Numerical Methods I

MATH-GA 2010.001/CSCI-GA 2420.001

Benjamin Peherstorfer
Courant Institute, NYU

Based on slides by G. Stadler and A. Donev

Today

Last time

- Iterative methods for systems of linear equations
- Started with conjugate gradient method

Today

- Conjugate gradients method
- Interpolation

Announcements

- Homework 5 is due Mon, Nov 21, 2022 before class

Conjugate gradient method

In the following A is symmetric positive definite.

Formulate solving $Ax = b$ as an optimization problem: Define

$$f(x) = \frac{1}{2}x^T Ax - b^T x\,,$$

and minimize

$$\min_{x \in \mathbb{R}^n} f(x)$$

Because A is positive definite, the function f is convex. It is sufficient to look at the gradient

$$\nabla f(x) = \frac{1}{2}A^T x + \frac{1}{2}Ax - b = Ax - b = -r(x) = 0 \iff Ax = b$$

What is the benefit of this point of view?

In the following A is symmetric positive definite.

Formulate solving $Ax = b$ as an optimization problem: Define

$$f(x) = \frac{1}{2}x^T Ax - b^T x\,,$$

and minimize

$$\min_{x\in\mathbb{R}^n} f(x)$$

Because A is positive definite, the function f is convex. It is sufficient to look at the gradient

$$\nabla f(x) = \frac{1}{2}A^T x + \frac{1}{2}Ax - b = Ax - b = -r(x) = 0 \iff Ax = b$$

What is the benefit of this point of view? We now can let loose all what we know about optimization to solve $Ax = b$

Our first try is applying the method of steepest descent in the direction of the negative gradient

$$-\nabla f = r$$

which happens to be the residual

$$\begin{aligned} r_k &= b - Ax_k \\ \alpha_k &= \frac{r_k^T r_k}{r_k^T A r_k} \\ x_{k+1} &= x_k + \alpha_k r_k \end{aligned}$$

The step length α_k minimizes $f(x_k + \alpha_k r_k)$ as a function of α_k $\rightsquigarrow$ board

For steepest descent, if A is spd, we obtain

$$\|x^* - x_k\|_A \leq \left(\frac{\kappa_2(A) - 1}{\kappa_2(A) + 1} \right)^k \|x^* - x_0\|_A \,,$$

where $\langle x, y \rangle_A = x^T A y$ and $\| \cdot \|_A = \sqrt{\langle \cdot, \cdot \rangle_A}$.

Proof $\rightsquigarrow$ board

SPD matrix A, apply steepest descent

$$\|x_k - x^*\|_A \leq \left(\frac{k_2(A)-1}{k_2(A)+1}\right)^k \|x^0 - x^*\|_A$$

$$x_{k+1}(\alpha) = x_k + \alpha r_k \qquad \left[\begin{aligned} r_k &= b - Ax_k \\ &= Ax^* - Ax_k = A(x^* - x_k) \end{aligned}\right] \checkmark$$

$$= x_k + \alpha A(x^* - x_k)$$

$$\underbrace{x^* - x_{k+1}(\alpha)}_{e_{k+1}(\alpha)} = x^* - x_k - \alpha A(x^* - x_k)$$

$$= (1 - \alpha A)\underbrace{(x^* - x_k)}_{e_k}$$

$$\|e_{k+1}(\alpha)\|_A^2 = e_k^T (1-\alpha A)^T A (1 - \alpha A) e_k$$

eigenbasis (ortho.) of A, $\{\lambda_j\}$, $\{z_j\}$

$$e_k = \sum_{j=1}^{N} a_j z_j$$

$$\|e_{k+1}(\alpha)\|_A^2 = \dots = \sum_{j=1}^{N} \lambda_j a_j^2 (1 - \alpha\lambda_j)^2$$

Set $\hat{\alpha} = \dfrac{2}{\underset{\max}{\lambda_1} + \underset{\min}{\lambda_N}}$

$$\| e_{k+1}(\hat{\alpha}) \|_A^2 = \sum_{j=1}^{N} \lambda_j a_j^2 \left(\frac{\lambda_1 + \lambda_N - 2\lambda_j}{\lambda_1 + \lambda_N} \right)^2$$

$$= \sum_{j=1}^{N} \lambda_j a_j^2 \; \frac{(\lambda_1 - \lambda_N)^2 - 4(\lambda_1 - \lambda_j)(\lambda_j - \lambda_N)}{(\lambda_1 + \lambda_N)^2}$$

$$\leq \frac{(\lambda_1 - \lambda_N)^2}{(\lambda_1 + \lambda_N)^2} \underbrace{\sum_{j=1}^{N} \lambda_j a_j^2}_{\| e_k \|_A^2}$$

steepest descent: choose α_k such that

$$f(x_{k+1}(\alpha_k)) = f(x_k + \alpha_k r_k)$$

is minimal

For the minimum x^*

$$f(x) = f(x^*) + \frac{1}{2} \| x^* - x \|_A^2$$

$$\min_{\alpha_k} f(x_k+\alpha_k r_k) \iff \min_{\alpha_k} \frac{1}{2}\|\underbrace{x^* - (x_k+\alpha_k r_k)}_{e_{k+1}(\alpha_k)}\|_A^2$$

