CS 6210: Matrix Computations

Lanczos and Arnoldi

David Bindel

2025-11-17

Krylov subspaces

The *Krylov subspace* of dimension k generated by $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$ is

$$\mathcal{K}_k(A,b) = \text{span}\{b, Ab, \ldots, A^{k-1}b\} = \{p(A)b : p \in \mathcal{P}_{k-1}\}.$$

Krylov subspaces are a natural choice for subspace-based methods for approximate linear solves, for two reasons:

- If all you are allowed to do with A is compute matrix-vector products, and the only vector at hand is b, what else would you do?
- The Krylov subspaces have excellent approximation properties.

Krylov subspaces have several properties that are worthy of comment. Because the vectors $A^j b$ are proportional to the vectors obtained in power iteration, one might reasonably (and correctly) assume that the space quickly contains good approximations to the eigenvectors associated with the largest magnitude eigenvalues. Krylov subspaces are also *shift-invariant*, i.e. for any σ

$$\mathcal{K}_k(A - \sigma I, b) = \mathcal{K}_k(A, b).$$

By choosing different shifts, we can see that the Krylov subspaces tend to quickly contain not only good approximations to the eigenvector associated with the largest magnitude eigenvalue, but to all "extremal" eigenvalues.

Most arguments about the approximation properties of Krylov subspaces derive from the characterization of the space as all vectors $p(A)b$ where $p \in \mathcal{P}_{k-1}$ and from the spectral mapping theorem, which says that if $A = V\Lambda V^{-1}$ then $p(A) = Vp(\Lambda)V^{-1}$. Hence, the distance between an arbitrary vector (say d) and the Krylov subspace is

$$\min_{p\in\mathcal{P}_{k-1}} \left\| V[p(\Lambda)V^{-1}b - V^{-1}d] \right\|.$$

As a specific example, suppose that we want to choose $\hat{x}$ in a Krylov subspace in order to minimize the residual $A\hat{x} - b$. Writing $\hat{x} = p(A)b$, we have that we want to minimize

$$\|[Ap(A) - I]b\| = \|q(A)b\|$$

where $q(z)$ is a polynomial of degree at most k such that $q(1) = 1$. The best possible residual in this case is bounded by

$$\|q(A)b\| \leq \kappa(V)\|q(\Lambda)\|\|b\|,$$

and so the relative residual can be bounded in terms of the condition number of V and the minimum value that can bound q on the spectrum of A subject to the constraint that $q(0) = 1$.

Chebyshev polynomials

Suppose now that A is symmetric positive definite, and we seek to minimize $\|q(A)b\| \leq \|q(\Lambda)\|\|b\|$. Controlling $q(z)$ on all the eigenvalues is a pain, but it turns out to be simple to instead bound $q(z)$ over some interval $[\alpha_1, \alpha_n]$ The polynomial we want is the *scaled and shifted Chebyshev polynomial*

$$q_m(z) = \frac{T_m\left((z - \bar{\alpha})/\rho\right)}{T_m\left(-\bar{\alpha}/\rho\right)}$$

where $\bar{\alpha} = (\alpha_n + \alpha_1)/2$ and $\rho = (\alpha_n - \alpha_1)/2$.

The Chebyshev polynomials T_m are defined by the recurrence

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{m+1}(x) &= 2xT_m(x) - T_{m-1}(x), \quad m \geq 1. \end{aligned}$$

The Chebyshev polynomials have a number of remarkable properties, but perhaps the most relevant in this setting is that

$$T_m(x) = \begin{cases} \cos(m\cos^{-1}(x)), & |x| \leq 1, \\ \cosh(m\cosh^{-1}(x)), & |x| \geq 1 \end{cases}.$$

Thus, $T_m(x)$ oscillates between ± 1 on the interval $[-1, 1]$, and then grows very quickly outside that interval. In particular,

$$T_m(1 + \epsilon) \geq \frac{1}{2}(1 + m\sqrt{2\epsilon}).$$

Thus, we have that on $[\alpha, \alpha_n]$, $|q_m| \leq \frac{2}{1+m\sqrt{2\epsilon}}$ where

$$\epsilon = \bar{\alpha}/\rho - 1 = \frac{2\alpha_1}{\alpha_n - \alpha_1} = 2\left(\kappa(A) - 1\right)^{-1},$$

and hence

$$|q_m(z)| \leq \frac{2}{1 + 2m/\sqrt{\kappa(A) - 1}}$$
$$= 2\left(1 - \frac{2m}{\sqrt{\kappa(A) - 1}}\right) + O\left(\frac{m^2}{\kappa(A-1)}\right).$$

Hence, we expect to reduce the optimal residual in this case by at least about $2/\sqrt{\kappa(A) - 1}$ at each step.

