-db2*cos(alpha).*(vel_alpha.^2)-
lg4*sin(beta).*acel_beta...
   -lg4*cos(beta).*(vel_beta.^2));
for i=1:N+1,
   for ii=1:2;   %iteraciones para aproximar Mros1,
      if ii==1,
         Mroz_b(i)=0;
      else,
         Mroz_b(i)=mu*Rint_b*sqrt(x(1)^2+x(2)^2)*sign(vel_alpha(i));
      end
      if mu==0,
         ii=2;      %Para que en este caso no itere.
      end
      A=[1 0 1 0 0 0
         0 1 0 1 0 0
         0 0 -db2*cos(alpha(i)) db2*sin(alpha(i)) 0 0
         0 0 -1 0 1 0
         0 0 0 -1 0 1
```

```
         0 0 -lg4*cos(beta(i)) lg4*sin(beta(i)) -(l-lg4)*cos(beta(i)) (l-
lg4)*sin(beta(i))];
      b=[-Fin5_x(i)
         F56_y(i)-Fin5_y(i)
         -F56_y(i)*db1*cos((pi/2)-(omega_b-alpha(i)))-Min5(i)-Mroz_b(i)
         -Fin4_x(i)
         -Fin4_y(i)
         -Min4(i)];
      x=A\b;
   end
   F15_x(i)=x(1); F15_y(i)=x(2); F45_x(i)=x(3); F45_y(i)=x(4);
F34_x(i)=x(5); F34_y(i)=x(6);
end

if ver,
   %Dibujo:
   plot(psi,F15_x),grid,title('F15_x (N) vs psi'),pause,
   plot(psi,F15_y),grid,title('F15_y (N) vs psi'),pause,
   plot(psi,Mroz_b),grid,title('Mroz_b (Nm) (rozamiento) vs psi'),pause,
   plot(psi,F45_x),grid,title('F45_x (N) vs psi'),pause,
   plot(psi,F45_y),grid,title('F45_y (N) vs psi'),pause,
   plot(psi,F34_x),grid,title('F34_x (N) vs psi'),pause,
   plot(psi,F34_y),grid,title('F34_y (N) vs psi'),pause,
end,
datosdsal_aux{7}{2}=F15_x;
datosdsal_aux{7}{3}=F15_y;
datosdsal_aux{7}{4}=sqrt(F15_x.^2+F15_y.^2);
datosdsal_aux{7}{5}=Mroz_b;
datosdsal_aux{7}{6}=F45_x;
datosdsal_aux{7}{7}=F45_y;
datosdsal_aux{7}{8}=sqrt(F45_x.^2+F45_y.^2);
datosdsal_aux{7}{9}=F34_x;
datosdsal_aux{7}{10}=F34_y;
datosdsal_aux{7}{11}=sqrt(F34_x.^2+F34_y.^2);

%Resuelvo el sólido 3 (acoplador):
%Ahora trabajo en ejes xp yp
F43_yp=(-F34_y)*cos(sigma)+(-F34_x)*sin(sigma);
F43_xp=-(-F34_y)*sin(sigma)+(-F34_x)*cos(sigma);
Fin3_yp=-m3*(-acel_ss);

Fin7_yp=-mrod*(-acel_ss);

nxp=(1/rrod)*(-e+ro_r.*cos(theta_r+psi));
nyp=(1/rrod)*(-ro.*sin(theta+psi)+ro_r.*sin(theta_r+psi));
txp=-nyp;
typ=nxp;

%El momento de desequilibrio,sin considerar rozamiento:
Mdeseq=ac*(Fin7_yp+F43_yp+Fin3_yp).*(nxp./nyp);

%Calulo nu_r (ángulo del radio vector con la normal):
dtheta_r=derivtheta(theta_r);
dro_r=deriv(ro_r);
droresptheta_r=dro_r./dtheta_r;
nu_r=atan(-droresptheta_r./ro_r);
```

```
vel_theta_rod=-(1/rrod)*(w.*ro_r.*cos(nu_r)+vel_ss.*sin(theta_r+psi+nu_r-
pi/2));
acel_theta_rod=deriv(vel_theta_rod).*w;
Min7=-Irod.*acel_theta_rod;

Nia=diag(zeros(N+1)); Ndb=Nia; Nda=Nia; Nib=Nia;   %Para que tengan algún
valor
for i=1:N+1,
   if lsup>ss(i)+ac+acs,
      apa=acs;
   else,
      apa=lsup-ac-ss(i);
   end
   if linf<ss(i)+ac-aci,
      apb=aci;
   else,
      apb=ss(i)+ac-linf;
   end
   if Mdeseq(i)>0,
      for ii=1:2,
         if ii==1,
            Mroz_rod(i)=0;
         else,

Mroz_rod(i)=mu*Rint_rod*sqrt(x(1)^2+x(2)^2)*sign(vel_theta_rod(i));
         end
         if mu==0,ii=2; end     %No hay que iterar.
         A=[1 0 -1 1 0 0
            0 1 mu*sign(vel_ss(i)) mu*sign(vel_ss(i)) 0 0
            -ac 0 -apa+mu*sign(vel_ss(i))*ancho/2 -apb-
mu*sign(vel_ss(i))*ancho/2 0 0
            -1 0 0 0 nxp(i) txp(i)
            0 -1 0 0 nyp(i) typ(i)
            0 0 0 0 0 -rrod];
         b=[-F43_xp(i)
            -F43_yp(i)-Fin3_yp(i)
            -Mroz_rod(i)
            0
            -Fin7_yp(i)
            Mroz_rod(i)-Min7(i)];
         x=A\b;
      end
      F73_xp(i)=x(1); F73_yp(i)=x(2); Nia(i)=x(3); Ndb(i)=x(4);
F27_n(i)=x(5); F27_t(i)=x(6);
   else,          %Ahora Mdeseq<=0.
       for ii=1:2,
         if ii==1,
            Mroz_rod(i)=0;
         else,

