Universidad Politécnica de Madrid–Escuela Técnica Superior de Ingenieros Industriales
Grado en Ingeniería en Tecnologías Industriales. Curso 2016-2017-3º
Matemáticas de Especialidad–Ingeniería Eléctrica

Mínimos cuadrados no lineales

José Luis de la Fuente O'Connor
jldelafuente@etsii.upm.es
joseluis.delafuente@upm.es

Índice

Definición del problema

- Los problemas no lineales de mínimos cuadrados reales surgen de modelos matemáticos que estudian el comportamiento de sistemas económicos, sociales y físicos y se quiere aproximarlos a formulaciones más estándar o fáciles de manejar, aunque no lineales en muchos casos.

- Se trata de en encontrar el mínimo de la suma de los cuadrados de m funciones no lineales; es decir,

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimizar}}\, f(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|_2^2$$

donde el vector de residuos[1] es $\boldsymbol{r}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^m = [r_1(\boldsymbol{x}), \ldots, r_m(\boldsymbol{x})]^T$ y cada $r_i(\boldsymbol{x})$, $i = 1, \ldots, m$, $m \geq n$, es una función no lineal de $\mathbb{R}^n$ en $\mathbb{R}$.

[1] Si $m = n$ se tiene un sistemas de ecuaciones no lineales como los que hemos estudiado.

- El problema real surge de la imposibilidad de encontrar la solución al sistema de ecuaciones $\boldsymbol{r}(\boldsymbol{x}) = \boldsymbol{0}$ y se quiere adaptar una pseudosolución que mejor la aproximase —de existir— de acuerdo con la norma euclídea.

- **Ejemplo** El ajuste de funciones no lineales a datos diversos. Se trata de aproximar una función $f(\boldsymbol{x}, t)$ no lineal a unos datos, definidos por ejemplo por un par y_i (valor) y t_i (tiempo), (y_i, t_i), $i = 1, \ldots, m$.

- Si $r_i(\boldsymbol{x})$ representa el error en la predicción que hace el modelo de la observación i,

$$r_i(\boldsymbol{x}) = y_i - f(\boldsymbol{x}, t_i), \quad i = 1, \ldots, m,$$

y se quiere minimizar la suma de los cuadrados de las desviaciones entre los valores reales y los predichos con el modelo, se llega a un problema del tipo

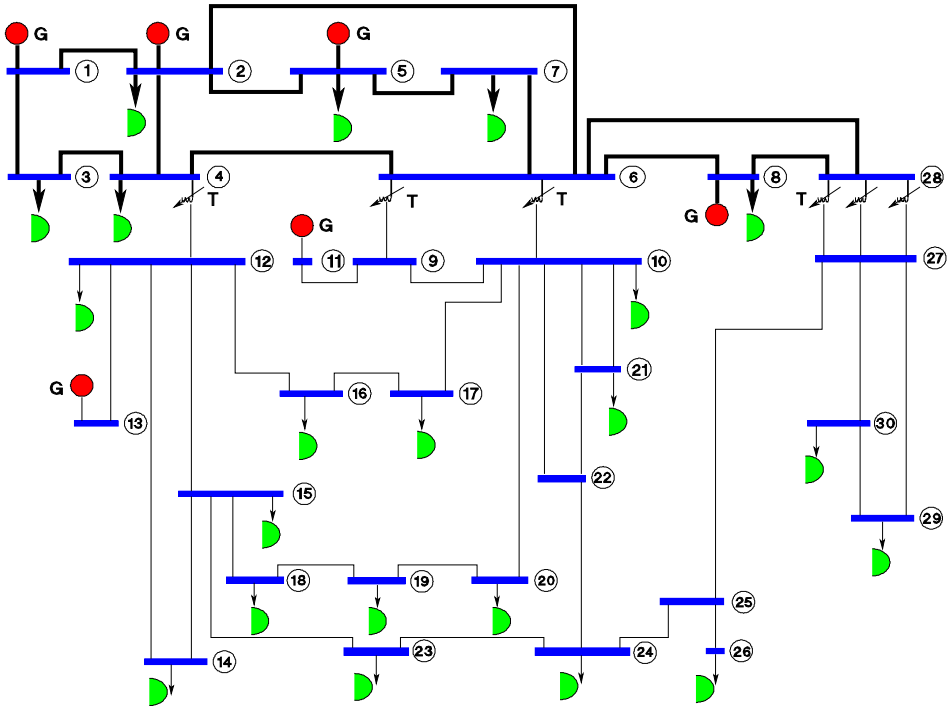
$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimizar}} \ \tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{f}(\boldsymbol{x})\|_2^2$$

Estimación del estado de sistemas eléctricos

- La estimación del estado es el proceso por el cual se determina el valor del vector de variables que rigen un sistema a partir de unos datos proporcionados por medidas efectuadas al mismo.

- Estas medidas no se pueden realizar con precisión absoluta, debido a la imperfección operativa de los aparatos que las registran. Para aumentar la certeza sobre sus resultado se disponen con un grado notable de redundancia.

- El proceso matemático de la estimación se basa en maximizar o minimizar unos criterios estadísticos determinados. El criterio más usado es el de minimizar la suma de los cuadrados de las desviaciones entre los valores reales —medidas— y los estimados.



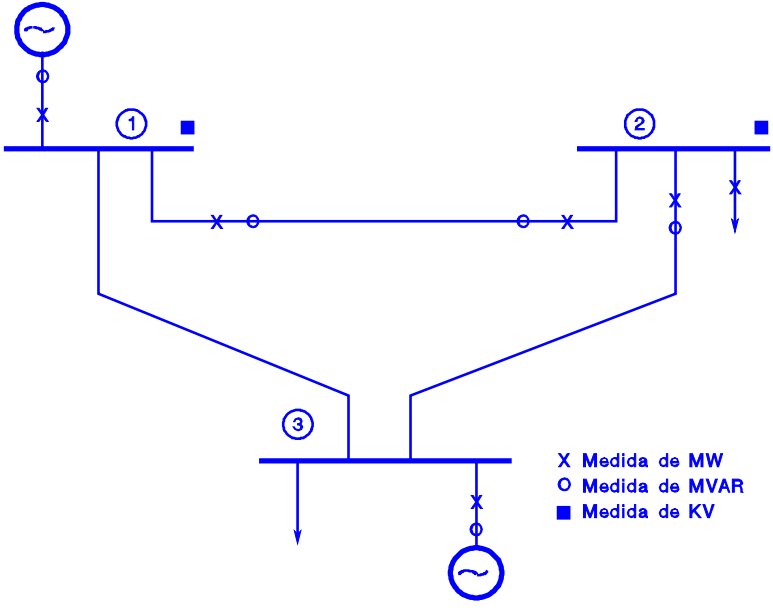
- En la operación, análisis y planificación de sistemas eléctricos de energía, uno de los asuntos de más relevancia técnica y económica es el de la estimación del estado de funcionamiento del sistema de generación y transporte.

- Para estimar el estado del sistema se instalan en su red física unos aparatos de medida que proporcionan el valor de diversas magnitudes de funcionamiento: tensiones en diversos puntos, flujos de potencia activa y reactiva por elementos de transporte, potencias activa y reactiva inyectadas, etc.

- Si todas estas medidas fuesen perfectas en un estado de operación concreto, las relaciones matemáticas que modelizan las leyes físicas que rigen su funcionamiento permitirían determinar la solución única de ese estado.

- Los errores aleatorios que incorporan los aparatos de medida introducen una incompatibilidad matemática en aquellas relaciones, por lo que el cálculo de la solución exacta no es posible teniendo que sustituirse por una estimación de la más probable.

