STAT 309: MATHEMATICAL COMPUTATIONS I
FALL 2015
LECTURE 16

1. WHY ITERATIVE METHODS

- if we have a linear system $A\mathbf{x} = \mathbf{b}$ where A is very, very large but is either sparse or structured (e.g., banded, Toeplitz, banded plus low-rank, semiseparable, Hierarchical, etc), the easiest way to exploit this is to use *iterative methods*
- these are methods that construct a sequence of vectors $\mathbf{x}^{(k)}$ so that $\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x} = A^{-1}\mathbf{b}$
- we shall focus on solving linear systems but there are also iterative methods for least squares problems, eigenvalue problems, singular value problems, etc — in fact for the last two, there are only iterative methods
- one big advantage of iterative methods is that we can control how accurate we want our solution, for example, if we want our solution to be ε-accurate (whether relative or absolute), then in principle we can stop as soon as

$$\|\mathbf{x}^{(k)} - \mathbf{x}\| < \varepsilon \quad \text{or} \quad \frac{\|\mathbf{x}^{(k)} - \mathbf{x}\|}{\|\mathbf{x}\|} < \varepsilon \tag{1.1}$$

- if, say, $n = 10,000$ but it takes only $k = 5$ iterations to reach our desired level of accuracy, then we have saved a lot of computations — direct methods like LU, QR, Cholesky, etc, do not allow this
- in practice of course we do not know $\mathbf{x} = A^{-1}\mathbf{b}$ and it might appear that we can't use forward errors like those in (1.1) to control accuracy but we will see later that we don't need to know $\mathbf{x}$ to gurantee (1.1)
- usually iterative methods converge in the limit to the solution but there are iterative methods that actually converge in finitely many steps
- for example, many *Krylov subspace methods* converge in k steps where k = number of distinct nonzero eigenvalues of A:
  - conjugate gradient (CG) method for symmetric positive definite A
  - minimal residual (MINRES) method for symmetric A
  - general minimial resitual (GMRES) method for general A
- there are three classes of iterative methods for $A\mathbf{x} = \mathbf{b}$
  - *splitting methods*: decompose A into the sum of two matrices

    $$A = M - N$$

    where M is easy to invert and then do

    $$M\mathbf{x}^{(k)} = N\mathbf{x}^{(k-1)} + \mathbf{b}$$

    these are also known as *one-step stationary methods*
  - *semi-iterative methods*: generate

    $$\mathbf{y}^{(k)} = B\mathbf{y}^{(k-1)} + \mathbf{c}$$

Date: December 2, 2015, version 1.1. Comments, bug reports: lekheng@galton.uchicago.edu.

for suitable B and $\mathbf{c}$ and then form

$$\mathbf{x}^{(k)} = \sum_{j=0}^{k} \alpha_{jk} \mathbf{y}^{(j)}$$

– *Krylov subspace methods*: find

$$\mathbf{x}^{(k)} \in \operatorname{span}\{\mathbf{b}, A\mathbf{b}, A^2\mathbf{b}, \ldots, A^k\mathbf{b}\}$$

in a way that approximates the solution, i.e., $\mathbf{x}^{(k)} \approx \mathbf{x}$, in some sense

- splitting methods and semi-iterative methods are often called *stationary methods* to distinguish them from Krylov subspace methods (although this is not so clear cut — for example, conjugate gradient method, the oldest Krylov subspace method, may also be viewed as a semi-iterative method)

2. splitting methods

- we want to solve $A\mathbf{x} = \mathbf{b}$ for $A \in \mathbb{R}^{n\times n}$ nonsingular
- we pick a suitable *splitting*

$$A = M - N$$

where M is nonsingular and easy to invert (not explicitly but in the sense that it is easy to solve $M\mathbf{x} = \mathbf{b}$ for any $\mathbf{b}$)

- from $A\mathbf{x} = \mathbf{b}$, we get

$$M\mathbf{x} = N\mathbf{x} + \mathbf{b} \tag{2.1}$$

- this inspires the iteration

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b} \tag{2.2}$$

- subtracting (2.2) from (2.1), we obtain

$$M(\mathbf{x} - \mathbf{x}^{(k+1)}) = N(\mathbf{x} - \mathbf{x}^{(k)})$$

- if we denote the *error* in $\mathbf{x}^{(k)}$ by $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$, then

$$\mathbf{e}^{(k+1)} = M^{-1}N\mathbf{e}^{(k)} =: B\mathbf{e}^{(k)}$$

- thus $\mathbf{e}^{(k)} = B\mathbf{e}^{(k)} = B^{k+1}\mathbf{e}^{(0)}$
- note that

$$\mathbf{x}^{(k)} \to \mathbf{x} \quad \text{if and only if} \quad \mathbf{e}^{(k)} \to \mathbf{0} \quad \text{if and only if} \quad \|\mathbf{e}^{(k)}\| \to 0$$

