Solving linear systems: iterative methods

Recall that one of the ways to find a solution of a scalar equation $f(x) = 0$ was to rewrite the equation in the form of a fixed point equation $x = \phi(x)$. Then we iterate the map ϕ, i.e. use the recursive formula

$$x_{k+1} = \phi(x_k) \qquad \text{with an initial guess } x_0,$$

to generate a sequence $x_1, x_2, x_3, \ldots$ that, under certain conditions, converges to a fixed point x^* of ϕ, which will be a solution of the original equation $f(x) = 0$.

In a similar manner, we can use an iterated matrix formula to generate a sequence of vectors

$$\mathbf{x}^{(1)},\ \mathbf{x}^{(2)},\ \mathbf{x}^{(3)}, \ldots$$

to approximate the solution of a linear system $A\mathbf{x} = \mathbf{b}$.
Note that it is customary to reserve *subscripts* for denoting the *components* of a vector (e.g. x_i denotes the ith component of the vector $\mathbf{x}$), so the *iterate number* (i.e. the index of the sequence) is usually written as a *superscript within parentheses*.

To derive a recursive formula from $A\mathbf{x} = \mathbf{b}$, we try to rewrite it in the form

$$P\mathbf{x} = Q\mathbf{x} + \mathbf{b}$$

so that $\mathbf{x}$ appears on both sides of the equation (in other words, split A into a difference of two matrices P and Q). Then, we iterate the formula

$$(\star) \qquad\qquad P\mathbf{x}^{(k+1)} = Q\mathbf{x}^{(k)} + \mathbf{b}$$

and hope that it generates a sequence which converges to the solution $\mathbf{x}^*$ of $A\mathbf{x} = \mathbf{b}$.
Depending on the choice of P and Q, it is sometimes straightforward to write $(\star)$ more explicitly as

$$\mathbf{x}^{(k+1)} = M\mathbf{x}^{(k)} + \mathbf{c},$$

which makes it easier to perform the iterations. M is called the "iteration matrix".

The simplest iterative method is the **Jacobi iteration scheme**.
For example, consider the linear system

$$\begin{aligned} 7x_1 + 2x_2 - 3x_3 + x_4 &= -1 \\ -4x_1 + 8x_2 + x_3 - 2x_4 &= 16 \\ 3x_1 - 2x_2 + 9x_3 + 2x_4 &= -3 \\ x_1 - x_2 + x_3 + 4x_4 &= 5 \end{aligned} \qquad \text{where} \quad A = \begin{pmatrix} 7 & 2 & -3 & 1 \\ -4 & 8 & 1 & -2 \\ 3 & -2 & 9 & 2 \\ 1 & -1 & 1 & 4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} -1 \\ 16 \\ -3 \\ 5 \end{pmatrix}$$

We rewrite each equation by moving all the off-diagonal terms to the right-hand side:

$$\begin{aligned} 7x_1 &= -1 - (2x_2 - 3x_3 + x_4) \\ 8x_2 &= 16 - (-4x_1 + x_3 - 2x_4) \\ 9x_3 &= -3 - (3x_1 - 2x_2 + 2x_4) \\ 4x_4 &= 5 - (x_1 - x_2 + x_3) \end{aligned}$$

The recursive formula is then

$$\begin{aligned}
7x_1^{(k+1)} &= -(2x_2^{(k)} - 3x_3^{(k)} + x_4^{(k)}) - 1 \\
8x_2^{(k+1)} &= -(-4x_1^{(k)} + x_3^{(k)} - 2x_4^{(k)}) + 16 \\
9x_3^{(k+1)} &= -(3x_1^{(k)} - 2x_2^{(k)} + 2x_4^{(k)}) - 3 \\
4x_4^{(k+1)} &= -(x_1^{(k)} - x_2^{(k)} + x_3^{(k)}) + 5
\end{aligned}
\quad \text{or} \quad
\begin{aligned}
x_1^{(k+1)} &= \frac{-(2x_2^{(k)} - 3x_3^{(k)} + x_4^{(k)})}{7} - \frac{1}{7} \\
x_2^{(k+1)} &= \frac{-(-4x_1^{(k)} + x_3^{(k)} - 2x_4^{(k)})}{8} + \frac{16}{8} \\
x_3^{(k+1)} &= \frac{-(3x_1^{(k)} - 2x_2^{(k)} + 2x_4^{(k)})}{9} - \frac{3}{9} \\
x_4^{(k+1)} &= \frac{-(x_1^{(k)} - x_2^{(k)} + x_3^{(k)}) + 5}{4} + \frac{5}{4}
\end{aligned}$$

Here P is a diagonal matrix whose elements are just the diagonal entries a_{ii} of A, and Q is a matrix with zeros along the diagonal and whose other terms are negative the corresponding terms in A (i.e. $-a_{ij}$). In this case, it is easy to see that the iteration matrix M is just Q with each term divided by the diagonal element of A in that row (i.e. $-a_{ij}/a_{ii}$).

