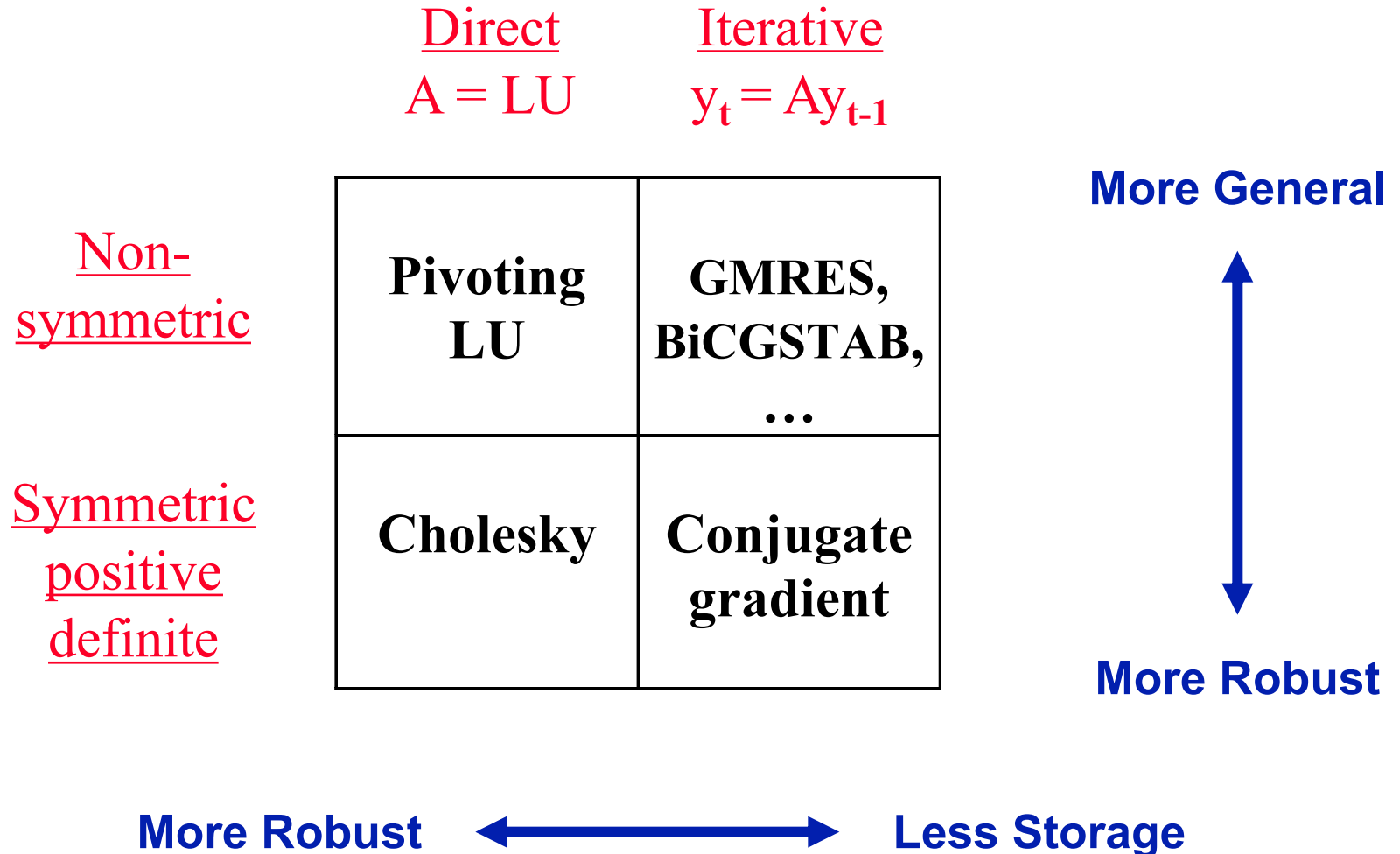


The Landscape of Sparse $Ax=b$ Solvers



Conjugate gradient iteration for $Ax = b$

$x_0 = 0$ approx solution

$r_0 = b$ residual = $b - Ax$

$d_0 = r_0$ search direction

for $t = 1, 2, 3, \dots$

$x_t = x_{t-1} + \dots$ new approx solution

$r_t = \dots$ new residual

$d_t = \dots$ new search direction

Conjugate gradient iteration for $Ax = b$

$x_0 = 0$ approx solution

$r_0 = b$ residual = $b - Ax$

$d_0 = r_0$ search direction

for $t = 1, 2, 3, \dots$

$\alpha_t = \dots$ step length

$x_t = x_{t-1} + \alpha_t d_{t-1}$ new approx solution

$r_t = \dots$ new residual

$d_t = \dots$ new search direction

Conjugate gradient iteration for $Ax = b$

$x_0 = 0$ approx solution

$r_0 = b$ residual = $b - Ax$

$d_0 = r_0$ search direction

for $t = 1, 2, 3, \dots$

$\alpha_t = (r_{t-1}^T r_{t-1}) / (d_{t-1}^T A d_{t-1})$ step length

$x_t = x_{t-1} + \alpha_t d_{t-1}$ new approx solution

$r_t = \dots$ new residual

$d_t = \dots$ new search direction

Conjugate gradient iteration for $Ax = b$

$x_0 = 0$ approx solution

$r_0 = b$ residual = $b - Ax$

$d_0 = r_0$ search direction

for $t = 1, 2, 3, \dots$

$\alpha_t = (r_{t-1}^T r_{t-1}) / (d_{t-1}^T A d_{t-1})$ step length

$x_t = x_{t-1} + \alpha_t d_{t-1}$ new approx solution

$r_t = \dots$ new residual

$\beta_t = (r_t^T r_t) / (r_{t-1}^T r_{t-1})$

$d_t = r_t + \beta_t d_{t-1}$ new search direction

Conjugate gradient iteration for $Ax = b$

$x_0 = 0$ approx solution

$r_0 = b$ residual = $b - Ax$

$d_0 = r_0$ search direction

for $t = 1, 2, 3, \dots$

$\alpha_t = (r_{t-1}^T r_{t-1}) / (d_{t-1}^T A d_{t-1})$ step length

