Estimación (Interpolación/extrapolación) en procesos estocásticos gaussianos

Vídeo en http://personales.upv.es/asala/YT/V/krigtm.html

Este código ejecutó correctamente en Matlab R2018b

Table of Contents

1.- Generación de los datos

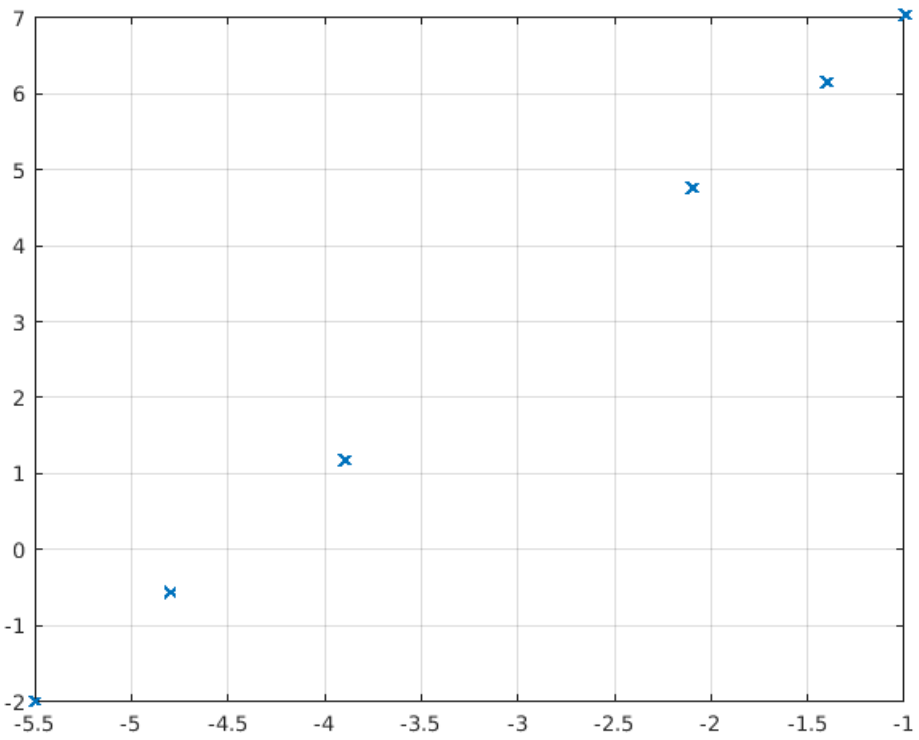
Obtenemos muestras en unos pocos instantes, y contaminados con un poco de ruido de medida

```
T=[-5.5 -4.8 -3.9 -2.1 -1.4 -1];%instantes arbitrarios, no equiespaciados
%Y=[0.25 0.9 0.85 -0.25 -1 -0.9]';
Y=2*T'+9;

N=length(T)
```

```
N =
     6
```

```
std_ruidomed=0.04;
var_ruidomed=std_ruidomed^2;
Y=Y+randn(N,1)*std_ruidomed;
plot(T,Y,'x','LineWidth',3), grid on, hold off, axis tight
```



2.- Regresión

Hagamos distintas pruebas de autocovarianzas o Power Spectral Density:

