Università della Svizzera italiana
Institute of Computing CI

Numerical Computing 2023

Lecture 8: Introduction to the conjugate-gradient method

Lisa Gaedke-Merzhäuser, Dr. Dimosthenis Pasadakis, Dr. Edoardo Vecchi

Content

- Motivation $\mathbf{Ax} = \mathbf{b}$
- Gradient-based Methods
  - Solve equivalent minimization problem to $\mathbf{Ax} = \mathbf{b}$
  - Method of Steepest Descent
  - Conjugate-Gradient Method
- Outlook: Preconditioning

Direct Methods

Let us consider the solution of a system of linear equations $\mathbf{Ax} = \mathbf{b}$, with **nonsingular** coefficient matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, constant vector $\mathbf{b} \in \mathbb{R}^n$ and (unknown) solution vector $\mathbf{x} \in \mathbb{R}^n$.

Direct methods compute the exact solution in n steps (in absence of roundoff error):

- Cost for classical Gaussian Elimination: $\mathcal{O}(n^3)$
- Cost for sparse matrices: $\mathcal{O}(n^{1.5})$ or $\mathcal{O}(n^2)$
- High memory consumption due to additional fill-in elements

Problem: Complexity of Gaussian Elimination ("direct method") for LARGE (sparse) linear systems of equations is too high.

Iterative Methods

Potential solution: Use iterative methods.

Iterative methods use an (arbitrary) initial guess ("starting point") to compute an approximate solution:

- Cost for conjugate-gradient algorithms: between $\mathcal{O}(n)$ and $\mathcal{O}(n^{3/2})$ (for an approximation using a fixed accuracy, e.g., 10^{-6})
- No need for additional memory
- Convergence after a few iterations (this depends, of course, also on A).

However, iterative methods are very often less robust and not as general as direct methods.

Sketch of an iterative method

Structure of the algorithm:

Start: $m = 0, \quad \mathbf{x}^{(m)} =$ Initial value

Iterate until error estimate $< \epsilon$:

Find a new solution $\mathbf{x}^{(m+1)}$

Compute new error estimate

- Is the new iterate always better?
- How can we get an error estimate?

Definition of Error and Residual

The *error* in step m is the deviation of $\mathbf{x}^{(m)}$ from the exact solution $\mathbf{x}$:

$$\mathbf{e}^{(m)} = \mathbf{x} - \mathbf{x}^{(m)} = A^{-1}\mathbf{b} - \mathbf{x}^{(m)}$$

Unfortunately we do not know the error during the iterations (otherwise we would know also the exact solution, thus defying the purpose of using iterative solvers)

The *residual* provides us a measure for the real error. It is relatively easy to compute:

$$\mathbf{r}^{(m)} = \mathbf{b} - A\mathbf{x}^{(m)} = A\mathbf{e}^{(m)}$$

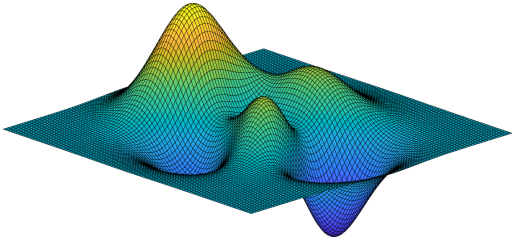
The fact that the residual is equivalent the A-transformed error ($A\mathbf{e}^{(m)}$) will be used later. Its *residual norm* is defined as $\|\mathbf{r}^{(m)}\|$, whereas the *relative residual norm* is $\frac{\|\mathbf{r}^{(m)}\|}{\|\mathbf{b}\|}$.

Notation

- $\langle \cdot, \cdot \rangle$ is the scalar product: $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^T \mathbf{b} = \sum_{i=1}^{n} a_i b_i$
- $\mathbf{x} := A^{-1}\mathbf{b}$ denotes the exact solution
- $\mathbf{x}^{(m)}$ is the approximation in the m-th iteration

Gradient-based methods

www.myswissalps.com



A class of first order methods utilized for finding the nearest local minimum of a function, by following the direction of its negative gradient at the current point (Cauchy 1847).

Minimization problem - 1D Case

Basic idea: Instead of $a \cdot x = b$ solve an equivalent minimization problem

$$f(x) := \tfrac{1}{2} a \cdot x^2 - b \cdot x \tag{1}$$

Precondition: Let a be *symmetric positive definite (spd), i.e. in this case* $a > 0$.
If we take the derivative of (1), it will lead to:

$$\begin{aligned} f'(x) &= 2 \cdot \tfrac{1}{2} x - b \\ &= ax - b \end{aligned} \tag{2}$$

Setting the equation to zero will lead to the original problem $a \cdot x = b$.

