

# IntPak. Ajuste e Interpolación Multivariable de Funciones

P. L. Matencio  
J. Vales Alonso

El ajuste e interpolación de funciones son, posiblemente, las herramientas de análisis numérico más utilizadas por ingenieros e investigadores en general. Los problemas de ajuste e interpolación de funciones se pueden generalizar, de forma natural, a varias dimensiones. A la hora de comparar varios métodos resulta útil tenerlos integrados en un sólo paquete, accesibles, todos ellos, desde una única función, y, utilizar un formato de datos de entrada común. En este trabajo se presenta un paquete de estas características, que hemos llamado IntPak y reúne, además de las funciones `Fit[]` e `Interpolation[]` de *Mathematica*, los métodos de interpolación de Lagrange, Trigonométrica, Splines de orden  $(m, s)$ , y de Producto Tensorial de Elementos Finitos de Hermite de clase  $C^m$ , así como ajuste mediante Splines de orden  $(m, s)$ . El lenguaje de alto nivel de *Mathematica* hace que la realización de computos donde intervienen gran cantidad de datos sea sencilla de implementar y de entender.

## ■ Introducción a la teoría de aproximación

La teoría de aproximación nos dice que, dado un conjunto  $\mathcal{X}$ , y  $\Phi$  un subconjunto de  $\mathcal{X}$ , podemos imponer condiciones de forma que a cualquier  $x \in \mathcal{X}$  le asociamos un  $\varphi_x \in \Phi$ , y una medida de la discrepancia entre  $x$  y  $\varphi_x$ .

En la práctica  $\mathcal{X}$  es un conjunto de objetos matemáticos. Normalmente,  $\mathcal{X}$  es un espacio vectorial sobre el cuerpo  $\mathbb{R}$ , por ejemplo  $\mathbb{R}^n(\mathbb{R})$ , con un producto escalar asociado,  $(\cdot, \cdot)$ , lo que permite inducir una norma  $\|\cdot\| = (\cdot, \cdot)^{1/2}$ , y,  $\Phi$  es un subespacio vectorial de dimensión finita  $n \in \mathbb{N}$ . Por lo tanto, dada una base  $\{\varphi_k\}_{k=1}^n$ , para cualquier  $\varphi \in \Phi_n$ , existe una única representación suya en dicha base  $\varphi = \sum_{i=1}^n C_i \varphi_i$ . Por el teorema de la proyección ortogonal, la condición necesaria y suficiente para que  $\varphi_x \in \Phi_n$  sea la aproximación óptima finita de  $x \in \mathcal{X}$  es que cumpla que  $(x - \varphi_x, \varphi) = 0, \forall \varphi \in \Phi_n$ .

Para obtener  $\varphi_x$  podemos resolver el sistema de ecuaciones siguiente:

$$\sum_{k=1}^n (\varphi_i, \varphi_k) C_k = (x, \varphi_i), \quad \forall i = 1, 2, \dots, n \quad (1)$$

donde,

$$G(\{\varphi_k\}) = \begin{pmatrix} (\varphi_1, \varphi_1) & \cdots & (\varphi_1, \varphi_n) \\ \vdots & \ddots & \vdots \\ (\varphi_n, \varphi_1) & \cdots & (\varphi_n, \varphi_n) \end{pmatrix} \quad (2)$$

es la matriz de Gram relativa a dicha base, y es una matriz simétrica y definida positiva. Sin embargo, en muchas ocasiones, el elemento  $x \in \mathcal{X}$ , cuya aproximación deseamos, es desconocido. Esto sucede si, por ejemplo, los datos de partida son el resultado de cierto problema o experimento.

## □ Ajuste Multivariable por Mínimos Cuadrados

El método de ajuste por mínimos cuadrados se fundamenta en el criterio de que la suma de los cuadrados de las diferencias entre la función de aproximación y los valores dato sea mínima [1].

Supongamos que tenemos  $n + 1$  parejas de datos  $(x_i, y_i)$  puntos datos  $x_i$ , y valores dato  $y_i$ , donde  $i = 0, 1, \dots, n$ . Cada punto dato está formado por  $h$  componentes,  $x_i = x_{i_1}, x_{i_2}, \dots, x_{i_h}$ , siendo  $h$  las variables independientes de la función aproximada. Se quiere generar una función  $y = f(x)$  que aproximadamente ajuste estos puntos dato usando el criterio de mínimos cuadrados.

La función  $f(x)$  será función de unos coeficientes  $a_0, a_1, \dots, a_m$ , siendo la cantidad de coeficientes,  $m + 1$ , menor que la cantidad de puntos dato,  $n + 1$ . Considerando que todos los valores dato tienen el mismo grado de fiabilidad, entonces podemos escribir  $S = \sum_{i=0}^n [y_i - f(x_i, a_0, a_1, \dots, a_m)]^2$

Bajo la hipótesis de diferenciable, una condición necesaria que debe satisfacer el mínimo (suponiendo que exista) es que la derivada parcial de la expresión  $S$  con

respecto a cada coeficiente, debe ser cero. Por lo tanto, podemos determinar los coeficiente de la función resolviendo el sistema de ecuaciones siguiente:

$$\frac{\partial S}{\partial a_j} = 0, \quad j = 0, 1, \dots, m \quad (3)$$

La condición (4) también puede ser suficiente para minimizar  $S$ , pero eso depende de la función  $f(x_i, a_0, a_1, \dots, a_m)$  específica que utilizemos. Por regla general, empleamos *combinaciones lineales* de funciones respecto de los coeficientes  $a_j$ .

Sea  $f(x)$  un elemento de la variedad lineal generada por la base:  $\{g_j(x)\}_{j=0,1,2,\dots,m}$

Entonces,  $f(x)$  se puede escribir como:

$$f(x, a_0, a_1, \dots, a_m) = a_0 g_0(x) + a_1 g_1(x) + \dots + a_m g_m(x) \quad (4)$$

Sustituyendo (4) en (3), nos queda un sistema de ecuaciones que podemos poner de forma matricial de la siguiente forma:

$$\begin{pmatrix} \sum_{i=0}^n g_0(x_i) g_0(x_i) & \dots & \sum_{i=0}^n g_m(x_i) g_0(x_i) \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^n g_0(x_i) g_m(x_i) & \dots & \sum_{i=0}^n g_m(x_i) g_m(x_i) \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_m \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^n y_i g_0(x_i) \\ \vdots \\ \sum_{i=0}^n y_i g_m(x_i) \end{pmatrix} \quad (5)$$

Si el sistema (7) es no singular, entonces, existe una única solución. Para ello, por ejemplo, podemos escoger las funciones  $g_j(x)$  dentro del espacio de los polinomios de grado menor o igual a  $m$ ,  $\{g_j(x)\}_{j=0,1,2,\dots,m} \in \mathcal{P}_m$ .

#### □ Resolviendo el Problema de Ajuste por Mínimos Cuadrados

El paquete `IntPak` contiene la opción de resolver la ajuste por mínimos cuadrados mediante la función `Fit[]` (en este punto, debemos cargar primero el paquete).

```
<< MultivarInterpol`IntPak`
```

```
? PLMIntp
```

---

PLMIntp[datos] permite elegir entre 7 métodos de aproximación:

1. Ajuste por Mínimos Cuadrados.
2. Interpolación mediante polinomios de Lagrange.
3. Interpolación usando Mathematica. Permite interpolar usando las derivadas de primer orden (clase=1).
4. Interpolación Trigonométrica.
5. Ajuste mediante splines tipo (m,s).  
El parámetro  $\epsilon$  (eps) toma un valor entre 0 y 1.
6. Interpolación mediante splines tipo (m,s).
7. Interpolación mediante producto tensorial: splines clase m.

