Combining implicit restarts and partial reorthogonalization in Lanczos bidiagonalization

Rasmus Munk Larsen
SCCM & SOI-MDI
Stanford University

UC Berkeley, April 2001

Overview

- Introduction
- Golub-Kahan (Lanczos) bidiagonalization and the SVD
- Partial (semi-) orthogonalization (PRO)
- Bidiagonalization with implicit restarts (IR)
- Combining PRO and IR
- Shift strategies for IR, dealing with close singular values
- Performance comparison between PROPACK, LANSO and ARPACK
- Conclusion

The singular value decomposition (SVD)

Computing the SVD of very large sparse matrices has numerous applications in, e.g.,

- Data mining: Information retrieval (LSI), clustering, ...
- Rank deficient and ill-posed (inverse) problems, regularization
- Image and signal processing (Karhunen-Loève transform)
- Data analysis in the physical and medical sciences
- ...

Definition: Let A be a rectangular $m \times n$ matrix with $m \geq n$, then the SVD of A is

$$A \;=\; U\,\Sigma\,V^T \;=\; \sum_{i=1}^{n} \sigma_i\, u_i\, v_i^T \;,$$

where the matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal and

$$\Sigma = \begin{matrix} n \\ m-n \end{matrix} \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} \;,$$

where $\Sigma_1 = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ and

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \cdots = \sigma_n = 0 \;,$$

r is the rank of A.

Equivalent symmetric eigenvalue problems

The SVD is normally computed via an equivalent symmetric eigenvalue problem:

Let the singular value decomposition of the $m \times n$ matrix A be

$$A = U \, \Sigma \, V^T$$

and assume without loss of generality that $m \geq n$. Then

$$V^T (A^T A) V = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2) \ ,$$

$$U^T (A A^T) U = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2, \underbrace{0, \ldots, 0}_{m-n}) \ .$$

Moreover, if $U = \left[\, U_1 \; U_2 \, \right]$ and

$$Y = \frac{1}{\sqrt{2}} \left[\begin{array}{ccc} U_1 & U_1 & \sqrt{2} U_2 \\ V & -V & 0 \end{array} \right] \ , \quad C = \left[\begin{array}{cc} 0 & A \\ A^T & 0 \end{array} \right]$$

then the orthonormal columns of the $(m+n) \times (m+n)$ matrix Y form an eigenvector basis for the 2-cyclic matrix C and

$$Y^T \, C \, Y \; = \; \mathrm{diag}(\sigma_1, \ldots, \sigma_n, -\sigma_1, \ldots, -\sigma_n, \underbrace{0, \ldots, 0}_{m-n}) \ .$$

The Lanczos algorithm and the SVD

When 2k steps of the Lanczos algorithm are applied to the 2-cyclic matrix C with starting vector

$$q_1 = (u_1^T, \underbrace{0, \ldots, 0}_{n})^T \ , \quad \|u_1\| = 1$$

it produces the special (Golub-Kahan) tridiagonal matrix

$$T_{2k} = \begin{pmatrix} 0 & \alpha_1 & & & \\ \alpha_1 & 0 & \beta_2 & & \\ & \beta_2 & 0 & \ddots & \\ & & \ddots & \ddots & \alpha_k \\ & & & \alpha_k & 0 \end{pmatrix} ,$$

and orthonormal vectors

$$q_{2j-1} = (u_j^T, 0)^T \ , \quad q_{2j} = (0, v_j^T)^T \ , \quad j = 1, \ldots, k \ ,$$

such that

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} Q_{2k} = Q_{2k} T_{2k} + \beta_{k+1} \begin{pmatrix} u_{k+1} \\ 0 \end{pmatrix} e_{2k}^T \ .$$

The extreme eigenvalues of T_{2k} converge (usually) rapidly to $\pm$ the largest singular values of A.

Using a symmetric eigensolver as a "black box"

Using a symmetric eigensolver as a "black box" for SVD has certain disadvantages.

Method 0: $A^T A$

- Severe loss of accuracy of small singular values if A is ill-conditioned.
- Fast when $n \ll m$ since only Lanczos vectors of length n need to be stored.

Method 1: $C = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$

- Lanczos vectors have length $m + n \Rightarrow$ Waste of memory and unnecessary work in reorthogonalization.
- Ritz values converge to pairs of $\pm\sigma_i \Rightarrow$ Twice as many iterations are needed.

To (almost) get the best of both worlds: Combine **Lanczos bidiagonalization** (LBD) with the efficient **semi-orthogonalization** and implicitly restarted Lanczos algorithms developed for the symmetric eigenvalue problem.

