Resolución numérica del problema no lineal de mínimos cuadrados. Aplicaciones a las estimación de parámetros de modelos matemáticos.

Dídac Blanco Morros

14 de febrero de 2023

USC
UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
FACULTADE DE MATEMÁTICAS

Índice.

Índice.

Conceptos previos

Un problema de optimización sin restricciones tiene la forma

$$\min_{x} f(x)$$

- $x \in \mathbb{R}^n$ y $f : \mathbb{R}^n \to \mathbb{R}$ es continuamente diferenciable.
- A f se le llama función objetivo.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.
- Mínimo x^*: existe $\delta > 0$ tal que $f(x^*) \leq f(x)$.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.
- Mínimo x^*: existe $\delta > 0$ tal que $f(x^*) \leq f(x)$.
- Mínimo estricto x^*: existe $\delta > 0$ tal que $f(x^*) < f(x)$ con $x \neq x^*$.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.
- Mínimo x^*: existe $\delta > 0$ tal que $f(x^*) \leq f(x)$.
- Mínimo estricto x^*: existe $\delta > 0$ tal que $f(x^*) < f(x)$ con $x \neq x^*$.
  - Local: para todo $x \in \mathbb{R}^n$ que satisface $\|x - x^*\| < \delta$.
  - Global: para todo $x \in \mathbb{R}^n$

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.
- Mínimo x^*: existe $\delta > 0$ tal que $f(x^*) \leq f(x)$.
- Mínimo estricto x^*: existe $\delta > 0$ tal que $f(x^*) < f(x)$ con $x \neq x^*$.
  - Local: para todo $x \in \mathbb{R}^n$ que satisface $\|x - x^*\| < \delta$.
  - Global: para todo $x \in \mathbb{R}^n$
- Producto escalar de x e y en $\mathbb{R}^n$: $\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$.

Conceptos previos

- Norma euclídea: $\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2}$.
- Punto estacionario de f: $\nabla f(x) = 0$.
- Mínimo x^*: existe $\delta > 0$ tal que $f(x^*) \leq f(x)$.
- Mínimo estricto x^*: existe $\delta > 0$ tal que $f(x^*) < f(x)$ con $x \neq x^*$.
  - Local: para todo $x \in \mathbb{R}^n$ que satisface $\|x - x^*\| < \delta$.
  - Global: para todo $x \in \mathbb{R}^n$
- Producto escalar de x e y en $\mathbb{R}^n$: $\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$.
- Dirección descendente de f en x: $d \in \mathbb{R}^n$ tal que $\langle \nabla f(x), d \rangle < 0$.

Condiciones de optimalidad

Consideremos el problema inicial

$$\min_{x} f(x)$$

Condiciones de optimalidad

Consideremos el problema inicial

$$\min_x f(x)$$

Teorema: Condición Necesaria de Primer Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ continuamente diferenciable en un conjunto abierto D. Si x^* es un mínimo local, entonces $\nabla f(x^*) = 0$.

Condiciones de optimalidad

Consideremos el problema inicial

$$\min_x f(x)$$

Teorema: Condición Necesaria de Primer Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ continuamente diferenciable en un conjunto abierto D. Si x^* es un mínimo local, entonces $\nabla f(x^*) = 0$.

Teorema: Condición Necesaria de Segundo Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ dos veces continuamente diferenciable en un conjunto abierto D. Si x^* es un mínimo local, entonces $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es definida positiva.

Condiciones de optimalidad

Consideremos el problema inicial

$$\min_x f(x)$$

Teorema: Condición Necesaria de Primer Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ continuamente diferenciable en un conjunto abierto D. Si x^* es un mínimo local, entonces $\nabla f(x^*) = 0$.

Teorema: Condición Necesaria de Segundo Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ dos veces continuamente diferenciable en un conjunto abierto D. Si x^* es un mínimo local, entonces $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es definida positiva.

Teorema: Condición Suficiente de Segundo Orden

Sea $f : D \subset \mathbb{R}^n \to \mathbb{R}$ dos veces continuamente diferenciable en un conjunto abierto D. Si $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es definida positiva, entonces $x^* \in D$ es un mínimo local.

Algoritmos. ¿Qué son?

- Conjunto de órdenes con normas y condiciones.

Algoritmos. ¿Qué son?

- Conjunto de órdenes con normas y condiciones.
- Estructura de bucles iterativos.

Algoritmos. ¿Qué son?

- Conjunto de órdenes con normas y condiciones.
- Estructura de bucles iterativos.
- Eficiencia computacional.

Algoritmos. ¿Qué son?

- Conjunto de órdenes con normas y condiciones.
- Estructura de bucles iterativos.
- Eficiencia computacional.
- Mínimos locales y globales.

Algoritmos. Esquema.

Paso 1. Paso inicial. Se definen las condiciones iniciales, suelen ser el primer punto de la sucesión $\{x_k\}$, valores para definir las condiciones de parada como el número máximo de iteraciones o el error aceptable, y otros parámetros.

Algoritmos. Esquema.

Paso 1. Paso inicial. Se definen las condiciones iniciales, suelen ser el primer punto de la sucesión $\{x_k\}$, valores para definir las condiciones de parada como el número máximo de iteraciones o el error aceptable, y otros parámetros.

