Efficient Matrix Preconditioners for Black Box Linear Algebra

WILLIAM J. TURNER

Department of Mathematics, North Carolina State University. Raleigh, NC 27965-8205

wjturner@math.ncsu.edu, http://www4.ncsu.edu/~wjturner

This work was done in collaboration with Li Chen, Wayne Eberly, Erich Kaltofen, B. David Saunders, and Gilles Villard. It was supported in part by the NSF under Grants CCR9988177, DMS7392, and INT9726763.

Black Box Linear Algebra

- External view of matrix as linear operator on vector space
- Information gained from applying to series of vectors
- Precondition to reduce matrix problems to computing minimum polynomials
- Delayed matrix multiplication ⇒ efficient matrix-vector products required

Matrix Problems

MINPOLY: Compute the minimum polynomial of A

LINSOLVE0: Compute $w \neq 0$ such that $Aw = 0$

LINSOLVE1: Given A and b, compute x such that $Ax = b$

DET: Compute $\det(A)$

RANK: Compute the rank of A

Preconditioning Problems

Linear Independence Preconditioning

PRECONDIND: The leading $r = \text{rank}(A)$ rows of A' are linearly independent

PRECONDRXR: The $r \times r$ leading principal minor of A' is nonzero

PRECONDSXS: Given $s \leq r$, the $s \times s$ leading principal minor of A' is nonzero

PRECONDGEN: For all $s \leq r$, the $s \times s$ leading principal minor of A' is nonzero

Nilpotent Block Preconditioning

PRECONDNIL: A' has no nilpotent blocks of size greater than 1

Cyclicity Preconditioning

PRECONDCYC: For A nonsingular, A' is nonsingular and cyclic:

$$\text{charpoly}(A') = \text{minpoly}(A')$$

PRECONDCYC-X: The nonsingular part of A' is cyclic:

$$\text{charpoly}(A') = \text{minpoly}(A') \cdot x^l = f(x) \cdot x^k$$

PRECONDSQUFREE-X: PRECONDCYC-X plus f is squarefree

PRECONDCYC: $\text{minpoly}(A') = f(x) \cdot x$ and $\text{charpoly}(A') = f(x) \cdot x^k$ where $f(0) \neq 0$

PRECONDSQUFREE: PRECONDCYCNIL plus f is squarefree

Diagonal Preconditioners

Implementation

- Multiply matrix on left by diagonal matrix with random elements:

$$A' = DA = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} A$$

Properties

- If $d_1, d_2, \ldots, d_n$ are chosen uniformly and independently from a finite subset S of field $\mathbb{F}$, A' is cyclic up to nilpotent blocks with probability at least $1 - n(n-1)/|S|$
- If A nonsingular and $S \subset \mathbb{F} \setminus \{0\}$, A' nonsingular and $\text{charpoly}(A') = \text{minpoly}(A')$

Importance

- For a large field $\mathbb{F}$, D is a sufficient preconditioner for PRECONDCYC
- For nonsingular A, solving MINPOLY for A' gives solution to DET for A

$$\det(A) = (-1)^n \frac{\text{constant term of } \text{minpoly}(A')}{\prod_{i=1}^{n} d_i}$$

Sparse Preconditioners

Implementation

- For given parameters $w_{i,j} \in [0,1],\ 1 \leq i,j \leq n$, say random matrix is chosen from the distribution if the (i,j) element is a uniform randomly chosen nonzero element of field $\mathbb{F}$ (or of a subset of $\mathbb{F}$) with probability $w_{i,j}$ and zero otherwise
- For any given subset S of $\mathbb{F}$ with $\sigma \geq 2$ elements and containing zero and for $\kappa \geq 1$, the distribution defined by

$$w_{i,j} = w_j = \min\{1 - 1/\sigma,\ \kappa(\log n)/j\}$$

  is the *Wiedemann distribution*
- If $A \in \mathbb{F}^{n \times n}$ and R and L^T are chosen from the Wiedemann distribution, precondition A as

$$A' = LAR$$

Properties

- If L and R each have at most $2n(\log n)(1 + \log n) + hn$ nonzero entries, A' has no nilpoetent blocks of size greater than 1 with probability at least

$$\left(1 - \frac{2}{n}\right)\left(\prod_{k=1}^{n}\left(1 - \frac{1}{\sigma^k}\right)\right)^2 - \frac{1}{2^h}$$

