Classical iterative methods for linear systems

Ed Bueler

MATH 615 Numerical Analysis of Differential Equations

27 February–1 March, 2017

example linear systems

- suppose we want to solve the linear system

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

  where $A \in \mathbb{R}^{m\times m}$ and $\mathbf{b} \in \mathbb{R}^m$, to find $\mathbf{x} \in \mathbb{R}^m$.

- throughout these notes we use just two examples:

  LS1

$$\begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix}$$

  LS2

$$\begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 1 & -2 & -3 \\ -1 & 1 & 1 & 0 \\ 0 & 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ 1 \\ 3 \end{bmatrix}$$

- on **P17** (Assignment #5) you will check that these are well-conditioned linear systems
- it is trivial to find solutions of LS1, LS2 using a "$\mathbf{x} = A\backslash\mathbf{b}$" black box, but these examples stand-in for the large linear systems we get from applying FD schemes to ODE and PDE problems

residual

- the *residual* of a vector $\mathbf{v}$ in linear system (1) is the vector

$$\mathbf{r}(\mathbf{v}) = \mathbf{b} - A\mathbf{v} \tag{2}$$

- making the residual zero is the same as solving the system:

$$A\mathbf{x} = \mathbf{b} \iff \mathbf{r}(\mathbf{x}) = 0$$

- evaluating $\mathbf{r}(\mathbf{v})$ needs a matrix-vector product and a vector subtraction
  - requires $O(m^2)$ operations at worst
  - by comparison, applying Gauss elimination to solve linear system (1) is an $O(m^3)$ operation in general
- FD schemes for DEs generate matrices A for which the majority, often 99% or more, of the entries are zero
  - a matrix with enough zeros to allow exploitation of that fact is called *sparse*
  - evaluating the residual of a sparse matrix typically requires $O(m)$ operations
  - even if A is sparse, A^{-1} is generally *dense*, i.e. most entries are nonzero

Richardson iteration

- *iterative methods* for linear system (1) attempt to solve it based only on operations like computing the residual, or applying A to a vector
  - one wants the sequence of approximations, the iterates, to *converge* to the solution $\mathbf{x} = A^{-1}\mathbf{b}$
  - Iterative methods always require an initial iterate $\mathbf{x}_0$
- *Richardson iteration* adds a multiple ω of the last residual at each step:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \omega(\mathbf{b} - A\mathbf{x}_k) \tag{3}$$

- for system LS1, using initial iterate $\mathbf{x}_0 = 0$ and $\omega = 1/5$, (3) gives:

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \mathbf{x}_1 = \begin{bmatrix} 0.4 \\ 0.2 \\ 0.8 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 0.6 \\ 0.16 \\ 1.04 \end{bmatrix}, \mathbf{x}_3 = \begin{bmatrix} 0.728 \\ 0.088 \\ 1.096 \end{bmatrix}, \dots, \mathbf{x}_{10} = \begin{bmatrix} 0.998 \\ -0.017 \\ 1.01 \end{bmatrix}, \dots$$

these iterates seem to be converging to $\mathbf{x} = [1\ 0\ 1]^\top$, which is the solution to LS1

recall: eigenvalues and vectors

- a complex number $\lambda \in \mathbb{C}$ is an *eigenvalue* of a square matrix $B \in \mathbb{R}^{m\times m}$ if there is a nonzero vector $\mathbf{v} \in \mathbb{C}^m$ so that $B\mathbf{v} = \lambda\mathbf{v}$
- the set of all eigenvalues of B is the *spectrum* $\sigma(B)$ of B
- the *spectral radius* $\rho(B)$ is the maximum absolute value of an eigenvalue:

$$\rho(B) = \max_{\lambda\in\sigma(B)} |\lambda|$$

- even if B is real, λ may be complex—the roots of a polynomial with real coefficients may be complex—*and* if λ is complex and B is real then $\mathbf{v}$ must be complex

spectral properties and convergence of iterations

- properties of a matrix B described in terms of eigenvalues are generically called *spectral properties*
- some examples:
  - $\rho(B)$
  - $\|B\|_2 = \sqrt{\rho(B^\top B)}$
  - the 2-norm condition number $\kappa(B) = \|B\|_2 / \|B^{-1}\|_2$
- a general idea:

  > *whether an iterative method for solving $A\mathbf{x} = \mathbf{b}$ converges, or not, depends on spectral properties of A, or on matrices built from A*