Paso 2. Test de parada. Se comprueba si se cumple alguna condición de parada.

Algoritmos. Esquema.

Paso 1. Paso inicial. Se definen las condiciones iniciales, suelen ser el primer punto de la sucesión $\{x_k\}$, valores para definir las condiciones de parada como el número máximo de iteraciones o el error aceptable, y otros parámetros.

Paso 2. Test de parada. Se comprueba si se cumple alguna condición de parada.

Paso 3. Proceso principal. Se realizan los cálculos necesarios para avanzar de x_k a x_{k+1}.

Algoritmos. Esquema.

Paso 1. Paso inicial. Se definen las condiciones iniciales, suelen ser el primer punto de la sucesión $\{x_k\}$, valores para definir las condiciones de parada como el número máximo de iteraciones o el error aceptable, y otros parámetros.

Paso 2. Test de parada. Se comprueba si se cumple alguna condición de parada.

Paso 3. Proceso principal. Se realizan los cálculos necesarios para avanzar de x_k a x_{k+1}.

Paso 4. Actualización y bucle. Se actualiza el valor de x_k a x_{k+1}, así como otros parámetros necesarios. Se repite el proceso desde el *paso 2.*

Búsqueda de línea.

- Se toma el punto inicial x_k (para cada iteración).
- Elige una dirección d_k con la misma dimensión que x_k.
  - La mayoría eligen una dirección de descenso, esto es, $d_k^T \nabla f_k < 0$.
  - Suele tener la forma $d_k = -B_k^{-1} \nabla f_k$.
- Se elige una longitud de paso $\alpha_k \in \mathbb{R}$.
- Se obtiene $x_{k+1} = x_k + \alpha_k d_k$.
  - El objetivo es que $f(x_{k+1}) < f(x_k)$.
  - Si se se soluciona el problema $\min_{\alpha_k} f(x_k + \alpha_k d_k)$, búsqueda de línea exacta.

Región de confianza.

- Se toma una pequeña región donde es más fácil predecir como se comporta la función.
- Primero se fija una distancia máxima Δ_k para definir una región

$$\Omega_k = \{x : \|x - x_k\| \leq \Delta_k\}.$$

- Para predecir el comportamiento en esta región, se utilizan aproximaciones como la siguiente:

$$m_k(p) := q^{(k)}(p) = f(x_k) + g_k^T p + \frac{1}{2} p^T G_k p.$$

Región de confianza.

Subproblema de optimización con restricciones

$$\begin{aligned} \min_{p} \quad & m_k(p) = f(x_k) + g_k^T p + \frac{1}{2} p^T B_k p \\ \text{s.a.} \quad & \|p\| \leq \Delta_k, \end{aligned}$$

Región de confianza.

Subproblema de optimización con restricciones

$$\begin{aligned} \min_{p} \quad & m_k(p) = f(x_k) + g_k^T p + \frac{1}{2} p^T B_k p \\ \text{s.a.} \quad & \|p\| \leq \Delta_k, \end{aligned}$$

Jorge Nocedal y Stephen Wright. *Numerical Optimization*. 2.ª ed. New York, NY: Springer, 2006

Wenyu Sun y Ya-Xiang Yuan. *Optimization theory and methods: Nonlinear programming*. 2006.ª ed. New York, NY: Springer, 2006

Región de confianza.

Teorema: Caracterización de la solución del subproblema

El vector p^* es una solución global si y solo si p^* es factible y existe un escalar $\lambda^* \geq 0$ tal que:

- $(B + \lambda^* I)p^* = -g$,
- $\lambda^*(\Delta - \|p^*\|) = 0$,
- $(B + \lambda^* I)$ es semidefinida positiva.

Región de confianza.

Ratio de aceptación

Para tener en cuenta la semejanza entre la aproximación y la función real, se define el ratio de aceptación:

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m(p_k)},$$

el cual será más grande cuanto más parecida sea la aproximación a la función real. Se tiene en cuenta para la elección de Δ_k.

Región de confianza. Esquema.

Paso 1. Dados $x_0, \bar{\Delta}, \Delta_0 \in (0, \bar{\Delta}), \epsilon \geq 0, 0 < \eta_1 \leq \eta_2 < 1$ y $0 < \gamma_1 < 1 < \gamma_2, k := 0$.

Paso 2. Si $\|g_k\| \leq \epsilon$ terminar.

Paso 3. Aproximar p_k resolviendo según la caracterización.

Paso 4. Calcular $f(x_k + p_k)$ y ρ_k. Definir

$$x_{k+1} = \begin{cases} x_k + p_k, & \text{si } \rho_k \geq \eta_1, \\ x_k, & \text{en otro caso.} \end{cases}$$

Paso 5. Si $\rho_k < \eta_1$ entonces $\Delta_{k+1} \in (0, \gamma_1 \Delta_k]$.
Si $\rho_k \in [\eta_1, \eta_2)$ entonces $\Delta_{k+1} \in [\gamma_1 \Delta_k, \Delta_k]$.
Si $\rho_k \geq \eta_2$ y $\|p_k\| = \Delta_k$ entonces
$\Delta_{k+1} \in [\Delta_k, \text{mín}\{\gamma_2 \Delta_k, \bar{\Delta}\}]$.

