Filtro de Kalman, ejemplo Matlab

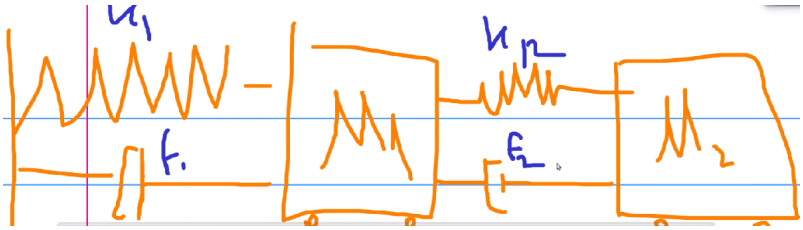
(c) 2022 Antonio Sala, Universitat Politècnica de València. Todos los derechos reservados.

Presentación en vídeo en: https://youtu.be/-zSwRWInYFo

Este código ejecutó sin errores en Matlab **R2022b**

Tabla de Contenidos

Modelo de proceso



Sistema con 2 masas y 2 muelles

```
k1=3;f1=.1;k12=6;f12=3; M2=1; M1=1; %parámetros físicos

%representación interna (tiempo contínuo)
A=[0    1    0    0; ...
  -(k1+k12)/M1 -(f1+f12)/M1 k12/M1 f12/M1; ...
   0    0    0    1;...
    k12/M2 f12/M2 -k12/M2 -f12/M2];
%entradas manipuladas:
Bu=[0; 0; 0;1/M2];
%modelo ruido de proceso:
Bpert=[0 0;1/M1 0;0 0;0 1/M2]; %son fuerzas también, sobre 2 masas
C=[1 0 0 0;0 0 1 0];%queremos "simular" las dos posiciones luego.
sistema_masa_muelles=ss(A,[Bu Bpert],C,0);
sistema_masa_muelles.InputName={'FManipulada','FRuidoMasa1','FRuidoMasa2'};
size(sistema_masa_muelles)
```

```
State-space model with 2 outputs, 3 inputs, and 4 states.
```

```
%polos del sistema
eig(A)
```

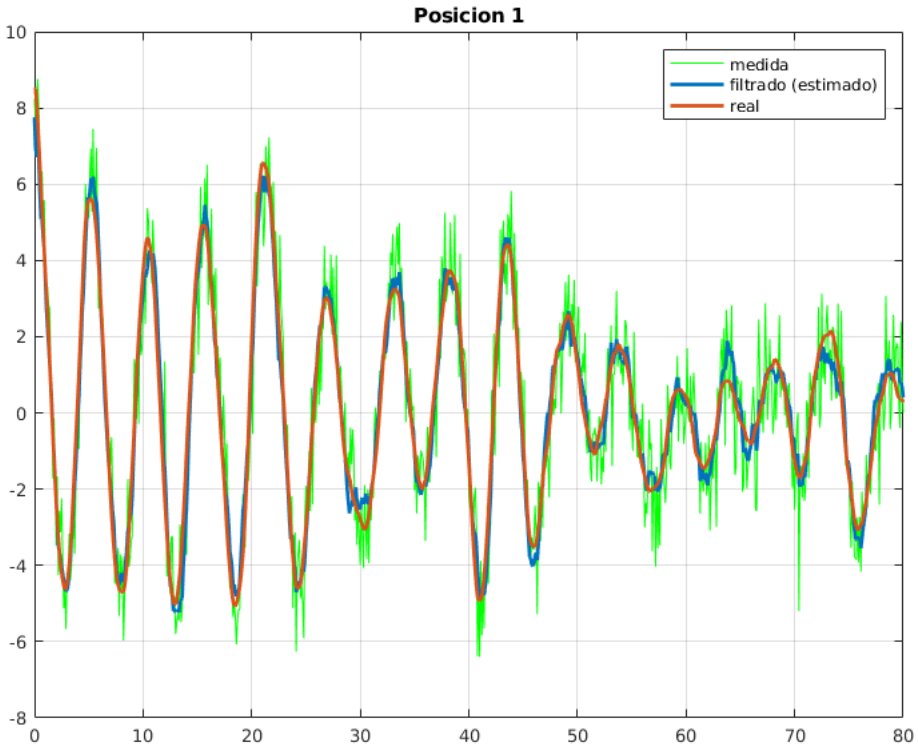
```
ans =
  -2.9942 + 2.0763i
  -2.9942 - 2.0763i
  -0.0558 + 1.1631i
  -0.0558 - 1.1631i
```

```
%discretización
PeriodoMuestreo=.1;
muellesd=c2d(sistema_masa_muelles,PeriodoMuestreo,'zoh'); %perturbaciones no es realmen
Bdiscruido=muellesd.b(:,2:3); BdiscF=muellesd.b(:,1);
```

