MS-E2122 - Nonlinear Optimization
Lecture VI

Fernando Dias

Department of Mathematics and Systems Analysis

Aalto University
School of Science

September 26, 2023

Outline of this lecture

Conjugate gradient method

Quasi-Newton method

Complexity, convergence and conditioning

Last Week

- Linear Search Methods;
- Gradient and Newton Methods.

Last week...

Outline of this lecture

Conjugate gradient method

Quasi-Newton method

Complexity, convergence and conditioning

The concept of conjugacy

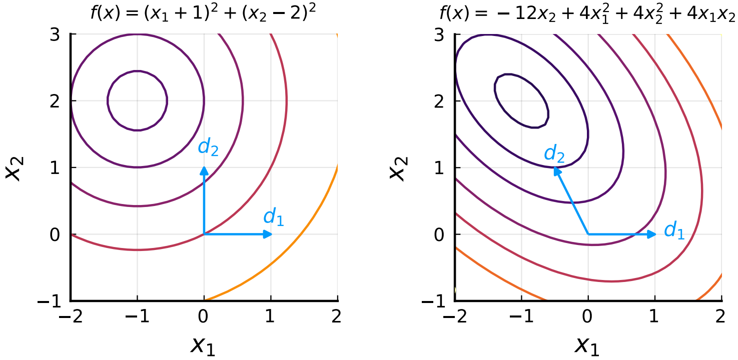
Definition 1

Let H be an $n \times n$ symmetric matrix. Vectors $d_1, \ldots, d_n$ are called (H-)conjugate if they are **linearly independent** and $d_i^\top H d_j = 0$, for all $i, j = 1, \ldots, n : i \neq j$.

The concept of conjugacy

Definition 1

Let H be an $n \times n$ symmetric matrix. Vectors $d_1, \dots, d_n$ are called $(H\text{-})$conjugate if they are **linearly independent** and $d_i^\top H d_j = 0$, for all $i, j = 1, \dots, n : i \neq j$.



$f(x) = (x_1 + 1)^2 + (x_2 - 2)^2$

$f(x) = -12x_2 + 4x_1^2 + 4x_2^2 + 4x_1x_2$

d_1 and d_2 are H-conjugates; on the left, $H = I$.

The concept of conjugacy

The motivation to use conjugates comes from quadratic problems, in particular their use as **approximations** for general functions.

Let $f(x) = c^\top x + \frac{1}{2}x^\top Hx$ with H symmetric and let $d_1, \dots, d_n$ be H-conjugate directions.

The concept of conjugacy

The motivation to use conjugates comes from quadratic problems, in particular their use as **approximations** for general functions.

Let $f(x) = c^\top x + \frac{1}{2}x^\top Hx$ with H symmetric and let $d_1, \dots, d_n$ be H-conjugate directions. Given x_0, any point x can be described as $x_0 + \sum_{j=1}^n \lambda_j d_j$. Thus

$$
\begin{aligned}
f(x) = F(\lambda) &= c^\top(x_0 + \sum_{j=1}^n \lambda_j d_j) + \frac{1}{2}(x_0 + \sum_{j=1}^n \lambda_j d_j)^\top H(x_0 + \sum_{j=1}^n \lambda_j d_j) \\
&= \sum_{j=1}^n \left[c^\top(x_0 + \lambda_j d_j) + \frac{1}{2}(x_0 + \lambda_j d_j)^\top H(x_0 + \lambda_j d_j)\right].
\end{aligned}
$$

The concept of conjugacy

The motivation to use conjugates comes from quadratic problems, in particular their use as **approximations** for general functions.

Let $f(x) = c^\top x + \frac{1}{2}x^\top Hx$ with H symmetric and let $d_1, \dots, d_n$ be H-conjugate directions. Given x_0, any point x can be described as $x_0 + \sum_{j=1}^n \lambda_j d_j$. Thus

$$f(x) = F(\lambda) = c^\top (x_0 + \sum_{j=1}^n \lambda_j d_j) + \frac{1}{2}(x_0 + \sum_{j=1}^n \lambda_j d_j)^\top H(x_0 + \sum_{j=1}^n \lambda_j d_j)$$

$$= \sum_{j=1}^n \left[c^\top (x_0 + \lambda_j d_j) + \frac{1}{2}(x_0 + \lambda_j d_j)^\top H(x_0 + \lambda_j d_j)\right].$$

Notice that $F(\lambda) = \sum_{j=1}^n F_j(\lambda_j)$ is separable. Assuming that H is positive definite ($d_j^\top H d_j > 0$), the optimal $\overline{\lambda}$ is given by

$$\overline{\lambda}_j = -\frac{c^\top d_j + x_0^\top H d_j}{d_j^\top H d_j}, \text{ for all } j = 1, \dots, n.$$

The concept of conjugacy

Example: min. $\left\{f(x) = -12x_2 + 4x_1^2 + 4x_2^2 + 4x_1x_2\right\}$

1. Notice that $H = \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix}$. Letting $d_1 = (1, 0)$, $d_2 = (a, b)$ must satisfy $d_1^\top H d_2 = 0 \Rightarrow 8a + 4b = 0$. Pick $d_2 = (-1, 2)$.

The concept of conjugacy

Example: min. $\{f(x) = -12x_2 + 4x_1^2 + 4x_2^2 + 4x_1x_2\}$

1. Notice that $H = \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix}$. Letting $d_1 = (1, 0)$, $d_2 = (a, b)$ must satisfy $d_1^\top H d_2 = 0 \Rightarrow 8a + 4b = 0$. Pick $d_2 = (-1, 2)$.

2. Calculate optimal $\overline{\lambda} = (\overline{\lambda}_1, \overline{\lambda}_2)$. For $\overline{\lambda}_1$, we have

$$\overline{\lambda}_1 = -\left(\begin{bmatrix} 0 \\ -12 \end{bmatrix}^\top \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)^{-1}$$

The concept of conjugacy

Example: min. $\left\{f(x) = -12x_2 + 4x_1^2 + 4x_2^2 + 4x_1x_2\right\}$

1. Notice that $H = \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix}$. Letting $d_1 = (1, 0)$, $d_2 = (a, b)$ must satisfy $d_1^\top H d_2 = 0 \Rightarrow 8a + 4b = 0$. Pick $d_2 = (-1, 2)$.