La forma de expresar los datos es:

```
{{x11,x12,...,x1n,f1},{x21,x22,...,x2n,f2},.....}
```

donde 'n' es el número de variables independientes, y  $f_i$  es el valor de la función en el punto de coordenadas  $(x_{i1}, \dots, x_{in})$ .

PLMIntp[] tiene la siguiente opción:

Grafica->False (o True)

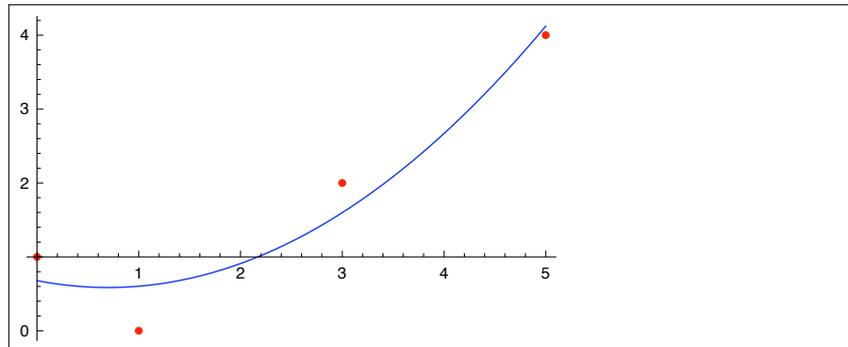
No dibuja (o sí dibuja) la función interpolada

```
data1 = {{0, 1}, {1, 0}, {3, 2}, {5, 4}};
PLMIntp[data1, Grafica -> True]
```

El dominio de la función aproximada es:

$\{\{x1, 0., 5.\}\}$

Aproximación por MÍNIMOS CUADRADOS...con la base:  
 $\{1, x1, x1^2\}$



$$0.678392 - 0.266332 x1 + 0.190955 x1^2$$

Ahora podemos ajustar una perturbación aleatoria de una Gaussiana.

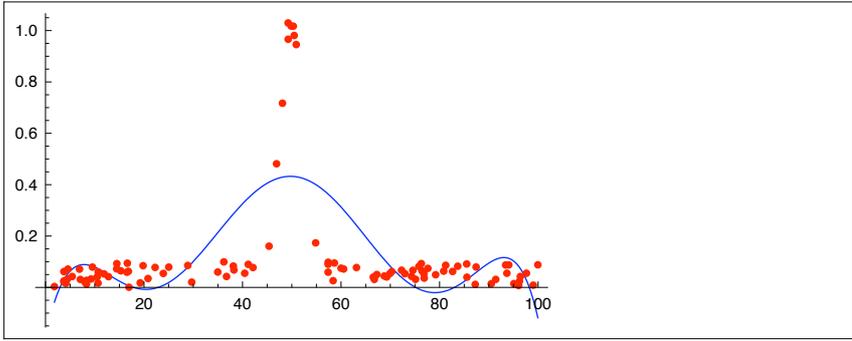
```
dataGauss = Block[{r = RandomReal[100, 100]}, Transpose[
  {r, Exp[-(r - 50)^2 / 10] + RandomReal[.1, 100]}]];]
```

Una función para generar bases del espacio vectorial de los polinomios,

```
nbase[n_Integer][x_] := Table[x^k, {k, 0, n}];
basePol = nbase[6][(x1 - 50) / 50];
```

La base la escalamos y desplazamos para que los coeficientes de las potencias más grandes no resulten valores demasiado bajos.

**PLMIntp[dataGauss, Grafica -> True];**

|  |
|--|
| El dominio de la función aproximada es:  |
| $\{\{x1, 1.81055, 99.943\}\}$  |
| Aproximación por MÍNIMOS CUADRADOS...con la base: $\left\{ 1, \frac{1}{50}(-50+x1), \frac{(-50+x1)^2}{2500}, \frac{(-50+x1)^3}{125\,000}, \frac{(-50+x1)^4}{6\,250\,000}, \frac{(-50+x1)^5}{312\,500\,000}, \frac{(-50+x1)^6}{15\,625\,000\,000} \right\}$ |
|   |

■ **Problema General de Interpolación**

Sea  $\Phi_n$  un espacio vectorial sobre el cuerpo real de dimensión finita  $n \in \mathbb{N}$ , y sean  $n$  formas lineales  $\mathcal{R}_i : \Phi_n \rightarrow \mathbb{R}$ ,  $1 \leq i \leq n$ , definidas en  $\Phi_n$ . Para un conjunto  $x_1, x_2, \dots, x_n$  de números reales dados, se pide encontrar un  $\varphi \in \Phi_n$ , tal que,

$$\mathcal{R}_i(\varphi) = x_i, \quad 1 \leq i \leq n \tag{6}$$

**Teorema 1.** *Sea  $\{\varphi_k\}_{k=1}^n$ , una base de  $\Phi_n$ . Una condición necesaria y suficiente para que el problema de interpolación (8) admita solución para valores cualquiera de los  $x_i$ , es que la matriz generalizada de Gram sea regular, es decir,*

$$\det(\mathcal{R}_i(\varphi_k)_{i,k=1,\dots,n}) \neq 0 \tag{7}$$

Además en este caso, la solución es única (la demostración de este teorema está en [2]).

■ **Interpolación Multivariable de Lagrange**

En general, el problema de interpolación polinómica podemos plantearlo de la siguiente forma:

Dado el conjunto cerrado  $\overline{D}$  de puntos dato  $(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h})$ , y sus valores dato asociados  $f(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h})$ , con  $i_1 = 0, 1, \dots, n_1$ ,  $i_2 = 0, 1, \dots, n_2, \dots, i_h = 0, 1, \dots, n_h$ , encontrar un polinomio  $p \in \mathcal{P}_{n_1, n_2, \dots, n_h}$  del espacio vectorial

$$\mathcal{P}_{n_1, n_2, \dots, n_h} = \left\{ p \mid p(x_1, x_2, \dots, x_h) = \sum_{\substack{0 \leq c_1 \leq n_1 \\ 0 \leq c_2 \leq n_2 \\ \vdots \\ 0 \leq c_h \leq n_h}} a_{c_1, c_2, \dots, c_h} x_1^{c_1} x_2^{c_2} \cdots x_h^{c_h}, a_{c_1, c_2, \dots, c_h} \in \mathbb{R} \right\} \quad (8)$$

de todos los polinomios de grado menor o igual que  $n_1$  en  $x_1$ , que  $n_2$  en  $x_2$ , ..., y que  $n_h$  en  $x_h$  tal que,

$$p(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}) = f(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}) \quad (9)$$

para todo punto dato del conjunto  $\overline{D}$ . Los valores  $c_1, c_2, \dots, c_h$  son exponentes, mientras que  $i_1, i_2, \dots, i_h$  son índices que referencian a un elemento dentro de la malla de puntos dato  $\overline{D}$ . Además, se cumple que  $\mathcal{P}_{n_1, n_2, \dots, n_h} \subset \mathcal{P}_m$  con  $0 \leq c_1 + c_2 + \dots + c_h \leq m$ , y, si los puntos dato son distintos entre sí, entonces el polinomio que cumple (11) es único, [3].

Para resolver este problema de interpolación podemos escoger el subespacio vectorial de los polinomios de Lagrange.