Algorithm Bidiag1 **(Paige & Saunders)**

1. Choose a starting vector $p_0 \in \mathbb{R}^m$, and let $\beta_1 = \|p_0\|, \;\; u_1 = p_0/\beta_1$ and $v_0 \equiv 0$
2. **for** $i = 1, \; 2, \ldots, k$ **do**
   $r_i = A^T u_i - \beta_i v_{i-1}$, $r_i =$ reorth(r_i)
   $\alpha_i = \|r_i\|, \quad v_i = r_i/\alpha_i$
   $p_i = Av_i - \alpha_i u_i$, $p_i =$ reorth(p_i)
   $\beta_{i+1} = \|p_i\|, \quad u_{i+1} = p_i/\beta_{i+1}$
   end

After k steps we have the decomposition:

$$\begin{aligned} AV_k &= U_{k+1}B_k \\ A^T U_{k+1} &= V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T \end{aligned}$$

where V_j and U_{j+1} have orthonormal columns and

$$B_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_k \\ & & & \beta_{k+1} \end{pmatrix}$$

The largest singular values of B_k converge (usually) rapidly to the largest singular values of A.

Partial reorthogonalization and Lanzos bidiagonalization

As argued above, Bidiag1 is equivalent to performing $2k+1$ steps of symmetric Lanczos on the matrix

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$$

with starting vector $(u_1, 0, \ldots, 0)^T \in \mathbb{R}^{m+n}$. Using Horst Simon's (1984) result about semiorthogonality for symmetric Lanczos gives us the following:

Corollary: Define the levels of orthogonality in Bidiag1 by $\mu_{ij} \equiv u_i^T u_j$ and $\nu_{ij} \equiv v_i^T v_j$. If

$$\max_{1 \le i,j \le k+1} |\mu_{ij}| \quad \le \quad \sqrt{\mathbf{u}/(2k+1)} \qquad \text{for } i \ne j \ ,$$

$$\max_{1 \le i,j \le k} |\nu_{ij}| \quad \le \quad \sqrt{\mathbf{u}/(2k+1)} \qquad \text{for } i \ne j \ ,$$

then

$$\tilde{U}_{k+1}^T A \tilde{V}_k = B_k + O(\mathbf{u}\|A\|) \ ,$$

where $U_{k+1} = \tilde{U}_{k+1}\tilde{J}_{k+1}$ and $V_k = \tilde{V}_k \tilde{K}_k$ are the compact QR-factorizations of U_{k+1} and V_k.

Therefore $\sigma(B_k)$ are Ritz values for A within $O(\mathbf{u}\|A\|)$.

The "ω-recurrences" for LBD

In finite precision arithmetic:

$$\begin{aligned}\alpha_j v_j &= A^T u_j - \beta_j v_{j-1} + f_j \\ \beta_{j+1} u_{j+1} &= A\, v_j - \alpha_j u_j + g_j \ ,\end{aligned}$$

where f_j and g_j represent round-off errors.

It is simple to show that $\mu_{j+1,i}$ and ν_{ji} satisfy the coupled recurrences:

$$\begin{aligned}\beta_{j+1}\mu_{j+1,i} &= \alpha_i \nu_{ji} + \beta_i \nu_{j,i-1} - \alpha_j \mu_{ji} \\ &\quad + u_i^T g_j - v_j^T f_i \ , \qquad (1) \\ \alpha_j \nu_{ji} &= \beta_{i+1}\mu_{j,i+1} + \alpha_i \mu_{ji} - \beta_j \nu_{j-1,i} \\ &\quad - u_j^T g_i + v_i^T f_j \ , \qquad (2)\end{aligned}$$

where $\mu_{ii} = \nu_{ii} = 1$ and $\mu_{0i} = \nu_{0i} \equiv 0$ for $1 \leq i \leq j$.

These recurrences were derived independently by Larsen 1998 and Simon & Zha 1997.

Partial reorthogonalization: *Use the recurrences to monitor the size of $\mu_{j+1,i}$ and ν_{ji}. Reorthogonalize only when necessary.*

Bounding the round-off terms

We can bound the size of the round-off term

$$
\begin{aligned}
|u_i^T g_j - v_j^T f_i| &\leq \|g_j\| + \|f_i\| \\
&\leq 4\,\mathbf{u}\,((\alpha_j^2 + \beta_{j+1}^2)^{1/2} + (\alpha_i^2 + \beta_i^2)^{1/2}) + \epsilon_{MV} \\
&\equiv \tau
\end{aligned}
$$

Round-off from matrix-vector multiply ϵ_{MV} is estimated conservatively: $\epsilon_{MV} \leq \mathbf{u}\,(\bar{n} + \bar{m})\,\|A\|$, where $\bar{n}$ $(\bar{m})$ is the maximum number of non-zeros per row (column) in A.

Conservative updating rules $\nu_{j-1,i} \rightarrow \nu_{ji}$ and $\mu_{ji} \rightarrow \mu_{j+1,i}$:

$$
\begin{aligned}
\nu'_{ji} &= \beta_{i+1}\mu_{j,i+1} + \alpha_i\mu_{ji} - \beta_j\nu_{j-1,i} \\
\nu_{ji} &= (\nu'_{ji} + \mathrm{sign}(\nu'_{ji})\tau)/\alpha_j
\end{aligned}
$$

$$
\begin{aligned}
\mu'_{j+1i} &= \alpha_i\nu_{ji} + \beta_i\nu_{j,i-1} - \alpha_j\mu_{ji} \\
\mu_{j+1,i} &= (\mu'_{j+1,i} + \mathrm{sign}(\mu'_{j+1,i})\tau)/\beta_{j+1}
\end{aligned}
$$