- Con el fin de aumentar la bondad de la estimación, así como poder identificar mediciones erróneas, el número de medidas que se efectúa suele ser redundante: bastante superior al estrictamente necesario para determinar el estado de funcionamiento real. Ejemplo:



- Recordemos las expresiones

$$P_i = |V_i|^2 \sum_{\substack{j=1 \\ j\neq i}}^{n} \left(G_{p_{ij}} + G_{s_{ij}}\right) - |V_i| \sum_{\substack{j=1 \\ j\neq i}}^{n} |V_j| \left[G_{s_{ij}} \cos(\theta_i - \theta_j) + B_{s_{ij}} \operatorname{sen}(\theta_i - \theta_j)\right]$$

$$Q_i = -|V_i|^2 \sum_{\substack{j=1 \\ j\neq i}}^{n} \left(B_{p_{ij}} + B_{s_{ij}}\right) - |V_i| \sum_{\substack{j=1 \\ j\neq i}}^{n} |V_j| \left[G_{s_{ij}} \operatorname{sen}(\theta_i - \theta_j) - B_{s_{ij}} \cos(\theta_i - \theta_j)\right]$$

donde: V_i es el módulo de la tensión en el nudo i;
θ_i el argumento de la tensión en el nudo i;
$G_{s_{ij}}$ la conductancia serie (constante) de la línea que une el nudo i con el nudo j;
$G_{p_{ij}}$ la conductancia a tierra (constante) de la línea que une el nudo i con el j;
$B_{s_{ij}}$ la susceptancia serie (constante) de la línea que une el nudo i con el nudo j; y
$B_{p_{ij}}$ la susceptancia a tierra (constante) de la línea que une el nudo i con el j.

- Los flujos de potencias activa y reactiva entre dos nudos i y j de una red están dados por las relaciones

$$P_{ij} = |V_i|^2 G_{s_{ij}} - |V_i||V_j| G_{s_{ij}} \cos(\theta_i - \theta_j) - |V_i||V_j| B_{s_{ij}} \operatorname{sen}(\theta_i - \theta_j) + |V_i|^2 G_{p_{ij}}$$
$$Q_{ij} = -|V_i|^2 B_{s_{ij}} - |V_i||V_j| G_{s_{ij}} \operatorname{sen}(\theta_i - \theta_j) + |V_i||V_j| B_{s_{ij}} \cos(\theta_i - \theta_j) - |V_i|^2 B_{p_{ij}}.$$

- En términos matemáticos, si se tiene una muestra $b_1, b_2, \ldots, b_m$ que define una medida de todos los aparatos, el sistema de ecuaciones que relaciona estas mediciones con las variables de estado $x_1, x_2, \ldots, x_n$, se puede expresar como

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= b_1 \\ f_2(x_1, x_2, \ldots, x_n) &= b_2 \\ &\vdots \\ f_m(x_1, x_2, \ldots, x_n) &= b_m, \end{aligned}$$

  donde $m \gg n$. Se supone que cada uno de los elementos de la función vectorial $\boldsymbol{f}(\boldsymbol{x})$ es exacto.

- Este sistema, debido a la imprecisión de los aparatos, suele ser matemáticamente incompatible, aunque esta incompatibilidad suele ser muy pequeña al ser pequeñas esas imprecisiones.

- Para el ejemplo de tres nudos de la figura anterior, tomando $\theta_1 = 0$ como referencia de ángulos, los parámetros que definen el sistema son los de esta tabla.

$\boldsymbol{b}$	$\boldsymbol{x}$	$\boldsymbol{f(x)}$
		V_1
		V_2
V_1		$V_1^2 \sum_{j=2,3} (G_{p_{1j}} + G_{s_{1j}}) - V_1 \sum_{j=2,3} V_j (G_{1j} \cos(\theta_1 - \theta_j) + B_{1j} \operatorname{sen}(\theta_1 - \theta_j))$
V_2		$-V_1^2 \sum_{j=2,3} (B_{p_{1j}} + B_{s_{1j}}) - V_1 \sum_{j=2,3} V_j (G_{1j} \operatorname{sen}(\theta_1 - \theta_j) - B_{1j} \cos(\theta_1 - \theta_j))$
P_1		$V_2^2 \sum_{j=1,3} (G_{p_{2j}} + G_{s_{2j}}) - V_2 \sum_{j=1,3} V_j (G_{2j} \cos(\theta_2 - \theta_j) + B_{2j} \operatorname{sen}(\theta_2 - \theta_j))$
Q_1		$-V_2^2 \sum_{j=1,3} (B_{p_{2j}} + B_{s_{2j}}) - V_2 \sum_{j=1,3} V_j (G_{2j} \operatorname{sen}(\theta_2 - \theta_j) - B_{2j} \cos(\theta_2 - \theta_j))$
P_2	V_1	$V_3^2 \sum_{j=1,2} (G_{p_{3j}} + G_{s_{3j}}) - V_3 \sum_{j=1,2} V_j (G_{3j} \cos(\theta_3 - \theta_j) + B_{3j} \operatorname{sen}(\theta_3 - \theta_j))$
P_3	V_2	$-V_3^2 \sum_{j=1,2} (B_{p_{3j}} + B_{s_{3j}}) - V_3 \sum_{j=1,2} V_j (G_{3j} \operatorname{sen}(\theta_3 - \theta_j) - B_{3j} \cos(\theta_3 - \theta_j))$
Q_3	θ_2	$V_1^2 G_{s_{12}} - V_1 V_2 (G_{s_{12}} \cos(\theta_1 - \theta_2) + B_{s_{12}} \operatorname{sen}(\theta_1 - \theta_2)) + V_1^2 G_{p_{12}}$
P_{12}	V_3	$-V_1^2 B_{s_{12}} - V_1 V_2 (G_{s_{12}} \operatorname{sen}(\theta_1 - \theta_2) - B_{s_{12}} \operatorname{sen}(\theta_1 - \theta_2)) - V_1^2 B_{p_{12}}$
Q_{12}	θ_3	$V_2^2 G_{s_{21}} - V_1 V_2 (G_{s_{21}} \cos(\theta_2 - \theta_1) + B_{s_{21}} \operatorname{sen}(\theta_2 - \theta_1)) + V_2^2 G_{p_{21}}$
P_{21}		$-V_2^2 B_{s_{21}} - V_1 V_2 (G_{s_{21}} \operatorname{sen}(\theta_2 - \theta_1) - B_{s_{21}} \operatorname{sen}(\theta_2 - \theta_1)) - V_2^2 B_{p_{21}}$
Q_{21}		$V_2^2 G_{s_{23}} - V_2 V_3 (G_{s_{23}} \cos(\theta_2 - \theta_3) + B_{s_{23}} \operatorname{sen}(\theta_2 - \theta_3)) + V_2^2 G_{p_{23}}$
P_{23}		$-V_2^2 B_{s_{23}} - V_2 V_3 (G_{s_{23}} \operatorname{sen}(\theta_2 - \theta_3) - B_{s_{23}} \operatorname{sen}(\theta_2 - \theta_3)) - V_2^2 B_{p_{23}}$
Q_{23}		

- Al no existir solución exacta del sistema, para poder estimar una que se acerque en algún sentido a esa ideal inalcanzable, es necesario definir un criterio, métrica (o estimador) en $\mathbb{R}^n$ que evalúe la bondad de una pseudosolución de ella. Los más usados son:
  - El de mínimos cuadrados.
  - El de máxima verosimilitud.

- El estimador de mínimos cuadrados elige como criterio de aproximación de la solución

$$\Phi(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} (b_i - f_i(x_1, x_2, \ldots, x_n))^2$$

y como objetivo hacer mínima la función $\Phi(x_1, x_2, \ldots, x_n)$:

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimizar}} \ \Phi(\boldsymbol{x})$$

- El de máxima verosimilitud es idéntico al de mínimos cuadrados cuando los errores de las mediciones tienen una distribución de probabilidad $N(0,\sigma)$: Ambos convergen en probabilidad a $\boldsymbol{x}$, son asintóticamente normales y consistentes para $m \to \infty$.

- Si un determinado aparato suministra la medida b, siendo b^{real} la que debería dar si la precisión de la medición fuese total, se tendrá que

$$b = b^{real} + \eta,$$

donde η es el error aleatorio propio del aparato de medida.