- the right-hand side $\mathbf{b}$ and the initial iterate $\mathbf{x}_0$ generally *do not* determine whether an iteration converges
  - a good choice of $\mathbf{x}_0$ *can* speed up convergence

convergence of the Richardson iteration

- rewrite the Richardson iteration (3) as

$$\mathbf{x}_{k+1} = (I - \omega A)\mathbf{x}_k + \omega \mathbf{b}$$

- the lemma on the next slide shows that the Richardson iteration converges if and only if all the eigenvalues of the matrix $I - \omega A$ are inside the unit circle:

$$\text{(3) converges if and only if } \rho(I - \omega A) < 1$$

  - ○ $\rho(I - \omega A) < 1$ means $(I - \omega A)\mathbf{x}_k$ is smaller in magnitude than $\mathbf{x}_k$
  - ○ *if* $\|I - \omega A\| < 1$ *then* (3) converges[1]

[1] recall $\rho(B) \leq \|B\|$ in any induced matrix norm

convergence lemma

Lemma

$$\mathbf{y}_{k+1} = M\mathbf{y}_k + \mathbf{c}$$

converges to the solution of $\mathbf{y} = M\mathbf{y} + \mathbf{c}$ *for all initial* $\mathbf{y}_0$ *if and only if*

$$\rho(M) < 1.$$

Proof.

Solve the iteration by writing out a few cases:

$$\begin{aligned}
\mathbf{y}_2 &= M(M\mathbf{y}_0 + \mathbf{c}) + \mathbf{c} = M^2\mathbf{y}_0 + (I + M)\mathbf{c}, \\
\mathbf{y}_3 &= M(M^2\mathbf{y}_0 + (I + M)\mathbf{c}) + \mathbf{c} = M^3\mathbf{y}_0 + (I + M + M^2)\mathbf{c}, \\
&\vdots
\end{aligned}$$

By induction we get $\mathbf{y}_k = M^k\mathbf{y}_0 + p_k(M)\mathbf{c}$ where $p_k(x) = 1 + x + x^2 + \cdots + x^{k-1}$. But $p_k(x) \to 1/(1 - x)$ as $k \to \infty$ iff $x \in (-1, 1)$. Also, $\rho(M) < 1$ iff $M^k \to 0$. Thus $\mathbf{y}_k \to (I - M)^{-1}\mathbf{c}$ iff $\rho(M) < 1$. □

convergence of the Richardson iteration 2

- since the Richardson iteration converges iff $\rho(I - \omega A) < 1$, we choose ω based on the principle that

$$\omega A \text{ should be close to the identity } I$$

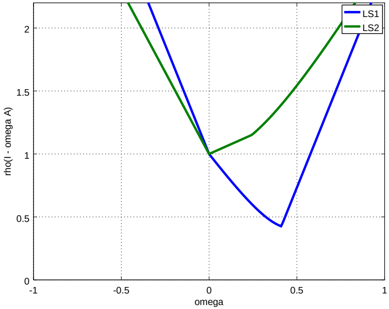
  - often not possible!

- in small cases we can graph $f(\omega) = \rho(I - \omega A)$:

```
omega = -1:.01:1;
rho = zeros(size(omega));
for j = 1:length(omega)
    M = eye(n) - omega(j) * A;
    rho(j) = max(abs(eig(M)));
end
plot(omega,rho)
```

for LS1: $\rho(I - \omega A)$ dips below 1 for $0 < \omega \lesssim 0.6$

for LS2: $\rho(I - \omega A) \geq 1$ always



- note $\rho(I - 0A) = 1 \quad \ldots \quad$ so no convergence when $\omega \approx 0$
- for LS1, figure suggests $\omega \approx 0.4$ gives fastest convergence

matrix splitting

- unlike Richardson, most classical iteration methods “split” the matrix A before iterating
- the best known, and simplest, iteration based on splitting is *Jacobi iteration*, which extracts the diagonal of A (and inverts it)
- the splitting we consider is

  $$A = D - L - U$$

  where
    - D is the diagonal of A
    - L is strictly lower triangular ($\ell_{ij} = 0$ if $i \le j$)
    - U is strictly upper triangular ($u_{ij} = 0$ if $i \ge j$)
- you can split *any* matrix this way
- see section 4.2 of the textbook

- so that D is an invertible matrix, for the remaining slides we assume *all diagonal entries of A are nonzero*: $\quad a_{ii} \neq 0$

Jacobi iteration

- the Jacobi iteration is

$$D\mathbf{x}_{k+1} = \mathbf{b} + (L+U)\mathbf{x}_k \tag{4}$$

- if it converges then $D\mathbf{x} = \mathbf{b} + (L+U)\mathbf{x}$, which is the same as $A\mathbf{x} = \mathbf{b}$
- we could also write it as $\mathbf{x}_{k+1} = D^{-1}\left(\mathbf{b} + (L+U)\mathbf{x}_k\right)$ or as

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j\neq i} a_{ij}x_j^{(k)}\right) \tag{5}$$

where $x_j^{(k)}$ denotes the jth entry of the kth iterate $\mathbf{x}_k$

- make sure you understand why (4) and (5) are the same!

