Università della Svizzera italiana
Institute of Computing CI

Optimization Methods **Spring 2024**

Student: Birindelli Leonardo **Discussed with:** Morales Jeferson

---

Midterm Project **Due date:** Sunday, 12 May 2024, 11:59 PM

Exercise 1

Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(x) = \frac{1}{2}x^T Ax - b^T x$ with A symmetric positive definite. Let x_m be the minimizer of the function f. Le v be an eigenvector of A, and let λ be the associated eigenvalue. Suppose that we use Steepest Descent (SD) method to minimize f and the starting point for the SD algorithm is $x_0 = x_m + v$.

1. Prove that the gradient at x_0 is $\nabla f(x_0) = \lambda v$.

   To begin with the proof, it is required to remark the gradient formula of the quadratic form function which is the following:

   $$\nabla f(x) = Ax - b$$

   Then, considering the starting point $x_0 = x_m + v$, it is possible to write the gradient at that position as :

   $$\begin{aligned} \nabla f(x_0) &= Ax_0 - b && (1)\\ &= A(x_m + v) - b && (2)\\ &= Ax_m + Av - b && (3) \end{aligned}$$

   We know from the hypothesys that x_m is the minimizer of the function f, thus, it means that its gradient must be equal to zero according to the first necessary condition for a minimizer:

   $$\begin{aligned} \nabla f(x_m) &= 0\\ Ax_m - b &= 0\\ Ax_m &= b \end{aligned}$$

   Substituting Ax_m inside the expression (3), we obtain that :

   $$\nabla f(x_0) = b - Av - b = Av$$

In addiction, we know beforehand the definition of eigenvalues and eigenvectors which affirms that :

$$Av = \lambda v \tag{4}$$

Hence it is possible to write that :

$$\nabla f(x_0) = \lambda v$$

Consequently, we have proved that $\nabla f(x_0) = \lambda v$.

2. How many iterations does the SD method take to minimize the function f if we use the optimal step length? Show the computations behind your reasoning.

First of all, we know that the Steepest Descent method uses the negative value of the gradient of function at the given point as the step direction to reach its minimizer. Therefore, that direction, starting from x_0, is equal to the following value as we proved previously:

$$p = -\nabla f(x_0) = -\lambda v$$

Now, we can show that the subsequent iteration x_1 is computed as following:

$$\begin{aligned} x_1 &= x_0 - \alpha\lambda v && (5)\\ &= x_m + v - \alpha\lambda v \quad \text{where } x_0 = x_m + v && (6) \end{aligned}$$

Secondly, we want to know the number of iterations of the gradient method in the case in which it is used the optimal step lenght, thus, we can write that:

$$\alpha_{opt} = \frac{\nabla f(x_i)^T \nabla f(x_i)}{\nabla f(x_i)^T A \nabla f(x_i)}$$

Hence, the optimal step size of the starting point x_0 is:

$$\begin{aligned} \alpha_{opt} &= \frac{\nabla f(x_0)^T \nabla f(x_0)}{\nabla f(x_0)^T A \nabla f(x_0)} && (7)\\ &= \frac{\cancel{\lambda^2} v^T v}{\cancel{\lambda^2} v^T A v} && (8)\\ &= \frac{v^T v}{v^T \lambda v} \qquad Av = \lambda v \ (4) && (9)\\ &= \frac{1}{\lambda} \frac{\cancel{v^T} \cancel{v}}{\cancel{v^T} \cancel{v}} && (10)\\ &= \frac{1}{\lambda} && (11) \end{aligned}$$

Knowing that l'optimal step size is equal to $\frac{1}{\lambda}$ we can semplify the expression 6, thus we achieve that:

$$\begin{aligned} x_1 &= x_m + v - \frac{1}{\cancel{\lambda}} \cancel{\lambda} v \\ x_1 &= x_m + v - v \\ x_1 &= x_m \end{aligned}$$

