Optimización y Control

Ignacio Delgado, Ignacio Gómez,
José M. Perales, José M. Vega

Escuela de Ingeniería Aeronáutica y del Espacio
Programa: **Grado en Ingeniería Aeronáutica**
Curso 2017-2018
Tema #2: Extremos locales sin restricciones

Index

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

PLANTEAMIENTO Y NOTACIÓN

Se trata de minimizar una **función objetivo**, que depende de N **variables de diseño**, $u_1, \ldots, u_N$; el vector $\boldsymbol{u} = (u_1, \ldots, u_N)$ se conoce como **vector de diseño**.

A lo largo de este curso se utiliza la siguiente **notación**:

- Las **letras negritas minúsculas** denotan, indistintamente, vectores o matrices columna, tales como $\boldsymbol{u} = (u_1, \ldots, u_N)$ o $\boldsymbol{u} = [u_1, \ldots, u_N]^\top$, donde el superíndice $^\top$ denota la transpuesta.
- El producto escalar de dos vectores, $\boldsymbol{u}^1 = (u_1^1, \ldots, u_N^1)$ y $\boldsymbol{u}^2 = (u_1^2, \ldots, u_N^2)$, se escribe entonces, indistintamente, o bien como $\boldsymbol{u}^1 \cdot \boldsymbol{u}^2$ o como $\boldsymbol{u}^{1\top} \boldsymbol{u}^2$, ya que

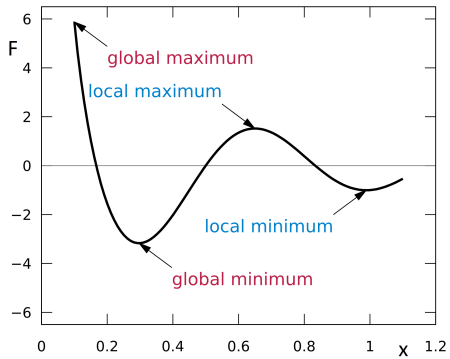
$$\boldsymbol{u}^1 \cdot \boldsymbol{u}^2 = u_1^1 u_1^2 + \ldots + u_N^1 u_N^2 = \boldsymbol{u}^{1\top} \boldsymbol{u}^2.$$

- Las **letras mayúsculas negritas denotan matrices** con más de una columna, tales como la matriz jacobiana y la matriz hessiana, que aparecerán con mucha frecuencia en lo sucesivo.

MÉTODOS LOCALES (I)

En este tema se ignoran las restricciones.

- Los métodos de tipo gradiente son **métodos iterativos** que se basan en **aproximaciones locales**, mediante desarrollos de Taylor.
- Requiren **hipótesis de diferenciabilidad** (derivadas segundas continuas de la función objetivo).
- Todos estos métodos requiren calcular el **gradiente** de la función objetivo en cada iteración.
- Proporcionan **óptimos locales**, es decir, mejores valores de la función objetivo que los de otros puntos en un entorno del espacio de diseño.



MÉTODOS LOCALES (I)

En este tema se ignoran las restricciones.

- Requiren una **condición inicial** que, en principio, debe ser próxima al mínimo local que se busca. En ausencia de ésta, pueden converger mal o no coverger.
- No obstante, combinados con otros ingredientes, tales como regiones de confianza o métodos de descenso, puede aegurarse convergencia (a mínimos locales).
- Son **métodos superlineales**: el error de la aproximación en las sucesivas iteraciones se comporta como $e_{k+1} \simeq ce_k^r$ para $k \to \infty$, donde $r > 1$ es el **orden del método**.

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

APROXIMACIÓN DE ORDEN DOS; PRIMERA CONDICIÓN NECESARIA

Los métodos de este tema se basan en la **aproximación de Taylor, de segundo orden,** de la función objetivo en torno a un punto $\boldsymbol{u}^0$:

$$F(\boldsymbol{u}) \simeq F(\boldsymbol{u}^0) + (\boldsymbol{g}^0)^\top (\boldsymbol{u} - \boldsymbol{u}^0) + \frac{1}{2}(\boldsymbol{u} - \boldsymbol{u}^0)^\top \boldsymbol{H}^0 (\boldsymbol{u} - \boldsymbol{u}^0).$$

Aquí, y en lo sucesivo:

- $\boldsymbol{g} = [\partial_{u_1} F, \ldots, \partial_{u_N} F]^\top$ denota el gradiente de la función objetivo.
- $\boldsymbol{H}$ denota la matrix hessiana, cuyos elementos son $H_{ij} = \partial^2_{u_i u_j} F$.

La matriz hessiana es simétrica, de modo que todos sus autovalores son reales.

Si $\boldsymbol{u}^{\min}$ corresponde a un mínimo local de F, debe ser (**primera condición necesaria**)

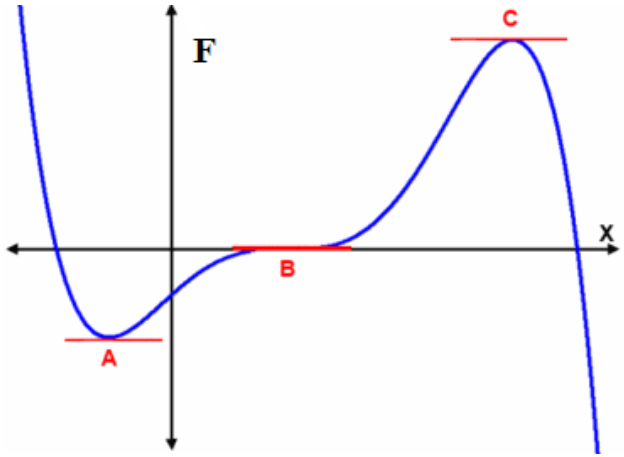
$$\boldsymbol{g}^{\min} = \boldsymbol{0}.$$

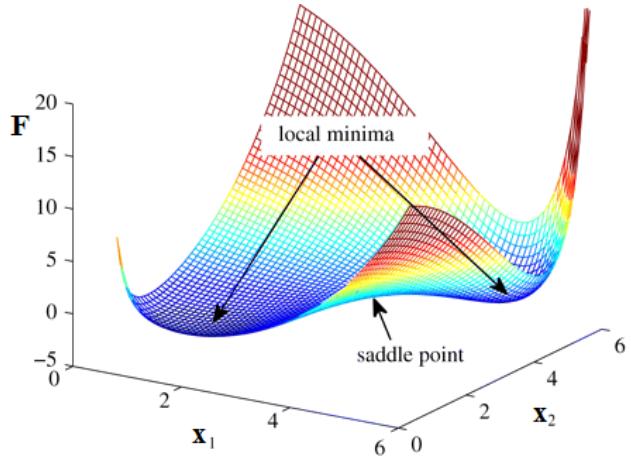
Esta ecuación vectorial proporciona N (tantas como el número de variables de diseño) ecuaciones escalares:

$$\partial_{u_1} F = 0, \ldots, \partial_{u_N} F = 0.$$

PRIMERA CONDICIÓN NECESARIA (PUNTOS ESTACIONARIOS)

Los puntos que verifican la primera condición necesaria (gradiente nulo) son los **puntos estacionarios** de F. En 1D pueden ser mínimos, máximos o puntos de inflexión; en 2D pueden ser mínimos, máximos o puntos de ensilladura.





SEGUNDA CONDICIÓN NECESARIA Y CONDICIÓN SUFICIENTE

Suponiendo que se verifica la condición necesaria, $\boldsymbol{g}^{\min} = \mathbf{0}$, la aproximación de orden dos se simplifica

$$F(\boldsymbol{u}) \simeq F(\boldsymbol{u}^{\min}) + \frac{1}{2}(\boldsymbol{u} - \boldsymbol{u}^{\min})^{\top} \boldsymbol{H}^{\min}(\boldsymbol{u} - \boldsymbol{u}^{\min}),$$

y se tienen las siguientes condiciones para que $\boldsymbol{u}^{\min}$ sea mínimo local:

- **Segunda condición necesaria**: la matriz hessiana es semi-definida positiva. Es decir, $(\boldsymbol{u} - \boldsymbol{u}^{\min})^{\top} \boldsymbol{H}^{\min}(\boldsymbol{u} - \boldsymbol{u}^{\min}) \geq 0$.
- La condición anterior es equivalente a exigir que **todos los autovalores de la matriz hessiana** (que son reales) ≥ 0. Esto require, en particular, que la traza y el determinante de $\boldsymbol{H}_0$ sean, ambos, ≥ 0.
- **Condición suficiente**: La matriz hessiana es definida positiva. Es decir, $(\boldsymbol{u} - \boldsymbol{u}^{\min})^{\top} \boldsymbol{H}^{\min}(\boldsymbol{u} - \boldsymbol{u}^{\min}) > 0$ si $\boldsymbol{u} \neq \boldsymbol{u}^{\min}$.
- La condición anterior es equivalente a exigir que **todos los autovalores de la matriz hessiana** (que son reales) sean > 0. Esto require, en particular, que la traza y el determinante de $\boldsymbol{H}_0$ sean, ambos, > 0

La approximación anterior implica que los errores en F son mucho más pequeños que los errores en $\boldsymbol{u}$; $\text{error}_F \sim (\text{error}_{\boldsymbol{u}})^2$.

SISTEMAS DE ECUACIONES NO LINEALES

Los posibles óptimos locales de la función objetivo $F = F(u_1, \ldots, u_N)$ son las soluciones del sistema de ecuaciones (no lineal, en general)

$$\partial_{u_1} F = 0, \ldots, \partial_{u_N} F = 0.$$

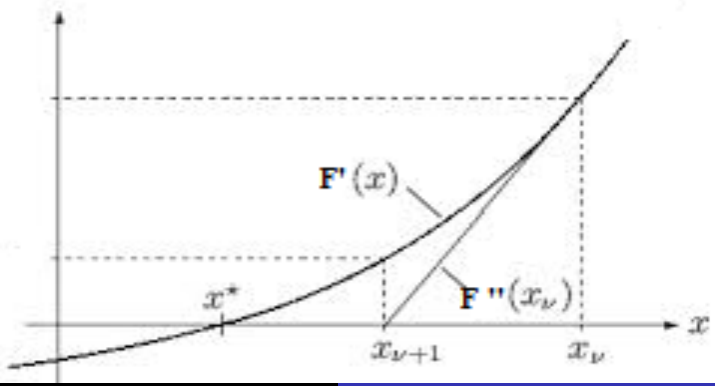
- La matriz jacobiana del sistema anterior es la matriz hessiana de F.
- En dimensión $N = 1$, se tiene una ecuación escalar con una incógnita. Los dos métodos iterativos básicos para resolver la ecuación son: el **método de Newton** y el **método de la secante**.
- Los métodos iterativos básicos para la resolución del sistema de ecuaciones anterior son generalizaciones de estos dos métodos.
- Sin embargo, la generalización de estos métodos (sobre todo, del método de la secante) require cierto cuidado.

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

MÉTODO DE NEWTON EN 1D

- **En el método de Newton**, en cada nueva iteración, el nuevo valor se calcula sustituyendo la ecuación por su recta tangente, y proporciona la nueva iteración como
$$u^{k+1} = u^k - F'(u^k)/F''(u^k).$$
- Se trata de un **método convergente, de segundo orden** (el error se comporta como $e_{k+1} \sim e_k^2$) si $F''(u) \neq 0$ y la iteración inicial es suficientemente buena.
- Por tanto, la convergencia es muy rápida (esencialmente, el número de cifras significativas exactas se dobla en cada iteración) una vez que los iterantes se acercan a la solución.
- Requiere una **estimación inicial** y conocer la **derivada segunda** de la función objetivo, F''.

