# Conjugate Gradient Method

- direct and indirect methods
- positive definite linear systems
- Krylov sequence
- spectral analysis of Krylov sequence
- preconditioning

# Three classes of methods for linear equations

methods to solve linear system $Ax = b$, $A \in \mathbf{R}^{n \times n}$

- **dense direct** (factor-solve methods)
  - runtime depends only on size; independent of data, structure, or sparsity
  - work well for $n$ up to a few thousand

- **sparse direct** (factor-solve methods)
  - runtime depends on size, sparsity pattern; (almost) independent of data
  - can work well for $n$ up to $10^4$ or $10^5$ (or more)
  - requires good heuristic for ordering

- **indirect** (iterative methods)
  - runtime depends on data, size, sparsity, required accuracy
  - requires tuning, preconditioning, . . .
  - good choice in many cases; only choice for $n = 10^6$ or larger

# Symmetric positive definite linear systems

SPD system of equations

$$Ax = b, \qquad A \in \mathbf{R}^{n\times n}, \qquad A = A^T \succ 0$$

examples

- Newton/interior-point search direction: $\nabla^2\phi(x)\Delta x = -\nabla\phi(x)$
- least-squares normal equations: $(A^TA)x = A^Tb$
- regularized least-squares: $(A^TA + \mu I)x = A^Tb$
- minimization of convex quadratic function $(1/2)x^TAx - b^Tx$
- solving (discretized) elliptic PDE ($e.g.$, Poisson equation)

- analysis of resistor circuit: $Gv = i$
  - $v$ is node voltage (vector), $i$ is (given) source current
  - $G$ is circuit conductance matrix

$$G_{ij} = \begin{cases} \text{total conductance incident on node } i & i = j \\ -(\text{conductance between nodes } i \text{ and } j) & i \neq j \end{cases}$$

# CG overview

- proposed by Hestenes and Stiefel in 1952 (as direct method)
- solves SPD system $Ax = b$
  - in theory ($i.e.$, exact arithmetic) in $n$ iterations
  - each iteration requires a few inner products in $\mathbf{R}^n$, and one matrix-vector multiply $z \rightarrow Az$
- for $A$ dense, matrix-vector multiply $z \rightarrow Az$ costs $n^2$, so total cost is $n^3$, same as direct methods
- get advantage over dense if matrix-vector multiply is cheaper than $n^2$
- with roundoff error, CG can work poorly (or not at all)
- but for some $A$ (and $b$), can get good approximate solution in $\ll n$ iterations

# Solution and error

- $x^\star = A^{-1}b$ is solution
- $x^\star$ minimizes (convex function) $f(x) = (1/2)x^TAx - b^Tx$
- $\nabla f(x) = Ax - b$ is gradient of $f$
- with $f^\star = f(x^\star)$, we have

$$\begin{aligned} f(x) - f^\star &= (1/2)x^TAx - b^Tx - (1/2)x^{\star T}Ax^\star + b^Tx^\star \\ &= (1/2)(x - x^\star)^TA(x - x^\star) \\ &= (1/2)\|x - x^\star\|_A^2 \end{aligned}$$

*i.e.*, $f(x) - f^\star$ is half of squared $A$-norm of error $x - x^\star$

- a relative measure (comparing $x$ to $0$):

$$\tau = \frac{f(x) - f^\star}{f(0) - f^\star} = \frac{\|x - x^\star\|_A^2}{\|x^\star\|_A^2}$$

  (fraction of maximum possible reduction in $f$, compared to $x = 0$)

# Residual

- $r = b - Ax$ is called the **residual** at $x$
- $r = -\nabla f(x) = A(x^\star - x)$
- in terms of $r$, we have

$$\begin{aligned} f(x) - f^\star &= (1/2)(x - x^\star)^T A(x - x^\star) \\ &= (1/2) r^T A^{-1} r \\ &= (1/2)\|r\|^2_{A^{-1}} \end{aligned}$$

- a commonly used measure of relative accuracy: $\eta = \|r\|/\|b\|$
- $\tau \leq \kappa(A)\eta^2$ ($\eta$ is easily computable from $x$; $\tau$ is not)

