Lecture 8: Fast Linear Solvers (Part 5)

Conjugate Gradient (CG) Method

- Solve $A\boldsymbol{x} = \boldsymbol{b}$ with A being an $n \times n$ symmetric positive definite matrix.
- Define the quadratic function

$$\phi(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{b}$$

  Suppose $\boldsymbol{x}$ minimizes $\phi(\boldsymbol{x})$, $\boldsymbol{x}$ is the solution to $A\boldsymbol{x} = \boldsymbol{b}$.

- $\nabla\phi(\boldsymbol{x}) = \left(\frac{\partial\phi}{\partial x_1}, \dots, \frac{\partial\phi}{\partial x_n}\right) = A\boldsymbol{x} - \boldsymbol{b}$
- The iteration takes form $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{v}^{(k)}$ where $\boldsymbol{v}^{(k)}$ is the search direction and α_k is the step size.
- Define $\boldsymbol{r}^{(k)} = \boldsymbol{b} - A\boldsymbol{x}^{(k)}$ to be the residual vector.

- Let $\boldsymbol{x}$ and $\boldsymbol{v} \neq \mathbf{0}$ $\phi(\boldsymbol{x}+\alpha\boldsymbol{v})$ be fixed vectors and α a real number variable.

Define:

$$h(\alpha)=\phi(\boldsymbol{x}+\alpha\boldsymbol{v})=\phi(\boldsymbol{x})+2\alpha<\boldsymbol{v},A\boldsymbol{x}-\boldsymbol{b}>+\alpha^2<\boldsymbol{v},A\boldsymbol{x}>$$

$h(\alpha)$ has a minimum when $h'(\alpha)=0$. This occurs when

$$\hat{\alpha}=\frac{\boldsymbol{v}^T(\boldsymbol{b}-A\boldsymbol{x})}{\boldsymbol{v}^{\boldsymbol{T}}A\boldsymbol{v}}.$$

So $h(\hat{\alpha})=\phi(\boldsymbol{x})-\frac{(\boldsymbol{v}^T(\boldsymbol{b}-A\boldsymbol{x}))^2}{\boldsymbol{v}^{\boldsymbol{T}}A\boldsymbol{v}}$.

Suppose $\boldsymbol{x}^*$ is a vector that minimizes $\phi(\boldsymbol{x})$. So $\phi(\boldsymbol{x}+\hat{\alpha}\boldsymbol{v})\geq\phi(\boldsymbol{x}^*)$. This implies $\boldsymbol{v}^T(\boldsymbol{b}-A\boldsymbol{x}^*)=0$. Therefore $\boldsymbol{b}-A\boldsymbol{x}^*=0$.

- For any $\boldsymbol{v} \neq \mathbf{0}$, $\phi(\boldsymbol{x} + \alpha \boldsymbol{v}) < \phi(\boldsymbol{x})$ unless $\boldsymbol{v}^T(\boldsymbol{b} - A\boldsymbol{x}) = 0$ with $\alpha = \frac{\boldsymbol{v}^T(\boldsymbol{b} - A\boldsymbol{x})}{\boldsymbol{v}^T A \boldsymbol{v}}$.
- How to choose the search direction $\boldsymbol{v}$?
  - **Method of steepest descent**: $\boldsymbol{v} = -\nabla\phi(\boldsymbol{x})$
    - Remark: Slow convergence for linear systems

Algorithm.
Let $\boldsymbol{x}^{(0)}$ be initial guess.
for $k = 1,2, \ldots$

$$\boldsymbol{v}^{(k)} = \boldsymbol{b} - A\boldsymbol{x}^{(k-1)}$$

$$\alpha_k = \frac{<\boldsymbol{v}^{(k)}, (\boldsymbol{b} - A\boldsymbol{x}^{(k-1)})>}{<\boldsymbol{v}^{(k)}, A\boldsymbol{v}^{(k)}>}$$