2. Calculate optimal $\overline{\lambda} = (\overline{\lambda}_1, \overline{\lambda}_2)$. For $\overline{\lambda}_1$, we have

$$\overline{\lambda}_1 = -\left(\begin{bmatrix} 0 \\ -12 \end{bmatrix}^\top \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)^{-1}$$

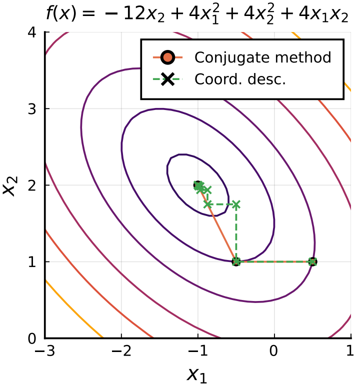
$$\overline{\lambda}_2 = -\left(\begin{bmatrix} 0 \\ -12 \end{bmatrix}^\top \begin{bmatrix} -1 \\ -2 \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} -1 \\ -2 \end{bmatrix}\right)\left(\begin{bmatrix} -1 \\ -2 \end{bmatrix}^\top \begin{bmatrix} 8 & 4 \\ 4 & 8 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix}\right)^{-1}$$

The concept of conjugacy

Example: For $x_0 = (\frac{1}{2}, 1)$, $\overline{\lambda}_1 = -1$ and $x_1 = x_0 + d_1\lambda_1 = (-\frac{1}{2}, 1)$.
Then $\overline{\lambda}_2 = \frac{1}{2}$ and $x_2 = x_1 + \frac{1}{2}d_2 = (-1, 2)$, which is optimal.

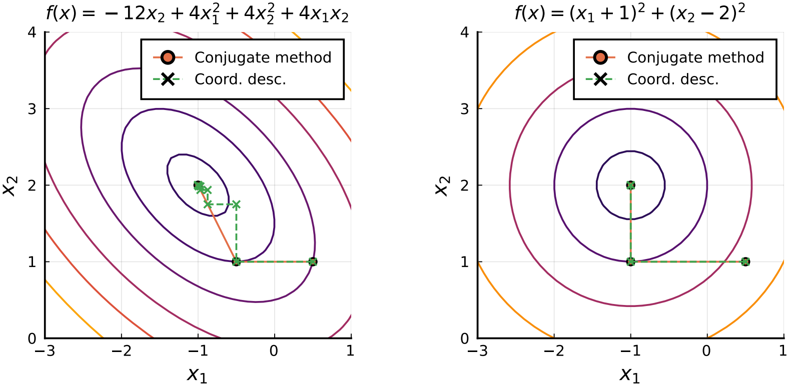
The concept of conjugacy

Example: For $x_0 = (\frac{1}{2}, 1)$, $\overline{\lambda}_1 = -1$ and $x_1 = x_0 + d_1\lambda_1 = (-\frac{1}{2}, 1)$. Then $\overline{\lambda}_2 = \frac{1}{2}$ and $x_2 = x_1 + \frac{1}{2}d_2 = (-1, 2)$, which is optimal.



The concept of conjugacy

Example: For $x_0 = (\frac{1}{2}, 1)$, $\overline{\lambda}_1 = -1$ and $x_1 = x_0 + d_1\lambda_1 = (-\frac{1}{2}, 1)$. Then $\overline{\lambda}_2 = \frac{1}{2}$ and $x_2 = x_1 + \frac{1}{2}d_2 = (-1, 2)$, which is optimal.



Optimising f with the conjugate method and coordinate descent (left). For $H = I$, both methods coincide (right).

Generating conjugate directions

The Gram-Schmidt method is used to obtain H-conjugate vectors.

1. Assume that $\xi_0, \ldots, \xi_k$ linearly independent vectors are known.
   Set $d_0 = \xi_0$.

Generating conjugate directions

The Gram-Schmidt method is used to obtain H-conjugate vectors.

1. Assume that $\xi_0, \dots, \xi_k$ linearly independent vectors are known. Set $d_0 = \xi_0$.
2. Set coefficients α^i_{k+1} such that d_{k+1} is H-conjugate to $d_0, \dots, d_k$

$$d_{k+1} = \xi_{k+1} + \sum_{l=0}^{k} \alpha^l_{k+1} d_l.$$

Generating conjugate directions

The Gram-Schmidt method is used to obtain H-conjugate vectors.

1. Assume that $\xi_0, \dots, \xi_k$ linearly independent vectors are known. Set $d_0 = \xi_0$.
2. Set coefficients α^i_{k+1} such that d_{k+1} is H-conjugate to $d_0, \dots, d_k$

$$d_{k+1} = \xi_{k+1} + \sum_{l=0}^{k} \alpha^l_{k+1} d_l.$$

3. H-conjugacy will be obtained if, for each $i = 0, \dots, k$,

$$d_{k+1}{}^\top H d_i = \xi_{k+1}{}^\top H d_i + \left(\sum_{l=0}^{k} \alpha^l_{k+1} d_l \right)^\top H d_i = 0.$$

Due to the H-conjugacy, $d_l^\top H d_k = 0$ for all $l \neq k$. Thus

$$\alpha^i_{k+1} = \frac{-\xi^\top_{k+1} H d_i}{d_i^\top H d_i}, \text{ for } i = 0, \dots, k. \tag{1}$$

The concept of conjugacy

The following are key properties of conjugate directions.