Mroz_rod(i)=mu*Rint_rod*sqrt(x(1)^2+x(2)^2)*sign(vel_theta_rod(i));
         end
         if mu==0,ii=2; end     %No hay que iterar.
         A=[1 0 1 -1 0 0
            0 1 mu*sign(vel_ss(i)) mu*sign(vel_ss(i)) 0 0
            -ac 0 apa-mu*sign(vel_ss(i))*ancho/2
apb+mu*sign(vel_ss(i))*ancho/2 0 0
```

```
            -1  0  0  0 nxp(i) txp(i)
            0 -1  0  0 nyp(i) typ(i)
            0  0  0  0  0 -rrod];
         b=[-F43_xp(i)
            -F43_yp(i)-Fin3_yp(i)
            -Mroz_rod(i)
            0
            -Fin7_yp(i)
            Mroz_rod(i)-Min7(i)];
         x=A\b;
      end
      F73_xp(i)=x(1); F73_yp(i)=x(2); Nda(i)=x(3); Nib(i)=x(4);
F27_n(i)=x(5); F27_t(i)=x(6);
   end
end


Tia_yp=mu*Nia.*sign(vel_ss');
Tdb_yp=mu*Ndb.*sign(vel_ss');
Tda_yp=mu*Nda.*sign(vel_ss');
Tib_yp=mu*Nib.*sign(vel_ss');

if ver,
   %dibujo:
   plot(psi,F73_xp),grid,title('F73_xp (N) vs psi'),pause,
   plot(psi,F73_yp),grid,title('F73_yp (N) vs psi'),pause,
   plot(psi,F27_n),grid,title('F27_n (N) vs psi'),pause,
   plot(psi,F27_t),grid,title('F27_t (rozamiento) (N) vs psi'),pause,
   plot(psi,Nia),grid,title('Nia (N) vs psi'),pause,
   plot(psi,Tia_yp),grid,title('Tia_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Ndb),grid,title('Ndb (N) vs psi'),pause,
   plot(psi,Tdb_yp),grid,title('Tdb_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Nda),grid,title('Nda (N) vs psi'),pause,
   plot(psi,Tda_yp),grid,title('Tda_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Nib),grid,title('Nib (N) vs psi'),pause,
   plot(psi,Tib_yp),grid,title('Tib_yp (rozamiento) (N) vs psi'),pause,
   plot(psi,Mroz_rod),grid,title('Mroz_rod (Nm) (rozamiento) vs
psi'),pause,
end,

datosdsal_aux{7}{12}=F73_xp;
datosdsal_aux{7}{13}=F73_yp;
datosdsal_aux{7}{14}=sqrt(F73_xp.^2+F73_yp.^2);
datosdsal_aux{7}{15}=F27_n;
datosdsal_aux{7}{16}=F27_t;
datosdsal_aux{7}{17}=Nia;
datosdsal_aux{7}{18}=Tia_yp;
datosdsal_aux{7}{19}=Ndb;
datosdsal_aux{7}{20}=Tdb_yp;
datosdsal_aux{7}{21}=Nda;
datosdsal_aux{7}{22}=Tda_yp;
datosdsal_aux{7}{23}=Nib;
datosdsal_aux{7}{24}=Tib_yp;
datosdsal_aux{7}{25}=Mroz_rod;

%Sólido 2 (leva)
%calculo las fuerzas y el momento de inercia(respecto del eje):
```

```
if size(w_paso)==[1 1],
   acel_psi=0;
else,
   w_pp=w_paso;
   dw_pp=fnder(w_paso);
   acel_psi=ppval(dw_pp,psi).*ppval(w_pp,psi);
end,

dtheta_r=derivtheta(theta_r);
Io2_aux=(masa_p_u_area/4)*ro_r.^4.*dtheta_r.*(2*pi/N);
Io2_aux(N+1)=0;
Io2=-sum(Io2_aux);
Min2=-Io2*acel_psi;

%Calculo el area por las coordenadas x e y del c.d.m,respecto de la leva.
AXG=0; AYG=0;
for i=1:20,
   if i==1,
      i_inf=1;
   else,
      i_inf=round((i-1)*(N/20));
   end
   i_sup=round(i*(N/20));
   theta_r_inf=theta_r(i_inf); theta_r_sup=theta_r(i_sup);
   theta_r_med=(theta_r_inf+theta_r_sup)/2;
   delta_theta_r=theta_r_inf-theta_r_sup;   %Pues theta decrece.
   ro_r_max=(ro_r(i_inf)+ro_r(i_sup))/2;
   for j=1:20,
      ro_r_inf=(j-1)*(ro_r_max/20); ro_r_sup=j*(ro_r_max/20);
      ro_r_med=(ro_r_inf+ro_r_sup)/2;
      delta_ro_r=ro_r_sup-ro_r_inf;