Paso 6. Calcular B_{k+1}, actualizar $m^{(k)}$ y $k := k + 1$. Ir al Paso 2.

Índice.

El problema.

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n,$$

El problema.

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n,$$

- x es el parámetro a estimar de dimensión n.

El problema.

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n,$$

- x es el parámetro a estimar de dimensión n.
- $\phi_x(t)$ es la función a ajustar.

El problema.

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n,$$

- x es el parámetro a estimar de dimensión n.
- $\phi_x(t)$ es la función a ajustar.
- $r_i(x) = \phi_x(t_i) - y_i$, para $i = 1, \ldots, m$ son los residuos.

El problema.

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n,$$

- x es el parámetro a estimar de dimensión n.
- $\phi_x(t)$ es la función a ajustar.
- $r_i(x) = \phi_x(t_i) - y_i$, para $i = 1, \ldots, m$ son los residuos.

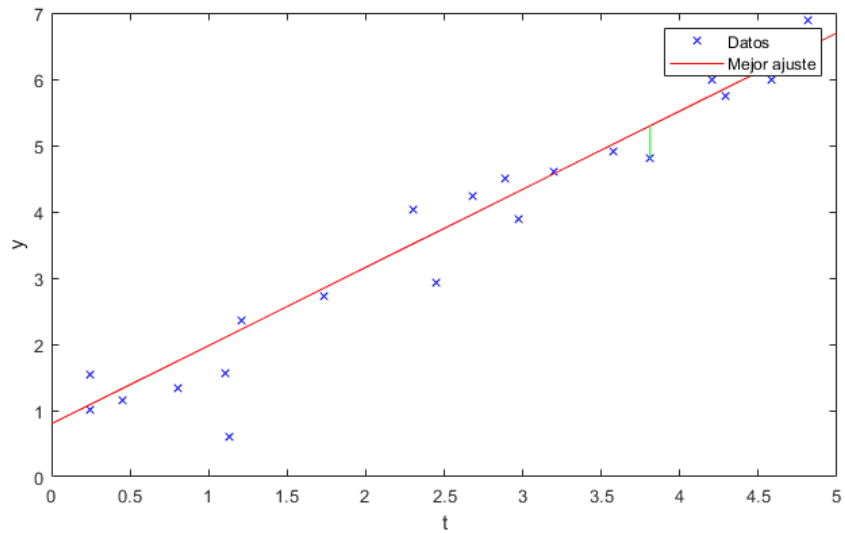
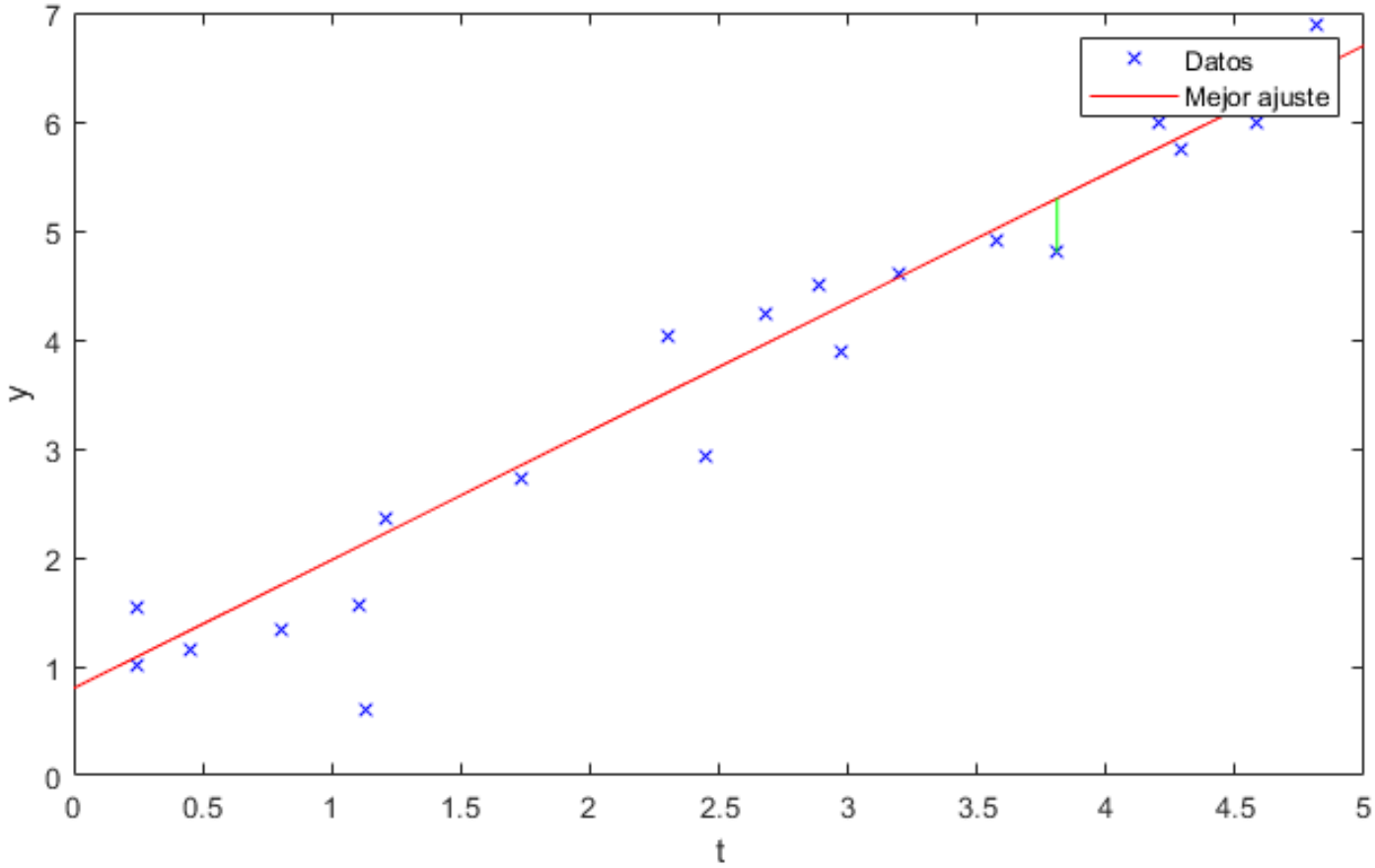
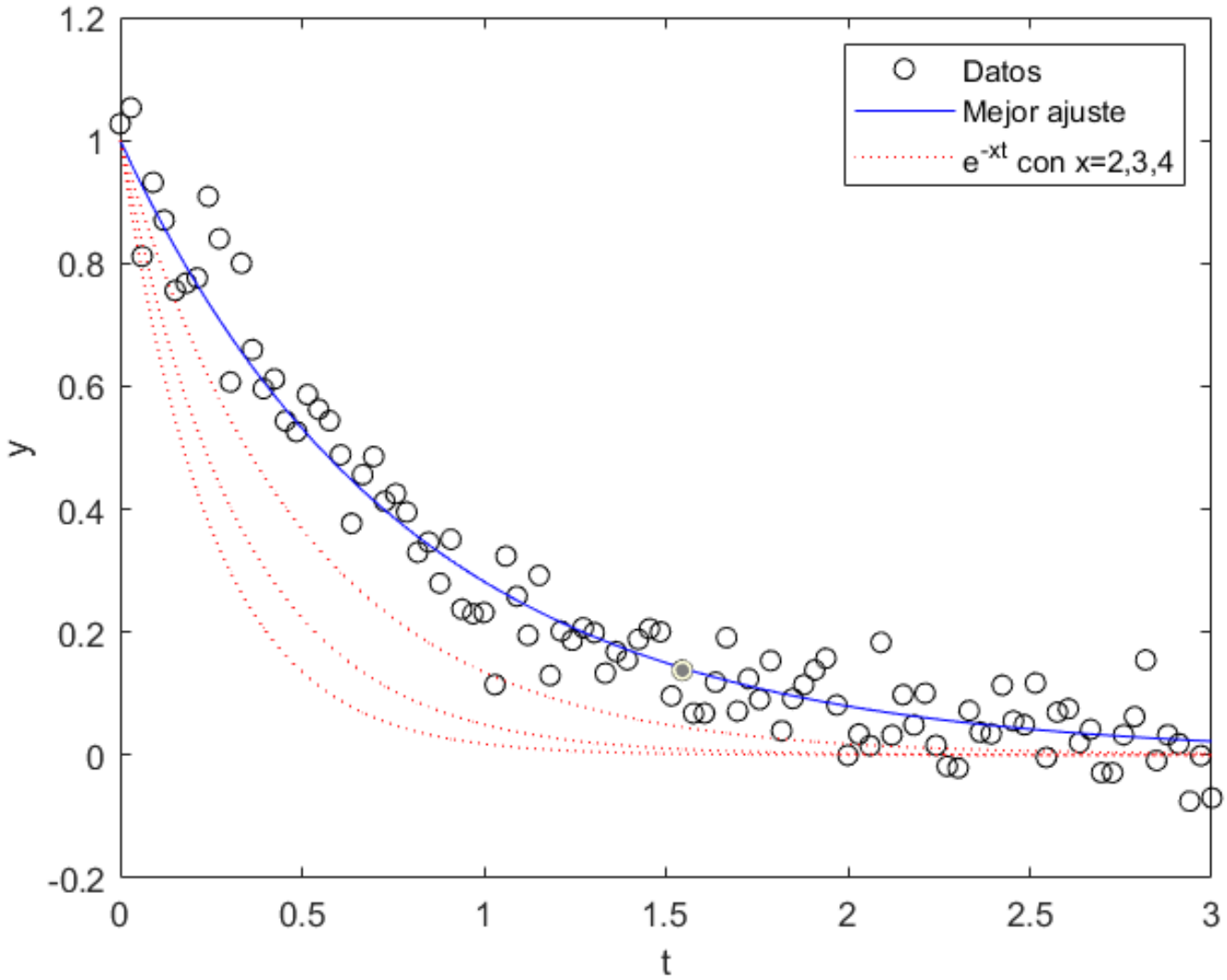


Figura: Ejemplo de ajuste de una recta a una muestra de puntos

Ejemplo de ajuste de una recta a una muestra de puntos.



Ejemplo de ajuste de una exponencial a una muestra de puntos.



El método de Gauss-Newton.

El método de Gauss-Newton.

- Se trata de una linealización del problema de mínimos cuadrados.