- Si η no está sesgado, su función de densidad de probabilidad

$$FDP(\eta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\eta^2}{2\sigma^2}}$$

es la normal de media cero y desviación típica σ.

- Como la media de η se supone cero, la media de la muestra real de b es b^{real}. La función de densidad de probabilidad de b es

$$FDP(b) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(b-b^{real})^2}{2\sigma^2}}$$

- Si se tiene un vector de m medidas, $\boldsymbol{b}$, en el que cada uno de sus elementos o coeficiente tiene una función de densidad de probabilidad como la descrita, la función de densidad de probabilidad conjunta de la muestra $b_1, \ldots, b_m$, supuestas todas las medidas independientes unas de otras, es

$$FDP(b_1, \ldots, b_m) = FDP(b_1) \cdot FDP(b_2) \cdots FDP(b_m) = \prod_{i=1}^{m} FDP(b_i).$$

A esta función se la denomina verosimilitud de los parámetros (los b_i^{real}) y se designa por $L(b_1^{real}, \ldots, b_m^{real}) = L(\boldsymbol{b}^{real})$.

- Si se quiere hacer máxima la verosimilitud (probabilidad) de que se obtenga como medida real la de la muestra $\boldsymbol{b}$,

$$L(\boldsymbol{b}^{real}) = \prod_{i=1}^{m} \left(\frac{1}{\sigma_i \sqrt{2\pi}} \right) e^{-\sum_{i=1}^{m} \frac{\left(b_i - b_i^{real}\right)^2}{2\sigma_i^2}},$$

habrá que maximizar L o, lo que debe conseguir el mismo efecto, $\ln L(\boldsymbol{b}^{real})$.

- Ahora bien, maximizar la función $\ln L(\boldsymbol{b}^{real})$ es lo mismo que

$$\text{maximizar} \left[-\sum_{i=1}^{m} \ln(\sigma_i \sqrt{2\pi}) - \sum_{i=1}^{m} \frac{\left(b_i - b_i^{real}\right)^2}{2\sigma_i^2} \right].$$

- Como $-\sum_{i=1}^{m} \ln(\sigma_i \sqrt{2\pi})$ es constante, este problema equivale a

$$\text{minimizar} \left[\sum_{i=1}^{m} \frac{\left(b_i - b_i^{real}\right)^2}{2\sigma_i^2} \right].$$

- Los parámetros $\boldsymbol{b}^{real}$ se relacionan a través de las variables de estado por la función no lineal antes mencionada

$$\boldsymbol{b}^{real} = \boldsymbol{f}(\boldsymbol{x}),$$

donde $\boldsymbol{x}$ es el vector de variables de estado: las tensiones en los nudos de la red y las tomas de los transformadores con regulación.

- El problema expresado en forma matricial resulta

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimizar}}\,[\boldsymbol{b}-\boldsymbol{f}(\boldsymbol{x})]^T\,\boldsymbol{\Theta}^{-1}\,[\boldsymbol{b}-\boldsymbol{f}(\boldsymbol{x})]\,,$$

donde la matriz

$$\boldsymbol{\Theta}=\begin{bmatrix}\sigma_1^2 & & \\ & \ddots & \\ & & \sigma_m^2\end{bmatrix}$$

es la matriz de covarianzas de las mediciones.

- Como esta matriz es definida positiva, su inversa se puede expresar de la forma $\boldsymbol{\Theta}^{-1}=\boldsymbol{W}^T\boldsymbol{W}$, dando lugar a la formulación

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimizar}}\,\|\boldsymbol{W}(\boldsymbol{b}-\boldsymbol{f}(\boldsymbol{x}))\|_2^2$$

idéntica en estructura al que planteábamos con el estimador de mínimos cuadrados.

Resolución numérica del problema

- Hay que resolver $\boldsymbol{r}(\boldsymbol{x}) = \boldsymbol{0}$, un sistema de m ecuaciones no lineales con n incógnitas. En principio $m \gg n$.

- Como se hizo cuando $m = n$, lo natural es generar un proceso iterativo de aproximación punto a punto a la solución en el que $\boldsymbol{r}(\boldsymbol{x})$ se modelice en cada uno de ellos mediante desarrollo en serie de Taylor hasta primeras derivadas del tipo

$$M(\boldsymbol{p}) \equiv \boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{J}(\boldsymbol{x})\boldsymbol{p}$$

  y se resuelva este sistema lineal, que será en general incompatible.

- La solución por mínimos cuadrados de cada uno de estos sistemas, como en el caso de Newton-Raphson, será una nueva dirección de movimiento a un nuevo punto del proceso iterativo que nos aproxime a la solución.

Método de Gauss-Newton

I – Definir un $\boldsymbol{x}_0$; hacer $k = 1$ y $\boldsymbol{x}_k \leftarrow \boldsymbol{x}_0$

II – Determinar $\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2} \|\boldsymbol{r}(\boldsymbol{x}_k) + \boldsymbol{J}(\boldsymbol{x}_k)(\boldsymbol{x} - \boldsymbol{x}_k)\|_2^2$

III – Si $\boldsymbol{x} - \boldsymbol{x}_k < Tol$, parar: problema resuelto;
si no, hacer $k = k + 1$, $\boldsymbol{x}_k = \boldsymbol{x}$ e ir al paso II.

- El subproblema del punto II es un problema lineal

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\text{minimizar}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2,$$

por lo que se puede resolver con los métodos que conocemos para mínimos cuadrados lineales: ecuaciones normales, factorización QR, descomposición en valores singulares, etc.

- **Ejemplo** Mediante Gauss-Newton, resolviendo los subproblemas mediante el operador \ de **Matlab**, determinar x_1 y x_2 de la función $e^{x_1+tx_2}$ que mejor ajuste los pares de puntos

$$\{(t_i, y_i)\} = \{(-2, 1/2), (-1, 1), (0, 2), (1, 4)\}.$$

La función $\boldsymbol{r}(\boldsymbol{x})$ es $\mathbb{R}^2 \to \mathbb{R}^4$; su matriz Jacobiana es

$$\boldsymbol{J}(\boldsymbol{x}) = \begin{bmatrix} e^{x_1-2x_2} & -2e^{x_1-2x_2} \\ e^{x_1-x_2} & -e^{x_1-x_2} \\ e^{x_1} & 0 \\ e^{x_1+x_2} & e^{x_1+x_2} \end{bmatrix}.$$

- Desde $\boldsymbol{x}_0 = [1,\ 1]^T$, el código **Matlab** que lo resuelve es este.

```
function GN111(fx,x)          % Gauss-Newton
 tol=sqrt(eps); dnor=1.0;
   while dnor>tol
      [f J] = fx(x);
      p=J\f';
      x=x-p;
      dnor=norm(p,inf)/norm(x,inf);
      s=norm(f)^2;
      fprintf(' %15.10e %15.10e %15.10e %15.10e\n',x,s,dnor);
   end
end
```

```
function [f J]=GaNew(x)
 f(1)=exp(x(1)-2*x(2))-0.5;
 f(2)=exp(x(1)-1.0*x(2))-1.0;
 f(3)=exp(x(1))-2.0;
 f(4)=exp(x(1)+x(2))-4.0;
 if nargout<2, return, end
 J(1,1)=exp(x(1)-2.0*x(2)); J(1,2)=-2*exp(x(1)-2*x(2));
 J(2,1)=exp(x(1)-x(2));     J(2,2)=-exp(x(1)-x(2));
 J(3,1)=exp(x(1));          J(3,2)=0;
 J(4,1)=exp(x(1)+x(2));     J(4,2)=exp(x(1)+x(2));
end
```