AXG=AXG+ro_r_med*cos(theta_r_med)*ro_r_med*delta_theta_r*delta_ro_r;

AYG=AYG+ro_r_med*sin(theta_r_med)*ro_r_med*delta_theta_r*delta_ro_r;
   end
end
%con minúsculas nos referimos al sistema de referencia de la barra fija
%Con respecto al tipo 5,solo hay que cambiarle el signo,pues los ejes en
un caso y en
%otro son en sentido contrario.
acel_axpg2=AXG*(cos(psi).*w.^2+sin(psi).*acel_psi)+AYG*(-
sin(psi).*w.^2+cos(psi).*acel_psi);
acel_aypg2=-AXG*(-
sin(psi).*w.^2+cos(psi).*acel_psi)+AYG*(cos(psi).*w.^2+sin(psi).*acel_psi
);
Fin2_xp=-masa_p_u_area*acel_axpg2;
Fin2_yp=-masa_p_u_area*acel_aypg2;

F12_xp=F27_n.*nxp+F27_t.*txp-Fin2_xp;
F12_yp=F27_n.*nyp+F27_t.*typ-Fin2_yp;
Mroz_l=-mu*Rint_l*sqrt(F12_xp.^2+F12_yp.^2);

uxp=-ro_r.*cos(theta_r+psi);
uyp=-ro_r.*sin(theta_r+psi);
vxp=-F27_n.*nxp-F27_t.*txp;
```

```
vyp=-F27_n.*nyp-F27_t.*typ;

Meje=-Min2-Mroz_l-(uxp.*vyp-uyp.*vxp);

if ver,
   %Dibujo:
   plot(psi,F12_xp),grid,title('F12_xp (N) vs psi'),pause,
   plot(psi,F12_yp),grid,title('F12_yp (N) vs psi'),pause,
   plot(psi,Mroz_l),grid,title('Mroz_l (Nm) (rozamiento) vs psi'),pause,
   plot(psi,Meje),grid,title('Meje (Nm) vs psi'),pause,
end,

datosdsal_aux{7}{26}=F12_xp;
datosdsal_aux{7}{27}=F12_yp;
datosdsal_aux{7}{28}=sqrt(F12_xp.^2+F12_yp.^2);
datosdsal_aux{7}{29}=Mroz_l;
datosdsal_aux{7}{30}=Meje;

%Calculo el trabajo:
trabajo_aux=Meje*(2*pi/N);
trabajo_aux(N+1)=0;  %El último seria repetitivo.

if ver,
   trabajo=sum(trabajo_aux),
else,
   trabajo=sum(trabajo_aux);
end,
datosdsal_aux{9}=trabajo;



function [s_pp,param_mod]=...
         limita(s_pp,param,prop,param_mod,igmax,igmin,smax,smin,n);



%Ahora modificaremos la curva, si es preciso,para que no sobre
%pase,o iguale,(según el caso),los límites.

%Vemos si algún dato no es coherente con smax,o smin.

%Para evitar errores numéricos, si hay algún punto que esté en el máximo
o mínimo, pero no
%Es exactamente igual a smax y smin (varía menos de 1e-11), modifico smax
o smin para que
%sea exactamente igual, y poder luego compararlos.

if abs(max(param_mod(:,2))-smax)<1e-11,
   smax=max(param_mod(:,2));
end
if abs(min(param_mod(:,2))-smin)<1e-11,
   smin=min(param_mod(:,2));
end
```

```
 for i=1:n,
    if prop(i,2)==1,   %es un dato.

       if (param_mod(i,2)>smax)|((param_mod(i,2)==smax)& igmax==0),
          %el dato se sale de [smin,smax].
          disp('psi => s o fi'),disp(param_mod(i,1)),
disp(param_mod(i,2)),
          disp('este dato no cumple las restricciones de posión'),
          s_pp=NaN;return,
       end

       if (param_mod(i,2)==smax)&(igmax==1),   %Si además fuera máximo,se
permitiria
          maximo=0;      %inicialmente suponemos que no lo es
          if (param_mod(i,3)==0)|(prop(i,3)~=1),
             %La s'=0,la cumple,o la puede cumplir.
             %Si s''<0 hay un máximo,si s''=0,puede ser que la curva
suba,pero
             %luego bajará,y se pondrá otro punto con s=smax,s'y s''=0,lo
que es
             %admisible.
             %Si s''>0,pero se puede cambiar,se cambia
             if (param_mod(i,4)<=0)|(prop(i,4)~=1),
                param_mod(i,3)=0; prop(i,3)=1; %Es necesario que s'=0,lo
fijo.
                if param_mod(i,4)>0 , param_mod(i,4)=0;end
                maximo=1;
             end
          end
          if maximo==0,
             disp('en psi => s o fi'),disp(param_mod(i,1)
),disp(param_mod(i,2)),
             disp('con la curva tocando a smax,sólo se permite un
máximo'),
             s_pp=NaN;return,
          end
       end


       if (param_mod(i,2)<smin)|((param_mod(i,2)==smin)& igmin==0),
          disp('psi => s o fi'),disp(param_mod(i,1)),
disp(param_mod(i,2)),
          disp('holaeste dato no cumple las restricciones de posición'),
          s_pp=NaN;return,
       end

       if (param_mod(i,2)==smin)&(igmin==1),   %Si además fuera mínimo,se
permitiria
          minimo=0;      %inicialmente suponemos que no lo es
          if (param_mod(i,3)==0)|(prop(i,3)~=1),
             %La s'=0,la cumple,o la puede cumplir.
             %Si s''>0 hay un mínimo,si s''=0,puede ser que la curva
baje,pero
             %luego subirá,y se pondrá otro punto con s=smin,s'y s''=0,lo
que es
             %admisible.
```