- the matrix $B = M^{-1}N$ is somtimes called the *iteration matrix*
- its spectral radius $\rho(B)$ governs convergence rate, i.e., how quickly the error goes to zero
- recall that if $\rho(B^k) < 1$ then $\mathbf{e}^{(k)} \to \mathbf{0}$ for all choices of $\mathbf{x}^{(0)}$
- we have the following theorem:

Theorem 1. $\mathbf{e}^{(k)} \to \mathbf{0}$ *as* $k \to \infty$ *for all* $\mathbf{e}^{(0)}$ *if and only if* $\rho(B) < 1$.

Proof. Note that $\mathbf{e}^{(k)} = B^{k+1}\mathbf{e}^{(0)} \to \mathbf{0}$ for all $\mathbf{e}^{(0)}$ is equivalent to $\lim_{k\to\infty} B^k = O$ (the zero matrix) since we could choose $\mathbf{e}^{(0)}$ to be each of the standard basis vectors $\mathbf{e}_1, \ldots, \mathbf{e}_n$ in turn and so we get

$$B^k = B^k I = B^k[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n] = [B^k\mathbf{e}_1, B^k\mathbf{e}_2, \ldots, B^k\mathbf{e}_n] \to [\mathbf{0}, \mathbf{0}, \ldots, \mathbf{0}] = O$$

as $k \to \infty$. Now by what we discussed in an earlier lecture (about the Jordan form), for a Jordan block,

$$J_r^k = \begin{bmatrix} \lambda_r^k & \binom{k}{1}\lambda_r^{k-1} & \binom{k}{2}\lambda_r^{k-2} & \cdots & \binom{k}{n_r-1}\lambda_r^{k-(n_r-1)} \\ & \ddots & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & \vdots \\ & & & & \lambda_r^k \end{bmatrix} \to O$$

as $k \to \infty$. Since B has a Jordan decomposition,

$$B = X \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_m \end{bmatrix} X^{-1},$$

we have

$$B^k = X \begin{bmatrix} J_1^k & & \\ & \ddots & \\ & & J_m^k \end{bmatrix} X^{-1} \to X \begin{bmatrix} O & & \\ & \ddots & \\ & & O \end{bmatrix} X^{-1} = O$$

as $k \to \infty$. □

- convergence can still occur if $\rho(B) = 1$, but in that case we must be careful in how we choose $\mathbf{x}^{(0)}$
- recall also that for all consistent norms,

  $$\rho(B) \leq \|B\|$$

  and

  $$\|B^k\| \leq \|B\|^k$$
- from $\mathbf{e}^{(k)} = B^k\mathbf{e}^{(0)}$, it follows that

  $$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{e}^{(0)}\|} \leq \|B\|^k$$
- so if we find a consistent norm with $\|B\| < 1$, then this gives a sufficient condition for convergence
- note that convergence does not depend on the choice of norms since on finite-dimensional spaces, all norms are equivalent
- if we can prove statements like $\|B^k\| \to 0$ or $\|\mathbf{e}^{(k)}\| \to 0$ for any one norm, we know that it will hold for all norms

3. CONVERGENCE RATE

- formally, for a sequence $\mathbf{x}_k$ that converges to $\mathbf{x}$, its *convergence rate* $r \in (0, 1)$ is defined to be

  $$r = \limsup_{k\to\infty} \frac{\|\mathbf{e}^{(k+1)}\|}{\|\mathbf{e}^{(k)}\|} = \limsup_{k\to\infty} \frac{\|\mathbf{x}^{(k+1)} - \mathbf{x}\|}{\|\mathbf{x}^{(k)} - \mathbf{x}\|}$$

  or alternatively, the smallest $r \in (0, 1)$ such that

  $$\|\mathbf{e}^{(k+1)}\| \leq r\|\mathbf{e}^{(k)}\| \qquad \text{for all } k \text{ sufficiently large}$$
- a sequence that has such a property is called *linearly convergent* and we will often say that an iterative algorithm is linearly convergent for a class of problem if it generates a linearly convergent sequence for all choices of initial points $\mathbf{x}^{(0)}$