El método de Gauss-Newton.

- Se trata de una linealización del problema de mínimos cuadrados.
- Para linealizar, se desprecia el término cuadrático de $r_i(x)$ y se obtiene el caso lineal:

$$f(x) = \frac{1}{2}\|Jx - y\|^2.$$

$$\nabla f(x) = J^T(Jx - y), \qquad \nabla^2 f(x) = J^T J.$$

El método de Gauss-Newton.

- Se trata de una linealización del problema de mínimos cuadrados.
- Para linealizar, se desprecia el término cuadrático de $r_i(x)$ y se obtiene el caso lineal:

$$f(x) = \frac{1}{2}\|Jx - y\|^2.$$

$$\nabla f(x) = J^T(Jx - y), \qquad \nabla^2 f(x) = J^T J.$$

- Se resuelve el sistema lineal $J^T Jx = J^T y$.

El método de Gauss-Newton.

- Se trata de una linealización del problema de mínimos cuadrados.
- Para linealizar, se desprecia el término cuadrático de $r_i(x)$ y se obtiene el caso lineal:

$$f(x) = \frac{1}{2}\|Jx - y\|^2.$$

$$\nabla f(x) = J^T(Jx - y), \qquad \nabla^2 f(x) = J^T J.$$

- Se resuelve el sistema lineal $J^T Jx = J^T y$.
- Convergencia local y cuadrática.

El método de Gauss-Newton.

Paso 1. Dados x_0 y $\epsilon > 0$, $k := 0$.

Paso 2. Si $\|g_k\| \leq \epsilon$, terminar.

Paso 3. Obtener el paso p_k resolviendo

$$J(x_k)^T J(x_k) p_k = -J(x_k)^T r(x_k). \tag{1}$$

Paso 4. Definimos $x_{k+1} = x_k + p_k$ y actualizamos $k = k + 1$. Ir a Paso 2.

Índice.

El método de Levenberg-Marquardt.

- Se mantiene la idea de la linealización de Gauss-Newton.

El método de Levenberg-Marquardt.

- Se mantiene la idea de la linealización de Gauss-Newton.
- Se cambia de enfoque a región de confianza.

El método de Levenberg-Marquardt.

- Se mantiene la idea de la linealización de Gauss-Newton.
- Se cambia de enfoque a región de confianza.

$$\begin{aligned} \min_{p} \quad & m_k(p) = \frac{1}{2}\|r_k\|^2 + p^T J_k^T r_k + \frac{1}{2} p^T J_k^T J_k p, \\ \text{s.a.} \quad & \|p\| \leq \Delta_k. \end{aligned}$$

El método de Levenberg-Marquardt.

- Se mantiene la idea de la linealización de Gauss-Newton.
- Se cambia de enfoque a región de confianza.

$$\begin{aligned} \min_{p} \quad & m_k(p) = \frac{1}{2}\|r_k\|^2 + p^T J_k^T r_k + \frac{1}{2} p^T J_k^T J_k p, \\ \text{s.a.} \quad & \|p\| \leq \Delta_k. \end{aligned}$$

Lema: Caracterización de la solución.

El vector p es solución del subproblema si y solo si p es factible y existe un $\lambda \geq 0$ tal que

$$\begin{aligned} (J^T J + \lambda I)p &= -J^T r, \\ \lambda(\Delta - \|p\|) &= 0. \end{aligned}$$

Propiedades del método de Levenberg-Marquardt.

- Convergencia global bajo ciertas condiciones.
- El ratio de convergencia es variable (ver Teorema 3.7).

Wenyu Sun y Ya-Xiang Yuan. *Optimization theory and methods: Nonlinear programming*. 2006.ª ed. New York, NY: Springer, 2006

Implementación del método de Levenberg-Marquardt.

Versión de Moré

Disponemos de la implementación de este método propuesta por Jorge J. Moré, con un estudio detallado del mismo en su artículo[1].

[1] Jorge J Moré. "The Levenberg-Marquardt algorithm: Implementation and theory". En: *Lecture Notes in Mathematics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, págs. 105-116.

Implementación del método de Levenberg-Marquardt.

Versión de Moré

Disponemos de la implementación de este método propuesta por Jorge J. Moré, con un estudio detallado del mismo en su artículo[1].

Veremos a continuación en detalle:

- Actualización del radio Δ_k.

[1] Jorge J Moré. "The Levenberg-Marquardt algorithm: Implementation and theory". En: *Lecture Notes in Mathematics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, págs. 105-116.

Implementación del método de Levenberg-Marquardt.

Versión de Moré

Disponemos de la implementación de este método propuesta por Jorge J. Moré, con un estudio detallado del mismo en su artículo[1].

Veremos a continuación en detalle:

- Actualización del radio Δ_k.
- El parámetro Levenberg-Marquardt.

[1] Jorge J Moré. "The Levenberg-Marquardt algorithm: Implementation and theory". En: *Lecture Notes in Mathematics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, págs. 105-116.

Implementación del método de Levenberg-Marquardt.

Versión de Moré

Disponemos de la implementación de este método propuesta por Jorge J. Moré, con un estudio detallado del mismo en su artículo[1].

Veremos a continuación en detalle:

- Actualización del radio Δ_k.
- El parámetro Levenberg-Marquardt.
- Matriz de escalado D_k.

