Notes for 2016-11-21

1 References

There is a lot of ground to cover when it comes to Krylov subspace methods, and we scarcely have time to do justice to the two most popular Krylov subspace methods (CG for the SPD case and GMRES elsewhere). Apart from the material in Golub and Van Loan and other standard texts, I highly recommend two books for a survey of other methods and some practical details:

1. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* by Barrett *et al*. This is freely available, and includes what you need to know to get started with various methods.

2. *Iterative methods for sparse linear systems* by Y. Saad. This is now in a second edition (available from SIAM), but you can also get the first edition at Saad's web page.

3. *Iterative methods for solving linear systems* by Anne Greenbaum (published by SIAM) is a fairly up-to-date treatment of the major iterative solvers for linear systems, including the whole family of Krylov subspace solvers as well as classical stationary iterations and multigrid methods.

4. *Iterative methods for linear and nonlinear equations* by C. T. Kelley is another SIAM book — are you seeing a theme? It covers CG and GMRES, though not the other Krylov iterations; however, it also covers nonlinear iterations. It is short and tutorial in nature.

2 GMRES

The *generalized minimal residual* (GMRES) method of solving linear systems works with general systems of linear equations. Next to CG, it is probably the second-most popular of the Krylov subspace iterations.

The GMRES method is so named because it chooses the solution from a linear subspace that minimizes the (Euclidean) norm of the residual over

successive Krylov subspaces. In terms of the Arnoldi decompositions

$$AQ_k = Q_{k+1}\bar{H}_k,$$

we have that $x_k = Q_k y_k$ where

$$y_k = \text{argmin}_y \|\bar{H}_k y - \|b\|e_1\|^2.$$

One can solve the Hessenberg least squares problem in $O(k^2)$ time, but this is generally a non-issue. The true cost of GMRES is in saving the basis (which can use memory very quickly) and in keeping the basis orthogonal.

Unlike the CG method, alas, the GMRES method does not boil down to a short recurrence through a sequence of clever tricks. Consequently, we generally cannot afford to run the iteration for many steps before *restart*. We usually denote the iteration with periodic restarting every m steps as GMRES(m). That is, at each step we

1. Start with an initial guess $\hat{x}$ from previous steps.
2. Form the residual $r = b - A\hat{x}$.
3. Run m steps of GMRES to approximately solve $Az = r$.
4. Update $\hat{x} := \hat{x} + z$.

The GMRES iteration is generally used with a preconditioner. The common default is to use preconditioning on the left, i.e. solve

$$M^{-1}Ax = M^{-1}b;$$

in this setting, GMRES minimizes not the original residual, but the *preconditioned* residual. To the extent that the preconditioner reduces the condition number of the problem overall, the norm of the preconditioned residual tends to be a better indicator for forward error than the norm of the unpreconditioned residual. Of course, one can also perform preconditioning on the right (i.e. changing the unknown), or perform two-sided preconditioning.

The standard GMRES iteration (along with CG and almost every other Krylov subspace iteration) assumes a single, fixed preconditioner. But what if we want to try several preconditioners at once, or perhaps to use Gauss-Southwell or a chaotic relaxation method for preconditioning? Or perhaps we want to use a variable number of steps of some other iteration to precondition

something like GMRES? For this purpose, it is useful to consider the *flexible GMRES* variant (FGMRES). Though it no longer technically is restricted to a Krylov subspace generated by a fixed matrix, the FGMRES iteration looks very similar to the standard GMRES iteration; we build an Arnoldi-like decomposition with the form

$$AZ_m = V_{m+1}\bar{H}_m$$

and then compute updates as a linear combination of the columns of Z_m by solving a least squares problem with $\bar{H}_m$. But here, each column of Z_m looks like $z_j = M_j^{-1} v_j$ where each M_j may be different.

3 Bi-Lanczos

So far, our focus has been on Krylov subspace methods that we can explain via the Lanczos or Arnoldi decompositions. The Lanczos-based CG has many attractive properties, but it only works with symmetric and positive definite matrices. One can apply CG to a system of normal equations — the so-called CGNE method — but this comes at the cost of squaring the condition number. There are also methods such as the LSQR iteration that implicitly work with the normal equations, but use an incrementally-computed version of the Golub-Kahan bi-diagonalization. The Arnoldi-based GMRES iteration works for more general classes of problems, and indeed it is the method of choice; but it comes at a stiff penalty in terms of orthogonalization costs.

Are there alternative methods that use short recurrences (like CG) but are appropriate for nonsymmetric matrices? There are several, though all have some drawbacks; the QMR and BiCG iterations may be the most popular. The key to the behavior of these methods comes from their use of a different decomposition, the *bi-orthogonal Lanczos factorization*:

$$\begin{aligned} AQ_j &= Q_j T_j + \beta_{j+1} q_{j+1} e_j^* \\ A^* P_j &= P_j T_j^* + \bar{\gamma}_{j+1} p_{j+1} e_j^* \\ P_j^* Q_j &= I. \end{aligned}$$

Here, the bases Q_j and P_j span Krylov subspaces generated by A and A^*, respectively (which means these algorithms require not only a function to apply A to a vector, but also a function to apply A^*). The bases are not

orthonormal, and indeed may become rather ill-conditioned. They *do* have a mutual orthogonality relationship, though, namely $P_j^* Q_j = I$.