Sea el espacio vectorial,

$$\mathcal{L}_{n_1, n_2, \dots, n_h} = \left\{ p \mid p(x_1, x_2, \dots, x_h) = \sum_{\substack{0 \leq i_1 \leq n_1 \\ 0 \leq i_2 \leq n_2 \\ \vdots \\ 0 \leq i_h \leq n_h}} a_{i_1, i_2, \dots, i_h} l_{n_1}^{i_1}(x_1) \cdots l_{n_h}^{i_h}(x_h), a_{i_1, i_2, \dots, i_h} \in \mathbb{R} \right\}$$

formado por polinomios de grado menor o igual que  $n_1$  en  $x_1, \dots$ , de grado menor o igual que  $n_h$  en  $x_h$ , siendo,

$$l_{n_1}^{i_1}(x_1) = \frac{\prod_{j=0, j \neq i_1}^{n_1} (x_1 - x_1^j)}{\prod_{j=0, j \neq i_1}^{n_1} (x_1^{i_1} - x_1^j)}, \dots, l_{n_h}^{i_h}(x_h) = \frac{\prod_{j=0, j \neq i_h}^{n_h} (x_h - x_h^j)}{\prod_{j=0, j \neq i_h}^{n_h} (x_h^{i_h} - x_h^j)}$$

los polinomios de Lagrange correspondientes a cada una de las dimensiones del punto dato. Se puede demostrar, que, una solución al problema de interpolación planteado en (11) es

$$p(x_1, x_2, \dots, x_h) = \sum_{\substack{0 \leq i_1 \leq n_1 \\ 0 \leq i_2 \leq n_2 \\ \vdots \\ 0 \leq i_h \leq n_h}} f(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}) l_{n_1, \dots, n_h}^{i_1, \dots, i_h}(x_1, x_2, \dots, x_h) \quad (10)$$

con

$$l_{n_1, \dots, n_h}^{i_1, \dots, i_h}(x_1, x_2, \dots, x_h) = l_{n_1}^{i_1}(x_1) \cdots l_{n_h}^{i_h}(x_h)$$

La base  $\left\{ \binom{i_1 \dots i_h}{n_1 \dots n_h} (x_1, x_2, \dots, x_h) \right\}_{\substack{i_1=0, \dots, n_1 \\ \vdots \\ i_h=0, \dots, n_h}}$  del espacio vectorial  $\mathcal{L}_{n_1, n_2, \dots, n_h}$  verifica que

la matriz generalizada de Gram es la matriz identidad. Además, la solución es única por tener matriz de Gram regular.

□ Algunos Ejemplos Interpolando con Polinomios de Lagrange

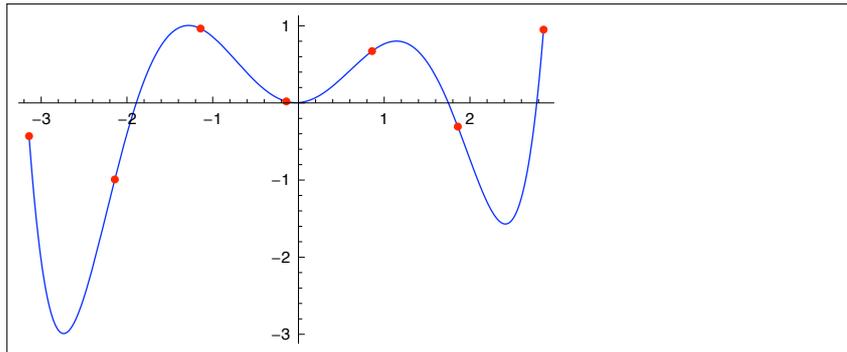
```
data3 = Table[N[{x1, Sin[x1^2]}], {x1, -Pi, Pi}];
```

```
PLMIntp[data3, Grafica -> True]
```

El dominio de la función aproximada es:

$\{\{x1, -3.14159, 2.85841\}\}$

Interpolación por LAGRANGE ...



```
0.0450828 (-2.77882 + x1) (-1.74815 + x1) (1.89118 + x1)
(3.16878 + x1) (0.00305992 + 0.0557603 x1 + x1^2)
```

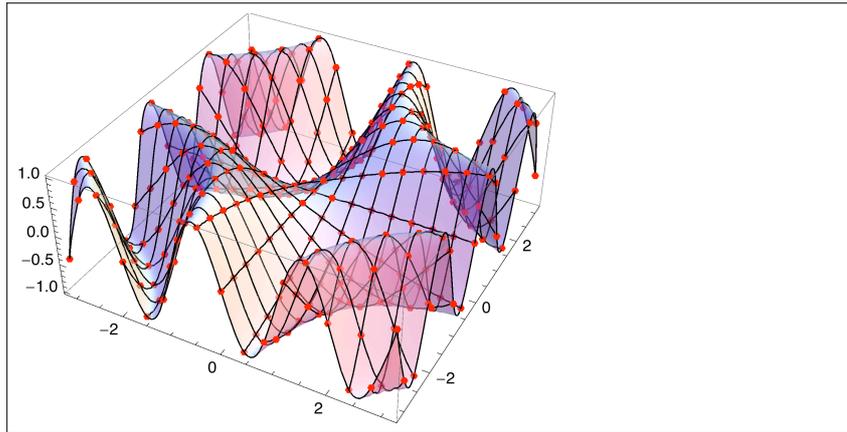
```
data4 = Flatten[Table[N[{x1, x2, Sin[x1 x2]}],
{x1, -Pi, Pi, Pi/8}, {x2, -Pi, Pi, Pi/8}], 1];
```

**PLMIntp[data4, Grafica → True]**

El dominio de la función aproximada es:

$\{\{x_1, -3.14159, 3.14159\}, \{x_2, -3.14159, 3.14159\}\}$

Interpolación por LAGRANGE ...



$0. + 0. x_2 + \ll 29 \gg +$   
 $x_1^7 (0. + 5.18668 \times 10^{-9} x_2 + \ll 18 \gg + 4.76033 \times 10^{-19} x_2^{16}) +$   
 $x_1^6 (0. + \ll 16 \gg + 1.11594 \times 10^{-15} x_2^{14} -$   
 $1.51544 \times 10^{-16} x_2^{15} + 8.7922 \times 10^{-19} x_2^{16})$

## ■ Interpolación Multivariable Trigonométrica

En este caso el espacio vectorial  $\Phi$  está formado por funciones trigonométricas. Dados  $N$  puntos dato,  $x_1, x_2, \dots, x_N$ , la base  $\tau_N \in \Phi_N$  es la siguiente:

$$\tau_N \equiv \begin{cases} \Phi_{2n+1} = \{1, \cos x, \sin x, \dots, \cos nx, \sin nx\}, & \text{si } N \notin 2\mathbb{N}^* \\ \Phi_{2n} = \{1, \cos x, \sin x, \dots, \cos nx\}, & \text{si } N \in 2\mathbb{N}^* \end{cases} \quad (11)$$

con  $\mathcal{R}_i(\tau_N) = \tau_N(x_i)$ , donde  $x_i \in [-\pi, \pi)$ ,  $1 \leq i \leq N$ , son abscisas dadas, las cuales tienen asociados  $f_i$ ,  $1 \leq i \leq N$ , valores dato. Por  $2\mathbb{N}^*$  denominamos los números naturales pares menos el cero.

El planteamiento del problema de interpolación consiste en encontrar en  $\tau_N$  la solución de

$$\tau_N(x_i) = f_i, \quad \forall i = 1, \dots, N \quad (12)$$

Para interpolar funciones de  $h$  variables independientes, con puntos dato tomados en los vértices de una malla  $h$ -dimensional, podemos formar una base generalizada

$\tau_{n_1, \dots, n_h}(x_1, \dots, x_h)$  a partir de las bases  $\tau_{n_i}$ ,  $1 \leq i \leq h$ , del espacio vectorial de las funciones trigonométricas construidas aplicando la regla (13) (sustituyendo  $N = n_i$ ),

$$\tau_{n_1, \dots, n_h}(x_1, \dots, x_h) = \{\tau_{n_1} \otimes \tau_{n_2} \otimes \dots \otimes \tau_{n_h}\}$$

donde  $\tau_{n_1} \otimes \tau_{n_2}$  es el producto de cada una de las funciones de la base de la componente  $x_1$ , por cada una de las funciones base de la componente  $x_2$ . Al operador  $\otimes$  lo denominamos *producto tensorial*, [3] y [4].