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha_k \boldsymbol{v}^{(k)}$$

end

Steepest descent method when $\frac{\lambda_{max}}{\lambda_{min}}$ is large

- Consider to solve $A\boldsymbol{x} = \boldsymbol{b}$ with $A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$, $\boldsymbol{b} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$ and the start vector $\boldsymbol{v} = \begin{bmatrix} -9 \\ -1 \end{bmatrix}$. Reduction of $||A\boldsymbol{x}^{(k)} - \boldsymbol{b}||_2 < 10^{-4}$.
  - With $\lambda_1 = 1, \lambda_2 = 2$, it takes about 10 iterations
  - With $\lambda_1 = 1, \lambda_2 = 10$, it takes about 40 iterations

- Second approach to choose the search direction $\boldsymbol{v}$?
  - *A-orthogonal approach*: use a set of nonzero direction vectors $\{\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(n)}\}$ that satisfy $< \boldsymbol{v}^{(i)}, A\boldsymbol{v}^{(j)} > = 0$, if $i \neq j$. The set $\{\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(n)}\}$ is called A-orthogonal.

- **Theorem**. Let $\{\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(n)}\}$ be an *A-orthogonal* set of nonzero vectors associated with the symmetric, positive definite matrix A, and let $\boldsymbol{x}^{(0)}$ be arbitrary. Define $\alpha_k = \frac{<\boldsymbol{v}^{(k)}, \left(\boldsymbol{b} - A\boldsymbol{x}^{(k-1)}\right)>}{<\boldsymbol{v}^{(k)}, A\boldsymbol{v}^{(k)}>}$ and $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha_k \boldsymbol{v}^{(k)}$ for $k = 1,2 \dots n$. Then $A\boldsymbol{x}^{(n)} = \boldsymbol{b}$ when arithmetic is exact.

Conjugate Gradient Method

- The conjugate gradient method of Hestenes and Stiefel.
- Main idea: Construct $\{\boldsymbol{v}^{(1)}, \boldsymbol{v}^{(2)} \ldots\}$ during iteration so that the residual vectors $\{\boldsymbol{r}^{(k)}\}$ are mutually orthogonal.

Algorithm of CG Method

Let $\boldsymbol{x}^{(0)}$ be initial guess.

Set $\boldsymbol{r}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^{(0)}$; $\boldsymbol{v}^{(1)} = \boldsymbol{r}^{(0)}$.

for $k = 1,2,\ldots$

$$\alpha_k = \frac{<\boldsymbol{r}^{(k-1)},\boldsymbol{r}^{(k-1)}>}{<\boldsymbol{v}^{(k)},A\boldsymbol{v}^{(k)}>}$$

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha_k \boldsymbol{v}^{(k)}$$

$\boldsymbol{r}^{(k)} = \boldsymbol{r}^{(k-1)} - \alpha_k A\boldsymbol{v}^{(k)}$ // construct residual

$$\rho_k =< \boldsymbol{r}^{(k)}, \boldsymbol{r}^{(k)} >$$

if $\sqrt{\rho_k} < \varepsilon$ **exit.** //convergence test

$$s_k = \frac{<\boldsymbol{r}^{(k)},\boldsymbol{r}^{(k)}>}{<\boldsymbol{r}^{(k-1)},\boldsymbol{r}^{(k-1)}>}$$

$\boldsymbol{v}^{(k+1)} = \boldsymbol{r}^{(k)} + s_k \boldsymbol{v}^{(k)}$ // construct new search direction

end

Remarks

- Constructed $\{\boldsymbol{v}^{(1)}, \boldsymbol{v}^{(2)} \ldots\}$ are pair-wise A-orthogonal.
- Each iteration, there are one matrix-vector multiplication, two dot products and three scalar multiplications.
- Due to round-off errors, in practice, we need more than n iterations to get the solution.
- If the matrix A is ill-conditioned, the CG method is sensitive to round-off errors (CG is not good as Gaussian elimination with pivoting).
- Main usage of CG is as iterative method applied to bettered conditioned system.