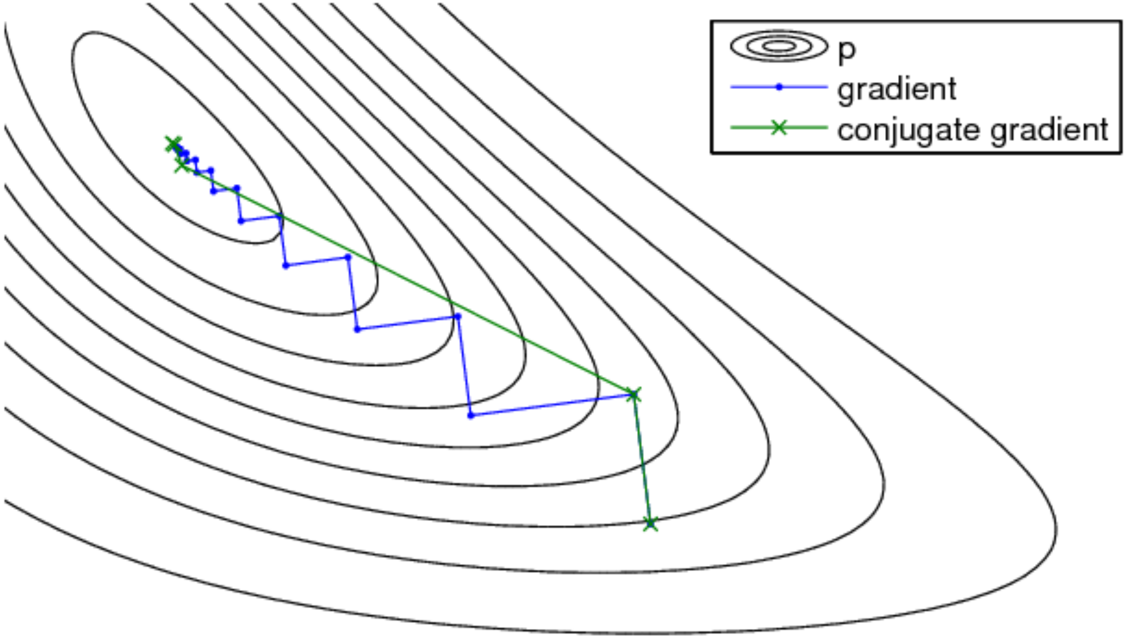
$$\|e_{k+1}(\alpha_k)\|_A^2 = \min_{\alpha} \|e_{k+1}(\alpha)\|_A^2$$

$$\le \|e_{k+1}(\hat{\alpha})\|_A$$

$$\le \left(\frac{\lambda_1-\lambda_N}{\lambda_1+\lambda_N}\right)\|e_k\|_A$$

$$\|e_k\|_A \le \underbrace{\left(\frac{\lambda_1-\lambda_N}{\lambda_1+\lambda_N}\right)}_{\parallel\; \frac{\kappa_2(A)-1}{\kappa_2(A)+1}}^{k} \|e_0\|_A$$

$$\kappa_2(A) = \frac{\lambda_1}{\lambda_N}$$



[Figure: Kuusela et al., 2009]

The convergence behavior of steepest descent in this context can be poor: we eventually get arbitrarily close to the minimum but we can always destroy something of the already achieved when applying the update $\rightsquigarrow$ can we find better search directions?

Conjugate gradient method

- What do all iterative methods we looked at so far have in common?

Conjugate gradient method

- What do all iterative methods we looked at so far have in common?
- All methods so far use information about x_{k-1} to get x_k. All information about earlier iterations is ignored.

Conjugate gradient method

- What do all iterative methods we looked at so far have in common?
- All methods so far use information about x_{k-1} to get x_k. All information about earlier iterations is ignored.

- The conjugate gradient (CG) method is a variation of steepest descent that *has a memory*.

- Let $p_1, \ldots, p_k$ be the directions up to step k, then CG uses the space

$$x_0 + \operatorname{span}\{p_1, \ldots, p_k\}, \qquad\qquad x_0 \text{ starting point}$$

  to find the next iterate x_k and thus

$$x_k = x_0 + \sum_{i=1}^{k} \alpha_i p_i$$

- (Recall that steepest descent uses only the search direction $p_k = r_{k-1} = -\nabla f(x_{k-1})$ to find the iterate x_k)

We want the following

a The search directions $p_1, \ldots, p_k$ should be linearly independent ("we don't destroy what we have achieved")

b We have ("we do the best we can at each step")

$$f(x_k) = \min_{x \in x_0 + \text{span}(p_1, \ldots, p_k)} f(x)$$

c The step x_k can be calculated easily from x_{k-1}

What do conditions (a) and (b) guarantee?

We want the following

a The search directions $p_1, \ldots, p_k$ should be linearly independent ("we don't destroy what we have achieved")

b We have ("we do the best we can at each step")

$$f(x_k) = \min_{x \in x_0 + \mathrm{span}(p_1, \ldots, p_k)} f(x)$$

c The step x_k can be calculated easily from x_{k-1}

What do conditions (a) and (b) guarantee? Convergence in N steps because at the N-th step we have $x_0 + \mathrm{span}(p_1, \ldots, p_N) = \mathbb{R}^N$ and thus we minimize f over $\mathbb{R}^N$

Let's start by writing

$$x_k = x_0 + P_{k-1}y + \alpha p_k \,,$$

where $P_{k-1} = [p_1, \ldots, p_{k-1}] \in \mathbb{R}^{N\times(k-1)}, y \in \mathbb{R}^{k-1}, \alpha \in \mathbb{R}$.

Our aim is to determine y and α. So let's look at minimizing $f(x_k)$ w.r.t. y and α

$$f(x_k) = \cdots = \underbrace{f(x_0 + P_{k-1}y)}_{\text{only depends on } y} + \alpha p_k^T A P_{k-1} y + \underbrace{\frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0}_{\text{only depends on } \alpha}$$

(recall that $f(x) = \frac{1}{2}x^T Ax - b^T x$).

Let's start by writing

$$x_k = x_0 + P_{k-1}y + \alpha p_k \,,$$

where $P_{k-1} = [p_1, \ldots, p_{k-1}] \in \mathbb{R}^{N\times(k-1)}, y \in \mathbb{R}^{k-1}, \alpha \in \mathbb{R}$.

Our aim is to determine y and α. So let's look at minimizing $f(x_k)$ w.r.t. y and α

$$f(x_k) = \cdots = \underbrace{f(x_0 + P_{k-1}y)}_{\text{only depends on } y} + \alpha p_k^T A P_{k-1} y + \underbrace{\frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0}_{\text{only depends on } \alpha}$$

(recall that $f(x) = \frac{1}{2}x^T Ax - b^T x$).
The mixed term in the middle depends on α and y, otherwise we could optimize separately for y and α. How should we choose p_k?