```
            %Si s''<0,pero se puede cambiar,se cambia
            if (param_mod(i,4)>=0)|(prop(i,4)~=1),
               param_mod(i,3)=0; prop(i,3)=1; %Es necesario que s'=0,lo
fijo
               if param_mod(i,4)<0 , param_mod(i,4)=0;end
               minimo=1;
            end
         end
         if minimo==0,
            disp('en psi => s o fi'),disp(param_mod(i,1)
),disp(param_mod(i,2)),
            disp('con la curva tocando a smin,sólo se permite un
mínimo'),
            s_pp=NaN;return,
         end
      end

   end
 end


 %Modificamos los puntos de control para que esten dentro de la banda.

 for i=1:n,
   if prop(i,2)~=1,  %No es dato.
      if param_mod(i,2)>=smax,  %Puede que se salga por arriba.
         %Se actua si no cumple las restricciones.
         if param_mod(i,2)>smax , %Habrá que modificarlo seguro
            param_mod(i,2)=smin+.99*(smax-smin);
         end
         if param_mod(i,2)==smax,
            if igmax==0,
               param_mod(i,2)=smin+.99*(smax-smin);   %pues no puede ser
=.
            else,    %se pide que sea igual.
               if (param_mod(i,3)==0) & (param_mod(i,4)<=0),
                  %se trata de un máximo,lo dejo como esta
                  %o,si no es máximo,luego se pondrá otro punto y
                  %formarán una linea horizontal en smax.
               else,
                  param_mod(i,2)=smin+.99*(smax-smin);
               end
            end
         end
      end
      if param_mod(i,2)<=smin,   %Puede que se salga por abajo.
         %Se actua si no cumple las restricciones.
         if param_mod(i,2)<smin , %Habrá que modificarlo seguro
            param_mod(i,2)=smin+.01*(smax-smin);
         end
         if param_mod(i,2)==smin,
            if igmin==0,
               param_mod(i,2)=smin+.01*(smax-smin);   %pues no puede ser
=.
            else,    %se pide que sea igual.
               if (param_mod(i,3)==0) & (param_mod(i,4)>=0),
                  %se trata de un mínimo,lo dejo como está.
```

```
                    %o,si no es mínimo,luego se pondrá otro punto y
                    %formarán una linea horizontal en smin.
                 else,
                    param_mod(i,2)=smin+.01*(smax-smin);
                 end
              end
           end
        end
     end
  end


  %modifico s_pp:
  s_pp=construyepp(param_mod);
  psi=0:2*pi/1000:2*pi;  %Para dibujar,por si ha camiado algún nodo
  if 0,        %Para q no lo dibuje.Si quieres pones un 1.

plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
     pause,
  end


%Si igmax=1 o igmin=1,debe alcanzarse el extremo en algún punto.Si el
usuario no lo da
%,se buscará el máximo,y se pondrá un nodo en ese punto,con
s=smax,s'=0,s''=0.En caso de
%que el máximo anterior coincidera con un nodo,y no se pueda cambiar,se
pondrá un nuevo
%nodo con s=smax,s'=s''=0,en el punto medio entre el nodo dato,y el
siguiente nodo dato
%de mayor valor de s.Análogamente se actuará en smin.

[psi_max,psi_min]=extremospp(s_pp);
if igmax==1;
   if (smax-ppval(s_pp,psi_max))>1.e-6,  %dejo tolerancia por errores
numéricos.
      %en este caso el máximo no se alcanza.
      if min(abs(param_mod(:,1)-psi_max))>1.e-6, %Considero que psi_max
no es nodo
         %Introduzco un nodo
         p=1;
         while (param_mod(p,1)<psi_max)   %psi_max está a la izquierda.
            %Todavia no es ese el hueco de psi_max.
            p=p+1;    %Avanza
         end
         %param_mod(p,1) esta inmediatamente a la derecha de psi_max.
         for i=size(param_mod(:,1)):-1:p,        %va para atrás.
            param_mod(i+1,:)=param_mod(i,:);
            prop(i+1,:)=prop(i,:);
         end
         param_mod(p,:)=[psi_max,smax,0,0];    %Nodo introducido,si
igmax=1;
         prop(p,:)=[1 1 1 1];
      else,  %el máximo se ha producido en un nodo ya existente.Lo
identificamos.
```