Importance

- For small field $\mathbb{F}$, the sparse Wiedemann preconditioner is a sufficient preconditioner for PRECONDNIL
- The Wiedemann sparse preconditioner along with the block Wiedemann algorithm solves LINSOLVE1

Butterfly Network Preconditioners

Implementation

l-dimensional butterfly network: For $n = 2^k$:

- recursive network of butterfly switches with 2^l nodes at each level such that at level m the nodes i and $i + 2^{m-1}$ are merged
- The network has a total of $s = n\log_2(n)/2$ switches
- Figure 1 illustrates a 3-dimensional network with 8 nodes at each level

Generalized butterfly network: For n not a power of two:

- Decompose $n = \sum_{i=1}^{p} 2^{l_i}$ where $l_1 < l_2 < \cdots < l_p$; let $n_i = 2^{l_i}$
- Lay out butterfly networks for each of the n_i blocks
- Connect butterfly networks recursively such that $\sum_{i=1}^{k-1} n_i$ is merged with the far right nodes of n_k as in figure 2
- The network has a depth of $\lceil \log_2(n) \rceil$ and a total of $s \leq n\lceil \log_2(n) \rceil /2$ butterfly switches

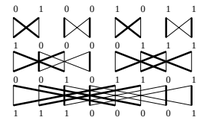

Figure 1: Butterfly network

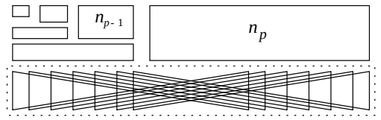

Figure 2: Generalized butterfly network

Preconditioning: Represent the k-th switch of the network by a matrix $\mathcal{E}_k$ such that $\mathcal{E}_k A$ switches the corresponding rows of A. Then, precondition A by

$$A' = \left(\prod_{k=1}^{s} \mathcal{E}_k\right) A$$

Properties

- The l-dimensional butterfly network can switch any r indices $1 \leq i_1 < \cdots < i_r \leq n$ into any desired contiguous block of indices, where wrap around outside preserves contiguity as in figure 1
- The generalized butterfly network can switch any r indices $1 \leq i_1 < \cdots < i_r \leq n$ into the contiguous block $1, 2, \ldots, r$
- The k-th switch can be implemented as the exchange matrix $\begin{bmatrix} 1 & a_k \\ 1 & 1 + a_k \end{bmatrix}$. Then, if $a_1, a_2, \ldots, a_s$ are chosen uniformly and independently from a finite subset S of field $\mathbb{F}$, the leading $r = \text{rank(A)}$ rows of $A' = \left(\prod_{k=1}^{s} \hat{\mathcal{E}}_k(a_k)\right) A$ are linearly independent with probability no less than

$$1 - \frac{r\lceil \log_2(n) \rceil}{|S|} \geq 1 - \frac{n\lceil \log_2(n) \rceil}{|S|}$$

Importance

- For a large field $\mathbb{F}$, the generalized butterfly network preconditioner is a sufficient preconditioner for PRECONDIND
- For a large field $\mathbb{F}$, multiplication on the left and right by generalized butterfly network matrices, $A'' = \left(\prod_{k=1}^{s} \hat{\mathcal{E}}_k(a_k)\right) A \left(\prod_{k=1}^{s} \hat{\mathcal{E}}_k(a_{s+k})\right)^T$ is a sufficient preconditioner for PRECONDGEN
- The preconditioner A'' along with a Monte Carlo test for nonsingularity from LINSOLVE0 gives a Monte Carlo algorithm for solving RANK