Theorem 2

Let $f(x) = c^\top x + \frac{1}{2}x^\top Hx$, where H is an $n \times n$ symmetric matrix. Let $d_1, \dots, d_n$ be H-conjugate, and let x_0 be an arbitrary starting point. Let λ_j be the optimal solution to $F_j(\lambda_j) = f(x_0 + \lambda_j d_j)$ for all $j = 1, \dots, n$. Then, for $k = 1, \dots, n$ we must have:

1. *x_{k+1} is optimal to min. $\{f(x) : x - x_0 \in L(d_1, \dots, d_k)\}$ where $L(d_1, \dots, d_k) = \left\{\sum_{j=1}^{k} \mu_j d_j : \mu_j \in \mathbb{R}, j = 1, \dots, k\right\}$;*
2. *$\nabla f(x_{k+1})^\top d_j = 0$, for all $j = 1, \dots, k$;*

The concept of conjugacy

The following are key properties of conjugate directions.

Theorem 2

Let $f(x) = c^\top x + \frac{1}{2}x^\top Hx$, where H is an $n \times n$ symmetric matrix. Let $d_1, \dots, d_n$ be H-conjugate, and let x_0 be an arbitrary starting point. Let λ_j be the optimal solution to $F_j(\lambda_j) = f(x_0 + \lambda_j d_j)$ for all $j = 1, \dots, n$. Then, for $k = 1, \dots, n$ we must have:

1. *x_{k+1} is optimal to min. $\{f(x) : x - x_0 \in L(d_1, \dots, d_k)\}$ where $L(d_1, \dots, d_k) = \left\{\sum_{j=1}^{k} \mu_j d_j : \mu_j \in \mathbb{R}, j = 1, \dots, k\right\}$;*
2. *$\nabla f(x_{k+1})^\top d_j = 0$, for all $j = 1, \dots, k$;*

Proof reasoning: $\nabla f(x_{k+1})$ is orthogonal to $L(d_1, \dots, d_k)$ since $F_j'(\lambda_j) = d_j^\top \nabla f(x_0 + \lambda_j d_j) = 0$ is the optimality condition for λ_j.

Remark: Theorem 2 guarantees that one can use $\nabla f(x_k)$ to generate conjugate directions.

Conjugate gradient method

The conjugate gradient method generates sequence of iterates

$$x_{k+1} = x_k + \lambda_k d_k,$$

where $d_0 = -\nabla f(x_0)$. Given x_{k+1} with $\nabla f(x_{k+1}) \neq 0$ we use (1) to generate d_{k+1} by making $\xi_{k+1} = -\nabla f(x_{k+1})$. Thus

$$d_{k+1} = -\nabla f(x_{k+1}) + \alpha_k d_k, \text{ with } \alpha_k = \frac{\nabla f(x_{k+1})^\top H d_k}{d_k^\top H d_k}.$$

Conjugate gradient method

The conjugate gradient method generates sequence of iterates

$$x_{k+1} = x_k + \lambda_k d_k,$$

where $d_0 = -\nabla f(x_0)$. Given x_{k+1} with $\nabla f(x_{k+1}) \neq 0$ we use (1) to generate d_{k+1} by making $\xi_{k+1} = -\nabla f(x_{k+1})$. Thus

$$d_{k+1} = -\nabla f(x_{k+1}) + \alpha_k d_k, \text{ with } \alpha_k = \frac{\nabla f(x_{k+1})^\top H d_k}{d_k^\top H d_k}.$$

Noticing that $\nabla f(x_{k+1}) - \nabla f(x_k) = H(x_{k+1} - x_k) = \lambda_k H d_k$ and that $d_k = -\nabla f(x_k) + \alpha_{k-1} d_{k-1}$, the step update becomes

$$d_{k+1} = -\nabla f(x_{k+1}) + \alpha_k d_k, \text{ with } \alpha_k = \frac{||\nabla f(x_{k+1})||^2}{||\nabla f(x_k)||^2}$$

Conjugate gradient method

Algorithm Conjugate gradient method

```
initialise. tolerance $\epsilon > 0$, initial point $x_0$, direction $d_0 = -\nabla f(x_0)$, $k = 1$
while $||\nabla f(x_k)|| > \epsilon$ do
    $y_0 = x_{k-1}$
    $d_0 = -\nabla f(y_0)$
    for $j = 1, \dots, n$ do
        $\overline{\lambda}_j = \mathrm{argmin}_{\lambda \geq 0} \{f(y_{j-1} + \lambda d_{j-1})\}$
        $y_j = y_{j-1} + \overline{\lambda}_j d_{j-1}$
        $d_j = -\nabla f(y_j) + \alpha_j d_{j-1}$, where $\alpha_j = \frac{||\nabla f(y_j)||^2}{||\nabla f(y_{j-1})||^2}$.
    end for
    $x_k = y_n$, $k = k + 1$
end while
return $x_k$.
```

Conjugate gradient method

Remarks:

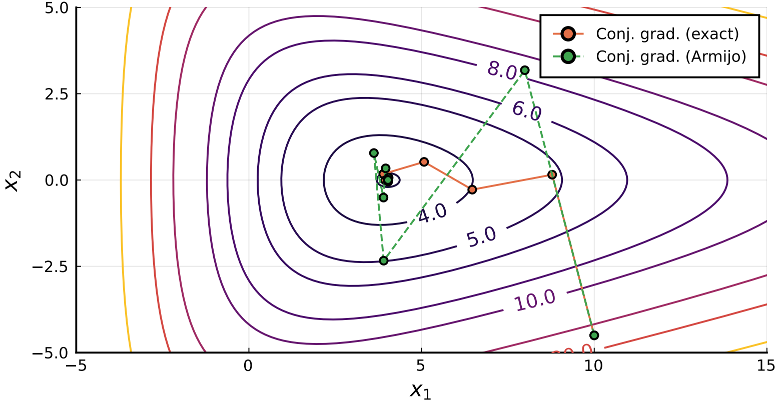
1. The conjugate gradient method with $\alpha_k = \frac{||\nabla f(x_{k+1})||^2}{||\nabla f(x_k)||^2}$ is due to Fletcher and Reevers.
2. Nonquadratic problems can also be solved, tipically taking more than n steps;
3. Using $\alpha_k = \frac{\nabla f(x_{k+1})^\top (\nabla f(x_{k+1}) - \nabla f(x_k))}{||\nabla f(x_k)||^2}$ has better numerical performance for nonquadratic problems;
4. Notice that the algorithm is restarted every n iterations to recover conjugacy;
5. Notice that

$$d_{j+1} = \frac{1}{\mu}\left[\mu(-\nabla f(x_{j+1})) + (1-\mu)d_j\right]$$

where $\mu = \frac{1}{(1+\alpha_j)}$. That is, d_{j+1} is a convex combination between steepest descent and conjugate direction.