```
         s_psi_max=param_mod(1,2);imax=1;psi_max=param_mod(1,1);
%inicializo
         for i=1:length(param_mod(:,2)),
            if param_mod(i,2)>s_psi_max;
               s_psi_max=param_mod(i,2);
               imax=i;
               psi_max=param_mod(i,1);
            end
         end
         %Vemos si es posible subir ese nodo.
         if (prop(imax,2)~=1)    %Ya cumple s'=0,s''<=0,pues es máximo.
            param_mod(imax,2)=smax;  %modifico el nodo.
            %En caso de que sea un extremo hay que hacer periódica la
curva.
            if (imax==1) | (imax==length(param_mod(:,1))),

param_mod(1,2)=smax;param_mod(length(param_mod(:,1)),2)=smax;
            end

         else,    %No es posible subir el nodo
            if (imax~=1) & (imax~=length(param_mod(:,1))),  %es un nodo
interior
               if param_mod(imax-1,2)>param_mod(imax+1,2),
                  imax_prox=imax-1;
               else,
                  imax_prox=imax+1;
               end
               %Introduzco el nodo.
               psi_max=(param_mod(imax,1)+param_mod(imax_prox,1))/2;
            else,           %el nodo está en el extremo
               n=length(param_mod(:,2));
               if param_mod(2,2)>param_mod(n-1,2),
                  psi_max=(param_mod(2,1)+param_mod(1,1))/2;
               else
                  psi_max=(param_mod(n-1,1)+param_mod(n,1))/2;
               end
            end

            p=1;
            while (param_mod(p,1)<psi_max)   %psi_max está a la
izquierda.
               %Todavia no es ese el hueco de psi_max.
               p=p+1;    %Avanza
            end
            %param_mod(p,1) esta inmediatamente a la derecha de psi_max.
            for i=size(param_mod(:,1)):-1:p,        %va para atrás.
               param_mod(i+1,:)=param_mod(i,:);
               prop(i+1,:)=prop(i,:);
            end
            param_mod(p,:)=[psi_max,smax,0,0];    %Nodo introducido,si
igmax=1;
            prop(p,:)=[1 1 1 1];
         end
      end
      s_pp=construyepp(param_mod);
      if 0,    %Para que no lo dibuje.Pones un 1 si quieres
```

```
plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
      end
   end
end

%Analogamente,si igmin==1:
[psi_max,psi_min]=extremospp(s_pp);
if igmin==1;
   if (ppval(s_pp,psi_min)-smin)>1.e-6,  %dejo tolerancia por errores
numéricos.
      %en este caso el mínimo no se alcanza.
      if min(abs(param_mod(:,1)-psi_min))>1.e-6, %Considero que psi_min
no es nodo
         %Introduzco un nodo
         p=1;
         while (param_mod(p,1)<psi_min)   %psi_min está a la izquierda.
            %Todavia no es ese el hueco de psi_min.
            p=p+1;    %Avanza
         end
         %param_mod(p,1) esta inmediatamente a la derecha de psi_min.
         for i=size(param_mod(:,1)):-1:p,        %va para atrás.
            param_mod(i+1,:)=param_mod(i,:);
            prop(i+1,:)=prop(i,:);
         end
         param_mod(p,:)=[psi_min,smin,0,0];    %Nodo introducido,si
igmin=1;
         prop(p,:)=[1 1 1 1];
      else,  %el mínimo se ha producido en un nodo ya existente.Lo
identificamos.
         s_psi_min=param_mod(1,2);imin=1;psi_min=param_mod(1,1);
%inicializo
         for i=1:length(param_mod(:,2)),
            if param_mod(i,2)<s_psi_min;
               s_psi_min=param_mod(i,2);
               imin=i;
               psi_min=param_mod(i,1);
            end
         end
         %Vemos si es posible bajar ese nodo.
         if (prop(imin,2)~=1)    %Ya cumple s'=0,s''>=0,pues es mínimo.
            param_mod(imin,2)=smin;  %modifico el nodo.
            %En caso de que sea un extremo hay que hacer periódica la
curva.
            if (imin==1) | (imin==length(param_mod(:,1))),

param_mod(1,2)=smin;param_mod(length(param_mod(:,1)),2)=smin;
            end
         else,    %No es posible bajar el nodo
            if (imin~=1) & (imin~=length(param_mod(:,1))),  %es un nodo
interior
               if param_mod(imin-1,2)<param_mod(imin+1,2),
                  imin_prox=imin-1;
               else,
                  imin_prox=imin+1;
               end
```

```
            %Introduzco el nodo.
            psi_min=(param_mod(imin,1)+param_mod(imin_prox,1))/2;
         else,           %el nodo está en el extremo
            n=length(param_mod(:,2));
            if param_mod(2,2)<param_mod(n-1,2),
               psi_min=(param_mod(2,1)+param_mod(1,1))/2;
            else
               psi_min=(param_mod(n-1,1)+param_mod(n,1))/2;
            end
         end

         p=1;
         while (param_mod(p,1)<psi_min)   %psi_min está a la
izquierda.
            %Todavia no es ese el hueco de psi_min.
            p=p+1;    %Avanza
         end
         %param_mod(p,1) esta inmediatamente a la derecha de psi_min.
         for i=size(param_mod(:,1)):-1:p,        %va para atrás.
            param_mod(i+1,:)=param_mod(i,:);
            prop(i+1,:)=prop(i,:);
         end
         param_mod(p,:)=[psi_min,smin,0,0];    %Nodo introducido,si
igmax=1;
         prop(p,:)=[1 1 1 1];
      end
   end
   s_pp=construyepp(param_mod);
   if 0,    %Para que no lo dibuje.pones un 1 si quieres

plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
   end
 end
end



 %Ahora busco los máximos y los mínimos,y si se salen de los limites,
 %en ese punto psi introduzco un nodo con s=smax,o smin; si se pide
 %que alcance los extremos, o un poco antes de llegar a los extremos,sino
 %se pide; s'=0, s''=0.
 %Si no hubiera puntos en los extremos,y se pide que se llegue,habrá que
 %introducir nodos de la misma forma que antes.
 itera=1;
 while itera,

