5.3. Preconditioning

- Direct solvers: sequential, loosing sparsity
- Iterative solvers: easy parallel and sparse, but possibly slowly convergent
- Combination of both methods (2 variants):
  1. Include preconditioner $M \approx A$ in the form $M^{-1}Ax = M^{-1}b$, such that
     - M is easy to deal with in parallel (reduced approximate direct solver)
     - spectrum of $M^{-1}A$ is much better clustered
  2. Or include preconditioner $M \approx A^{-1}$ in the form $MAx = Mb$, such that
     - M is easy to deal with in parallel (reduced approximate inverse)
     - Spectrum of MA is much better clustered
- General both-sided preconditioning:

$$Ax = b \Leftrightarrow MA(Ky) = Mb \text{ and } Ky = x$$

Overview of Following Preconditioners

- Stationary preconditioners
- ILU preconditioner
- Polynomial preconditioners
- Sparse approximate inverse preconditioners:
  - SPAI for the general nonsymmetric case
  - FSPAI for the SPD case
  - MSPAI using SPAI with probing

5.3.1. Stationary Preconditioners

Consider preconditioner $M \approx A$ in the form $M^{-1}Ax = M^{-1}b$.

- Stationary iteration to splitting $A = M - N$. Convergence depends on
$$\|I - M^{-1}A\| < 1$$
- That is exactly a condition for a good preconditioner: spectrum clustered around 1, $M^{-1}A \approx I$
- If splitting leads to fast convergence, than M is also a good preconditioner.
- In this sense, PCG with stationary preconditioner M can be seen as an accelaration of the stationary method with splitting $M - N$.

Stationary Preconditioners (cont.)

- Jacobi splitting with $D = \mathrm{diag}(A)$ gives Jacobi preconditioner $M := D$.
- Gauss-Seidel splitting $M = D - L$ leads to GS preconditioner.
- Relaxation:

$$x^{(k,\mathrm{new})} := (1-\omega)x^{(k-1)} + \omega x^{(k)}$$

  As convex combination of old and new iterate (Jacobi or GS)
- Symmetrization: first iteration with preconditioner M, second iteration with M^T.

$$M_{\mathrm{new}} := M + M^T - M^T A M$$

- Special case: damped GS $\rightarrow$ SSOR:

$$M_{\mathrm{new}} = \frac{1}{2-\omega}\left(\frac{1}{\omega}D - L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D - L\right)^T$$

5.3.2. ILU Preconditioner

- Apply Gaussian elimination algorithm, but only on allowed pattern $\Rightarrow$ incomplete LU factorization called ILU.
- Reduce in the `for`-loops the indices to the indices with
  - allowed pattern, e.g. ILU(0) for pattern of A
  - values that are not too small, ILUT for ILU with treshold
- Leads to approximate LU factorization

$$A = LU + R, \quad \text{preconditioner} \quad M = LU$$

  with all ignored fill-in entries collected in R.
- MILU (modified ILU): collect all ignored fill-in entries on the related main diagonal elements $\rightarrow$ maintains the row sum or the action on $(1, 1, \ldots, 1)^T$.

Overview Explicit Preconditioners

- ILU and stationary methods use approximations on A itself.
- The resulting preconditioners are given by triangular matrices L, that have to be solved in each iteration step: $L^{-1}x^{(k)}$! Strongly sequential!
- Jacobi easy to parallelize, but slow convergence.
- Question: How to derive preconditioners that lead to fast convergence and are easy to parallelize?
- Until now considered variant 1.; In the following variant 2.:

  Find approximations on $M \approx A^{-1}$. Then the solution of the linear system given by the preconditioner, is only $Mx^{(k)}$, a matrix vector product!

Parallel Preconditioning

- Find preconditioner M, that satisfies three conditions:
  - **(i)** The computation of M is fast in parallel
  - **(ii)** The application Mx in each iteration step is easy in parallel
  - **(iii)** The spectrum AM or MA is clustered $\rightarrow$ fast convergence
- Examples:

  For GS is (i) ok, but not (ii)

  For Jacobi (i) and (ii) ok, but not (iii)

  For ILU (iii) ok, but not (i) and (ii)
- Note that preconditioners also have to be well defined! Zero on diagonals?

5.3.3. Polynomial Preconditioners

- Characteristic polynomial for A:

$$0 = q(A) = \gamma_n A^n + \gamma_{n-1} A^{n-1} + \ldots + \gamma_1 A + \gamma_0 I$$

  Gives polynomial representation for A^{-1} ($\gamma_0 \neq 0$):

$$A^{-1} = \frac{1}{\gamma_0}\left(-\gamma_n A^{n-1} - \gamma_{n-1} A^{n-2} - \ldots - \gamma_1 I\right) = p(A)$$

- Therefore, it makes sense to approximate A^{-1} by a polynomial.
- Better approximation by finding region $S \in \mathbb{R}$ or $\mathbb{C}$ that contains nearly all eigenvalues, and then find polynomial p that is near the inverse in S:

$$\min_{p_n} \|P(A)A - I\|, \quad \min_{p_n(x)} \left(\max_{\lambda \text{ eigenvalue}} |p(\lambda)\lambda - 1| \right)$$

- Solution: Normalized Chebyshev polynomials.