Let's start by writing

$$x_k = x_0 + P_{k-1}y + \alpha p_k \ ,$$

where $P_{k-1} = [p_1, \dots, p_{k-1}] \in \mathbb{R}^{N\times(k-1)}, y \in \mathbb{R}^{k-1}, \alpha \in \mathbb{R}$.

Our aim is to determine y and α. So let's look at minimizing $f(x_k)$ w.r.t. y and α

$$f(x_k) = \cdots = \underbrace{f(x_0 + P_{k-1}y)}_{\text{only depends on } y} + \alpha p_k^T A P_{k-1} y + \underbrace{\frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0}_{\text{only depends on } \alpha}$$

(recall that $f(x) = \frac{1}{2}x^T A x - b^T x$).
The mixed term in the middle depends on α and y, otherwise we could optimize separately for y and α. How should we choose p_k?

Let's choose the search direction p_k such that

$$p_k^T A P_{k-1} = 0$$

which means

$$p_k \in \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}^{\perp}$$

Thus, with $p_k^T A P_{k-1} = 0$ we get

$$\min_{x_k \in x_0 + \text{span}\{p_1,\dots,p_k\}} f(x_k) = \min_{y \in \mathbb{R}^{k-1}} f(x_0 + P_{k-1}y) + \min_{\alpha \in \mathbb{R}} \left(\frac{\alpha^2}{2} p_k^T A p_k - \alpha p_k^T r_0 \right)$$

- The first minimization problem is solved for $y = y_{k-1}$ computed from step $k-1$ and then $x_{k-1} = x_0 + P_{k-1} y_{k-1}$ satisfies

$$f(x_{k-1}) = \min_{x_0 + \text{span}\{p_1,\dots,p_{k-1}\}} f(x)$$

- The solution to the second minimization problem is just a scalar

$$\alpha_k = \frac{p_k^T r_0}{p_k^T A p_k}$$

$\rightsquigarrow$ satisfy conditions (b) and (c) from above.

- We said the search directions $p_1, \ldots, p_k$ have to be conjugate, i.e., orthogonal w.r.t. A

$$p_i^T A p_j = 0\,, \qquad i, j = 1, \ldots, k\,, i \neq j \tag{1}$$

- One can show that (1) implies that $p_1, \ldots, p_k$ are linearly independent (w.r.t. $\langle \cdot, \cdot, \rangle$), which satisfies condition (a)

- To find the search direction p_k, we want to combine positive aspects of steepest descent and conjugate gradients. In steepest descent we have $p_k = r_{k-1}$. So let's stay close to r_{k-1} but additionally enforce that p_k is A-conjugate to previous search directions $p_1, \ldots, p_{k-1}$

How can we achieve this?

- We said the search directions $p_1, \dots, p_k$ have to be conjugate, i.e., orthogonal w.r.t. A

$$p_i^T A p_j = 0\,, \qquad i, j = 1, \dots, k\,, i \neq j \tag{1}$$

- One can show that (1) implies that $p_1, \dots, p_k$ are linearly independent (w.r.t. $\langle \cdot, \cdot, \rangle$), which satisfies condition (a)

- To find the search direction p_k, we want to combine positive aspects of steepest descent and conjugate gradients. In steepest descent we have $p_k = r_{k-1}$. So let's stay close to r_{k-1} but additionally enforce that p_k is A-conjugate to previous search directions $p_1, \dots, p_{k-1}$

How can we achieve this? $\rightsquigarrow$ Gram-Schmidt orthogonalization

Apply Gram-Schmidt to r_{k-1} so that we obtain p_k that is A-conjugate to $p_1, \dots, p_{k-1}$

$$p_k = r_{k-1} - \sum_{j=1}^{k-1} \frac{\langle r_{k-1}, p_j \rangle_A}{\|p_j\|_A^2} p_j$$

We need following technical statements $\rightsquigarrow$ board:

- If $r_{k-1} = b - Ax_{k-1} \neq 0$, then there exists $p_k \in \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}^\perp$ such that $p_k^T r_{k-1} \neq 0$ and $p_k^T r_{k-1} = p_k^T r_0$
- It then follows that (why is this helpful?)

$$\alpha_k = \frac{p_k^T r_0}{p_k^T A p_k} = \frac{p_k^T r_{k-1}}{p_k^T A p_k}$$

- If $r_j \neq 0$ for $j < k$, then

$$\langle r_{k-1}, p_j \rangle_A = 0\,, \qquad j < k-1\,.$$

Apply Gram-Schmidt to r_{k-1} so that we obtain p_k that is A-conjugate to $p_1, \dots, p_{k-1}$

$$p_k = r_{k-1} - \sum_{j=1}^{k-1} \frac{\langle r_{k-1}, p_j \rangle_A}{\|p_j\|_A^2} p_j$$

We need following technical statements $\rightsquigarrow$ board:

- If $r_{k-1} = b - Ax_{k-1} \neq 0$, then there exists $p_k \in \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}^\perp$ such that $p_k^T r_{k-1} \neq 0$ and $p_k^T r_{k-1} = p_k^T r_0$
- It then follows that (why is this helpful?)

$$\alpha_k = \frac{p_k^T r_0}{p_k^T A p_k} = \frac{p_k^T r_{k-1}}{p_k^T A p_k}$$

- If $r_j \neq 0$ for $j < k$, then

$$\langle r_{k-1}, p_j \rangle_A = 0\,, \qquad j < k - 1\,.$$

to obtain that (why is this useful?)