Minimization problem

Basic idea: Instead of $A\mathbf{x} = \mathbf{b}$ solve an equivalent minimization problem

$$f(\mathbf{x}) := \tfrac{1}{2}\langle A\mathbf{x}, \mathbf{x}\rangle - \langle \mathbf{b}, \mathbf{x}\rangle \tag{3}$$

Precondition: Let A be a *symmetric positive definite (spd) matrix:*

$$\langle A\mathbf{x}, \mathbf{x}\rangle > 0, \quad \text{if } \mathbf{x} \neq \text{zero vector} \tag{4}$$

If we take the derivative of (3), it will lead to:

$$\begin{aligned} f'(\mathbf{x}) &= \tfrac{1}{2}A^T\mathbf{x} + \tfrac{1}{2}A\mathbf{x} - \mathbf{b} \\ &= A\mathbf{x} - \mathbf{b} \qquad \Big| \text{Condition: } A^T = A \end{aligned} \tag{5}$$

Setting the equation to zero will lead to the original problem $A\mathbf{x} = \mathbf{b}$.

Minimization problem, Example

$$\mathbf{A} = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 2 \\ -8 \end{bmatrix}$$

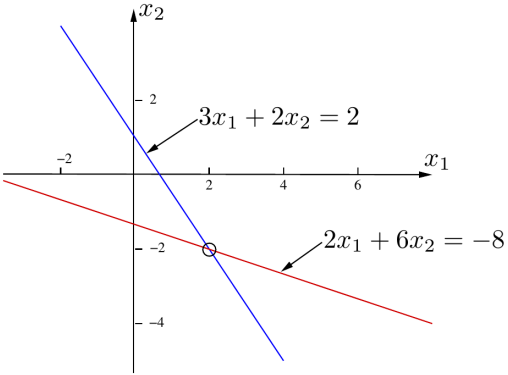


Figure: Linear equations with solution at the intersection of both lines.

Minimization problem, Example

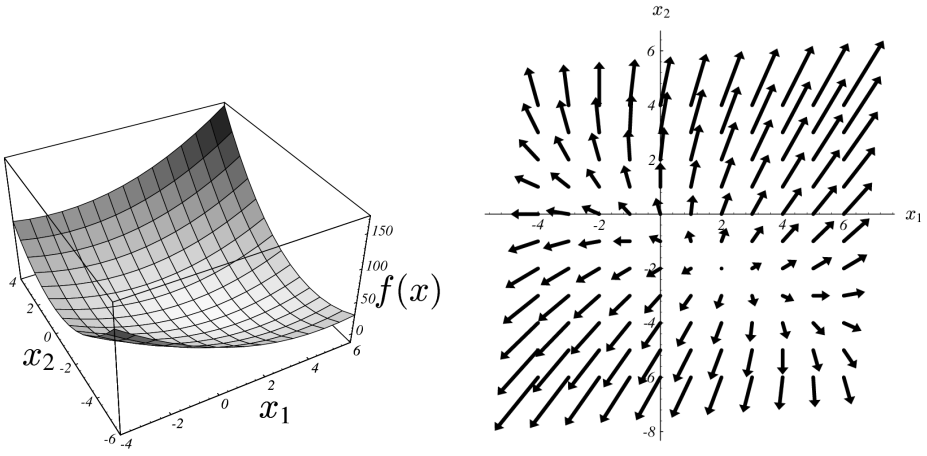


Figure: Quadratic Form $f(x)$ and gradients $f'(x)$

Method of steepest descent

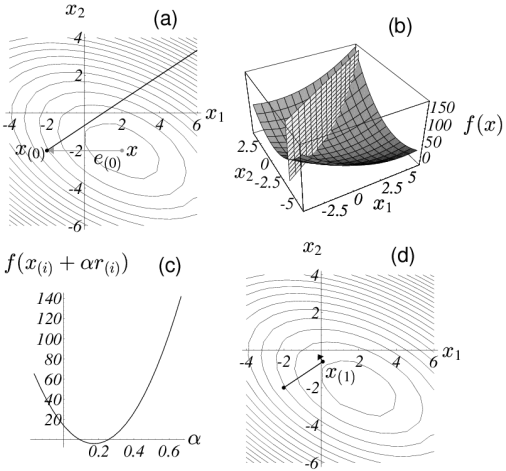
Basic idea: Take one step in the direction of the negative gradients at the point $\mathbf{x}^{(m)}$, i.e. in the direction of $-f'(\mathbf{x}^{(m)})$.

- Use a step length, until f is minimal along the search direction
- From equation (5) it follows, that $-f'(\mathbf{x}^{(m)}) = \mathbf{r}^{(m)}$

$$
\begin{aligned}
&\underline{\text{Start:}}\ \mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)} \\
&\textbf{until } \|\mathbf{r}^{(m)}\| < \epsilon\text{:} \\
&\quad \text{Find } \alpha\text{, such that } f(\mathbf{x}^{(m)} + \alpha\mathbf{r}^{(m)}) \text{ minimal} \\
&\quad \mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} + \alpha\mathbf{r}^{(m)} \\
&\quad \mathbf{r}^{(m+1)} = \mathbf{b} - A\mathbf{x}^{(m+1)}
\end{aligned}
$$