Conjugate gradient method

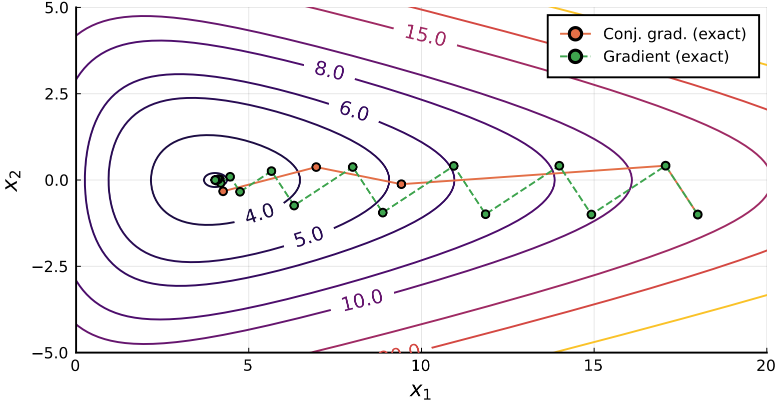
$$f(x) = e^{(-(x_1-3)/2)} + e^{((4x_2+x_1)/10)} + e^{((-4x_2+x_1)/10)}$$



Conjugate gradient method applied to f. Convergence is observed in 24 steps using exact line search and 28 using Armijo's rule ($\epsilon = 10^{-4}$)

Conjugate gradient method

$$f(x) = e^{(-(x_1-3)/2)} + e^{((4x_2+x_1)/10)} + e^{((-4x_2+x_1)/10)}$$



Comparing the gradient method and the conjugate gradient with different starting point. The gradient method takes 20 steps while the conjugate gradient takes 8 steps to converge ($\epsilon = 10^{-4}$)

Outline of this lecture

Conjugate gradient method

Quasi-Newton method

Complexity, convergence and conditioning

Quasi-Newton methods

Can be seen as approximations of Newton's method that do not utilise Hessians or their inverses.

Instead, they rely on search directions $d_k = -D_k \nabla f(x_k)$, where D_k approximates the inverse Hessian $H(x_k)^{-1}$ using **only gradient information**.

D_k is computed using local curvature information at x. Let

$$
\begin{aligned}
p_k &= \lambda_k d_k = x_{k+1} - x_k \\
q_k &= \nabla f(x_{k+1}) - \nabla f(x_k) = H(x_{k+1} - x_k) = H p_k .
\end{aligned}
$$

Quasi-Newton methods

Can be seen as approximations of Newton's method that do not utilise Hessians or their inverses.

Instead, they rely on search directions $d_k = -D_k \nabla f(x_k)$, where D_k approximates the inverse Hessian $H(x_k)^{-1}$ using **only gradient information**.

D_k is computed using local curvature information at x. Let

$$\begin{aligned} p_k &= \lambda_k d_k = x_{k+1} - x_k \\ q_k &= \nabla f(x_{k+1}) - \nabla f(x_k) = H(x_{k+1} - x_k) = H p_k. \end{aligned}$$

When $f(x)$ is quadratic, these methods approximate H^{-1} by updating D_k using p_k and q_k such that D_n converges to H^{-1}.

Quasi-Newton methods

From an initial guess D_0, we make $D_{k+1} = D_k + C_k$ such that at $k = n$, we observe $D_n = H^{-1}$.

Quasi-Newton methods

From an initial guess D_0, we make $D_{k+1} = D_k + C_k$ such that at $k = n$, we observe $D_n = H^{-1}$.

This holds when p_j, for all $j = 1, \ldots, k$, are eigenvectors of $D_{k+1}H$ with unit eigenvalues, i.e.,

$$D_{k+1}Hp_j = p_j \Rightarrow D_{k+1}q_j = p_j, \; j = 1, \ldots, k$$
$$p_j = D_k q_j + C_k q_j = D_k H p_j + C_k q_j = p_j + C_k q_j, \; j = 1, \ldots, k-1,$$

which implies that $C_k q_j = 0$, for $j = 1, \ldots, k-1$.

Quasi-Newton methods

From an initial guess D_0, we make $D_{k+1} = D_k + C_k$ such that at $k = n$, we observe $D_n = H^{-1}$.

This holds when p_j, for all $j = 1, \ldots, k$, are eigenvectors of $D_{k+1}H$ with unit eigenvalues, i.e.,

$$D_{k+1}Hp_j = p_j \Rightarrow D_{k+1}q_j = p_j,\ j = 1, \ldots, k$$
$$p_j = D_kq_j + C_kq_j = D_kHp_j + C_kq_j = p_j + C_kq_j,\ j = 1, \ldots, k-1,$$

which implies that $C_kq_j = 0$, for $j = 1, \ldots, k-1$.

Now, for $j = k$, we require

$$D_{k+1}q_k = p_k = D_kq_k + C_kq_k \text{ or } C_kq_k = p_k - D_kq_k.$$

These two conditions form the Quasi-Newton conditions, which guarantee that $D_n = H^{-1}$.