$$p_k = r_{k-1} - \frac{\langle r_{k-1}, p_{k-1} \rangle_A}{\|p_{k-1}\|_A^2} p_{k-1}$$

$$r_{k-1} = b - Ax_{k-1} \neq 0$$

$$\Rightarrow \exists p_k \in \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}^{\perp}$$

$$p_k^T r_{k-1} \neq 0$$

For $k=1$: $\quad p_1 = r_0$

$k>1$: because $\quad r_{k-1} \neq 0$

$$x^* = A^{-1}b \notin x_0 + \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

$$\underbrace{b - Ax_0}_{r_0} \notin \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}$$

$$p_k \in \operatorname{span}\{Ap_1, \dots, Ap_{k-1}\}^{\perp}$$

$$p_k^T r_0 \neq 0$$

$$\begin{aligned}
p_k^T r_{k-1} &= p_k^T (b - Ax_{k-1}) \\
&= p_k^T (b - A(x_0 + P_{k-1} y_{k-1})) \\
&= \underbrace{p_k^T \overbrace{(b - Ax_0)}^{r_0}}_{\neq 0} - \underbrace{p_k^T A P_{k-1}}_{=0} y
\end{aligned}$$

$$p_k^T r_{k-1} \neq 0$$

$$\forall p \in \operatorname{span}\{p_1, \dots, p_q\} : \quad p^T r_q = 0$$

$$\bar{x} = \arg\min \|Ax - b\|_2^2$$

$$r = b - A\bar{x} \perp \operatorname{col}(A)$$

$$\min_{x \in x_0 + \operatorname{span}\{p_1, \dots, p_q\}} f(x)$$

$$\Rightarrow (*) \quad p^T r_q = 0 \qquad \forall p \in \operatorname{span}\{p_1, \dots, p_q\}$$

Now want

$$\langle r_{q-1}, p_j \rangle_A = 0 \qquad j < q-1$$

$$x_j = x_{j-1} + \alpha_j p_j$$

$$r_j = b - Ax_j = \underbrace{b - Ax_{j-1}}_{r_{j-1}} - \alpha_j A p_j$$

$$\alpha_j A p_j = r_{j-1} - r_j \qquad j < q-1$$

Gram - Schmidt

$$r_j = p_{j+1} + \sum_{i=1}^{j} \frac{\langle r_j, p_i \rangle_A}{\|p_i\|_A^2} p_i \in \operatorname{span}\{p_1, \dots, p_{j+1}\}$$

$$\operatorname{span}\{p_1, \dots, p_{j+1}\} \subseteq \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

$$\alpha_j A p_j = r_{j-1} - r_j \in \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

gram - schmidt

$$\alpha_j = \frac{r_{j-1}^T p_j}{\|p_j\|_A^2} = \frac{1}{\|p_j\|_A^2} \left[\|r_{j-1}\|_2^2 - \sum_{j=1}^{j-1} \frac{\langle r_{j-1}, p_i \rangle_A}{\|p_i\|_A^2} \underbrace{r_{j-1}^T p_i}_{\text{from } (*)} \right]$$

$$= \frac{\|r_{j-1}\|_2^2}{\|p_j\|_A^2} > 0$$

$$\alpha_j \neq 0$$

$$A p_j \in \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

$$\langle r_{k-1}, p_j \rangle_A = r_{k-1}^T A p_j$$

$$= \langle r_{k-1}, A p_j \rangle_2$$

$$A p_j \in \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

$$\langle r_{k-1}, p \rangle = 0 \quad \forall p \in \operatorname{span}\{p_1, \dots, p_{k-1}\}$$

$$\Rightarrow \langle r_{k-1}, p_j \rangle_A = 0$$

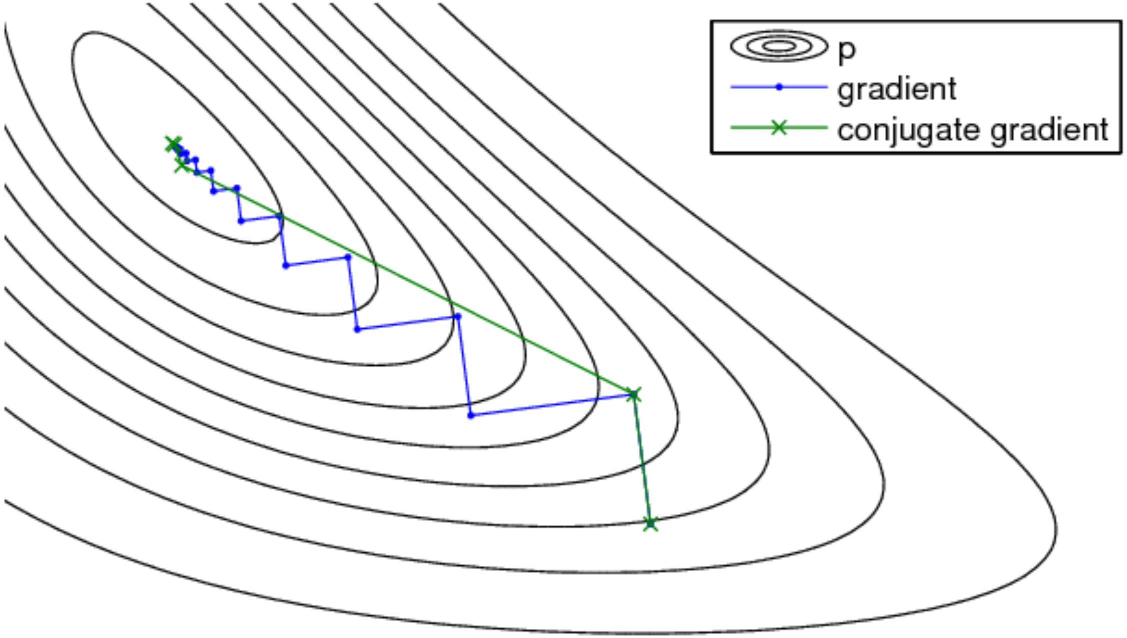
$$p_k = r_{k-1} - \frac{\langle r_{k-1}, p_{k-1} \rangle_A}{\| p_{k-1} \|_A^2} p_{k-1}$$

