2022-03-16

1 Linear Solves and Quadratic Minimization

We have already briefly described an argument that Jacobi iteration converges for strictly row diagonally dominant matrices. We now discuss an argument that Gauss-Seidel converges (or at least part of such an argument). In the process, we will see a useful way of reformulating the solution of symmetric positive definite linear systems that will prepare us for our upcoming discussion of conjugate gradient methods.

Let A be a symmetric positive definite matrix, and consider the "energy" function

$$\phi(x) = \frac{1}{2} x^T A x - x^T b.$$

The stationary point for this function is the point at which the derivative in any direction is zero. That is, for any direction vector u,

$$\begin{aligned}
0 &= \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \phi(x + \epsilon u) \\
&= \frac{1}{2} u^T A x + \frac{1}{2} x^T A u - u^T b \\
&= u^T (Ax - b)
\end{aligned}$$

Except in pathological instances, a directional derivative can be written as the dot product of a direction vector and a gradient; in this case, we have

$$\nabla \phi = Ax - b.$$

Hence, minimizing ϕ is equivalent to solving $Ax = b$ [1].

Now that we have a characterization of the solution of $Ax = b$ in terms of an optimization problem, what can we do with it? One simple approach is to think of a sweep through all the unknowns, adjusting each variable in term to minimize the energy; that is, we compute a correction Δx_j to node j such that

$$\Delta x_j = \operatorname{argmin}_z \phi(x + z e_j)$$

[1] If you are unconvinced that this is a minimum, work through the algebra to show that $\phi(A^{-1}b + w) = \frac{1}{2} w^T A w$ for any w.

Note that
$$\frac{d}{dz}\phi(x+ze_j) = e_j^T(A(x+ze_j)-b),$$
and the update $x_j := x_j + \Delta x_j$ is equivalent to choosing a new x_j to set this derivative equal to zero. But this is exactly what the Gauss-Seidel update does! Hence, we can see Gauss-Seidel in two different ways: as a stationary method for solving a linear system, or as an optimization method that constantly makes progress toward a solution that minimizes the energy [2]. The latter perspective can be turned (with a little work) into a convergence proof for Gauss-Seidel on positive-definite linear systems.

2 Extrapolation: A Hint of Things to Come

Stationary iterations are simple. Methods like Jacobi or Gauss-Seidel are easy to program, and it's (relatively) easy to analyze their convergence. But these methods are also often slow. We'll talk next time about more powerful *Krylov subspace* methods that use stationary iterations as a building block.

There are many ways to motivate Krylov subspace methods. We'll pick one motivating idea that extends beyond the land of linear solvers and into other applications as well. The key to this idea is the observation that the error in our iteration follows a simple pattern:
$$x^{(k)} - x = e^{(k)} = R^k e^{(0)}, \quad R = M^{-1}N.$$
Suppose R is diagonalizable, i.e. $R = V\Lambda V^{-1}$, and let $V^{-1}e^{(0)} = c$. Then we have
$$e^{(k)} = V\Lambda^k V^{-1} e^{(0)} = V\Lambda^k c = \sum_{j=1}^n v_j \lambda_j^k c_j.$$
Assuming there is a unique dominant eigenvalue, the behavior of the error is dominated by that eigenvalue for large k, i.e.
$$e^{(k+1)} \approx \lambda_1 e^{(k)}.$$
Note that this means
$$x^{(k+1)} - x^{(k)} = e^{(k+1)} - e^{(k)} \approx (\lambda_1 - 1)e^{(k)}.$$

[2]Later in the class, we'll see this as coordinate-descent with exact line search.

If we have an estimate for λ_1, we can write

$$x = x^{(k)} - e^{(k)} \approx x^{(k)} - \frac{x^{(k+1)} - x^{(k)}}{\lambda_1 - 1}.$$

That is, we might hope to get a better estimate of x than is provided by $x^{(k)}$ or $x^{(k+1)}$ individually by taking an appropriate linear combination of $x^{(k)}$ and $x^{(k+1)}$.

How might we get an estimate for λ_1? In some cases, we might be able to guess a good estimate from other context. Otherwise, though, we might estimate λ_1 via a Rayleigh quotient. Let $u^{(k+1)} = x^{(k+1)} - x^{(k)}$; we know from our iteration equation that $u^{(k+1)} = Ru^{(k)}$, so we might try to use Rayleigh quotients to estimate λ_1:

$$\rho_R(u^{(k)}) = \frac{u^{(k)} \cdot Ru^{(k)}}{\|u^{(k)}\|^2} = \frac{u^{(k)} \cdot u^{(k+1)}}{\|u^{(k)}\|^2}.$$

Plugging this into our estimate for the error correction gives us a transformed sequence

$$\check{x}^{(k)} = x^{(k)} - \frac{u^{(k)}}{\rho_R(u^{(k)}) - 1},$$

which generally converges to the true solution faster than the original sequence converged.

This idea generalizes: if we have a sequence of approximations $x^{(0)}, \ldots, x^{(k)}$, why not ask for the "best" approximation that can be written as a linear combination of the $x^{(j)}$? This is the notion underlying Krylov methods, which we will discuss next time.

3 An Aside on Extrapolation

Many students are only exposed to extrapolation methods as a trick or an aside in a numerical methods course. This really does not do justice to a family of methods with a deep history and connections across the breadth of mathematics. Extrapolation methods are not only a sometimes-miraculous-seeming tool in numerics, but they are intimately connected to continued fractions and rational approximation (itself an under-studied area in modern times!), to questions in number theory (e.g. the proofs that e and π are transcendental), to time series analysis, to filtering and signal processing, and to many other pure and applied topics. They have been re-invented repeatedly across areas and cultures. Recent literature often names the methods after 20th-century mathematicians, but the ideas go back at least to the 17th century, and not just in Europe (what is now called Aitken's delta-squared process was known to 17th century Japanese mathematics as a way of accelerating series converging to π). I really like the historical survey by Brezinski for some of these connections and background.

The earliest extrapolation methods were seen as sequence transformations, converting one sequence into another sequence with faster convergence. Indeed, sometimes these methods convert divergent sequences into convergent ones, giving us the notion of an "anti-limit!" But I like to think of a hierarchy of methods that one can use depending on how much additional context one has:

- Standard extrapolation methods convert one sequence $x_1, x_2, \ldots$ to a new sequence $\check{x}_1, \check{x}_2, \ldots$ that converges more quickly. The transformation usually involves a model for the error that is fit to successive entries in the original sequence. But otherwise, we don't explicitly use anything about how the sequence is generated.

- Acceleration or mixing procedures (like Anderson acceleration or Pulay mixing) explicitly assume that we have a fixed point iteration $x^{(k+1)} = G(x^{(k)})$ with a known function G. Unlike more general sequence extrapolation methods, these methods apply G as part of the computation of the new sequence.

- Other methods, including the Krylov subspace methods that we are about to describe, are posed in terms of approximately solving a system of equations $F(x) = 0$ or minimizing some objective function $\phi(x)$. These methods construct a space of possible approximations from linear combinations of the iterates of another method, and then use some ansatz to choose the "best possible" approximation from that space. This "best possible" approximation might be one that minimizes some residual error $\|F(x)\|$ over the subspace, or it might minimize $\phi(x)$, or it might satisfy some other condition.

Methods that use more detailed knowledge of a system of equations or optimization problem we want to solve are less general than some of the classical methods – they aren't going to help us at all if we want to accelerate a sequence converging to π! But they also tend to be less numerically delicate than standard extrapolation measure, which my their nature often involve a lot of cancellation effects in order to estimate errors from successive steps.