Regresión mixta kernel+paramétrica, análisis comparativo

Presentación en vídeo: http://personales.upv.es/asala/YT/V/semipcaso.html

Este código funcionó correctamente con Matlab **R2020b**

Objetivos: Esto es una prueba de regresión Kernel + componente paramétrico, analizando diferentes opciones de regresores.

Tabla de Contenidos

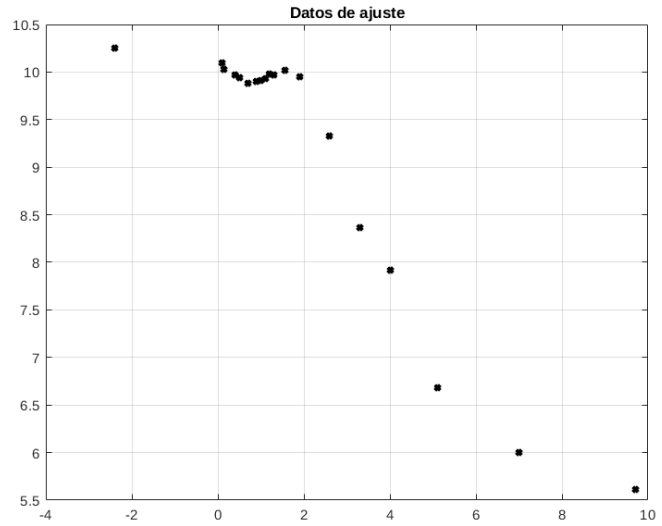
Datos a ajustar

Estos serían unos datos para ajustar un modelo

```
X=[-2.4  .1 .15  .4 .7 .9 1 1.1 1.3 1.55 .5:.7:4 5.1 7 9.7]';
%X=rand(500,1)*100-40;
N=length(X)
```

```
N = 19
```

```
funcverdadera=@(X) sqrt(1.02+sin(0.8*X).^3)*2.1+8-0.55*X; %desconocida, claro
y=funcverdadera(X)+0.02*randn(N,1); %ruido de medida en las muestras
plot(X,y,'xk','LineWidth',3), grid on, title('Datos de ajuste')
```



Si analizamos más o menos "grosso modo":

```
My=max(y), my=min(y)
```

```
My = 10.2520
my = 5.6138
```

Los datos suben y bajan un incremento

```
0.5*(My-my)
```

```
ans = 2.3191
```

alrededor de

```
0.5*(My+my)
```

```
ans = 7.9329
```

Con los diferentes modelos, comprobaremos el ajuste de los modelos en estos puntos:

```
x_test=(-6:0.1:14)'; NT=length(x_test);
y_verdadera=funcverdadera(x_test);
```

Regresión

Mínimos cuadrados clásicos, sin información "a priori"

Formalmente, tenemos que hacer los datos a media cero

```
mean(y)
```

```
ans = 9.1446
```

Pero como suponemos que hay correlación entre muestras cercanas en x (por eso "Kernel") la media de arriba no "cuadra" con la intuición de que la función de x subyacente parece tener una media de 8 en el intervalo donde hay datos... Eso se corregirá luego.

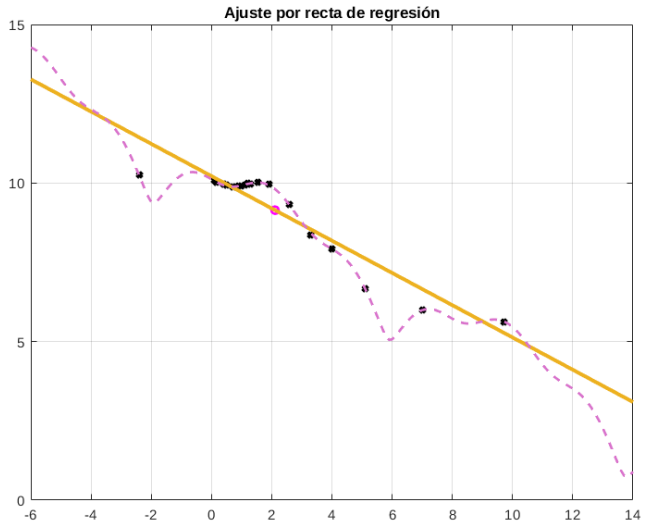
```
Xincr=X-mean(X);
yincr=y-mean(y);
```

