Faster solver for multiple linear systems via Block Conjugate Gradient

NYU’s Courant Institute of Mathematical Sciences

William Lu Mentor: Florian Wechsung July-28th
Question

Find $x$ such that $Ax = b$

- LU Decomposition: high computational cost
- Steepest Descent
- Conjugate Gradient: search directions are orthogonal
Sometimes we want to solve many problems at the same time.

i.e. $A X = B$, $X \in R^{n \times l}$, $B \in R^{n \times l}$

Can we do better than just solving each of them separately? For instance, solving $\ell$ linear systems using Block Conjugate Gradient once instead of using CG $\ell$ times.

We have this hope because of the concept of memory communication cost and information sharing.

Less communication between CPU and memory; Share information between linear systems due to larger Krylov subspace.
CG & Block CG

**Algorithm 1 CG**

1. **Input**: Matrix $A$, a guessed solution $x_0$, a RHS $b$, and a threshold.
2: $r_0 = b - Ax_0$
3: if $r_0$ is smaller than the threshold, return $x_0$.
4: $p_0 = r_0$
5: **while** true **do**
6: $\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$
7: $x_{k+1} = x_k + \alpha_k p_k$
8: $r_{k+1} = r_k - \alpha_k A p_k$
9: if $r_{k+1}$ is smaller than the threshold then
10: exit the loop
11: **else**
12: $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
13: $p_{k+1} = r_{k+1} + \beta_k p_k$
14: End Repeat
15: return $x_{k+1}$

**Algorithm 2 Block CG**

1. **Input**: Matrix $A$, a guessed solution $X_0$, a RHS $B$, and a threshold.
2: $R_0 = B - AX_0$
3: if $R_0$ is smaller than the threshold, return $X_0$.
4: $P_0 = R_0$
5: **while** true **do**
6: $\Lambda_k = (P_k^T A P_k)^{-1}$
7: $X_{k+1} = X_k + P_k \Lambda_k$
8: $R_{k+1} = R_k - A P_k \Lambda_k$
9: if $R_{k+1}$ is smaller than the threshold then
10: exit the loop
11: **else**
12: $\Phi_k = (R_k^T R_k)^{-1}$
13: $P_{k+1} = R_{k+1} + P_k \Phi_k$
14: End Repeat
15: return $X_{k+1}$

- $Ax = b, AX = B$
- $b, B$ : the RHS
- $A$ : the positive def symmetric matrix
- $r_k, R_k$ : k-th residual vectors
- $p_k, P_k$ : k-th search direction
- $x_k, X_k$ : k-th iteration
- $\alpha_k, \Lambda_k$ : k-th coefficient for step magnitude
- $\beta_k, \phi_k$ : k-th coefficient for search direction
• CG's Convergence Estimate:
  \[ e_k = x_k - x \]
  \[ \kappa = \frac{\lambda_n}{\lambda_1}, \text{ condition number} \]
  • Convergence theorem:
    \[ ||e_k||_A \leq 2\left(\frac{\sqrt{\kappa - 1}}{\sqrt{\kappa + 1}}\right)^k ||e_0||_A \]
    \[ ||e_k||_A = (e_k^T A e_k)^{\frac{1}{2}} \]

• BCG's Convergence Estimate:
  \[ E_k = X_k - X \]
  \[ \kappa_\ell = \frac{\lambda_n}{\lambda_\ell}, \text{ where } \lambda_\ell \text{ is the } \ell \text{th largest eigenvalue of } A \]
  • Convergence theorem:
    \[ ||E_k||_A \leq 2\left(\frac{\sqrt{\kappa_\ell - 1}}{\sqrt{\kappa_\ell + 1}}\right)^k ||E_0||_A \]
    \[ ||E_k||_A = (E_k^T A E_k)^{\frac{1}{2}} \]
Preconditioned CG & Preconditioned Block CG

Why preconditioning? What’s the preconditioner?

- \( AX = B \)
- \( M : \) preconditioner
- \( M^{-1} \approx A^{-1} \). \( M^{-1}A \approx I \).
- \( M^{-1}AX = M^{-1}B \)
CG & PCG

Spread of Eigenvalues
Number of iterations for Block is fewer

- matrix of size 3362
- $\ell$ is the block size
- $T_{\{PCG,\ell\}} = \text{Iter}_{\{PCG,1\}} \cdot T_{\{\text{MatVec},1\}} \cdot \ell$
- $T_{\{PBCG,\ell\}} = \text{Iter}_{\{PBCG,\ell\}} \cdot T_{\{\text{MatMat},\ell\}}$
Fewer iterations. Each iteration is cheaper for large block size.

\[ A_{\{n\}}, n \in \{882, 3362, 13122\} \]

- \[ T_{\{PCG, \ell\}} = Iter_{\{PCG, 1\}} \cdot T_{\{MatVec, 1\}} \cdot \ell \]
- \[ T_{\{PBCG, \ell\}} = Iter_{\{PBCG, \ell\}} \cdot T_{\{MatMat, \ell\}} \]

\[ T_{\{MatMat, \ell\}} = T_{\{A, \ell\}} + T_{\{P, \ell\}} \]
Solving $\ell$ linear systems using PBCG once is faster than using PCG $\ell$ times.

$$A_{\{n\}}, n \in \{882, 3362, 13122\}$$

![Graph showing time comparison between PBCG and PCG for solving many RHS](image)
Conclusions

- Solving $\ell$ linear systems using BCG or PBCG once is faster than using CG or PCG $\ell$ times separately. We need fewer iterations, and each iteration is cheaper for BCG when the linear system is large.
- The larger the Block size, the fewer the number of iterations.
- A well chosen preconditioner lead to fewer iterations.
- Using Pseudo Inverse can deal with singular matrices.
Accomplishment

- Fast iterative methods and concepts
- Implementation and testing of four algorithms

Future work

- Block versions of other iterative methods. e.g. GMRES.
- Use specialized routine for the sparse matrix & dense matrix product.
- Integrate solver into PDE code and make code publicly available.


Questions?
Thank You