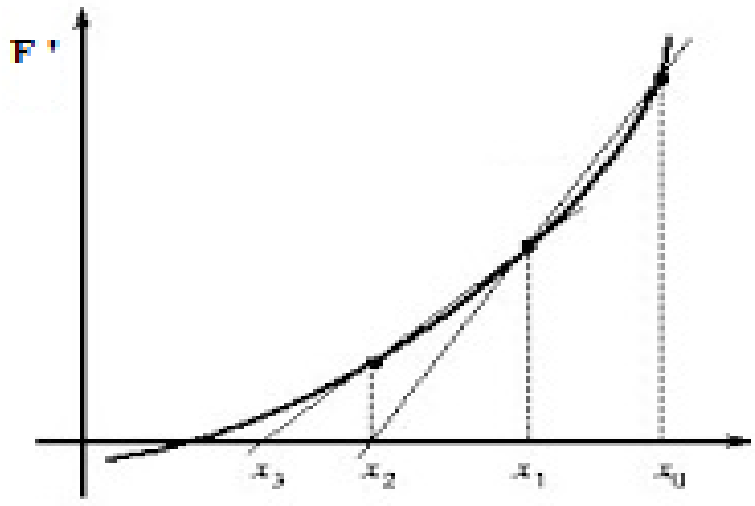


MÉTODO DE LA SECANTE EN 1D (I)

- **En el método de la secante**, cada nueva iteración se calcula, en función de las dos anteriores, mediante la recta que pasa por los dos puntos, como

$$u^{k+1} = u^k - F'(u^k)(u^k - u^{k-1})/[F'(u^k) - F'(u^{k-1})].$$

- Requiere dos aproximaciones iniciales.

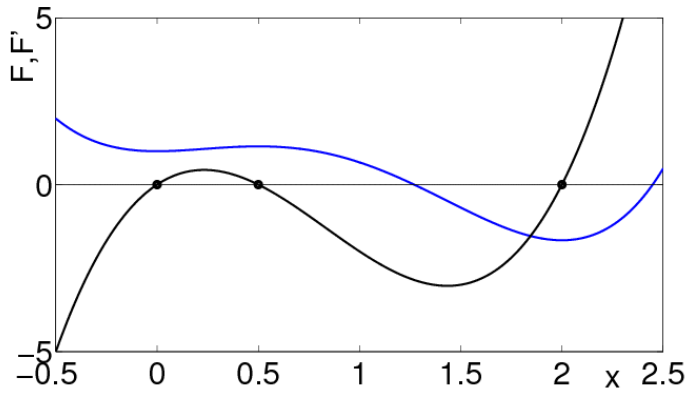


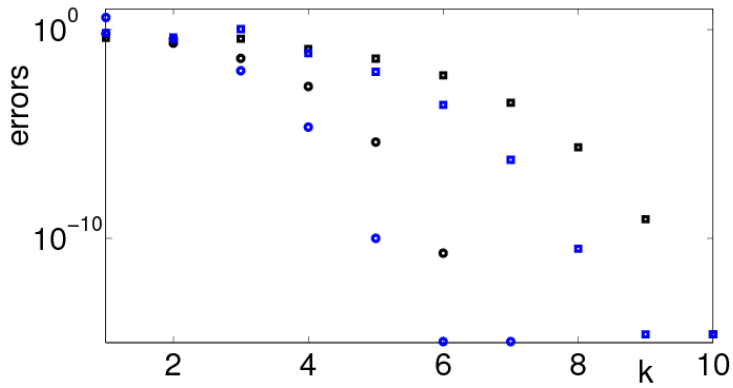
MÉTODO DE LA SECANTE EN 1D (II)

- Se trata de un **método convergente, de orden** $r = (1+\sqrt{5})/2 \simeq 1{,}6$ (el error se comporta como $e_{k+1} \sim e_k^r$) si $F''(u) \neq 0$ y las dos interaciones iniciales son suficientemente buenas.
- Esencialmente, dos pasos de este método son mejores que un paso del método de Newton ($2r \simeq 3{,}2 > 2$).
- Puede hacerse convergente si se modifica adecuadamente (para que $F'(u^{k+1})$ y $F'(u^k)$ tengan signos opuestos).
- No requiere conocer F''; ésta se estima mediante los resultados de las dos últimas iteraciones.
- Puede combinarse con **bisección** (en los primeros pasos) y con **interpolación cuadrática**.

APLICACIÓN A UNA FUNCIÓN 'BIMODAL' (I)

Consideramos la función 'bimodal' $F = (1 + x^2)^2 - 10x^3/3$ (línea azul en la figura izda), cuya derivada es $F' = 4x(1 + x^2) - 10x^2$ (línea negra en la figura izquierda). Presenta dos mínimos locales, en $x = 0$ and 2, where $F = 1$ and $-5/3$, respectivamente (el mínimo global es $F = -5/3$), y un máximo local en $x = 1/2$.





Tomando x^0 =1.6 y x^1 =1.7, tanto el método de Newton como el de la secante convergen al mínimo global. La figura de la izda muestra los errores (en las sucesivas iteraciones) resultantes de aplicar el método de Newton (círculos) y de la secante (cuadrados), midiendo los errores con los valores de x (negro) y de F (azul).

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

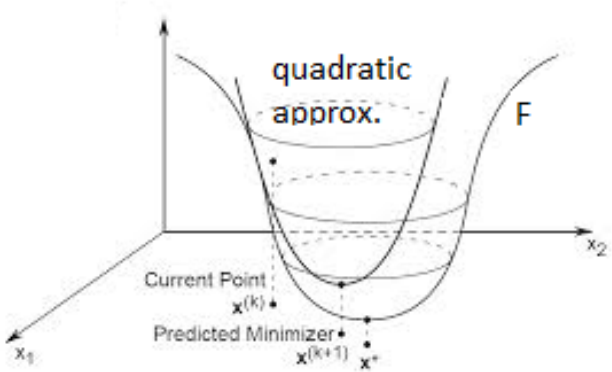
MÉTODO DE NEWTON

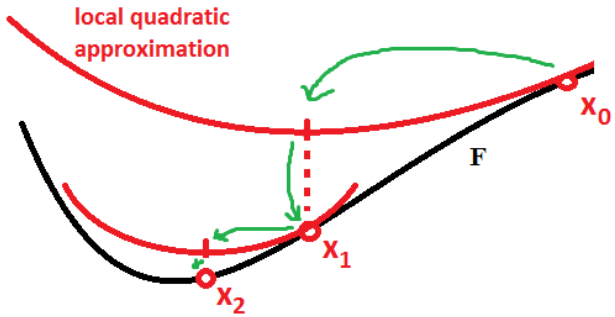
Dos enfoques equivalentes, suponiendo que la matriz hessiana es definida positiva:

- **Método de Newton** para la resolución del sistema no lineal $\boldsymbol{g}(\boldsymbol{u}) = \boldsymbol{0}$: conduce al sistema iterativo
$$\boldsymbol{H}^k(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k) = -\boldsymbol{g}^k,$$
donde se ha tenido en cuenta que el jacobiano de $\boldsymbol{g}$ es la matriz hessiana.
- **Programación cuadrática secuencial**: se basa en la aproximación cuadrática
$$F(\boldsymbol{u}) \simeq F(\boldsymbol{u}^k) + \boldsymbol{g}^{k\top}(\boldsymbol{u} - \boldsymbol{u}^k) + \frac{1}{2}(\boldsymbol{u} - \boldsymbol{u}^k)^\top \boldsymbol{H}^k(\boldsymbol{u} - \boldsymbol{u}^k),$$
cuyo mínimo viene dado por la misma ecuación que proporciona el método de Newton.
- El método de Newton es un **método convergente, de segundo orden**, si la aproximación inicial es suficientemente buena.
- Para condiciones iniciales arbitrarias, el método puede hacerse convergente (a un mínimo local), combinándolo con descenso (se verá más adelante) o con regiones de confianza.

MÉTODO DE NEWTON (II)

Tanto el método de Newton como la programación cuadrática secuencial conducen al mismo sistema de ecuaciones y se basan en la misma idea: en cada interación, se sustituye la función objetivo por su aproximación cuadrática, obtenida mediante el desarrollo de Taylor de segundo orden.





RESOLUCIÓN DEL SISTEMA LINEAL

El sistema lineal $\boldsymbol{H}^k(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k) = -\boldsymbol{g}^k$ puede resolverse utilizando uno o varios de los siguientes métodos generales:

- Mediante **eliminación gaussiana** (descomposición LU) o **factorización Cholesky** de la matriz hessiana.
- **Gradiente conjugado** (considerado más adelante).
- **Precondicionando la matriz hessiana** para hacerla mejor condicionada. Se precondiciona de dos modos:
  - Efectuando el cambio de variable $\boldsymbol{u} = \boldsymbol{P}\boldsymbol{v}$ en el sistema lineal, que se reescribe como $\boldsymbol{H}^k\boldsymbol{P}(\boldsymbol{v}^{k+1} - \boldsymbol{v}^k) = -\boldsymbol{g}^k$. Obviamente, la mejor elección (que conviene tener en cuenta) sería $\boldsymbol{P} = (\boldsymbol{H}^k)^{-1}$, pero por eficiencia computacional conviene que $\boldsymbol{P}$ sea constante en las iteraciones.
  - Multiplicando ambos miembros del sistema lineal por una matriz no singular $\boldsymbol{Q}$, para obtener $\boldsymbol{Q}\boldsymbol{H}^k(\boldsymbol{u}^{k+1} - \boldsymbol{u}^k) = -\boldsymbol{Q}\boldsymbol{g}^k$. Nuevamente, la mejor elección sería $\boldsymbol{Q} = (\boldsymbol{H}^k)^{-1}$, pero conviene que $\boldsymbol{Q}$ sea constante.