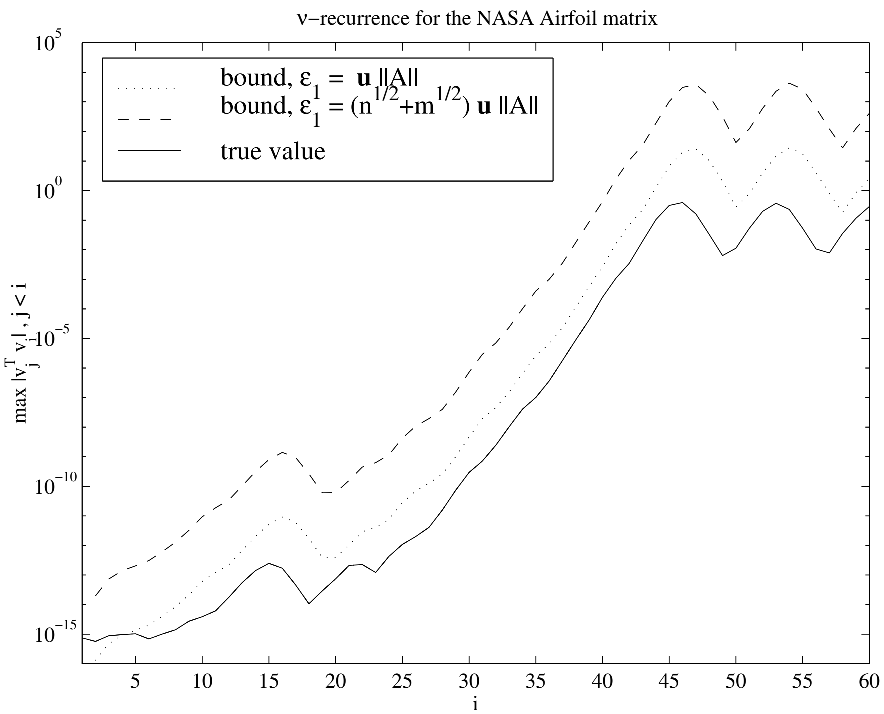
Outline of Algorithm LBDPRO

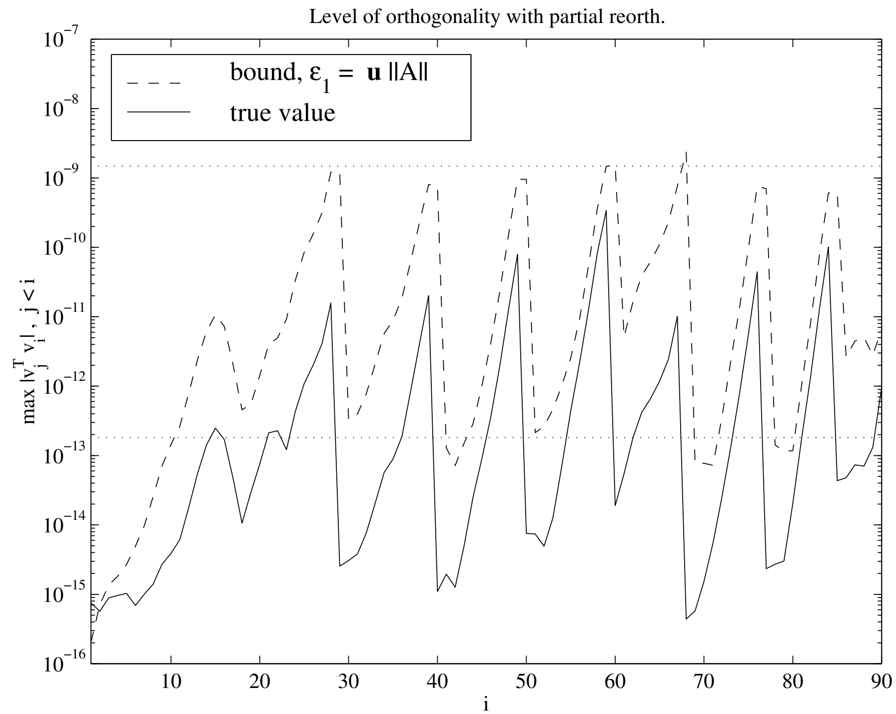
Lanczos bidiagonalization (Bidiag1) with Partial Reorthogonalization:

```
force = FALSE
for j = 1, 2, ..., k do
    α_j v_j = A^T u_j − β_j v_{j−1}
    Update ν_{j−1,i} → ν_{ji}
    if max_{1≤i<j} |ν_{ji}| > tol or force
        Reorthogonalize v_j
        force = (max_{1≤i<j} |ν_{ji}| > tol)
    end
    β_{j+1} u_{j+1} = A v_j − α_j u_j
    Update μ_{ji} → μ_{j+1,i}
    if max_{1≤i<j+1} |μ_{j+1,i}| > tol or force
        Reorthogonalize u_{j+1}
        force = (max_{1≤i<j+1} |μ_{j+1,i}| > tol)
    end
end
```

- The variable "force" causes extra reorthogonalizations, which are necessary due to the coupling between ν_{ji} and $\mu_{j+1,i}$.