- if
$$\limsup_{k\to\infty} \frac{\|\mathbf{e}^{(k+1)}\|}{\|\mathbf{e}^{(k)}\|} = 0,$$
we say that the sequence (resp. algorithm) is *superlinearly convergent*
- if there exists $M > 0$ such that
$$\|\mathbf{e}^{(k+1)}\| \leq M\|\mathbf{e}^{(k)}\|^2 \qquad \text{for all } k \text{ sufficiently large,}$$
we say that the sequence (resp. algorithm) is *quadratically convergent*
- note that M does not need to be in $(0, 1)$
- more generally the largest p for which there exists $M > 0$ such that
$$\|\mathbf{e}^{(k+1)}\| \leq M\|\mathbf{e}^{(k)}\|^p \qquad \text{for all } k \text{ sufficiently large,}$$
is called the *order of convergence*

4. Jacobi method

- the simplest splitting is to take M to be the diagonal part of A and $-N$ to be the off-diagonal part — this works as long as the diagonal elements of A is nonzero (but the iterates may not converge)
- if we write $A\mathbf{x} = \mathbf{b}$ in coordinate form,
$$\sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, \dots, n,$$
then
$$a_{ii}x_i = b_i - \sum_{i\neq j} a_{ij}x_j,$$
or
$$x_i = \frac{1}{a_{ii}}\Big[b_i - \sum_{j\neq i} a_{ij}x_j\Big] \tag{4.1}$$
- in other words,
$$M = \begin{bmatrix} a_{11} & & & \\ & \ddots & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}, \quad N = -\begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}$$
- our iteration is therefore
$$x_i^{(k+1)} = \frac{1}{a_{ii}}\Big[b_i - \sum_{j\neq i} a_{ij}x_j^{(k)}\Big],$$
known as the *Jacobi method*
- if we write $A = L + D + U$ where
$$L = \begin{bmatrix} 0 & & & \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}, \quad D = \begin{bmatrix} a_{11} & & & \\ & \ddots & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}, \quad U = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & \ddots & & \vdots \\ & & \ddots & a_{n-1,n} \\ & & & 0 \end{bmatrix}$$
the the Jacobi mathod can be written in matrix form as
$$D\mathbf{x}^{(k+1)} = -(L + U)\mathbf{x}^{(k)} + \mathbf{b} \tag{4.2}$$

- the iteration matrix is

$$M^{-1}N = -\begin{bmatrix} 0 & \frac{a_{12}}{a_{11}} & \cdots & \frac{a_{1n}}{a_{11}} \\ \frac{a_{21}}{a_{22}} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \frac{a_{n1}}{a_{nn}} & \cdots & \frac{a_{n,n-1}}{a_{nn}} & 0 \end{bmatrix} =: B_J$$

- so if

$$\|B_J\|_\infty = \max_{1\le i\le n} \sum_{j\ne i} \left|\frac{a_{ij}}{a_{ii}}\right| < 1,$$

  i.e., if A is *strictly diagonally dominant*, then the iteration converges
- therefore, a sufficient condition for convergence of the Jacobi method is $\|B_J\|_\infty < 1$ where

$$b_{ij} = \begin{cases} -\dfrac{a_{ij}}{a_{ii}} & i \ne j, \\ 0 & i = j \end{cases}$$

- for example, suppose

$$A = \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix},$$

  then $\|B_J\|_\infty = \frac{1}{2}$ and so the Jacobi method converges rapidly
- on the other hand, if

$$A = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix},$$

  which arises from discretizing the one-dimensional Laplacian, then $\|B_J\|_\infty = 1$
- a more subtle analysis can be used to show convergence in this case, but convergence is slow
- note that for these two examples, $\mathbf{x}^{(0)} \leftarrow \mathbf{x}^{(1)}$ when *all* elements of $\mathbf{x}^{(1)}$ have been computed — this is a waste of storage; we need only $n+2$ elements of storage of A above
- this shows that the ordering of equations is very important
- if we reorder the equations in such a way that odd-numbered equations and even-numbered equations are grouped separately, then we obtain, for the latter example,

$$A = \begin{bmatrix} 2 & & & & & -1 & & & \\ & 2 & & & & -1 & -1 & & \\ & & \ddots & & & & -1 & -1 & \\ & & & \ddots & & & & \ddots & \ddots \\ & & & & & & & & \\ -1 & -1 & & & & 2 & & & \\ & -1 & -1 & & & & 2 & & \\ & & \ddots & \ddots & & & & \ddots & \\ & & & \ddots & \ddots & & & & \ddots \\ & & & & -1 & & & & 2 \end{bmatrix}$$

- then, we can solve for all odd indices, then all even indicies, independently of each other
- not only does this approach save storage space but it also lends itself to parallelism

5. Gauss–Seidel method

- in the Jacobi method, we compute $x_i^{(k+1)}$ using the elements of $\mathbf{x}^{(k)}$, even though $x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}$ are already known

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

- a general adage in numerical computations is: *use the latest information available*
- the *Gauss–Seidel* method is designed to take advantage of the latest information available about $\mathbf{x}$:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right] \tag{5.1}$$