The conjugate gradient method

Choose $x_0 \in \mathbb{R}^N$ and set $p_0 = 0$. For $k = 1, 2, 3, \ldots,$ stop if $r_{k-1} = b - Ax_{k-1}$ small

1. Set
$$\beta_{k-1} = \frac{\langle r_{k-1}, p_{k-1} \rangle_A}{\|p_{k-1}\|_A^2}$$
2. Set
$$p_k = r_{k-1} - \beta_{k-1} p_{k-1}$$
3. Set
$$\alpha_k = \frac{r_{k-1}^T p_k}{\|p_k\|_A^2}$$
4. Set
$$x_k = x_{k-1} + \alpha_k p_k$$
5. Set
$$r_k = b - Ax_k$$

   and check for convergence



[Figure: Kuusela et al., 2009]

It can be shown that for $k \geq 1$ and $e_j \neq 0, j < k$ it holds

$$\|e_k\|_A \leq 2 \left(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1} \right)^k \|e_0\|_A$$

for spd matrices A. $\rightsquigarrow$ Trefethen & Bau

Krylov subspace

Given an spd matrix $A \in \mathbb{R}^{N \times N}$, the Krylov subspace of order k is

$$\mathcal{K}_k(A, r_0) = \text{span}\left\{r_0, Ar_0, \dots, A^{k-1}r_0\right\}$$

where, e.g., $r_0 = b - Ax_0$

All search directions of CG are in $\mathcal{K}_k(A, r_0)$ and all iterates $x_1, x_2, \dots, x_k$ are in $x_0 + \mathcal{K}_k(A, r_0)$

There is a range of other methods that apply to more general matrices than spd that build on approximations in Krylov subspaces to accelerate convergence (e.g., GMRES (general residual method))

- There are also methods for finding eigenvalues via Krylov methods (Lanczos, Arnoldi iterations)
- Think of Krylov methods has having a memory of previous iterations, whereas, e.g., a power method only looks at the previous iteration (if you like stochastic processes, think of Markovian vs. non-Markovian dynamics)

Matlab implementation

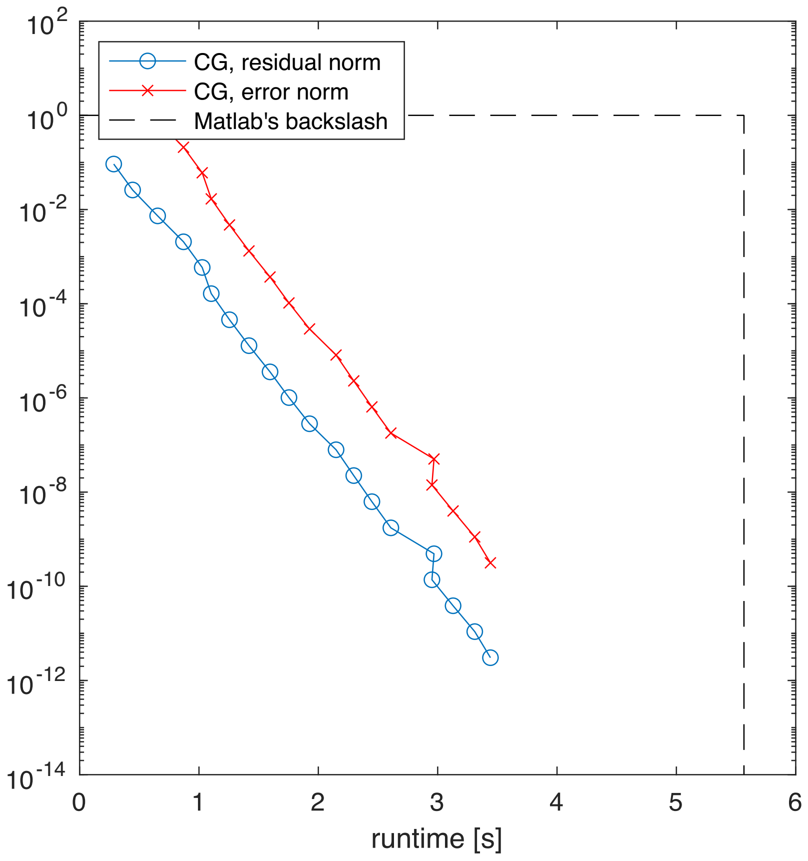
```
function x = conjgrad(A, b, maxIter)

[N, ~] = size(A);
x = zeros(N, 1);
r = b - A*x;
p = r;
alpha = (r'*p)/(p'*A*p);
x = x + alpha*p;
r = b - A*x;

for i=1:maxIter
    beta = (r'*A*p)/(p'*A*p);
    p = r - beta*p;
    alpha = (r'*p)/(p'*A*p);
    x = x + alpha*p;
    r = b - A*x;
end
```

Experiment with 10000×10000 spd matrix

Condition number of this matrix is ≈ 5 (very! well conditioned)



Discussion of the CG method

- In principle, the CG algorithm is a direct solver because is converges after N steps; however, it is mostly used as an iterative method because we don't want to wait for N steps

- The convergence speed of the CG method depends on matrix properties as well. Fast convergence if the spectrum is clustered.

- However, similarly slow convergence can be expected for matrices coming from PDE discretizations and therefore preconditioning is necessary

$$Q^{-1}Ax = Q^{-1}b$$

- Preconditioned CG methods (for example multigrid can act as a preconditioner) are among the fastest solvers and achieve $\mathcal{O}(N)$ in ideal settings.

Interpolation

Function approximation

Consider a function $f \in \mathcal{V}$ in a function space $\mathcal{V}$. Let now $\phi_1, \ldots, \phi_n$ be a basis of an n-dimensional space $\mathcal{V}_n$.

Function approximation

Consider a function $f \in \mathcal{V}$ in a function space $\mathcal{V}$. Let now $\phi_1, \ldots, \phi_n$ be a basis of an n-dimensional space $\mathcal{V}_n$.