   %Vemos primero si se pasa por arriba.
   [psi_max,psi_min]=extremospp(s_pp);
   if ((ppval(s_pp,psi_max)-smax)>1.e-
6)|((igmax==0)&((ppval(s_pp,psi_max)-smax)>=0)),
      p=1;
      while (param_mod(p,1)<psi_max)   %psi_max está a la izquierda.
         %Todavia no es ese el hueco de psi_max.
         p=p+1;    %Avanza
      end
      %param_mod(p,1) esta inmediatamente a la derecha de psi_max.
```

```
        for i=size(param_mod(:,1)):-1:p,        %va para atrás.
           param_mod(i+1,:)=param_mod(i,:);
        end
        param_mod(p,:)=[psi_max,smax,0,0];    %Nodo introducido,si
igmax=1;
        if igmax==0,                          %Si igmax=0
           param_mod(p,2)=smin+.99*(smax-smin);
        end
        s_pp=construyepp(param_mod);
        if 0,    %Para que no lo dibuje.Pones un 1 si quieres

plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
        end,

   end

   %Vemos ahora si se pasa por abajo.
   [psi_max,psi_min]=extremospp(s_pp);
   if ((smin-ppval(s_pp,psi_min))>1.e-6)|((igmin==0)&((smin-
ppval(s_pp,psi_min))>=0)),
        p=1;
        while (param_mod(p,1)<psi_min)   %psi_min está a la izquierda.
           %Todavia no es ese el hueco de psi_min.
           p=p+1;    %Avanza
        end
        %param_mod(p,1) esta inmediatamente a la derecha de psi_min.
        for i=size(param_mod(:,1)):-1:p,        %va para atrás.
           param_mod(i+1,:)=param_mod(i,:);
        end
        param_mod(p,:)=[psi_min,smin,0,0];    %Nodo introducido,si
igmax=1;
        if igmin==0,                          %Si igmax=0
           param_mod(p,2)=smin+.01*(smax-smin);
        end
        s_pp=construyepp(param_mod);
        if 0,    %Para que no lo dibuje.Pones un 1 si quieres

plot(psi,ppval(s_pp,psi),param_mod(:,1),param_mod(:,2),'*',psi,smax,'r',p
si,smin,'r'),
        end

   end


   %Veamos ahora si no es necesario seguir iterando
   [psi_max,psi_min]=extremospp(s_pp);
   if ((ppval(s_pp,psi_max)-smax)>1.e-
6)|((igmax==0)&((ppval(s_pp,psi_max)-smax)>=0))|...
         ((smin-ppval(s_pp,psi_min))>1.e-6)|((igmin==0)&((smin-
ppval(s_pp,psi_min))>=0)),
      itera=1;
   else
      itera=0;
   end
end
```

```
function matop=matrizoptimizada(param1,param2,RPM,maxim,minim)

%La manera de llamar a esta función es escribiendo en la linea de
comando:
%MATOP=matrizoptimizada(PARAM1,PARAM2,RPM,MAXIM,MINIM)
%Donde MATOP es una matriz óptima que es una matriz organizada igual que
la matriz con la
%que se dan los datos de ÁNGULO POSICICIÓN VELOCIDAD y ACELERACIÓN en el
programa
%intent1. Esta matriz puede grabarse en un fichero con el nombre: PUNTOS
(nombre de la matriz)
%y luego darle el nombre del fichero al programa intent1 para que la use.
%PARAM1 contiene en cada elemento una cota inferior para el valor que ese
elemento
%pueda alcanzar en matop.
%PARAM2 contiene en cada elemento una cota superior para el valor que ese
elemento
%pueda alcanzar en matop.
%Tanto en PARAM1 como en PARAM2 deben estar contruidas con valores
numéricos.
%RPM son las revoluciones, bien como un escalar o bien como una matriz si
varían con PSI,
%de la mismaforma que se explica en intent1.
%MAXIM es una cota máxima para el valor que pueda alcanzar en cualquier
ángulo la variable
%que representa la POSCIÓN.
%MINIM es una cota mínima para el valor que pueda alcanzar en cualquier
ángulo la variable
%que representa la POSICIÓN.
%Calcularemos una matriz de entrada óptima , a partir de las matrices de
entrada límite.
%La forma de optimizar es explorando puntos aleatorios, y almacenando los
mejores. Luego se
%busca un mínimo local partiendo de cada punto almacenado, quedandonos
después con el mejor.
%En la otimización se pueden tener en cuenta la aceleración, la
sobreaceleración, y el radio
%de curvatura mínimo, para evitar cúspides.
%Se pueden controlar las iteraciones del programa con los parámetros
ITER1 e ITER2, y
%también la función objetivo con los parámetros PONDERA1 Y PONDERA2.Estos
parámetros están
%al principio del programa y ese pueden cambiar.A continuación se
presentan valores posibles,
%y una explicación de cada parámetro:
%ITER1=100000;  %Número de puntos aleatorios que se exploran
%ITER2=100;   %De los puntos anteriores, en los ITER2 mejores, se busca
un mínimo local.
```

```
%PONDERA1=30; %Además de minimizar la aceleración también se minimiza la
sobreaceleración.
%en la fución objetivo, la aceleración cuenta PONDERA1 veces la
sobreaceleración.Si no
%importa la sobreaceleración pongase un número alto.
%PONDERA2=30; %Si vale 0, no se máximiza el mínimo radio de curvatura
(para evitar cúspides),
%sólo se minimiza la aceleración.De no valer 0, es el número de veces más
grande que es
%en la fución objetivo el término para el radio de curvatura, comparado
con el término para
%minimizar la aceleración+sobreaceleración. Si se quieren evitar las
cúspides debe ser
%un número alto.
%Una buena estrategia es empezar con pocos puntos y un rango amplio de
variabilidad, y a
%medida que va mejorando la solución darle más puntos y con un rango de
variabilidad más
%estrecho.