- El proceso hasta la solución $\boldsymbol{x} = [\ln 2, \ln 2]^T$, con

```
>> GN111(@GaNew,[1;1])
```

es el de esta tabla

k	x_1	x_2	$\|\boldsymbol{r}\|_2^2$	$\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_\infty / \|\boldsymbol{x}_k\|_\infty$
1	7,5406407460540E-1	7,8581936682613E-1	4.6113768156e-01	3.1296749716e-1
2	6,9782818348320E-1	6,9931189327404E-1	2.0073845116e-03	1.2370370632e-1
3	6,9317290130691E-1	6,9317774998543E-1	5.3683607855e-08	8.8493084521e-3
4	6,9314718125839E-1	6,9314718138758E-1	3.9452570113e-17	4.4101156284e-5
5	6,9314718055994E-1	6,9314718055994E-1	8.0118685687e-31	1.1940287961e-9

La salida de **Matlab**

```
>> GN111(@GaNew,[1;1])
 7.5406407955e-01 7.8581936683e-01 1.2019085869e+01 3.1296749716e-01
 6.9782818219e-01 6.9931189370e-01 4.6113768156e-01 1.2370370632e-01
 6.9317290132e-01 6.9317774998e-01 2.0073845116e-03 8.8493084521e-03
 6.9314718126e-01 6.9314718139e-01 5.3683607855e-08 4.4101156284e-05
 6.9314718056e-01 6.9314718056e-01 3.9452547217e-17 1.1940285355e-09
```

Método de Levenberg-Marquardt

- Se formuló inicialmente en 1944 por Kenneth Levenberg, EE.UU., 1919-1973, y se desarrolló en 1963 por Donald W. Marquardt, EE.UU., 1929-1997.

- Para evitar las dificultades del de Gauss-Newton cuando a lo largo del proceso la matriz Jacobiana no tiene rango completo o está mal condicionada, se propuso que la dirección $\boldsymbol{p}_k = \boldsymbol{x} - \boldsymbol{x}_k$ saliese de la solución del subproblema

$$\underset{\|\boldsymbol{p}_k\| \leq \Delta_k}{\text{mín.}} \ \tfrac{1}{2}\|\boldsymbol{r}(\boldsymbol{x}_k) + \boldsymbol{J}(\boldsymbol{x}_k)\boldsymbol{p}_k\|_2^2,$$

  supuesto que se conoce un número Δ_k tal que el modelo lineal de $\boldsymbol{r}(\boldsymbol{x}_k)$ representa suficientemente bien la función dentro de la esfera de confianza de radio Δ_k.

- Si la función decrece en esta dirección, se hace $\boldsymbol{x} + \boldsymbol{p}$ el nuevo punto del proceso y se adapta Δ con algún criterio. Si no decrece, y $\boldsymbol{x} \neq \boldsymbol{x}^*$, se modifica la estrategia con Δ para que en la próxima iteración se mejore la función.

- Si la dirección del subproblema, sin tener en cuenta la restricción $\|\boldsymbol{p}_k\| \leq \Delta_k$, verifica ésta, el paso sería el mismo; si no, se puede comprobar que existirá un $\lambda > 0$ tal que

$$(\boldsymbol{J}^T\boldsymbol{J} + \lambda \boldsymbol{I})\boldsymbol{p}^{LM} = -\boldsymbol{J}^T\boldsymbol{r},$$

$$\lambda(\Delta - \|\boldsymbol{p}^{LM}\|) = 0,$$

  por lo que $\boldsymbol{p}_k = \boldsymbol{p}^{LM}$ sería una solución del subproblema que satisface $\|\boldsymbol{p}_k\| = \Delta_k$.

- Estas condiciones son una adaptación a este problema de otras generales para problemas de optimización con condiciones, como es el caso que impone el estar dentro de la región de confianza.

- El método, implícitamente, es una permanente elección entre la dirección de Gauss-Newton y la de máxima pendiente, o alguna entre ambas.

- En efecto:

  - Cuando $\lambda = 0$ se tiene la dirección de Gauss-Newton

$$\boldsymbol{p}_k^{LM} = -\left(\boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{J}(\boldsymbol{x}_k)\right)^{-1} \boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{r}(\boldsymbol{x}_k).$$

  - Cuando λ es muy grande

$$\boldsymbol{p}_k^{LM} \approx -\frac{1}{\lambda} \boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{r}(\boldsymbol{x}_k),$$

    por lo que la dirección es colineal con la de máxima pendiente.

- La que sigue es una versión algorítmica del método, con una adaptación del parámetro λ de iteración en iteración[2] muy simple.

I – Definir un $\boldsymbol{x}_0 \in \mathbb{R}^n$; hacer $\lambda = 0{,}01$, $k = 1$ y $\boldsymbol{x}_k \leftarrow \boldsymbol{x}_0$

II – Calcular $\boldsymbol{p}_k = -\left(\boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{J}(\boldsymbol{x}_k) + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{J}(\boldsymbol{x}_k)^T \boldsymbol{r}(\boldsymbol{x}_k)$

III – **if** $(\|\boldsymbol{r}(\boldsymbol{x}_k + \boldsymbol{p}_k)\|_2^2 < \|\boldsymbol{r}(\boldsymbol{x}_k)\|_2^2)$ **then**

si $\boldsymbol{p}_k \leq Tol$, parar: problema resuelto

si $\boldsymbol{p}_k > Tol$, hacer: $\lambda \leftarrow \lambda/10$, $k \leftarrow k+1$, $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \boldsymbol{p}$

y volver al paso II

else

$\lambda \leftarrow 10 \cdot \lambda$

Volver al paso II sin tener que calcular $\boldsymbol{J}(\boldsymbol{x}_k)$

end

[2] Otras formas de cambiar λ son las que se estudian al presentar los métodos de región de confianza para minimizar funciones sin condiciones.

- **Ejemplo** Mediante Levenberg-Marquardt, ajustar a la función

$$f(x) = \frac{b_1}{1 + b_2 e^{t b_3}}$$

el conjunto de puntos de la tabla.

t_i	y_i	t_i	y_i
1	5,308	7	31,443
2	7,24	8	38,558
3	9,638	9	50,156
4	12,866	10	62,948
5	17,069	11	75,995
6	23,192	12	91,972