# Krylov subspace

(a.k.a. controllability subspace)

$$\begin{aligned} \mathcal{K}_k &= \operatorname{span}\{b, Ab, \ldots, A^{k-1}b\} \\ &= \{p(A)b \mid p \text{ polynomial}, \ \deg p < k\} \end{aligned}$$

we define the *Krylov sequence* $x^{(1)}, x^{(2)}, \ldots$ as

$$x^{(k)} = \operatorname*{argmin}_{x \in \mathcal{K}_k} \ f(x) = \operatorname*{argmin}_{x \in \mathcal{K}_k} \ \|x - x^\star\|_A^2$$

the CG algorithm (among others) generates the Krylov sequence

# Properties of Krylov sequence

- $f(x^{(k+1)}) \leq f(x^{(k)})$ (but $\|r\|$ can increase)
- $x^{(n)} = x^\star$ ($i.e.$, $x^\star \in \mathcal{K}_n$ even when $\mathcal{K}_n \neq \mathbf{R}^n$)
- $x^{(k)} = p_k(A)b$, where $p_k$ is a polynomial with $\deg p_k < k$
- less obvious: there is a *two-term recurrence*

$$x^{(k+1)} = x^{(k)} + \alpha_k r^{(k)} + \beta_k (x^{(k)} - x^{(k-1)})$$

  for some $\alpha_k$, $\beta_k$ (basis of CG algorithm)

# Cayley-Hamilton theorem

characteristic polynomial of $A$:

$$\chi(s) = \det(sI - A) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n$$

by Caley-Hamilton theorem

$$\chi(A) = A^n + \alpha_1 A^{n-1} + \cdots + \alpha_n I = 0$$

and so

$$A^{-1} = -(1/\alpha_n)A^{n-1} - (\alpha_1/\alpha_n)A^{n-2} - \cdots - (\alpha_{n-1}/\alpha_n)I$$

in particular, we see that $x^\star = A^{-1}b \in \mathcal{K}_n$

# Spectral analysis of Krylov sequence

- $A = Q\Lambda Q^T$, $Q$ orthogonal, $\Lambda = \mathbf{diag}(\lambda_1, \ldots, \lambda_n)$
- define $y = Q^T x$, $\bar{b} = Q^T b$, $y^\star = Q^T x^\star$
- in terms of $y$, we have

$$\begin{aligned} f(x) = \bar{f}(y) &= (1/2)x^T Q\Lambda Q^T x - b^T Q Q^T x \\ &= (1/2)y^T \Lambda y - \bar{b}^T y \\ &= \sum_{i=1}^n \left((1/2)\lambda_i y_i^2 - \bar{b}_i y_i\right) \end{aligned}$$

so $y_i^\star = \bar{b}_i/\lambda_i$, $f^\star = -(1/2)\sum_{i=1}^n \bar{b}_i^2/\lambda_i$

Krylov sequence in terms of $y$

$$y^{(k)} = \operatorname*{argmin}_{y \in \bar{\mathcal{K}}_k} \bar{f}(y), \qquad \bar{\mathcal{K}}_k = \operatorname{span}\{\bar{b}, \Lambda\bar{b}, \ldots, \Lambda^{k-1}\bar{b}\}$$

$$y_i^{(k)} = p_k(\lambda_i)\bar{b}_i, \qquad \deg p_k < k$$

$$p_k = \operatorname*{argmin}_{\deg p < k} \sum_{i=1}^{n} \bar{b}_i^2 \left((1/2)\lambda_i p(\lambda_i)^2 - p(\lambda_i)\right)$$

$$
\begin{aligned}
f(x^{(k)}) - f^\star &= \bar{f}(y^{(k)}) - f^\star \\
&= \min_{\deg p < k} (1/2) \sum_{i=1}^{n} \bar{b}_i^2 \frac{(\lambda_i p(\lambda_i) - 1)^2}{\lambda_i} \\
&= \min_{\deg p < k} (1/2) \sum_{i=1}^{n} \bar{y}_i^{\star 2} \lambda_i (\lambda_i p(\lambda_i) - 1)^2 \\
&= \min_{\deg q \leq k,\ q(0)=1} (1/2) \sum_{i=1}^{n} \bar{y}_i^{\star 2} \lambda_i q(\lambda_i)^2 \\
&= \min_{\deg q \leq k,\ q(0)=1} (1/2) \sum_{i=1}^{n} \bar{b}_i^2 \frac{q(\lambda_i)^2}{\lambda_i}
\end{aligned}
$$

$$
\begin{aligned}
\tau_k &= \frac{\min_{\deg q \leq k,\ q(0)=1} \sum_{i=1}^n \bar{y}_i^{\star 2} \lambda_i q(\lambda_i)^2}{\sum_{i=1}^n \bar{y}_i^{\star 2} \lambda_i} \\
&\leq \min_{\deg q \leq k,\ q(0)=1} \left( \max_{i=1,\ldots,n} q(\lambda_i)^2 \right)
\end{aligned}
$$