Con el fin de estudiar la existencia y unicidad de la solución (14), se reduce la matriz de Gram asociada a la base  $\tau_{n_1, \dots, n_h}(x_1, \dots, x_h)$  a su forma compleja, [2]. De esta forma podemos demostrar que dicha matriz de Gram es regular, si y solo sí, para cada componente  $j = 1, \dots, h$  de los puntos dato, se cumple que  $x_j^k \neq x_j^l$ , si  $k \neq l$ ,  $0 \leq k, l \leq n_j$ .

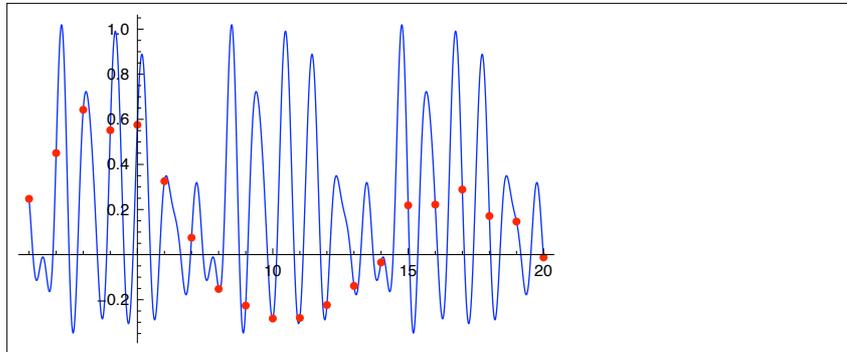
#### □ Algunos Ejemplos de Interpolación Trigonométrica

```
data5 = Table[
  {x, BesselJ[1, x / 2] + 0.2 (Random[] - 1 / 2)}, {x, 1, 20}];
PLMIntp[data5, Grafica -> True]
```

El dominio de la función aproximada es:

{{x1, 1., 20.}}

Interpolación TRIGONOMÉTRICA...



```
0.209244 - 0.107272 Cos[x1] -
0.0134326 Cos[2 x1] + <<27>> + 0.107158 Sin[9 x1]
```

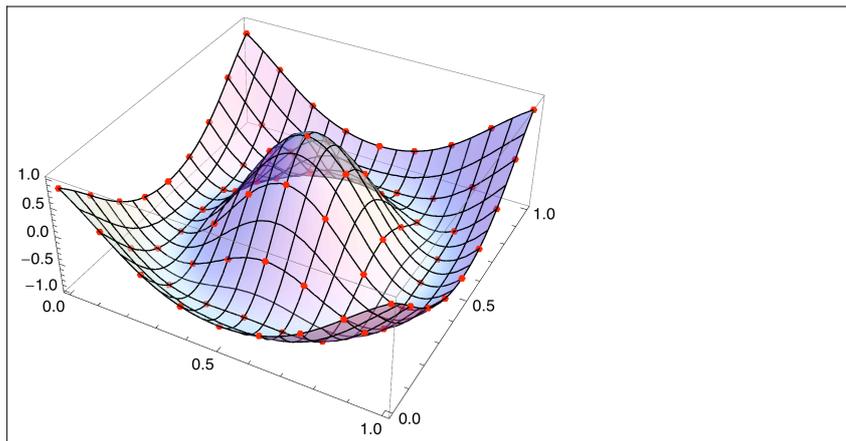
```
data6 = Flatten[
  Table[{x, y, Cos[8 Sqrt[(x - 0.5)^2 + (y - 0.5)^2]}],
  {x, 0, 1, 1 / 8}, {y, 0, 1, 1 / 8}], 1];
```

**PLMIntp[data6, Grafica → True]**

El dominio de la función aproximada es:

$\{\{x1, 0., 1.\}, \{x2, 0., 1.\}\}$

Interpolación TRIGONOMÉTRICA...



237 708. - 340 806. Cos[x1] +  
 <<117>> + 875.905 Sin[4 x1] Sin[4 x2]

### ■ Interpolación Multivariable por Producto Tensorial de Elementos finitos de Hermite de clase $C^m$

En esta sección, abordamos la interpolación en una malla  $h$ -dimensional de puntos dato, cuando, no sólo tenemos los valores de la función (valores dato), sino, que también, tenemos *los valores de las derivadas*. Primeramente, resolvemos el problema de interpolación unidimensional por producto tensorial con derivadas en los puntos dato, introduciendo el concepto de elemento finito. El segundo paso consiste en extender el problema para interpolar funciones dada una malla de puntos con valores conocidos de sus derivadas en los puntos dato.

### □ Interpolación Unidimensional de Hermite de clase $C^m$

Dados  $N+1$  puntos dato  $x_0, x_1, \dots, x_N$  distintos, y dados los valores dato  $f_{x_0}^{(j)}, f_{x_1}^{(j)}, \dots, f_{x_N}^{(j)}$ ,  $j=0, \dots, m$ , valores dato junto con sus derivadas asociadas a cada punto dato, debemos encontrar un polinomio  $p(x)$  tal que,

$$p(x_i) = f_{x_i}, \quad p^{(j)}(x_i) = f_{x_i}^{(j)}, \quad i = 0, \dots, N, \quad y, \quad j = 1, \dots, m \quad (13)$$

Por tanto, existen  $(N+1)(m+1)$  ecuaciones, es decir, necesitamos encontrar una

función del espacio vectorial  $\mathcal{P}_{(N+1)(m+1)-1}$  que cumplan las restricciones de interpolación (15). En [4], se llega al siguiente resultado:

**Teorema 2.** *Existe un único polinomio en el espacio vectorial  $\mathcal{P}_{(N+1)(m+1)-1}$  que satisface las condiciones (15).*

**Definición 1.** *Para cada valor dato  $f_{x_i}^{(j)}$ ,  $i = 0, \dots, N$ , y,  $j = 0, \dots, m$ , se puede construir una función  $u_{x_i}^j(x)$  cuya derivada  $j$ -ésima tenga imagen 1 en el punto dato  $x_i$ , e imagen 0 en el resto de derivadas y de puntos, es decir,*

$$\begin{cases} (u_{x_i}^j)^{(p)}(x_i) = \begin{cases} 1, & p = j \\ 0, & p \neq j \end{cases} \\ (u_{x_i}^j)^{(p)}(x_t) = 0 \end{cases} \quad (14)$$

donde  $p, j = 0, \dots, m$ . A estas funciones las denominamos funciones cardinales.

Si seleccionamos una base  $\varphi$  de funciones cardinales del espacio vectorial  $\mathcal{P}_{(N+1)(m+1)-1}$  obtenemos una matriz generalizada de Gram que es la matriz identidad. Por ejemplo, los polinomios de Lagrange son funciones cardinales.

Si restringimos el problema de interpolación de Hermite a un intervalo  $[x_0, x_1]$ , tenemos  $(2m + 2)$  condiciones, y, la función que debemos encontrar pertenece al espacio  $\mathcal{P}_{2m+1}([x_0, x_1])$ , siendo el problema, así planteado, unisolvente. La solución requiere calcular  $2m + 2$  funciones cardinales,

$$\varphi_{2m+2} = \{u_{x_0}^0, u_{x_0}^1, \dots, u_{x_0}^m, u_{x_1}^0, u_{x_1}^1, \dots, u_{x_1}^m\} \quad (15)$$

Por tanto, la solución al problema de interpolación de Hermite en el intervalo  $[x_0, x_1]$  es

$$p(x) = \sum_{j=0}^m f^{(j)}(x_0) u_{x_0}^j(x) + \sum_{j=0}^m f^{(j)}(x_1) u_{x_1}^j(x) \quad (16)$$

Podemos resolver el problema de interpolación planteado en (13) mediante producto tensorial de funciones cardinales.