CG as Krylov Subspace Method

Theorem. $\boldsymbol{x}^{(k)}$ of the CG method minimizes the function $\phi(\boldsymbol{x})$ with respect to the subspace

$\mathrm{K}_k\left(A, \boldsymbol{r}^{(0)}\right) =$
$span\{\boldsymbol{r}^{(0)}, A\boldsymbol{r}^{(0)}, A^2\boldsymbol{r}^{(0)}, \ldots, A^{k-1}\boldsymbol{r}^{(0)}\}$.

I.e.

$$\phi\left(\boldsymbol{x}^{(k)}\right) = min_{c_i}\phi\left(\boldsymbol{x}^{(0)} + \sum_{i=0}^{k-1} c_i A^i \boldsymbol{r}^{(0)}\right)$$

The subspace $\mathrm{K}_k\left(A, \boldsymbol{r}^{(0)}\right)$ is called Krylov subspace.

Error Estimate

- Define an *energy norm* $||\cdot||_A$ of vector $\boldsymbol{u}$ with respect to matrix A: $||\boldsymbol{u}||_A = (\boldsymbol{u}^T A\boldsymbol{u})^{1/2}$
- Define the error $\boldsymbol{e}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^*$ where $\boldsymbol{x}^*$ is the exact solution.
- **Theorem.**

  $||\boldsymbol{x}^{(k)} - \boldsymbol{x}^*||_A \leq 2(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1})^k ||\boldsymbol{x}^{(0)} - \boldsymbol{x}^*||_A$ with

  $\kappa(A) = cond(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)} \geq 1.$

Remark: Convergence is fast if matrix A is well-conditioned.

Preconditioning

Let the symmetric positive definite matrix M be a preconditioner for A and $LL^T = M$ be its Cholesky factorization. $M^{-1}A$ is better conditioned than A.

The preconditioned system of equations is

$$M^{-1}A\boldsymbol{x} = M^{-1}\boldsymbol{b}$$

or

$$L^{-T}L^{-1}A\boldsymbol{x} = L^{-T}L^{-1}\boldsymbol{b}$$

where $L^{-T} = (L^T)^{-1}$.

Multiply with L^T to obtain

$$L^{-1}AL^{-T}L^T\boldsymbol{x} = L^{-1}\boldsymbol{b}$$

Define: $\tilde{A} = L^{-1}AL^{-T}; \widetilde{\boldsymbol{x}} = L^T\boldsymbol{x}; \widetilde{\boldsymbol{b}} = L^{-1}\boldsymbol{b}$

Now apply CG to $\tilde{A}\widetilde{\boldsymbol{x}} = \widetilde{\boldsymbol{b}}$.

Preconditioned CG Method

- Define $\boldsymbol{z}^{(k)} = M^{-1}\boldsymbol{r}^{(k)}$ to be the preconditioned residual.

Let $\boldsymbol{x}^{(0)}$ be initial guess.
Set $\boldsymbol{r}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^{(0)}$; Solve $M\boldsymbol{z}^{(0)} = \boldsymbol{r}^{(0)}$ for $\boldsymbol{z}^{(0)}$
Set $\boldsymbol{v}^{(1)} = \boldsymbol{z}^{(0)}$
for $k = 1,2, \ldots$

$$\alpha_k = \frac{<\boldsymbol{z}^{(k-1)}, \boldsymbol{r}^{(k-1)}>}{<\boldsymbol{v}^{(k)}, A\boldsymbol{v}^{(k)}>}$$
$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha_k \boldsymbol{v}^{(k)}$$
$$\boldsymbol{r}^{(k)} = \boldsymbol{r}^{(k-1)} - \alpha_k A\boldsymbol{v}^{(k)}$$

solve $M\boldsymbol{z}^{(k)} = \boldsymbol{r}^{(k)}$ for $\boldsymbol{z}^{(k)}$

$$\rho_k =< \boldsymbol{r}^{(k)}, \boldsymbol{r}^{(k)} >$$

if $\sqrt{\rho_k} < \varepsilon$ **exit.** //convergence test

$$s_k = \frac{<\boldsymbol{z}^{(k)}, \boldsymbol{r}^{(k)}>}{<\boldsymbol{z}^{(k-1)}, \boldsymbol{r}^{(k-1)}>}$$
$$\boldsymbol{v}^{(k+1)} = \boldsymbol{r}^{(k)} + s_k \boldsymbol{v}^{(k)}$$

end

Incomplete Cholesky Factorization

- Assume A is symmetric and positive definite. A is sparse.
- Factor $A = LL^T + R, \quad R \neq \mathbf{0}$. L has similar sparse structure as A.