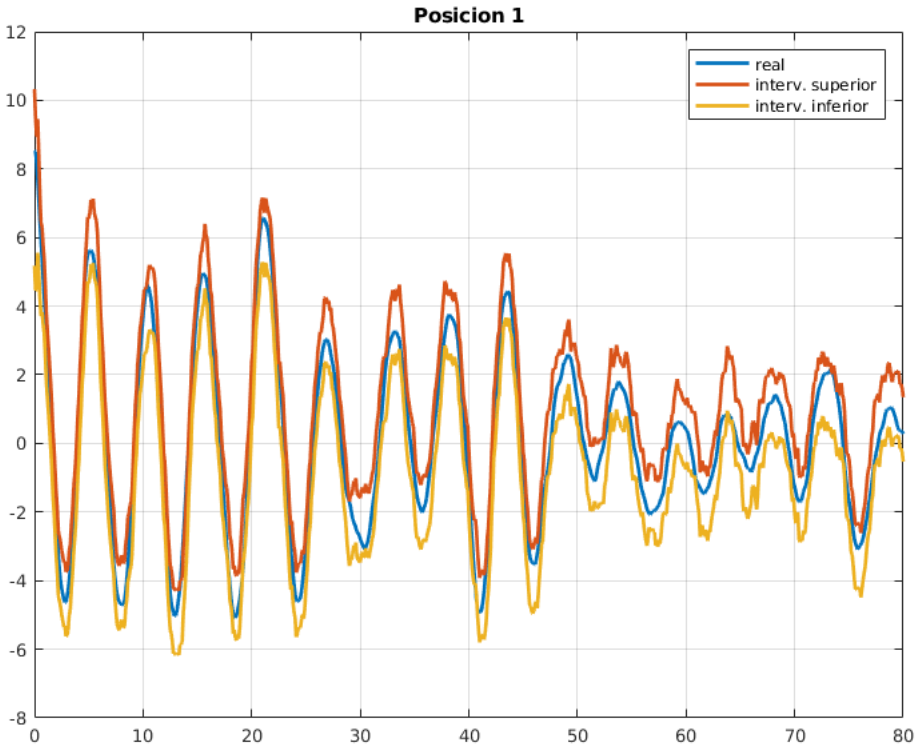
Filtro de Kalman no estacionario

```
PeriodosSimulacion=800;
Tiempos=PeriodoMuestreo*(0:PeriodosSimulacion);


FuerzaEntradaManipulada=1.5*sin(Tiempos*.4)';

%simulación ruido de proceso aleatorio
desvtipruidoproceso=diag([7 2]);
EntradasPerturbacion=randn(length(Tiempos),2)*desvtipruidoproceso;
VCRuidoProc=desvtipruidoproceso^2;

Entradas=[FuerzaEntradaManipulada  EntradasPerturbacion];



%Csens=[1 0 0 0]; %medimos posición 1.
%selección de sensores, compararemos con
%Csens=[0 0 4 0]; %medimos posición 2 con mejor sensor
Csens=[1 0 0 0;0 0 1 0];%medimos ambas con igual sensor

n_sensores=size(Csens,1);
desvtipruidomedida=1;
VCRuidoMed=desvtipruidomedida^2*eye(n_sensores);


%inicializamos variables para guardar gráficas...
grintcf=zeros(length(Tiempos),2);
ysr=zeros(length(Tiempos),2);
y=zeros(length(Tiempos),2);
grfymed=zeros(length(Tiempos),2);

%condicion inicial "real" t=0
Xreal=[8 9 2 3]';
%condicion inicial observador
Xestimado=[0 0 0 0]';
VCEstado=80^2*eye(4);%estimado inicial muy incierto

%bucle de simulación

for k=0:PeriodosSimulacion
    %leer sensores ruidosos
```

