Introduction to Iterative Methods for Solving $Ax = b$

One of the most important problems in mathematics is to find the values of $x_1, x_2, \ldots, x_n$ that satisfy the system of equations

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}.$$

That is, we want to solve for $\boldsymbol{x}$ in $A\boldsymbol{x} = \boldsymbol{b}$ where

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

Suppose we had the true values of $x_2, x_3, \ldots, x_n$. Then we could use these values, along with the first equation (or any other equation where the x_1 coefficient is nonzero), to find the true value of x_1. We usually won't have the true values of $x_2, x_3, \ldots, x_n$, but suppose we had pretty good approximations (or guesses) for $x_2, x_3, \ldots, x_n$. Then we could use these values to find a pretty good approximation for x_1. Similarly, we probably could find a pretty good approximation for any x_i if we had good approximations for the values of $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$. The idea that is emerging here is that if we currently have approximations for the values of $x_1, x_2, \ldots, x_n$, then we can use these current values to find new (and hopefully better!) approximations for $x_1, x_2, \ldots, x_n$.

The fundamental idea of an iterative method is to use $\boldsymbol{x}^{current}$, a current approximation (or guess) for the true solution $\boldsymbol{x}$ (where $A\boldsymbol{x} = \boldsymbol{b}$), to find a new approximation $\boldsymbol{x}^{new}$, where $\boldsymbol{x}^{new}$ is closer to $\boldsymbol{x}$ than $\boldsymbol{x}^{current}$ is. We then use this new approximation as the current approximation to find yet another, better approximation. In fact, *iterate* simply means *repeat*, so an iterative method repeats this process over and over, each time using the current approximation to produce a better approximation for the true solution, until the current approximation is sufficiently close to the true solution (or until you realize that the sequence of approximations resulting from these iterations is not converging to the true solution). So given $\boldsymbol{x}^{(0)}$, an initial guess or approximation for the true solution $\boldsymbol{x}$, use $\boldsymbol{x}^{(0)}$ to find (we'll discuss how in a minute) a new approximation $\boldsymbol{x}^{(1)}$, then use $\boldsymbol{x}^{(1)}$ to find yet another, better approximation $\boldsymbol{x}^{(2)}$, and so on. In general, we use $\boldsymbol{x}^{(k)}$ to find a new approximation $\boldsymbol{x}^{(k+1)}$. The expectation is that $\boldsymbol{x}^{(k)} \to \boldsymbol{x}$ as $k \to \infty$. Of course, we hope that after a relatively small number of iterations our current approximation $\boldsymbol{x}^{(k)}$ is close enough to $\boldsymbol{x}$ for our needs.

Note that since we don't actually have the true solution $\boldsymbol{x}$ (if we did, why would we be trying to find it?), we can't check to see how close our current approximation $\boldsymbol{x}^{(k)}$ is to $\boldsymbol{x}$. One common way to check the closeness of $\boldsymbol{x}_k$ to $\boldsymbol{x}$ is by checking instead how close $A\boldsymbol{x}^{(k)}$ is to $A\boldsymbol{x}$; that is, how close $A\boldsymbol{x}^{(k)}$ is to $\boldsymbol{b}$. Another way to check the accuracy of our current approximation is by looking at the difference in successive approximations $\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$. The idea is that we generally expect $\boldsymbol{x}^{(k)}$ to close to $\boldsymbol{x}$ if $\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$ is small.

By the way, this idea of iteration is certainly not unique to linear algebra. An example that is familiar to all of us is Newton's Method, which finds approximations to the root x of $f(x)=0$. Given a current approximation $x^{(k)}$ to x, Newton's Method is to use $x^{(k)}$ to find $x^{(k+1)}$ using

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}.$$

Under the right conditions, $x^{(k)} \to x$ (that is, $f(x^{(k)}) \to 0$) as $k \to \infty$. Interestingly, the underlying theory of Newton's Method is actually found in certain iterative methods that solve our problem $A\boldsymbol{x} = \boldsymbol{b}$. Those methods are discussed in numerical linear algebra courses.