- if there is a polynomial $q$ of degree $k$, with $q(0) = 1$, that is small on the spectrum of $A$, then $f(x^{(k)}) - f^\star$ is small

- if eigenvalues are clustered in $k$ groups, then $y^{(k)}$ is a good approximate solution

- if solution $x^\star$ is approximately a linear combination of $k$ eigenvectors of $A$, then $y^{(k)}$ is a good approximate solution

# A bound on convergence rate

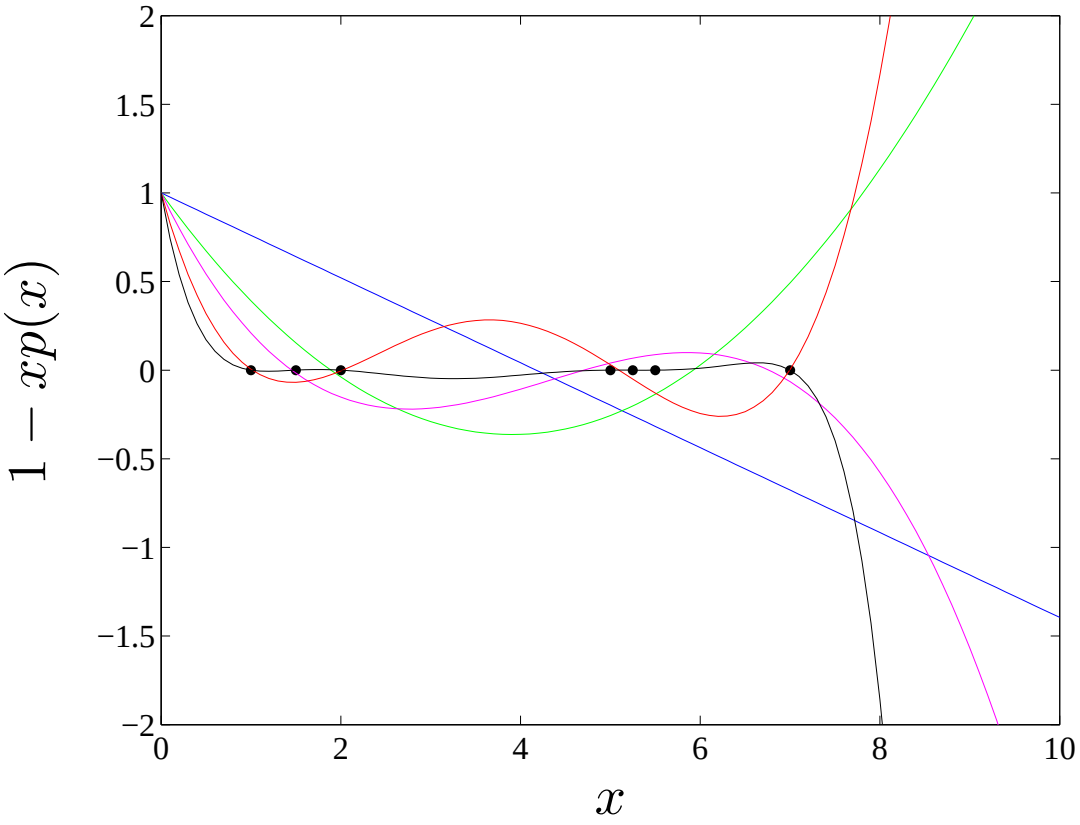
- taking $q$ as Chebyshev polynomial of degree $k$, that is small on interval $[\lambda_{\min}, \lambda_{\max}]$, we get

$$\tau_k \leq \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k, \qquad \kappa = \lambda_{\max}/\lambda_{\min}$$