- Partiendo de $\boldsymbol{x}_0 = [200, 30, -0{,}4]^T$, el código **Matlab** que lo resuelve es este:

```
function Levmar_99
% Levenberg-Marquardt
   m=12; n=3; x=[200;30;-0.4]; mu=0.01; J=zeros(m,n);
   jtj=zeros(n,n); dnor=1;  f=fx(x);
   tol=sqrt(eps)*norm(x,inf); update=1;

   while dnor>tol
      if update==1
         f=fx(x); J=derf(x);
         jtj=J'*J;
         res=norm(f)^2;
      end
      a=jtj+mu*eye(n);
      s=a\(J'*f);
      b=x-s;
      f1=fx(b);
      res1=norm(f1)^2;
      if res1<res
         x=b; f=f1;
         dnor=norm(s,inf)/norm(x,inf);
         fprintf('%15.10e %15.10e %15.10e %15.10e %15.10e %15.10e\n',...
                 x,res1,mu,dnor);
         mu=mu/10; update=1;
      else
         mu=mu*10; update=0;
      end
   end
end

function f=fx(x)
   f=zeros(12,1);
   f(1)  = x(1)/(1+x(2)*exp(x(3)))-5.308;
   f(2)  = x(1)/(1+x(2)*exp(2*x(3)))-7.24;
   f(3)  = x(1)/(1+x(2)*exp(3*x(3)))-9.638;
   f(4)  = x(1)/(1+x(2)*exp(4*x(3)))-12.866;
   f(5)  = x(1)/(1+x(2)*exp(5*x(3)))-17.069;
   f(6)  = x(1)/(1+x(2)*exp(6*x(3)))-23.192;
   f(7)  = x(1)/(1+x(2)*exp(7*x(3)))-31.443;
   f(8)  = x(1)/(1+x(2)*exp(8*x(3)))-38.558;
   f(9)  = x(1)/(1+x(2)*exp(9*x(3)))-50.156;
   f(10) = x(1)/(1+x(2)*exp(10*x(3)))-62.948;
   f(11) = x(1)/(1+x(2)*exp(11*x(3)))-75.995;
   f(12) = x(1)/(1+x(2)*exp(12*x(3)))-91.972;
end
```

```
function J=derf(x)
   J(1,1) =1/(1+x(2)*exp(x(3)));
   J(1,2) =-x(1)*exp(x(3))/(1+x(2)*exp(x(3)))^2;
   J(1,3) =-x(1)*x(2)*exp(x(3))/(1+x(2)*exp(x(3)))^2;
   J(2,1) =1/(1+x(2)*exp(2*x(3)));
   J(2,2) =-x(1)*exp(2*x(3))/(1+x(2)*exp(2*x(3)))^2;
   J(2,3) =-x(1)*x(2)*exp(2*x(3))*2/(1+x(2)*exp(2*x(3)))^2;
   J(3,1) =1/(1+x(2)*exp(3*x(3)));
   J(3,2) =-x(1)*exp(3*x(3))/(1+x(2)*exp(3*x(3)))^2;
   J(3,3) =-x(1)*x(2)*exp(3*x(3))*3/(1+x(2)*exp(3*x(3)))^2;
   J(4,1) =1/(1+x(2)*exp(4*x(3)));
   J(4,2) =-x(1)*exp(4*x(3))/(1+x(2)*exp(4*x(3)))^2;
   J(4,3) =-x(1)*x(2)*exp(4*x(3))*4/(1+x(2)*exp(4*x(3)))^2;
   J(5,1) =1/(1+x(2)*exp(5*x(3)));
   J(5,2) =-x(1)*exp(5*x(3))/(1+x(2)*exp(5*x(3)))^2;
   J(5,3) =-x(1)*x(2)*exp(5*x(3))*5/(1+x(2)*exp(5*x(3)))^2;
   J(6,1) =1/(1+x(2)*exp(6*x(3)));
   J(6,2) =-x(1)*exp(6*x(3))/(1+x(2)*exp(6*x(3)))^2;
   J(6,3) =-x(1)*x(2)*exp(6*x(3))*6/(1+x(2)*exp(6*x(3)))^2;
   J(7,1) =1/(1+x(2)*exp(7*x(3)));
   J(7,2) =-x(1)*exp(7*x(3))/(1+x(2)*exp(7*x(3)))^2;
   J(7,3) =-x(1)*x(2)*exp(7*x(3))*7/(1+x(2)*exp(7*x(3)))^2;
   J(8,1) =1/(1+x(2)*exp(8*x(3)));
   J(8,2) =-x(1)*exp(8*x(3))/(1+x(2)*exp(8*x(3)))^2;
   J(8,3) =-x(1)*x(2)*exp(8*x(3))*8/(1+x(2)*exp(8*x(3)))^2;
   J(9,1) =1/(1+x(2)*exp(9*x(3)));
   J(9,2) =-x(1)*exp(9*x(3))/(1+x(2)*exp(9*x(3)))^2;
   J(9,3) =-x(1)*x(2)*exp(9*x(3))*9/(1+x(2)*exp(9*x(3)))^2;
   J(10,1)=1/(1+x(2)*exp(10*x(3)));
   J(10,2)=-x(1)*exp(10*x(3))/(1+x(2)*exp(10*x(3)))^2;
   J(10,3)=-x(1)*x(2)*exp(10*x(3))*10/(1+x(2)*exp(10*x(3)))^2;
   J(11,1)=1/(1+x(2)*exp(11*x(3)));
   J(11,2)=-x(1)*exp(11*x(3))/(1+x(2)*exp(11*x(3)))^2;
   J(11,3)=-x(1)*x(2)*exp(11*x(3))*11/(1+x(2)*exp(11*x(3)))^2;
   J(12,1)=1/(1+x(2)*exp(12*x(3)));
   J(12,2)=-x(1)*exp(12*x(3))/(1+x(2)*exp(12*x(3)))^2;
   J(12,3)=-x(1)*x(2)*exp(12*x(3))*12/(1+x(2)*exp(12*x(3)))^2;

end
```

- La matriz jacobiana se ha calculado analíticamente. Cuando la complejidad de su cálculo es mayor, se puede aproximar por diferencias finitas.

- Los puntos del proceso iterativo que se obtienen son los de la tabla.

```
>> Levmar_99
  1 1.4756872319e+02 3.0915753991e+01 -3.3304661156e-01 3.2095285441e+02 1.0000000000e-02 3.5530074172e-01
  2 1.7485492970e+02 4.1322472852e+01 -3.1183181926e-01 1.9720323863e+01 1.0000000000e-03 1.5605054173e-01
  3 1.9441635874e+02 4.8327112423e+01 -3.1344957806e-01 2.6657614929e+00 1.0000000000e-04 1.0061616816e-01
  4 1.9613176244e+02 4.9079231125e+01 -3.1359042350e-01 2.5873011226e+00 1.0000000000e-05 8.7461799758e-03
  5 1.9618553722e+02 4.9091542376e+01 -3.1357009855e-01 2.5872773966e+00 1.0000000000e-06 2.7410166257e-04
  6 1.9618625588e+02 4.9091638512e+01 -3.1356973267e-01 2.5872773953e+00 1.0000000000e-07 3.6631679415e-06
  7 1.9618626172e+02 4.9091639449e+01 -3.1356972996e-01 2.5872773953e+00 1.0000000000e-08 2.9764224702e-08
```

- Con la Jacobiana por diferencias finitas partiendo de $\boldsymbol{x} = [10, 1, 1]^T$:

```
function Levmar_99_1
% Levenberg-Marquardt
 global h
 n=3; x=[10;1;1]; mu=0.01; h=sqrt(eps); dnor=1;  k=0;
 tol=h*norm(x,inf); update=1;
   while dnor>tol
      if update==1
         f=fx(x);
         J=derf(x);
         jtj=J'*J;
         res=norm(f)^2;
      end
      a=jtj+mu*eye(n);
      s=a\(J'*f);
      b=x-s;
      f1=fx(b); res1=norm(f1)^2;
      if res1<res
         x=b;
         f=f1;
         dnor=norm(s,inf)/norm(x,inf); k=k+1;
         fprintf('%3.0f %15.10e %15.10e %15.10e %15.10e %15.10e %15.10e\n',...
            k,x,res1,mu,dnor);
         mu=mu/10;
         update=1;
      else
         mu=mu*10;
         update=0;
      end
   end
end
```

```
function f=fx(x)
 f     = zeros(12,1);
 f(1)  = x(1)/(1+x(2)*exp(x(3)))-5.308;
 f(2)  = x(1)/(1+x(2)*exp(2*x(3)))-7.24;
 f(3)  = x(1)/(1+x(2)*exp(3*x(3)))-9.638;
 f(4)  = x(1)/(1+x(2)*exp(4*x(3)))-12.866;
 f(5)  = x(1)/(1+x(2)*exp(5*x(3)))-17.069;
 f(6)  = x(1)/(1+x(2)*exp(6*x(3)))-23.192;
 f(7)  = x(1)/(1+x(2)*exp(7*x(3)))-31.443;
 f(8)  = x(1)/(1+x(2)*exp(8*x(3)))-38.558;
 f(9)  = x(1)/(1+x(2)*exp(9*x(3)))-50.156;
 f(10) = x(1)/(1+x(2)*exp(10*x(3)))-62.948;
 f(11) = x(1)/(1+x(2)*exp(11*x(3)))-75.995;
 f(12) = x(1)/(1+x(2)*exp(12*x(3)))-91.972;
end

function J=derf(x)
 global h
 J        =zeros(12,3);
 J(1:12,1)=(fx([x(1)+h;x(2);x(3)])-fx([x(1)-h;x(2);x(3)]))/2/h;
 J(1:12,2)=(fx([x(1);x(2)+h;x(3)])-fx([x(1);x(2)-h;x(3)]))/2/h;
 J(1:12,3)=(fx([x(1);x(2);x(3)+h])-fx([x(1);x(2);x(3)-h]))/2/h;
end
```