Jacobi's Method

Perhaps the simplest approach to designing an iterative method for solving $A\boldsymbol{x} = \boldsymbol{b}$ is Jacobi's Method. The strategy of this method is to use the first equation and the current values of $x_2^{(k)}, x_3^{(k)}, \ldots, x_n^{(k)}$ to find a new value $x_1^{(k+1)}$, and similarly to find a new value $x_i^{(k+1)}$ using the $i^{\underline{\text{th}}}$ equation and the old values of the other variables. That is, given current values $x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}$, find new values by solving for $x_1^{(k+1)}, x_2^{(k+1)}, \ldots, x_n^{(k+1)}$ in

$$\begin{array}{ccccccccc}
a_{11}x_1^{(k+1)} & + & a_{12}x_2^{(k)} & + & \cdots & + & a_{1n}x_n^{(k)} & = & b_1 \\
a_{21}x_1^{(k)} & + & a_{22}x_2^{(k+1)} & + & \cdots & + & a_{2n}x_n^{(k)} & = & b_2 \\
\vdots & & \vdots & & \ddots & & \vdots & & \vdots \\
a_{n1}x_1^{(k)} & + & a_{n2}x_2^{(k)} & + & \cdots & + & a_{nn}x_n^{(k+1)} & = & b_n
\end{array}.$$

This can also be written as

$$\begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^{(k+1)} + \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1\,n} \\ a_{n1} & \cdots & a_{n\,n-1} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^{(k)} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

To be clear, we point out that in $x_i^{(k)}$, i is the $i^{\underline{\text{th}}}$ element of vector $(x_1^{(k)}, x_2^{(k)}, \ldots, x_i^{(k)}, \ldots, x_n^{(k)})$, and k is the particular *iteration* (not the $k^{\underline{\text{th}}}$ power of x).

Where D, L and U are the *diagonal*, *lower triangular* and *upper triangular* parts of A, respectively,

$$D = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{nn} \end{bmatrix}, \quad L = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a_{n1} & \cdots & a_{n\,n-1} & 0 \end{bmatrix} \text{ and } U = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1\,n} \\ 0 & \cdots & 0 & 0 \end{bmatrix},$$

then Jacobi's Method can be written more concisely in matrix-vector notation as

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

so that

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

Example 1 Apply Jacobi's Method to the system

$$\begin{array}{rcrcrcr} 4x_1 & - & x_2 & - & x_3 & = & 3 \\ -2x_1 & + & 6x_2 & + & x_3 & = & 9 \\ -x_1 & + & x_2 & + & 7x_3 & = & -6 \end{array}.$$

At each step, given the current values $x_1^{(k)}, x_2^{(k)}, x_3^{(k)}$, solve for $x_1^{(k+1)}, x_2^{(k+1)}, x_3^{(k+1)}$ in

$$\begin{array}{rcrcrcr} 4x_1^{(k+1)} & - & x_2^{(k)} & - & x_3^{(k)} & = & 3 \\ -2x_1^{(k)} & + & 6x_2^{(k+1)} & + & x_3^{(k)} & = & 9 \\ -x_1^{(k)} & + & x_2^{(k)} & + & 7x_3^{(k+1)} & = & -6 \end{array}$$

So if our initial guess $\boldsymbol{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)})$ is the zero vector $\boldsymbol{0} = (0,0,0)$—a common initial guess unless we have some additional information which causes us to choose some other initial guess—then we find $\boldsymbol{x}^{(1)} = (x_1^{(1)}, x_2^{(1)}, x_3^{(1)})$ by solving for them in

$$\begin{array}{rcrcrcr} 4x_1^{(1)} & - & 0 & - & 0 & = & 3 \\ -2\cdot 0 & + & 6x_2^{(1)} & + & 0 & = & 9 \\ -0 & + & 0 & + & 7x_3^{(1)} & = & -6 \end{array}$$