The task that we are interested in is finding a function $f^* \in \mathcal{V}_n$ that approximates f, i.e.,

$$f^*(x) = \sum_{i=1}^{n} c_i \phi_i(x)\,,$$

with coefficients $c_1, \ldots, c_n$.

Function approximation

Consider a function $f \in \mathcal{V}$ in a function space $\mathcal{V}$. Let now $\phi_1, \dots, \phi_n$ be a basis of an n-dimensional space $\mathcal{V}_n$.

The task that we are interested in is finding a function $f^* \in \mathcal{V}_n$ that approximates f, i.e.,

$$f^*(x) = \sum_{i=1}^{n} c_i \phi_i(x)\,,$$

with coefficients $c_1, \dots, c_n$.

If we have an inner product, the best-approximation of f in $\mathcal{V}_n$ w.r.t. the induced norm is given by the projection

$$f^* = \Pi_n f\,,$$

where Π_n is the orthogonal projection onto $\mathcal{V}_n$.

Function approximation

Consider a function $f \in \mathcal{V}$ in a function space $\mathcal{V}$. Let now $\phi_1, \ldots, \phi_n$ be a basis of an n-dimensional space $\mathcal{V}_n$.

The task that we are interested in is finding a function $f^* \in \mathcal{V}_n$ that approximates f, i.e.,

$$f^*(x) = \sum_{i=1}^{n} c_i \phi_i(x) \,,$$

with coefficients $c_1, \ldots, c_n$.

If we have an inner product, the best-approximation of f in $\mathcal{V}_n$ w.r.t. the induced norm is given by the projection

$$f^* = \Pi_n f \,,$$

where Π_n is the orthogonal projection onto $\mathcal{V}_n$.

However, in many cases, we cannot directly compute the projection of f onto $\mathcal{V}_n$ because we have "too little knowledge about f" $\rightsquigarrow$ interpolation (/regression)

Interpolation

Consider n pairs of data samples $(x_i, y_i), i = 1, \ldots, n$ with

$$y_i = f(x_i)$$

Based on $\{(x_i, y_i)\}_{i=1}^n$, we now would like to find an approximation $\tilde{f} \in \mathcal{V}_n$ that is "close" to f.

For example, we could enforce the interpolation condition, namely that it holds

$$\tilde{f}(x_i) = f(x_i), \qquad i = 1, \ldots, n$$

We could also use regression $(m > n)$ and minimize, e.g.,

$$\frac{1}{m} \sum_{i=1}^{m} |y_i - \tilde{f}(x_i)|^2$$

The error of $\tilde{f}$ w.r.t. f can then typically be split into two components (we will formalize this moving forward): which?

The error of $\tilde{f}$ w.r.t. f can then typically be split into two components (we will formalize this moving forward): which?

$$\|\tilde{f} - f\| \leq \Lambda(x_1, \ldots, x_n)\|f^* - f\|$$

The projection error $\|f^* - f\|$ describes the best we can do in the space $\mathcal{V}_n$. Even if we had "full knowledge" of f so that we could compute $f^* = \Pi_n f$, we are limited by the space $\mathcal{V}_n$

Intuitively, we'd also expect that the error of $\tilde{f}$ depends on the points $x_1, \ldots, x_n$ at which we have samples of f. This is captured by the "constant" $\Lambda(x_1, \ldots, x_n)$ that is independent of f but depends on $x_1, \ldots, x_n$.

Polynomial interpolation

Consider $n+1$ pairs $(x_i, y_i), i = 0, \ldots, n$ of a function f with

$$y_i = f(x_i)$$

Polynomial interpolation

Consider $n+1$ pairs $(x_i, y_i), i = 0, \dots, n$ of a function f with

$$y_i = f(x_i)$$

Let now $\mathbb{P}_n$ be the set of all polynomials up to degree n over $\mathbb{R}$ so that we have for all $P \in \mathbb{P}_n$

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 \,, \qquad a_n, \dots, a_0 \in \mathbb{R}$$

Polynomial interpolation

Consider $n+1$ pairs $(x_i, y_i), i = 0, \ldots, n$ of a function f with

$$y_i = f(x_i)$$

Let now $\mathbb{P}_n$ be the set of all polynomials up to degree n over $\mathbb{R}$ so that we have for all $P \in \mathbb{P}_n$

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 \,, \qquad a_n, \ldots, a_0 \in \mathbb{R}$$

We would like to find a $P \in \mathbb{P}_n$ such that

$$P(x_i) = y_i \,, \qquad i = 0, \ldots, n$$

- The P is what $\tilde{f}$ was on the previous slide
- By saying P is a polynomial of degree n, we fixed the space $\mathcal{V}_{n+1}$ with the notation of the previous slide

Theorem: Given $n+1$ points (x_i, y_i) with pairwise distinct $x_0, \dots, x_n$, there exists a unique polynomial $P \in \mathbb{P}_n$ such that

$$P(x_i) = y_i\,, \qquad i = 0, \dots, n$$

We sometimes refer to this unique polynomial as $P_f(\cdot|x_0, \dots, x_n)$

Instead of proving this theorem, let's try to construct $P_f(\cdot|x_0, \dots, x_n)$. What do we need to construct $P \in \mathbb{P}_n$ for a data set $\{(x_i, y_i)\}_{i=0}^n$?

Theorem: Given $n+1$ points (x_i, y_i) with pairwise distinct $x_0, \ldots, x_n$, there exists a unique polynomial $P \in \mathbb{P}_n$ such that

$$P(x_i) = y_i\,, \qquad i = 0, \ldots, n$$

We sometimes refer to this unique polynomial as $P_f(\cdot|x_0, \ldots, x_n)$

Instead of proving this theorem, let's try to construct $P_f(\cdot|x_0, \ldots, x_n)$. What do we need to construct $P \in \mathbb{P}_n$ for a data set $\{(x_i, y_i)\}_{i=0}^n$? A basis of $\mathbb{P}_n$.