$x_t = x_{t-1} + \alpha_t d_{t-1}$ new approx solution

$r_t = r_{t-1} - \alpha_t A d_{t-1}$ new residual

$\beta_t = (r_t^T r_t) / (r_{t-1}^T r_{t-1})$

$d_t = r_t + \beta_t d_{t-1}$ new search direction

Conjugate gradient iteration

$$\mathbf{x}_0 = \mathbf{0}, \quad \mathbf{r}_0 = \mathbf{b}, \quad \mathbf{d}_0 = \mathbf{r}_0$$

for $t = 1, 2, 3, \dots$

$$\alpha_t = (\mathbf{r}_{t-1}^T \mathbf{r}_{t-1}) / (\mathbf{d}_{t-1}^T \mathbf{A} \mathbf{d}_{t-1}) \quad \text{step length}$$

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \alpha_t \mathbf{d}_{t-1} \quad \text{approx solution}$$

$$\mathbf{r}_t = \mathbf{r}_{t-1} - \alpha_t \mathbf{A} \mathbf{d}_{t-1} \quad \text{residual}$$

$$\beta_t = (\mathbf{r}_t^T \mathbf{r}_t) / (\mathbf{r}_{t-1}^T \mathbf{r}_{t-1}) \quad \text{improvement}$$

$$\mathbf{d}_t = \mathbf{r}_t + \beta_t \mathbf{d}_{t-1} \quad \text{search direction}$$

- One matrix-vector multiplication per iteration
- Two vector dot products per iteration
- Four n -vectors of working storage

Conjugate gradient: Orthogonal sequences

- Krylov subspace: $K_t(A, b) = \text{span}(b, Ab, A^2b, \dots, A^{t-1}b)$
- Conjugate gradient algorithm:
 - for $t = 1, 2, 3, \dots$
 - find $x_t \in K_t(A, b)$
 - such that $r_t = (b - Ax_t) \perp K_t(A, b)$
- Notice $r_t \in K_{t+1}(A, b)$, so $r_t \perp r_k$ for all $k < t$
- Similarly, the “directions” are A-orthogonal:
$$(x_t - x_{t-1})^T \cdot A \cdot (x_k - x_{k-1}) = 0$$
- The magic: Short recurrences. . .
 - A is symmetric \Rightarrow can get next residual and direction from the previous one, without saving them all.