Quasi-Newton methods

The Davidon-Fletcher-Powell (DFP) update is:

$$D_{k+1} = D_k + \frac{p_k p_k^\top}{p_k^\top q_k} - \frac{D_k q_k q_k^\top D_k}{q_k^\top D_k q_k} = D_k + C_k^{DFP}$$

It can be shown that the following hold for C^{DFP}:

- $C_k^{DFP} q_j = C_k^{DFP} H p_j$
  $= \frac{p_k p_k^\top H p_j}{p_k^\top q_k} - \frac{D_k q_k p_k^\top H D_k H p_j}{q_k^\top D_k q_k} = 0,$ for $j = 1, \dots, k-1;$
- $C_k^{DFP} q_k = \frac{p_k p_k^\top q_k}{p_k^\top q_k} - \frac{D_k q_k q_k^\top D_k q_k}{q_k^\top D_k q_k} = p_k - D_k q_k.$

Quasi-Newton methods

The Davidon-Fletcher-Powell (DFP) update is:

$$D_{k+1} = D_k + \frac{p_k p_k^\top}{p_k^\top q_k} - \frac{D_k q_k q_k^\top D_k}{q_k^\top D_k q_k} = D_k + C_k^{DFP}$$

It can be shown that the following hold for C^{DFP}:

- $C_k^{DFP} q_j = C_k^{DFP} H p_j$
$$= \frac{p_k p_k^\top H p_j}{p_k^\top q_k} - \frac{D_k q_k p_k^\top H D_k H p_j}{q_k^\top D_k q_k} = 0, \quad \text{for } j = 1, \dots, k-1;$$
- $C_k^{DFP} q_k = \frac{p_k p_k^\top q_k}{p_k^\top q_k} - \frac{D_k q_k q_k^\top D_k q_k}{q_k^\top D_k q_k} = p_k - D_k q_k.$

The most important Quasi-Newton method is Broyden-Fletcher-Goldfarb-Shanno (BFGS). BFGS augments C^{DFP} to **mitigate numerical difficulties** from **near-singular approximations**.

Quasi-Newton methods

BFGS is part of the Broyden family of updates:

$$C^B = C^{DFP} + \phi \frac{\tau_j v_k v_k^\top}{p_k^\top q_k}, \text{ with } \phi \in (0,1),$$

where $v_k = p_k - \left(\frac{1}{\tau_k}\right) D_k q_k$, $\tau_k = \frac{q_j^\top D_k q_k}{p_k^\top q_k}$. Using $\phi = 1$, we obtain

$$C_k^{BFGS} = \frac{p_k p_k^\top}{p_k^\top q_k}\left(1 + \frac{q_k^\top D_k q_k}{p_k^\top q_k}\right) - \frac{D_k q_k p_k^\top + p_k q_k^\top D_k}{p_k^\top q_k}.$$

Quasi-Newton methods

BFGS is part of the Broyden family of updates:

$$C^B = C^{DFP} + \phi \frac{\tau_j v_k v_k^\top}{p_k^\top q_k}, \text{ with } \phi \in (0,1),$$

where $v_k = p_k - \left(\frac{1}{\tau_k}\right) D_k q_k$, $\tau_k = \frac{q_j^\top D_k q_k}{p_k^\top q_k}$. Using $\phi = 1$, we obtain

$$C_k^{BFGS} = \frac{p_k p_k^\top}{p_k^\top q_k}\left(1 + \frac{q_k^\top D_k q_k}{p_k^\top q_k}\right) - \frac{D_k q_k p_k^\top + p_k q_k^\top D_k}{p_k^\top q_k}.$$

Remarks:

1. BFGS is often presented approximating the Hessian directly (B_k). Then $D_{k+1} = B_{k+1}^{-1} = (B_k + \overline{C}_k^{BFGS})^{-1}$, with

$$\overline{C}_k^{BFGS} = \frac{q_k q_k^\top}{q_k^\top p_k} - \frac{B_k p_k p_k^\top B_k}{p_k^\top B_k p_k}.$$

Quasi-Newton methods

BFGS is part of the Broyden family of updates:

$$C^B = C^{DFP} + \phi \frac{\tau_j v_k v_k^\top}{p_k^\top q_k}, \text{ with } \phi \in (0,1),$$

where $v_k = p_k - \left(\frac{1}{\tau_k}\right) D_k q_k$, $\tau_k = \frac{q_j^\top D_k q_k}{p_k^\top q_k}$. Using $\phi = 1$, we obtain

$$C_k^{BFGS} = \frac{p_k p_k^\top}{p_k^\top q_k}\left(1 + \frac{q_k^\top D_k q_k}{p_k^\top q_k}\right) - \frac{D_k q_k p_k^\top + p_k q_k^\top D_k}{p_k^\top q_k}.$$

Remarks:

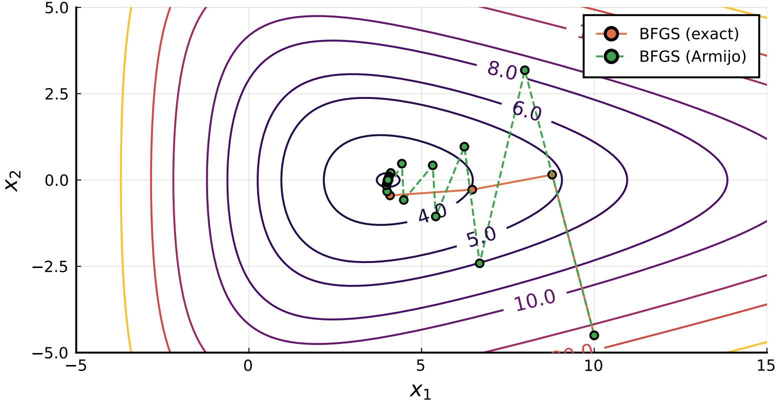
1. BFGS is often presented approximating the Hessian directly (B_k). Then $D_{k+1} = B_{k+1}^{-1} = (B_k + \overline{C}_k^{BFGS})^{-1}$, with

$$\overline{C}_k^{BFGS} = \frac{q_k q_k^\top}{q_k^\top p_k} - \frac{B_k p_k p_k^\top B_k}{p_k^\top B_k p_k}.$$