La matriz D_k se sustituye en la siguiente expresión por la matriz identidad:

$$(J^T J + \lambda I)p = -J^T r.$$

[1] Jorge J Moré. "The Levenberg-Marquardt algorithm: Implementation and theory". En: *Lecture Notes in Mathematics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1978, págs. 105-116.

Implementación del método de Levenberg-Marquardt.

Actualización del radio Δ_k

En este caso particular el ratio viene dado por

$$\rho = \frac{\|r(x_k)\|^2 - \|r(x_k + p_k)\|^2}{\|r(x_k)\|^2 - \|r(x_k) + J(x_k)p_k\|^2},$$

y obtenemos la siguiente expresión explícita:

$$\rho = \frac{1 - \left[\frac{\|r(x_k+p_k)\|}{\|r(x_k)\|}\right]^2}{\left[\frac{\|J_k p\|}{\|r(x_k)\|}\right]^2 + 2\left[\frac{\lambda^{1/2}\|D_k p\|}{\|r(x_k)\|}\right]^2}.$$

Implementación del método de Levenberg-Marquardt.

El parámetro Levenberg-Marquardt

En esta implementación, se acepta $\alpha > 0$ como parámetro de Levenberg-Marquardt Si

$$|\phi(\alpha)| \leq \sigma\Delta,$$

siendo

$$\phi(\alpha) = \left\| D(J^T J + \alpha D^T D)^{-1} J^T r \right\| - \Delta,$$

y $\sigma \in (0, 1)$ es un parámetro que controla la aceptación de α.

Implementación del método de Levenberg-Marquardt.

Paso 1. Definir valores iniciales α_0, l_0 y u_0. Definir coeficiente de parada σ.

Paso 2. Si $\alpha_k \notin (l_k, u_k)$, definir $\alpha_k = \text{máx}\{0{,}001 u_k, (l_k u_k)^{1/2}\}$.

Paso 3. Calcular $\phi(\alpha_k)$ y $\phi'(\alpha_k)$. Si $|\phi(\alpha)| < \sigma\Delta$, terminar.

Paso 4. Actualizar l_k y u_k:

$$l_{k+1} = \text{máx}\{l_k, \alpha_k - \frac{\phi(\alpha_k)}{\phi'(\alpha_k)}\},$$

$$u_{k+1} = \begin{cases} \alpha_k, & \text{si } \phi(\alpha_k) < 0, \\ u_k, & \text{en otro caso.} \end{cases}$$

Paso 5. Actualizar α_{k+1} según

$$\alpha_{k+1} = \alpha_k - \left[\frac{\phi(\alpha_k) + \Delta}{\Delta}\right]\left[\frac{\phi(\alpha_k)}{\phi'(\alpha_k)}\right].$$

Volver al Paso 2.

Implementación del método de Levenberg-Marquardt.

Escalado

Para reducir posibles problemas de escalado, se utiliza una matriz diagonal D_k definido de la siguiente forma:

$$D_k = diag(d_1^{(k)}, \ldots, d_n^{(k)}),$$

con

$$d_i^{(k)} = \|\partial_i r(x_k)\|, k \geq 0.$$

En cada paso, se actualiza cada $d_i^{(k)}$ si este aumenta, es decir,

$$d_i^{(k)} = \text{máx}\{d_i^{(k-1)}, \|\partial_i r(x_k)\|\}, k \geq 1.$$

Esquema del algoritmo. I.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

Esquema del algoritmo. I.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

Paso 2. Si $\left\| J_k (D_k)^{-1} f(x_k) \right\| < \epsilon$, terminar.

Esquema del algoritmo. I.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

Paso 2. Si $\left\| J_k (D_k)^{-1} f(x_k) \right\| < \epsilon$, terminar.

Paso 3. Sea $\sigma \in (0,1)$. Si $\left\| D_k J_k^- r_k \right\| \leq (1+\sigma)\Delta_k$, asignar $\lambda_k = 0$ y $p_k = -J_k^- r_k$. En otro caso, encontar λ_k tal que si

$$\begin{bmatrix} J_k \\ \lambda_k^{\frac{1}{2}} D_k \end{bmatrix} p_k \cong - \begin{bmatrix} r_k \\ 0 \end{bmatrix},$$

entonces

$$(1-\sigma)\Delta_k \leq \|D_k p_k\| \leq (1+\sigma)\Delta_k.$$

Esquema del algoritmo. I.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

Paso 2. Si $\left\| J_k (D_k)^{-1} f(x_k) \right\| < \epsilon$, terminar.

Paso 3. Sea $\sigma \in (0,1)$. Si $\left\| D_k J_k^- r_k \right\| \leq (1+\sigma)\Delta_k$, asignar $\lambda_k = 0$ y $p_k = -J_k^- r_k$. En otro caso, encontar λ_k tal que si

$$\begin{bmatrix} J_k \\ \lambda_k^{\frac{1}{2}} D_k \end{bmatrix} p_k \cong - \begin{bmatrix} r_k \\ 0 \end{bmatrix},$$

entonces

$$(1-\sigma)\Delta_k \leq \|D_k p_k\| \leq (1+\sigma)\Delta_k.$$

Paso 4. Determinar la relación ρ_k entre reducción real y la esperada.