Conjugate gradient: Convergence

- In exact arithmetic, CG converges in n steps
(completely unrealistic!!)
- Accuracy after t steps of CG is related to:
 - consider polynomials of degree t that are equal to 1 at 0.
 - how small can such a polynomial be at all the eigenvalues of A ?
- Thus, eigenvalues close together are good.
- **Condition number:** $\kappa(A) = \|A\|_2 \|A^{-1}\|_2 = \lambda_{\max}(A) / \lambda_{\min}(A)$
- Residual is reduced by a constant factor by $O(\kappa^{1/2}(A))$ iterations of CG.

Preconditioners

- Suppose you had a matrix B such that:
 1. condition number $\kappa(B^{-1}A)$ is small
 2. $By = z$ is easy to solve
- Then you could solve $(B^{-1}A)x = B^{-1}b$ instead of $Ax = b$
 - Actually $(B^{-1/2}AB^{-1/2})(B^{1/2}x) = B^{-1/2}b$, but never mind...
- $B = A$ is great for (1), not for (2)
- $B = I$ is great for (2), not for (1)
- Domain-specific approximations sometimes work
- $B = \text{diagonal of } A$ sometimes works
- Better: blend in some direct-methods ideas. . .

Preconditioned conjugate gradient iteration

$$x_0 = 0, \quad r_0 = b, \quad d_0 = B^{-1} r_0, \quad y_0 = B^{-1} r_0$$

for $t = 1, 2, 3, \dots$

$$\alpha_t = (y_{t-1}^T r_{t-1}) / (d_{t-1}^T A d_{t-1}) \quad \text{step length}$$

$$x_t = x_{t-1} + \alpha_t d_{t-1} \quad \text{approx solution}$$

$$r_t = r_{t-1} - \alpha_t A d_{t-1} \quad \text{residual}$$

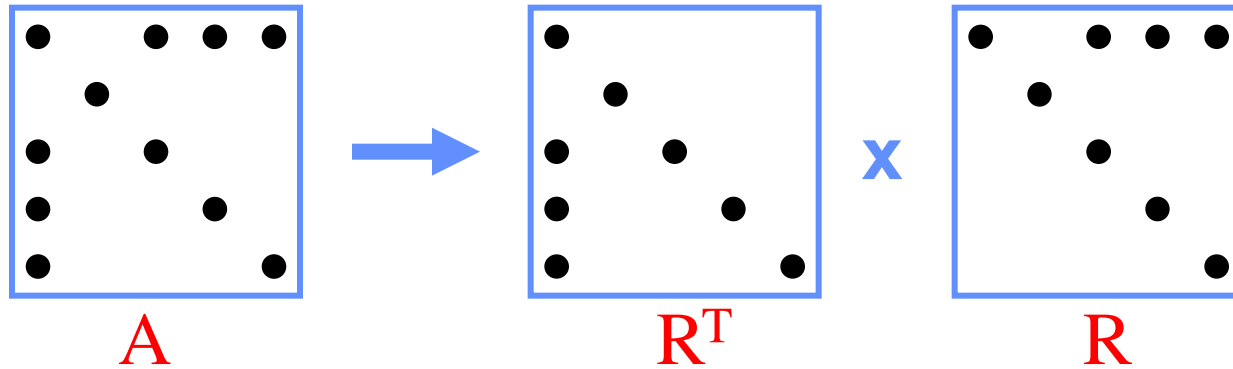
$$y_t = B^{-1} r_t \quad \text{preconditioning solve}$$

$$\beta_t = (y_t^T r_t) / (y_{t-1}^T r_{t-1}) \quad \text{improvement}$$

$$d_t = y_t + \beta_t d_{t-1} \quad \text{search direction}$$

- One matrix-vector multiplication per iteration
- One solve with preconditioner per iteration