- The residual is a measure for both the *error* and *the search direction!*

Method of steepest descent



Method of steepest descent

Minimize f along the search vector $\mathbf{p}^{(m)}$:

Approach:

$$f(\mathbf{x}^{(m+1)}) = f(\mathbf{x}^{(m)} + \alpha \mathbf{p}^{(m)}) \stackrel{!}{=} \min \quad \Longleftrightarrow \quad \frac{d}{d\alpha} f(\mathbf{x}^{(m+1)}) = 0 \tag{6}$$

with the chain rule

$$\begin{aligned} \frac{d}{d\alpha} f(\mathbf{x}^{(m+1)}) &= \langle f'(\mathbf{x}^{(m+1)}), \frac{d}{d\alpha}(\mathbf{x}^{(m)} + \alpha \mathbf{p}^{(m)}) \rangle \\ &= \langle f'(\mathbf{x}^{(m+1)}), \mathbf{p}^{(m)} \rangle \\ &= \langle \mathbf{r}^{(m+1)}, \mathbf{p}^{(m)} \rangle \end{aligned}$$

for steepest descent we will use $\mathbf{p}^{(m)} = \mathbf{r}^{(m)}$. The more general (and interesting) case with $\mathbf{p}^{(m)} \neq \mathbf{r}^{(m)}$ will be discussed later.

Method of steepest descent

We search for an α, such that $\mathbf{r}^{(m+1)} \perp \mathbf{p}^{(m)}$.

Method of steepest descent

We search for an α, such that $\mathbf{r}^{(m+1)} \perp \mathbf{p}^{(m)}$.

Rearranging for α:

$$
\begin{aligned}
\langle \mathbf{r}^{(m+1)}, \mathbf{p}^{(m)} \rangle &= 0 \\
\langle b - A\mathbf{x}^{(m+1)}, \mathbf{p}^{(m)} \rangle &= 0 \\
\langle b - A(\mathbf{x}^{(m)} + \alpha \mathbf{p}^{(m)}), \mathbf{p}^{(m)} \rangle &= 0 \\
\langle b - A\mathbf{x}^{(m)}, \mathbf{p}^{(m)} \rangle - \alpha \langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle &= 0 \\
\langle b - A\mathbf{x}^{(m)}, \mathbf{p}^{(m)} \rangle &= \alpha \langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle \\
\langle \mathbf{r}^{(m)}, \mathbf{p}^{(m)} \rangle &= \alpha \langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle \\
\alpha &= \frac{\langle \mathbf{r}^{(m)}, \mathbf{p}^{(m)} \rangle}{\langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle}
\end{aligned}
$$

Method of steepest descent

- The final algorithm will be:

$$\underline{\text{Start:}}\ \mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$$

$$\textbf{until } \|\mathbf{r}^{(m)}\| < \epsilon\text{:}$$

$$\alpha^{(m)} = \frac{\langle \mathbf{r}^{(m)}, \mathbf{r}^{(m)} \rangle}{\langle A\mathbf{r}^{(m)}, \mathbf{r}^{(m)} \rangle}$$

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} + \alpha^{(m)}\mathbf{r}^{(m)}$$

$$\mathbf{r}^{(m+1)} = \mathbf{r}^{(m)} - \alpha^{(m)} A\mathbf{r}^{(m)}$$

- We avoid the matrix-vector product $A\mathbf{x}^{(m)}$ for the computation of $\mathbf{r}^{(m+1)}$ since $\mathbf{r}^{(m+1)} = \mathbf{r}^{(m)} - \alpha^{(m)} A\mathbf{r}^{(m)}$

Method of steepest descent

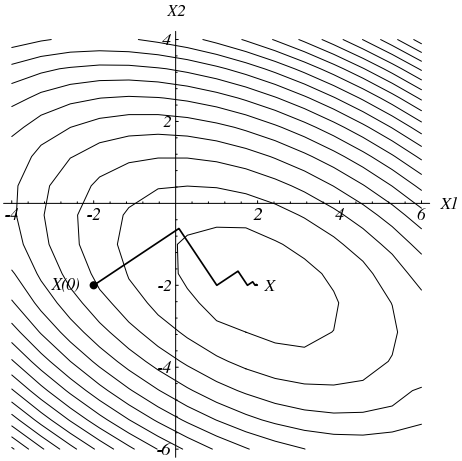


Figure: Method of steepest descent with $\mathbf{x}^{(0)} = (-2 \ -2)^T$

Convergence of steepest descent

Consider the functional $\mathbf{f}(x) = \frac{1}{2} < A\mathbf{x}, \mathbf{x} > - < \mathbf{b}, \mathbf{x} >$, which we would like to minimize to compute the solution of $A\mathbf{x} = \mathbf{b}$.

Example: $n = 2$

A has 2 orthonormal eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$ with associated eigenvalues λ_1 and λ_2. We can write each vector $\mathbf{x}$ as a linear combination of eigenvectors $\mathbf{x} = a_1 \cdot \mathbf{v}_1 + a_2 \cdot \mathbf{v}_2, \quad a_1, a_2 \in \mathbb{R}$.