El estimado mínimo-cuadrático (recta de regresión) tiene de pendiente:

```
Th_1=pinv(Xincr)*yincr
```

```
Th_1 = -0.5088
```

```
plot(X,y,'xk','LineWidth',3), grid on, hold on %datos
plot(mean(X),mean(y),'om','LineWidth',2)
y_regresionlineal=Th_1*(x_test-mean(X))+mean(y);
plot(x_test,y_regresionlineal,'LineWidth',3) %predicción, deshaciendo cambio incrementa
hold on, plot(x_test,y_verdadera,'--','LineWidth',2,'Color',[.85 .45 .8]),hold off
title("Ajuste por recta de regresión")
```



El "intervalo de confianza 2.6σ" (sólo válido si los datos hubiesen sido generados por un modelo lineal+ruido de medida y tuviéramos infinitos datos) sería:

```
err=yincr-Th_1*Xincr;
std_th=sqrt(var(err)*inv(X'*X))
```

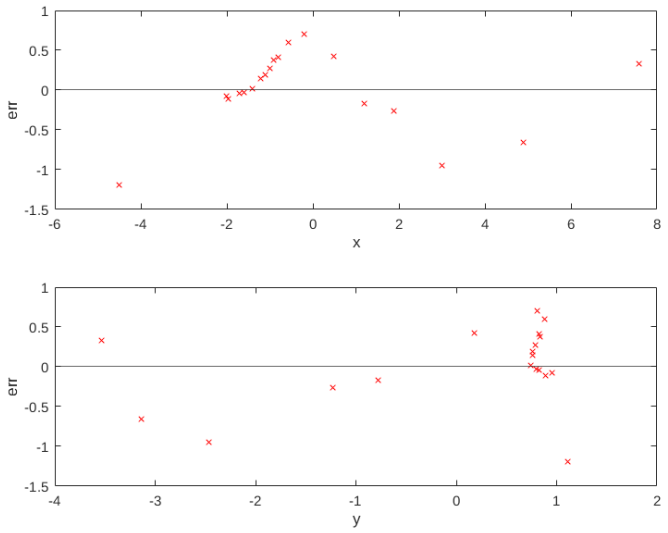
```
std_th = 0.0333
```

```
[Th_1-2.6*std_th Th_1+2.6*std_th]
```

```
ans = 1×2
   -0.5955   -0.4222
```

El error debería ser independiente de x, y de y:

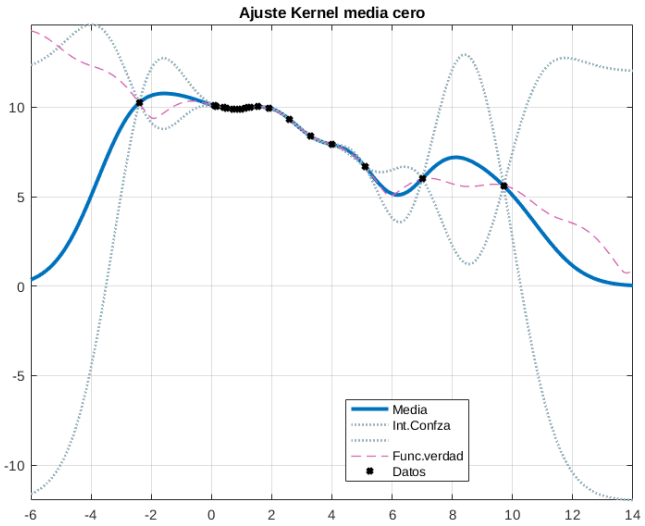
```
subplot(2,1,1)
plot(Xincr,err,'xr'), xlabel('x'),ylabel('err'), yline(0);
subplot(2,1,2)
plot(yincr,err,'xr'), xlabel('y'),ylabel('err'), yline(0);
```



Regresión "Kernel" (simple Krigging, media cero)

```
M=6^2;sg=1.4;lam=.02^2; %hiperparámetros

KernelReg=KernelRegressionClass(M,sg,[],lam); %el componente paramétrico es inexistente
KernelReg.DoKernelRegression(X,y);
y_SK=KernelReg.Predict(x_test);
figure()
KernelReg.PlotFittedFunction();
hold on, plot(x_test,y_verdadera,'--','LineWidth',1,'Color',[.85 .4 .7]),
KernelReg.PlotTrainData(); hold off
legend('Media','Int.Confza','','Func.verdad','Datos','Location','best')
title("Ajuste Kernel media cero")
```