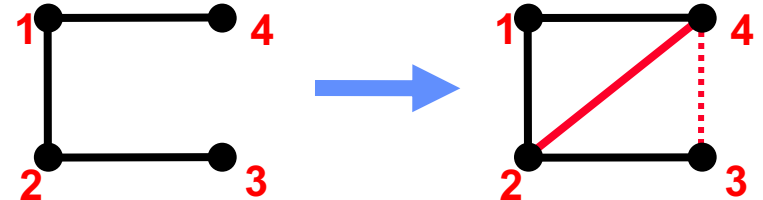
Incomplete Cholesky factorization (IC, ILU)



- Compute factors of A by Gaussian elimination, but ignore fill
- Preconditioner $B = R^T R \approx A$, not formed explicitly
- Compute $B^{-1}z$ by triangular solves (in time $\text{nnz}(A)$)
- Total storage is $O(\text{nnz}(A))$, static data structure
- Either symmetric (IC) or nonsymmetric (ILU)

Incomplete Cholesky and ILU: Variants

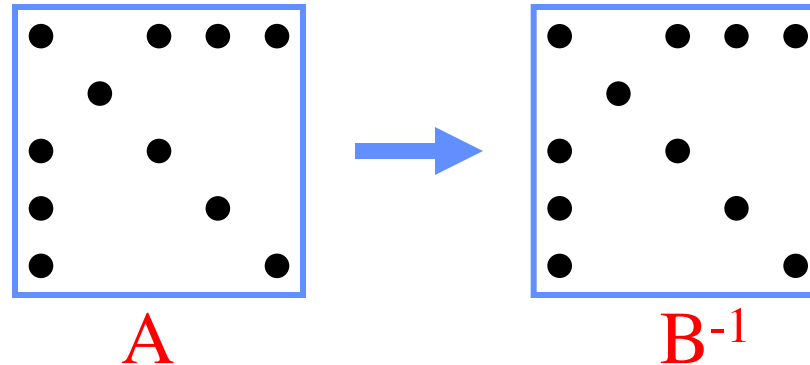
- Allow one or more “levels of fill”
 - unpredictable storage requirements
- Allow fill whose magnitude exceeds a “drop tolerance”
 - may get better approximate factors than levels of fill
 - unpredictable storage requirements
 - choice of tolerance is ad hoc
- Partial pivoting (for nonsymmetric A)
- “Modified ILU” (MIC): Add dropped fill to diagonal of U or R
 - A and $R^T R$ have same row sums
 - good in some PDE contexts
 - preserves null space for Laplacian A



