# An approach for symmetric positive definite matrices

We now assume that the system matrix is symmetric and positive definite (SPD), and discuss a different iterative approach.
Recall the problem we want to solve: given $\underline{b} \in \mathbb{R}^n$, and $A \in \mathbb{R}^n \times \mathbb{R}^n$, we look for $\underline{x}^* \in \mathbb{R}^n$ solution of

$$A\underline{x}^* = \underline{b} \tag{1}$$

Since $A$ is SPD, we can define a scalar product associated with $A:\ (A\underline{x}, \underline{y}) = \underline{y}^T A\underline{x}$. If $A$ is also positive definite, then

$$(A\underline{x}, \underline{x}) > 0 \quad \forall \underline{x} \not\equiv \underline{0}.$$

Then we can introduce the functional $F : \mathbb{R}^n \rightarrow \mathbb{R}$ defined as:

$$F(\underline{v}) := \frac{1}{2}(A\underline{v}, \underline{v}) - (\underline{b}, \underline{v}) \qquad \forall \underline{v} \in \mathbb{R}^n \tag{2}$$

**Theorem 1**

*If $A \in \mathbb{R}^n \times \mathbb{R}^n$ is SPD, problem* (1) *has a unique solution, and is equivalent to the following minimum problem for the functional defined in* (2)*:*

$$\begin{cases} \text{find } \underline{u} \in \mathbb{R}^n \text{ such that} \\ F(\underline{u}) \leq F(\underline{v}) \qquad \forall \underline{v} \in \mathbb{R}^n \end{cases} \tag{3}$$

*(that is,* (3) *has a unique solution $\underline{u} \in \mathbb{R}^n$, and $\underline{u} \equiv \underline{x}^*$).*

## Proof.

Since $A$ is positive definite, problem (1) has a unique solution ($det(A) \neq 0$). Now, $F$ is a quadratic functional (hence, differentiable), and

$$\underline{\nabla} F(\underline{v}) = \begin{bmatrix} \frac{\partial F}{\partial v_1} \\ \frac{\partial F}{\partial v_2} \\ \vdots \\ \frac{\partial F}{\partial v_n} \end{bmatrix} = A\underline{v} - \underline{b} \quad H(F) = A \qquad (H(F) = \text{Hessian matrix})$$

Since $A$ is positive definite, the matrix $H(F)$ has positive eigenvalues (and real because $A$ is symmetric). Hence, $F$ is strictly convex, that is, it has a unique minimum. Let $\underline{u} \in \mathbb{R}^n$ be the point of minimum. As such, it verifies

$$\underline{\nabla} F(\underline{u}) = \underline{0} \quad \longrightarrow \quad A\underline{u} - \underline{b} = \underline{0}.$$

Since the solution of (1) is unique, $\underline{u} \equiv \underline{x}^*$. □

# Descent Methods

Given the equivalence between the linear system (1) and the minimum problem (3), we look for $\underline{x}^*$ as minimum point for $F(\underline{x})$.

Starting from an initial guess $\underline{x}^{(0)}$ (any), we want to construct a sequence $\underline{x}^{(k)}$ converging to $\underline{x}^*$ in the following way:

$$\underline{x}^{(0)}\text{given. Then, for } k = 1, 2, \cdots \text{ set} \quad \underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{p}^{(k)}$$

- $\underline{p}^{(k)}$ are directions of descent,
- $\alpha_k$ are numbers that tell us how much to descent along $\underline{p}^{(k)}$.

They have to be chosen to guarantee descent, that is, to guarantee that

$$F(\underline{x}^{(k+1)}) < F(\underline{x}^{(k)}) \quad \forall k.$$

# Descent methods

The optimal value of $\alpha_k$ can be computed by imposing

$$\frac{\partial}{\partial \alpha} F(x^{(k)} + \alpha p^{(k)}) = 0$$

which guarantees maximum descent of $F$ along the descent direction $p^{(k)}$. Indeed,

$$\begin{aligned} F(x^{(k)} + \alpha p^{(k)}) &= \frac{1}{2}\left(A(x^{(k)} + \alpha p^{(k)}), x^{(k)} + \alpha p^{(k)}\right) - \left(b, x^{(k)} + \alpha p^{(k)}\right) \\ &= \frac{\alpha^2}{2}\left(Ap^{(k)}, p^{(k)}\right) + \alpha\left(Ax^{(k)} - b, p^{(k)}\right) + \left(\frac{1}{2}Ax^{(k)} - b, x^{(k)}\right) \end{aligned}$$