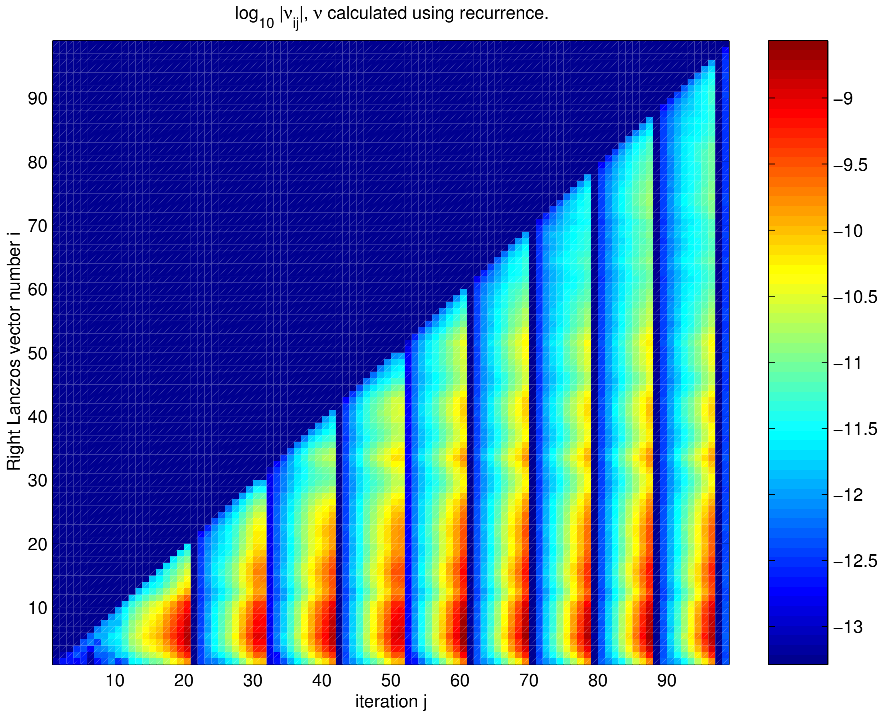
Illustration of recurrences

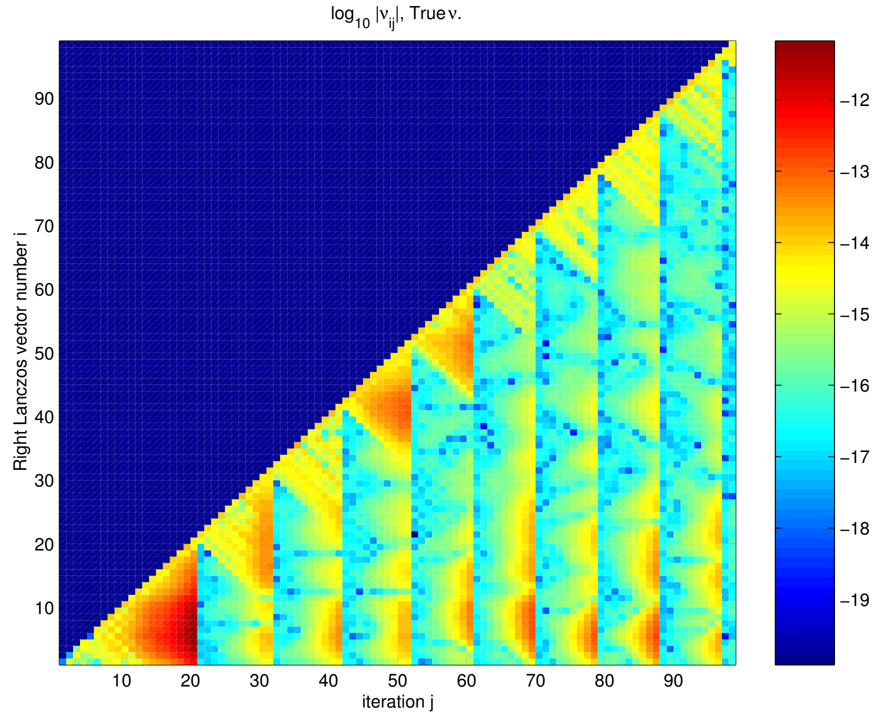




Partial reorthogonalization (next slide) reduced the work compared to full reorthogonalization from $10100 \longrightarrow 926$ inner products!