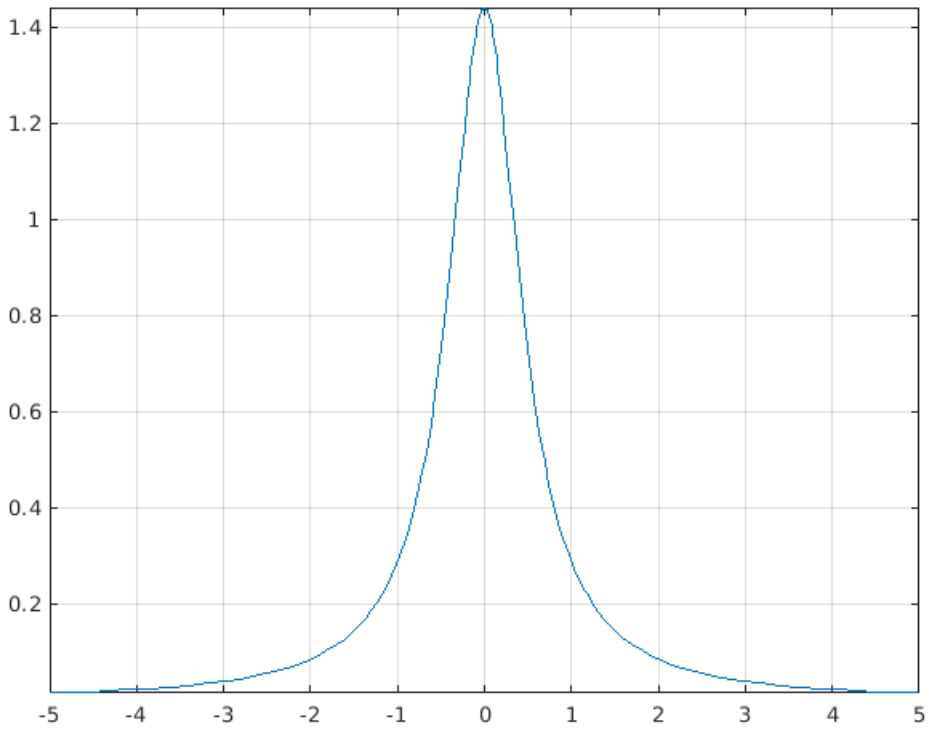
```
prueba=1;
switch prueba
    case 1
        G=@(s) .6/(s+.5);%primer orden, covarianza exponecial.
    case 2
        bw=1.5/10;
        G=@(s) 1.5*bw^2/(s^2+sqrt(2)*bw*s+bw^2);%Butterworth
    case 3
        G=@(s) 3/((s+1)^2+3^2);%1/(s^2+1*s+5);%Resonante
end

syms w real
PowerSpectralDensity=simplify(G(1i*w)*G(-1i*w))
```

```
PowerSpectralDensity =
```

$$\frac{36}{100\,w^2+25}$$

```
fplot(PowerSpectralDensity), grid on
```

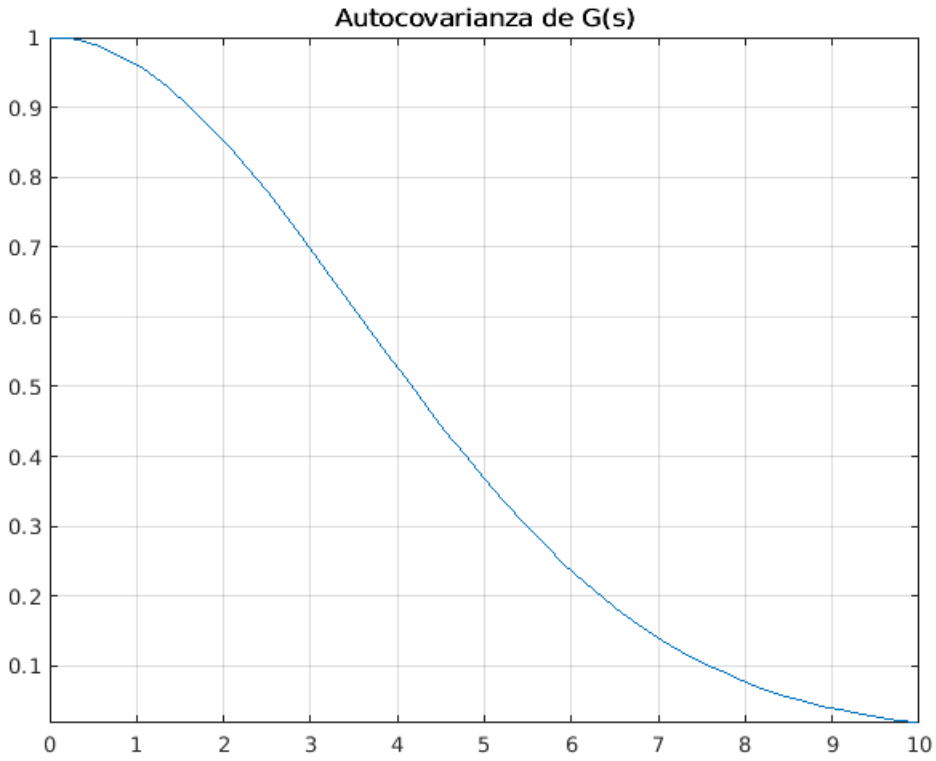



```
%AutoCovarianza=simplify(ifourier(PowerSpectralDensity,'t'))
syms t real
AutoCovarianza=exp(-t^2/5^2)
```

```
AutoCovarianza =
```

