Useful properties of matrix and vector norms

We previously saw that

$$\|Ax\| \leq ||A|| \cdot ||x||$$  \hspace{1cm} (1)

for any matrix $A$, and any vector $x$ (of dimensions $m \times m$ and $m \times 1$, respectively).

Note that, when writing an expression such as (1), the matrix norm $||A||$ is understood to be the inferred norm from the vector norm used in $||Ax||$ and $||x||$. Thus,

$$||Ax||_1 \leq ||A||_1 \cdot ||x||_1$$

and

$$||Ax||_\infty \leq ||A||_\infty \cdot ||x||_\infty$$

are both valid, but we cannot mix and match, e.g.:

$$||Ax||_\infty \leq ||A||_2 \cdot ||x||_1$$

When solving a linear system $Ax = b$, computer algorithms are only providing an approximation ($x_{\text{approx}}$) to the exact solution ($x_{\text{exact}}$). This is due to factors such as finite precision, roundoff errors or even imperfect solution algorithms. In either case, we have an error (error vector, in fact) defined as

$$e = x_{\text{approx}} - x_{\text{exact}}$$

Naturally, we would like to have an understanding of the magnitude of this error (e.g., some appropriate norm $||e||$). The problem is that we do not know the exact, pristine solution $x_{\text{exact}}$.

One remedy is offered via the residual vector defined as:

$$r = b - Ax_{\text{approx}}$$
The vector $r$ is something we can compute practically since it involves only known quantities ($b, A, x_{\text{approx}}$). Furthermore, we have:

$$
    r = b - Ax_{\text{approx}} \\
    = Ax_{\text{exact}} - Ax_{\text{approx}} \\
    = -A(x_{\text{approx}} - x_{\text{exact}}) \\
    = -Ae \\
\Rightarrow r = -Ae \\
\Rightarrow e = -A^{-1}r
$$

The last equation links the error with the residual. Furthermore, we can write

$$
    ||e|| = ||A^{-1}r|| \leq ||A^{-1}|| \cdot ||r||
$$

This equation provides a bound for the error, as a function of $||A^{-1}||$ and the norm of the computable vector $r$! Note that:

- We can obtain this estimate without knowing the exact solution, but
- We need $||A^{-1}||$ and generally, computing $||A^{-1}||$ is just as difficult (if not more) than finding $x_{\text{exact}}$. However, there are special cases where an estimate of $||A^{-1}||$ can be obtained.

### A different source of error

Sometimes, the right hand side ($b$) of $Ax = b$ has errors that make it deviate from its intended value. For example, in the Vandermonde matrix method for polynomial interpolation, $b$ contains the samples $(y_1 = f(x_1), y_2, \ldots, y_n)$ where $y_i = f(x_i)$. An error in a measuring device supposed to sample $f(x)$ could lead to erroneous readings $y_i^* = f(x_i)$ instead of $y_i$. In general, measuring inaccuracies can lead to the right hand side vector $b$ being misrepresented as $b^* (\neq b)$.

In this case, instead of the intended solution $x = A^{-1}b$, we in fact compute $x^* = A^{-1}b^*$. How important is the error $e = x^* - x$ that is caused by this misrepresentation of $b$?

Let us introduce some notation. Let $\delta b = b^* - b, \delta x = x^* - x, Ax = b, Ax^* = b^*$. Then

$$
    A(x^* - x) = b^* - b \\
    A\delta x = \delta b \\
    \delta x = A^{-1}\delta b
$$

Taking norms,

$$
    ||\delta x|| = ||A^{-1}\delta b|| \leq ||A^{-1}|| \cdot ||\delta b||
$$

Thus, the error in the computed solution $\delta x$ is proportional to the error in $b$. 

2
An even more relevant question is: How does the relative error $||\delta x||/||x|| = ||x^* - x||/||x||$ compare to the relative error in $b$ ($||\delta b||/||b||$)? This may be more useful to know, since $||\delta b||$ may be impossible to compute (if we don’t know the real $b$!). For this, we write

$$Ax = b \implies ||b|| = ||Ax|| \leq ||A|| \cdot ||x||$$

Multiplying equations (2) and (3) gives

$$\frac{||\delta x||}{||x||} \leq ||A|| \cdot ||A^{-1}|| \cdot \frac{||\delta b||}{||b||}$$

Thus, the relative error in $x$ is bounded by a multiple of the relative error in $b$! The multiplicative constant $\kappa(A) = ||A|| \cdot ||A^{-1}||$ is called the condition number of $A$, and is an important measure of the sensitivity of a linear system $Ax = b$ to being solved on a computer, in the presence of inaccurate values. For e.g., if the relative error $||\delta b||/||b||$ is .0001%, but $\kappa(A) = 100.000$ (could happen!), then we could have up to a 10% error in the computed $x$!