We shall write Matlab code to perform Jacobi iterations and test it on this system.
What happens if we switch the first two equations around (i.e. interchange row 1 and row 2 of A)?

Under what conditions does an iterative method converge?

THEOREM An iterative method with iteration matrix M converges if and only if all the eigenvalues of M have magnitude strictly less than 1.

DEFINITION The matrix A is said to be *strictly diagonal dominant by row* if, in each row, the magnitude of the diagonal term is strictly greater than the sum of the magnitudes of the off-diagonal terms:

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \qquad \text{for } i = 1, \ldots, n.$$

THEOREM If A is strictly diagonal dominant by row, then the Jacobi iteration scheme converges to the solution of $A\mathbf{x} = \mathbf{b}$.

An important enhancement of the Jacobi scheme is the **Gauss–Seidel iteration scheme**. For the example above, the Gauss–Seidel formula is

$$\begin{aligned}
7x_1^{(k+1)} &= -(2x_2^{(k)} - 3x_3^{(k)} + x_4^{(k)}) - 1 \\
8x_2^{(k+1)} &= -(-4x_1^{(k+1)} + x_3^{(k)} - 2x_4^{(k)}) + 16 \\
9x_3^{(k+1)} &= -(3x_1^{(k+1)} - 2x_2^{(k+1)} + 2x_4^{(k)}) - 3 \\
4x_4^{(k+1)} &= -(x_1^{(k+1)} - x_2^{(k+1)} + x_3^{(k+1)}) + 5
\end{aligned}$$

In other words, we update the components of $\mathbf{x}$ one by one, starting with x_1, and use the *updated* value of x_1 to compute the new x_2, then use the updated values of x_1 and x_2 to compute the new x_3, and so on. If we move the "new" $(k+1)$th iterates back to the left-hand side, the system looks like

$$\begin{aligned}
7x_1^{(k+1)} &= -(2x_2^{(k)} - 3x_3^{(k)} + x_4^{(k)}) - 1 \\
-4x_1^{(k+1)} + 8x_2^{(k+1)} &= -(x_3^{(k)} - 2x_4^{(k)}) + 16 \\
3x_1^{(k+1)} - 2x_2^{(k+1)} + 9x_3^{(k+1)} &= -(2x_4^{(k)}) - 3 \\
x_1^{(k+1)} - x_2^{(k+1)} + x_3^{(k+1)} + 4x_4^{(k+1)} &= 5
\end{aligned}$$

So P is a lower triangular matrix `tril(A)` while Q is upper triangular with zeros along the diagonal.

Measuring magnitude, distance and error for vectors

For two- or three-dimensional vectors, the most common way of measuring magnitude is based on the Pythagorean theorem:

$$\text{if } \mathbf{u} = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2, \text{ then the magnitude of } \mathbf{u} \text{ is } \sqrt{x^2+y^2}$$

$$\text{if } \mathbf{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3, \text{ then the magnitude of } \mathbf{v} \text{ is } \sqrt{x^2+y^2+z^2}$$

We can generalize this to n-dimensional vectors:

$$\text{if } \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n, \quad \text{then the magnitude of } \mathbf{x} \text{ is } \sqrt{x_1^2+x_2^2+\cdots+x_3^2}$$

This measure of magnitude is called the "Euclidean norm", denoted by $\|\cdot\|_2$, so

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}$$

There are many other "norms" (ways of measuring magnitude).
For each $p = 1, 2, 3, \ldots, \infty$, the "$p$-norm" of $\mathbf{x}$ in $\mathbb{R}^n$ or $\mathbb{C}^n$ is defined as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$$

Important special cases are:

- $p = 1$: $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$ (the "grid" norm)
- $p = 2$: $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$ (the "Euclidean" norm)
- $p = \infty$: $\|\mathbf{x}\|_\infty = \max_{i=1,2,\ldots,n} |x_i|$ (the "max" norm—picks out the maximum absolute value of the vector's components)

Different norms may be suitable for different applications, but the 1-norm, 2-norm and ∞-norm are the most frequently used. They are accessed in Matlab using `norm(x,1)`, `norm(x)` and `norm(x,inf)`, respectively (the default is the 2-norm).

The distance between two vectors $\mathbf{v}$ and $\mathbf{w}$ is $\|\mathbf{v}-\mathbf{w}\|$, i.e. the norm of the difference of the vectors.

Recall the various *stopping criteria* that we used for iteratively generated (scalar) sequences; these all have vector analogues, where the absolute values are replaced by norms. For instance:

$$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| \leq \text{tolerance}$$

$$\frac{\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|}{\|\mathbf{x}^{(k)}\|} \leq \text{tolerance}$$

$$\|\mathbf{r}\| = \|\mathbf{b} - A\mathbf{x}^{(k)}\| \leq \text{tolerance}$$

The vector $\mathbf{r} := \mathbf{b} - A\mathbf{x}^{(k)}$ is called the "residual" (we expect it to be the zero vector if $\mathbf{x}^{(k)}$ is equal to the exact solution).