Theorem: Given $n+1$ points (x_i, y_i) with pairwise distinct $x_0, \ldots, x_n$, there exists a unique polynomial $P \in \mathbb{P}_n$ such that

$$P(x_i) = y_i \,, \qquad i = 0, \ldots, n$$

We sometimes refer to this unique polynomial as $P_f(\cdot|x_0, \ldots, x_n)$

Instead of proving this theorem, let's try to construct $P_f(\cdot|x_0, \ldots, x_n)$. What do we need to construct $P \in \mathbb{P}_n$ for a data set $\{(x_i, y_i)\}_{i=0}^n$? A basis of $\mathbb{P}_n$. Let's give the monomial basis $1, x, x^2, \ldots, x^n$ a chance $\rightsquigarrow$ board

$$p(x) = a_n x^n + \dots + a_1 x + a_0$$

$n+1$ unknowns

$a_n, \dots, a_0$

$n+1$ equations

$p(x_0) = y_0$

$\vdots$

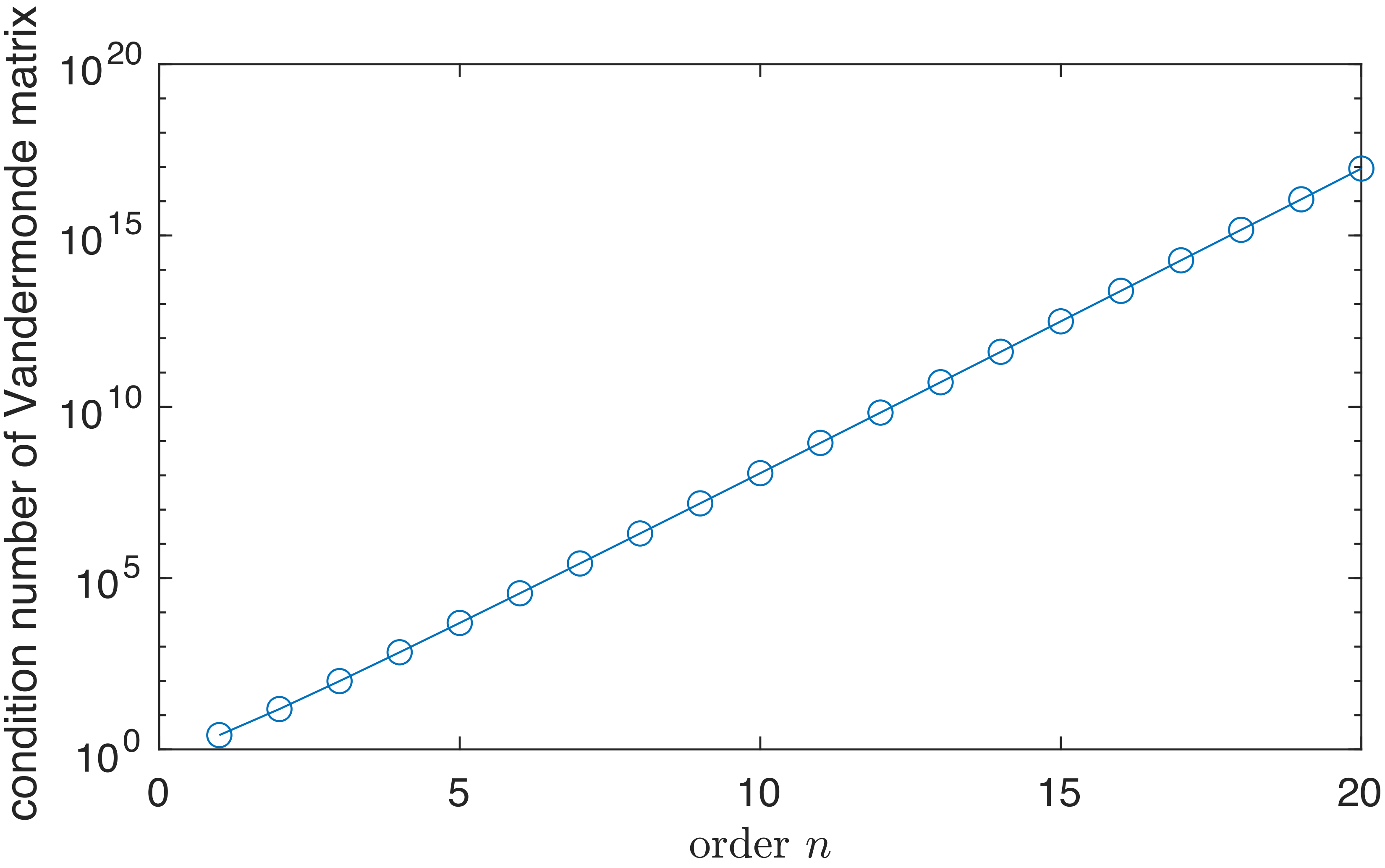
$p(x_n) = y_n$

$$\underbrace{\begin{bmatrix} x_0^n & x_0^{n-1} & \cdots & x_0 & 1 \\ \vdots & & & & \\ x_n^n & x_n^{n-1} & \cdots & x_n & 1 \end{bmatrix}}_{(n+1)\times(n+1)} \underbrace{\begin{bmatrix} a_n \\ \vdots \\ a_1 \\ a_0 \end{bmatrix}}_{n+1} = \begin{bmatrix} y_0 \\ \vdots \\ y_n \end{bmatrix}$$