**Why is this always relevant?**

Simply, almost any $b$ will have some small relative error due to the fact that it is represented on a computer up to machine precision! The relative error will be at least as much as the machine epsilon due to roundoff!

$$\frac{||\delta b||}{||b||} \geq \varepsilon \approx 10^{-7} \quad \text{(in single precision)}$$

But how bad can the condition number get? Very bad at times. For example, Hilbert matrices $H_n \in \mathbb{R}^{n \times n}$ are defined as

$$(H_n)_{ij} = \frac{1}{i + j - 1}$$

Considering a specific instance for $n = 5$,

$$H_5 = \begin{bmatrix}
1 & 1/2 & 1/3 & 1/4 & 1/5 \\
1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\
1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\
1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\
1/5 & 1/6 & 1/7 & 1/8 & 1/9
\end{bmatrix}, \quad \kappa_\infty(H_5) = ||H_5||_\infty \cdot ||H_5^{-1}||_\infty \approx 10^6$$

Thus, any attempt at solving $H_5x = b$ would be subject to a relative error up to 10% just due to roundoff errors in $b$!

Another case: near-singular matrices

$$A = \begin{bmatrix}
1 & 2 \\
3 & 6 + \varepsilon
\end{bmatrix}$$

As $\varepsilon \to 0$, $A$ becomes singular (non-invertible). In this case, $\kappa(A) \to \infty$. 

3
What is the best case for $\kappa(A)$?

**Lemma 1.** For any vector-induced matrix norm, we have $||I|| = 1$.

**Proof.** From the definition,

$$||I|| = \max_{x \neq 0} \frac{||Ix||}{||x||} = \max_{x \neq 0} \frac{||x||}{||x||}$$

Using property (iv) of matrix norms gives

$$I = A \cdot A^{-1} \Rightarrow 1 = ||I|| = ||A \cdot A^{-1}|| \leq ||A|| \cdot ||A^{-1}||$$

Thus, $\kappa(A) \geq 1$. The “best” conditioned matrices are of the form $A = c \cdot I$, and have $\kappa(A) = 1$.

**Solving linear systems of equations**

Our general strategy for solving a system $Ax = b$ will be to transform it to an equivalent, but easier to solve problem (or problems). An example of an easier sub-problem is a triangular system $Ux = b$, where

$$U = \begin{bmatrix} u_{11} & u_{12} & \ldots & u_{1n} \\ 0 & u_{22} & \ldots & u_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & u_{nn} \end{bmatrix}$$

is an upper triangular matrix. Here is an example, illustrating how such systems are easy to solve:

$$\begin{bmatrix} 1 & 2 & 2 \\ 0 & -4 & -6 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -6 \\ 1 \end{bmatrix}$$

From the 3rd row, solve $x_3 = -1$ and replace $x_3$ in the previous (2nd) equation:

$$\begin{bmatrix} 1 & 2 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -12 \\ -1 \end{bmatrix}$$

From the 2nd row, solve $x_2 = 3$ and replace $x_2$ in the previous (1st) equation:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -3 \\ -1 \end{bmatrix}$$
We can write this procedure formally in pseudo-code:

Algorithm 1 Back substitution for upper triangular system

1: for $j = n \ldots 1$ do
2: if $u_{jj} = 0$ then
3: return. $\triangleright$ matrix is singular
4: end if
5: $x_j \leftarrow b_j/u_{jj}$
6: for $i = 1 \ldots j - 1$ do
7: $b_i \leftarrow b_i - u_{ij}x_j$
8: end for
9: end for

By counting how many times the loops are executed, we see that $n$ divisions are required, and $\sum_{j=1}^{n} (j - 1) = O(n^2)$ multiplications and subtractions. Overall, the cost of back substitution is $O(n^2)$. 