Ordinary Krigging (media desconocida)

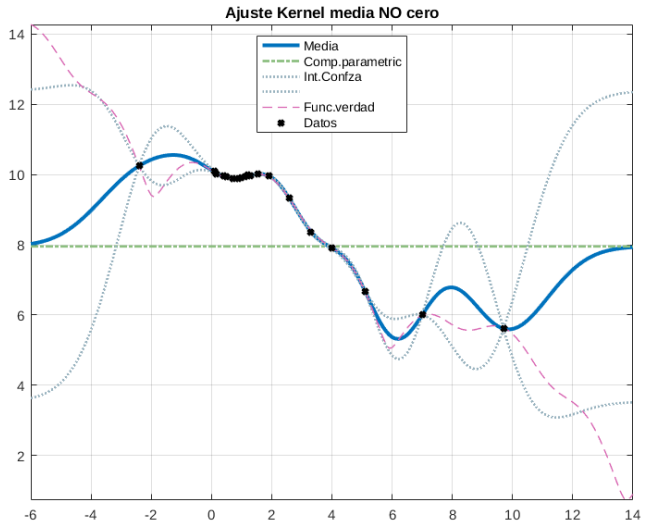
```
M=2^2;sg=1.4;lam=.02^2;Sigma_TH=diag(8^2);
KernelReg=KernelRegressionClass(M,sg,Sigma_TH,lam);
KernelReg.phi=@(x)  ones(size(x));
KernelReg.DoKernelRegression(X,y);
```

Los parámetros (media) estimados son:

```
KernelReg.result.th
```

```
ans = 7.9553
```

```
y_OK=KernelReg.Predict(x_test);
KernelReg.PlotFittedFunction();
hold on, plot(x_test,y_verdadera,'--','LineWidth',1,'Color',[.85 .4 .7]),
KernelReg.PlotTrainData(); hold off
legend('Media','Comp.parametric','Int.Confza','','Func.verdad','Datos','Location','best
title("Ajuste Kernel media NO cero")
```



El "intervalo de confianza 2.6σ" del componente paramétrico (sólo válido si los datos hubiesen sido generados por un modelo como el que hemos supuesto a priori) sería:

```
KernelReg.result.th+[-2.6*KernelReg.result.stdth 2.6*KernelReg.result.stdth]
```

```
ans = 1×2
    5.4950   10.4157
```

Kernel Ridge-Regresión con componente lineal

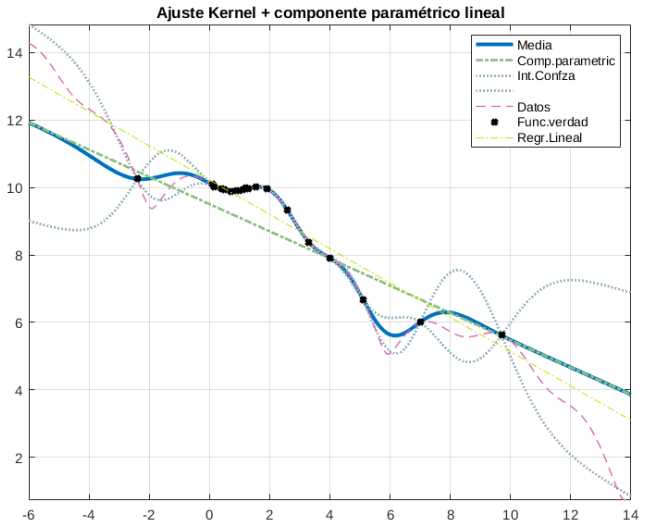
```
M=1^2;sg=1.2;lam=.02^2;Sigma_TH=diag([0.35^2 9^2]);
KernelReg=KernelRegressionClass(M,sg,Sigma_TH,lam);
KernelReg.phi=@(x) [x ones(size(x))];
KernelReg.DoKernelRegression(X,y);
```

los parámetros (pendiente, offset constante) estimados son:

```
KernelReg.result.th
```

```
ans = 2×1
   -0.4035
    9.5081
```

```
y_LK=KernelReg.Predict(x_test);
KernelReg.PlotFittedFunction();
hold on, plot(x_test,y_verdadera,'--','LineWidth',1,'Color',[.85 .4 .7])
KernelReg.PlotTrainData();
plot(x_test,y_regresionlineal,'-.','Color',[.75 .9 .0]),hold off
legend('Media','Comp.parametric','Int.Confza','','Datos','Func.verdad','Regr.Lineal')
title("Ajuste Kernel + componente paramétrico lineal")
```