- if we write $A = L + D + U$ where

$$L = \begin{bmatrix} 0 & & & \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix}, \quad D = \begin{bmatrix} a_{11} & & & \\ & \ddots & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}, \quad U = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & \ddots & & \vdots \\ & & \ddots & a_{n-1,n} \\ & & & 0 \end{bmatrix}$$

  then the Gauss–Seidel iteration can be written in matrix form as

$$D\mathbf{x}^{(k+1)} = \mathbf{b} - L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)},$$

  or

$$(D + L)\mathbf{x}^{(k+1)} = -U\mathbf{x}^{(k)} + \mathbf{b} \tag{5.2}$$

  which yields

$$\mathbf{x}^{(k+1)} = -(D + L)^{-1} U\mathbf{x}^{(k)} + (D + L)^{-1}\mathbf{b}$$

- thus the iteration matrix for the Gauss–Seidel method is

$$B_{GS} = -(D + L)^{-1} U$$

  as opposed to the iteration matrix for the Jacobi method

$$B_J = -D^{-1}(L + U)$$

- in some cases (cf. last line of Section 9)

$$\rho(B_{GS}) = \rho(B_J)^2$$

  so the Gauss–Seidel method converges twice as fast
- on the other hand, note that Gauss–Seidel is very sequential, i.e., it does not lend itself to parallelism
- note that the matrix forms for Jacobi and Gauss–Seidel (4.2) and (5.2) are only convenient representations useful in mathematical analysis of the methods, one should *never* implement these algorithms in such forms, instead use (4.1) and (5.1)
- we saw earlier that a sufficient condition for convergence of the Jacobi method is $\|B_J\|_\infty < 1$ where

$$b_{ij} = \begin{cases} -\dfrac{a_{ij}}{a_{ii}} & i \neq j, \\ 0 & i = j \end{cases}$$

- since
$$\|B_J\|_\infty = \max_i \sum_{j\neq i} \left|\frac{a_{ij}}{a_{ii}}\right| < 1,$$
this is equivalent to saying that A is strictly diagonally dominant
- we will see that this is also enough to guarantee the convergence of Gauss–Seidel, i.e., if A is strictly diagonally dominant, then Gauss–Seidel is convergent
- define
$$r_i = \sum_{j\neq i} \left|\frac{a_{ij}}{a_{ii}}\right|, \quad r = \max_i r_i$$

Theorem 2. *If $r < 1$, then $\rho(B_{GS}) < 1$, i.e., the Gauss–Seidel iteration converges if A is strictly diagonally dominant.*

Proof. The proof proceeds using induction on the elements of $\mathbf{e}^{(k)}$. We have
$$(D + L)\mathbf{e}^{(k+1)} = -U\mathbf{e}^{(k)},$$
which can be written as
$$\sum_{j=1}^{i} a_{ij} e_j^{(k+1)} = -\sum_{j=i+1}^{n} a_{ij} e_j^{(k)}, \quad i = 1, \ldots, n.$$
Thus
$$e_i^{(k+1)} = -\sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} e_j^{(k)} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} e_j^{(k+1)}, \quad i = 1, \ldots, n.$$
For $i = 1$, we have
$$|e_1^{(k+1)}| \leq \sum_{j=2}^{n} \left|\frac{a_{ij}}{a_{11}}\right| |e_j^{(k)}| \leq r_1 \|\mathbf{e}^{(k)}\|_\infty \leq r\|\mathbf{e}^{(k)}\|_\infty.$$
Assume that for $p = 1, \ldots, i-1$,
$$|e_p^{(k+1)}| \leq \|\mathbf{e}^{(k)}\|_\infty r_p \leq r\|\mathbf{e}^{(k)}\|_\infty.$$
Then,
$$\begin{aligned}
|e_i^{(k+1)}| &\leq \sum_{j=1}^{i-1} \left|\frac{a_{ij}}{a_{ii}}\right| |e_j^{(k+1)}| + \sum_{j=i+1}^{n} \left|\frac{a_{ij}}{a_{ii}}\right| |e_j^{(k)}| \\
&\leq r\|\mathbf{e}^{(k)}\|_\infty \sum_{j=1}^{i-1} \left|\frac{a_{ij}}{a_{ii}}\right| + \|\mathbf{e}^{(k)}\|_\infty \sum_{j=i+1}^{n} \left|\frac{a_{ij}}{a_{ii}}\right| \\
&\leq \|\mathbf{e}^{(k)}\|_\infty \sum_{j\neq i} \left|\frac{a_{ij}}{a_{ii}}\right| \\
&= r_i \|\mathbf{e}^{(k)}\|_\infty \\
&\leq r\|\mathbf{e}^{(k)}\|_\infty.
\end{aligned}$$
Therefore
$$\|\mathbf{e}^{(k+1)}\|_\infty \leq r\|\mathbf{e}^{(k)}\|_\infty \leq r^{k+1}\|\mathbf{e}^{(0)}\|_\infty,$$
from which it follows that
$$\lim_{k\to\infty} \|\mathbf{e}^{(k)}\|_\infty = 0$$
since $r < 1$. □