**Definition 2.** *Si una función  $f$  pertenece al espacio  $C^m[a, b]$  y, si el valor de  $f^{(i)}(u)$  está dado para algún  $i$ ,  $0 \leq i \leq m$ , y  $u \in [a, b]$ , entonces, el par ordenado  $(u; i)$  se denomina nodo, y,  $f^{(i)}(u)$  es el valor nodal de  $f$  en dicho nodo. Además, dos nodos  $(u; i)$  y  $(v; j)$  son iguales, si, y sólo si,  $u = v$  e  $i = j$ .*

El problema de interpolación unidimensional de un conjunto de puntos dato  $x_0 < x_1 < \dots < x_N$  dados, y un conjunto de valores dato asociados,  $f_1, \dots, f_n$ , podemos atacarlo de forma *global*, o bien, proceder localmente, sub-intervalo a sub-intervalo. Por ejemplo, en cada sub-intervalo  $[x_i, x_{i+1}]$ ,  $i = 0, \dots, N - 1$  podemos obtener un interpolante lineal,  $l_i(x) = [(f_{i+1} - f_i)/(x_{i+1} - x_i)](x - x_i) + f_i$ , el cual, está definido sólo en  $[x_i, x_{i+1}]$ . Uniendo los interpolantes de cada sub-intervalo construimos un único interpolante  $l(x)$  definido en  $[x_0, x_N]$ , cuya existencia está demostrada en [4].

Las componentes de cada problema de interpolación, por separado, determinan un

*elemento finito*, y son las siguientes:

1. Un intervalo  $[x_i, x_{i+1}]$ ;
2. Los nodos  $(x_i; 0)$  y  $(x_{i+1}; 0)$ ; y
3. El espacio vectorial  $\mathcal{P}_1$ .

Por ejemplo, podemos resolver el problema de interpolación de Hermite, planteado en (13), pero particularizando en el intervalo  $[0, 1]$ , y, con valores de la función, y, sólo de su primera derivada,  $f_0, f_1, f_0^{(1)}, f_1^{(1)}$ , en los puntos dato 0 y 1. En este caso el elemento finito es:

1. El intervalo  $[0, 1]$ ;
2. Los nodos  $(0; 0), (0; 1), (1; 0), (1; 1)$ ; y
3. El espacio vectorial  $\mathcal{P}_3$ .

Aplicando (16) obtenemos las siguientes funciones cardinales, una por nodo:

$$\begin{aligned} u_0^0(x) &= (x-1)^2(2x+1), & u_1^0(x) &= x^2(3-2x) \\ u_0^1(x) &= x(x-1)^2, & u_1^1(x) &= x^2(x-1) \end{aligned} \quad (17)$$

En el problema de interpolación de Hermite planteado en (13), cada sub-intervalo  $[x_i, x_{i+1}]$ , tiene el siguiente elemento finito:

1. El intervalo  $[x_i, x_{i+1}]$ ;
2. Los nodos  $(x_i; j)$  y  $(x_{i+1}; j)$ ,  $j = 0, \dots, m$ ; y
3. El espacio vectorial  $\mathcal{P}_{2m+1}$ .

Las funciones cardinales de cada uno de estos nodos,  $2(m+1)N$ , podemos definir las a partir de (14) mediante la siguiente *aplicación afín*:

$$\begin{aligned} F : [x_i, x_{i+1}] &\rightarrow [0, 1] \\ x &\rightarrow ax + b = (x - x_i)/(x_{i+1} - x_i) \end{aligned} \quad (18)$$

Dualmente a (17) para el intervalo  $[x_0, x_1]$ , en el sub-intervalo  $[x_i, x_{i+1}]$  se pueden definir las funciones base cardinales siguientes:

$$\varphi_{2m+2} = \left\{ u_{i,0}^0, u_{i,0}^1, \dots, u_{i,0}^m, u_{i,1}^0, u_{i,1}^1, \dots, u_{i,1}^m \right\}$$

con,

$$u_{i,0}^j = (x_{i+1} - x_i)^j u_0^j \left[ \frac{x - x_i}{x_{i+1} - x_i} \right], \quad u_{i,1}^j = (x_{i+1} - x_i)^j u_1^j \left[ \frac{x - x_i}{x_{i+1} - x_i} \right] \quad (19)$$

para  $j = 0, \dots, m$ .

Por tanto, para resolver el problema de interpolación (13), con  $N$  sub-intervalos, se puede definir para cada nodo  $(x_i; j)$ ,  $i = 1, \dots, (N-1)$  y  $j = 0, \dots, m$ , la función cardinal siguiente:

$$w_i^j(x) = \begin{cases} u_{i,1}^j(x), & x \in [x_{i-1}, x_i] \\ u_{i,0}^j(x), & x \in [x_i, x_{i+1}] \\ 0, & \text{resto} \end{cases} \quad (20)$$

excepto en los nodos extremos,  $(x_0; j)$  y  $(x_N; j)$ , con  $j = 0, \dots, m$ , donde construimos la funciones cardinales siguientes:

$$w_0^j(x) = \begin{cases} u_{0,0}^j, & x \in [x_0, x_1] \\ 0, & \text{resto} \end{cases}, \quad w_N^j(x) = \begin{cases} u_{N,1}^j, & x \in [x_{N-1}, x_N] \\ 0, & \text{resto} \end{cases} \quad (21)$$

Aplicando (16) con las funciones base (20) y (21) obtenemos la expresión:

$$P(x) = \sum_{i=0}^N \sum_{j=0}^m f^{(j)}(x_i) w_i^j(x) \quad (22)$$

que es la solución, única (Teorema 2), del problema de interpolación unidimensional de Hermite (13) de clase  $C^m$  utilizando elementos finitos.

## □ Interpolación en una Malla de Puntos

El planteamiento del problema en una malla de puntos es el siguiente:

Dado el conjunto cerrado  $\bar{D}$  de puntos dato  $(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h})$ , y sus valores dato asociados  $f(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h})$ , así como, también dados los valores de las derivadas  $\frac{\partial^k f}{\partial x_1^{j_1} \partial x_2^{j_2} \dots \partial x_h^{j_h}}(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h})$ ,  $\{k = 1, \dots, m; j_1 + \dots + j_h = k\}$ , entonces, el problema de interpolación consiste en encontrar una función  $P \in C^m(\bar{D})$ , tal que,

$$\frac{\partial^k P}{\partial x_1^{j_1} \partial x_2^{j_2} \dots \partial x_h^{j_h}}(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}) = \frac{\partial^k f}{\partial x_1^{j_1} \partial x_2^{j_2} \dots \partial x_h^{j_h}}(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}), \quad (23)$$

con  $i_1 = 0, 1, \dots, n_1$ ,  $i_2 = 0, 1, \dots, n_2$ ,  $\dots$ ,  $i_h = 0, 1, \dots, n_h$ , y,  $\{k = 0, \dots, m; j_1 + \dots + j_h = k\}$ .

Abordamos este problema de interpolación multidimensional extendiendo la noción de *nodo*, y la tecnica de interpolación unidimensional de Hermite con elementos finitos.

**Definition 3.** Si una función  $f$ , definida sobre  $\bar{D}$ , pertenece al espacio de las funciones  $C^{l_1 + \dots + l_h}(\bar{D})$ ,  $(l_1 + \dots + l_h = m)$ , donde  $\bar{D}$  es una malla de puntos dato, y, si el valor de  $\partial^{j_1 + \dots + j_h} f(x_1^{i_1}, \dots, x_h^{i_h}) / \partial x_1^{j_1} \dots \partial x_h^{j_h}$  está dado para algún  $j_1, \dots, j_h$ , con  $(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}) \in \bar{D}$ , entonces, se denomina *nodo* a  $(x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}; j_1, \dots, j_h)$  y, llamamos *valor nodal* de  $f$  en dicho nodo al valor de la derivada parcial  $\partial^{j_1 + \dots + j_h} f(x_1^{i_1}, \dots, x_h^{i_h}) / \partial x_1^{j_1} \dots \partial x_h^{j_h}$ .