ITER1=150000;  %Número de puntos aleatorios que se exploran
ITER2=75;   %De los puntos anteriores, en los ITER2 mejores, se busca un
mínimo local.

PONDERA1=15; %Además de minimizar la aceleración también se minimiza la
sobreaceleración.
%en la fución objetivo, la aceleración cuenta PONDERA1 veces la
sobreaceleración.Si no
%importa la sobreaceleración pongase un número alto.

PONDERA2=60; %Si vale 0, no se máximiza el mínimo radio de curvatura
(para evitar cúspides),
%sólo se minimiza la aceleración.De no valer 0, es el número de veces más
grande que es
%en la fución objetivo el término para el radio de curvatura, comparado
con el término para
%minimizar la aceleración+sobreaceleración. Si se quieren evitar las
cúspides debe ser
%un número alto.


param1(:,4)=param1(:,4)/1000;     %Escalo la aceleración para que
funcione mejor fims
param2(:,4)=param2(:,4)/1000;

if ~all(param2(:)>=param1(:)),
   error('la segunda matriz debe contener cotas superiores, y la primera
inferiores'),
end
if max(param1(:,2))>maxim | min(param1(:,2))<minim,
   error('los puntos deben estar entre el valor mínimo y el máximo'),
end
if max(param2(:,2))>maxim | min(param2(:,2))<minim,
   error('los puntos deben estar entre el valor mínimo y el máximo'),
end
```

```
param1(:,1)=param1(:,1)*(2*pi/360);    %Lo paso a radianes.
param2(:,1)=param2(:,1)*(2*pi/360);    %Lo paso a radianes.

if size(RPM)~=[1 1],
   RPM(:,1)=RPM(:,1)*(2*pi/360);           %Lo paso a radianes.
end


%La velocidad angular:
%wpas es un escalar, o un polinomio a trozos.
if size(RPM)==[1 1],
   w=RPM*2*pi/60;   %si es un escalar.
   w_pas=w;
else
   w=[RPM(:,1),RPM(:,2)*(2*pi/60)]; %en radianes/segundo.
   m=size(w);
   m=m(1);
   if abs(w(1,2)-w(m,2))<1.e-12,     %si son iguales,un spline periódico
      w_pp=spline_per(w(:,1),w(:,2));
   else,
      w_pp=spline(w(:,1),w(:,2));    %si no,un spline normal
   end
   w_pas=w_pp;
end


%Trabajamos ahora con vectores, para la función fmins, pasándole
solamente las componentes
%que varían.
%Tengase en cuenta que contruye el vector por columnas.
vect1=param1(:);
vect2=param2(:);


%Buscamos una buena solución inicial, para usar fmin.Para ello variamos
aleatoriamente
%las componentes variables, dentro de su intervalo.
valor_mejorbruto_5=1e150*ones(ITER2,1);

for k=1:ITER1,
   if rem(k,100)==0,k,end,       %Para ver por donde va.
   j=1;
   for i=1:length(vect1),
      if vect1(i)~=vect2(i),
         if k==500,      %Una vez pruebo en la media, y con aceleración
cero.
            vect_prue(j)=(vect1(i)+vect2(i))/2;
            if abs(vect1(i))>100 | abs(vect2(i))>100, vect_prue(j)=0;end
%es una aceleración
            j=j+1;
         else,
            vect_prue(j)=vect1(i)+rand(1)*(vect2(i)-vect1(i));
            j=j+1;
         end
      end
   end
```

```
valor_prue=evaluamatriz(vect_prue,vect1,vect2,w_pas,maxim,minim,PONDERA1,
PONDERA2);

   if valor_prue<max(valor_mejorbruto_5),    %Entra entre los 5 primeros
      [v,i]=max(valor_mejorbruto_5);
      valor_mejorbruto_5(i)=valor_prue;         %Sustituyo el peor por el
nuevo.
      vect0_5(:,i)=vect_prue';
   end
end


options=foptions;
options(14)=1000*length(vect0_5(:,1));
options(2)=1e6;

valor_mejorfino=1e151;
for i=1:length(valor_mejorbruto_5),
   i,

vect_5=fmins('evaluamatriz',vect0_5(:,i),options,'',vect1,vect2,w_pas,max
im,minim,PONDERA1,PONDERA2),

valor_5=evaluamatriz(vect_5,vect1,vect2,w_pas,maxim,minim,PONDERA1,PONDER
A2),
   if valor_5<valor_mejorfino,  %Es el mejor
      vect=vect_5;
      valor_mejorfino=valor_5;
   end,
end,
vect,
valor_mejorfino,
debeserigualalanterior=evaluamatriz(vect,vect1,vect2,w_pas,maxim,minim,PO
NDERA1,PONDERA2),