Incomplete Cholesky and ILU: Issues

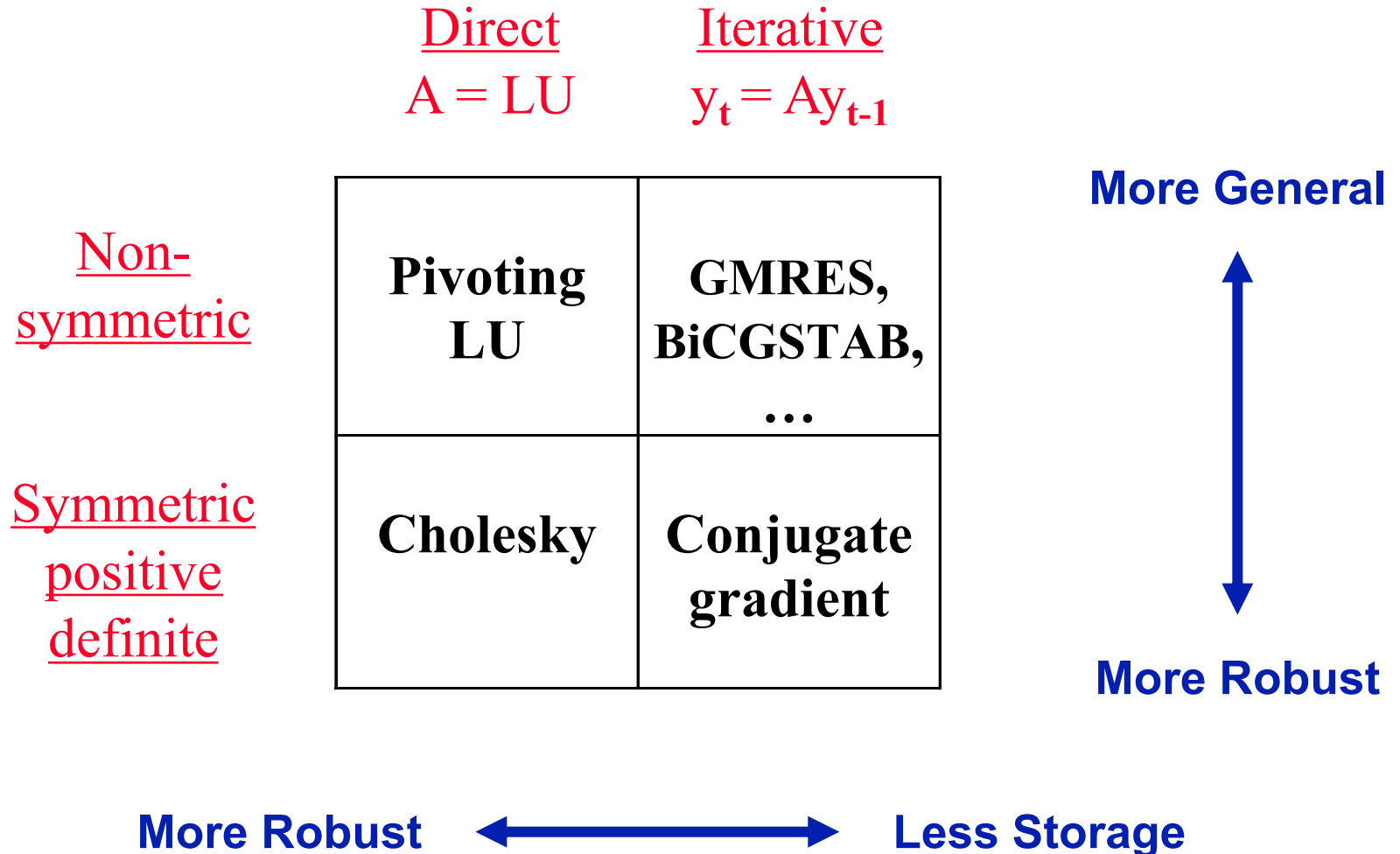
- Choice of parameters
 - **good**: smooth transition from iterative to direct methods
 - **bad**: very ad hoc, problem-dependent
 - **tradeoff**: time per iteration (more fill => more time)
vs # of iterations (more fill => fewer iters)
- Effectiveness
 - condition number usually improves (only) by constant factor
 - except for some PDE problems, and
[new!] with carefully randomized dropping for Laplacians
 - still, often good when tuned for a particular class of problems
- Parallelism
 - triangular solves are not very parallel
 - reordering for parallel triangular solve by graph coloring

Sparse approximate inverses



- Compute $B^{-1} \approx A$ explicitly
- Minimize $\|A B^{-1} - I\|_F$ (in parallel, by columns)
- Variants: factored form of B^{-1} , more fill, . .
- Good: very parallel, seldom breaks down
- Bad: effectiveness varies widely

The Landscape of Sparse $Ax=b$ Solvers



Hierarchy of matrix classes (all real)

- General nonsymmetric
- Diagonalizable
- Normal
- Symmetric indefinite
- Symmetric positive (semi)definite = Factor width n
 - Factor width $2 < k < n$
- Diagonally dominant SPSD = Factor width 2
- Generalized Laplacian = Symm diag dominant M-matrix
- Graph Laplacian

Other Krylov subspace methods

- Nonsymmetric linear systems:
 - **GMRES:**
for $t = 1, 2, 3, \dots$
 find $x_t \in K_t(A, b)$ such that $r_t = (Ax_t - b) \perp K_t(A, b)$
But, no short recurrence \Rightarrow save old vectors \Rightarrow lots more space
(Usually “restarted” every k iterations to use less space.)
 - **BiCGStab, QMR, etc.:**
Two spaces $K_t(A, b)$ and $K_t(A^T, b)$ w/ mutually orthogonal bases
Short recurrences $\Rightarrow O(n)$ space, but less robust
 - Convergence and preconditioning more delicate than CG
 - Active area of current research
- Eigenvalues: **Lanczos** (symmetric), **Arnoldi** (nonsymmetric)