\+ Advant. of pol. prec.: better in parallel (mtx-vec product, BLAS 2)

\- Disadvantage: non-optimal approximation in Krylov subspace

5.3.4. Sparse Approximate Inverses

- Other approach for approximating A^{-1} by norm minimization

$$\min_{M \in \mathcal{P}} \|AM - I\| \qquad \text{over some sparsity pattern } \mathcal{P}.$$

- Choice of the norm?
  - analytic (to allow the explicit solution of this problem)
  - easy to compute (in parallel)

- Optimal norm is Frobenius norm:

$$\|A\|_F^2 := \sum_{i,j=1}^{n} a_{i,j}^2 = \sum_{j=1}^{n} \|A_{\bullet,j}\|^2 = \text{trace}(A^T A)$$

SPAI in Parallel

- First, we choose pattern $\mathcal{P}$ in a static way a priori, e.g. as the pattern of A

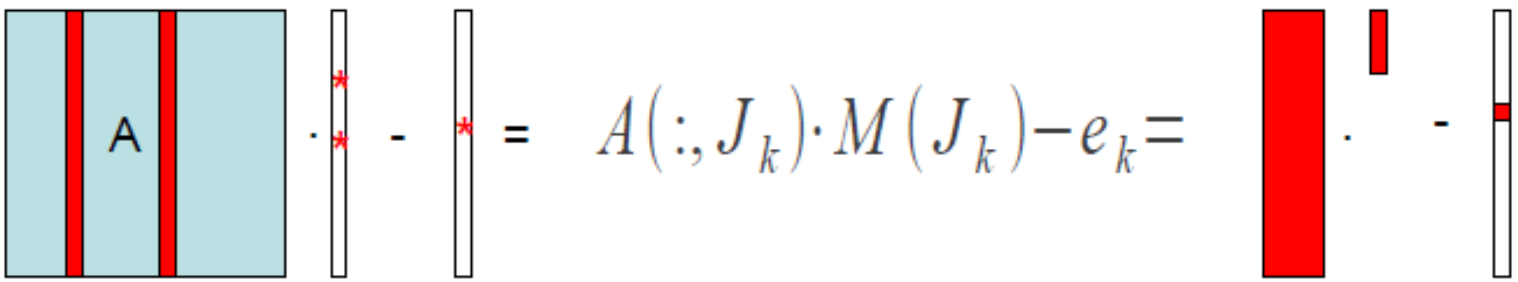
$$\min_{M \in \mathcal{P}} \|AM - I\|_F^2 = \min_{M \in \mathcal{P}} \sum_{k=1}^{n} \|(AM - I)e_k\|_2^2 = \sum_{k=1}^{n} \min_{M_k \in \mathcal{P}_k} \|AM_k - e_k\|_2^2$$

- Hence, to minimize the Frobenius norm, we have to solve n least squares problems in the sparse columns of M!
- This can be done fully in parallel!
- But costs for least squares problems?

SPAI and Least Squares

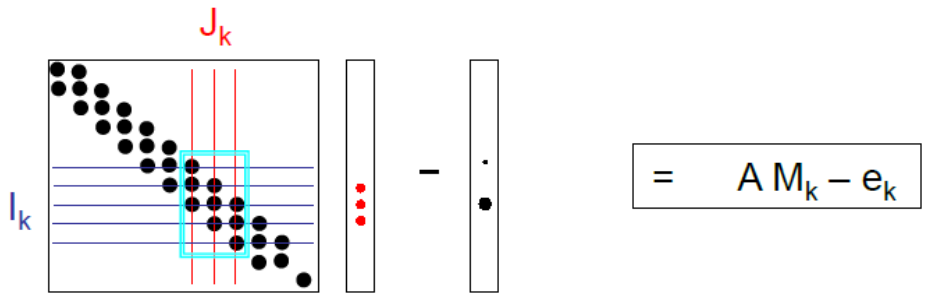
$$\min_{M_k \in \mathcal{P}_k} \|AM_k - e_k\|_2^2 = \min_{M_k \in \mathcal{P}_k} \left\| A \begin{pmatrix} 0 \\ * \\ 0 \\ * \\ 0 \\ 0 \end{pmatrix} - e_k \right\|_2^2 = \min_{M_k \in \mathcal{P}_k} \|A(:, \mathcal{J}_k) M_k(\mathcal{J}_k) - e_k\|_2^2$$

Denote by $\mathcal{J}_k$ the set of allowed indices in the kth column of M.