Details of the bi-orthogonal Lanczos factorization and related iterative algorithms can be found in the references. For the present, we satisfy ourseves with a few observations:

- The GMRES iteration shows monotonic reduction in the preconditioned residual, even with restarting. CG shows monotonic reduction in the error or residual when measured in appropriate norms. The methods based on bi-orthogonal Lanczos, however, can show rather erratic convergence; errors decay in general, but they may exhibit intermediate local increases. BiCG is generally more erratic than QMR.

- Even in exact arithmetic, the subspace bases formed by bi-Lanczos may become rather ill-conditioned.

- The bi-orthogonal iterations sometimes show *breakdown* behavior where the local approximation problem becomes singular. This can be overcome using *lookahead* techniques, though it complicates the algorithm.

The relative simplicity of GMRES — both in theory and in implementation — perhaps explains its relative popularity. Nonetheless, these other methods are worth knowing about.

4 Extrapolation and mixing

When we discussed CG, we also briefly discussed *nonlinear* CG methods (e.g. Fletcher-Reeves). One can similarly extend Krylov subspace ideas to accelerate nonlinear equation solving methods; that is, given a fixed point iteration

$$x^{(k+1)} = G(x^{(k)}),$$

we can accelerate the computation of the fixed point by taking an appropriate linear combination of the iterates $x^{(k)}$. This is a powerful idea; indeed, it is so powerful that it can be used to compute repulsive fixed points where the usual iteration would diverge! The techniques usually go under the heading of *extrapolation methods* (including Reduced Rank Extrapolation or RRE, Minimal Polynomial Extrapolation or MPE, and Vector Padé Extrapolation); and *acceleration* or *mixing* techniques, the most popular of which is

the *Anderson acceleration* method. Applied to the iterates of a stationary linear system solver, these techniques are all formally equivalent to Krylov subspace solvers. In particular, RRE is equivalent to GMRES (in exact arithmetic).

The idea behind extrapolation methods is to exploit systematic patterns in the convergence of fixed point iteration. For example, suppose the error iteration gave us (approximately)

$$e^{(k)} = \sum_{j=1}^{m} v^{(j)} \alpha_j^k$$

where the vectors $v^{(j)}$ and the exponents α_j were unknown. The hope is that we can learn the parameters of the error iteration by fitting a model to the update sequence:

$$u^{(k)} = x^{(k+1)} - x^{(k)} = e^{(k+1)} - e^{(k)} = \sum_{j=1}^{m} (\alpha_j - 1) v^{(j)} \alpha_j^k.$$

If $p(z) = c_0 + c_1 z + \ldots + c_m z^m$ is a polynomial such that $p(\alpha_j) = 0$ for each α_j, then we should satisfy

$$\sum_{j=1}^{m} c_j u^{(k+j)} = 0.$$

If we look at enough update steps, we can determine both the coefficient vectors and the exponents.

With an appropriate model, extrapolation methods can produce rather astonishing results. Of course, extrapolation methods are subject to issues of overfitting, and (particularly when the convergence is irregular) may produce results that are wildly incorrect.

4.1 Communication-Avoiding (CA) Krylov

In highly parallel computing systems, the cost of computing with Krylov subspaces may be dominated not by the matrix-vector products, but by the cost of computing dot products for the purpose of orthogonalization. Repeatedly applying matrix-vector products may involves rather local communication patterns, but dot products involve a global communication. Of course, we

could (in principle) form a power basis for the Krylov subspace; but this basis is typically too ill-conditioned for serious work. So what is one to do?

The *communication-avoiding* Krylov methods use the power of polynomials to thread between the Scylla of synchronization costs and the Charybdis of catastrophic ill-conditioning. In general, we write Krylov subspace bases as

$$\mathcal{K}_k(A, b) = \text{span}\{p_j(A)b\}_{j=0}^{(k-1)}.$$

where $p_j(z)$ is a degree j polynomial. In the case of the power basis, $p_j(z) = z^j$; and in the case of the Lanczos or Arnoldi bases, $p_j(z)$ is chosen fully adaptively. The communication avoiding approach is to choose $p_j(z)$ in advance, but using information about the spectra to ensure that the vectors $p_j(A)b$ are not too nearly co-linear.

As with some of the other topics in this section, the big idea behind communication-avoiding Krylov methods is simple, but there are too many details to give a full treatment in the time we have allocated. For those interested in such details, I recommend the 2010 Ph.D. thesis of Mark Hoemmen.