Vandermonde matrix V_n

$$\det(V_n) = \prod_{i=0}^{n} \prod_{j=i+1}^{n} (x_i - x_j)$$

Solve $V_n a = y$ $\rightsquigarrow$ linear system

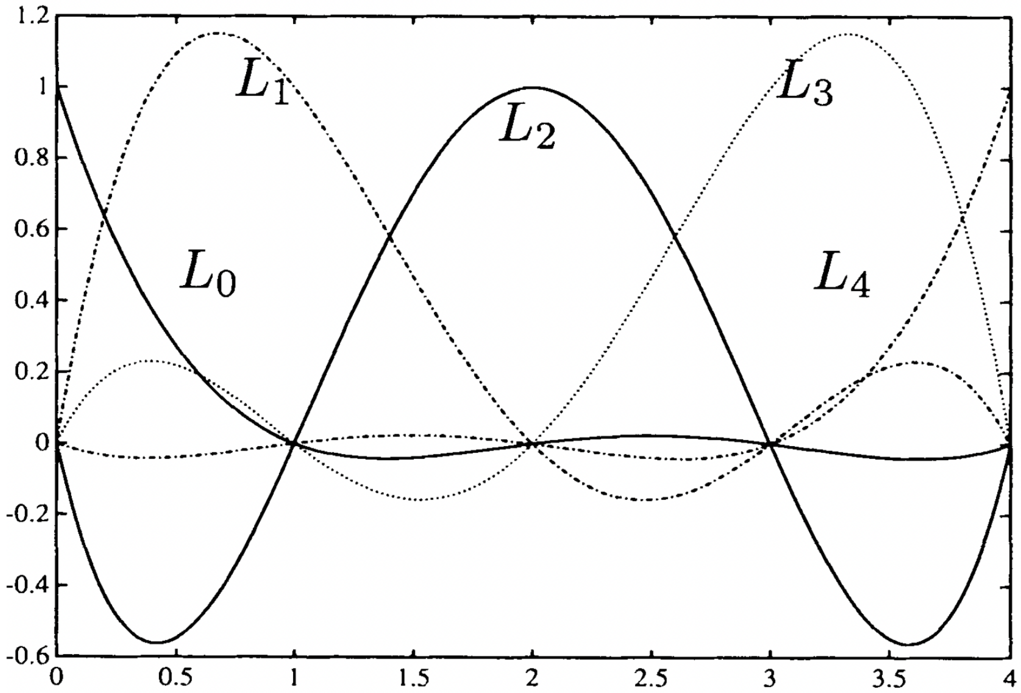


$\rightsquigarrow$ we really should look for another basis

Lagrange basis

The Lagrange polynomials $L_0, \ldots, L_n \in \mathbb{P}_n$ are uniquely defined for distinct $x_0, \ldots, x_n$

$$L_i(x_j) = \delta_{ij}\,, \qquad L_i \in \mathbb{P}_n\,.$$



Lagrange polynomials up to order $n = 4$ for equidistant $x_0, \ldots, x_4$. [Figure: Deuflhard]

The corresponding explicit formula is

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}, \qquad i = 0, \ldots, n$$

What are the coefficients $a_n, \ldots, a_0$ so that $P(x_i) = y_i$ for $i = 0, \ldots, n$?

The corresponding explicit formula is

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}\,, \qquad i = 0, \ldots, n$$

What are the coefficients $a_n, \ldots, a_0$ so that $P(x_i) = y_i$ for $i = 0, \ldots, n$?

$$P(x) = \sum_{i=0}^{n} y_i L_i(x)$$

because

$$P(x_j) = \sum_{i=0}^{n} y_i L_i(x_j) = \sum_{i=0}^{n} y_i \delta_{ij} = y_j$$

If we have the basis $L_0, \ldots, L_n$, we obtain the polynomial P for free

The corresponding explicit formula is

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}, \qquad i = 0, \ldots, n$$

What are the coefficients $a_n, \ldots, a_0$ so that $P(x_i) = y_i$ for $i = 0, \ldots, n$?

$$P(x) = \sum_{i=0}^{n} y_i L_i(x)$$

because

$$P(x_j) = \sum_{i=0}^{n} y_i L_i(x_j) = \sum_{i=0}^{n} y_i \delta_{ij} = y_j$$

If we have the basis $L_0, \ldots, L_n$, we obtain the polynomial P for free but the cost of evaluating the polynomial is too high for practical computations

The Lagrange polynomials are orthogonal w.r.t. the following inner product over $\mathbb{P}_n$

$$\langle P, Q \rangle = \sum_{i=0}^{n} P(x_i) Q(x_i)\,, \qquad P, Q \in \mathbb{P}_n$$

Let's try to generalize this to other scalar products to find other orthogonal bases

Orthogonal polynomials

Define an inner product between functions:

$$(f, g) = \int_a^b \omega(x) f(x) g(x)\, dx,$$

where $\omega(x) > 0$ for $a \leq x \leq b$ is a weight function. The induced norm is $\|f\| := \sqrt{(f, f)}$.

Let $P_0, P_1, P_2, \ldots, P_K$ be polynomials of $0, 1, 2, \ldots, K$ order, respectively. They are called orthogonal polynomials on $[a, b]$ with respect to the weight function $\omega(x)$ if it holds

$$(P_i, P_j) = \int_a^b \omega(x) P_i(x) P_j(x) \mathrm{d}x = \delta_{ij}\gamma_i\,, \qquad i, j = 0, \ldots, K\,,$$

with $\gamma_i = \|P_i\|^2 > 0$.

To define orthogonal polynomials uniquely, we require the leading coefficient to be one, i.e.,

$$P_k(x) = x^k + \dots$$

Theorem: There exist uniquely determined orthogonal polynomials $P_k \in \mathbb{P}_k$ with leading coefficient 1. These polynomials satisfy the 3-term recurrence relation:

$$P_k(x) = (x + a_k)P_{k-1}(x) + b_k P_{k-2}(x), \quad k = 2, 3, \dots$$

with starting values $P_0 = 1$, $P_1 = x + a_1$, where

$$a_k = -\frac{(xP_{k-1}, P_{k-1})}{(P_{k-1}, P_{k-1})}, \quad b_k = -\frac{(P_{k-1}, P_{k-1})}{(P_{k-2}, P_{k-2})}$$

Proof: $\rightsquigarrow$ board