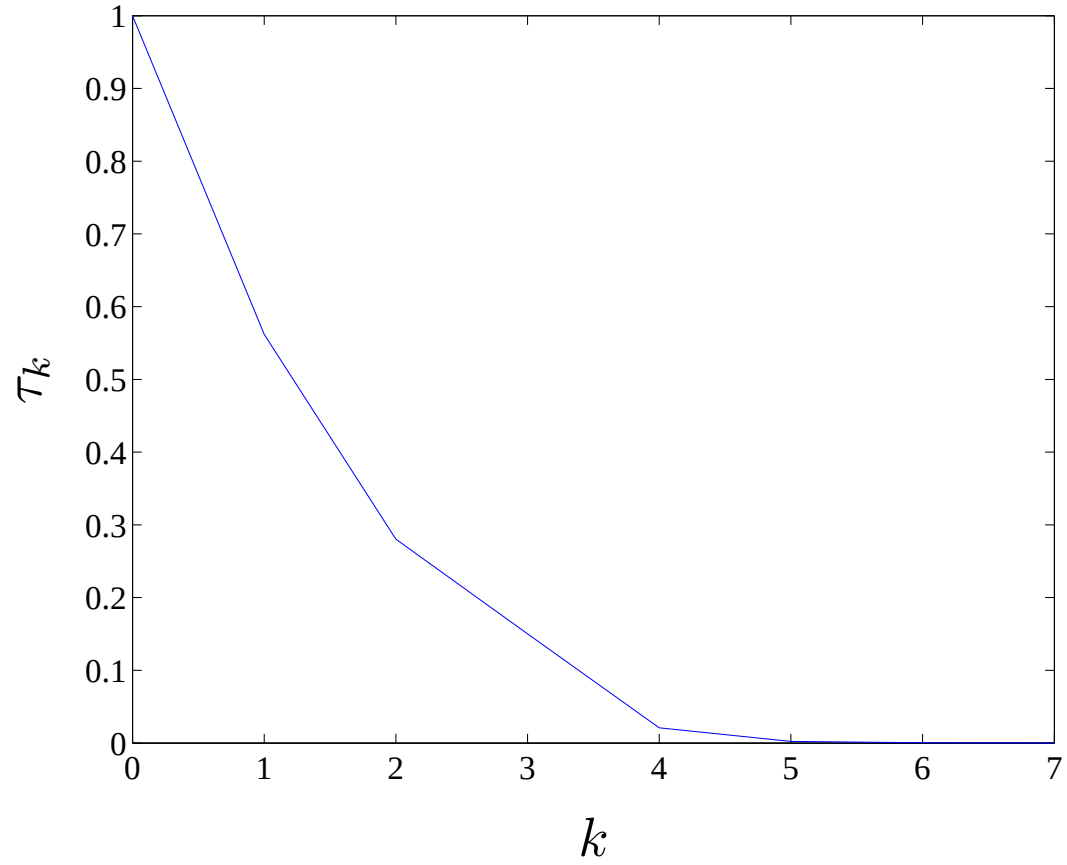
- convergence can be much faster than this, if spectrum of $A$ is spread but clustered

