Polynomial Filtering for Large, Sparse SVD Computations

Jared Aurentz, Anthony P. Austin, and Vasileios Kalantzis

Summary

Many applications require the computation of a few singular values and vectors of a large, sparse matrix. We present a polynomial filtering technique for accelerating such computations. Our method is competitive with existing algorithms and is particularly effective when many singular values are required.

Background

Lanczos bidiagonalization is an efficient and scalable method for computing a few leading singular values of a large matrix.

- Using the two-step recurrence

$$\begin{aligned} Av_j &= \alpha_j u_j + \beta_{j-1} u_{j-1} \\ A^* u_j &= \alpha_j v_j + \beta_j v_{j+1} \end{aligned}$$

  compute matrices U_k and V_k with orthonormal columns $u_1, \ldots, u_k$ and $v_1, \ldots, v_k$ (Lanczos vectors) such that

$$B_k = U_k^* A V_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \alpha_{k-1} & \beta_{k-1} \\ & & & & \alpha_k \end{bmatrix}.$$

- The singular values of B_k approximate those of A.
- As k increases, the largest singular values converge first.
- The better-separated these values are from the rest, the faster the convergence.
- The method engages A only through matrix-vector products.

The Problem

What if we need many singular values, possibly not the leading ones?

- One approach: Keep taking Lanczos steps until all desired values converge.
  - The number of steps needed may be quite large.
  - Each step produces two new Lanczos vectors, increasing memory usage and orthogonalization costs.
- Better idea: Apply a spectral transformation.
  - Move the desired values to the high end of the spectrum.
  - The classic choice is the shift-and-invert transformation—quite effective but expensive for large matrices (must solve linear systems).

Our Method

We propose using a polynomial filter to accelerate the computation.

- If p is a polynomial and

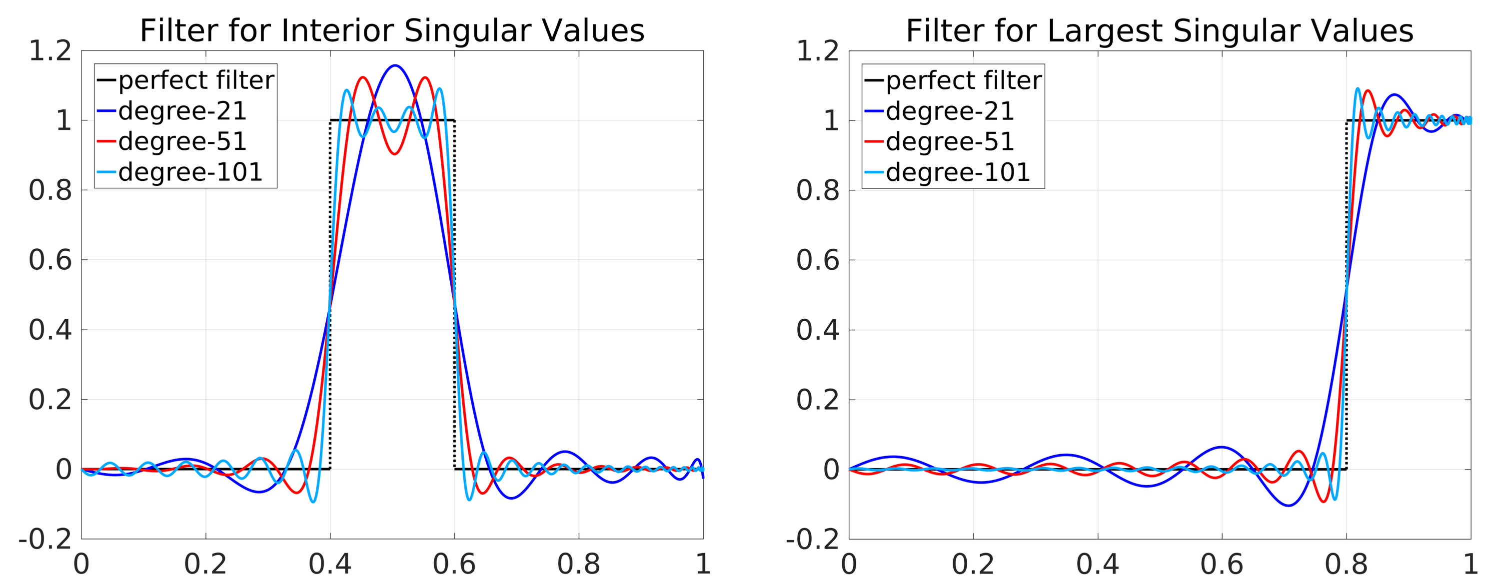
$$A = U\Sigma V^*$$

  is the SVD of A, then

$$Ap(A^*A) = U\Sigma p(\Sigma^2)V^* = Uq(\Sigma)V^*, \qquad q(x) = xp(x^2).$$

- $Ap(A^*A)$ has the same singular vectors as A, but its singular values have been transformed by the filter q.

By selecting p so that q is large on the singular values of interest and small on the rest, the Lanczos algorithm applied to $Ap(A^*A)$ will rapidly pick out singular vectors corresponding to the desired values. We choose p so that q is a Chebyshev least-squares approximation to the characteristic function of the interval containing the singular values of interest. Note that q always has odd symmetry.



Benefits:

- Better isolation of the wanted singular values means fewer Lanczos steps are needed for convergence.
- Fewer steps means fewer Lanczos vectors, saving memory and effort spent on orthogonalization.
- Since the filter is a polynomial, the method engages A only via matrix-vector products—superior scaling to shift-and-invert.
- Interior singular values can be easily computed by choosing a filter that de-emphasizes the extreme ones.
- If many interior singular values are required, multiple search intervals can be processed in parallel.
- The method is easier to implement than restarted Lanczos.

Numerical Results

We implemented our method on top of the Lanczos bidiagonalization routines available in the SLEPc library. Some practical details:

- We scale the matrix so that its singular values lie in [0, 1] using an initial estimate of the leading singular value, which we get from a few (≈10) steps of unfiltered Lanczos.
- We employ full reorthogonalization to ensure orthogonality of the computed singular vectors.

Example: We compute the leading 100 singular values of the "dawson5" matrix from the UF Sparse Matrix Collection.

- Problem size: 51,537 × 51,537 with 4,653,901 nonzeros.
- We use our method with degree-11 and degree-17 filters on [0.87, 1].
- We compare with unfiltered Lanczos, the thick-restart Lanczos solver in SLEPc, and `svds` in MATLAB.

The results are summarized in the following pair of tables. The filtered Lanczos methods use more matrix-vector products but spend far less time on orthogonalization, leading to significant computational savings.

Method	Time (s)	Lanczos Steps	Mat-vecs
Unfiltered Lanczos	83	831	1,862
Filtered Lanczos (degree-11)	18	256	5,961
Filtered Lanczos (degree-17)	20	228	8,081
Thick-restart Lanczos	37	228 (6 restarts)	1,872
MATLAB `svds`	30	—	—

Method	% of Time		
	Mat-vecs	Orthogonalization	Other
Unfiltered Lanczos	4	93	3
Filtered Lanczos (degree-11)	54	41	5
Filtered Lanczos (degree-17)	67	29	4
Thick-restart Lanczos	7	84	9

References

- Golub, G. H. and Van Loan, C. Matrix Computations, 4th ed., Johns Hopkins University Press, Baltimore, 2013.
- Hernandez, V., Roman, J. E., and Vidal, V. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. ACM Trans. Math. Soft. 31 (2005), pp. 351-362.