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

REGIONES DE CONFIANZA: DEFINICIÓN

Idea: calcular $\boldsymbol{u}^k$ en regiones de confianza, $||\boldsymbol{u} - \boldsymbol{u}_0^k|| \leq \rho^k$, cuyos centros y radios se definen de modo adaptativo, de modo que se van aproximando a un mínimo local. Para ello se considera la cantidad:

$$r_k = \frac{F(\boldsymbol{u}^{k-1}) - F(\boldsymbol{u}^k)}{\tilde{F}(\boldsymbol{u}^{k-1}) - \tilde{F}(\boldsymbol{u}^k)},$$

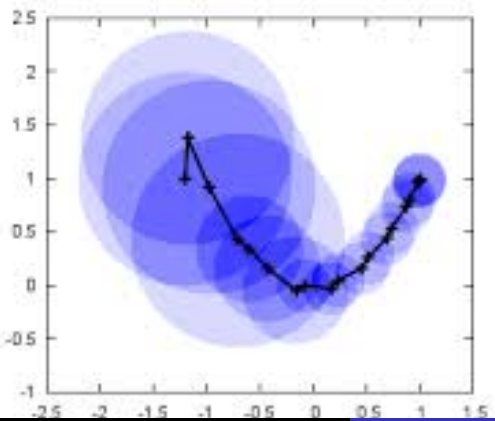
donde $\tilde{F}$ es la aproximación cuadrática de F asociada al método de Newton. Para definir la siguiente región de confianza, se consideran tres casos, según el comportamiento del método:

- **Muy bueno**: $r_k > 3/4$. Se toma $\boldsymbol{u}_0^{k+1} = \boldsymbol{u}^k$ y $\rho^{k+1} = \text{máx}\{2\rho^k, \rho^{\text{max}}\}$, donde el radio máximo admisible ρ^{max} es un parámetro calibrable.
- **Bueno**: $1/4 \leq r_k \leq 3/4$. Se toma $\boldsymbol{u}_0^{k+1} = \boldsymbol{u}^k$, $\rho^{k+1} = \rho^k$.
- **Malo**: $\eta \leq r_k < 1/4$ (con η calibrable y tal que $0 < \eta < 1/4$). Se toma $\boldsymbol{u}_0^{k+1} = \boldsymbol{u}^k$ y $\rho^{k+1} = \rho^k/4$.
- **Muy malo**: $r_k < \eta$. Se toma $\boldsymbol{u}_0^{k+1} = \boldsymbol{u}_0^k$ y $\rho^{k+1} = \rho^k/4$.

REGIONES DE CONFIANZA: CÁLCULO DE u^k

Idea: u^k podría resolverse aplicando el método de Newton (con restricciones) en la región de confianza. Para disminuír el esfuerzo computacional, en vez de ello:

- En las primeras iteraciones, u^k es la intersección de la frontera de la región de confianza con la recta que pasa por centro de la región de confianza y tiene la dirección de g^k (descenso más empinado). Tal punto se conoce como **punto de Cauchy**.
- En las últimas iteraciones, se aplica una un paso del método Newton. Si el resultado es interior a la región de confianza, tal punto se toma como u^k. En otro caso, u^k es la intersección con la frontera de la región de confianza de la recta que une el centro con el resultado de aplicar Newton. Tal punto se conoce como **punto de Newton**.



Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

VENTAJA/DESVENTAJAS DEL MÉTODO DE NEWTON

- **Principal ventaja:** es un método de segundo orden y muy robusto si se combina con regiones de confianza.
- **Principal desventaja:** cálculo problemático de la matriz hessiana. Posibles métodos:
  - Análíticamente. Imposible o muy tedioso en problemas de ingenería. En cambio, suele ser posible para funciones objetivo estrictamente cuadráticas (por ejemplo, en mínimos cuadrados), para las que, además, el método converge en un paso; para éstas, el método puede ser el indicado.
  - Simbólicamente, utilizando software específico. Problemático en problemas de ingenería.
  - Numéricamente, mediante diferencias finitas. Require N^2 evaluaciones de la función objetivo en cada paso (muy costoso computacionalmente).

ALTERNATIVAS AL MÉTODO DE NEWTON

- Desgraciadamente, el método de la secante, tal cual, no es generalizable a dimensión mayor. Para estimar el nuevo punto ($N > 1$ incógnitas) hacen falta N condiciones; la recta que pasa por las dos últimas iteraciones proporciona solamente una ecuación.
- En cambio, se han desarrollado otros métodos, también **iterativos**, de dos tipos, conocidos como **métodos quasi-Newton**:
  - **Métodos de tipo Broyden**: van estimando aproximaciones de la inversa de la matriz hessiana cada vez mejores, y calculan nuevos iterantes como en el método de Newton modificado.
  - **Métodos de descenso**: no utilizan la matriz hessiana. Van estimando direcciones de descenso cada vez mejores y resuelven problemas de optimización unidimentionales en esas direcciones.

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

MÉTODOS DE TIPO BROYDEN (I)

- En los **métodos de tipo Broyden**, se procede como en el método de Newton, pero sustituyendo el cálculo exacto de la matriz hessiana $\boldsymbol{H}$ en cada paso de la iteración (o, mejor, de su inversa, $\boldsymbol{B} \simeq \boldsymbol{H}^{-1}$, para evitar invertir $\boldsymbol{H}$ en cada paso) por un cálculo iterativo aproximado.
- La aproximación de $\boldsymbol{H}^{-1}$ en el paso proporciona $\boldsymbol{u}^{k+1} = \boldsymbol{u}^k - \boldsymbol{B}^k \boldsymbol{g}^k$ (aquí se está resolviendo el mismo problema lineal que en el método de Newton).
- La approximación $\boldsymbol{B}^k \simeq (\boldsymbol{H}^k)^{-1}$ se calcula a partir de $\boldsymbol{u}^k$, $\boldsymbol{u}^{k-1}$, $\boldsymbol{g}^k$ y $\boldsymbol{g}^{k-1}$.
- Esto puede hacerse de varios modos. Pero todos ellos utilizan la siguiente relación entre el vector gradiente y la matriz hessiana (teniendo en cuenta que la segunda es la matriz jacobiana del primero):

$$\boldsymbol{H}^{k-1} \Delta \boldsymbol{u}^k = \Delta \boldsymbol{g}^k, \text{ con } \Delta \boldsymbol{u}^k = \boldsymbol{u}^k - \boldsymbol{u}^{k-1}, \Delta \boldsymbol{g}^k = \boldsymbol{g}(u^k) - \boldsymbol{g}(\boldsymbol{u}^{k-1}).$$

- Esta relación se impone siempre a las aproximaciones de la matriz hessiana que se buscan. Pero se trata solamente de una relación vectorial, que es insuficiente para estimar una matriz.
- Hay que añadir, por tanto, **condiciones adicionales** para calcular la aproximación $\boldsymbol{H}^k$. Tales condiciones determinan los distintos métodos de Broyden.

MÉTODOS DE TIPO BROYDEN (II)

- En el método de Broyden básico, debido a Davidson, Fletcher y Powell (1964), conocido como **método DFP**, de entre las matrices que verifican la condición mencionada en la transparencia anterior, se elige la **matriz simétrica** que minimiza la norma $||\boldsymbol{H}^k - \boldsymbol{H}^{k-1}||_{\text{Fro}}$, donde $||\cdot||_{\text{Fro}}$ es la **norma de Frobenius** (raíz cuadrada de la suma de los cuadrados de los elementos de la matriz).
- El problema de minimización anterior es estrictamente cuadrático, con restricciones lineales. Se resulve analíticamente mediante multiplicadores de Lagrange (siguiente tema).
- El método más popular (y eficaz) en la actualidad se debe a Broyden, Fletcher, Goldfarb y Shanno (1970), y se conoce como **método BFGS**. En vez de minimizar $||\boldsymbol{H}^k - \boldsymbol{H}^{k-1}||_{\text{Fro}}$, se minimiza la diferencia entre sus inversas, es decir, se minimiza $||\boldsymbol{B}^k - \boldsymbol{B}^{k-1}||_{\text{Fro}}$. Por tanto, el método produce directamente aproximaciones de la inversa de la matriz hessiana.

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

Método de Davidson-Fletcher-Powell (DFP)

- En el método DFP, la aproximación de la matriz hessiana en las sucesivas iteraciones es

$$\boldsymbol{H}^k = \left(\boldsymbol{I} - \frac{\Delta \boldsymbol{g}^k (\Delta \boldsymbol{u}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}\right) \boldsymbol{H}^{k-1} \left(\boldsymbol{I} - \frac{\Delta \boldsymbol{u}^k (\Delta \boldsymbol{g}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}\right) + \frac{\Delta \boldsymbol{g}^k (\Delta \boldsymbol{g}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}.$$

- O, también, teniendo en cuenta que $\boldsymbol{H}^{k-1} \Delta \boldsymbol{u}^k = \Delta \boldsymbol{g}^k$ y $\Delta \boldsymbol{u}^k = \boldsymbol{B}^{k-1} \Delta \boldsymbol{g}^k$, se obtiene la siguiente fórmula para su inversa, $\boldsymbol{B}^k = (\boldsymbol{H}^k)^{-1}$, que es la que realmente interesa:

$$\boldsymbol{B}^k = \boldsymbol{B}^{k-1} - \frac{\boldsymbol{B}^{k-1} \Delta \boldsymbol{g}^k (\Delta \boldsymbol{g}^k)^\top \boldsymbol{B}^{k-1}}{\Delta \boldsymbol{g}^k \boldsymbol{B}^{k-1} (\Delta \boldsymbol{g}^k)^\top} + \frac{\Delta \boldsymbol{u}^k (\Delta \boldsymbol{u}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}.$$

- Como condición inicial, se toma la matriz unidad, es decir

$$\boldsymbol{H}^0 = (\boldsymbol{H}^0)^{-1} = \boldsymbol{I}.$$

MÉTODO DE BROYDEN-FLETCHER-GOLDFARB-SHANNO (BFGS)

- En el método BFGS, aproximación de la inversa de la matriz hessiana en las sucesivas interaciones es

$$\boldsymbol{B}^k = \left(\boldsymbol{I} - \frac{\Delta \boldsymbol{g}^k (\Delta \boldsymbol{u}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}\right) \boldsymbol{B}^{k-1} \left(\boldsymbol{I} - \frac{\Delta \boldsymbol{u}^k (\Delta \boldsymbol{g}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}\right) + \frac{\Delta \boldsymbol{u}^k (\Delta \boldsymbol{u}^k)^\top}{(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k}.$$

- Notese que, como era de esperar, esta fórmula coincide con la que proporciona $\boldsymbol{H}^k$ en el método DFP, pero cambiando en todas partes $\boldsymbol{H}$ por $\boldsymbol{B}$ e itercambiando los papeles de $\Delta \boldsymbol{g}$ y $\Delta \boldsymbol{u}$.
- Y la propia matriz hessiana se podría calcular como

$$\boldsymbol{H}^k = \boldsymbol{H}^{k-1} - \frac{\boldsymbol{H}^{k-1} \Delta \boldsymbol{u}^k (\Delta \boldsymbol{u}^k)^\top \boldsymbol{H}^{k-1}}{\Delta \boldsymbol{u}^k \boldsymbol{H}^{k-1} (\Delta \boldsymbol{u}^k)^\top} + \frac{\Delta \boldsymbol{g}^k (\Delta \boldsymbol{g}^k)^\top}{(\Delta \boldsymbol{u}^k)^\top \Delta \boldsymbol{g}^k}.$$

- Nuevamente, la condición inicial es la matriz unidad, es decir

$$\boldsymbol{H}^0 = (\boldsymbol{H}^0)^{-1} = \boldsymbol{I}.$$

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

MEJORAS PARA GRANDES SISTEMAS

En principio, las matrices $[H]$ y $[H]^{-1}$ son matrices llenas de tamaño $N \times N$, de modo que si $N \gg 1$ el coste computacional y la memoria necesaria para almacenarlas pueden ser muy grandes. Para evitarlo, se han desarrollado dos familias de métodos:

- **BFGS de baja memoria** (low-memory BFGS o L-BFGS). Tiene en cuenta que las matrices $\Delta \boldsymbol{u}^k (\Delta \boldsymbol{g}^k)^\top$, $\Delta \boldsymbol{g}^k (\Delta \boldsymbol{u}^k)^\top$ $\Delta \boldsymbol{u}^k (\Delta \boldsymbol{u}^k)^\top$ que aparecen en la fórmula BFGS son matrices de rango uno que provienen del producto de vectores. En vez de calcular explícitamente (y almacenar) esas matrices, el cálculo iterativo de la fórmula BFGS puede factorizarse de modo que $\boldsymbol{B}^k$ se escribe como el producto de tres matrices, cuyos tamaños son $N \times n_k$, $n_k \times n_k$ y $n_k \times N$, donde $n_k \ll N$ crece de uno en uno a lo largo de las iteraciones, comenzando con el valor $n_1 = 1$.
- **Conservación de patrones dispersos**. Ocurre a veces que la matriz hessiana o su inversa es una matriz dispersa. En ese caso, se impone a la approximación Broiden que converve la misma estructura dispersa, imponiendo tal condición lineal a la fórmula (del mismo modo en que se impuso que fuese simétrica).