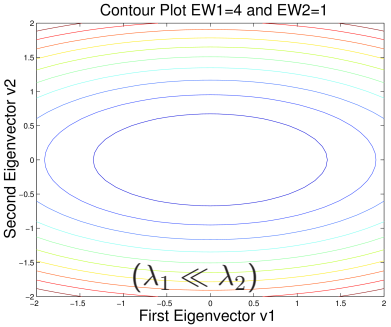
The following equation holds ($\mathbf{b} = 0$):

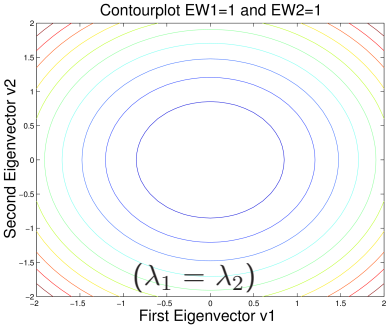
$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} = \frac{1}{2}(a_1\mathbf{v}_1 + a_2\mathbf{v}_2)^T(a_1\lambda_1\mathbf{v}_1 + a_2\lambda_2\mathbf{v}_2) \tag{7}$$

$$= \frac{1}{2}(a_1^2\lambda_1 + a_2^2\lambda_2) \tag{8}$$

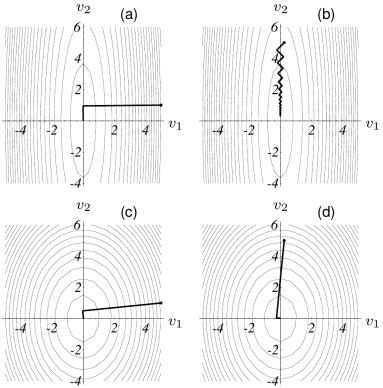
Convergence of steepest descent

Contour lines of the functional:



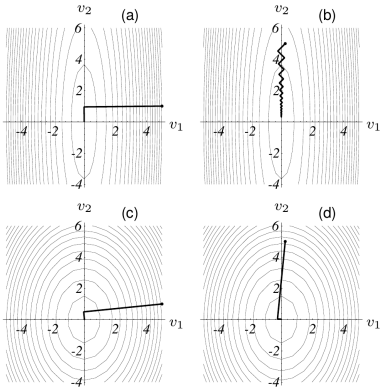


Impact on the convergence



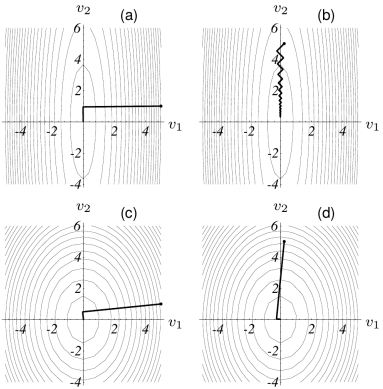
- (a) Large condition number, good starting point $\longrightarrow$ by accident fast convergence using steepest descent

Impact on the convergence



- (a) Large condition number, good starting point $\longrightarrow$ by accident fast convergence using steepest descent (b) Large condition number, bad starting point $\longrightarrow$ very slow convergence

Impact on the convergence



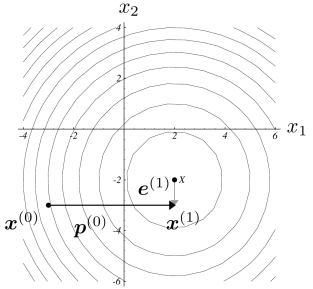
- (a) Large condition number, good starting point $\longrightarrow$ by accident fast convergence using steepest descent (b) Large condition number, bad starting point $\longrightarrow$ very slow convergence (c-d) Small condition number $\longrightarrow$ good convergence independent of the starting vector $\longrightarrow$ Preconditioning

«Conjugate Gradient Method»

Conjugate Gradient Search Directions

Basic idea: In each iteration we take one optimal search step.

- *Approach:* We define n orthogonal search directions $\mathbf{p}^{(0)}$ until $\mathbf{p}^{(n-1)}$. Use maximally n steps.
- The minimization criteria is the orthogonality of $\mathbf{e}^{(m+1)}$, with: $\mathbf{p}^{(m)} \perp \mathbf{e}^{(m+1)}$



Idea

- Using this step we eliminate the error component of this particular search direction
- If $\mathbf{p}^{(m)} \perp \mathbf{e}^{(m+1)}$, then the scalar-product must be $= 0$:

$$\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m+1)} \rangle = 0 \tag{9}$$

$$\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m)} + \alpha^{(m)} \mathbf{p}^{(m)} \rangle = 0 \tag{10}$$

$$\alpha^{(m)} = -\frac{\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle} \tag{11}$$

Idea

- Using this step we eliminate the error component of this particular search direction
- If $\mathbf{p}^{(m)} \perp \mathbf{e}^{(m+1)}$, then the scalar-product must be $= 0$:

$$\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m+1)} \rangle = 0 \tag{12}$$

$$\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m)} + \alpha^{(m)} \mathbf{p}^{(m)} \rangle = 0 \tag{13}$$

$$\alpha^{(m)} = -\frac{\langle \mathbf{p}^{(m)}, \mathbf{e}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle} \tag{14}$$

- This is not really helpful since we do not know $\mathbf{e}^{(m)}$!