# Small example

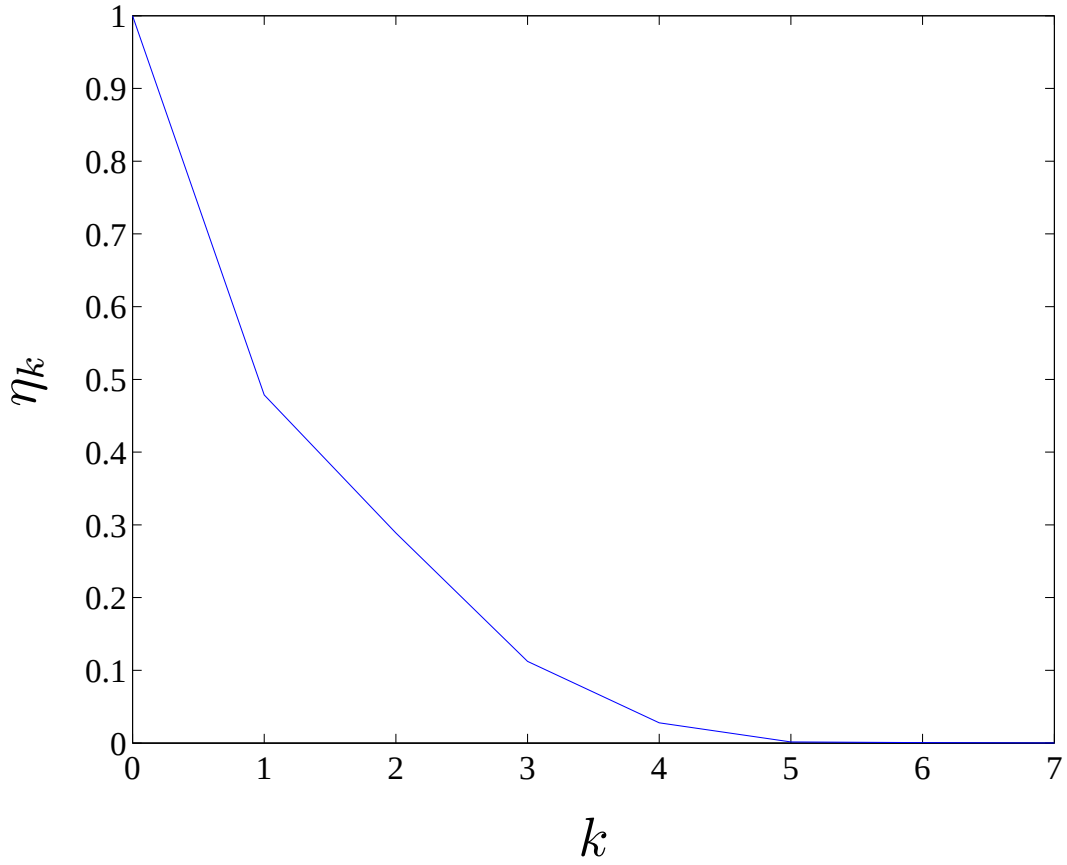
$A \in \mathbf{R}^{7\times 7}$, spectrum shown as filled circles; $p_1, p_2, p_3, p_4$, and $p_7$ shown