Estimated level of orthogonality...and the truth





Iterative SVD algorithm LBDSVD

1. Input N, ϵ_{tol}, and $k_{\max}$
2. Set $k = \min(2N, k_{\max})$
3. Use LBDPRO to extend the bidiagonalization to

$$AV_k = U_{k+1}B_k$$

4. Compute the Ritz values $\theta_1, \theta_2, \ldots, \theta_k$ and error bounds

$$e = \|u_{k+1}\|(|p_{k+1,1}|, |p_{k+1,2}|, \ldots, |p_{k+1,k}|) \ ,$$

   where

$$B_k = P_{k+1} \ \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_k) \, Q_k^T \ ,$$

   is the SVD of B_k and $(p_{k+1,1}, p_{k+1,2}, \ldots, p_{k+1,k})$ is the last row of P_{k+1}.
5. Refine error bounds using the gap-theorem
6. If $e_1, e_2, \ldots, e_N < \epsilon_{tol}$ then **goto 8**
7. If $k < k_{\max}$ then increase k and **goto 3** else **fail**
8. If singular vectors are needed then compute a full SVD of

$$B_k = P_{k+1} \ \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_k) \, Q_k^T \ ,$$

   and form Ritz vectors $\bar{U} = U_{k+1}P_{k+1}(:, 1\!:\!N)$, and $\bar{V} = V_kQ_k(:, 1\!:\!N)$.

Implicitly restarted bidiagonalization

Following Björck, Grimme and Van Dooren (1995), we notice that after $k+p$ steps of Bidiag1 we have

$$(AA^T)U_{k+p+1} = U_{k+p+1}(B_{k+p}B_{k+p}^T) + \alpha_{k+p+1}Av_{k+p+1}e_{k+p+1}$$

Here one could use the implicitly restarted Lanczos algorithm of Sorensen et al. on AA^T, which applies implicitly shifted QR steps to $T_{k+p} = B_{k+p}B_{k+p}^T$. However, a more stable approach is to apply Golub-Kahan SVD steps to B_{k+p} directly:

1. First compute a Givens rotation $G_l^{(1)}$ such that

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix} \begin{bmatrix} \alpha_1^2 - \mu^2 \\ \alpha_1\beta_1 \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix} .$$

2. Then bring $G_l^{(1)}B_{k+p}$ back to bidiagonal form by applying $k-1$ additional rotations from the left and from the right to "chase the bulge":

$$B_{k+p}^+ = G_l^{(k)} \cdots G_l^{(2)} G_l^{(1)} B_{k+p} G_r^{(1)} \cdots G_r^{(k-1)} = Q_l B_{k+p} Q_r^T .$$

Implicitly restarted bidiagonalization

3. By applying the rotations to the left and right Lanczos vectors, we can recover a bidiagonalization

$$AV^+_{k+p-1} = U^+_{k+p}B^+_{k+p-1} \ ,$$

were

$$\begin{aligned} U^+_{k+p} &= U_{k+p+1}Q_l(:, 1:k+p) \ , \\ V^+_{k+p-1} &= V_{k+p}Q_r(:, 1:k+p-1) \ . \end{aligned}$$

The updated quantities are what would have been uptained from $k+p-1$ steps of Bidiag1 with starting vector

$$u^+_1 = (AA^T - \mu^2 I)u_1 \ .$$

If this algorithm is repeated for p shifts $\mu_1, \mu_2, \ldots, \mu_p$ we obtain a bidiagonalization

$$AV^+_k = U^+_{k+1}B^+_k \ ,$$

corresponding to the starting vector

$$u^+_1 = \prod_{i=1}^{p}(AA^T - \mu_i^2 I)u_1 \ .$$

Implicitly restarted SVD algorithm LBDIR

1. Input k, p, and ϵ_{tol}
2. Use LBDPRO to extend the bidiagonalization to

$$AV_{k+p} = U_{k+p+1} B_{k+p}$$

3. Compute the Ritz values $\theta_1, \theta_2, \ldots, \theta_{k+p}$ and error bounds

$$e = \|u_{k+p+1}\| (|p_{k+p+1,1}|, |p_{k+p+1,2}|, \ldots, |p_{k+p+1,k+p}|) \ ,$$

   where

$$B_{k+p} = P_{k+p+1} \ \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_{k+p}) \ Q_{k+p}^T \ ,$$

   is the SVD of B_{k+p} and $(p_{k+p+1,1}, p_{k+p+1,2}, \ldots, p_{k+p+1,k+p})$ is the last row of P_{k+p+1}
4. Refine error bounds using the gap-theorem
5. If $e_1, e_2, \ldots, e_k < \epsilon_{tol}$ then **goto 8**
6. Select p shift $\mu_1, \mu_2, \ldots, \mu_p$
7. Apply p restarting steps to obtain

$$AV_k^+ = U_{k+1}^+ B_k^+ \ ,$$

   goto 2

8. If singular vectors are needed then compute a full SVD of

$$B_{k+p} = P_{k+p+1}\ \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_{k+p})\, Q_{k+p}^T\ ,$$

and form Ritz vectors $\bar{U} = U_{k+p+1} P_{k+p+1}(:, 1:k)$, and $\bar{V} = V_{k+p} Q_{k+p}(:, 1:k)$.