for $k = 1, \dots, n$
  $l_{kk} = \sqrt{a_{kk}}$
  for $i = k + 1, \dots, n$
    $l_{ik} = \frac{a_{ik}}{l_{kk}}$
    for $j = k + 1, \dots, n$
      if $a_{ij} = 0$ **then**
        $l_{ij} = 0$
      else
        $a_{ij} = a_{ij} - l_{ik} l_{kj}$
      endif
    endfor
  endfor
endfor

Jacobi Preconditioning

In diagonal or Jacobi preconditioning

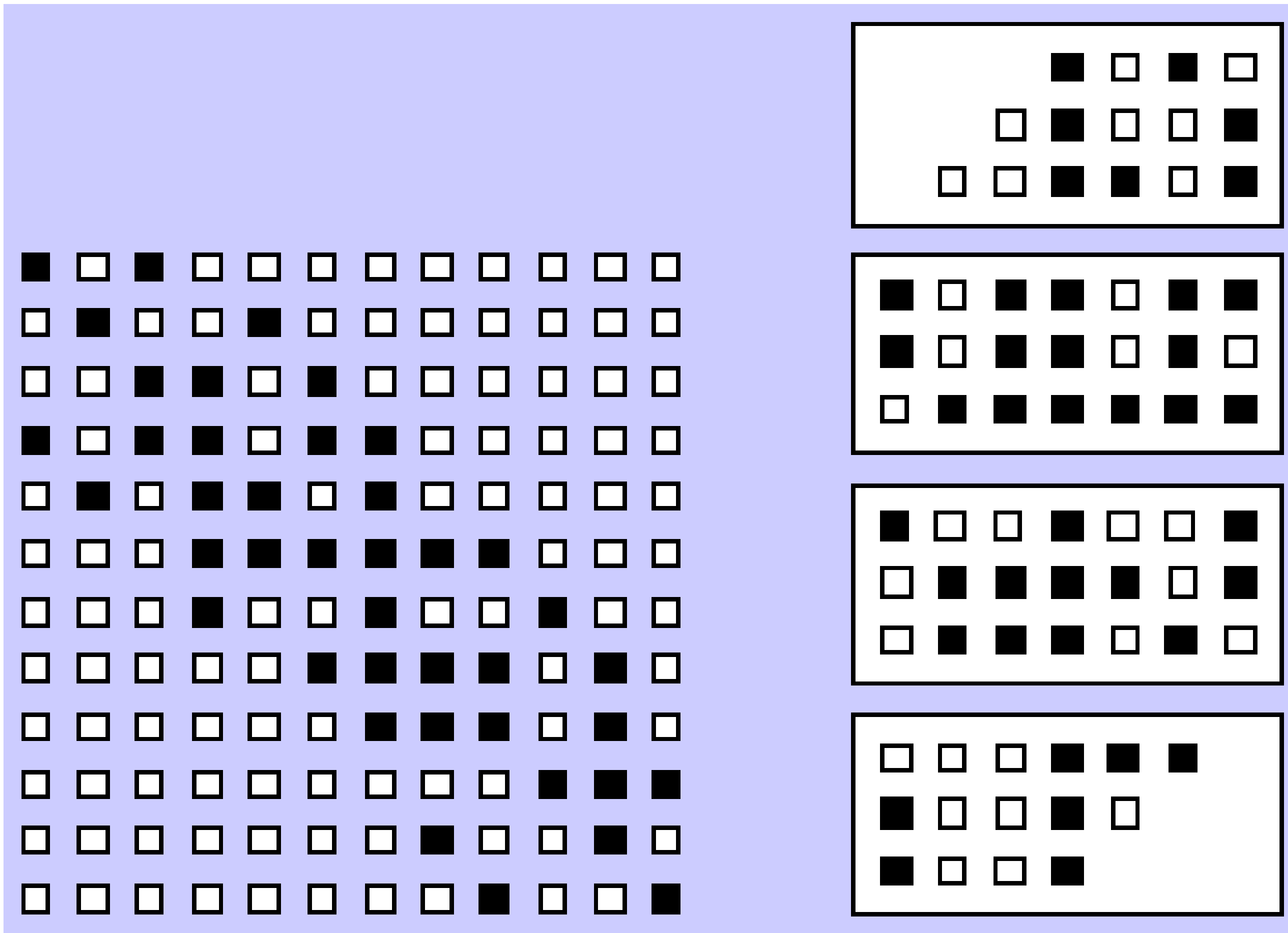
$M = diag(A)$

- Jacobi preconditioning is cheap if it works, i.e. solving $M\boldsymbol{z}^{(k)} = \boldsymbol{r}^{(k)}$ for $\boldsymbol{z}^{(k)}$ almost cost nothing.

References

- CT. Kelley. Iterative Methods for Linear and Nonlinear Equations
- T. F. Chan and H. A. van der Vorst, Approximate and incomplete factorizations, D. E. Keyes, A. Sameh, and V. Venkatakrishnan, eds., *Parallel Numerical Algorithms*, pp. 167-202, Kluwer, 1997
- M. J. Grote and T. Huckle, Parallel preconditioning with sparse approximate inverses, *SIAM J. Sci. Comput.* 18:838-853, 1997
- Y. Saad, Highly parallel preconditioners for general sparse matrices, G. Golub, A. Greenbaum, and M. Luskin, eds., *Recent Advances in Iterative Methods*, pp. 165-199, Springer-Verlag, 1994
- H. A. van der Vorst, High performance preconditioning, *SIAM J. Sci. Stat. Comput.* 10:1174-1185, 1989

Row-wise Block Striped Decomposition of a Symmetrically Banded Matrix