Esquema del algoritmo. II.

Paso 5. Si $\rho_k \leq 0{,}0001$, asignar $x_{k+1} = x_k$ y $J_{k+1} = J_k$.
Si $\rho_k > 0{,}0001$, asignar $x_{k+1} = x_k + p_k$ y calcular
$J_{k+1} = J(x_{k+1})$.

Esquema del algoritmo. II.

Paso 5. Si $\rho_k \leq 0{,}0001$, asignar $x_{k+1} = x_k$ y $J_{k+1} = J_k$.
Si $\rho_k > 0{,}0001$, asignar $x_{k+1} = x_k + p_k$ y calcular $J_{k+1} = J(x_{k+1})$.

Paso 6. Si $\rho_k \leq \frac{1}{4}$, asignar $\Delta_{k+1} \in \left[\frac{1}{10}\Delta_k, \frac{1}{4}\Delta_k\right]$.
Si $\rho_k \in \left[\frac{1}{4}, \frac{3}{4}\right]$ y $\lambda_k = 0$, o si $\rho_k \geq \frac{3}{4}$, asignar $\Delta_{k+1} = 2\|D_k p_k\|$.

Esquema del algoritmo. II.

Paso 5. Si $\rho_k \leq 0{,}0001$, asignar $x_{k+1} = x_k$ y $J_{k+1} = J_k$.
Si $\rho_k > 0{,}0001$, asignar $x_{k+1} = x_k + p_k$ y calcular $J_{k+1} = J(x_{k+1})$.

Paso 6. Si $\rho_k \leq \frac{1}{4}$, asignar $\Delta_{k+1} \in \left[\frac{1}{10}\Delta_k, \frac{1}{4}\Delta_k\right]$.
Si $\rho_k \in \left[\frac{1}{4}, \frac{3}{4}\right]$ y $\lambda_k = 0$, o si $\rho_k \geq \frac{3}{4}$, asignar $\Delta_{k+1} = 2\|D_k p_k\|$.

Paso 7. Actualizar D_k y volver al Paso 2.

Índice.

Inicialización.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

Inicialización.

Paso 1. Definir valores iniciales x_0, Δ_0 y ϵ y determinar J_k, D_k, y $f(x_0)$ a partir de la función f determinada.

```
1  function [xk, iter] = myLevMar(fun, x0)
2  % setup
3  [n,m,xk,Delta,funk,lambdak, maxit, maxit2,
       eps]= myLevMarSetup(fun,x0);
4
5  % calculo inicial de Jk y Dk
6  dkm1=zeros(length(xk));
7  [Jk, Dk, dkm1] = JDk(fun,xk,dkm1);
```

Inicialización.

```
1  function [n,m,xk,Delta,funk,lambdak, maxit,
       maxit2, eps] = myLevMarSetup(fun,x0)
2  % editable
3  Delta = 3;
4  lambdak = 1;
5  maxit = 50;
6  maxit2 = 10; % miteraciones maximas en
       hebden.m
7  eps = 0.001;
8
9  % no editable
10 n = length(x0);
11 m = length(fun(x0));
12 xk = x0;
13 funk = fun(xk)';
```

Inicialización.

```
1  function [Jk,Dk,v] = JDk(fun,xk,dkm1)
2
3  Jk = jacob(fun,xk);
4
5  % Compute diagonal of Dk
6  di = @(i) max(dkm1(i),norm(Jk(:,i)));
7
8  v = arrayfun(di, 1:length(xk));
9  Dk = diag(v);
```

Inicialización.

```
1  function J = jacob(fun,xk)
2  h = 1e-5;
3  n = length(xk);
4  m = length(fun(xk));
5  J = zeros(m,n);
6  for j = 1:n
7       hj = zeros(1,n);
8       hj(j) = h;
9       J(:,j) = (fun(xk+hj)-fun(xk))./h;
10 end
```

Test de parada.

Paso 2. Si $\left\| J_k(D_k)^{-1} f(x_k) \right\| < \epsilon$, terminar.

Test de parada.

Paso 2. Si $\left\| J_k(D_k)^{-1} f(x_k) \right\| < \epsilon$, terminar.

```
try
     test = norm((Jk*inv(Dk))'*funk);
catch
     test = norm((Jk*inv(Dk))'*funk');
end
iter=0;
while test > eps && iter < maxit
```

Bucle.

```
1 % solucion problema lineal
2 % descomposicion QR
3 [Q,R,Pi] = qr(Jk); % calcula Jk*Pi = Q*R
4 % ajuste de R
5 T = R(1:n,1:n);
6 S = R(1:n,n+1:end);
7
8 % matriz $J^{-}$
9 Jm = Jmfun(Pi,T,Q,n,m);
```

Bucle.

Paso 3. Sea $\sigma \in (0,1)$. Si $\left\| D_k J_k^- r_k \right\| \leq (1+\sigma)\Delta_k$, asignar $\lambda_k = 0$ y $p_k = -J_k^- r_k$. En otro caso, encontar λ_k tal que si

$$\begin{bmatrix} J_k \\ \lambda_k^{\frac{1}{2}} D_k \end{bmatrix} p_k \cong - \begin{bmatrix} r_k \\ 0 \end{bmatrix},$$

entonces $(1-\sigma)\Delta_k \leq \|D_k p_k\| \leq (1+\sigma)\Delta_k$.