Outline

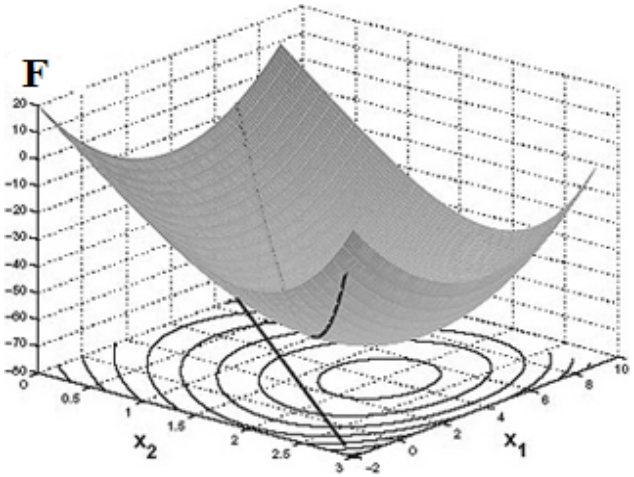
1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

MÉTODOS DE DESCENSO (I)

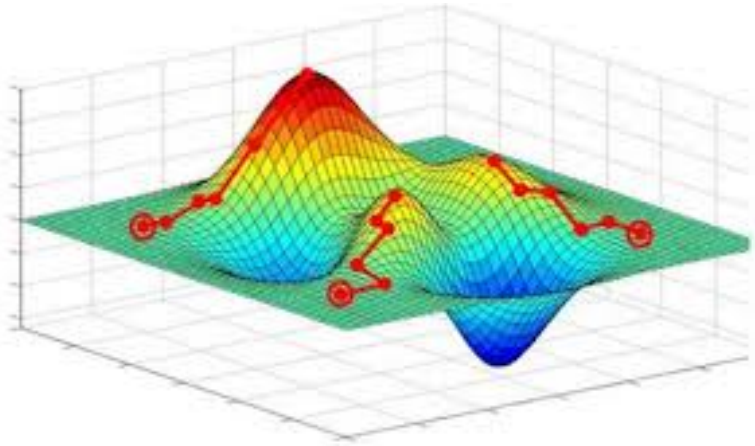
- Los métodos de descenso son métodos iterativos, en los que cada iteración consiste en dos pasos:
  - Seleccionar una **dirección de descenso**, $\boldsymbol{d}^{k+1}$. Los distintos métodos de descenso difieren justamente en cómo se hace esta selección.
  - **Minimizar la función objetivo a lo largo de esa dirección**, es decir, a lo largo de la recta $\boldsymbol{u} = \boldsymbol{u}^k + \alpha \boldsymbol{d}^{k+1}$, donde $\boldsymbol{u}^k$ es la aproximación de $\boldsymbol{u}$ en el paso anterior.
- La **minimización unidimensional** (cálculo de α) puede hacerse, por ejemplo, mediante un método de Newton unidimensional o, mejor, mediante el **método de la secante**.

MÉTODOS DE DESCENSO (II)

Sin embargo, efecturar las minimizaciones unidimensionales con precisión conlleva un esfuerzo computacional inútil en las primeras iteraciones. En vez de ello, solamente se asegura que **la función objetivo disminuye apropiadamente** en esas iteraciones.



Minimización en cada paso



Método iterativo (dibujando $-F$)

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

DESCENSO MÁS EMPINADO (STEEPEST DESCENT)

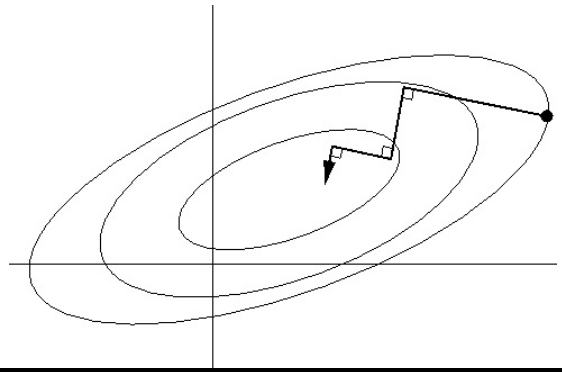
- El **método de descenso más empinado** responde a la idea intuitiva de seleccionar como dirección de descenso aquella en la que la función objetivo presenta, localmente, un descenso más pronunciado.
- Esa dirección es la del gradiente de la función. Por tanto, en cada paso, la dirección de descenso se toma como

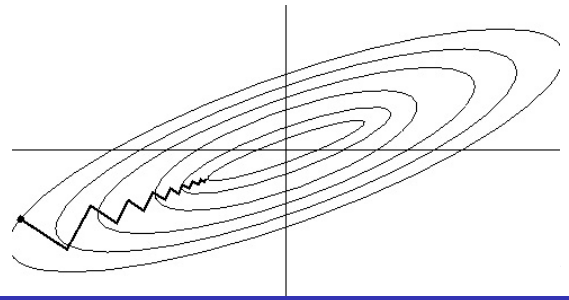
$$\boldsymbol{d}^{k+1} = -\boldsymbol{g}^{k}.$$

- A pesar de lo intuitivo que es, este método no es óptimo y, de hecho, converge muy mal cuando la matriz hessiana está mal condicionada, es decir, cuando tiene autovalores muy dispares.

EJEMPLO EN DIMENSIÓN DOS

- En dimensión $N = 2$, las líneas de nivel de la función objetivo son, aproximadamente, una familia autosemejante de elipses concéntricas, cuyo centro es precisamente el mínimo que se busca.
- Si la excentricidad es distinta de uno, la trayectoria óptima (la que pasa por el centro) y la de máximo descenso, son distintas.
- Cada dirección de descenso es perpendicular a la linea de nivel inicial y tangente a la final.
- Las trayectorias son de tipo zig-zag, tanto más pronunciado cuanto mayor es la excentricidad.
- La dificultad persiste en dimensión $N \geq 2$, y es tanto mayor cuanto peor condicionada esté la matriz hessiana; el mal condicionamiento de la matriz hessiana puede deberse a un escalado inapropiado de las variables de diseño.





USO DE DIRECCIONES CONJUGADAS: MÉTODO

La dirección de descenso óptima se calcula mediante direcciones conjugadas. Para minimizar la aproximación cuadrática

$$F(\boldsymbol{u}) - F(\boldsymbol{u}^k) = (\boldsymbol{u} - \boldsymbol{u}^k)^\top \boldsymbol{H}^k (\boldsymbol{u} - \boldsymbol{u}^k)/2 + \boldsymbol{g}^{k\top} (\boldsymbol{u} - \boldsymbol{u}^k)$$

- Se calculan (por ejemplo, mediante el método de Gramm-Schmidt) N direcciones conjugadas, tales que $\boldsymbol{d}_i^\top \boldsymbol{H}^k \boldsymbol{d}_j = 0$ si $i \neq j$ y $\boldsymbol{d}_i^\top \boldsymbol{H}^k \boldsymbol{d}_i = 1$.
- Si se hace

  $$\boldsymbol{u} - \boldsymbol{u}^k = \alpha_1 \boldsymbol{d}_1 + \ldots + \alpha_N \boldsymbol{d}_N,$$

  se tiene

  $$F(\boldsymbol{u}) - F(\boldsymbol{u}^k) = (\boldsymbol{u} - \boldsymbol{u}^k)^\top \boldsymbol{H}^k (\boldsymbol{u} - \boldsymbol{u}^k)/2 + \boldsymbol{g}^{k\top} (\boldsymbol{u} - \boldsymbol{u}^k) \equiv$$

  $$(\alpha_1^2 + \ldots + \alpha_N^2)/2 + \boldsymbol{g}^{k\top} (\alpha_1 \boldsymbol{d}_1 + \ldots + \alpha_N \boldsymbol{d}_N).$$

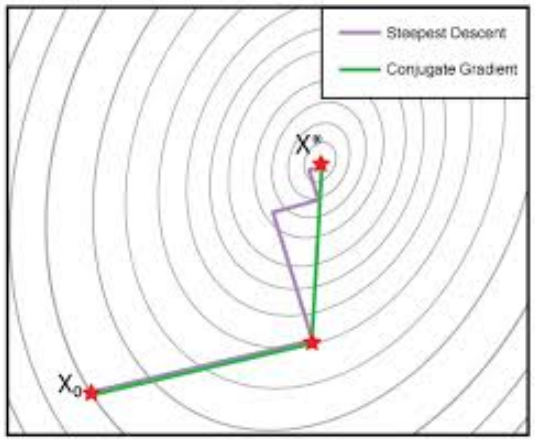
- Obviamente, el mínimo viene dado por $\alpha_i = -\boldsymbol{g}^{k\top} \boldsymbol{d}_i$ para $i = 1, \ldots, N$.

USO DE DIRECCIONES CONJUGADAS: INTERPRETACIÓN

- El método de la transparencia anterior puede verse como:
  - Un modo de diagonalizar la matriz hessiana, facilitando la resolución del sistema lineal $\boldsymbol{H}^k \boldsymbol{u}^{k+1} = -\boldsymbol{g}^k$ en el método de Newton.
  - Un modo de seleccionar N direcciones de descenso eficaces (las direcciones conjugadas).
- **Existen infinitos conjuntos de direcciones conjugadas**.
- **Si la matriz hessiana es constante, el método resultante converge en $N-1$ pasos** (necesarios para calcular los N vectores conjugados).
- Si, en cambio, la matriz hessiana no es constante, los $N-1$ pasos deben completarse en cada paso de la iteración del método de Newton.
- Para mejorar la eficacia del método, conviene **mezclar ambas iteraciones**, la del método de Newton y la necesaria para calcular las direcciones conjugadas.
- Pero **la mezcla de las dos iteraciones no es obvia**: si se actualiza la matriz hessiana en cada paso, se pierde la conjugación (exacta) con las direcciones anteriores.

GRADIENTE CONJUGADO (CONJUGATE GRADIENT): IDEA

- Los **métodos de gradiente conjugado** son métodos de descenso en que las sucesivas direcciones de descenso son **asintóticamente conjugadas** (es decir, aproximadamente conjugadas en las cercanías del punto óptimo) entre sí.
- El número de interaciones necesarias es del orden de N.
- Como existen infinitos conjuntos direcciones conjugadas, éstas pueden definirse imponiendo condiciones adicionales (distintas fórmulas de gradiente conjugado).
- Para mejorar la eficacia computacional del método, se busca que cada nueva dirección conjugada pueda calcularse **sin utilizar explícitamente la matriz hessiana ni todas las direcciones conjugadas anteriores**.