Row decomposition

Parallel CG Algorithm

- Assume a row-wise block-striped decomposition of matrix A and partition all vectors uniformly among tasks.

Let $\boldsymbol{x}^{(0)}$ be initial guess.

Set $\boldsymbol{r}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^{(0)}$; Solve $M\boldsymbol{z}^{(0)} = \boldsymbol{r}^{(0)}$ for $\boldsymbol{z}^{(0)}$

Set $\boldsymbol{v}^{(1)} = \boldsymbol{z}^{(0)}$

for $k = 1,2, \ldots$

$\boldsymbol{g} = A\boldsymbol{v}^{(k)}$ // parallel matrix-vector multiplication

$zr =< \boldsymbol{z}^{(k-1)}, \boldsymbol{r}^{(k-1)} >$ // parallel dot product by MPI_Allreduce

$\alpha_k = \frac{zr}{<\boldsymbol{v}^{(k)},\boldsymbol{g}>}$ // parallel dot product by MPI_Allreduce

$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha_k \boldsymbol{v}^{(k)}$ //

$\boldsymbol{r}^{(k)} = \boldsymbol{r}^{(k-1)} - \alpha_k \boldsymbol{g}$ //

solve $M\boldsymbol{z}^{(k)} = \boldsymbol{r}^{(k)}$ for $\boldsymbol{z}^{(k)}$ // Solve matrix system, can involve additional complexity

$\rho_k =< \boldsymbol{r}^{(k)}, \boldsymbol{r}^{(k)} >$ // MPI_Allreduce

if $\sqrt{\rho_k} < \varepsilon$ **exit.** //convergence test

$zr_{\_n} =< \boldsymbol{z}^{(k)}, \boldsymbol{r}^{(k)} >$ // parallel dot product

$s_k = \frac{zr_{\_n}}{zr}$

$\boldsymbol{v}^{(k+1)} = \boldsymbol{r}^{(k)} + s_k \boldsymbol{v}^{(k)}$

end