Then, knowing that x_m is the function minimizer as initial hypothesis, its gradient is equal to zero according to the first necessary condition for minimizer point and given this information we can write the next equation:

$$\nabla f(x_m) = \nabla f(x_1) = 0$$

In conclusion, it is possible to notice that the Steepest Descent method converges in only 1 iteration.
In particular, the single iteration that brings the iterative method to converge, directly depends on the fact that we are using the optimal step lenght in the process, $\alpha_{opt} = \frac{1}{\lambda}$. This information lets the steepest method to find the best step size in order to get closer towards the minimizer of the function. On the other hand, if $\alpha \neq \frac{1}{\lambda}$, the method could have required at least 1 iteration concluding that:

$$x_1 \neq x_m \wedge \nabla f(x_m) \neq \nabla f(x_1) \Longrightarrow f(x_1) > f(x_m) \wedge \nabla f(x_1) > \nabla f(x_m)$$

Exercise 2

Given a starting point $x_0 \in \mathbb{R}^n$ and a set of conjugate directions $\{p_0, p_1, ..., p_{n-1}\}$, we generate the sequence $\{x_k\}$ by setting

$$x_{k+1} = x_k + \alpha_k p_k \tag{12}$$

where

$$\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k} \tag{13}$$

and r_k is the residual, as defined in class.

Consider the following theorem:
Theorem 1 *For any $x_0 \in \mathbb{R}^n$, the sequence $\{x_k\}$ generated by the conjugate gradient direction algorithm converges to the solution x^* of the linear system $Ax = b$ in at most n steps.*
Prove the theorem and explain carefully every step of your reasoning.

Proof:
To begin with, it can be noticed from the hypothesis that the final solution x^* can be written directly in term of the iterative process (12) where the step size is computed according to the formula (13). Thus, x^* is equal to:

$$x^* = \underbrace{\underbrace{\underbrace{x_0 + \alpha_0 p_0}_{x_1} + \alpha_1 p_1}_{x_2} + \alpha_2 p_2}_{x_3} + \cdots + \alpha_{n-1} p_{n-1}; \tag{14}$$

Since, the directions $\{p_0, p_1, p_2, \ldots, p_{n-1}\}$ are linearly independent and orthogonal due to the conjugacy property between each other :

$$< p_i, Ap_j >= 0 \ \forall i \neq j \text{ where } A \text{ is s.p.d} \tag{15}$$

They must be a span of the whole n real space:

$$\mathbb{R}^n = \{p_0, p_1, p_2, \ldots, p_{n-1}\}$$

This affirmation allows to write a generic vector $x \in \mathbb{R}^n$ merely as a linear combination of these directions:

$$x = \sum_{i=0}^{n-1} \gamma_i p_i$$

Especially, this definition can also be estended to the vectors defined by the difference between the final solution x^* and the initial value x_0 given that also these vectors are part of the same real space $\mathbb{R}^n$:

$$\underbrace{e}_{\text{Initial error}} = x^* - x = \sum_{i=0}^{n-1} \sigma_i p_i$$

for some coefficients σ_i.

Now, I want to proof that the coefficients σ_i of the basis p_i match exactly the the optimal step sizes. To achieve this condition, I need to multiply the initial error with the quantity $p_k^T A$, then:

$$\begin{aligned} p_k^T Ae &= p_k^T A(x^* - x_0) && (16)\\ &= p_k^T A \sum_{i=0}^{n-1} \sigma_i p_i && (17)\\ &= \sum_{i=0}^{n-1} p_k^T A \sigma_i p_i && (18)\\ &= \sum_{i=0}^{n-1} \sigma_i p_k^T A p_i && (19)\\ &= \sigma_k p_k^T A p_k \quad \text{Conjugacy property (15)} && (20) \end{aligned}$$

Hence, it is possible to write that :

$$p_k^T A(x^* - x_0) = \sigma_k p_k^T A p_k \implies \sigma_k = \frac{p_k^T A(x^* - x_0)}{p_k^T A p_k}$$