Setup for Numerical Experiments

Test matrices from Matrix Market:

Name	m	n	nnz(A)
WELL1850	1850	712	8758
ILLC1850	1850	712	8758
TOLS4000	4000	4000	8784
MHD4800A	4800	4800	102252
AF23560	23560	23560	460598
FIDAPM11	90449	90449	1921955

Software:

Algorithm	Subroutine
Lanczos bidiagonalization with PRO	LBDSVD
Lanczos bidiagonalization with PRO & IR	LBDIR
Lanczos with PRO on A^TA	LANSO
Lanczos with PRO on C	LANSO
IRL on A^TA	ARPACK
IRL on C	ARPACK

Hardware and software used:

- 600 MHz Pentium III CPU, 512 KB L2 cache, IEEE arithmetic
- RedHat GNU/Linux 7.1, GNU 2.96-79 compiler suite
- ASCI Red BLAS by Greg Henry, LAPACK 3.0 from Netlib

Is it stable?

The fundamental question is: Are the Lanczos vectors still semiorthogonal after a restart?

Before applying the shifts we have that

$$V_{k+p}^T V_{k+p} = I + E \ , \quad |E| < \sqrt{\frac{\mathbf{u}}{2k+1}}$$

and similarly for U_{k+p+1}.

Therefore the updated vectors satisfy the following bound

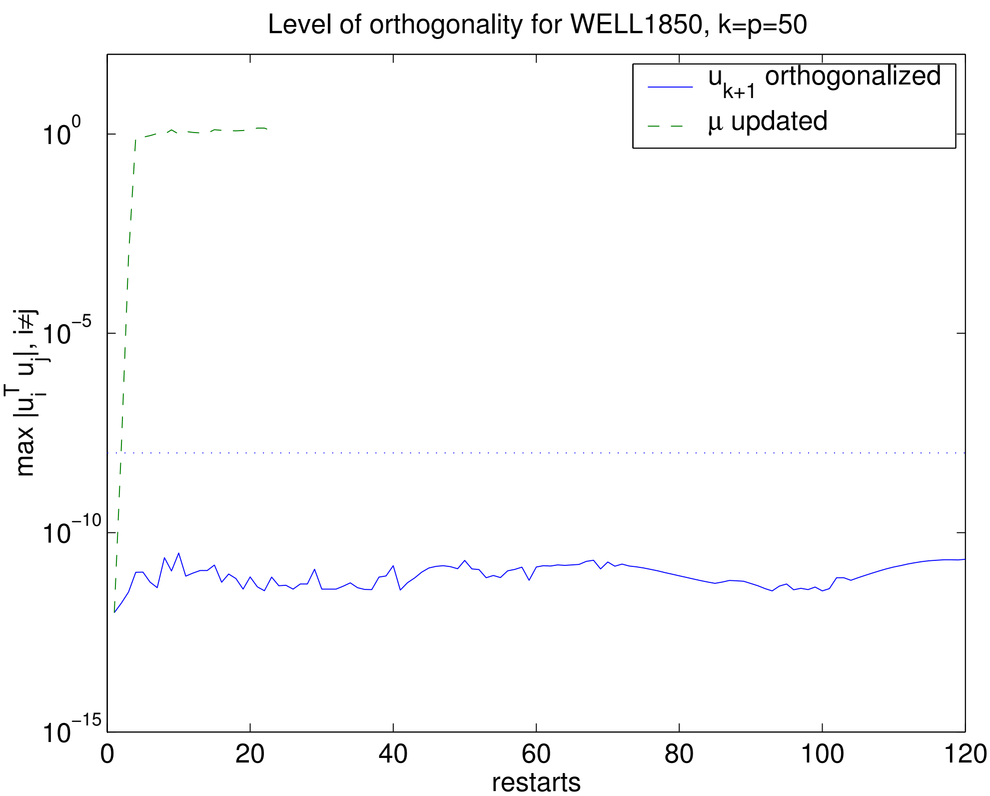
$$\begin{aligned} |I - (V_{k+p}^+)^T V_{k+p}^+| &= |I - Q_l^T V_{k+p}^T V_{k+p} Q_l| \\ &= |Q_l^T E Q_l| \\ &\leq \|E\|_2 \\ &\leq (k+p)\sqrt{\frac{\mathbf{u}}{2k+1}} \end{aligned}$$

So we may experience some further loss of orthogonality due to the implicit restarting.

Is it stable? (cont.)

In practice we have found that it is sufficient to orthogonalize u_{k+1}^+ against $u_1^+, u_2^+, \ldots, u_k^+$ and v_k^+ against $v_1^+, v_2^+, \ldots, v_{k-1}^+$ before extending the bidiagonalization.

This set of precautions manages to preserve semiorthogonality, even after many restarts, as illustrated below:



In our experience, the singular values computed with the restarted algorithm were just as accurate as those computed without restarts.

Is it worth the trouble?

How much is gained, compared to full reorthogonalization, by applying partial reorthogonalization to LBD and its implicitly restarted variant?

For computing $k = 100$ singular values we get:

Program	LBDSVD		
	n	# of DOTS	efficiency
WELL1850	500	80798	68%
ILLC1850	403	44908	72%
TOLS4000	315	20020	80%
MHD4800A	203	42213	0%
AF23560	299	37369	57%
FIDAPM11	301	29791	67%

Program	LBDIR($p = 100$)		
	restarts	# of DOTS	efficiency
WELL1850	3	32550	75%
ILLC1850	2	25733	74%
TOLS4000	2	21580	78%
MHD4800A	0	41811	0%
AF23560	1	27263	61%
FIDAPM11	1	21628	69%

Shift strategies

The selection of the shift $\mu_1, \mu_2, \ldots, \mu_p$ is crucial to the efficiency of a restarted algorithm.