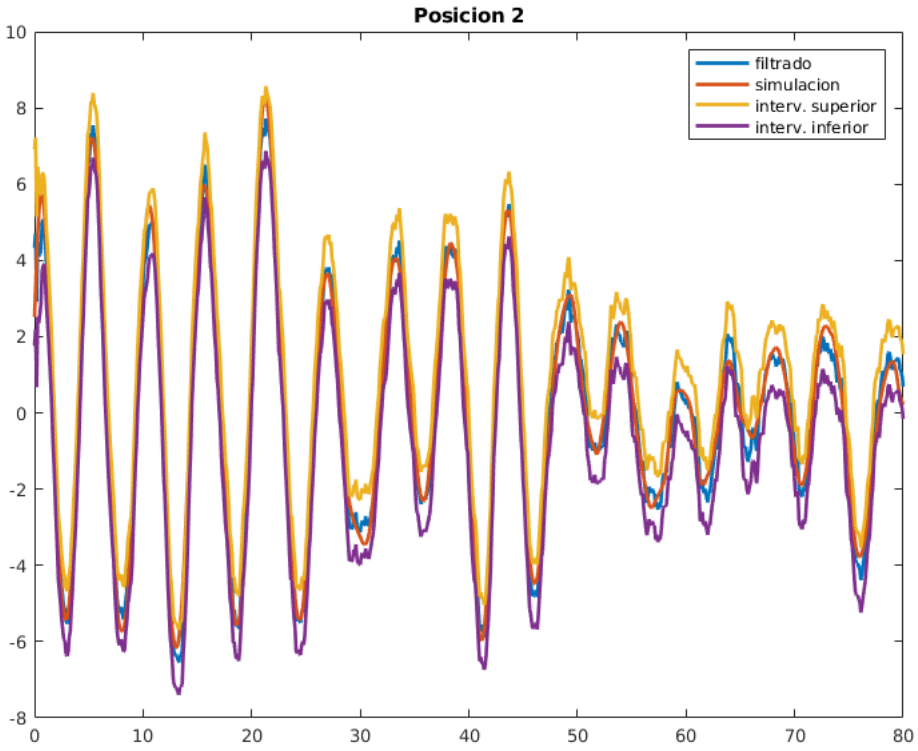
```
    ymed=Csens*Xreal+randn(n_sensores,1)*desvtipruidomedida;
    grfymed(k+1,:)=ymed;


    %¡PARTE IMPORTANTE, código del observador!
    %predictor
    VCEstado_sinmedir=muellesd.A*VCEstado*muellesd.A'+Bdiscruido*VCRuidoProc*Bdiscruido
    %corrector:
    GananciaKalman=VCEstado_sinmedir*Csens'*inv(Csens*VCEstado_sinmedir*Csens'+VCRuidoM
    Xestimado=Xestimado+GananciaKalman*(ymed-Csens*Xestimado);
    VCEstado=VCEstado_sinmedir-GananciaKalman*Csens*VCEstado_sinmedir;
    %hacemos simétrica si no lo es por redondeo:
    VCEstado=(VCEstado+VCEstado')/2;%esto es relevante en precisión finita... hay otras
    %simulamos estado estimado para el próximo muestreo:
    Xestimado=muellesd.A*Xestimado+BdiscF*FuerzaEntradaManipulada(k+1,:)'; %manipuladas

    %proceso real: escribir actuadores y esperar un período de muestreo
    Xreal=muellesd.A*Xreal+muellesd.b*Entradas(k+1,:)'; %manipuladas y perturbación, cl


    %para la gráfica: estimado de las dos posiciones, e intervalos de confianza...
    VCSalida=C*VCEstado*C';%de las DOS posiciones
    grintcf(1+k,:)=2.57583*sqrt(diag(VCSalida));%intervalo confianza... 99% http://math
    ysr(1+k,:)=C*Xestimado;
    y(1+k,:)=C*Xreal;%estas son las posiciones del proceso real, inaccesibles para nosc
end

ysup=ysr+grintcf;
yinf=ysr-grintcf;
plot(Tiempos,grfymed(:,1),'g')
hold on
plot(Tiempos,[ysr(:,1) y(:,1) ],'LineWidth',2)
hold off
title('Posicion 1')
legend('medida','filtrado (estimado)','real');
grid on
```