- while both the Jacobi method and the Gauss–Seidel method both converge if A is diagonally dominant, convergence can be slow in some cases
- for example, for

$$A = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n \times n}$$

  we have

$$-D^{-1}(L+U) = \begin{bmatrix} 0 & 1/2 & & \\ 1/2 & \ddots & \ddots & \\ & \ddots & \ddots & 1/2 \\ & & 1/2 & 0 \end{bmatrix}$$

  and therefore

$$\rho(B_J) = \cos\frac{\pi}{n+1} = \cos \pi h \approx 1 - \frac{\pi^2 h^2}{2} + \cdots$$

  which is approximately 1 for small $h = 1/(n+1)$
- suppose $B_J = B_J^{\mathsf{T}}$, then

$$\frac{\|\mathbf{e}^{(k)}\|_2}{\|\mathbf{e}^{(0)}\|_2} \leq \|B_J\|_2^k = \rho(B_J)^k$$

- if we want $\|\mathbf{e}^{(k)}\|_2 / \|\mathbf{e}^{(0)}\|_2 \leq \varepsilon$, then setting $\rho^k = \varepsilon$, we get that

$$k = \frac{-\log \varepsilon}{-\log \rho}$$

  is the number of iterations necessary for convergence
- so $\rho = \rho(A)$ controls the rate of convergence

6. SOR method

- another general adage in numerical computations is: *don't discard previous information, try to use it too*
- applying this to Gauss–Seidel, we could try to use both $x_j^{(k+1)}$ and $x_j^{(k)}$ for $j = 1, \ldots, i-1$ to obtain $x_i^{(k+1)}$ — this yields the method of *successive over relaxation* (SOR)
- this is given by the iteration

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right] + (1-\omega) x_i^{(k)}$$

  or

$$\mathbf{x}_{\text{SOR}}^{(k+1)} = \omega \mathbf{x}_{\text{GS}}^{(k+1)} + (1-\omega) \mathbf{x}_{\text{SOR}}^{(k)}$$

- the quantity ω is called the *relaxation parameter*
- if $\omega = 1$, then the SOR method reduces to the Gauss–Seidel method, i.e.,

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

- the name 'over relaxation' comes from choosing $\omega > 1$

- in matrix form, the iteration can be written as

$$D\mathbf{x}^{(k+1)} = \omega(\mathbf{b} - L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)}) + (1-\omega)D\mathbf{x}^{(k)}$$

  which can be rearranged to obtain

$$(D + \omega L)\mathbf{x}^{(k+1)} = \omega\mathbf{b} + [(1-\omega)D - \omega U]\mathbf{x}^{(k)}$$

  or

$$\mathbf{x}^{(k+1)} = \left(\frac{1}{\omega}D + L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D - U\right]\mathbf{x}^{(k)} + \left(\frac{1}{\omega}D + L\right)^{-1}\mathbf{b} \tag{6.1}$$

- the iteration matrix is

$$B_\omega = \left(\frac{1}{\omega}D + L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D - U\right]$$

- since $B_1 = B_{GS}$, if we pick some $\omega \neq 1$ such that

$$\rho(B_\omega) < \rho(B_1),$$

  we would improve the convergence of Gauss–Seidel
- so SOR is at least as fast as Gauss–Seidel and often faster
- in fact, we will see later that for certain types of matrices, one can pick ω so that $\rho(B_\omega)$ is minimized
- note that if $A\mathbf{x} = \mathbf{b}$, then

$$D\mathbf{x} = \omega(\mathbf{b} - L\mathbf{x} - U\mathbf{x}) + (1-\omega)D\mathbf{x}$$

  and so

$$\mathbf{x} = \left(\frac{1}{\omega}D + L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D - U\right]\mathbf{x}^* + \left(\frac{1}{\omega}D + L\right)^{-1}\mathbf{b} \tag{6.2}$$