Intuition: The shifts should be chosen such that the polynomial filter

$$u_1^+ = \prod_{i=1}^{p} (AA^T - \mu_i^2 I) u_1 \ .$$

removes components in u_1 corresponding to the unwanted part of the spectrum and retains components in the desired part. Examples:

- *Exact shifts*: Use $\theta_{k+1}, \theta_{k+2}, \ldots, \theta_{k+p}$.
- *Chebychev shifts*: Use zeros of T_p scaled to an interval containing the unwanted part of the spectrum.
- *Leja point shifts*: Use Leja points for interval containing the unwanted part of the spectrum.

Lehoucq, Sorenson & Yang (ARPACK) use exact shifts, while Calvetti, Reichel & Sorensen recommend shift based on Leja points.

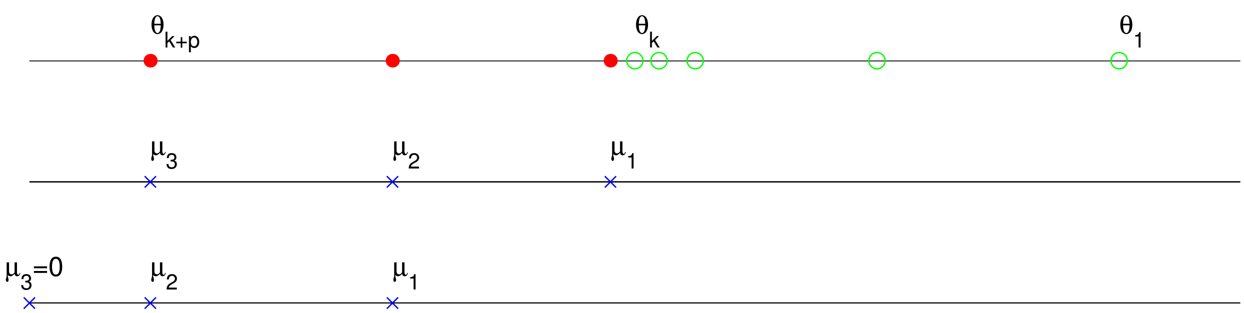
We find that exact shifts perform slightly better, provided close singular values are accounted for. If not, all strategies are prone to *very poor performance* or even *stagnation*!

Clusters of singular values

The normal shift strategies fail when σ_k and σ_{k+1} are close. When θ_{k+1} is used as an exact shift, the component along the kth singular vector is greatly damped in

$$u_1^+ = \prod_{i=k+1}^{k+p} (AA^T - \theta_i I) u_1$$

This can cause θ_k to converge very slowly to σ_k (or not at all).



A simple but very effective solution is to require that the relative gap

$$\text{relgap}_{ki} \equiv \frac{(\theta_k - e_k) - \mu_i}{\theta_k}$$

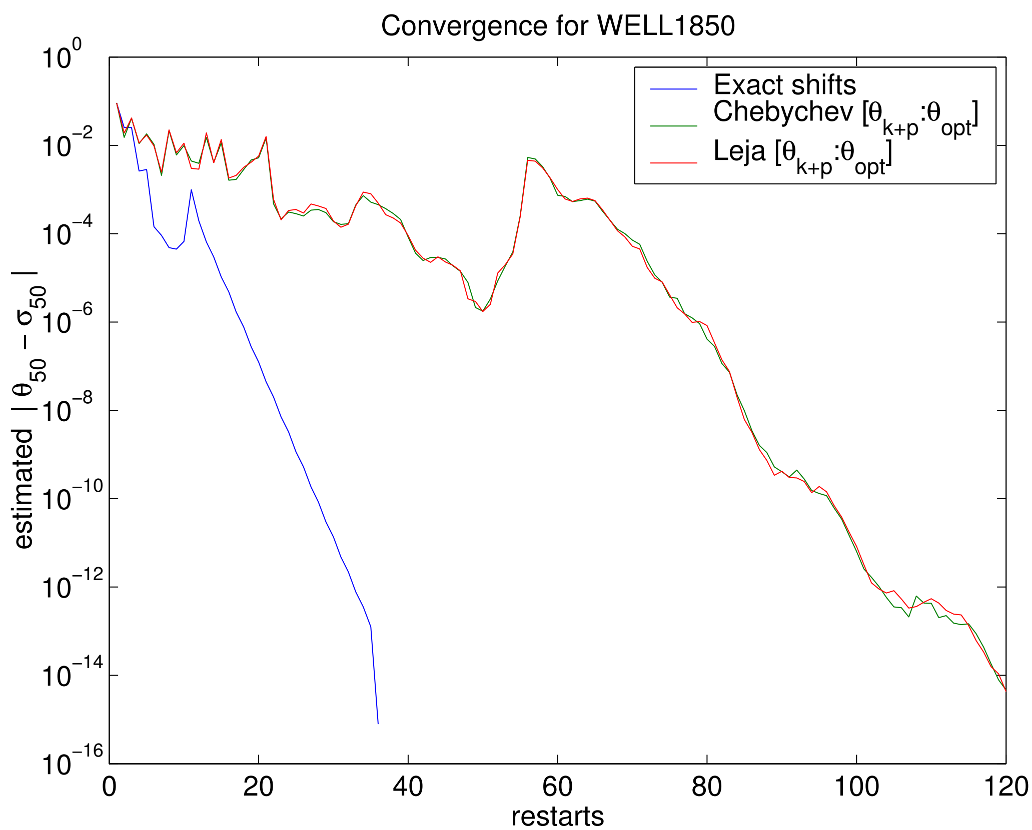
between the smallest Ritz value θ_k and all shifts $\mu_i,\ i = 1, \ldots, p$ be larger than some prescribed tolerance. Experimentally we have found that requiring $\text{relgap}_{ki} > 10^{-3}$ seems to work well. Bad shifts can, e.g., be replaced by zero shifts.