However, it is still required to prove that $A(x^* - x_0) = -r_k$ (where $-r_k = b - Ax_k$) in order to confirm that the coefficients σ_i correspond to the optimal step sizes.
The generic iteration x_k , if it is generated by the iterative procedure (12), then we have that :

$$x_k = \underbrace{\underbrace{\underbrace{x_0 + \alpha_0 p_0}_{x_1} + \alpha_1 p_1}_{x_2} + \alpha_2 p_2}_{x_3} + \cdots + \alpha_{k-1} p_{k-1};$$

Subsequently, we aim to pre-multiply the expression with $p_k^T A$ and using the conjugacy property we obtain that:

$$p_k^T A p_k = p_k^T A x_0$$

Then, substituing the previous expression to the formula σ_k we obtain that:

$$\begin{aligned}
\sigma_k &= \frac{p_k^T A(x^* - x_0)}{p_k^T A p_k} \\
&= \frac{p_k^T A(x^* - x_k)}{p_k^T A p_k} \quad (p_k^T A x_0 = p_k^T A p_k) \\
&= \frac{p_k^T A x^* - p_k^T A x_k}{p_k^T A p_k} \\
&= \frac{p_k^T b - p_k^T A x_k}{p_k^T A p_k} \quad (Ax^* = b) \\
&= \frac{p_k^T (\overbrace{b - A x_k}^{-r_k})}{p_k^T A p_k} \\
&= -\frac{r_k p_k}{p_k^T A p_k} \\
&= \alpha_k
\end{aligned}$$

In conclusion, we prove that the coefficients $\sigma_k = \alpha_k$ and consequently we verified that the algorithm converges in at most n steps:

$$x^* - x_0 = \sum_{i=0}^{n-1} \sigma_i p_i = \sum_{i=0}^{n-1} \alpha_i p_i$$

$$x^* = x_0 + \sum_{i=0}^{n-1} \alpha_i p_i$$

Exercise 3

Consider the linear system $Ax = b$, where the matrix A is a symmetric positive definitive diagonal matrix constructed in four different ways:

$A = \text{diag}([1:10])$;

$A = \text{diag}(ones(1, 10))$;

$A = \text{diag}([1, 1, 1, 3, 4, 5, 5, 5, 10, 10])$;

$A = \text{diag}([1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0])$;

1. How many distinct eigenvalues has each matrix?
   The first and the fourth matrix have exactly 10 unique eigenvalues, the second one has 1 unique eigenvalue and the third one has 5 distinct eigenvalues.

2. Implement the CG method (CG.m).

   The implementation of the CG method is avaiable in the project folder `code` and it is denominated as `CG.m`.

3. Construct a right-hand side $b = rand(10, 1)$ and apply the Conjugate Gradient method to solve the system for each A.
   The code implementation is available in the matlab file denominated as `code/Ex3.m` in the code section commented as "`3.2 & 3.3`".

4. Compute the logarithm energy norm of the error (i.e. $log((x - x^*)TA(x - x^*))$) for each matrix and plot it with respect to the number of iteration.