- subtracting (6.2) from (6.1), we get

$$\mathbf{e}^{(k+1)} = B_\omega\mathbf{e}^{(k)}$$

- note that

$$\begin{aligned}
\det B_\omega &= \det\left(\frac{1}{\omega}D + L\right)^{-1}\det\left[\left(\frac{1}{\omega} - 1\right)D - U\right] \\
&= \frac{1}{\det\left(\frac{1}{\omega}D + L\right)}\det\left[\left(\frac{1}{\omega} - 1\right)D - U\right] \\
&= \frac{\omega^n}{\prod_{i=1}^n a_{ii}}\frac{(1-\omega)^n\prod_{i=1}^n a_{ii}}{\omega^n} \\
&= (1-\omega)^n
\end{aligned}$$

- therefore $\prod_{i=1}^n \lambda_i = (1-\omega)^n$ where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of B_ω, with $|\lambda_1| \geq \cdots \geq |\lambda_n|$
- hence we get

$$|\lambda_1|^n \geq (1-\omega)^n$$

  which means that we must have $|\lambda_1| = \rho(B_\omega) < 1$ for convergence
- it follows that a necessary condition for convergence of SOR is

$$0 < \omega < 2$$

- if A is symmetric positive definite, then the condition $0 < \omega < 2$ is also sufficient — a result of Ostrowski implies that for such an A, $\rho(B_\omega) < 1$ iff $0 < \omega < 2$

- suppose $A \in \mathbb{R}^{n\times n}$ is a symmetric matrix, then $U = L^\mathsf{T}$ and if we set
$$M_\omega = \frac{\omega}{2-\omega}\left(\frac{1}{\omega}D + L\right)D^{-1}\left(\frac{1}{\omega}D + L^\mathsf{T}\right)$$
and define our iteration as
$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k-1)} - M_\omega^{-1}(A\mathbf{x}^{(k)} - \mathbf{b})$$
- one may also define a nonlinear version of SOR for iterations of the form $\mathbf{x}^{(k+1)} = f(\mathbf{x}^{(k)})$ where f is some nonlinear function $f : \mathbb{R}^n \to \mathbb{R}^n$:
$$\mathbf{x}_{\text{SOR}}^{(k+1)} = (1-\omega)\mathbf{x}_{\text{SOR}}^{(k)} + \omega f(\mathbf{x}_{\text{SOR}}^{(k)})$$

7. block Gauss–Seidel

- consider the standard problem of solving *Poisson equation* on a domain R in two dimensions,
$$\begin{aligned} -\Delta u &= f, \quad (x,y) \in R,\\ u &= g, \quad (x,y) \in \partial R \end{aligned}$$
- here $\Delta u = u_{xx} + u_{yy}$ denotes the two-dimensional Laplacian
- let us take R to be the unit rectangle $[0,1]\times[0,1]$ and discretize the problem using a uniform grid with spacing $h = 1/(n+1)$ in the x and y directions, and gridpoints $x_i = ih$, $i = 0,\dots,n+1$, and $y_j = jh$, $j = 0,\dots,n+1$
- then, for $i,j = 1,\dots,n$, the differential equation may be replaced by a difference approximation
$$\frac{-u_{i-1,j} + 2u_{ij} - u_{i+1,j}}{h^2} + \frac{-u_{i,j+1} + 2u_{ij} - u_{i,j+1}}{h^2} = f_{ij},$$
where $u_{ij} = u(x_i, y_j)$ and $f_{ij} = f(x_i, y_j)$
- from the boundary conditions, we have
$$u_{0j} = g(x_0, y_j), \quad j = 1,2,\dots,n,$$
and similar conditions for the other gridpoints along the boundary
- if we write $\mathbf{u}_j = [u_{1j},\dots,u_{nj}]^\mathsf{T}$, we have
$$-\mathbf{u}_{j-1} + T\mathbf{u}_j - \mathbf{u}_{j+1} = \mathbf{f}_j$$
where
$$T = \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix}, \quad [\mathbf{f}_j]_i = \begin{cases} h^2 f_{1j} + g(x_0, j) & i = 1,\\ h^2 f_{ij} & i = 2,\dots,n-1,\\ h^2 f_{Nj} + g(x_N, j) & i = N \end{cases}$$
- thus we can solve the problem on the entire domain by solving $A\mathbf{u} = \mathbf{f}$ where
$$A = \begin{bmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & T \end{bmatrix}$$
- we say that A is a *block tridiagonal matrix*
- we can solve $A\mathbf{u} = \mathbf{f}$ using an iteration
$$T\mathbf{u}^{(k+1)} = \mathbf{f}_j + \mathbf{u}_{j-1}^{(k)} + \mathbf{u}_{j+1}^{(k)}$$

- this is an example of a *block Jacobi* iteration, since it involves solving the system $A\mathbf{u} = \mathbf{f}$ by applying the Jacobi method to A, except each block of size $n \times n$ is treated as a single element
- similarly, we can use the *block Gauss–Seidel* iteration

$$T\mathbf{u}_j^{(k+1)} = \mathbf{f}_j + \mathbf{u}_{j-1}^{(k+1)} + \mathbf{u}_j^{(k)}$$