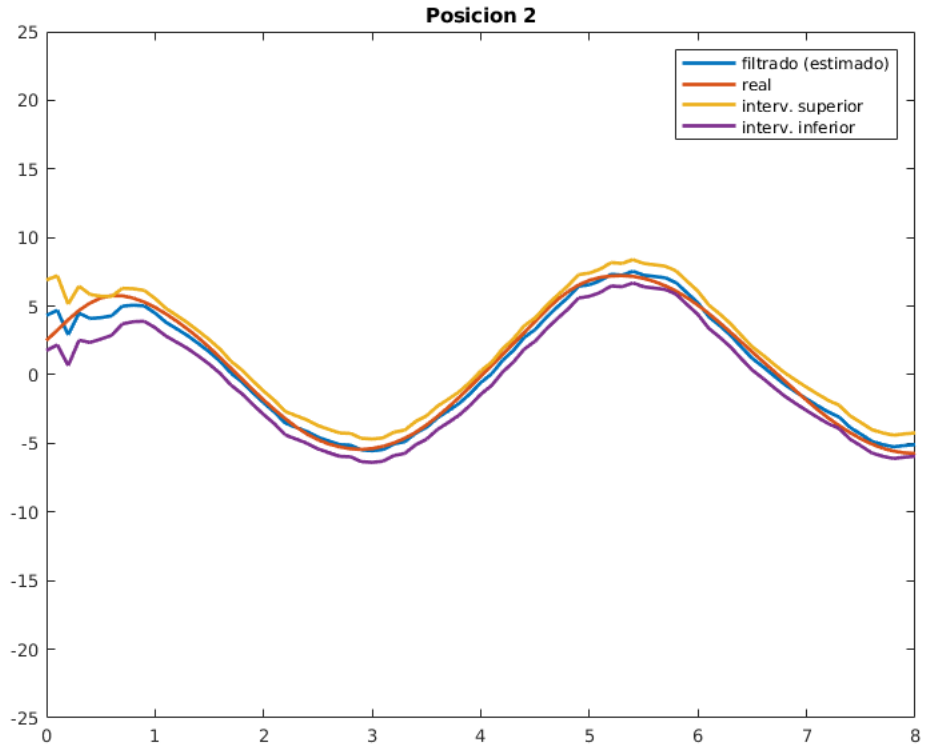

```
plot(Tiempos,[y(:,1)  ysup(:,1) yinf(:,1)],'LineWidth',2)
title('Posicion 1')
legend('real','interv. superior','interv. inferior');
grid on
```



