41 - Preconditioning and Incomplete Factorizations

Per-Olof Persson
persson@berkeley.edu

Department of Mathematics
University of California, Berkeley

Math 228A Numerical Solutions of Differential Equations

### Preconditioned Conjugate Gradients

- To keep symmetry, solve \((C^{-1}AC^{-*})C^*x = C^{-1}b\) with \(CC^* = M\).
- Can be written in terms of \(M^{-1}\) only, without reference to \(C\):

  **Algorithm: Preconditioned Conjugate Gradients Method**

  \[
  x_0 = 0, \quad r_0 = b \\
  p_0 = M^{-1}r_0, \quad z_0 = p_0 \\
  \text{for } n = 1, 2, 3, \ldots 
  \begin{align*}
  \alpha_n &= (r_n^T z_n - r_{n-1}^T z_{n-1}) / (p_n^T A p_n - 1) & \text{step length} \\
  x_n &= x_{n-1} + \alpha_n p_{n-1} & \text{approximate solution} \\
  r_n &= r_{n-1} - \alpha_n A p_{n-1} & \text{residual} \\
  z_n &= M^{-1} r_n & \text{preconditioning} \\
  \beta_n &= (r_n^T z_n) / (r_{n-1}^T z_{n-1}) & \text{improvement this step} \\
  p_n &= z_n + \beta_n p_{n-1} & \text{search direction}
  \end{align*}
  \]

### 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 \(O(\text{nnz}(A))\).
- Total storage is \(O(\text{nnz}(A))\), static data structure.
- Either symmetric (IC) or nonsymmetric (ILU).

### Commonly Used Preconditioners

- A preconditioner should “approximately solve” the problem \(Ax = b\).
- **Jacobi preconditioning** - \(M = \text{diag}(A)\), very simple and cheap, might improve certain problems but usually insufficient.
- **Block-Jacobi preconditioning** - Use block-diagonal instead of diagonal. Another variant is using several diagonals (e.g. tridiagonal).
- **Classical iterative methods** - Precondition by applying one step of Jacobi, Gauss-Seidel, SOR, or SSOR.
- **Incomplete factorizations** - Perform Gaussian elimination but ignore fill, results in approximate factors \(A \approx LU\) or \(A \approx R^T R\) (more later).
- **Coarse-grid approximations** - For a PDE discretized on a grid, a preconditioner can be formed by transferring the solution to a coarser grid, solving a smaller problem, then transferring back (**multigrid**).

### 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 \(U (R)\)
  - \(A\) and \(R^T R\) have same row sums
  - Good in some PDE contexts
Incomplete Cholesky and ILU: Issues

- Choice of parameters
  - Good: Smooth transition from iterative to direct methods
  - Bad: Very ad hoc, problem-dependent
  - Trade-off: Time per iteration vs \# of iterations (more fill \(\rightarrow\) more time but fewer iterations)

- Effectiveness
  - Condition number usually improves (only) by constant factor (except MIC for some problems from PDEs)
  - 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

Complexity of Linear Solvers

- Time to solve the Poisson model problem on regular mesh with \(N\) nodes:

<table>
<thead>
<tr>
<th>Solver</th>
<th>1-D</th>
<th>2-D</th>
<th>3-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparse Cholesky</td>
<td>(O(N))</td>
<td>(O(N^{1.5}))</td>
<td>(O(N^2))</td>
</tr>
<tr>
<td>CG, exact arith.</td>
<td>(O(N^2))</td>
<td>(O(N^2))</td>
<td>(O(N^2))</td>
</tr>
<tr>
<td>CG, no precond.</td>
<td>(O(N^2))</td>
<td>(O(N^{1.5}))</td>
<td>(O(N^{1.25}))</td>
</tr>
<tr>
<td>CG, modified IC</td>
<td>(O(N^{1.5}))</td>
<td>(O(N^{1.25}))</td>
<td>(O(N^{1.17}))</td>
</tr>
<tr>
<td>Multigrid</td>
<td>(O(N))</td>
<td>(O(N))</td>
<td>(O(N))</td>
</tr>
</tbody>
</table>