Chebyshev: Uses and Limitations

We previously sketched out an approach for analyzing the convergence of methods based on Krylov subspaces:

1. Characterize the Krylov subspace of interest in terms of polynomials, i.e. $\mathcal{K}_k(A, b) = \{p(A)b : p \in \mathcal{P}_{k-1}\}$.
2. For $\hat{x} = p(A)b$, write an associated error (or residual) in terms of a related polynomial in A.
3. Phrase the problem of minimizing the error, residual, etc. in terms of minimizing a polynomial $q(z)$ on the spectrum of A (call this $\Lambda(A)$). The polynomial q must generally satisfy some side constraints that prevent the zero polynomial from being a valid solution.
4. Let $\Lambda(A) \subset \Omega$, and write
$$\max_{\lambda \in \Lambda(A)} |q(\lambda)| \leq \max_{z \in \Omega} |q(z)|.$$
The set Ω should be simpler to work with than the set of eigenvalues. The simplest case is when A is symmetric positive definite and $\Omega = [\lambda_1, \lambda_n]$.
5. The optimization problem can usually be phrased in terms of special polynomial families. The simplest case, when Ω is just an interval, usually leads to an analysis via Chebyshev polynomials.

The analysis sketched above is the basis for the convergence analysis of the Chebyshev semi-iteration, the conjugate gradient method, and (with various twists) several other Krylov subspace methods.

The advantage of this type of analysis is that it leads to convergence bounds in terms of some relatively simple property of the matrix, such as the condition number. The disadvantage is that the approximation of the spectral set $\Lambda(A)$ by a bounding region Ω can lead to rather pessimistic bounds. In practice, the extent to which we are able to find good solutions in a Krylov subspace often depends on the "clumpiness" of the eigenvalues. Unfortunately, this "clumpiness" is rather difficult to reason about a priori! Thus, the right way to evaluate the

convergence of Krylov methods in practice is usually to try them out, plot the convergence curves, and see what happens.

Arnoldi

Krylov subspaces are good spaces for approximation schemes. But the power basis (i.e. the basis $A^j b$ for $j = 0, \ldots, k-1$) is not good for numerical work. The vectors in the power basis tend to converge toward the dominant eigenvector, and so the power basis quickly becomes ill-conditioned. We would much rather have orthonormal bases for the same spaces. This is where the Arnoldi iteration and its kin come in.

Each step of the Arnoldi iteration consists of two pieces:

- Compute Aq_k to get a vector in the space of dimension $k+1$
- Orthogonalize Aq_k against $q_1, \ldots, q_k$ using Gram-Schmidt. Scale the remainder to unit length.

If we keep track of the coefficients in the Gram-Schmidt process in a matrix H, we have

$$h_{k+1,k} q_{k+1} = Aq_k - \sum_{j=1}^{k} q_j h_{jk}$$

where $h_{jk} = q_j^T A q_k$ and $h_{k+1,k}$ is the normalization constant. Rearranging slightly, we have

$$Aq_k = \sum_{j=1}^{k+1} q_j h_{jk}.$$

Defining $Q_k = \begin{bmatrix} q_1 & q_2 & \ldots & q_k \end{bmatrix}$, we have the *Arnoldi decomposition*

$$AQ_k = Q_{k+1} \bar{H}_k, \quad \bar{H}_k = \begin{bmatrix} h_{11} & h_{12} & \ldots & h_{1k} \\ h_{21} & h_{22} & \ldots & h_{2k} \\ & h_{32} & \ldots & h_{3k} \\ & & \ddots & \vdots \\ & & & h_{k+1,k} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}.$$

The Arnoldi *decomposition* is simply the leading k columns of an upper Hessenberg reduction of the original matrix A. The Arnoldi *algorithm* is the interlaced multiply-and-orthogonalize process used to obtain the decomposition. Unfortunately, the modified Gram-Schmidt algorithm — though more stable than classical Gram-Schmidt! — is nonetheless unstable in general. The sore point occurs when we start with a matrix that lies too near the span of the vectors we orthogonalize against; in this case, by the time we finish orthogonalizing against previous vectors, we have cancelled away so much of the original vector that what is left may be substantially

contaminated by roundoff. For this reason, the "twice is enough" reorthogonalization process of Kahan and Parlett is useful: this says that if the remainder after orthogonalization is too small compared to what we started with, then we should perhaps refine our computation by orthogonalizing the remainder again.

Lanczos

Now suppose that A is a symmetric matrix. In this case, the Arnoldi decomposition takes a special form: the upper Hessenberg matrix is now a symmetric upper Hessenber matrix (aka a tridiagonal), and we dub the resulting decomposition the *Lanczos* decomposition:

$$AQ_k = Q_{k+1}\bar{T}_k, \quad T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & \\ & \beta_2 & \alpha_3 & \beta_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} \\ & & & & \beta_{k-1} & \alpha_k \\ & & & & & \beta_k \end{bmatrix}$$

The Lanczos algorithm is the specialization of the Arnoldi algorithm to the symmetric case. In exact arithmetic, the tridiagonal form of the coefficient matrix allows us to do only a constant amount of orthogonalization work at each step.

Sadly, the Lanczos algorithm in floating point behaves rather differently from the algorithm in exact arithmetic. In particular, the iteration tends to "restart" periodically as the space starts to get very good approximations of eigenvectors. One can deal with this via full reorthogonalization, as with the Arnoldi iteration; but then the method loses the luster of low cost, as we have to orthogonalize against several vetors periodically. An alternate *selective orthogonalization* strategy proposed by Parlett and Scott lets us orthogonalize only against a few previous vectors, which are associated with converged eigenvector approximations (Ritz vectors). But, as we shall see, such orthogonalization is mostly useful when we want to solve eigenvalue problems. For linear systems, it tends not to be necessary.