- Se obtiene lo que sigue.

```
>> Levmar_99_1
  1 2.3152737458e+001 1.8021498292e+001 -1.3632831505e+001 1.1044395775e+004 1.0000000000e-002 7.3518297019e-001
  2 3.5521957325e+001 1.8106135944e+001 -1.2119438256e+001 9.2052245497e+003 1.0000000000e-002 3.4821335305e-001
  3 3.5535714828e+001 1.8690543439e+001 -1.5346094174e+000 7.3006550602e+003 1.0000000000e-002 2.9786452559e-001
  4 3.6382251480e+001 1.8685898203e+001 -5.3375592376e-001 6.3081204894e+003 1.0000000000e+002 2.7509388586e-002
  5 4.3864894722e+001 1.9510291916e+001 -7.1102849011e-001 4.2163999640e+003 1.0000000000e+001 1.7058386414e-001
  6 5.1372002147e+001 1.9466354784e+001 -4.2121047306e-001 4.0802626251e+003 1.0000000000e+001 1.4613227267e-001
  7 7.0428818614e+001 2.7344396915e+001 -5.2068992834e-001 1.1367192170e+003 1.0000000000e+000 2.7058265128e-001
  8 8.5236277490e+001 2.8137323946e+001 -3.8266075582e-001 1.0542743764e+003 1.0000000000e+000 1.7372249600e-001
  9 1.1239893980e+002 4.0287929157e+001 -3.9946571714e-001 9.4953809441e+001 1.0000000000e-001 2.4166297615e-001
 10 1.2617277706e+002 4.3008165823e+001 -3.7604998322e-001 5.1169151088e+001 1.0000000000e-001 1.0916647459e-001
 11 1.3399326931e+002 4.4781231173e+001 -3.7097156146e-001 2.8353413348e+001 1.0000000000e-001 5.8364814061e-002
 12 1.3977022988e+002 4.5331502547e+001 -3.6390738007e-001 2.1054836175e+001 1.0000000000e-001 4.1331838508e-002
 13 1.6069048866e+002 4.4865704491e+001 -3.3356360353e-001 1.8934102960e+001 1.0000000000e-002 1.3018977636e-001
 14 1.8513588826e+002 4.7484943675e+001 -3.1687615138e-001 8.7955954889e+000 1.0000000000e-003 1.3204030744e-001
 15 1.9533332296e+002 4.8976332187e+001 -3.1377968469e-001 2.6426289142e+000 1.0000000000e-004 5.2205299870e-002
 16 1.9617433148e+002 4.9090128611e+001 -3.1357474036e-001 2.5872794922e+000 1.0000000000e-005 4.2870467197e-003
 17 1.9618630640e+002 4.9091641090e+001 -3.1356969799e-001 2.5872773953e+000 1.0000000000e-006 6.1038510228e-005
 18 1.9618621815e+002 4.9091635363e+001 -3.1356975593e-001 2.5872773953e+000 1.0000000000e-007 4.4983495179e-007
 19 1.9618623975e+002 4.9091640103e+001 -3.1356974850e-001 2.5872773953e+000 1.0000000000e-002 1.1005699249e-007
>>
```

- Algo más sofisticado para variar λ y su resultado es lo que sigue.

```
function  [x k] = LevenbergMarquardt_2
global h
eta1=sqrt(eps); eta2=eta1; x=[10;1;1]; n = length(x); k=1; h=eta1;
f = fx(x); J = derf(x);
A = J'*J;
g = J'*f; ng = norm(g,inf);
F = (f'*f)/2;
mu = eta1 * max(diag(A));
nu = 2; stop = 0;

while ~stop
   if  ng <= eta2, stop = 1;
   else
      p = (A + mu*eye(n))\-g; np = norm(p,inf);
      nx = eta2 + norm(x,inf);
      if np <= eta2*nx, stop = 2; end
   end
   if  ~stop
      xnew = x + p;
      fn = fx(xnew); Jn = derf(xnew);
      Fn = (fn'*fn)/2;
      dL = (p'*(mu*p - g))/2; dF = F - Fn;
      if dL > 0 && dF > 0                          % Se adapta x y m
         x = xnew; F = Fn; J = Jn; f = fn;
         A = J'*J;
         g = J'*f; ng = norm(g,inf);
         mu = mu * max(1/3, 1-(2*dF/dL-1)^3);     % Fórmula adapt. mu en referencias
         nu = 2;
      else
         mu = mu*nu; nu = 2*nu;
      end
      k = k + 1;
      dnor=norm(p,inf)/norm(x,inf);
      fprintf('%3.0f %12.5e %12.5e %12.5e %12.5e %12.5e %12.5e\n',k,x,Fn,mu,dnor);
   end
end
end
```

```
function f=fx(x)
f     = zeros(12,1);
f(1)  = x(1)/(1+x(2)*exp(x(3)))-5.308;
f(2)  = x(1)/(1+x(2)*exp(2*x(3)))-7.24;
f(3)  = x(1)/(1+x(2)*exp(3*x(3)))-9.638;
f(4)  = x(1)/(1+x(2)*exp(4*x(3)))-12.866;
f(5)  = x(1)/(1+x(2)*exp(5*x(3)))-17.069;
f(6)  = x(1)/(1+x(2)*exp(6*x(3)))-23.192;
f(7)  = x(1)/(1+x(2)*exp(7*x(3)))-31.443;
f(8)  = x(1)/(1+x(2)*exp(8*x(3)))-38.558;
f(9)  = x(1)/(1+x(2)*exp(9*x(3)))-50.156;
f(10) = x(1)/(1+x(2)*exp(10*x(3)))-62.948;
f(11) = x(1)/(1+x(2)*exp(11*x(3)))-75.995;
f(12) = x(1)/(1+x(2)*exp(12*x(3)))-91.972;
end

function J=derf(x)
global h
J        =zeros(12,3);
J(1:12,1)=(fx([x(1)+h;x(2);x(3)])-fx([x(1)-h;x(2);x(3)]))/2/h;
J(1:12,2)=(fx([x(1);x(2)+h;x(3)])-fx([x(1);x(2)-h;x(3)]))/2/h;
J(1:12,3)=(fx([x(1);x(2);x(3)+h])-fx([x(1);x(2);x(3)-h]))/2/h;
end
```

```
>> [X]=LevenbergMarquardt_2
  2 1.00000e+001 1.00000e+000 1.00000e+000 6.09210e+007 3.20327e-007 3.21187e+002
  3 1.00000e+001 1.00000e+000 1.00000e+000 6.05210e+007 1.28131e-006 3.20139e+002
  4 1.00000e+001 1.00000e+000 1.00000e+000 5.82011e+007 1.02505e-005 3.13992e+002
  5 1.00000e+001 1.00000e+000 1.00000e+000 4.17315e+007 1.64008e-004 2.66267e+002
  6 1.00000e+001 1.00000e+000 1.00000e+000 3.04228e+006 5.24824e-003 7.37065e+001
  7 3.71830e+001 2.03134e+001 -1.39754e+001 4.61905e+003 1.74941e-003 7.31060e-001
  8 3.55330e+001 2.08570e+001 -2.85638e+000 4.13884e+003 5.83138e-004 3.12920e-001
  9 3.55330e+001 2.08570e+001 -2.85638e+000 1.21779e+004 1.16628e-003 7.60834e+000
 10 3.55330e+001 2.08570e+001 -2.85638e+000 1.21779e+004 4.66510e-003 7.13223e+000
 11 3.55330e+001 2.08570e+001 -2.85638e+000 1.21779e+004 3.73208e-002 5.18497e+000
 12 3.55330e+001 2.08570e+001 -2.85638e+000 1.21789e+004 5.97133e-001 1.45888e+000
 13 3.55330e+001 2.08570e+001 -2.85638e+000 1.08549e+004 1.91083e+001 1.25892e-001
 14 3.70743e+001 2.08320e+001 -8.45833e-001 2.86866e+003 6.36942e+000 5.42303e-002
 15 3.70743e+001 2.08320e+001 -8.45833e-001 4.57304e+003 1.27388e+001 3.13411e-001
 16 3.70743e+001 2.08320e+001 -8.45833e-001 2.96395e+003 5.09554e+001 2.00676e-001
 17 3.94274e+001 2.08200e+001 -5.56041e-001 2.67565e+003 5.09596e+001 5.96813e-002
 18 4.10697e+001 2.09711e+001 -7.41820e-001 2.36432e+003 4.98849e+001 3.99886e-002
 19 4.31862e+001 2.09854e+001 -5.65439e-001 2.17933e+003 4.91881e+001 4.90094e-002
 20 4.48055e+001 2.11303e+001 -6.70109e-001 1.96543e+003 4.12763e+001 3.61395e-002
 21 4.69944e+001 2.11853e+001 -5.62055e-001 1.78755e+003 3.59446e+001 4.65787e-002
 22 4.89704e+001 2.13752e+001 -6.12569e-001 1.59878e+003 1.50058e+001 4.03496e-002
 23 5.35482e+001 2.15958e+001 -5.19378e-001 1.31483e+003 1.14216e+001 8.54895e-002
 24 5.77611e+001 2.22507e+001 -5.42187e-001 1.03419e+003 3.80721e+000 7.29365e-002
 25 6.75450e+001 2.32354e+001 -4.43173e-001 7.28564e+002 3.60218e+000 1.44851e-001
 26 7.30187e+001 2.54797e+001 -4.76258e-001 4.74737e+002 1.20073e+000 7.49634e-002
 27 8.56068e+001 2.82368e+001 -4.08341e-001 3.05397e+002 1.17121e+000 1.47046e-001
 28 9.17724e+001 3.22479e+001 -4.29597e-001 1.66694e+002 3.90403e-001 6.71834e-002
 29 1.04566e+002 3.65437e+001 -3.98435e-001 8.96279e+001 3.14695e-001 1.22346e-001
 30 1.12665e+002 4.06796e+001 -3.99237e-001 4.63231e+001 1.04898e-001 7.18929e-002
 31 1.25845e+002 4.32662e+001 -3.77662e-001 2.49768e+001 8.23411e-002 1.04731e-001
 32 1.35084e+002 4.48533e+001 -3.69135e-001 1.35599e+001 2.74470e-002 6.83906e-002
 33 1.49370e+002 4.49653e+001 -3.48691e-001 8.04215e+000 2.53594e-002 9.56446e-002
 34 1.58295e+002 4.57702e+001 -3.41526e-001 4.27044e+000 8.45312e-003 5.63779e-002
 35 1.71229e+002 4.64779e+001 -3.29024e-001 2.70729e+000 7.50803e-003 7.55398e-002
 36 1.79072e+002 4.73221e+001 -3.24136e-001 1.73929e+000 2.50268e-003 4.37960e-002
 37 1.87986e+002 4.81609e+001 -3.18035e-001 1.42192e+000 1.63585e-003 4.74182e-002
 38 1.92841e+002 4.87157e+001 -3.15399e-001 1.30936e+000 5.45282e-004 2.51780e-002
 39 1.95496e+002 4.90113e+001 -3.13931e-001 1.29440e+000 1.81761e-004 1.35793e-002
 40 1.96128e+002 4.90848e+001 -3.13601e-001 1.29364e+000 6.05869e-005 3.22121e-003
 41 1.96184e+002 4.90914e+001 -3.13571e-001 1.29364e+000 2.01956e-005 2.88957e-004
 42 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 6.73187e-006 9.74625e-006
 43 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 7.08937e-006 2.54040e-007
 44 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 1.41787e-005 1.27817e-006
 45 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 5.67150e-005 1.27412e-006
 46 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 4.53720e-004 1.25034e-006
 47 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 7.25951e-003 1.06488e-006
 48 1.96186e+002 4.90916e+001 -3.13570e-001 1.29364e+000 2.32304e-001 3.00599e-007
X =
  1.0e+002 *
   1.961862713737317
   0.490916428999001
  -0.003135697293093
```