Conjugate Search Direction

One solution is, instead of orthogonal direction, we will use *A-orthogonal* or *conjugate* vectors.

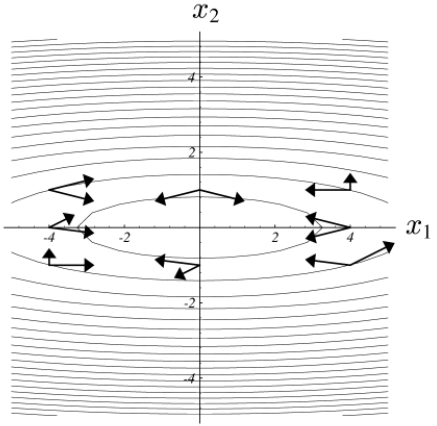
Definition: Two vectors $\mathbf{a}, \mathbf{b}$ are *A-orthogonal* or *conjugate* if and only if:

$$\mathbf{a}^T A\mathbf{b} = \langle \mathbf{a}, A\mathbf{b} \rangle = 0 \quad \Longleftrightarrow \quad \mathbf{a} \perp_A \mathbf{b}$$

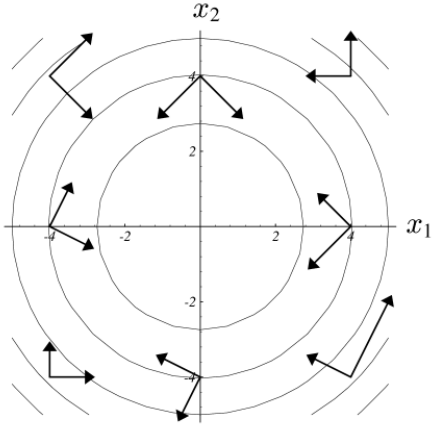
Let $\mathbf{p}^{(m)} \perp_A \mathbf{e}^{(m+1)}$ and $\mathbf{r}^{(m)} = A\mathbf{e}^{(m)}$, then

$$\begin{aligned}
\langle \mathbf{p}^{(m)}, A\mathbf{e}^{(m+1)} \rangle &= 0 && (15)\\
\langle \mathbf{p}^{(m)}, A\mathbf{e}^{(m)} + \alpha^{(m)} A\mathbf{p}^{(m)} \rangle &= 0 && (16)\\
\alpha^{(m)} &= -\frac{\langle \mathbf{p}^{(m)}, A\mathbf{e}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, A\mathbf{p}^{(m)} \rangle} && (17)\\
&= \frac{\langle \mathbf{p}^{(m)}, \mathbf{r}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, A\mathbf{p}^{(m)} \rangle} && (18)
\end{aligned}$$

Conjugate Search Direction

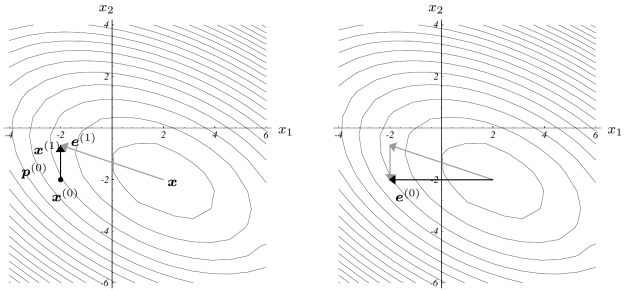


A-Conjugate vectors



rectified vectors

Elimination of A-orthogonal components



(a) (b)

(a) $\mathbf{x}^{(1)}$ is computed such that $\mathbf{e}^{(1)}$ is A-orthogonal to $\mathbf{p}^{(0)}$.
(b) $\mathbf{e}^{(m)}$ can be expressed as a sum of A-orthogonal components (gray arrows). Each iteration eliminates such a component.

Conjugate Directions

Condition: We would like to minimize $f'(x^{(m)} + \alpha p^{(m)}) \stackrel{!}{=} min$ (see (16))

The algorithm must satisfy the following iteration:

Start: $\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$

For all $m = 1, \ldots, n$:

Select search vector $\mathbf{p}^{(m)}$ which is conjugate to all previously computed $\mathbf{p}^{(l)}, l < m$

$$\alpha^{(m)} = \frac{\langle \mathbf{r}^{(m)}, \mathbf{p}^{(m)} \rangle}{\langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle}$$

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} + \alpha^{(m)} \mathbf{p}^{(m)}$$

$$\mathbf{r}^{(m+1)} = \mathbf{r}^{(m)} - \alpha^{(m)} A\mathbf{p}^{(m)}$$

Question: How to find n A-orthogonal search vectors?