Example of poor convergence for close σ's

For the matrix WELL1850, σ_{50} has several close neighbors:

σ_{48}	1.409645143251147
σ_{49}	1.409203443807433
σ_{50}	1.408180353484225
σ_{51}	1.408059653705621
σ_{52}	1.408003552724529
σ_{53}	1.407571434622690

With $k = p = 50$ and traditional shifts the convergence of θ_{50} is terrible:



Example continued

With a minimal relative gap tolerance of 10^{-3}, the fast convergence is recovered:

PROPACK: **Software package for large-scale SVD**

Main components:

`DLANBPRO`	:	Lanczos bidiagonalization with partial reorth.
`DLANSVD`	:	Singular value decomposition
`DLANSVD_IRL`	:	`DLANSVD` with implicit restarts

Important implementation details:

— respecting coupling between μ and ν

— extended local reorthogonalization

— iterated Gram-Schmidt reorth. (DGKS, BLAS-2)

— recovery from near zero α_i or β_i

— proper estimation of $\|A\|$

— Currently uses DBDSQR for partial and divide-and-conquer for full bidiagonal SVD (B. Grosser's Holy Grail code?).

— IRL: updating Lanczos vectors using BLAS-3

URL: `http://soi.stanford.edu/~rmunk/PROPACK`

Performance comparison

The routines LBDSVD and LBDIR were compared with LANSO and ARPACK. The table shows CPU-time in seconds used to compute the 100 largest singular values.

Program	LBDSVD	LBDIR	LANSO		ARPACK	
Matrix	A		A^TA	C	A^TA	C
WELL1850	2.79	3.16	1.21	2.73	7.22	48.01
ILLC1850	1.91	2.36	1.55	3.17	5.31	36.75
TOLS4000	2.42	5.21	4.01	8.07	25.86	90.96
MHD4800A	6.16	6.04	7.33	37.95	15.14	162.48
AF23560	35.39	34.93	46.71	199.30	156.69	644.11
FIDAPM11	32.98	33.36	38.16	151.78	133.96	600.72

- LBDSVD and LBDIR significantly faster than other backwards stable methods.
- LANSO consistently faster than ARPACK on the same problem.
- LANSO(A^TA) (not surprisingly) is the fastest for rectangular matrices where $m \gg n$ (WELL1850 and ILL1850).

Performance computing fewer singular values

First 10 singular values:

Program	LBDSVD	LBDIR	LANSO		ARPACK	
Matrix	A		A^TA	C	A^TA	C
WELL1850	0.26	0.27	0.11	0.74	0.18	1.27
ILLC1850	0.18	0.20	0.20	0.76	0.15	1.02
TOLS4000	0.71	0.77	1.00	5.90	2.41	9.61
MHD4800A	0.40	0.45	0.42	1.26	0.91	4.84
AF23560	4.12	4.62	4.80	15.08	9.62	30.16
FIDAPM11	5.98	6.73	7.86	23.11	24.12	72.08

First 50 singular values:

Program	LBDSVD	LBDIR	LANSO		ARPACK	
Matrix	A		A^TA	C	A^TA	C
WELL1850	3.36	2.84	1.02	2.69	3.27	28.64
ILLC1850	1.49	1.49	0.73	3.12	2.45	20.97
TOLS4000	1.45	1.81	6.37	7.41	10.24	37.64
MHD4800A	2.01	1.99	2.39	8.23	6.42	38.86
AF23560	16.97	17.70	18.56	70.44	55.86	212.16
FIDAPM11	16.97	18.05	24.90	66.80	61.65	207.84

Conclusion

- Implicitly restarted bidiagonalization based on Golub-Kahan SVD steps has been implemented, and appears to be fast and accurate.
- It seems that partial reorthogonalization can be succesfully combined with implicit restarting techniques without loss of stability, although a rigorous proof was not given.
- A simple adaptive shifting strategy significantly improves performance if the user chooses the cut-off point in a cluster.
- The resulting algorithm is significantly faster than other Lanczos based codes if high accuracy is required.