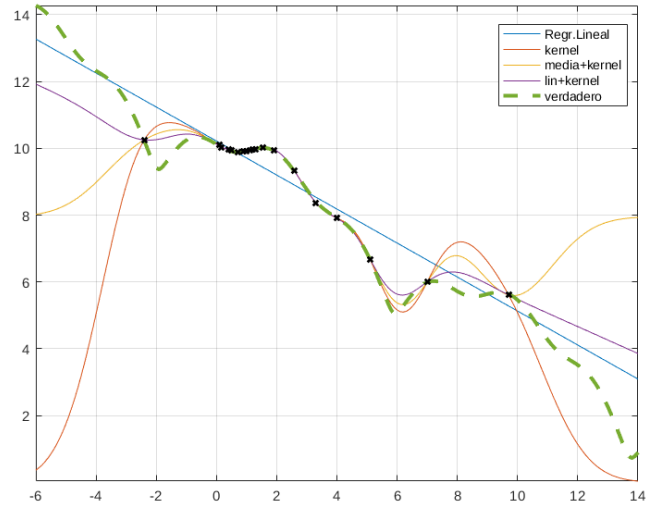
Obsérvese que el componente paramétrico estimado no coincide con la recta de regresión. la recta de regresión enfatiza demasiado la nube de puntos entre 0 y 2.

```
KernelReg.result.th+[-2.6*KernelReg.result.stdth 2.6*KernelReg.result.stdth]
```

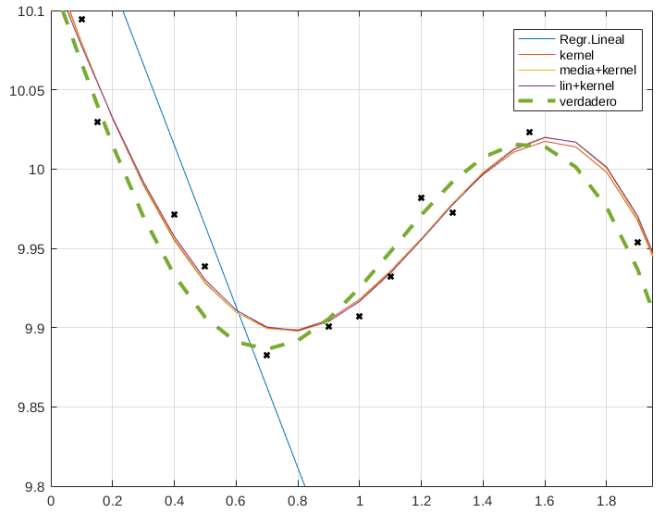
```
ans = 2×2
   -0.6643   -0.1427
    8.0070   11.0092
```

Superponemos todos los resultados

```
for i=1:2 %repito figura
    figure()
    plot(x_test,[y_regresionlineal y_SK y_OK y_LK]), grid on, axis tight
    hold on
    plot(x_test,y_verdadera,'--','LineWidth',3)
    plot(X,y,'xk','LineWidth',2)
    hold off
    legend('Regr.Lineal','kernel','media+kernel','lin+kernel','verdadero','Location','k
end
```




```
axis([0 1.95 9.8 10.1]) %cambio ejes
```



Conclusiones

Diferentes modelos semiparamétricos extrapolan de forma diferente; también se nota más la diferencia entre muestras muy separadas, que es donde más importancia tienen las suposiciones "a priori" sobre la media y la suavidad de la función a aproximar.

Nota: los hiperparámetros (varianza y distancia típica del Kernel, ruido de medida, varianza de parámetros del componente paramétrico) deberían ser sintonizados evaluando el ajuste a un conjunto de datos de validación, o eligiendo aquéllos para los que los datos disponibles tengan

más probabilidad (ver, por ejemplo la documentación https://es.mathworks.com/help/stats/exact-gpr-method.html).

.