Método de Newton

- Está basado en el método de Newton para resolver problemas de optimización sin condiciones, como lo es

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimizar}}\, f(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^{m} r_i^2(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|_2^2.$$

- Si la función es continua y tiene derivadas parciales hasta segundo orden continuas, el método se basa en el modelo de $f(\boldsymbol{x})$ del desarrollo de Taylor hasta segundo orden de derivadas:

$$f(\boldsymbol{x}+\boldsymbol{p}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T\boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^T\nabla^2 f(\boldsymbol{x})\,\boldsymbol{p} + \mathcal{O}\left(\|\boldsymbol{p}\|^3\right).$$

- En esta última expresión,

$$\nabla f(\boldsymbol{x}) = \left[\frac{\partial f(\boldsymbol{x})}{\partial x_1} \; \frac{\partial f(\boldsymbol{x})}{\partial x_2} \; \cdots \; \frac{\partial f(\boldsymbol{x})}{\partial x_n} \right]^T,$$

es el vector gradiente, y

$$\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f(\boldsymbol{x})}{\partial^2 x_1} & \frac{\partial^2 f(\boldsymbol{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\boldsymbol{x})}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(\boldsymbol{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\boldsymbol{x})}{\partial^2 x_2} & \cdots & \frac{\partial^2 f(\boldsymbol{x})}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\boldsymbol{x})}{\partial x_n \partial x_1} & \frac{\partial^2 f(\boldsymbol{x})}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(\boldsymbol{x})}{\partial^2 x_n} \end{bmatrix}$$

la matriz Hessiana: La matriz Jacobiana del vector gradiente.

- En el caso de la función $f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{r}(\boldsymbol{x})\|_2^2$,

$$\begin{aligned}
\nabla f(\boldsymbol{x}) &= \sum_{j=1}^{m} r_j(\boldsymbol{x}) \nabla r_j(\boldsymbol{x}) = \boldsymbol{J}(\boldsymbol{x})^T \boldsymbol{r}(\boldsymbol{x}), \\
\nabla^2 f(\boldsymbol{x}) &= \sum_{j=1}^{m} \nabla r_j(\boldsymbol{x}) \nabla r_j(\boldsymbol{x})^T + \sum_{j=1}^{m} r_j(\boldsymbol{x}) \nabla^2 r_j(\boldsymbol{x}) \\
&= \boldsymbol{J}(\boldsymbol{x})^T \boldsymbol{J}(\boldsymbol{x}) + \sum_{j=1}^{m} r_j(\boldsymbol{x}) \nabla^2 r_j(\boldsymbol{x}).
\end{aligned}$$

- La condición necesaria de óptimo, $\nabla f(\boldsymbol{x}^*) = 0$, despreciando el término $\mathcal{O}\left(\|\boldsymbol{p}\|^3\right)$, conduce a un sistema lineal, $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, en este caso

$$\boxed{\nabla f(\boldsymbol{x}) + \nabla^2 f(\boldsymbol{x}) \boldsymbol{p} = \boldsymbol{0}}$$

cuya solución es la dirección de movimiento hacia el óptimo.

- El algoritmo es este:

I – Definir un $\boldsymbol{x}_0$ y condiciones de partida

II – Resolver el sistema lineal $\nabla f(\boldsymbol{x}) + \nabla^2 f(\boldsymbol{x})\,(\boldsymbol{x} - \boldsymbol{x}_k) = \boldsymbol{0}$

III – Si $\boldsymbol{x} - \boldsymbol{x}_k < Tol$, parar: el problema está resuelto;
si no, hacer $k = k + 1$, $\boldsymbol{x}_k = \boldsymbol{x}$ e ir al paso II

Lo costoso de este procedimiento es calcular la matriz Hessiana y resolver el sistema del Paso II.

- El método de Gauss-Newton se puede ver como una modificación del de Newton con la simplificación $\nabla^2 f(\boldsymbol{x}) \approx \boldsymbol{J}^T \boldsymbol{J}$ y $\nabla f(\boldsymbol{x}) = \boldsymbol{J}^T \boldsymbol{r}(\boldsymbol{x})$.