SPAI and Sparse Least Squares

- $A(:, \mathcal{J}_k)$ is a sparse rectangular matrix. $\mathcal{I}_k$ is index set called *shadow* of $\mathcal{J}_k$.
- Delete superfluous zeros in least squares problem:
  - For index set $\mathcal{J}_k$ in M_k keep only $A(:, \mathcal{J}_k)$
  - In $A(:, \mathcal{J}_k)$ keep only nonzero rows $A(\mathcal{I}_k, \mathcal{J}_k)$



- Hence, reduction of sparse LS to $A(\mathcal{I}_k, \mathcal{J}_k)$:

$$\min_{M_k \in \mathcal{P}_k} \|A(\mathcal{I}_k, \mathcal{J}_k)M(\mathcal{J}_k) - e(\mathcal{I}_k)\|_2^2 = \min_{M_k \in \mathcal{P}_k} \left\|\widetilde{A}\widetilde{M}_k - \widetilde{e}_k\right\|_2^2$$

- Solve small LS problem by Householder *QR* for

$$A(\mathcal{I}_k, \mathcal{J}_k) \to \widetilde{M}_k \to M_k$$

Sparsity Pattern: Static A Priori Choice

- A^{-1} will be no more sparse!
- As a priori choice of a good approximate sparsity pattern for M we can choose the pattern of
  - A^k or $(A^T)^k$
  - $(A^T A)^k A^T$ for some $k = 1, 2$
  - A_ϵ with sparsified A
  - a combination of above
- A_ϵ by sparsification of A: delete all entries with $|A_{ij}| < \epsilon$

Sparsity Pattern: Dynamic Pattern Finding

- Start with thin approximate pattern $\mathcal{J}_k$ for M_k
- Compute optimal column $M_{k,\text{opt}}(\mathcal{J}_k)$ by least squares
- Find new entry j for M_k such that $M_{k,\text{opt}}(\mathcal{J}_k) + \lambda e_j$ has smaller residual in the Frobenius norm.

$$\begin{aligned}\min_{M_k \in \mathcal{P}_k} \left\| A(M_k + \lambda e_j) - e_k \right\|_2^2 &= \min_{M_k \in \mathcal{P}_k} \left\| (AM_k - e_k) + \lambda A e_j \right\|_2^2 = \\ &= \min \left(\|r_k\|_2^2 + 2\lambda (r_k^T A_j) + \lambda^2 \|A_j\|_2^2 \right)\end{aligned}$$

For each possible index candidate j compute

$$\lambda_j = -\frac{r_k^T A_j}{\|A_j\|_2^2}$$

Choose index j with $r_k^T A_j \neq 0$ and $j = \arg\min(\lambda_j)$ since

$$\min = \|r_k\|_2^2 - \frac{(r_k^T A_j)^2}{\|A_j\|_2^2}.$$

Error Estimates

- Assume that for all columns holds

$$\|r_k\| = \|AM_k - e_k\| < \epsilon$$

- Then

$$\begin{aligned} \|AM - I\|_F &\leq \sqrt{n}\epsilon \\ \|AM - I\|_2 &\leq \sqrt{n}\epsilon \\ \|AM - I\|_1 &\leq \sqrt{p}\epsilon \end{aligned}$$

with p being the maximum number of nonzero entries in r_k, $k = 1, \ldots, n$.

Overview Following SPAI Variants

1. Block SPAI
2. Iterative SPAI
3. Factorized SPAI (FSPAI)
4. Modified SPAI (MSPAI)

Further Variants of SPAI

1. Block SPAI

- Partition the matrix M in small blocks (e.g., 2×2 or 3×3) and apply the Frobenius norm minimization with blockwise pattern.
- Advantages:
  - Underlying block structure will also appear in the pattern of $A^{-1} \rightarrow$ improved pattern
  - Block operations are more efficient
  - Less least squares problems to solve

2. Iterative SPAI

- Start with pattern of $A \rightarrow M_1$
- Construct M_2 relative to new matrix AM_1
- Construct M_3 relative to new matrix AM_1M_2
- ...
- Advantage: cheaper (but inferior approximation)

3. Factorized SPAI, FSPAI

- Parallel Preconditioner for SPD matrices
- Approximate the inverse Cholesky factor of $A = L_A^T L_A$

$$A^{-1} = (L_A^T L_A)^{-1} = L_A^{-1} L_A^{-T} \approx L L^T$$

$$L_A^{-1} \approx L \qquad \rightarrow \qquad \min \| L_A L - I \|_F$$

- Normal equations give $A(\mathcal{J}_k, \mathcal{J}_k) L(\mathcal{J}_k) = \alpha e_k$

4. Modified SPAI, MSPAI: Combining Targeting and Probing in classical SPAI formulation.