- note A is also a *banded* matrix but the band is sparse and Gaussian elimination may fill-in the whole band
- however, the equations can be re-ordered to avoid fill-in

8. PROPERTY A

- let A be symmetric positive definite so that in particular $a_{ii} > 0$ for $i = 1, \ldots, n$
- we can use diagonal scaling to obtain a matrix $D^{-1/2}AD^{-1/2}$ with all diagonal elements equal to 1 by setting

$$D = \begin{bmatrix} a_{11} & & \\ & \ddots & \\ & & a_{nn} \end{bmatrix}$$

- this matrix has all kinds of nice properties; in particular, it allows decoupling of equations
- a matrix $A \in \mathbb{R}^{n \times n}$ is said to have *Property A* if there is a permutation matrix Π such that

$$\Pi^{\mathsf{T}} A \Pi = \begin{bmatrix} I_p & F \\ F^{\mathsf{T}} & I_q \end{bmatrix} \tag{8.1}$$

- for example, suppose we have a tridiagonal matrix of the form

$$A = \begin{bmatrix} 1 & a_1 & & & \\ a_1 & \ddots & \ddots & & \\ & \ddots & \ddots & a_{n-1} & \\ & & a_{n-1} & 1 & \end{bmatrix}$$

- then by choosing Π so that odd-numbered rows and columns are grouped together, followed by even-numbered rows and columns, we obtain

$$\Pi^{\mathsf{T}} A \Pi = \begin{bmatrix} 1 & & & & a_1 & & & \\ & \ddots & & & a_2 & \ddots & & \\ & & \ddots & & & \ddots & \ddots & \\ & & & 1 & & & \ddots & \ddots \\ a_1 & a_2 & & & 1 & & & \\ & \ddots & \ddots & & & \ddots & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \ddots & & & & \ddots \end{bmatrix}$$

- it should be noted that a matrix arising from the discretization of a PDE in two dimensions using a 5-point stencil (like what we have in the previous section for Poisson equation) has Property A, but a matrix based on a 9-point stencil does not
- however a matrix based on a 9-point stencil does have *block Property A*

- for example, if

$$A = \begin{bmatrix} A_1 & B_1 & & \\ B_1^\mathsf{T} & \ddots & \ddots & \\ & \ddots & \ddots & B_{n-1} \\ & & B_{n-1}^\mathsf{T} & A_n \end{bmatrix}$$

- then we can choose Π so that

$$\Pi^\mathsf{T} A \Pi = \begin{bmatrix} A_1 & & & & B_1^\mathsf{T} & & & \\ & A_3 & & & B_2^\mathsf{T} & B_3^\mathsf{T} & & \\ & & \ddots & & & \ddots & \ddots & \\ & & & 1 & & & \ddots & \ddots \\ B_1 & B_2 & & & A_2 & & & \\ & B_3 & \ddots & & & A_4 & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & \ddots & & & & \ddots \end{bmatrix}$$

9. optimal sor parameter

- we now show that for a matrix of the form (8.1), we can choose an optimal parameter ω for the SOR method
- let F be a $p \times q$ matrix with $p \geq q$, and let $F = U\Sigma V^\mathsf{T}$ be the SVD of F
- so we have

$$A = \begin{bmatrix} UU^\mathsf{T} & U\Sigma V^\mathsf{T} \\ V\Sigma^\mathsf{T} U^\mathsf{T} & VV^\mathsf{T} \end{bmatrix} = \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} I & \Sigma \\ \Sigma^\mathsf{T} & I \end{bmatrix} \begin{bmatrix} U^\mathsf{T} & 0 \\ 0 & V^\mathsf{T} \end{bmatrix}$$

- since the left and right matrices above denote a similarity transformation, it follows that

$$\lambda(A) = \lambda(\tilde{A}), \quad \tilde{A} = \begin{bmatrix} I & \Sigma \\ \Sigma^\mathsf{T} & I \end{bmatrix}$$

- reordering the rows and columns of $\tilde{A}$, we obtain a block diagonal matrix, where each diagonal block is a 2×2 matrix of the form

$$\begin{bmatrix} 1 & \sigma_i \\ \sigma_i & 1 \end{bmatrix}, \quad i = 1, \ldots, q$$

