Aproximación por mínimos cuadrados

```
Clear["Global`*"]
```

- ***Dada la tabla de de datos a aproximar***

x_k	x_1	x_2	$\cdots$	x_n
y_k	y_1	y_2	$\cdots$	y_n

introducimos los datos de la forma

```
x = {x1, x2, ...., xn}; y = {y1, y2, ...., yn}; n = Length[x];
```

Confeccionamos también la lista de puntos por si queremos dibujarlos para compararlos con la función que los aproxima.

```
datos = Table[{x[[i]], y[[i]]}, {i, n}]
```

Aproximación lineal de los datos $y=c_0+c_1 z$

```
(* Formación de la matriz y el vector
  que nos proporcionan los parámetros c0 y c1 *)
Clear[A1];
A1 = Table[x[[i]]^(j-1), {i, n}, {j, 2}];
y = Table[y[[j]], {j, n}];
matriz1 = Transpose[A1].A1;
u1 = Transpose[A1].y;
sol1 = LinearSolve[matriz, u1]
```

```
f1 = sol1.{1, z}
  (*formo la ecuación de la recta con un producto escalar*)
```

```
puntos1 =
  ListPlot[datos, PlotStyle → {PointSize[0.02], RGBColor[0, 0, 1]}];
```

```
a = Min[x];
b = Max[x];
graf1 = Plot[f1, {z, a, b}, DisplayFunction → Identity, PlotRange → All];
```

```
s1 = Show[puntos1, graf1];
```

Ajuste cuadrático $y = c_0 + c_1 z + c_2 z^2$

```
(* Formación de la matriz y el vector
  que nos proporcionan los parámetros c0 , c1 y c2*)
Clear[A2];
A2 = Table[x[[i]]^(j-1), {i, n}, {j, 3}];
y = Table[y[[j]], {j, n}];
matriz2 = Transpose[A2].A2;
u2 = Transpose[A2].y;
sol2 = LinearSolve[matriz2, u2]
```

```
f2 = sol2.{1, z, z^2}
  (*formo la parábola con un producto escalar*)
```

```
puntos2 =
  ListPlot[datos, PlotStyle → {PointSize[0.02], RGBColor[1, 0, 0]}];
```

```
a = Min[x];
b = Max[x];
graf2 = Plot[f2, {z, a, b}, DisplayFunction → Identity, PlotRange → All];
```

```
s2 = Show[puntos2, graf2];
```

Mostramos simultáneamente los datos y sus ajustes lineal y cuadrático

```
s = GraphicsArray[{s1, s2}];
```

```
Show[s];
```

Ajuste Exponencial $y=e^{c_0+c_1 z}$

Tomando logaritmos neperianos en ambos miembros obtenemos Logy=$c_0 + c_1 z$, con lo que el problema pasa a ser un ajuste lineal tomando como nuevas abcisas la mismas que las anteriores y como nuevas ordenadas los logaritmos neperianos de las ordenadas de partida.

Pasamos a construir dicho ajuste lineal con los datos descritos

```
Clear[A3];
A3 = Table[x[[i]]^(j-1), {i, n}, {j, 2}];
y = Table[y[[j]], {j, n}];
matriz3 = Transpose[A3].A3;
u3 = Transpose[A3].Log[y];
sol3 = LinearSolve[matriz3, u3]
```

formo la exponencial $y=e^{c_0+c_1 z}$

```
f3 = Exp[sol3.{1, z} ]
```

```
puntos3 =
  ListPlot[datos, PlotStyle → {PointSize[0.02], RGBColor[0, 1, 0]}];
```

```
a = Min[x];
b = Max[x];
graf3 =
  Plot[f3, {z, a, b}, DisplayFunction → Identity, PlotRange → All];
```

```
s3 = Show[puntos3, graf3];
```

Mostramos ahora los tres ajustes efectuados a los datos introducidos

```
ns = GraphicsArray[{s1, s2, s3}];
```

```
Show[ns];
```

Ejercicios

1.- ¿Es la recta y=2x+3 la que mejor aproxima los datos (0,3), (2,7), (3,9) y (11,25) en el sentido de mínimos cuadrados?

2.- La ley de Hooke establece que la longitud y, que se estira un resorte es proporcional a la fuerza aplicada, x; a saber, y=ax+b, siendo b la longitud del resorte, en reposo y a es una constante que depende del resorte. Se colocan cinco pesos diferentes en el extremo del resorte

peso en Kg, x	2	4	6	8	10
Longitud en cm, y	4.1	5.9	8	10.1	12.2

Determine la longitud inicial y la constante del resorte utilizando el método de mínimos cuadrados.

3.- Supongamos que, según la ley de Lambert-Beer, la absorbancia de uncomplejo se obtiene a partir de la concentración mediante técnicas espectrofotométricas. Se dipone de los datos:

c_i , (concentración)	1	2	3	5	10
A_i , (absorbancia)	0.1	0.36	0.57	1.09	2.05

a) Calcule el ajuste lineal por mínimos cuarados para tales datos.

b) Halle el ajuste cuadrático de mínimos cuadrados.

c) Compare ambos modelos para comparar los valores de A(c) calculando, en cada caso, la estimación de A(2) y A(5).