GRADIENTE CONJUGADO: FÓRMULAS

Las distintas **fórmulas de gradiente conjugado** difieren entre sí en la expresión de la nueva dirección conjugada, $\boldsymbol{d}^{k+1}$, en términos de la anterior, $\boldsymbol{d}^k$, y del gradiente en el paso anterior, $\boldsymbol{g}^k$. Pueden agruparse en la forma

$$\boldsymbol{d}^{k+1} = -\boldsymbol{g}^k + \beta^k \boldsymbol{d}^k, \quad \boldsymbol{d}^1 = -\boldsymbol{g}^0,$$

donde el coeficiente β_k suele depender de los dos pasos anteriores, de acuerdo con distintas fórmulas. En todos los casos, las dos primeras iteraciones se efectúan mediante el método del descenso más empinado.

- La fórmula más básica se debe a **Hestenes y Stiefel** (1959):

$$\beta^k = (\boldsymbol{g}^k)^\top (\boldsymbol{g}^k - \boldsymbol{g}^{k-1}) / [(\boldsymbol{g}^{k-1})^\top (\boldsymbol{g}^k - \boldsymbol{g}^{k-1})].$$

- Una de las más populares es la de **Fretcher-Reeves** (1964):

$$\beta^k = |\boldsymbol{g}^k|^2 / |\boldsymbol{g}^{k-1}|^2.$$

- La de **Hager-Zang** (2005) es apropiada para funciones objetivo fuertemente no lineales

$$\beta^k = \left(\boldsymbol{g}^k - \boldsymbol{g}^{k-1} - 2\boldsymbol{d}^k \frac{||\boldsymbol{g}^k - \boldsymbol{g}^{k-1}||^2}{(\boldsymbol{d}^k)^\top (\boldsymbol{g}^k - \boldsymbol{g}^{k-1}} \right)^\top \frac{\boldsymbol{g}^k}{(\boldsymbol{d}^k)^\top (\boldsymbol{g}^k - \boldsymbol{g}^{k-1})}$$

Outline

1. Preliminares
   - Teoría básica
   - Optimización en 1D
2. Método de Newton en dimensión mayor
   - El método
   - Regiones de confianza
   - Discusión del método de Newton
3. Métodos de tipo Broyden
   - Idea general
   - Métodos DFP y BFGS
   - Manejo eficiente de la memoria
4. Métodos de descenso
   - Idea general
   - Descenso más empinado y gradiente conjugado
   - Variantes y selección del paso
5. Aplicaciones con MatLab
6. Cálculo del vector gradiente y método del adjunto

GRADIENTE CONJUGADO: VARIANTES

- **Reducciones 2D**: β^k puede calcularse minizando la F, o una aproximación cuadrática suya (mediante la hessiana o la hessiana approximada), en el plano generado por $\boldsymbol{g}^k$ y $\boldsymbol{d}^k$.
- **Gradiente conjugado precondicionado** (pre-conditioned conjugate gradient). Se efectúa el cambio de variable $\boldsymbol{u} = \boldsymbol{P}\boldsymbol{v}$ (antes de aplicar el método), que transforma $\boldsymbol{d}$ y $\boldsymbol{g}$ como $\boldsymbol{d} \to \boldsymbol{P}^{-1}\boldsymbol{d}$ y $\boldsymbol{g} \to \boldsymbol{P}^\top \boldsymbol{g}$. Las fórmulas $\boldsymbol{d}^{k+1} = -\boldsymbol{g}^k + \beta^k \boldsymbol{d}^k$ y $\boldsymbol{d}^1 = -\boldsymbol{g}^0$ se transforman en $\boldsymbol{d}^{k+1} = -\boldsymbol{g}^k \boldsymbol{q} + \hat{\beta}_k \boldsymbol{d}^k$ y $\boldsymbol{d}^1 = -\boldsymbol{Q}\boldsymbol{g}^0$, respectivamente, con $\boldsymbol{Q} = \boldsymbol{P}\boldsymbol{P}^\top$ = simétrica y definida positiva. El nuevo parámetro $\hat{\beta}^k$ se calcula efectuando la transformación anterior en las distintas fórmulas. Por ejemplo, la de Fretcher-Reeves se transforma en $\hat{\beta}^k = (\boldsymbol{g}^k)^\top \boldsymbol{Q}\boldsymbol{g}^k / (\boldsymbol{g}^{k-1})^\top \boldsymbol{Q}\boldsymbol{g}^{k-1}$.
- Una posible selección de la matriz $\boldsymbol{Q}$ es la inversa de la matriz Hessiana (o su approximación mediante BFGS), si se dispone de ellas.

SELECCIÓN DEL PASO

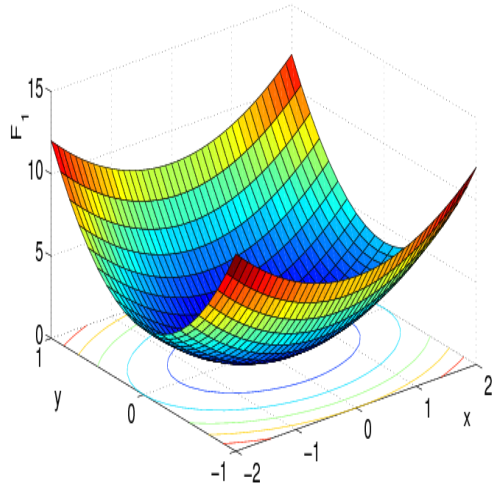
Optimizar con precisión la función objetivo en los primeros pasos de los métodos de descenso puede requerir un esfuerzo computacional inútil. En vez de ello, solamente se impone un descenso adecuado de la función objetivo, imponiendo:

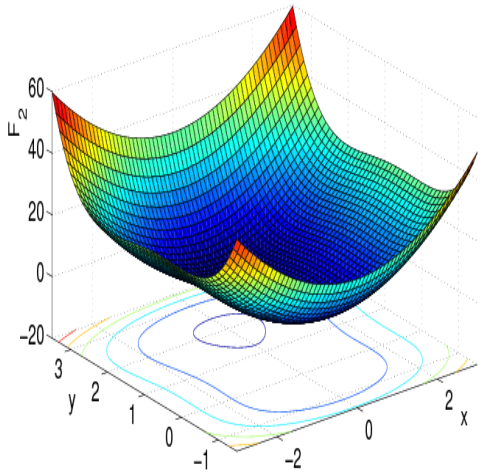
- **Primera condición de curvatura** (para imponer que F sea convexa en la dirección de descenso): como $\boldsymbol{H}^k$ es definida positiva, debe ser
$$(\Delta \boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k > 0.$$
- **Primera condición de Wolfe** (también llamada **condición de Armijo**), que limita superiormente el paso $\Delta \boldsymbol{u}^k = \boldsymbol{u}^{k+1} - \boldsymbol{u}^k$:
$$F(\boldsymbol{u}^{k+1}) - F(\boldsymbol{u}^k) \leq c_1 (\boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k.$$
- **Segunda condición de Wolfe** (también llamada segunda condición de curvatura), que limita inferiormente el paso:
$$(\boldsymbol{g}^{k+1})^\top \Delta \boldsymbol{u}^k \geq c_2 (\boldsymbol{g}^k)^\top \Delta \boldsymbol{u}^k.$$
- Los coeficientes ajustables c_1 y c_2 deben verificar $0 < c_1 < c_2 < 1$. Por defecto, suele tomarse c_1 muy pequeño, y c_2 próximo a 1 en descenso basado en Newton y quasi-Newton; en gradiente conjugado, c_2 suele tomarse más pequeño.

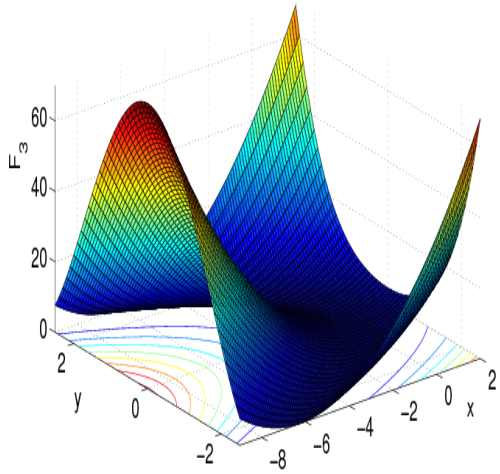
APLICACIONES EN 2D: FUNCIONES 'CONVEXA', 'BIMODAL' Y 'BANANA'

En lo que sigue, se consideran las siguientes funciones, definidas con anterioridad:

$$F_1 = x^2 + 8y^2, \quad F_2 = 3x^2 + (1+y^2)^2 - 10y^3/3, \quad F_3 = (x+y^2)^2 + (1+y)^2/100.$$







OPTIMIZACIÓN MEDIANTE *fminunc* DE MATLAB: algoritmos

fminunc usa dos familias de algoritmos:

1. 'Trust region': combina el método de Newton con regiones de confianza. Debe proporcionarse el gradiente. La matriz hessiana puede, o bien proporcionarse (opcion 'Derivatives: gradient and hessian supplied') o calcularse mediante diferencias finitas (opcion 'Derivatives: approximated by the solver'). El problema lineal en cada iteración ('subproblem algorithm') o bien se resuelve mediante factorización Cholesky ('Cholesky factorization') o mediante gradiente conjugado precondicionado ('preconditioned congugate gradient'), que es la opción por defecto.
2. 'Quasi-Newton': utiliza descenso basado en las condiciones de Wolfe. El vector gradiente puede proporcionarse (opcion 'Derivatives: gradient supplied')) o calcularse mediante diferencias finitas (opcion 'Derivatives: approximated by the solver'). La dirección de descenso puede calcularse mediante varias opciones, especificadas en 'Hessian update': utilizando los métodos BFGS o DFP, o mediane descenso más empiando.

Se dan valores por defecto para el número máximo de iteraciones, el número máximo de evaluaciones de la función objetivo y las tolerancias máximas en las variables de diseño y en la función objetivo. Alternativamente, esos parámetros del algoritmo pueden definirse.