Conjugate Gradient Method

Basic idea: Use the residual to construct the next conjugate search direction, where

$$\mathbf{p}^{(m+1)} = \mathbf{r}^{(m+1)} + \underbrace{\beta^{(m+1)}}_{\text{search for}} \mathbf{p}^{(m)} \tag{19}$$

and $\underbrace{\langle \mathbf{p}^{(m+1)}, A\mathbf{p}^{(m)} \rangle = 0}_{\text{A-orthogonal}}$ we obtain:

$$\begin{aligned} 0 &= \langle \mathbf{p}^{(m+1)}, A\mathbf{p}^{(m)} \rangle \\ &= \langle \underbrace{\mathbf{r}^{(m+1)} + \beta^{(m+1)}\mathbf{p}^{(m)}}_{(19)\text{ used}}, A\mathbf{p}^{(m)} \rangle \\ &= \langle \mathbf{r}^{(m+1)}, A\mathbf{p}^{(m)} \rangle + \beta^{(m+1)} \langle \mathbf{p}^{(m)}, A\mathbf{p}^{(m)} \rangle \\ \beta^{(m+1)} &= -\frac{\langle \mathbf{r}^{(m+1)}, A\mathbf{p}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, A\mathbf{p}^{(m)} \rangle} \end{aligned}$$

Conjugate Gradient Method

- the *initial-search vector* is $\mathbf{r}^{(0)}$ (as in steepest descent)
- similar to the steepest descent we can eliminate the additional matrix-vector product (exercise):

$$\beta^{(m+1)} = -\frac{\langle \mathbf{r}^{(m+1)}, \mathbf{A}\mathbf{p}^{(m)} \rangle}{\langle \mathbf{p}^{(m)}, \mathbf{A}\mathbf{p}^{(m)} \rangle} = \frac{\langle \mathbf{r}^{(m+1)}, \mathbf{r}^{(m+1)} \rangle}{\langle \mathbf{r}^{(m)}, \mathbf{r}^{(m)} \rangle}$$

- The method how to construct the n conjugate vectors is the *Gram-Schmidt process*.
- The iterative construction of the search space is of the form

$$U_m := \operatorname{span}\{\mathbf{p}^{(0)}, A\mathbf{p}^{(0)}, A^2\mathbf{p}^{(0)}, \ldots, A^n\mathbf{p}^{(0)}\}$$

This space is called *Krylov-Subspace*.

Conjugate Gradient Method

Start: $\mathbf{r}^{(0)} := \mathbf{b} - A\mathbf{x}^{(0)}$ with $\mathbf{x}^{(0)}$ arbitrary

$\mathbf{p}^{(0)} := \mathbf{r}^{(0)}$

for all $m = 0, \ldots, n-1$:

$$\alpha^{(m)} = \frac{\langle \mathbf{r}^{(m)}, \mathbf{p}^{(m)} \rangle}{\langle A\mathbf{p}^{(m)}, \mathbf{p}^{(m)} \rangle}$$

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} + \alpha^{(m)} \mathbf{p}^{(m)}$$

$$\mathbf{r}^{(m+1)} = \mathbf{r}^{(m)} - \alpha^{(m)} A\mathbf{p}^{(m)}$$

$$\beta^{(m+1)} = \frac{\langle \mathbf{r}^{(m+1)}, \mathbf{r}^{(m+1)} \rangle}{\langle \mathbf{r}^{(m)}, \mathbf{r}^{(m)} \rangle}$$

$$\mathbf{p}^{(m+1)} = \mathbf{r}^{(m+1)} + \beta^{(m+1)} \mathbf{p}^{(m)}$$

Hestenes & Stiefel 1952.

Conjugate Gradient Method

Elements of the algorithm:

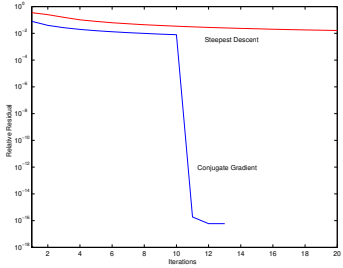
- $\mathbf{x}^{(m)}$: Exact solution.
- $\mathbf{r}^{(m)}$: Residual of the exact solution.
- $\mathbf{p}^{(m)}$: Search direction.
- $\alpha^{(m)}$: Optimal step length (factor) in the search direction $\mathbf{p}^{(m)}$ for next iterate $\mathbf{x}^{(m+1)}$.
- $\beta^{(m+1)}$: Factor for $\mathbf{p}^{(m)}$ to compute from $\mathbf{p}^{(m)}$ and $\mathbf{r}^{(m+1)}$ a new search direction $\mathbf{p}^{(m+1)}$ which is A-orthogonal to $\mathbf{p}^{(m)}$.

CG – Theory and Praxis

- The exact solution will be computed after n iterations
- As a result CG is actually a direct method
- In praxis we usually need much less iterations since we only need to compute an approximate solution.