%Construyo la matriz:
matop=param1;      %para que queden grabados los valores constantes,
p=1;
for j=1:4,
   for i=1:length(vect1)/4,
      if param1(i,j)~=param2(i,j),   %Distintos límites, es un parámetro
variable.
         matop(i,j)=vect(p);
         p=p+1;                 %Avanzo un puesto en el vector de
variables.
      end
   end
end

matop(:,1)=matop(:,1)*360/(2*pi);     %en grados
matop(:,4)=matop(:,4)*1000;           %lo reescalo
```

```
function [ro_r,theta_r]=prim_a_r(ro,theta,beta,rrod)
%Ahora calculemos el perfil real (hasta ahora hemos visto el primitivo),
%para ello r_real=r_primitivo-rrod*en, siendo en el vector
normal,unitario,
%exterior a la curva.
%Trabajamos con complejos.

r_p=ro.*cos(theta)+(ro.*sin(theta))*i;
e_n=(r_p./ro).*(cos(beta)+sin(beta)*i);   %giro el vector unitario,para
que sea normal.
r_r=r_p-rrod*e_n;
ro_r=abs(r_r);
theta_r=angle(r_r);        %este valor habrá que hacerlo continuo.
for i=2:length(ro_r),
   if (theta_r(i)-theta_r(i-1))>pi/4,    %hay un salto.
      j=1;
      while((theta_r(i)+j*2*pi-theta_r(i-1))>pi/4) & ((theta_r(i)-j*2*pi-
theta_r(i-1))>pi/4),
         j=j+1;
      end
      if (theta_r(i)+j*2*pi-theta_r(i-1))<pi/4,
         theta_r(i)=theta_r(i)+j*2*pi;
      else
         theta_r(i)=theta_r(i)-j*2*pi;
      end
   end
end
```

```
function pp=spline_per(x,y)

%Se trata de construir un spline periódico.Hay n+1 nodos,y n polinomios.
%Las ecuaciones las tomo de guion de la asignatura: Ampliación de
Matemáticas.
%Polinomios: a(j)+b(j)*(x-x(j))+c(j)*(x-x(j))^2+d(j)*(x-x(j))^3.

%Veo si hay posibles errores.
if abs(y(1)-y(length(y)))>1.e-11,error('en los extremos debe valer
igual');end
y(length(y))=y(1);      %Por si habia decimales que no era =.
p=size(x);
if min(p)~=1,error('x debe ser un vector');end
if p(1)<p(2),x=x';end
p=size(y);
if min(p)~=1,error('x debe ser un vector');end
if p(1)<p(2),y=y';end
if length(x)~=length(y),error('deben ser de las mismas dimensiones');end
for i=1:(length(x)-1),
   if x(i)>=x(i+1),error('x debe ser monótono');end
end
```

```
%Si se dan sólo tres nodos,el algoritmo no funciona bien,asi que en ese
caso
%se pondrá otro nodo,simétrico respecto del centro.
if length(x)==3,
   eje=(x(1)+x(3))/2;    %eje de simetria.
   dist=x(2)-eje;
   if dist==0,dist=1.e-6;end   %Si el punto esta en el eje
   x(4)=x(3);     %desplazo.
   aux=eje-dist;
   if x(2)>aux,
      x(3)=x(2);x(2)=aux;
   else,
      x(3)=aux;
   end
   y(4)=y(3);y(3)=y(2);
end


h=diff(x);
n=length(x)-1;  %n= numero de polinomios.n+1=numero de nodos.
a=y;            %el calculo de los a(j) es inmediato.a(n+1) se define
asi.

%Construyo la matriz A.
A=zeros(n+1);
A(1,1)=2*(h(1)+h(n));  A(1,2)=h(1);  A(1,n)=h(n);
A(n+1,1)=2*h(n); A(n+1,n+1)=-2*h(n);
for i=2:n,
   A(i,i-1)=h(i-1);
   A(i,i)=2*(h(i-1)+h(i));
   A(i,i+1)=h(i);
end
%Construyo el vector de incógnitas: I.
I(1)=(3/h(1))*(a(2)-a(1))-(3/h(n))*(a(n+1)-a(n));
for i=2:n,
   I(i)=(3/h(i))*(a(i+1)-a(i))-(3/h(i-1))*(a(i)-a(i-1));
end
I(n+1)=0;
I=I';   %Para tener un vector columna.

%Resuelvo:
c=A\I;

%Calculo b(j):
for j=1:n,
   b(j)=(1/h(j))*(a(j+1)-a(j))-(h(j)/3)*(2*c(j)+c(j+1));
end
b=b';

%Calculo d(j):
for j=1:n,
   d(j)=(c(j+1)-c(j))/(3*h(j));
end
d=d';
coef=[d(1:n),c(1:n),b(1:n),a(1:n)];  %Matriz de coeficientes.
pp=mkpp(x,coef);
```

```
if 0,            %Para que no lo haga.Si quieres ver la derivadas pones
1.
   disp('el spline periódico construido:'),
   disp('las derivadas en los extremos son:'),
   d_pp=fnder(pp);
   dd_pp=fnder(d_pp);
   ppval(d_pp,x(1)),
   ppval(d_pp,x(length(x))),
   ppval(dd_pp,x(1)),
   ppval(dd_pp,x(length(x))),
   fnplt(pp);pause,
end
```

Nota: las funciones que se usan para optimizar son:'matrizoptimizada', y 'evaluamatriz'.