**Teorema 3.** Sean  $M_{r_i}$ ,  $r_i = 0, \dots, N_i$ , para  $i = 1, 2, \dots, h$ , conjuntos de nodos distintos dados, contenidos en los intervalos  $[a_i, b_i]$ , y, que engloban, al menos, hasta la  $l_i$ -ésima derivada,  $i = 1, 2, \dots, h$ , respectivamente. Sean  $\mathcal{F}_i$  familias de funciones de  $C^{l_i}[a_i, b_i]$ ,  $i = 1, 2, \dots, h$ , tales que, existe en cada uno de dichos espacios una única función  $f_i \in \mathcal{F}_i$  que tome los valores nodales dados. Entonces, existe un único interpolante  $P$  sobre  $[a_i, b_i]$ ,  $i = 1, \dots, h$ , cuyos nodos son  $P_{r_1, \dots, r_h}$  con  $r_i = 0, \dots, N_i$ ,  $i = 1, \dots, h$  obtenidos de combinar todos los nodos  $M_{r_i}$  de la siguiente forma:

Si  $M_{r_p} = (x_p^{i_p}; j_p)$ ,  $\{0 \leq i_p \leq n_p, 0 \leq j_p \leq l_p\}$ , para  $p = 1, 2, \dots, h$ , entonces,

$$P_{r_1, \dots, r_h} = (x_1^{i_1}, x_2^{i_2}, \dots, x_h^{i_h}; j_1, \dots, j_h)$$

y, el valor nodal de  $P$  en los nodos  $P_{r_1, \dots, r_h}$  es, entonces,

$$\frac{\partial^{j_1 + \dots + j_h} f(x_1^{i_1}, \dots, x_h^{i_h})}{\partial x_1^{j_1} \dots \partial x_h^{j_h}}$$

Además, si las funciones base cardinales para la interpolación en  $\mathcal{F}_i$ , son  $w_{r_i}$ ,  $r_i = 0, \dots, N_i$ ,  $i = 1, \dots, h$ , y, si los valores nodales de  $P$  en  $P_{r_1, \dots, r_h}$  se denominan  $v_{r_1, \dots, r_h}$ , entonces, la expresión para el interpolante  $P$  es

$$P(x_1, \dots, x_h) = \sum_{r_1=0}^{N_1} \dots \sum_{r_h=0}^{N_h} v_{r_1, \dots, r_h} w_{r_1}(x_1) \dots w_{r_h}(x_h)$$

donde los  $(N_1 + 1)(N_2 + 1) \dots (N_h + 1)$  productos  $w_{r_1}(x_1) \dots w_{r_h}(x_h)$  son las funciones cardinales para el problema de interpolación con nodos  $P_{r_1, \dots, r_h}$ .

Por lo tanto, se puede definir una función  $q \in \mathcal{F}$ , con  $\mathcal{F} = \mathcal{F}_1 \otimes \mathcal{F}_2 \otimes \dots \otimes \mathcal{F}_h$ , producto tensorial de las  $h$  familias de funciones  $\mathcal{F}_j$ ,  $j = 1, \dots, h$ , tal que,

$$\left\{ \begin{array}{l} q_{i_1, \dots, i_h}^{j_1, \dots, j_h}(x_1, \dots, x_h) = w_{i_1}^{j_1}(x_1) \dots w_{i_h}^{j_h}(x_h) \\ i_1 = 0, 1, \dots, n_1, i_2 = 0, 1, \dots, n_2, \dots, i_h = 0, 1, \dots, n_h \\ j_1 + \dots + j_h \leq m \end{array} \right. \quad (24)$$

que toma el valor 1 sólo en el nodo  $P_{r_1, \dots, r_h}$ , y el valor 0 en el resto de los nodos

A partir del Teorema 3 podemos obtener la solución del problema de interpolación multidimensional planteado en (25), y expresarla de la siguiente forma:

$$P(x_1, \dots, x_h) = \sum_{i_1, \dots, i_h=0}^{n_1, \dots, n_h} \sum_{j_1 + \dots + j_h=0}^m \frac{\partial^{j_1 + \dots + j_h} f(x_1^{i_1}, \dots, x_h^{i_h})}{\partial x_1^{j_1} \dots \partial x_h^{j_h}} q_{i_1, \dots, i_h}^{j_1, \dots, j_h}(x_1, \dots, x_h) \quad (25)$$

#### □ Algunos Ejemplos de Uso

Podemos ver la diferencia con el método de interpolación trigonométrica,

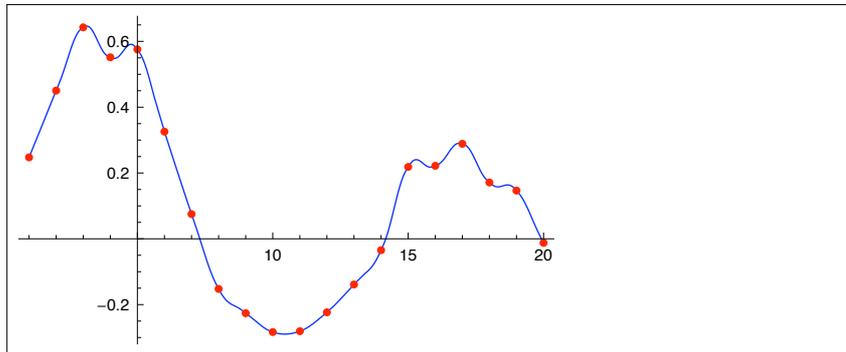
```
PLMIntp[data5, Grafica -> True]
```

El dominio de la función aproximada es:

```
{{x1, 1., 20.}}
```

EL RESULTADO ESTARÁ EN LA  
FUNCIÓN GLOBAL 'PteFH[{x1,...,xn}]'.

Interpolando con SPLINES clase 2 ...



Evaluar la variable global 'PteFH[{punto\_dato}]'

Podemos calcular el valor de interpolante en el punto  $x_1 = 10$ .

```
PteFH[{10}]
```

```
-0.28341
```

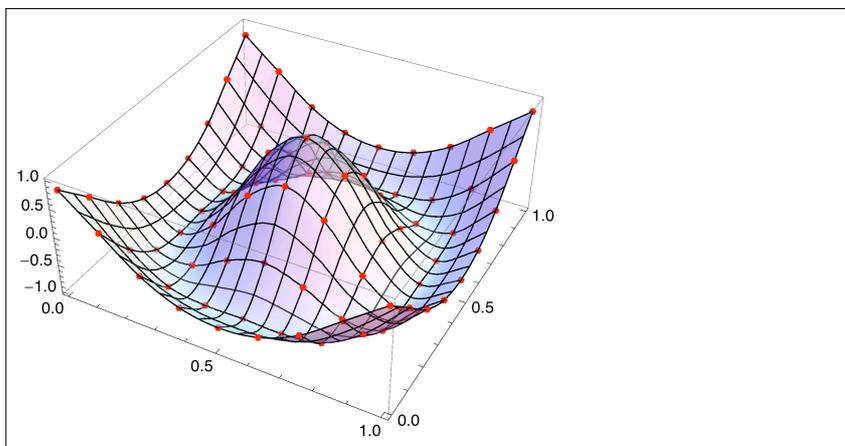
**PLMIntp[data6, Grafica → True]**

El dominio de la función aproximada es:

`{{x1, 0., 1.}, {x2, 0., 1.}}`

EL RESULTADO ESTARÁ EN LA  
FUNCIÓN GLOBAL 'PteFH[{x1,...,xn}]'.