So $\boldsymbol{x}^{(1)} = (x_1^{(1)}, x_2^{(1)}, x_3^{(1)}) = (3/4,\ 9/6, -6/7) = (\,0.750,\ 1.500, -0.857)$. We iterate this process to find a sequence of increasingly better approximations $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \ldots$. We are interested in the error between the true solution $\boldsymbol{x}$ and the approximation $\boldsymbol{x}^{(k)}$ at each iteration: $\boldsymbol{error}^{(k)} = \boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$. Obviously, we wouldn't normally have the true solution $\boldsymbol{x}$. However, in order to better understand the behavior of an iterative method, it is very enlightening to use the method in solving a system $A\boldsymbol{x} = \boldsymbol{b}$ where we *do* know the true solution, and analyze how quickly the approximations it produces are converging to the true solution. For this example, the true solution is $\boldsymbol{x} = (\,1,\ 2, -1)$. In case you haven't yet learned about the norm of a vector, we briefly describe it (actually, there are multiple norms, so we will describe the most common and useful norm). If $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, then the norm of vector $\boldsymbol{x}$ is $\|\boldsymbol{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$. The norm $\|\boldsymbol{x}\|$ is simply a way to measure the *length* or *size* of a vector. Notice, for example, in the following table, that the norm of the *error* $\|\boldsymbol{e}^{(k)}\|$ is becoming progressively smaller. This means that the approximations are becoming progressively better.

k	$\boldsymbol{x}^{(k)}$			$\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$			$\boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$			$\lVert \boldsymbol{e}^{(k)} \rVert$	$\dfrac{\lVert \boldsymbol{e}^{(k)} \rVert}{\lVert \boldsymbol{e}^{(k-1)} \rVert}$
0	0.000	0.000	0.000	—	—	—	1.000	2.000	-1.000	2.449	—
1	0.750	1.500	-0.857	0.750	1.500	-0.857	0.250	0.500	-0.143	0.557	0.236
2	0.911	1.893	-0.964	0.161	0.393	-0.107	0.089	0.107	-0.036	0.144	0.250
3	0.982	1.964	-0.997	0.071	0.071	-0.033	0.018	0.036	-0.003	0.040	0.278
4	0.992	1.994	-0.997	0.010	0.029	0.000	0.008	0.006	-0.003	0.011	0.269
5	0.999	1.997	-1.000	0.007	0.003	-0.003	0.001	0.003	0.000	0.003	0.310
6	0.999	2.000	-1.000	0.000	0.003	0.001	0.001	0.000	0.000	0.001	0.288
7	1.000	2.000	-1.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.366
8	1.000	2.000	-1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.339

For this example, we stop iterating after all three ways of measuring the current error $\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$, $\boldsymbol{e}^{(k)}$, and $\lVert \boldsymbol{e}^{(k)} \rVert$ are = 0 to three decimal places. In practice, you would normally choose just one of the three measurements of error in determining when to stop.

Gauss-Seidel Method

Let us now take Jacobi's Method one step further. Where the true solution is $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, if $x_1^{(k+1)}$ is a better approximation to the true value of x_1 than $x_1^{(k)}$ is, then it would make sense that once we have found the *new* value $x_1^{(k+1)}$ to use it (rather than the *old* value $x_1^{(k)}$) in finding $x_2^{(k+1)}, \ldots, x_n^{(k+1)}$. So $x_1^{(k+1)}$ is found just as before, but in finding $x_2^{(k+1)}$, instead of using the *old* value of $x_1^{(k)}$ and the old values $x_3^{(k)}, \ldots, x_n^{(k)}$, we now use the *new* value $x_1^{(k+1)}$ and the old values $x_3^{(k)}, \ldots, x_n^{(k)}$ in order to find $x_2^{(k+1)}$, and similarly for finding $x_3^{(k+1)}, \ldots, x_n^{(k+1)}$. This technique is called the Gauss-Seidel Method (even though, as noted by Gilbert Strang in his text, "Introduction to Applied Mathematics," Gauss didn't know about it and Seidel didn't recommend it), and is described by:

$$\begin{array}{ccccccccc}
a_{11}x_1^{(k+1)} & + & a_{12}x_2^{(k)} & + & \cdots & + & a_{1n}x_n^{(k)} & = & b_1 \\
a_{21}x_1^{(k+1)} & + & a_{22}x_2^{(k+1)} & + & \cdots & + & a_{2n}x_n^{(k)} & = & b_2 \\
\vdots & & \vdots & & \ddots & & \vdots & & \vdots \\
a_{n1}x_1^{(k+1)} & + & a_{n2}x_2^{(k+1)} & + & \cdots & + & a_{nn}x_n^{(k+1)} & = & b_n
\end{array}.$$