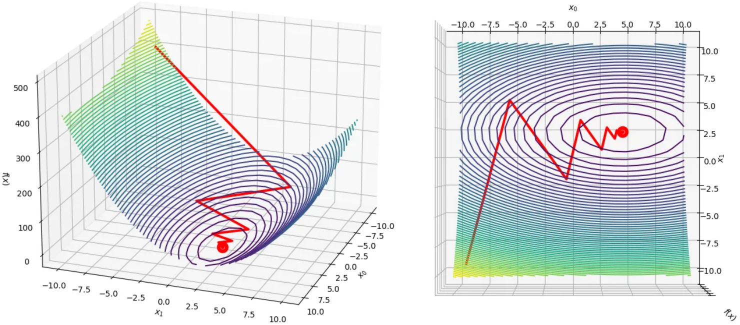
With respect to the variable $\alpha$, this function a quadratic polynomial, its graph is a parabola and its minimum occurs at the stationary point:

$$\frac{\partial}{\partial \alpha} F(x^{(k)} + \alpha p^{(k)}) = \alpha\left(Ap^{(k)}, p^{(k)}\right) + \left(Ax^{(k)} - b, p^{(k)}\right) = 0$$

$$\alpha_k = \text{ optimal } \alpha = \frac{(\underline{b} - A\underline{x}^{(k)}, \underline{p}^{(k)})}{(A\underline{p}^{(k)}, \underline{p}^{(k)})} = \frac{(\underline{r}^{(k)}, \underline{p}^{(k)})}{(A\underline{p}^{(k)}, \underline{p}^{(k)})}$$

# Steepest Descent Method (or Gradient Method)

$$\underline{p}^{(k)} = \text{direction of steepest descent} = -\underline{\nabla} F(\underline{x}^{(k)}) = b - A\underline{x}^{(k)} = \underline{r}^{(k)}$$



# Steepest Descent Method (or Gradient Method)

$$\underline{p}^{(k)} = \text{direction of steepest descent} = -\underline{\nabla} F(\underline{x}^{(k)}) = b - A\underline{x}^{(k)} = \underline{r}^{(k)}$$

This is also seen from the from the Taylor expansion:

$$F(\underline{x}^{(k)} + \alpha \underline{p}^{(k)}) = F(\underline{x}^{(k)}) + (\alpha p^{(k)})^T \nabla F(\underline{x}^{(k)}) + O(\|\alpha p^{(k)}\|^2)$$

Thus $F(\underline{x}^{(k+1)}) \leq F(x)$ if $p^{(k)} = -\nabla F(\underline{x})$ and $\alpha > 0$ is small enough:

$$F(\underline{x}^{(k+1)}) = F(\underline{x}^{(k)}) - \alpha \|\nabla F(\underline{x}^{(k)})\|^2 + O(\alpha^2)$$

# Pseudocode for Steepest Descent Method

## Steepest Descent Method

Input: $A \in \mathbb{R}^{n \times n}$ SPD, $\underline{b} \in \mathbb{R}^n$, $\underline{x}^{(0)} \in \mathbb{R}^n$, $tol \in \mathbb{R}^+$, maxiter $\in \mathbb{N}$

$\underline{r}^{(0)} = \underline{b} - A\underline{x}^{(0)}$

**for** $k = 1, 2, \ldots,$ maxiter:

$\quad \underline{y} = A\underline{r}^{(k-1)}$

$\quad \alpha_{k-1} = \left(\underline{r}^{(k-1)}, \underline{r}^{(k-1)}\right) / \left(\underline{y}, \underline{r}^{(k-1)}\right)$

$\quad \underline{x}^{(k)} = \underline{x}^{(k-1)} + \alpha_{k-1}\underline{r}^{(k-1)}$

$\quad \underline{r}^{(k)} = \underline{b} - A\underline{x}^{(k)} = \underline{r}^{(k-1)} - \alpha_{k-1}\underline{y}$

$\quad$ If Stopping criteria are satisfied exit the loop

**end**

Output: $\underline{x}^{(k)}$

Like for all iterative methods, the dominant computational cost is given by the matrix-vector product with $A$. Hence the cost is roughly $2n^2$ FLOPs per iteration.