Interpolando con SPLINES clase 1 ...



Evaluar la variable global 'PteFH[{punto\_dato}]'

`puntoj = {Random[Real, {0, 1}], Random[Real, {0, 1}]}`

`{0.899678, 0.458773}`

`PteFH[puntoj]`

`-0.981863`

## ■ Splines de Orden $(m, s)$

En la resolución de un problema de interpolación podemos buscar una solución que, de alguna forma, sea *óptimo*. En [4] se discute un criterio basado en la minimización del siguiente funcional,

$$J(S) = \int_a^b \left[ \left( \frac{d^2}{dx^2} S(x) \right)^2 dx \right] \quad (26)$$

donde  $S(x)$  es una función *spline* para  $x \in [a, b]$ . El funcional  $J(S)$  representa, aproximadamente, la energía de deformación de una *tabla flexible*, tal y como se estudia en [4]. El valor de la energía de deformación, en cualquier punto del spline, dependerá

de la curvatura del spline en dicho punto. La búsqueda del interpolante parte de la condición de minimizar el funcional (26).

En el caso de los  $(m, s)$ -splines el criterio que se emplea es la minimización de la *semi-norma*  $|v|_{m,s}$ , tal y como indica en [5], para un conjunto de puntos dato.

Al contrario de los métodos de interpolación vistos hasta ahora, con  $(m, s)$ -splines no es necesario que los puntos dato estén en una malla para que el problema sea unisolvente. Sí será necesario que los puntos dato sean distintos entre sí.

Considerando que:

- Todos los espacios son *reales*.
- Para todo  $l \in \mathbb{N}$ , se designa por  $\mathcal{P}_l$  al espacio vectorial de las funciones polinómicas de  $n$  variables,  $x_1, x_2, \dots, x_n$ , de grado  $\leq l$ , es decir,

$$\{p \in \mathcal{P}_l, p(x) = \sum_{\alpha_1 + \dots + \alpha_n \leq l} x_1^{\alpha_1} \cdots x_n^{\alpha_n}; x \in \mathbb{R}^n\}$$

- Para todo  $l \in \mathbb{N}$  y todo abierto no vacío  $\Omega$  de  $\mathbb{R}^n$ , designamos por  $\mathcal{P}_l(\Omega)$  al sub-espacio vectorial de las funciones  $\mathcal{P}_l$  definidas en  $\Omega$ .

#### □ Aproximación con $(m, s)$ - Splines

Sean  $n, m \in \mathbb{N}^*$ ,  $s \in \mathbb{R}$ , tales que,

$$-m + n/2 < s < n/2 \quad (27)$$

y, dados

- un abierto no vacío, acotado y conexo  $\Omega \in \mathbb{R}^n$ ;
- un sub-conjunto  $\mathcal{D} \subset \mathbb{R}_+^*$ , admitiendo el 0 como punto de acumulación; y
- para todo  $d \in \mathcal{D}$ , el conjunto  $A^d$  de  $N$  puntos ordenados de  $\Omega$ , tales que,

$$\sup_{x \in \Omega} \delta(x, A^d) = d$$

donde  $\delta$  es la distancia euclídea en  $\mathbb{R}^n$ . Se verifica que para un  $d$  lo bastante pequeño, el conjunto  $A^d$  es  $\mathcal{P}_{m-1}$  - unisolvente.

Si se conserva la relación (27), denotando por  $\langle \cdot \rangle$  a la norma euclídea, y, por  $\langle \cdot, \cdot \rangle$  al producto escalar euclídeo (ambos en  $\mathbb{R}^n$ ), entonces, para todo  $\varepsilon > 0$ , para todo  $d \in \mathcal{D}$ , y, para todo  $\beta^d \in \mathbb{R}^n$  se puede poner

$$\forall v \in X^{m,s}, J_\varepsilon^d(v) = \langle \rho^d v - \beta^d \rangle^2 + \varepsilon |v|_{m,s}^2$$

donde  $X^{m,s}$  es un espacio funcional de *semi-Hilbert*, cualquier  $v \in X^{m,s}$ , y  $\rho^d$  es un operador, definidos en [5].

Se denomina *spline de aproximación de orden*  $(m, s)$  relativo a  $A^d$ ,  $\beta^d$ , y  $\varepsilon$ , a todo  $\sigma_\varepsilon^d$ , tal que,

$$\begin{cases} \sigma_\varepsilon^d \in X^{m,s} \\ \forall v \in X^{m,s}, J_\varepsilon^d(\sigma_\varepsilon^d) \leq J_\varepsilon^d(v) \end{cases} \quad (28)$$

El criterio utilizado para encontrar los  $(m, s)$  – splines de aproximación, consiste en minimizar la función  $J_\varepsilon^d$ , que contiene un primer término en el que interviene una solución de aproximación por mínimos cuadrados, y, un segundo término en el que aparece la semi-norma  $|v|_{m,s}$  elevada al cuadrado.

En [5] se demuestra que (28) admite una solución única, y podemos obtenerla resolviendo el sistema de ecuaciones siguiente:

$$\begin{cases} \varepsilon C^* \lambda_b + \sum_{a \in A^d} \lambda_a K_{2m+2s-n}(b-a) + \sum_{j=1}^M C_j p_j(b) = \beta_b^d, & \varepsilon > 0 \\ \sum_{a \in A^d} \lambda_a p_j(a) = 0, & \forall j = 1, \dots, M \end{cases} \quad (29)$$

donde  $\{p_j, 1 \leq j \leq M\}$  es una base del espacio  $\mathcal{P}_{m-1}$ ,  $M = \dim \mathcal{P}_{m-1}$ , y donde  $\lambda_a, a \in A^d$ ,  $\lambda_b, b \in A^d$ , y  $C_j, 1 \leq j \leq M$ , son coeficientes a determinar. La función  $K_{2m+2s-n}(x)$  es

$$K_{2m+2s-n}(x) = \begin{cases} |x|^{2m+2s-n} & , \text{ si } 2m+2s \neq 2l, l \in \mathbb{N}^* \\ |x|^{2m+2s-n} \log |x| & , \text{ si } 2m+2s = 2l, l \in \mathbb{N}^* \end{cases}$$

La constante  $C^*$  se puede explicitar como

$$C^* = \begin{cases} \frac{(2\pi)^{2m}}{C_1} & , \text{ si } 2m+2s \neq 2l, l \in \mathbb{N}^* \\ \frac{(2\pi)^{2m}}{C_2} & , \text{ si } 2m+2s = 2l, l \in \mathbb{N}^* \end{cases}$$

donde,

$$C_1 = \frac{\pi^{(2m+2s-n/2)} \Gamma\left(\frac{n}{2} - m - s\right)}{\Gamma(m+s)} \quad , \quad C_2 = \frac{2\pi^{(2m+2s-n/2)} (-1)^{(m+s-n/2+1)}}{\Gamma(m+s) \left(m+s-\frac{n}{2}\right)!}$$

Por tanto, para los  $(m, s)$  – splines de aproximación tienen una matriz de coeficientes cuya diagonal tiene un valor  $\varepsilon C^*$ . Si el valor de  $\varepsilon$  es cero, obtenemos la función interpolante  $\sigma^d$ .

□ Aplicando el método de  $(m, s)$  – splines

Ajuste de la función de BesselJ alterada con un valor aleatorio.