Test example

Convergence of steepest descent and conjugate gradients $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} = \texttt{tridiag}(-1, 2, -1) \in \mathbb{R}^{20\times 20}$ and $\mathbf{b}$ is a random vector.



After n iterations, where $n = \texttt{dim}(\mathbf{A})$ the CG-methods converges against the exact solution.

Other popular Krylov Subspace Methods

Book: «Templates for the Solution of Linear System»
`http://www.netlib.org/linalg/html_templates/Templates.html`

Some popular iterative Krylov Subspace methods:

Name		Condition
MinRES	Minimal Residual	$A = A^T$, A indef.
CG	Conjugate Gradient	$A = A^T$, A s.p.d
QMR	Quasi-Minimal Residual	unsymmetric
BICGSTAB	Biconjugate Gradient Stabilized	unsymmetric
CGS	Conjugate Gradient Square	unsymmetric
GMRES	Generalized Minimal Residual	unsymmetric

« Preconditioning »

Preconditioning

Problem: The rate of convergence of CG heavily depends on the condition number $\kappa(A)$. More precisely we have

$$\left\|x^{(m)} - x^*\right\|_A \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^m \|x_0 - x^*\|_A .$$

where

$$\kappa(A) = \|A\|_2\|A^{-1}\|_2 = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}.$$

Thus, if $\kappa(A) \gg 1$, slow convergence.

Preconditioning

Basic idea: Improve the conditioning number of A through a multiplication with a "preconditioner" M: $\kappa(M^{-1}A) \ll \kappa(A)$

- Solve equivalent problem: $M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$
- M should be easy to invert
- In order to use the CG method, the resulting matrix $M^{-1}A$ must be symmetric positive definite
- We can split M in the following form $EE^T = M$, so that we can transform the problem $A\mathbf{x} = \mathbf{b}$ into

$$E^{-1}AE^{-T}\hat{\mathbf{x}} = E^{-1}\mathbf{b}, \quad \hat{\mathbf{x}} = E^T\mathbf{x} \tag{20}$$

  where the matrix $E^{-1}AE^{-T}$ is *spd*.

Preconditioning

Here are a few method

- *Diagonal preconditioning*. Choose: $M = \mathrm{diag}(A)$
- *Incomplete LU-Decomposition*. Choose: $M = LU = A - R$.
  - Important that the Fill-In and the time for the factorization will be small.

$\rightarrow$ In Project 4 you will see CG and different preconditioners in practice!

Literature

- Jonathan R. Shewchuk, *An Introduction to the Conjugate Gradient method without the Agonizing Pain*, 1994. `http://www-2.cs.cmu.edu/~jrs/jrspapers.html#cg`
- Book: «Templates for the Solution of Linear System» `http://www.netlib.org/linalg/html_templates/Templates.html`
- James W. Demmel, *Applied Numerical Linear Algebra*, Siam 1997

QUESTIONS?

Jacobi Iteration Method

Basic idea: Solve the equation for every component x_j using the current approximation of $\mathbf{x}$:

- For each individual elements of $\mathbf{x}$ we will use the following iterations method:

$$x_j^{(m+1)} = \frac{1}{a_{jj}}(b_j - \sum_{k \neq j} a_{jk} x_k^{(m)}) \tag{21}$$

- We will only use elements from the previous iteration.
- Note that the order in which the equations are examined is irrelevant, since the Jacobi method treats them independently.

Jacobi Iteration Method: Example for $j = 1$

$$\begin{pmatrix} 1.5 & 0.5 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} x_1^{(m)} \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$$

$$x_1^{(m+1)} = \frac{1}{1.5}(2 - 0.5 * 3) = \frac{1}{3}$$

$$\begin{pmatrix} 1.5 & 0.5 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} \frac{1}{3} \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$$

Jacobi Iteration Method

Matrix formulation: Split A in:

$$A = D + E$$

where D are all diagonal elements of A and E are the off-diagonal elements of A.

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ D\mathbf{x} &= -E\mathbf{x} + \mathbf{b} \\ \mathbf{x} &= \underbrace{-D^{-1}E}_{R_J}\mathbf{x} + \underbrace{D^{-1}\mathbf{b}}_{c_J} \\ \mathbf{x} &= R_J\mathbf{x} + \mathbf{c}_J, \end{aligned} \tag{22}$$

Renaming results in:

$$\mathbf{x}^{(m+1)} = R_J\mathbf{x}^{(m)} + \mathbf{c}_J. \tag{23}$$

Convergence of the Jacobi iteration

The fundamental convergence results can be proven by using an eigenvalue analyses of R.