# Convergence



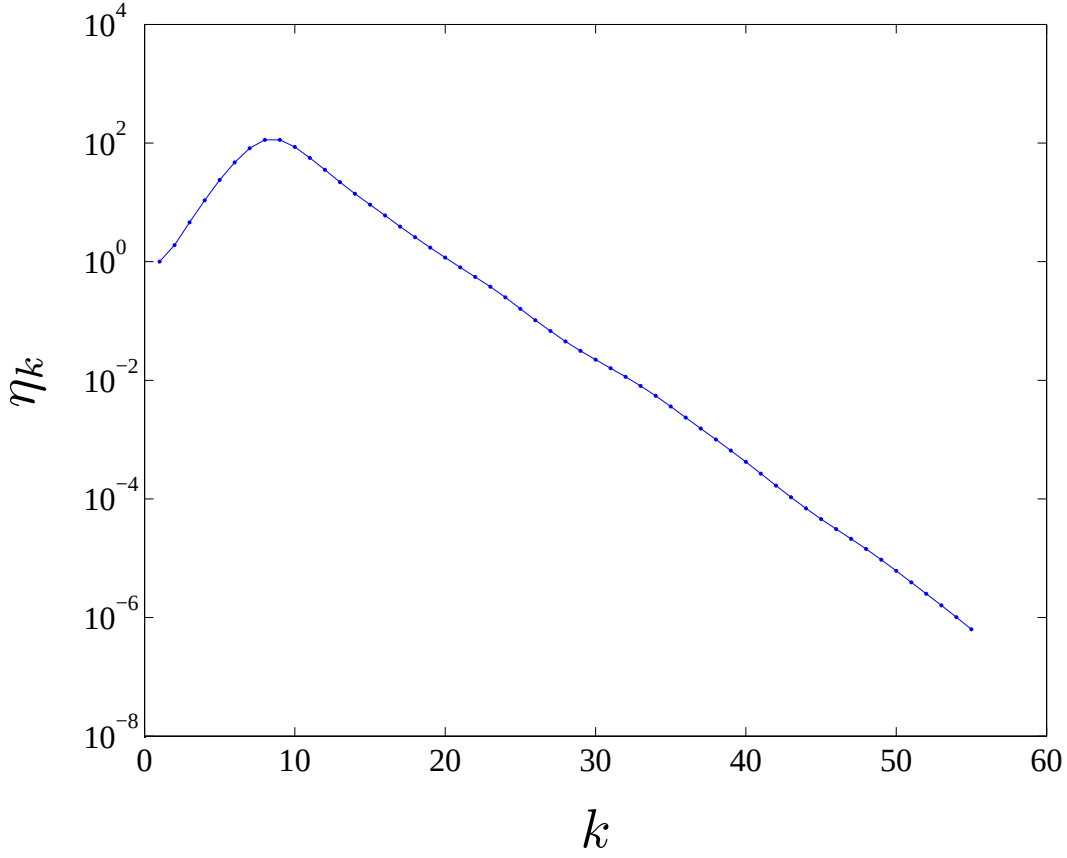
# Residual convergence



# Larger example

- solve $Gv = i$, resistor network with $10^5$ nodes
- average node degree $10$; around $10^6$ nonzeros in $G$
- random topology with one grounded node
- nonzero branch conductances uniform on $[0, 1]$
- external current $i$ uniform on $[0, 1]$
- sparse Cholesky factorization of $G$ requires too much memory

# Residual convergence



# CG algorithm

(follows C. T. Kelley)

$x := 0, \quad r := b, \quad \rho_0 := \|r\|^2$

for $k = 1, \ldots, N_{\max}$

quit if $\sqrt{\rho_{k-1}} \leq \epsilon \|b\|$

if $k = 1$ then $p := r$; else $p := r + (\rho_{k-1}/\rho_{k-2})p$

$w := Ap$

$\alpha := \rho_{k-1}/p^T w$

$x := x + \alpha p$

$r := r - \alpha w$

$\rho_k := \|r\|^2$

# Efficient matrix-vector multiply

- sparse $A$
- structured ($e.g.$, sparse) plus low rank
- products of easy-to-multiply matrices
- fast transforms (FFT, wavelet, . . . )
- inverses of lower/upper triangular (by forward/backward substitution)
- fast Gauss transform, for $A_{ij} = \exp(-\|v_i - v_j\|^2/\sigma^2)$ (via multipole)

# Shifting

- suppose we have guess $\hat{x}$ of solution $x^\star$
- we can solve $Az = b - A\hat{x}$ using CG, then get $x^\star = \hat{x} + z$
- in this case $x^{(k)} = \hat{x} + z^{(k)} = \operatorname*{argmin}_{x \in \hat{x} + \mathcal{K}_k} f(x)$

  ($\hat{x} + \mathcal{K}_k$ is called *shifted Krylov subspace*)
- same as initializing CG alg with $x := \hat{x}$, $r := b - Ax$
- good for 'warm start', $i.e.$, solving $Ax = b$ starting from a good initial guess ($e.g.$, the solution of another system $\tilde{A}x = \tilde{b}$, with $A \approx \tilde{A}$, $b \approx \tilde{b}$)

# Preconditioned conjugate gradient algorithm

- idea: apply CG after linear change of coordinates $x = Ty$, $\det T \neq 0$
- use CG to solve $T^T A T y = T^T b$; then set $x^\star = T^{-1} y^\star$
- $T$ or $M = TT^T$ is called *preconditioner*
- in naive implementation, each iteration requires multiplies by $T$ and $T^T$ (and $A$); also need to compute $x^\star = T^{-1} y^\star$ at end
- can re-arrange computation so each iteration requires one multiply by $M$ (and $A$), and no final solve $x^\star = T^{-1} y^\star$
- called *preconditioned conjugate gradient* (PCG) algorithm

# Choice of preconditioner

- if spectrum of $T^T AT$ (which is the same as the spectrum of $MA$) is clustered, PCG converges fast
- extreme case: $M = A^{-1}$
- trade-off between enhanced convergence, and extra cost of multiplication by $M$ at each step
- goal is to find $M$ that is cheap to multiply, and approximate inverse of $A$ (or at least has a more clustered spectrum than $A$)