$$e^{-\frac{t^2}{25}}$$

```
fplot(AutoCovarianza,[0,10]), grid on, title('Autocovarianza de G(s)')
```




```
cosa=matlabFunction(AutoCovarianza);
desvtipprocesogaussiano=sqrt(cosa(0)) %desv.tip. estacionaria
```

```
desvtipprocesogaussiano =
     1
```

```
norma2deG=norm(G(tf('s')))
```

```
norma2deG =
   0.600000000000000
```

```
kappa=@(t1,t2) cosa(abs(t2-t1));

K=KappaTT(T,T,kappa)
```

```
K = 6×6
   1.000000000000000   0.980590831202428   0.902668412080942   0.629770381401003 ···
   0.980590831202428   1.000000000000000   0.968119256916563   0.747067303137196
   0.902668412080942   0.968119256916563   1.000000000000000   0.878446739349931
   0.629770381401003   0.747067303137196   0.878446739349931   1.000000000000000
   0.510481949742289   0.629770381401003   0.778800783071405   0.980590831202428
   0.444858066222941   0.561243736443235   0.714337313735957   0.952752609803211
```

```
format long
eig(K)
```

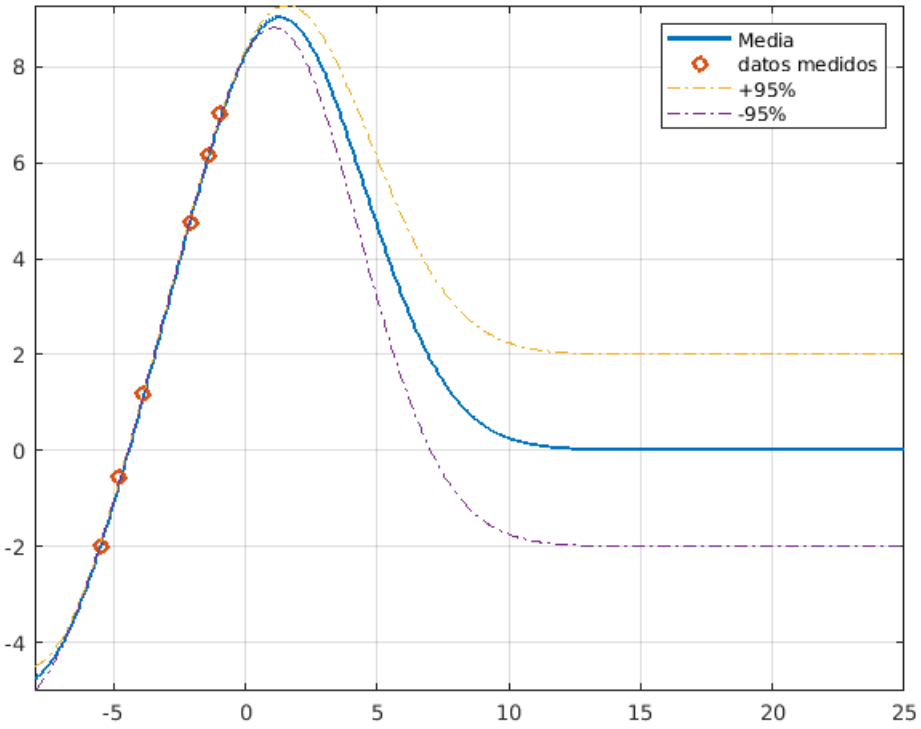
```
ans = 6×1
   0.000000929017075
   0.000040709466043
```

```
0.002254837504692
0.046089793274293
1.045262110893347
4.906351619844552
```

3.- Estimación en estados intermedios

```
t_todos=-8:0.05:25;
Ntodo=length(t_todos);
lista_media=zeros(1,Ntodo);
lista_desvtip=zeros(1,Ntodo);
prediccion=@(t) KappaTT(t,T,kappa)*inv(K+var_ruidomed*eye(N))*Y;
for i=1:Ntodo
     lista_media(i)=prediccion(t_todos(i));

lista_desvtip(i)=sqrt(varianzaestimada(t_todos(i),T,kappa,var_ruidomed));
end
plot(t_todos,lista_media,'LineWidth',2)
hold on
plot(T,Y,'o','LineWidth',3)
plot(t_todos,lista_media+lista_desvtip*2,'-.')
plot(t_todos,lista_media-lista_desvtip*2,'-.')
hold off, grid on, axis tight
legend('Media','datos medidos','+95%','-95%')
```



Apéndice: funciones auxiliares

```
function K=KappaTT(T1,T2,kappa)
Nx=length(T1);
Ny=length(T2);
K=zeros(Nx,Ny);
for i=1:Nx
    for j=1:Ny
        K(i,j)=kappa(T1(i),T2(j));
    end
end
end

function V=varianzaestimada(t,T,kappa,varruidomed)
    Cov=KappaTT(T,t,kappa);
    K=KappaTT(T,T,kappa);
    V=kappa(t,t)-Cov'*inv(K+varruidomed^2*eye(size(K,2)))*Cov;
end
```