2. The limited memory BFGS (l-BFGS) is a computationally efficient implementation with minimum information storage.

Quasi-Newton methods

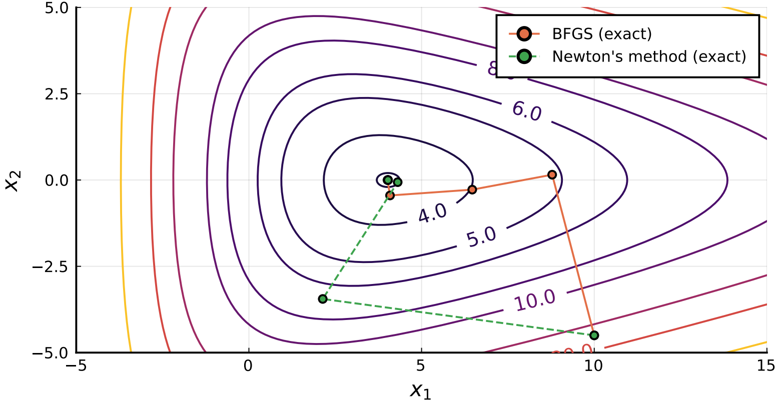
$$f(x) = e^{(-(x_1-3)/2)} + e^{((4x_2+x_1)/10)} + e^{((-4x_2+x_1)/10)}$$



BFGS method applied to f. Convergence is observed in 6 steps using exact line search and 30 using Armijo's rule ($\epsilon = 10^{-4}$)

Quasi-Newton methods

$$f(x) = e^{(-(x_1-3)/2)} + e^{((4x_2+x_1)/10)} + e^{((-4x_2+x_1)/10)}$$



Comparing the BFGS method with the Newton's method. Convergence is observed in 4 steps with the Newton's method and 6 with BFGS ($\epsilon = 10^{-4}$)

Outline of this lecture

Conjugate gradient method

Quasi-Newton method

Complexity, convergence and conditioning

Complexity, convergence and conditioning

Complexity: we use the **Big-O notation** (link) to bound the maximum number of iterations of algorithms.

Complexity, convergence and conditioning

Complexity: we use the **Big-O notation** (link) to bound the maximum number of iterations of algorithms.

Definition 3 (Polynomial (efficient) algorithms)

Given a problem P, a problem instance $X \in P$ with length $L(X)$ in binary representation, and an algorithm A that solves X, let $f_A(X)$ be the number of *elementary calculations* required to run A on X. Then, the running time of A on X is proportional to

$$f_A^*(n) = \sup_X \{f_A(X) : L(X) = n\}.$$

Algorithm A is *polynomial* for a problem P if $f_A^*(n) = O(n^p)$ for some integer p.

Complexity, convergence and conditioning

Complexity: we use the **Big-O notation** (link) to bound the maximum number of iterations of algorithms.

Definition 3 (Polynomial (efficient) algorithms)

Given a problem P, a problem instance $X \in P$ with length $L(X)$ in binary representation, and an algorithm A that solves X, let $f_A(X)$ be the number of *elementary calculations* required to run A on X. Then, the running time of A on X is proportional to

$$f_A^*(n) = \sup_X \{f_A(X) : L(X) = n\} .$$

Algorithm A is *polynomial* for a problem P if $f_A^*(n) = O(n^p)$ for some integer p.

Remark: complexity bounds consider the performance of algorithms from a worst-case performance perspective.

Complexity, convergence and conditioning

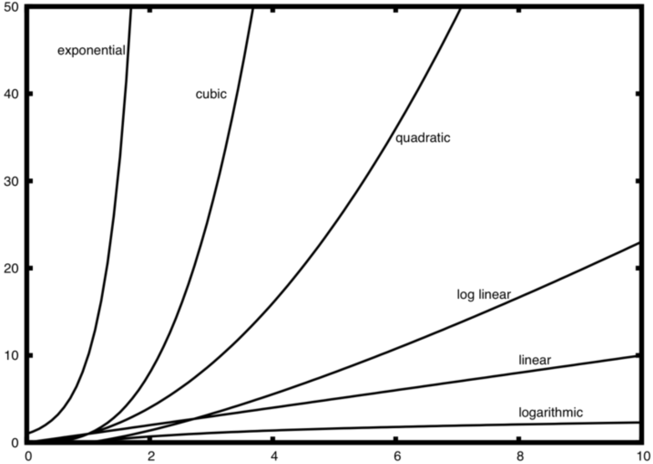


Figure: Comparing functions bounds (source link)

Complexity, convergence and conditioning

Local analysis often provides practical information by focusing on the behaviour of a sequence $\{x_k\}$ converging to a unique point $\overline{x}$.

Complexity, convergence and conditioning

Local analysis often provides practical information by focusing on the behaviour of a sequence $\{x_k\}$ converging to a unique point $\overline{x}$.

Rate of convergence considers an error function $e : \mathbb{R}^n \to \mathbb{R}$ such that $e(x) \geq 0$ for $x \in \mathbb{R}^n$. Typical choices are:

1. $e(x) = ||x - \overline{x}||$
2. $e(x) = |f(x) - f(\overline{x})|$

The sequence $\{e(x)\}$ is then compared with the geometric progression β^k, with $k = 1, 2, \ldots$ and $\beta \in (0, 1)$.

Complexity, convergence and conditioning

Local analysis often provides practical information by focusing on the behaviour of a sequence $\{x_k\}$ converging to a unique point $\overline{x}$.

Rate of convergence considers an error function $e : \mathbb{R}^n \to \mathbb{R}$ such that $e(x) \geq 0$ for $x \in \mathbb{R}^n$. Typical choices are:

1. $e(x) = ||x - \overline{x}||$
2. $e(x) = |f(x) - f(\overline{x})|$

The sequence $\{e(x)\}$ is then compared with the geometric progression β^k, with $k = 1, 2, \ldots$ and $\beta \in (0, 1)$.