OPTIMIZACIÓN MEDIANTE *fminunc* DE MATLAB: implementación

Es necesario el punto inicial, como '$[u_1^0\ u_2^0\ \ldots\ u_N^0]$', y la función objetivo, como '@fun', donde la función 'fun' debe estar definida, (a la vez que su gradiente y su matriz hessiana, si estos son necesarios) mediante un archivo 'fun.m' en el mismo directorio desde el que se está ejecutando la aplicación. Por ejemplo, la función banana2D si introduce como '@banana', y se define como:

function [f,g,h] = banana(x)
$f = (x(1) + x(2)^2)^2 + (1 + x(2))^2/100;$
if nargout>1
$g(1) = 2 * (x(1) + x(2)^2);$
$g(2) = 4 * x(2) * (x(1) + x(2)^2) + (1 + x(2))/50;$
$h(1,1) = 2;$
$h(1,2) = 4 * x(2);$
$h(2,1) = h(1,2);$
$h(2,2) = 4 * x(1) + 12 * x(2)^2 + 1/50;$
end

OPTIMIZACIÓN MEDIANTE *fminunc* DE MATLAB: aplicación

En lo que sigue, se dan varias tablas en las que se proporiciona utilizan las opciones de MatLab por defecto. En cada tabla:

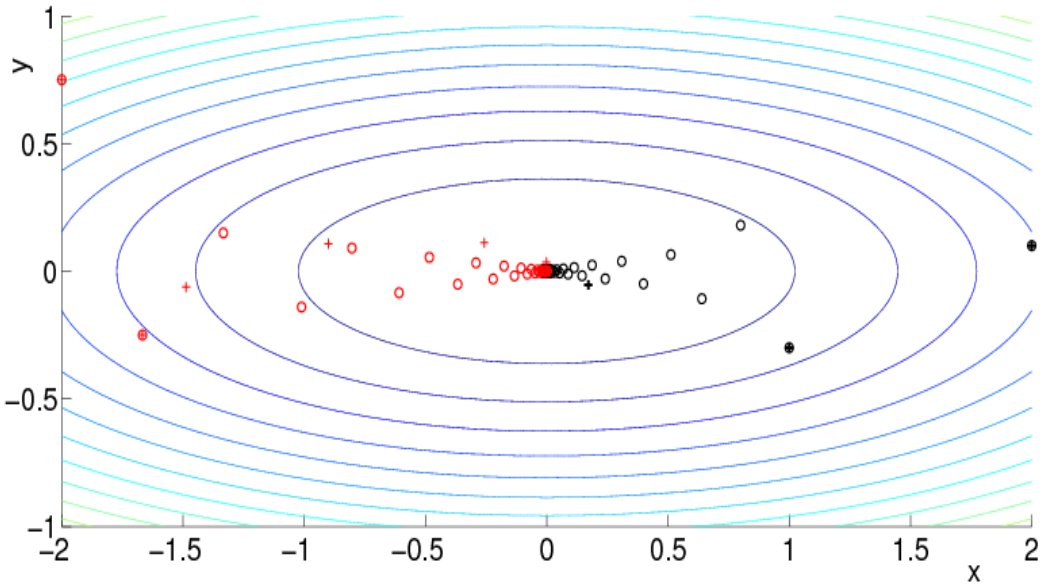
- Se proporicona el **número de iteraciones y de evaluaciones** de la función objetivo, así como el **error** en el cálculo del punto donde se alcanza el mínimo, medido como la máxima de la diferencia entre coordenadas: error=$\max_i |u_i^{\text{exacto}} - u_i^{\text{aprox.}}|$.
- Se aplican **8 métodos** (notación para las tablas, entre paréntesis): 'Trust region' dando $\boldsymbol{g}$ y $\boldsymbol{H}$ (TR+G+H) y dando sólo $\boldsymbol{g}$ (TR+G). 'Quasi-Newton', con seis opciones: 'BFGS' dando $\boldsymbol{g}$ (BFGS+G) o sin dar $\boldsymbol{g}$ (BFGS); 'DFP' dando $\boldsymbol{g}$ (DFP+G) o sin dar $\boldsymbol{g}$ (DFP); steepest descend' dando $\boldsymbol{g}$ (SD+G) o sin dar $\boldsymbol{g}$ (SD).
- Para cada caso, se utilizan tres puntos iniciales representativos, que se indican.

APLICACIÓN A LA FUNCIÓN CONVEXA: RESULTADOS

Initial point	(2,0)			(2,0.1)			(-2,1.5)		
Algorithm	iter.	eval.	error	iter.	eval.	error	iter.	eval.	error
TR+G+H	1	-	10^{-16}	1	-	$2 \cdot 10^{-16}$	1	-	$6 \cdot 10^{-16}$
TR+G	1	-	10^{-16}	1	-	$2 \cdot 10^{-16}$	1	-	$6 \cdot 10^{-16}$
BFGS+G	2	3	10^{-16}	3	4	$4 \cdot 10^{-7}$	9	10	$9 \cdot 10^{-8}$
BFGS	3	15	$6 \cdot 10^{-9}$	3	12	$4 \cdot 10^{-7}$	9	30	$8 \cdot 10^{-8}$
DFP+G	2	3	10^{-16}	5	6	$4 \cdot 10^{-12}$	13	14	$8 \cdot 10^{-7}$
DFP	3	15	$6 \cdot 10^{-9}$	5	18	$7 \cdot 10^{-9}$	13	42	$7 \cdot 10^{-7}$
SD+G	2	4	10^{-6}	52	104	$3 \cdot 10^{-6}$	49	98	10^{-5}
SD	2	12	$3 \cdot 10^{-7}$	33	200	$4 \cdot 10^{-4}$	33	200	$6 \cdot 10^{-4}$

- **Condiciones iniciales**: (2,0) (en el eje de simetría), (2,0.1) (muy próximo al eje de simetría) y (-2,1.5) (genérico).
- Mejores prestaciones: en el orden que se dan.
- Por ser F_1 estrictamente cuadrática, el método de Newton converge en un paso siempre. BFGS y DFP se comportan de modo no muy distinto.
- El descenso más empinado se comporta mucho peor y de modo errático. Muy buen comportamiento para el primer punto inicial, pero muy malo para el segundo, que está próximo.

FUNCIÓN CONVEXA: EVOLUCIÓN DE LAS ITERACIONES



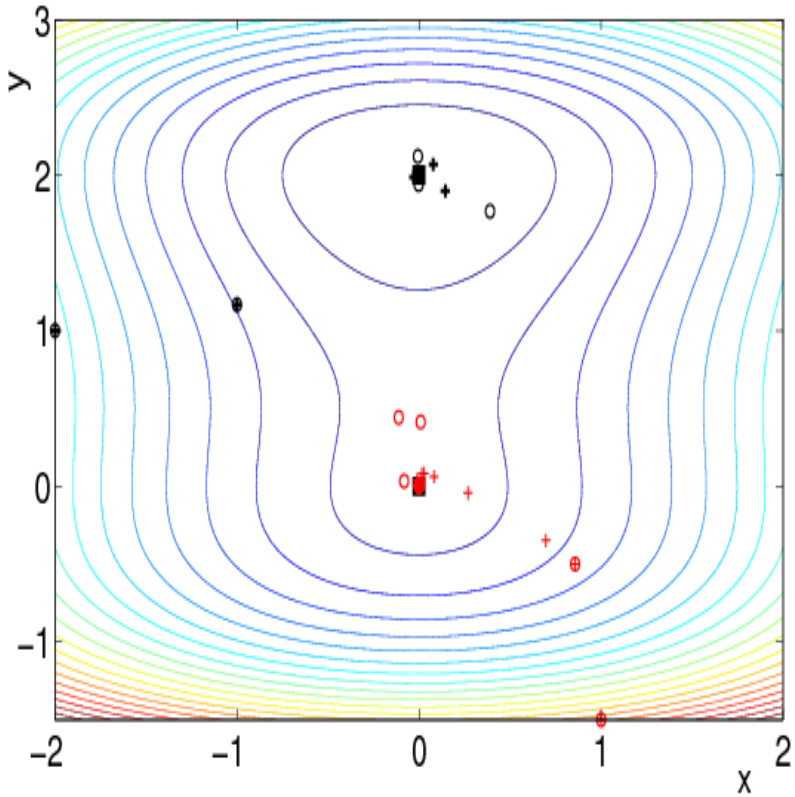
- Iteraciones utilizando BFGS+G (cruces) y SD+G (círculos), con condición inicial en $(2,0{,}1)$ (negro) y $(-2,1{,}5)$ (rojo).
- Como referencia, se representan algunas curvas de nivel.
- Nótese el comportamiento zigzagueante de SD+G.

APLICACIÓN A LA FUNCIÓN BIMODAL

Pto inicial	(0,3)			(-2,1)			(1,-1.5)		
Algoritmo	iter.	eval.	error	iter.	eval.	error	iter.	eval.	error
TR+G+H	5	-	$6{\cdot}10^{-8}$	8	-	$6\cdot10^{-12}$	7	-	$5{\cdot}10^{-10}$
TR+G	5	-	$6{\cdot}10^{-8}$	8	-	$6{\cdot}10^{-12}$	7	-	$5{\cdot}10^{-10}$
BFGS+G	1	2	10^{-16}	7	8	$2{\cdot}10^{-7}$	11	12	$2{\cdot}10^{-7}$
BFGS	1	2	10^{-16}	7	24	$2{\cdot}10^{-7}$	11	36	$2{\cdot}10^{-7}$
DFP+G	1	2	10^{-16}	7	8	$1.2{\cdot}10^{-6}$	19	20	$7{\cdot}10^{-6}$
DFP	1	6	10^{-16}	7	24	$1.2{\cdot}10^{-6}$	19	60	$7{\cdot}10^{-6}$
SD+G	1	2	10^{-16}	12	24	$3{\cdot}10^{-7}$	9	20	10^{-6}
SD	1	6	10^{-16}	12	72	$3{\cdot}10^{-7}$	9	60	10^{-6}

- **Condiciones iniciales**: (0,3) (en el eje de simetría), (-2,1) (genérica en el dominio de atracción del mínimo global) y (1,-1.5) (genérica en dominio de atracción del segundo mínimo).
- Mejores prestaciones: en el orden que se dan, nuevamente, pero las prestaciones son parecidas porque las líneas de nivel tienen excentricidad pequeña.

FUNCIÓN BIMODAL: EVOLUCIÓN DE LAS ITERACIONES



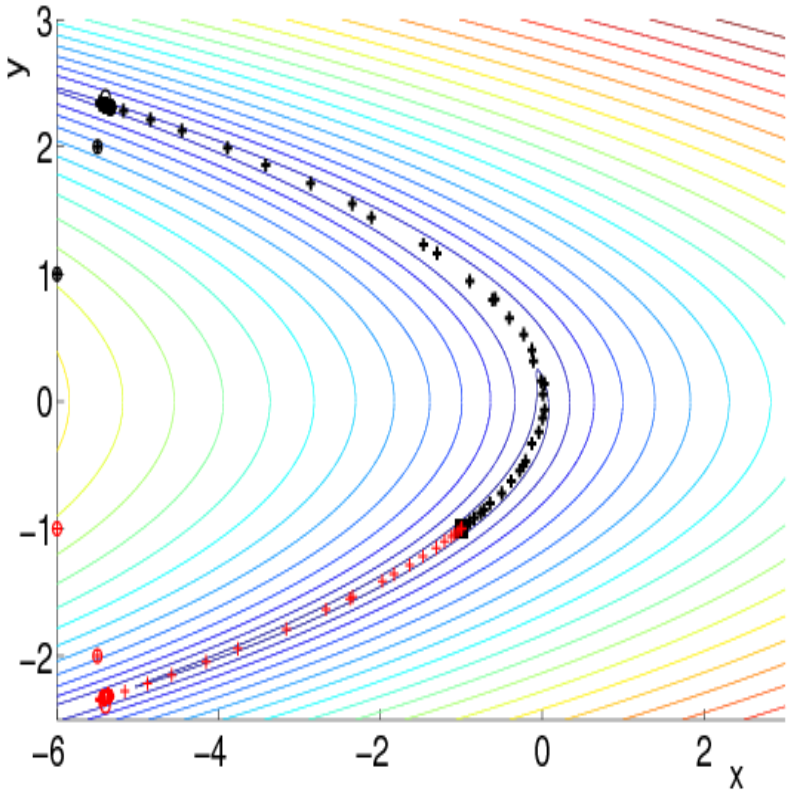
Iteraciones utilizando BFGS+G (cruces) y SD+G (círculos), con condición inicial en $(-2, 1)$ (negro) y $(1, -2)$ (rojo).