Gauss-Seidel iteration

- *Gauss-Seidel iteration* extracts the non-strict lower-triangular part of A (and inverts it)
- again if $A = D - L - U$ then it is

$$(D - L)\mathbf{x}_{k+1} = b + U\mathbf{x}_k \tag{6}$$

- we could also write it "$\mathbf{x}_{k+1} = (D-L)^{-1}\left(b + U\mathbf{x}_k\right)$" but that would miss the point!
- instead we write it as $D\mathbf{x}_{k+1} = b + U\mathbf{x}_k + L\mathbf{x}_{k+1}$ or equivalently:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j>i} a_{ij} x_j^{(k)} - \sum_{j<i} a_{ij} x_j^{(k+1)}\right) \tag{7}$$

- the lower-triangular entries of A apply to *those entries of* $\mathbf{x}_{k+1}$ *which have already been computed*
- form (7) is actually *easier* to implement than Jacobi (5) (why?)

convergence conditions for Jacobi and Gauss-Seidel

- the convergence lemma says that
  - Jacobi iteration converges if and only if $\rho(D^{-1}(L+U)) < 1$
  - Gauss-Seidel iteration converges if and only if $\rho((D-L)^{-1}U) < 1$
- these conditions are hard to use in practice because computing a spectral radius can be just as hard as solving the original system

diagonally-dominant matrices

- *definition.* A is *strictly diagonally-dominant* if $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$
  - LS1 is strictly diagonally-dominant
  - LS2 is not
- two relatively-famous theorems[2] are these:
  - *theorem.* if A is strictly diagonally-dominant then both the Jacobi and Gauss-Seidel iterations converge
  - *Theorem.* if A is symmetric positive definite then Gauss-Seidel iteration converges
- unlike the "$\rho(\dots) < 1$" conditions on the last slide:
  - it is easy to check diagonal-dominance, and it is a common property of the matrices coming from FD schemes on ODEs and PDEs
  - these are only *sufficient* conditions, e.g. there are nonsymmetric A, which are *not* diagonally-dominant, but for which the iterations converge
- see problems **P19** and **P20**

[2] section 11.2 of Golub and van Loan, *Matrix Computations*, 4th edition 2013

past

- the Jacobi and Gauss-Seidel iterations are from the 19th century
  - Richardson iteration first appears in a 1910 publication
- the early history of numerical partial differential equations, e.g. in the 1920 to 1970 period, heavily used these classical iterations
  - a generalization of Gauss-Seidel iteration called *successive over-relaxation*, was a particular favorite around 1970; see section 4.2 of the textbook
- none of these iterations work on system LS2

recent past and future

- there are better iterative ideas, and they flourished starting in the 1980-90s . . . and far into the future
  - among the best known are CG = *conjugate gradients* (actually from 1950-60s) and GMRES = *generalized minimum residuals* (from a 1986 paper by Saad and Schultz)
  - GMRES works (i.e. converges at some rate) on LS2
  - *but* there can be no "good iteration" with a universally-fast convergence rate[3]
- iteration to solve linear systems is the future:
  - it is obligatory on sufficiently-big systems
  - it works better in parallel than direct methods like Gauss elimination
  - it can exploit partial knowledge of the underlying model

[3] remarkably, there is a 1992 theorem by Nachtigal, Reddy, and Trefethen that says this

biographies

- Gauss (1777–1855) did big stuff, not just the little Gauss-Seidel thing:

  `en.wikipedia.org/wiki/Carl_Friedrich_Gauss`

- Jacobi (1804–1851) also has his name on the "Jacobian", the matrix of derivatives appearing in Newton's method for systems of equations:

  `en.wikipedia.org/wiki/Carl_Gustav_Jacob_Jacobi`

- Seidel (1821–1896) is relatively little known:

  `en.wikipedia.org/wiki/Philipp_Ludwig_von_Seidel`

- Richardson (1881–1953) is the most interesting. He invented numerical weather forecasting, doing it by-hand for fun during WWI. Later, as a pacifist and quaker, he quit the subject entirely when he found his meteorological work was of most value to chemical weapons engineers and the British Air Force:

  `en.wikipedia.org/wiki/Lewis_Fry_Richardson`