Linear convergence: $e(x)$ converges linearly if exists $q > 0$ and $\beta \in (0, 1)$ such that $e(x_k) \leq q\beta^k$ for all k, which implies that

$$\lim_{k\to\infty} \sup \frac{e(x_{k+1})}{e(x_k)} \leq \beta.$$

Complexity, convergence and conditioning

Order p convergence: $e(x)$ converges superlinearly if there exists $\beta \in (0,1)$, $q > 0$, and $p > 1$ such that $e(x_k) \leq q\beta^{p^k}$ for all k.

$p = 2$ is the quadratic convergence case. Any p-order convergence with $p > 1$ is obtained if:

$$\lim_{k\to\infty} \sup \frac{e(x_{k+1})}{e(x_k)^p} < \infty, \text{ which is true if } \lim_{k\to\infty} \sup \frac{e(x_{k+1})}{e(x_k)} = 0.$$

Complexity, convergence and conditioning

Order p convergence: $e(x)$ converges superlinearly if there exists $\beta \in (0,1)$, $q > 0$, and $p > 1$ such that $e(x_k) \leq q\beta^{p^k}$ for all k.

$p = 2$ is the quadratic convergence case. Any p-order convergence with $p > 1$ is obtained if:

$$\lim_{k\to\infty} \sup \frac{e(x_{k+1})}{e(x_k)^p} < \infty, \text{ which is true if } \lim_{k\to\infty} \sup \frac{e(x_{k+1})}{e(x_k)} = 0.$$

Remarks:

1. Nonlinear optimisation methods often converge linearly. This may be satisfactory if β is not close to 1.
2. Several algorithms attain superlinear convergence for particular problems. Newton's method is an important example.

Complexity, convergence and conditioning

Theorem 4 (Convergence of the gradient method)

Let $f(x) = \frac{1}{2}x^\top Hx$ where H is a positive definite symmetric matrix. Suppose $f(x)$ is minimised with the gradient method using an exact line search. Let $\underline{\lambda} = \min_{i=1,\dots,n} \lambda_i$ and $\overline{\lambda} = \max_{i=1,\dots,n} \lambda_i$, where λ_i are eigenvalues of H. Then, for all k,

$$\frac{f(x_{k+1})}{f(x_k)} \leq \left(\frac{\overline{\lambda} - \underline{\lambda}}{\overline{\lambda} + \underline{\lambda}}\right)^2$$

Complexity, convergence and conditioning

Theorem 4 (Convergence of the gradient method)

Let $f(x) = \frac{1}{2}x^\top Hx$ where H is a positive definite symmetric matrix. Suppose $f(x)$ is minimised with the gradient method using an exact line search. Let $\underline{\lambda} = \min_{i=1,\ldots,n} \lambda_i$ and $\overline{\lambda} = \max_{i=1,\ldots,n} \lambda_i$, where λ_i are eigenvalues of H. Then, for all k,

$$\frac{f(x_{k+1})}{f(x_k)} \leq \left(\frac{\overline{\lambda} - \underline{\lambda}}{\overline{\lambda} + \underline{\lambda}}\right)^2$$

Remarks:

1. Notice the effect of scaling due to dependence on eigenvalues;

Complexity, convergence and conditioning

Theorem 4 (Convergence of the gradient method)

Let $f(x) = \frac{1}{2}x^\top Hx$ where H is a positive definite symmetric matrix. Suppose $f(x)$ is minimised with the gradient method using an exact line search. Let $\underline{\lambda} = \min_{i=1,\dots,n} \lambda_i$ and $\overline{\lambda} = \max_{i=1,\dots,n} \lambda_i$, where λ_i are eigenvalues of H. Then, for all k,

$$\frac{f(x_{k+1})}{f(x_k)} \leq \left(\frac{\overline{\lambda} - \underline{\lambda}}{\overline{\lambda} + \underline{\lambda}}\right)^2$$

Remarks:

1. Notice the effect of scaling due to dependence on eigenvalues;
2. The same result applies to general functions, as well as using Armijo's rule instead of exact line search.

Complexity, convergence and conditioning

Theorem 5 (Convergence of Newton's method - general case)

Let $g : \mathbb{R}^n \to \mathbb{R}^n$ *be differentiable,* $\overline{x}$ *such that* $g(\overline{x}) = 0$*, and let* $\{e(x_k)\} = \{||x_k - \overline{x}||\}$*. Moreover, let* $N_\delta(\overline{x}) = \{x : ||x - \overline{x}|| \leq \delta\}$ *for some* $\delta > 0$*. Then*

1. *There exists* $\delta > 0$ *such that if* $x_0 \in N_\delta(\overline{x})$*, the sequence* $\{x_k\}$ *with* $x_{k+1} = x_k - (\nabla g(x_k)^\top)^{-1} g(x_k)$ *belongs to* $N_\delta(\overline{x})$ *and converges to* $\overline{x}$*, while* $\{e(x_k)\}$ *converges superlinearly.*

Complexity, convergence and conditioning

Theorem 5 (Convergence of Newton's method - general case)

Let $g : \mathbb{R}^n \to \mathbb{R}^n$ be differentiable, $\overline{x}$ such that $g(\overline{x}) = 0$, and let $\{e(x_k)\} = \{||x_k - \overline{x}||\}$. Moreover, let $N_\delta(\overline{x}) = \{x : ||x - \overline{x}|| \leq \delta\}$ for some $\delta > 0$. Then