Bucle.

Paso 3. Sea $\sigma \in (0,1)$. Si $\left\| D_k J_k^- r_k \right\| \leq (1+\sigma)\Delta_k$, asignar $\lambda_k = 0$ y $p_k = -J_k^- r_k$. En otro caso, encontar λ_k tal que si

$$\begin{bmatrix} J_k \\ \lambda_k^{\frac{1}{2}} D_k \end{bmatrix} p_k \cong - \begin{bmatrix} r_k \\ 0 \end{bmatrix},$$

entonces $(1-\sigma)\Delta_k \leq \|D_k p_k\| \leq (1+\sigma)\Delta_k$.

```
% parametro levenberg-marquardt
[pk, lambdak] = pkfun(Delta, Jk, Dk, funk,
    Jm, Q, Pi, n, m, lambdak, maxit2);
```

Bucle.

```
1  function [pk, lambdak] = pkfun(Delta, Jk, Dk
       , funk, Jm, Q, Pi, n, m, lambdak, maxit)
2
3  phi = @(a) norm(Dk*inv(Jk'*Jk+a.*(Dk'*Dk))*
       Jk'*funk) - Delta;
4  if phi(0) == 0
5       lambdak = 0;
6       pk = -Jm*funk;
7       pk = pk';
8  else
9       lambdak = hebden(phi, Delta, Jk, Dk,
            funk, Q, Pi, n, m, lambdak, maxit);
10      pk = pk2(funk, Q, Pi, Jk, Dk, lambdak, n
            , m);
11 end
```

Bucle.

Paso 4. Determinar la relación ρ_k entre reducción real y la esperada.

Bucle.

Paso 4. Determinar la relación ρ_k entre reducción real y la esperada.

```
1 rho = rhofun(funk,pk,lambdak,Dk,Jk,fun, xk);
```

Bucle.

Paso 4. Determinar la relación ρ_k entre reducción real y la esperada.

```
1 rho = rhofun(funk,pk,lambdak,Dk,Jk,fun, xk);
```

```
1 function rho = rhofun(funk,pk,lambdak,Dk,Jk,
      fun,xk)
2 funplus = fun(xk+pk);
3
4 rhonum=1-(norm(funplus)/norm(funk))^2;
5 rhoden=(norm(Jk*pk')/norm(funk))^2+2*(sqrt(
      lambdak)*norm(Dk*pk')/norm(funk))^2;
6 rho = rhonum/rhoden;
```

Bucle.

Paso 5. Si $\rho_k \leq 0{,}0001$, asignar $x_{k+1} = x_k$ y $J_{k+1} = J_k$.
Si $\rho_k > 0{,}0001$, asignar $x_{k+1} = x_k + p_k$ y calcular
$J_{k+1} = J(x_{k+1})$.

Bucle.

Paso 5. Si $\rho_k \leq 0{,}0001$, asignar $x_{k+1} = x_k$ y $J_{k+1} = J_k$.
Si $\rho_k > 0{,}0001$, asignar $x_{k+1} = x_k + p_k$ y calcular
$J_{k+1} = J(x_{k+1})$.

```
1  if rho <= 0.0001
2      % xk+1 = xk y jk+1=jk
3  else
4      xk = xk+pk;
5      % calculo de Jk, Dk con el nuevo xk
6      funk = fun(xk)';
7      [Jk, Dk, dkm1] = JDk(fun,xk,dkm1);
8      iter=iter+1;
9      test = norm((Jk*inv(Dk))'*fun(xk)');
10 end
```

Bucle.

Paso 6. Si $\rho_k \leq \frac{1}{4}$, asignar $\Delta_{k+1} \in \left[\frac{1}{10}\Delta_k, \frac{1}{4}\Delta_k\right]$.
Si $\rho_k \in \left[\frac{1}{4}, \frac{3}{4}\right]$ y $\lambda_k = 0$, o si $\rho_k \geq \frac{3}{4}$, asignar
$\Delta_{k+1} = 2\left\|D_k p_k\right\|$.

Bucle.

Paso 6. Si $\rho_k \leq \frac{1}{4}$, asignar $\Delta_{k+1} \in \left[\frac{1}{10}\Delta_k, \frac{1}{4}\Delta_k\right]$.
Si $\rho_k \in \left[\frac{1}{4}, \frac{3}{4}\right]$ y $\lambda_k = 0$, o si $\rho_k \geq \frac{3}{4}$, asignar $\Delta_{k+1} = 2\left\|D_k p_k\right\|$.

```
% actualizacion de Delta
if rho <= 0.25
    Delta = Delta/2;
elseif rho >= 0.75
    Delta = 2*norm(Dk*pk');
elseif lambdak == 0
    Delta = 2*norm(Dk*pk');
end
```

Bucle.

Paso 7. Actualizar D_k y volver al Paso 2.

Bucle.

Paso 7. Actualizar D_k y volver al Paso 2.

```
    end
xk = real(xk);
```

Fin.

Resolución numérica del problema no lineal de mínimos cuadrados. Aplicaciones a las estimación de parámetros de modelos matemáticos.

Dídac Blanco Morros.

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
FACULTADE DE MATEMÁTICAS