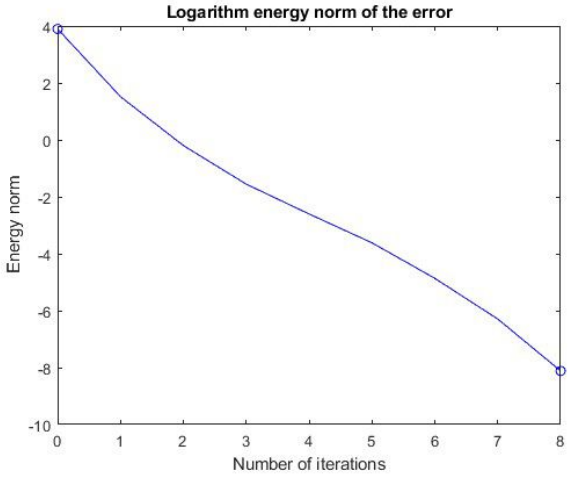

Figure 1: First matrix

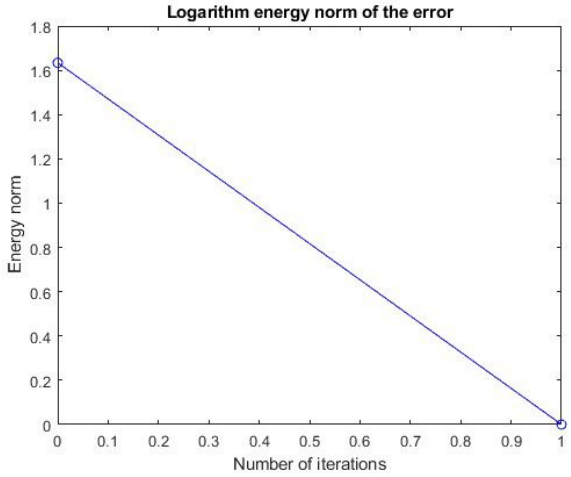

Figure 2: Second matrix

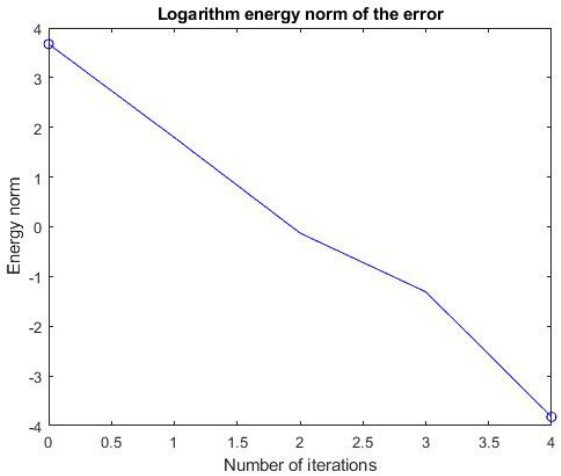

Figure 3: Third matrix

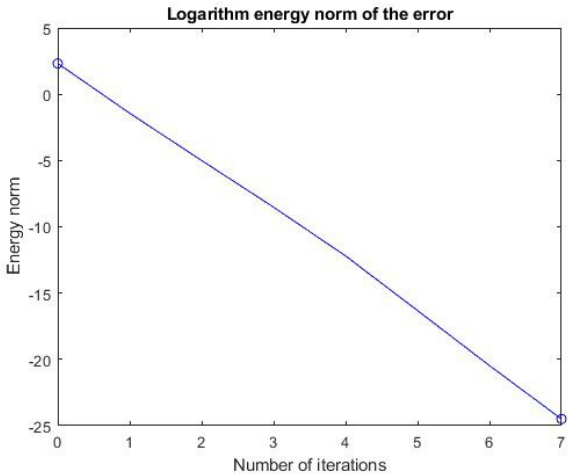

Figure 4: Fourth matrix

Note: the code implementation for generating the plots is available in the matlab file denominated as `code/Ex3.m` in the code section commented as "`3.4`".

5. Comment on the convergence of the method for the different matrices. What can you say observing the number of iterations obtained and the number of clusters of the eigenvalues of the related matrix?

   The A1 and A4 matrixes have equally distributed and distinct eigenvalues and in their domain and this kind of distribution afflicts negatively the performance of the CG method because with a uniform distribution of eigenvalues, the method needs to reduce the error uniformly along all directions, which consequently requires more iterations in the iterative process. On the other hand, the A3 matrix has a skewed distribution of eigenvalues and, moreover, some of them are also repeated in the diagonal of the matrix, those two characteristics allows the method to quickly and firsly reduce the error in the directions corresponding to the large eigenvalues, which can lead to faster convergence.

In conclusion, A2 matrix has only one eigenvalue because has all the elements on the diagonal equal to each other and this situation makes the method converges in only one iteration because the direction of the first residual will already be conjugate to all other directions, and thus the solution can be found immediately.