1. *There exists $\delta > 0$ such that if $x_0 \in N_\delta(\overline{x})$, the sequence $\{x_k\}$ with $x_{k+1} = x_k - (\nabla g(x_k)^\top)^{-1} g(x_k)$ belongs to $N_\delta(\overline{x})$ and converges to $\overline{x}$, while $\{e(x_k)\}$ converges superlinearly.*
2. *If for some $L > 0$, $M > 0$, and for all $x, y \in N_\delta(\overline{x})$, $\lambda \in (0, \delta]$*

$$\nabla g(x) - \nabla g(y) \leq L||x - y|| \quad \textit{and} \quad ||(\nabla g(x_k)^\top)^{-1}|| \leq M,$$

then, if $x_0 \in N_\delta(\overline{x})$, we have for $k = 0, 1, \ldots$

$$||x_{k+1} - \overline{x}|| \leq \frac{LM}{2} ||x_k - \overline{x}||^2.$$

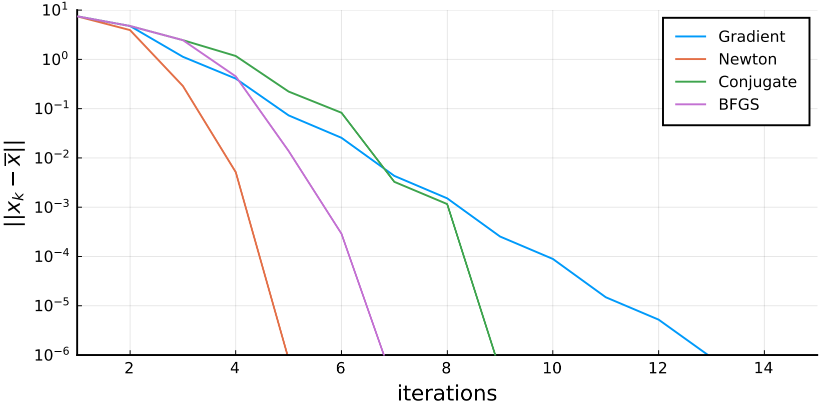
If $\frac{LM\delta}{2} < 1$ and $x_0 \in N_\delta(\overline{x})$, $\{e(x_k)\}$ converges quadratically.

Complexity, convergence and conditioning

Remarks:

1. Parts 1. and 2. are denoted as linear (damped) and quadratic phases (respectively) in the Newton's method progress.
2. Notice that convergence is established without dependency on the ratio of the matrix eigenvalues. Thus, progress is independent of problem scaling. This affine invariance of Newton's method is one of its most important feature.
3. Modern convex analysis using self-concordance provides fairly tight bounds in terms of convergence rate and complexity.

Complexity, convergence and conditioning



Convergence rate for $\epsilon = 10^{-6}$ using exact search

Complexity, convergence and conditioning

Conditioning: the condition number for a symmetric matrix A is

$$\kappa = ||A||_2||A^{-1}||_2 = \frac{\max_{i=1,\dots,n}\{\lambda_i\}}{\min_{i=1,\dots,n}\{\lambda_i\}} = \frac{\overline{\lambda}}{\underline{\lambda}}$$

Large κ implies that numerical errors will be amplified after repeated iterations (i.e., matrix inversions).

Complexity, convergence and conditioning

Conditioning: the condition number for a symmetric matrix A is

$$\kappa = ||A||_2||A^{-1}||_2 = \frac{\max_{i=1,\dots,n}\{\lambda_i\}}{\min_{i=1,\dots,n}\{\lambda_i\}} = \frac{\overline{\lambda}}{\underline{\lambda}}$$

Large κ implies that numerical errors will be amplified after repeated iterations (i.e., matrix inversions).

Remarks:

1. Examining the κ of Hessian matrices is important, since most algorithms rely on first- or second- order approximations.

Complexity, convergence and conditioning

Conditioning: the condition number for a symmetric matrix A is

$$\kappa = ||A||_2||A^{-1}||_2 = \frac{\max_{i=1,\dots,n}\{\lambda_i\}}{\min_{i=1,\dots,n}\{\lambda_i\}} = \frac{\overline{\lambda}}{\underline{\lambda}}$$

Large κ implies that numerical errors will be amplified after repeated iterations (i.e., matrix inversions).

Remarks:

1. Examining the κ of Hessian matrices is important, since most algorithms rely on first- or second- order approximations.
2. Larger κ implies that the level curves of the quadratic approximations are "stretched" ellipsoids, which compromises the convergence of first-order methods (cf. Theorem 4).

Complexity, convergence and conditioning

Conditioning: the condition number for a symmetric matrix A is

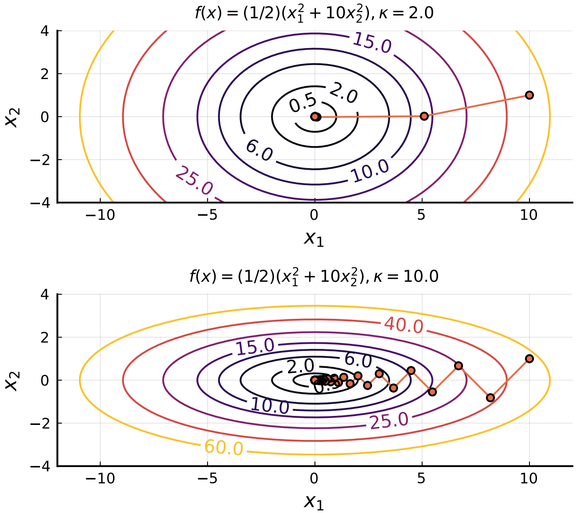
$$\kappa = ||A||_2||A^{-1}||_2 = \frac{\max_{i=1,\dots,n}\{\lambda_i\}}{\min_{i=1,\dots,n}\{\lambda_i\}} = \frac{\overline{\lambda}}{\underline{\lambda}}$$

Large κ implies that numerical errors will be amplified after repeated iterations (i.e., matrix inversions).

Remarks:

1. Examining the κ of Hessian matrices is important, since most algorithms rely on first- or second- order approximations.
2. Larger κ implies that the level curves of the quadratic approximations are "stretched" ellipsoids, which compromises the convergence of first-order methods (cf. Theorem 4).
3. Good values depend on the problem. $\kappa \geq 10^k$ (very) roughly means losing k digits of accuracy per iteration.

Complexity, convergence and conditioning



$f(x) = (1/2)(x_1^2 + 10x_2^2), \kappa = 2.0$

$f(x) = (1/2)(x_1^2 + 10x_2^2), \kappa = 10.0$

The gradient method with exact line search for different κ.

THANK YOU