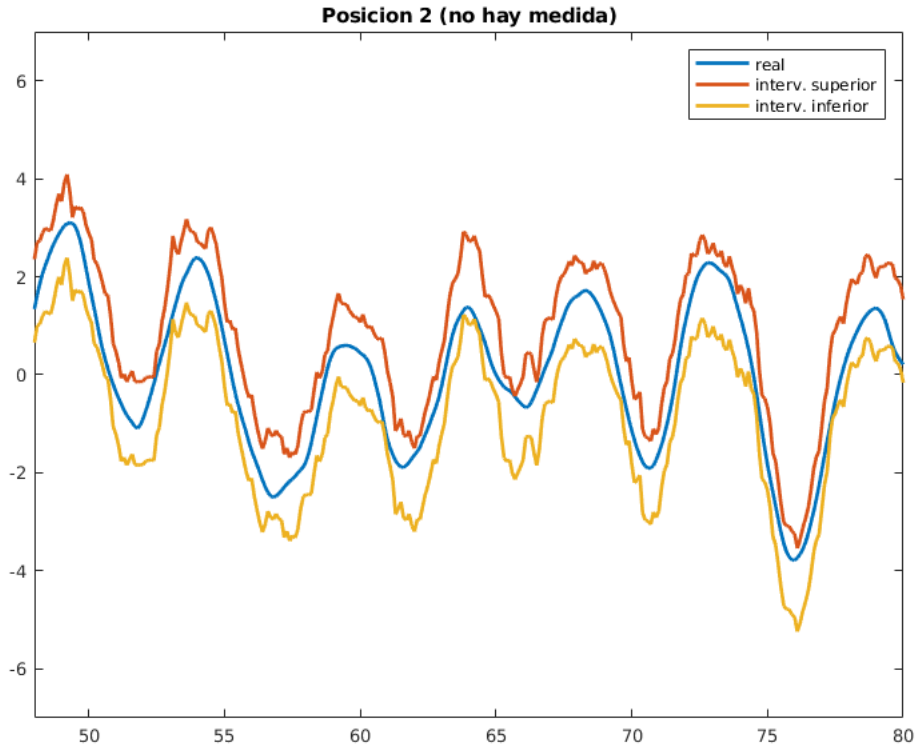

```
plot(Tiempos,[ysr(:,2) y(:,2) ysup(:,2) yinf(:,2)],'LineWidth',2)
title('Posicion 2')
legend('filtrado','simulacion','interv. superior','interv. inferior');
```