We need to remember the following two definitions:

- The *Spectral Radius* of matrix A is:

$$\rho(A) = max\{|\lambda| : \lambda \ eigenvalue \ of \ A\}. \tag{24}$$

- The *Spectral condition number* is given by:

$$\kappa = \lambda_{max}/\lambda_{min}. \tag{25}$$

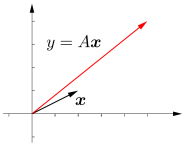
Excursus: Eigenvectors and eigenvalues

- Let A a symmetric, real $n \times n$ matrix. The vector $\mathbf{x}$ *is called an eigenvector to the scalar eigenvalue* λ, if the following equation is satisfied:

$$A\mathbf{x} = \lambda\mathbf{x} \tag{26}$$

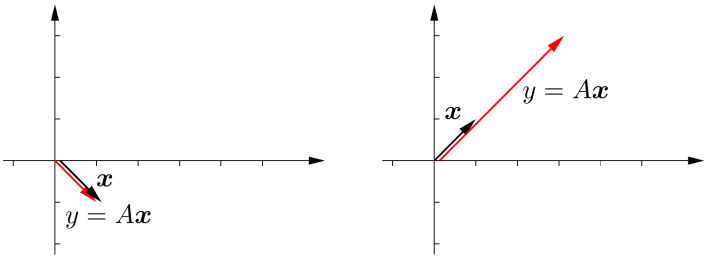
- Geometric interpretation of the matrix-vector multiplication: The multiplication of a matrix A with a vector $\mathbf{x}$ is a linear operator, which consists of a scaling and a rotation of a vector $\mathbf{x}$, e.g.:

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \quad \mathbf{y} = A\mathbf{x} = \begin{pmatrix} 5 \\ 4 \end{pmatrix}$$



Eigenvalue analysis

- The eigenvector $\mathbf{x}$ to the associated eigenvalue will be scaled with λ under the transformation $Ax = \lambda x$.
- The matrix A has the following eigenvalues and eigenvectors: $\mathbf{x}_1 = (1\ 1)^T$ with eigenvalue $\lambda_1 = 3$ and $\mathbf{x}_2 = (1\ -1)^T$ with eigenvalue $\lambda_2 = 1$.



Eigenvalue analysis

Theorem: A symmetric, real matrix of dimension $n \times n$ has n eigenvalues λ_k with associated eigenvectors $\mathbf{x}_k$ (k=1, ..., n). The eigenvectors$\{x_1, x_2, ..., x_n\}$ build a complete system of n orthonormal vectors, e.g.

$$\langle x_i, x_j \rangle = 1 \quad \text{if } i = j, \quad \text{and} \quad \langle x_i, x_j \rangle = 0 \quad \text{if } i \neq j \tag{27}$$

- A simple—but practically irrelevant—alternative to compute the eigenvalues of matrix A is the *characteristic polynomial*:

$$p(\lambda) = \det(A - \lambda E). \tag{28}$$

Example:

$$\begin{vmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{vmatrix} = (2-\lambda)^2 - 1 = 0 \quad \Longrightarrow \quad \lambda_1 = 1,\ \lambda_2 = 3$$

Convergence of the Jacobi-Iteration

Theorem: The iteration $\mathbf{x}^{(m)} = R_J \mathbf{x}^{(m+1)} + \mathbf{c_J}$ converges for all starting vectors $\mathbf{x}^{(0)}$ against a solution $\mathbf{x}^*$ if and only if $\rho(R_J) < 1$.

Explanation: Every vector from $\mathbb{R}^n$ can be written as a linear combination of eigenvectors from R (if R is not singular). If one eigenvalue is larger than 1, then the Jacobi-Iteration will not converge.

General criteria for good convergence :

- $\kappa \approx 1$, small ρ
- A "close" to the identity matrix I
- All these methods will convergence in one iteration of $A = \beta I$

Jacobi-Iteration

Here we can see how the iteration will affect the error term $\mathbf{e}^{(m)}$:

$$\begin{aligned}
\mathbf{x}^{(m+1)} &= R_J\mathbf{x}^{(m)} + \mathbf{c}_J \\
&= R_J(\mathbf{x} + \mathbf{e}^{(m)}) + \mathbf{c}_J \\
&= \underbrace{R_J\mathbf{x} + \mathbf{c}_J}_{\text{siehe (23)}} + R_J\mathbf{e}^{(m)} \\
&= \mathbf{x} + R_J\mathbf{e}^{(m)} \\
\mathbf{e}^{(m+1)} &= R_J\mathbf{e}^{(m)}. \qquad \Big| (\mathbf{x} - \mathbf{x}^{(m+1)} = \mathbf{e}^{(m+1)})
\end{aligned} \tag{29}$$

- If $\rho(R) < 1$, then the error will converges 0 (with $m \to \infty$)

Gauss-Seidel Iteration

- A better, but still very easy method, is the *Gauss-Seidel-Iteration*.
- In principle, the idea is very similar to the Jacobi method, but is uses immediately new components of $x_j^{(m+1)}$

$$x_j^{(m+1)} = \frac{1}{a_{jj}} \left(b_j - \underbrace{\sum_{k=1}^{j-1} a_{jk} x_k^{(m+1)}}_{\text{new } x_j} - \underbrace{\sum_{k=j+1}^{n} a_{jk} x_k^{(m)}}_{\text{old } x_j} \right) \tag{30}$$