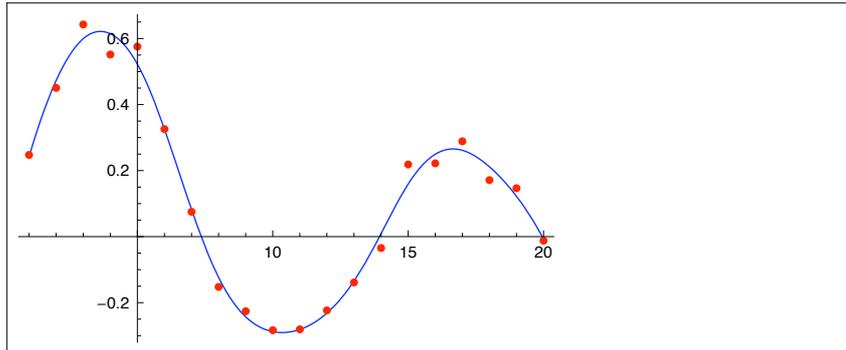
**PLMIntp[data5, Grafica → True]**

El dominio de la función aproximada es:

$\{\{x1, 1., 20.\}\}$

Estoy APROXIMANDO con SPLINES (m,s); Epsilon= 0.2 ...

s= 0 m= 3



$$-3.17635 + \ll 30 \gg + 0.578783 x1 - 0.0362353 x1^2$$

El interpolante para los mismos datos es el siguiente:

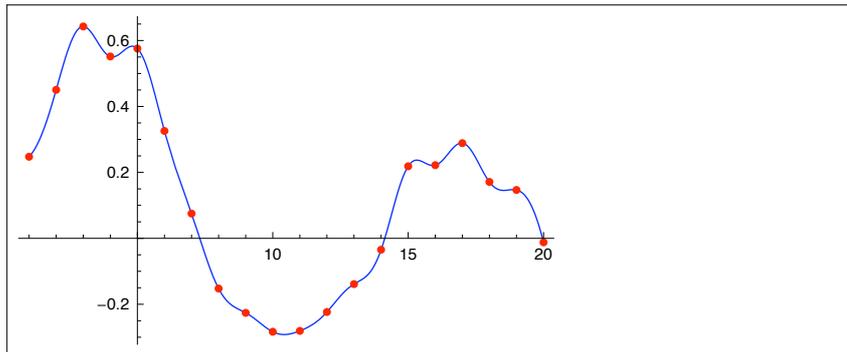
**PLMIntp[data5, Grafica → True]**

El dominio de la función aproximada es:

$\{\{x1, 1., 20.\}\}$

Estoy INTERPOLANDO con SPLINES (m,s) ...

s= 0 m= 3



$-31.5065 + \ll 30 \gg + 3.24122 x1 - 0.0230485 x1^2$

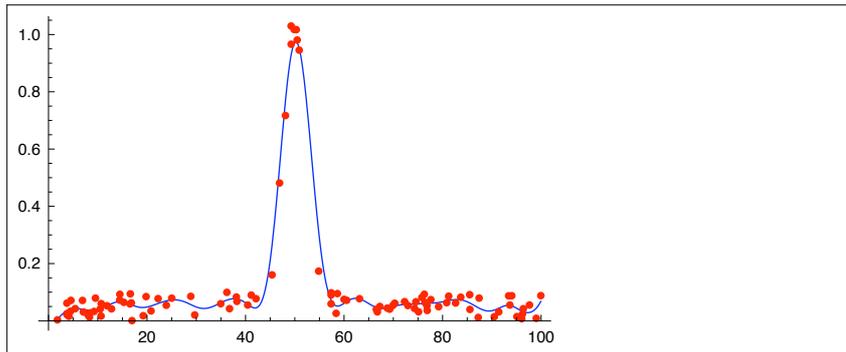
**PLMIntp[dataGauss, Grafica → True]**

El dominio de la función aproximada es:

`{{x1, 1.81055, 99.943}}`

Estoy APROXIMANDO con SPLINES (m,s); Epsilon= 7 ...

s= 0 m= 3



$$32.1017 - 0.0000121061 \left( (-99.943 + \langle\langle 2 \rangle\rangle)^2 \right)^{5/2} + \langle\langle 148 \rangle\rangle + 0.00178357 x1^2$$

La función Sin[xy] que interpolabamos con Lagrange, ahora la podemos ajustar con splines – (m, s).

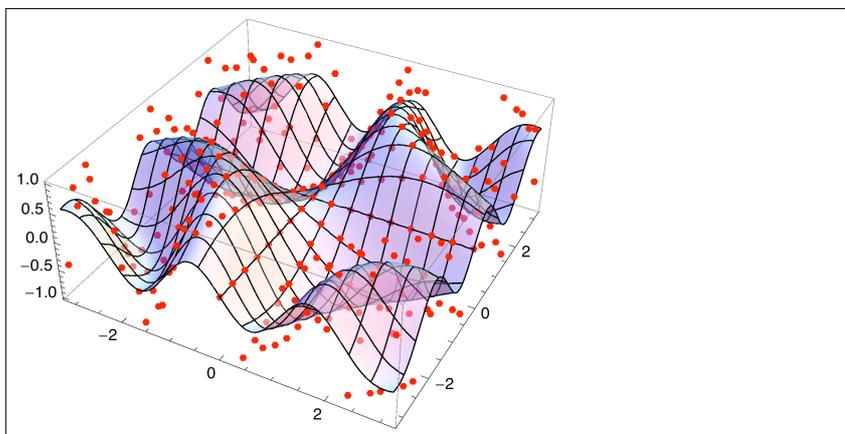
**PLMIntp[data4, Grafica → True]**

El dominio de la función aproximada es:

$\{\{x1, -3.14159, 3.14159\}, \{x2, -3.14159, 3.14159\}\}$

Estoy APROXIMANDO con SPLINES (m,s); Epsilon= 0.1 ...

s= 0 m= 3



$$-7.3491 \times 10^{-12} - 5.05333 \times 10^{-12} x_1 + \ll 436 \gg +$$

$$0.386373 \left( (\ll 18 \gg + x_1)^2 + (\ll 18 \gg + \ll 2 \gg)^2 \right)^2$$

$$\text{Log} \left[ \sqrt{(3.14159 + x_1)^2 + (3.14159 + x_2)^2} \right]$$

## ■ Conclusión

En este artículo hemos demostrado la simplicidad del paquete *Intpak* agrupando varios métodos de aproximación multivariable con *Mathematica*. Su implementación conlleva un esfuerzo especial para que la utilización del software por parte del usuario sea lo más fácil y transparente posible. *Mathematica* proporciona una solución excelente gracias a su lenguaje de programación de alto nivel y sus capacidades de cómputo numérico.

La utilidad de los métodos de aproximación va más allá de la generación, asistida por computadora, de curvas y superficies. En algunas situaciones resulta más conveniente encontrar una función de ajuste que un interpolante. Por ejemplo, analizando un problema donde los datos no sean exactos o estén tomados de algún experimento.

El paquete *Intpak* puede ampliarse fácilmente con nuevos métodos gracias a su estructura modular. También pueden incorporarse futuras mejoras en la representación

gráfica de funciones de aproximación, por ejemplo, para comparar métodos de forma más sencilla, o, realizar demostraciones con *Mathematica*.

## ■ Referencias

- [1] G.R. Lindfield and J.E.T. Penny, *Microcomputers in Numerical Analysis*, West Sussex: Ellis Horwood Series in Mathematics and its Applications, 1989.
- [2] M. Gasca, *Cálculo Numérico (Unidad Didáctica /1)*, Madrid: Universidad Nacional de Educación a Distancia. Impresos y Revistas S.A., 1991.
- [3] G. Hämmerlin and K.-H. Hoffmann, *Numerical Mathematics*, New York: Springer Verlag, 1991.
- [4] P. Lancaster and K. Salkauskas, *Curve and Surface Fitting. An Introduction*, London: Academic Press, 1986.
- [5] M.C. López de Silanes and R. Arcangéli, "Estimations de l'erreur d'approximation par splines d'interpolation et d'ajustement d'ordre (m,s)," *Numerische Mathematik*, **N.56**, Springer Verlag 1989 pp. 449-467.