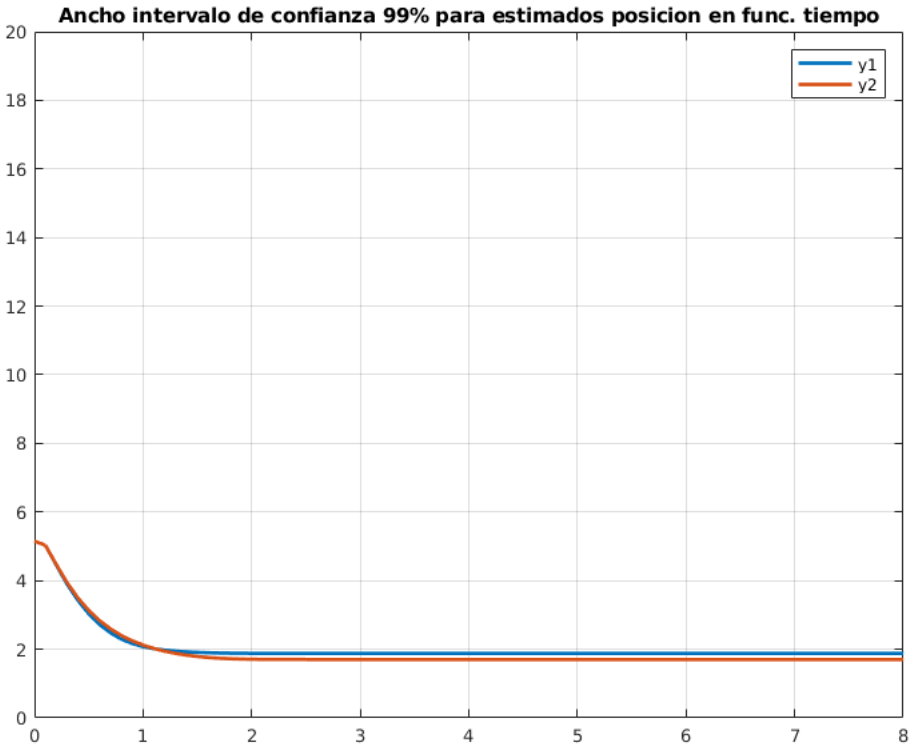

```
%un ZOOM INICIAL:
plot(Tiempos,[ysr(:,2) y(:,2) ysup(:,2) yinf(:,2)],'LineWidth',2)
title('Posicion 2')
legend('filtrado (estimado)','real','interv. superior','interv. inferior');
axis([0 Tiempos(end)*.1 -25 25])
```




```
%un ZOOM FINAL:
plot(Tiempos,[y(:,2) ysup(:,2) yinf(:,2)],'LineWidth',2)
title('Posicion 2 (no hay medida)')
legend('real','interv. superior','interv. inferior');
axis([Tiempos(end)*.6 Tiempos(end) -7 7])
```




```
plot(Tiempos,[ysup(:,1)-yinf(:,1) ysup(:,2)-yinf(:,2)],'LineWidth',2)
title('Ancho intervalo de confianza 99% para estimados posicion en func. tiempo')
legend('y1','y2');
axis([0 Tiempos(end)*.1 0 20])
grid on
```



Solución estacionaria:

```
GananciaKalman
```

```
GananciaKalman =
    0.1328    0.1068
    0.1700    0.0751
    0.1068    0.1088
    0.2153    0.1272
```

```
VCEstado_sinmedir
```

```
VCEstado_sinmedir =
    0.1705    0.2095    0.1403    0.2699
    0.2095    0.8873    0.1094    0.5732
    0.1403    0.1094    0.1389    0.1751
    0.2699    0.5732    0.1751    0.6942
```

```
VCEstado
```

```
VCEstado =
    0.1328    0.1700    0.1068    0.2153
    0.1700    0.8435    0.0751    0.5141
    0.1068    0.0751    0.1088    0.1272
    0.2153    0.5141    0.1272    0.6138
```

```
```

Comparamos con solución estacionaria Matlab dlqe:

```
[M,P,Z,E]=dlqe(muellesd.A,Bdiscruido,Csens,VCRuidoProc,VCRuidoMed)
```

```
M =
    0.1328    0.1068
    0.1700    0.0751
    0.1068    0.1088
    0.2153    0.1272
P =
    0.1705    0.2095    0.1403    0.2699
    0.2095    0.8873    0.1094    0.5732
    0.1403    0.1094    0.1389    0.1751
    0.2699    0.5732    0.1751    0.6942
Z =
    0.1328    0.1700    0.1068    0.2153
    0.1700    0.8435    0.0751    0.5141
    0.1068    0.0751    0.1088    0.1272
    0.2153    0.5141    0.1272    0.6138
E =
   0.7211 + 0.1630i
   0.7211 - 0.1630i
   0.8577 + 0.1459i
   0.8577 - 0.1459i
```

Selección de sensores, comparando varias opciones

```
sqrt(diag(VCSalida))'
```

```
ans =
    0.3645    0.3299
```

```
% vamos a ver sqrt(varianza total) de las dos posiciones estimadas, tiene unidades de l
sqrt(trace(VCSalida))
```

```
ans = 0.4916
```

```
%idem de estado completo
sqrt(diag(VCEstado))'
```

```
ans =
    0.3645    0.9184    0.3299    0.7834
```

```
sqrt(trace(VCEstado))
```

```
ans = 1.3034
```

```
%comparar diversas opciones!
```