# Convergence of Steepest Descent Method

**Theorem 2**

*The Steepest Descent method converges for all initial guess $\underline{x}^{(0)}$. Moreover it holds:*

$$\left\|x - x^{(k)}\right\|_A \leq \left(\frac{\kappa_2(A) - 1}{\kappa_2(A) + 1}\right)^k \left\|x - x^{(0)}\right\|_A$$

*where $\|v\|_A = \sqrt{v^T A v}$ is the $A-$norm.*

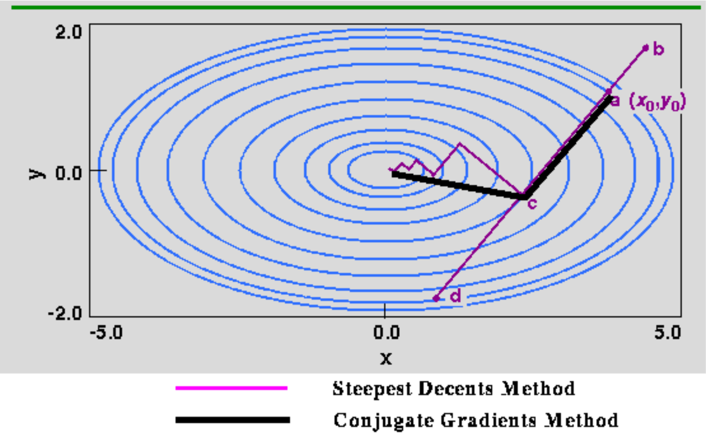
Convergence is guaranteed, but can be very slow if $A$ is ill-conditioned.

# Summary and extensions of gradient methods...

We have our functional $F(\underline{v}) := \frac{1}{2}(A\underline{v}, \underline{v}) - (\underline{b}, \underline{v})$ to minimize and use $\underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{p}^{(k)}$, where $p^{(k)} = -\underline{\nabla} F(\underline{x}^{(k)}) = b - A\underline{x}^{(k)}$. Possible alternatives are:

- to simplify the calculation of $p^{(k)} = -\underline{\nabla} F(\underline{x}^{(k)})$, e.g. in the **stochastic gradient descent** method, used in machine learning: we save time per each iteration at the expenses of an increased number of iterations to reach a given accuracy;
- to find better descent directions $p^{(k)}$, such that the convergence at a given tolerance requires less iterations, as in the **conjugate gradient** method

# Steepest descents vs. Conjugate Gradient



# Conjugate Gradient method

with $\underline{p}^{(0)} = -\underline{\nabla} F(\underline{x}^{(0)})$, at each iteration $k$ take $\underline{p}^{(k)}$ in the plane $span\{\underline{r}^{(k)}, \underline{p}^{(k-1)}\}$, that is:

$$\underline{p}^{(k)} = \underline{r}^{(k)} - \beta_k \underline{p}^{(k-1)}$$

where $\beta_k$ is chosen so that $\underline{p}^{(k)}$ is $A-$orthogonal to $\underline{p}^{(k-1)}$, i.e. $\left(\underline{p}^{(k)}\right)^T A\underline{p}^{(k-1)} = 0$ (orthogonal in the scalar product associated with $A$). It can be proven that

$$\left(\underline{p}^{(k)}\right)^T A\underline{p}^{(j)} = 0, \qquad j = 1, \ldots, k-1.$$

This approach is faster than the steepest descent. Actually, the method converges in less than $n$ iterations ($n$=dimension of the system), so it can be considered a direct method.

**Matlab function**: $x =$**pcg**$(A, b, ...)$

# Pseudocode for Conjugate Gradient Method

## Conjugate Gradient Method

Input: $A \in \mathbb{R}^{n\times n}$ SPD, $\underline{b} \in \mathbb{R}^n$, $\underline{x}^{(0)} \in \mathbb{R}^n$, $tol \in \mathbb{R}^+$, maxiter $\in \mathbb{N}$

$\underline{r}^{(0)} = \underline{b} - A\underline{x}^{(0)}$

$\underline{p}^{(0)} = \underline{r}^{(0)}$

**for** $k = 1, 2, \ldots,$ maxiter:

  $\underline{y} = A\underline{p}^{(k-1)}$

  $\alpha_{k-1} = \left(\underline{p}^{(k-1)}, \underline{r}^{(k-1)}\right) / \left(\underline{y}, \underline{p}^{(k-1)}\right)$

  $\underline{x}^{(k)} = \underline{x}^{(k-1)} + \alpha_{k-1}\underline{p}^{(k-1)}$

  $\underline{r}^{(k)} = \underline{b} - A\underline{x}^{(k)} = \underline{r}^{(k-1)} - \alpha_{k-1}\underline{y}$

  $\beta_{k-1} = \left(\underline{y}, \underline{r}^{(k)}\right) / \left(\underline{y}, \underline{p}^{(k-1)}\right)$

  $\underline{p}^{(k)} = \underline{r}^{(k)} - \beta_{k-1}\underline{p}^{(k-1)}$

  If Stopping criteria are satisfied exit the loop

**end**

Output: $\underline{x}^{(k)}$