OPTIMIZACIÓN 2D MEDIANTE *minumc* DE MATLAB: 'BANANA'

Punto inic.	(-6,1)			(-6,-1)			(-1.1,-1.1)		
Algoritmo	iter.	eval.	error	iter.	eval.	error	iter.	eval.	error
TR+G+H	35	-	$2{\cdot}10^{-5}$	26	-	$2{\cdot}10^{-6}$	4	-	$5{\cdot}10^{-8}$
TR+G	35	-	$2{\cdot}10^{-5}$	26	-	$2{\cdot}10^{-6}$	4	-	$5{\cdot}10^{-8}$
BFGS+G	45	59	$2{\cdot}10^{-4}$	30	36	$2{\cdot}10^{-6}$	12	16	$8{\cdot}10^{-8}$
BFGS	46	186	$3{\cdot}10^{-5}$	30	108	$7{\cdot}10^{-6}$	12	48	$9{\cdot}10^{-6}$
DFP+G	81	110	$2{\cdot}10^{-6}$	178	200	1.8	188	200	0.085
DFP	60	200	3.8	53	200	2.7	60	200	0.026
SD+G	80	200	4.3	80	200	4.4	99	200	0.11
SD	26	200	4.3	27	200	4.4	33	200	0.11

- **Condiciones iniciales**: (-6,1) y (-6,-1) (genéricas, alejadas del mínimo) y (-1.1,-1.1) (cercana al mínimo).
- Mejores prestaciones: en el orden que se dan.
- BFGS+G y BFGS funcionan bien siempre; DFP+G no.
- DFP, SD+G y SD no funcionan bien nunca.

FUNCIÓN BANANA: EVOLUCIÓN DE LAS ITERACIONES



Iteraciones utilizando BFGS+G (cruces) y SD+G (círculos), con condición inicial en $(-6, 1)$ (negro) y $(-6, -1)$ (rojo).

APLICACIONES EN DIMENSIÓN MAYOR QUE 2

Cualquier función 2D puede extenderse de modo genérico a un número mayor de dimensiones.

- Funciones **bimodal ND** y **banana ND**:

$$f_{\mathrm{ND}}(\boldsymbol{u}) = f_{\mathrm{2D}}(u_1, u_2) + \hat{\boldsymbol{u}}^\top \boldsymbol{B}\hat{\boldsymbol{u}}/2 + (\hat{\boldsymbol{u}}^\top \boldsymbol{B}\hat{\boldsymbol{u}})^2/4,$$

  donde f_{2D} es la función (bimodal o banana) 2D, $\hat{\boldsymbol{u}} = [u_3, \ldots, u_N]^\top$ y $\boldsymbol{B}$ **simétrica y definida positiva**. Se tomará $\boldsymbol{B} = \boldsymbol{I} + \boldsymbol{A}^\top \boldsymbol{A}/(N-2)$, con $\boldsymbol{I}$ = matriz unidad de orden $N-2$ y $\boldsymbol{A}$ = matriz cuadrada de orden $N-2$, cuyos elementos son $A_{ij} = \sin\sqrt{i+2j}$.

- Los mínimos son de la forma $(u_1, u_2, 0, \ldots, 0)$, donde (u_1, u_2) son los mínimos de f_{2D}.

- Para comparar con el caso 2D: los puntos iniciales se tomarán como $(u_1^0, u_2^0, 1, \ldots, 1)$, donde (u_1^0, u_2^0) son las condiciones iniciales para el caso 2D.

EXTENSIÓN A 100D DE LA FUNCIÓN BIMODAL

Punto inic.	(0,3)			(-2,1)			(1,-1.5)		
Algoritmo	iter.	eval.	error	iter.	eval.	error	iter.	eval.	error
TR+G+H	9	-	510^{-7}	11	-	$7 \cdot 10^{-13}$	9	-	$4 \cdot 10^{-7}$
TR+G	9	-	$5 \cdot 10^{-7}$	11	-	$7 \cdot 10^{-13}$	9	-	$4 \cdot 10^{-7}$
BFGS+G	14	15	$2 \cdot 10^{-6}$	16	20	$3 \cdot 10^{-6}$	23	26	10^{-6}
BFGS	14	1515	$2 \cdot 10^{-6}$	16	2020	$3 \cdot 10^{-6}$	23	2626	10^{-6}
DFP+G	52	53	$5 \cdot 10^{-5}$	98	103	10^{-5}	400	410	1,6
DFP	52	5353	$5 \cdot 10^{-5}$	95	10000	$9 \cdot 10^{-5}$	90	10000	0,08 *
SD+G	9	19	$6 \cdot 10^{-6}$	10	21	10^{-5}	12	27	$6 \cdot 10^{-6}$ *
SD	9	1919	$6 \cdot 10^{-6}$	10	2121	10^{-5}	12	2727	$7 \cdot 10^{-6}$ *

- **Parámetros del método**: los de MatLab por defecto.
- **Condiciones iniciales**: se indican sólo las dos primeras componentes; las demás son iguales a 1.
- Resultados parecidos al caso 2D, excepto porque la condición inicial está en el dominio de atracción del mínimo global en algunos casos (indicados con asterisco) y en el dominio de atracción del mínimo local en los demás.
- Cuando las derivadas se aproximan mediante diferencias finitas, el número de evaluaciones es unas 50 veces mayor que en el caso 2D, como era de esperar.

EXTENSIÓN A 100D DE LA FUNCIÓN BANANA

Punto inic.	(0,3)			(-2,1)			(1,-1.5)		
Algoritmo	iter.	eval.	error	iter.	eval.	error	iter.	eval.	error
TR+G+H	103	-	$6\cdot10^{-4}$	99	-	$5\cdot10^{-5}$	20	-	$2.3\cdot10^{-3}$
TR+G	105	-	$4\cdot10^{-5}$	100	-	$1.2\cdot10^{-3}$	20	-	$3\cdot10^{-3}$
BFGS+G	79	81	4.4	79	81	4.4	79	81	0.15
BFGS	79	8181	4.4	79	8181	4.4	79	8181	0.15
TR+G+H	28	-	$5\cdot10^{-5}$	11	-	$6\cdot10^{-10}$	10	-	$2\cdot10^{-10}$
TR+G	28	-	$5\cdot10^{-5}$	11	-	$6\cdot10^{-10}$	10	-	$2\cdot10^{-10}$
BFGS+G	152	166	$5\cdot10^{-3}$	131	137	$5\cdot10^{-3}$	94	96	0.037
BFGS	151	16165	$2.2\cdot10^{-3}$	129	13635	0.018	94	9696	0.037
BFGS+G	155	168	10^{-5}	150	156	$3\cdot10^{-4}$	117	119	$4\cdot10^{-4}$
BFGS	153	16687	$3\cdot10^{-4}$	150	15756	10^{-5}	117	12019	$4\cdot10^{-4}$

- No se consideran DFP ni SD, que fallaban en 2D.
- Primeras cuatro columnas: con opciones MatLab por defecto.
- Columnas 5-8: reescalando las nuevas variables con un factor de 1/10, para que no enmascaren el pequeño término de la función banana 2D que da curvatura al valle.
- Últimas dos columnas: disminyendo las tolerancias por defecto a 10^{-8}.

ERRORES: EFECTO

- Además de los **errores de redondeo**, que ya se tienen en cuenta en las herramientas de optimización, los simuladores numéricos de disciplinas técnicas están afectados de **errores de discretización y truncación**.
- Si estos errores tienen un **comportamiento suave** al variar las variables de diseño, su efecto será, simplemente, contaminar la función objetivo y, por tanto, la solución.
- Si, en cambio, como sucede con frecuencia, los errores tienen un **comportamiento altamente oscilatorio**, errores de muy pequeña amplitud (mucho menor que la tolerancia del método) pueden tener un **efecto dramático** en los cálculos de derivadas mediante diferencias finitas y, por tanto en los métodos de optimización de tipo gradiente.
- La razón para tal comportamiento anómalo es que, estrictamente, la función objetivo contaminada tiene una **gran cantidad de mínimos relativos** y las métodos pueden converger a cualquiera de ellos (que la herramienta ve como verdaderos mínimos locales).

ERRORES: FILTRADO

Existen **métodos generales** (que se salen del alcance de este curso) para filtrar errores altamente oscilatorios, suavizando la aproximación. Por ejemplo, pueden construirse **modelos surrogados** de la función objetivo:

- Approximando F mediante mínimos cuadrados basados en funciones suaves.
- Approximando F mediante interpolación basada en una discretización de tamaño grande (digamos, al menos unas diez veces mayor) comparado con el periodo de las oscilaciones debidas a errores.

Una alternativa, sencilla, a estos métodos consiste en filtrar el carácter altamente oscilatorio de los errores:

- Limitando inferiormente el tamaño del intervalo en que se aplican diferencias finitas ('minimum perturbation' en la aproximación de derivadas de la herramienta MatLab). Como regla general, tal perturbación mínima debe tomarse entre 10 y 100 veces menor que el periodo de las oscilaciones debidas a errores.
- Con ello se aumenta el error asociado al cálculo mediante diferencias finitas, de modo que también deben aumentarse la tolerancia, tanto en las variables de diseño como en la función objetivo ('Xtolerance' y 'Function tolerance' en el criterio de parada).

ERRORES: EJEMPLO

Como ilustración, se considera la aplicación del algoritmo BFGS a la función bimodal contaminada con un error aleatorio de tamaño ε, tomando el punto (-2,1) como condición inicial.

- Para $\varepsilon = 10^{-8}$ (cien veces más pequeño que las tolerancias por defecto del método), el método sólo es capaz de calcular el mínimo con una precisión de 10^{-3}.
- Si, en cambio, se toma una perturbación mínima de 10^{-4}, el error final decrece a $4 \cdot 10^{-5}$.
- Si se toma $\varepsilon = 10^{-5}$ (diez veces mayor que la tolerancia, pero muy pequeño), el método converge después de dos pasos a un mínimo local espúreo muy próximo al punto inicial.
- Pero, nuevamente, subiendo la perturbación mínima a 10^{-3}, el método proporciona la solución con una precisión de $4 \cdot 10^{-4}$.

OPTIMIZACIÓN SIN RESTRICCIONES: CONCLUSIONES

Las conclusiones siguientes se refieren a **situaciones genéricas**. Como siempre, se pueden encontrar **contraejemplos más o menos exóticos** que contradicen algunas de las conclusiones.

- El método que converge más deprisa y mejor es el **método de Newton**, y el peor, el descenso más empinado.
- Pero el método de Newton conlleva un elevado **coste computacional**, debido a la necesidad de calcular o aproximar la matriz hessiana. Puede ser muy aconsejable para funciones objetivo estrictamente cuadráticas.
- Para funciones objetivo generales, el **método BFGS** representa, a día de hoy, **el mejor compromiso** entre convergencia y coste computacional.
- El escalado de las variables es muy importante, y puede afectar mucho al comportamiento de los métodos.
- Los errores pueden tener un efecto inesperado en el comportamiento de los métodos si las derivadas se calculan por diferencias finitas.
- Todos los métodos requieren **calcular o aproximar**, en cada paso, **la función objetivo y su gradiente**, cuyo coste puede ser muy alto debido a las disciplinas técnicas. El cálculo del gradiente en presencia de disciplinas técnicas complejas se trata en el siguiente epígrafe.