- Resolver por Newton el ajuste no lineal de $e^{x_1+tx_2}$ sería así. La complicación de obtener la matriz Hessiana en considerable.

```
function LSQ_New_1(fx,x)
% Mínimos cuadrados Newton
 tol=sqrt(eps); p=ones(2,1); [f grad Hess] = fx(x);
   while abs(grad'*p)>tol
      p=-Hess\grad;
      x=x+p;
      [f grad Hess] = fx(x);
      fprintf(' %15.10e %15.10e %15.10e\n',x,f);
   end
end
```

```
function [f g H] = Newt_Ls(x)
 f=(exp(x(1)-2*x(2))-0.5)^2+(exp(x(1)-x(2))-1.0)^2+(exp(x(1))-2.0)^2+(exp(x(1)+x(2))-4.0)^2;

 g = [2*(exp(x(1)-2*x(2))-0.5)*exp(x(1)-2*x(2))+2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+...
 2*(exp(x(1))-2)*exp(x(1))+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2));...
 -4*(exp(x(1)-2*x(2))-0.5)*exp(x(1)-2*x(2))-2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+...
 2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2))];

 H=[2*exp(x(1)-2*x(2))^2+2*(exp(x(1)-2*x(2))-1/2)*exp(x(1)-2*x(2))+2*exp(x(1)-x(2))^2+...
 2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+2*exp(x(1))^2+2*(exp(x(1))-2)*exp(x(1))+...
 2*exp(x(1)+x(2))^2+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2)), ...  % Coeficiente (1,1)
 -4*exp(x(1)-2*x(2))^2-4*(exp(x(1)-2*x(2))-1/2)*exp(x(1)-2*x(2))-2*exp(x(1)-x(2))^2-...
 2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+2*exp(x(1)+x(2))^2+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2));  %Coef (1,2)
 -4*exp(x(1)-2*x(2))^2-4*(exp(x(1)-2*x(2))-1/2)*exp(x(1)-2*x(2))-2*exp(x(1)-x(2))^2-...
 2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+2*exp(x(1)+x(2))^2+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2)), ... %Coef (2,1)
 8*exp(x(1)-2*x(2))^2+8*(exp(x(1)-2*x(2))-1/2)*exp(x(1)-2*x(2))+2*exp(x(1)-x(2))^2+ ...
 2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+2*exp(x(1)+x(2))^2+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2))]; %Coef (2,1)
end
```

- Se puede usar el cálculo de la Hessiana por diferencias finitas...

```
function LSQ_New_diff(fx,x)         % Mínimos cuadrados Newton
 global h
 tol=sqrt(eps); p=ones(2,1); h=sqrt(eps); [f grad Hess]=fx(x);
 while abs(grad'*p)>tol
    p=-Hess\grad;
    x=x+p;
    [f grad Hess] = fx(x);
    fprintf(' %15.10e %15.10e %15.10e\n',x,f);
 end
end
```

```
function [f g H] = Newt_Ls_1(x)
 global h
 f=(exp(x(1)-2*x(2))-0.5)^2+(exp(x(1)-x(2))-1.0)^2+(exp(x(1))-2.0)^2+(exp(x(1)+x(2))-4.0)^2;
 if nargout<2, return, end
 g = [2*(exp(x(1)-2*x(2))-0.5)*exp(x(1)-2*x(2))+2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+...
 2*(exp(x(1))-2)*exp(x(1))+2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2));...
 -4*(exp(x(1)-2*x(2))-0.5)*exp(x(1)-2*x(2))-2*(exp(x(1)-x(2))-1)*exp(x(1)-x(2))+...
 2*(exp(x(1)+x(2))-4)*exp(x(1)+x(2))];
 if nargout<3, return, end
 x1=[x(1)+h;x(2)]; [f1 g1]=Newt_Ls_1(x1);
 H(1:2,1)=(g1-g)/h;
 x1=[x(1);x(2)+h]; [f1 g1]=Newt_Ls_1(x1);
 H(1:2,2)=(g1-g)/h;
end
```

- Y más:

```
function LSQ_New_dif_1(x)          % Mínimos cuadrados Newton
 global h
   tol=sqrt(eps); h=sqrt(eps)/norm(x); p=ones(2,1);
   while abs(grad(x)'*p)>tol
      p=-Hess(x)\grad(x);
      x=x+p;
      fprintf(' %15.10e %15.10e %15.10e\n',x,fx(x));
   end
end

function f=fx(x)
 f=(exp(x(1)-2*x(2))-0.5)^2+(exp(x(1)-x(2))-1.0)^2+...
   (exp(x(1))-2.0)^2+(exp(x(1)+x(2))-4.0)^2;
end

function g=grad(x)
 global h
 g=[(fx([x(1)+h,x(2)])-fx([x(1)-h,x(2)]))/(2*h);
    (fx([x(1),x(2)+h])-fx([x(1),x(2)-h]))/(2*h)];
end

function H=Hess(x)
 global h
 H(1:2,1)=(grad([x(1)+h;x(2)])-grad(x))/h;
 H(1:2,2)=(grad([x(1);x(2)+h])-grad(x))/h;
end
```

```
>> LSQ_New_1(@Newt_Ls,[1;1])
 8.1564455682e-001 8.6820863544e-001 2.0026578055e+000
 7.3026817718e-001 7.4718705926e-001 1.5293213074e-001
 6.9788477856e-001 6.9905288119e-001 1.9358782157e-003
 6.9322068406e-001 6.9323478800e-001 4.3977229201e-007
 6.9314719772e-001 6.9314720089e-001 2.3813689172e-014
>> LSQ_New_diff(@Newt_Ls_1,[1;1])
 8.1564455807e-001 8.6820863891e-001 2.0026578782e+000
 7.3026818132e-001 7.4718706243e-001 1.5293215651e-001
 6.9788478088e-001 6.9905288265e-001 1.9358796093e-003
 6.9322068433e-001 6.9323478806e-001 4.3977415379e-007
 6.9314719772e-001 6.9314720089e-001 2.3817330783e-014
>> LSQ_New_dif_1([1;1])
 8.4944340872e-01 8.7730513107e-01 2.7563553502e+00
 7.5875713784e-01 6.8293863250e-01 7.8534508448e-02
 6.9908967430e-01 6.9134017676e-01 5.0043933563e-04
 6.9320781375e-01 6.9312744209e-01 5.0431466984e-08
 6.9314718697e-01 6.9314717816e-01 5.3057457678e-16
```

- Por último, usando la potencia del cálculo simbólico de **Matlab** se puede conseguir algo como esto:

```
function LSQ_New_sym_2(x1,y1)          % Mínimos cuadrados Newton
   syms x y
   tol=sqrt(eps);
   f(x,y)=(exp(x-2*y)-0.5)^2+(exp(x-y)-1.0)^2+(exp(x)-2.0)^2+(exp(x+y)-4.0)^2;
   gra=gradient(f,[x y]); hes=hessian(f,[x y]);
   vec=-hes\gra;
   while abs(gra(x1,y1)'*[x1;y1])>tol
      xx=double(vec(x1,y1));
      x1=x1+xx(1); y1=y1+xx(2);
      fprintf(' %15.10e %15.10e %15.10e\n',x1,y1,double(f(x1,y1)));
   end
end
```

```
>> LSQ_New_sym_2(1,1)
 8.1564455682e-01 8.6820863544e-01 2.0026578055e+00
 7.3026817718e-01 7.4718705926e-01 1.5293213074e-01
 6.9788477856e-01 6.9905288119e-01 1.9358782157e-03
 6.9322068406e-01 6.9323478800e-01 4.3977229201e-07
 6.9314719772e-01 6.9314720089e-01 2.3813689162e-14
 6.9314718056e-01 6.9314718056e-01 7.2279874085e-29
```

Seguro que se puede estructurar un programa bastante mejor.