This can also be written,

$$\begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a_{n1} & \cdots & a_{n\,n-1} & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^{(k+1)} + \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1\,n} \\ 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^{(k)} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

That is,

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

so that

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

Example 2 Apply the Gauss-Seidel Method to the system from Example 1:

$$\begin{array}{rcrcrcr} 4x_1 & - & x_2 & - & x_3 & = & 3 \\ -2x_1 & + & 6x_2 & + & x_3 & = & 9 \\ -x_1 & + & x_2 & + & 7x_3 & = & -6 \end{array}.$$

At each step, given the current values $x_1^{(k)}, x_2^{(k)}, x_3^{(k)}$, solve for $x_1^{(k+1)}, x_2^{(k+1)}, x_3^{(k+1)}$ in

$$\begin{array}{rcrcrcr} 4x_1^{(k+1)} & - & x_2^{(k)} & - & x_3^{(k)} & = & 3 \\ -2x_1^{(k+1)} & + & 6x_2^{(k+1)} & + & x_3^{(k)} & = & 9 \\ -x_1^{(k+1)} & + & x_2^{(k+1)} & + & 7x_3^{(k+1)} & = & -6 \end{array}.$$

In order to compare our results using the Gauss-Seidel Method to our results when using the Jacobi Method, we again choose $\boldsymbol{x}^{(0)} = \boldsymbol{0}$. We then find $\boldsymbol{x}^{(1)} = (x_1^{(1)}, x_2^{(1)}, x_3^{(1)})$ by solving for them in

$$\begin{array}{rcrcrcr} 4x_1^{(1)} & - & 0 & - & 0 & = & 3 \\ -2x_1^{(1)} & + & 6x_2^{(1)} & + & 0 & = & 9 \\ -x_1^{(1)} & + & x_2^{(1)} & + & 7x_3^{(1)} & = & -6 \end{array}.$$

Let us be clear about how we solve this. We first solve for $x_1^{(1)}$ in the first equation, and find that $x_1^{(1)} = 3/4 = 0.750$. We then solve for $x_2^{(1)}$ in the second equation, using the new value of $x_1^{(1)} = 0.750$, and find that $x_2^{(1)} = [9 + (2)(0.750)]/6 = 1.750$. Finally, we solve for $x_3^{(1)}$ using the third equation, using the new values of $x_1^{(1)} = 0.750$ and $x_2^{(1)} = 1.750$, and find that $x_3^{(1)} = [-6 + 0.750 - 1.750]/7 = -1.000$. The result then of this first iteration of the Gauss-Seidel Method is $\mathbf{x}^{(1)} = (0.750, 1.750, -1.000)$. We iterate this process to find a sequence of increasingly better approximations $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \ldots$ and find the same sort of table as in Example 1:

k	$\boldsymbol{x}^{(k)}$			$\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$			$\boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$			$\lVert \boldsymbol{e}^{(k)} \rVert$	$\frac{\lVert \boldsymbol{e}^{(k)} \rVert}{\lVert \boldsymbol{e}^{(k-1)} \rVert}$
0	0.000	0.000	0.000	—	—	—	1.000	2.000	-1.000	2.449	—
1	0.750	1.750	-1.000	0.750	1.750	-1.000	0.250	0.250	0.000	0.354	0.144
2	0.938	1.979	-1.006	0.188	0.229	-0.006	0.063	0.021	0.006	0.066	0.187
3	0.993	1.999	-1.001	0.056	0.020	0.005	0.007	0.001	0.001	0.007	0.104
4	0.999	2.000	-1.000	0.006	0.001	0.001	0.001	0.000	0.000	0.001	0.075
5	1.000	2.000	-1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.052

As we did for the previous example, we again stop iterating after all three ways of measuring the current error $\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$, $\boldsymbol{e}^{(k)}$, and $\lVert \boldsymbol{e}^{(k)} \rVert$ are = 0 to three decimal places. Notice that this sequence of iterations converges to the true solution $(1,-2,1)$ much more quickly than we found in Example 1 using the Jacobi Method. This is generally expected, since with the Gauss-Seidel Method we use new values as we find them, rather than waiting until the subsequent iteration to use any new values, as is done with the Jacobi Method.