- the eigenvalues of $\tilde{A}$ are the eigenvalues of all of these diagonal blocks, which are $\lambda = 1 \pm \sigma_i$
- these eigenvalues must be positive since A is positive definite, so it follows that

$$0 < \sigma_i < 1, \quad i = 1, \ldots, q$$

- now consider the SOR operator

$$\begin{aligned} \mathcal{L}_\omega &= \left(\frac{1}{\omega} I + L\right)^{-1} \left(\left(\frac{1}{\omega} - 1\right) I - U\right) \\ &= \begin{bmatrix} \frac{1}{\omega} I & 0 \\ F^\mathsf{T} & \frac{1}{\omega} I \end{bmatrix}^{-1} \begin{bmatrix} \left(\frac{1}{\omega} - 1\right) I & -F \\ 0 & \left(\frac{1}{\omega} - 1\right) I \end{bmatrix} \end{aligned}$$

where

$$L = \begin{bmatrix} 0 & 0 \\ F^\mathsf{T} & 0 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & F \\ 0 & 0 \end{bmatrix}$$

- we can explicitly invert the first matrix to obtain

$$\mathcal{L}_\omega = \begin{bmatrix} \omega I & 0 \\ -\omega^2 F^\mathsf{T} & \omega I \end{bmatrix} \begin{bmatrix} \left(\frac{1}{\omega} - 1\right) I & -F \\ 0 & \left(\frac{1}{\omega} - 1\right) I \end{bmatrix} = \begin{bmatrix} (1-\omega)\, I & -\omega F \\ (\omega^2 - \omega) F^\mathsf{T} & (1-\omega)\, I + \omega^2 F^\mathsf{T} F \end{bmatrix}$$

- using the SVD of F again, we obtain

$$\begin{aligned} \mathcal{L}_\omega &= \begin{bmatrix} (1-\omega)\, U U^\mathsf{T} & -\omega U \Sigma V^\mathsf{T} \\ (\omega^2 - \omega) V \Sigma^\mathsf{T} U^\mathsf{T} & (1-\omega)\, V V^\mathsf{T} + \omega^2 V \Sigma^\mathsf{T} \Sigma V^\mathsf{T} \end{bmatrix} \\ &= \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} (1-\omega)\, I & -\omega \Sigma \\ (\omega^2 - \omega) \Sigma^\mathsf{T} & (1-\omega)\, I + \omega^2 \Sigma^\mathsf{T} \Sigma \end{bmatrix} \begin{bmatrix} U^\mathsf{T} & 0 \\ 0 & V^\mathsf{T} \end{bmatrix} \end{aligned}$$

- define

$$\Gamma(\omega) = \begin{bmatrix} (1-\omega)\, I & -\omega \Sigma \\ (\omega^2 - \omega) \Sigma^\mathsf{T} & (1-\omega)\, I + \omega^2 \Sigma^\mathsf{T} \Sigma \end{bmatrix}$$

- then $\lambda(\mathcal{L}_\omega) = \lambda(\Gamma(\omega))$ and $\|\mathcal{L}_\omega\|_2 = \|\Gamma(\omega)\|_2$
- recall that

$$\mathbf{e}^k = \mathcal{L}_\omega^k \mathbf{e}^{(0)}$$

- ideally, we want to choose ω so that $\|\mathcal{L}_\omega^k\|$ is minimized, but this is an open problem
- however, David Young showed how to compute ω so that $\rho(\mathcal{L}_\omega)$ is minimized
- since each block of $\Gamma(\omega)$ is a diagonal matrix, we can use the same reordering trick as before to obtain a block diagonal matrix, where each diagonal block is a 2×2 matrix of the form

$$\Gamma_i = \begin{bmatrix} (1-\omega) & -\omega\sigma_i \\ (\omega^2 - \omega)\sigma_i & (1-\omega) + \omega^2 \sigma_i^2 \end{bmatrix}, \quad i = 1, \ldots, q$$

- the eigenvalues μ of Γ_i satisfy the characteristic equation

$$(1 - \omega - \mu)^2 - \mu \sigma_i^2 \omega^2 = 0$$

- note that when $\omega = 0$, then $|\mu| = 1$, indicating divergence
- if $\omega = 1$, corresponding to the Gauss–Seidel method, then $\mu = 0$ or $\mu = \sigma_i^2$. If $\omega = 2$, then the eigenvalues are complex conjugates with $|\mu| = 1$
- therefore there exists an ω where μ becomes complex:

$$\hat{\omega} = \frac{2}{1 + \sqrt{1 - \sigma_i^2}}$$

- thus, $|\mu(\omega_1)| > |\mu(\omega_2)|$ for $\omega_1 > \omega_2 > \hat{\omega}$
- note that the eigenvalues of the Gauss–Seidel matrix are 0 or σ_i^2, while the eigenvalues of the Jacobi matrix are $\pm\sigma_i$
- therefore we can expect Gauss–Seidel to converge twice as fast as Jacobi for matrices with Property A