OPTIMIZACIÓN SIN RESTRICCIONES: EJERCICIO

Considérese el problema de encontrar la función $u = u(x)$ que (para valores fijos del parámetro $0 \leq w \leq 1$) minimiza el funcional

$$F(u) = \int_0^1 [w|u'(x)^2 + (1-w)u(x)^2]\, dx, \quad \text{con } u(0) = 1, u(1) = 0.$$

a. Discretice el funcional en una malla equiespaciada de N puntos, $x_1 = 0, x_2 = 1/(N-1), \ldots, x_N = 1$, para obtener la aproximación discreta $F(u) \simeq F^{\text{disc}}(u_1, \ldots, u_N)$ donde $u_j = u(x_j)$ para $j = 1, \ldots, N$.

b. Minimice F^{disc} mediante MatLab, analizando la dependencia de la solución del número de puntos y del parámetro w.

c. Minimize directamente $F(u) \simeq F^{\text{disc}}$, interpretando el significado de la ecuación 'gradiente-igual-a-cero'.

En el apartado (a) puede utilizar distintas discretizaciones. Utilice, al menos, la *regla del trapecio* y la *regla del punto medio*, $\int_0^1 f(x)\, dx \simeq \frac{1}{N-1} \sum_{j=1}^{N-1} f_{j+1/2}$, y compare ventajas e inconvenientes de ambas.

FUNCIÓN OBJETIVO DEFINIDAS IMPLÍCITAMENTE

Hasta ahora, se ha supuesto que la función objetivo depende explícitamente del vector de diseño $\boldsymbol{u} \in \mathbb{R}^N$. Sin embargo, debido a las **disciplinas técnicas**, puede depender también de otro vector $\boldsymbol{v} \in \mathbb{R}^M$, es decir,

$$F = F(\boldsymbol{u}, \boldsymbol{v}),$$

donde $\boldsymbol{v}$ está relacionado con $\boldsymbol{u}$ implícitamente mediante una ecuación vectorial ($\boldsymbol{h} \in \mathbb{R}^M$) del tipo

$$\boldsymbol{h}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}.$$

Se supone que el jacobiano de $\boldsymbol{h}$ respecto de $\boldsymbol{v}$, $\partial_{\boldsymbol{v}} \boldsymbol{h}$, es no singular: la ecuación anterior es resoluble localmente (teorema de la función implícita) en la forma $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{u})$.

Situación típica en optimización multidisciplinar: p.e., si se optimiza un ala para obtener resistencia mínima, $\boldsymbol{v}$ puede ser el campo de velocidades del flujo exterior y la ecuación $\boldsymbol{h} = 0$ incluír las ecuaciones del campo fluido convenientemente discretizadas.

DISCIPLINAS TÉCNICAS: OBSERVACIONES

- Si la variable de diseño fuese $(\boldsymbol{u}, \boldsymbol{v})$ se tendría un problema de optimización con una restricción. Pero no es así, la variable de diseño es sólo $\boldsymbol{u}$.
- En cambio, se trata de calcular:

$$\min_{\boldsymbol{u}} F(\boldsymbol{u}, \boldsymbol{v}), \quad \text{con } \boldsymbol{h}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0},$$

  con restricciones, estrictas y/o unilaterales, que pueden depender también de las variables adicionales $\boldsymbol{v}$.
- Si $M \gg 1$ (p.e., en aerodinámica), resolver la ecuación implícita $\boldsymbol{h} = \boldsymbol{0}$ requerir un esfuerzo computacional enorme. $M \sim 10^7$ es típico para avión completo.
- Si $M \gg N \gg 1$, es inviable eliminar $\boldsymbol{v}$ resolviendo la ecuación $\boldsymbol{g} = \boldsymbol{0}$ para reescribir la función objetivo como $\hat{F}(\boldsymbol{u}) = F(\boldsymbol{u}, \boldsymbol{v}(\boldsymbol{u}))$ y calcular $\boldsymbol{\nabla} \hat{F}$ por diferencias finitas; esfuerzo computacional $\sim M \cdot N$ en el mejor de los casos.

CÁLCULO IMPLÍCITO DEL VECTOR GRADIENTE

Cálculo el gradiente de la función $\hat{F}(\boldsymbol{u}) \equiv F(\boldsymbol{u}, \boldsymbol{v}(\boldsymbol{u}))$, con $\boldsymbol{v}(\boldsymbol{u})$ viene dada por $\boldsymbol{h}(\boldsymbol{u}, \boldsymbol{v}(\boldsymbol{u})) = \boldsymbol{0}$.

- Derivando respecto de $\boldsymbol{u}$ en ambas expresiones y aplicando la regla de la cadena, se tiene (para la matriz columna $\boldsymbol{\nabla}\hat{F}$)

$$\boldsymbol{\nabla}\hat{F}^{\top} = \partial_{\boldsymbol{u}} F + \partial_{\boldsymbol{v}} F \partial_{\boldsymbol{u}} \boldsymbol{v}, \quad \partial_{\boldsymbol{u}} \boldsymbol{h} + \partial_{\boldsymbol{v}} \boldsymbol{h} \partial_{\boldsymbol{u}} \boldsymbol{v} = \boldsymbol{0}.$$

  Típicamente, las matrices fila $\partial_{\boldsymbol{u}} F$ y $\partial_{\boldsymbol{v}} F$ y las matrices rectangulares $\partial_{\boldsymbol{u}} \boldsymbol{h}$ y $\partial_{\boldsymbol{v}} \boldsymbol{h}$ se calculan analíticamente.

- Resolviendo (formalmente) el segundo sistema lineal y sustituyendo en la primera ecuación se obtiene

$$\boldsymbol{\nabla}\hat{F}^{\top} = \partial_{\boldsymbol{u}} F - \partial_{\boldsymbol{v}} F (\partial_{\boldsymbol{v}} \boldsymbol{h})^{-1} \partial_{\boldsymbol{u}} \boldsymbol{h}.$$

  Resolver numéricamente el sistema $\partial_{\boldsymbol{u}} \boldsymbol{h} + \partial_{\boldsymbol{v}} \boldsymbol{h} \partial_{\boldsymbol{u}} \boldsymbol{v} = \boldsymbol{0}$ tiene un **coste computacional gigantesto** si $M \gg N >> 1$: $M \cdot N$ operaciones si la matriz $\partial_{\boldsymbol{v}} \boldsymbol{h}$ es dispersa, y $M^2 N$ operaciones si es llena.

MÉTODO DEL ADJUNTO

Método del adjunto:

- Se reescribe la expresión

$$\nabla \hat{F}^\top = \partial_{\boldsymbol{u}} F - \partial_{\boldsymbol{v}} F (\partial_{\boldsymbol{v}} \boldsymbol{h})^{-1} \partial_{\boldsymbol{u}} \boldsymbol{h}$$

como

$$\nabla \hat{F}^\top = \partial_{\boldsymbol{u}} F - \boldsymbol{w}^\top \partial_{\boldsymbol{u}} \boldsymbol{h}, \quad \text{con } \boldsymbol{w}^\top = \partial_{\boldsymbol{v}} F^\top (\partial_{\boldsymbol{v}} \boldsymbol{h})^{-1} \quad (\boldsymbol{w} \in \mathbb{R}^M).$$

- Se postmultiplica la segunda ecuación por $\partial_{\boldsymbol{v}} \boldsymbol{h}$ y se toma la traspuesta, obteniendo el **sistema adjunto** (para calcular numéricamente $\boldsymbol{w}$)

$$(\nabla_{\boldsymbol{v}} \boldsymbol{h})^\top \boldsymbol{w} = \partial_{\boldsymbol{v}} F^\top,$$

- **Coste computacional**: $\sim M$ si $\nabla_{\boldsymbol{v}} \boldsymbol{h}$) es dispersa y $\sim M^2$ si es llena; es decir, **¡N veces menor que resolviendo en problema directo!**.
- Sustituyendo $\boldsymbol{w}$ en $\nabla \hat{F}^\top = \partial_{\boldsymbol{u}} F - \boldsymbol{w}^\top \partial_{\boldsymbol{u}} \boldsymbol{h}$ (coste $\sim N$ si $\partial_{\boldsymbol{u}} \boldsymbol{h}$ es dispersa y $\sim M \cdot N$ si es llena) se obtiene el gradiente.

MÉTODO DEL ADJUNTO: RESUMEN

- Se tiene la función objetivo

$$\hat{F}(u) \equiv F(\boldsymbol{u}, \boldsymbol{v}(\boldsymbol{u})), \quad \text{con } \boldsymbol{v}(\boldsymbol{u}) \text{ dada por } \boldsymbol{h}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}.$$

  Es decir, para calcular la función objetivo para cada valor de la función de diseño, hay que resolver el problema (no lineal, en general) $\boldsymbol{h}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}$, que involucra a la variable auxiliar $\boldsymbol{v}$.

- El gradiente de la función objetivo viene dado por

$$\boldsymbol{\nabla} \hat{F} = \partial_{\boldsymbol{u}} F - \boldsymbol{w}^{\top} \partial_{\boldsymbol{u}} \boldsymbol{h},$$

  donde el vector $\boldsymbol{w}$ se calcula resolviendo el problema lineal (problema adjunto)

$$(\boldsymbol{\nabla}_{\boldsymbol{v}} \boldsymbol{h})^{\top} \boldsymbol{w} = \boldsymbol{\nabla}_{\boldsymbol{v}} F^{\top}.$$

- Los vectores $\partial_{\boldsymbol{u}} F$ y $\boldsymbol{\nabla}_{\boldsymbol{v}} F$ y las matrices $\boldsymbol{\nabla}_{\boldsymbol{u}} \boldsymbol{h}$ y $\boldsymbol{\nabla}_{\boldsymbol{v}} \boldsymbol{h}$ se calculan **analíticamente** derivando en la función objetivo F y en la función vectorial $\boldsymbol{h}$.

MÉTODO DEL ADJUNTO: COMENTARIOS

- Muchas **herramientas numéricas para sistemas de ecuaciones complejas** (fluidos, estructuras) tienen implementado el adjunto.
- Se ha considerado un problema con una sola ecuación (vectorial) implícita. Pero el método se extiende de modo natural a **varias disciplinas técnicas**. De hecho, formalmente, todas ellas se podrían reunir variables y ecuaciones en una sóla, que obedecería a la formulación anterior. Sin embargo, conviene separarlas, entre otras muchas razones, para tener un diseño modular.
- **Diseño de forma:** Cuando el vector de diseño incluye una función de forma (p.e., de un ala) convenientemente discretizada, $N \gg 1$ y hay que combinar adjunto y cambios de malla.
- **Problemas inversos** (p.e., inspección de materiales mediante ultrasonidos, imagen médica): la función objetivo puede medir diferencias entre datos medidos y calculados, y las variables/funciones de diseño ser propiedades del medio (impedancia acústica, índice de refracción).