# Some generic preconditioners

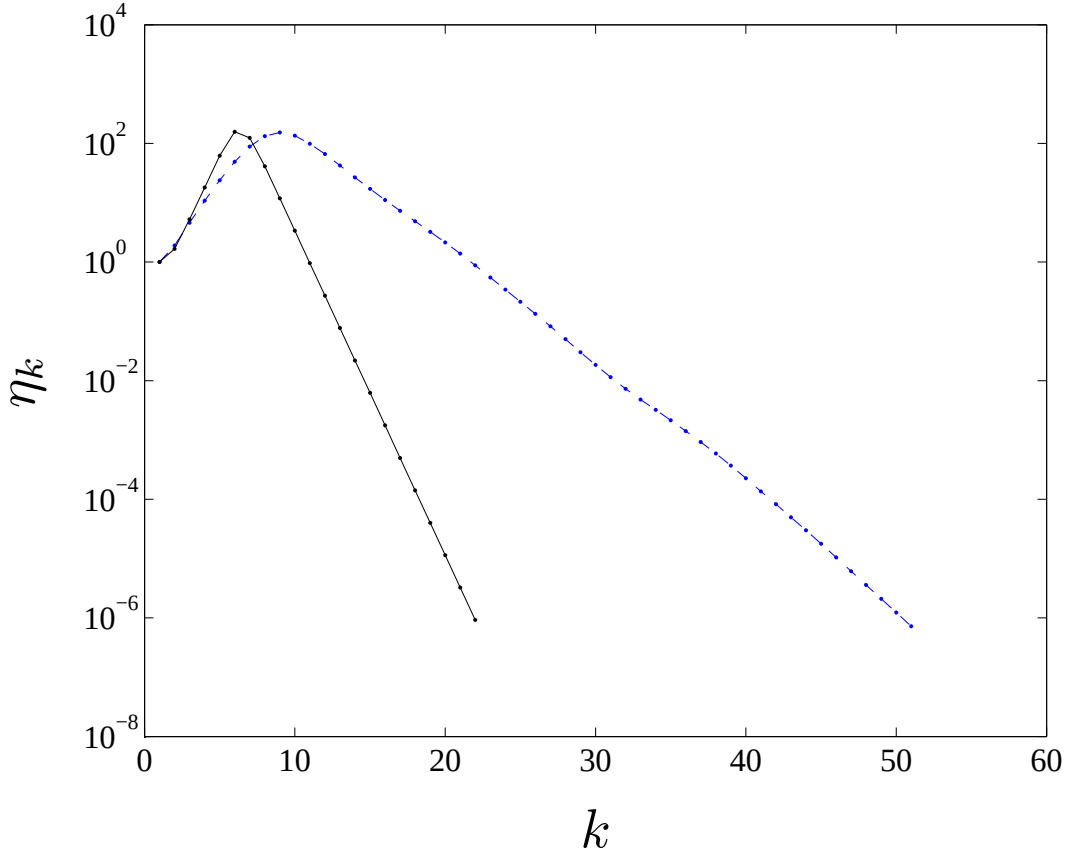
- diagonal: $M = \mathbf{diag}(1/A_{11}, \ldots, 1/A_{nn})$

- incomplete/approximate Cholesky factorization: use $M = \hat{A}^{-1}$, where $\hat{A} = \hat{L}\hat{L}^T$ is an approximation of $A$ with cheap Cholesky factorization
  - compute Cholesky factorization of $\hat{A}$, $\hat{A} = \hat{L}\hat{L}^T$
  - at each iteration, compute $Mz = \hat{L}^{-T}\hat{L}^{-1}z$ via forward/backward substitution

- examples
  - $\hat{A}$ is central $k$-wide band of $A$
  - $\hat{L}$ obtained by sparse Cholesky factorization of $A$, ignoring small elements in $A$, or refusing to create excessive fill-in

# Larger example

residual convergence with and without diagonal preconditioning



# CG summary

- in theory (with exact arithmetic) converges to solution in $n$ steps
  - the bad news: due to numerical round-off errors, can take more than $n$ steps (or fail to converge)
  - the good news: with luck ($i.e.$, good spectrum of $A$), can get good approximate solution in $\ll n$ steps
- each step requires $z \rightarrow Az$ multiplication
  - can exploit a variety of structure in $A$
  - in many cases, never form or store the matrix $A$
- compared to direct (factor-solve) methods, CG is less reliable, data dependent; often requires good (problem-dependent) preconditioner
- but, when it works, can solve extremely large systems