MATH 612
Computational methods for equation solving and function minimization – Week # 5

Instructor: Francisco-Javier 'Pancho' Sayas

Spring 2014 – University of Delaware

Plan for this week

- Discuss any problems you couldn't solve previous lectures
- You have to read lectures 12, 13, and 14
- I'll go over some aspects of these lectures and try to get to Lecture 15
- The second HW assignment is due Friday

Remember that...

... I'll keep on updating, correcting, and modifying the slides until the end of each week.

CONDITIONING

The condition number of a matrix

Let A be a square invertible matrix. Its condition number is the quantity

$$\kappa(A) = \|A\| \, \|A^{-1}\|$$

- $\kappa(A)$ depends on the norm we use, so we should be tagging it with the same name as the norm we use
- $\kappa(A^{-1}) = \kappa(A)$
- $\kappa(AB) \leq \kappa(A)\kappa(B)$ (because $\|A\,B\| \leq \|A\| \, \|B\|$ – which is required in matrix norms)

The condition number of a matrix (2)

Assume that we are using one of the $\|\cdot\|_p$ norms to define the condition number. Then

$$\kappa(c\,I) = 1 \qquad \forall c \neq 0$$

and for every matrix

$$\kappa(A) \geq 1.$$

The sprectral condition number. When $p = 2$

$$\kappa(A) = \frac{\sigma_1}{\sigma_m},$$

where $\sigma_1 \geq \ldots \geq \sigma_m > 0$ are the singular values of A. (This is a good moment to recall a certain hyperellipsoid.) Note that if Q is unitary, then $\kappa(Q) = 1$. Why?

Three tests

We are going to focus in the spectral condition number.

1. We want to measure the sensitivity of
$$x \mapsto Ax$$
to small perturbances in x.
2. We will then measure the sensitivity of the system
$$Ax = b$$
to small perturbances in b.
3. Finally we will look at the first problem again when A is what's perturbed.

A **large condition** number of A means a large dispersion of the singular values of A.

First test

We use the full=reduced SVD of $A = U\Sigma V^*$. Let:

$$\begin{aligned} x &= v_m & \|x\|_2 &= 1, \\ Ax &= \sigma_m u_m & \|Ax\|_2 &= \sigma_m, \\ \delta x &= \varepsilon v_1 & \|\delta x\|_2 &= \varepsilon, \\ A(x+\delta x) &= \varepsilon\sigma_1 u_1 + \sigma_m u_m, & & \\ A(x+\delta x) - Ax &= \varepsilon\sigma_1 u_1 & \|A(x+\delta x) - Ax\|_2 &= \varepsilon\sigma_1 \end{aligned}$$

We then compute the relative **error** with respect to the relative error of data:

$$\frac{\|A(x+\delta x) - Ax\|_2}{\|Ax\|_2} \frac{1}{\frac{\|\delta x\|_2}{\|x\|_2}} = \kappa(A).$$

A theorem

In general, it's not difficult to show that

$$\sup_{\delta x \neq 0} \frac{\|A(x+\delta x) - Ax\|}{\|Ax\|} \frac{\|x\|}{\|\delta x\|} = \|A\| \frac{\|x\|}{\|Ax\|} \leq \kappa(A).$$

The quantity in red is called the condition number of the operation $x \mapsto Ax$ with respect to perturbations in x.

The example of the previous slide shows how for a general invertible matrix and the 2-norm, there are vectors such that the condition number of the operation $x \mapsto Ax$ is $\kappa(A)$.

Is this bad? In a way. The result says that if your condition number is very large, no matter how small your data perturbation is you could get a very large relative error in your computation.

A second test

The problem of solving $Ax = b$ is equivalent to the operator $b \mapsto A^{-1}b$. Therefore, we just apply the conclusions of the previous slide to A^{-1}:

$$\begin{aligned}\sup_{\delta b \neq 0} \frac{\|A^{-1}(b+\delta b) - A^{-1}b\|}{\|A^{-1}b\|} \frac{\|b\|}{\|\delta b\|} &= \|A^{-1}\| \frac{\|b\|}{\|A^{-1}b\|} \\ &\leq \kappa(A^{-1}) = \kappa(A).\end{aligned}$$

We are also able to find examples in the 2-norm where the upper estimate holds as an equality.

Third test: dependence w.r.t. the matrix

Try and figure out what happened to A when adding δA:

$$
\begin{aligned}
A &= U\Sigma V^* & \|A\|_2 &= \sigma_1, \\
\delta A &= U(\varepsilon e_m e_m^*)V^* & \|\delta A\|_2 &= \varepsilon, \\
x &= v_m, & & \\
Ax &= \sigma_m u_m & \|Ax\|_2 &= \sigma_m, \\
(A+\delta A)x &= (\sigma_m + \varepsilon)u_m & & \\
(A+\delta A)x - Ax &= \varepsilon u_m & \|(A+\delta A)x - Ax\|_2 &= \varepsilon
\end{aligned}
$$

This is how the computation of Ax is affected:

$$
\frac{\|(A+\delta A)x - Ax\|_2}{\|Ax\|_2} \frac{1}{\frac{\|\delta A\|_2}{\|A\|_2}} \le \kappa(A)
$$

The corresponding theorem

Just with formulas:

$$\sup_{\delta A \neq 0} \frac{\|(A + \delta A)x - Ax\|}{\|Ax\|} \frac{\|A\|}{\|\delta A\|} \leq \kappa(A)$$

The same bound applies to the sensitivity of solving $Ax = b$ with respect to perturbations of A. In this case the perturbations have to be small enough so that $A + \delta A$ is still invertible.

Small interruption with an argument

If $Ax = \lambda x$, with $x \neq 0$, then

$$|\lambda| = \frac{\|Ax\|}{\|x\|} \leq \|A\|.$$

Therefore if $\|A\| < 1$ (in any given norm), then $I \pm A$ are invertible (because $\lambda = \mp 1$ cannot be eigenvalues of A). Finally, if

$$\|\delta A\| < \frac{1}{\|A^{-1}\|}$$

then

$$\|A^{-1}\delta A\| < 1$$

and

$$A + \delta A = A(I + A^{-1}\delta A) \text{ is invertible}$$

Introducing the Hilbert matrix

The following matrix

$$H_n = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{bmatrix}$$

$$H_{ij} = \frac{1}{i+j-1}$$

is known to have exceptionally bad condition number.

```
>> cond(hilb(5))
ans =
    4.766072502433796e+005
>> cond(hilb(6))
ans =
    1.495105864148109e+007
>> cond(hilb(7))
ans =
    4.753673562966472e+008
>> cond(hilb(8))
ans =
    1.525757555001589e+010
>> cond(hilb(9))
ans =
    4.931541097528780e+011
>> cond(hilb(10))
ans =
    1.602492277132444e+013
```

An experiment with Hilbert's matrix

```
>> n=10;
>> A=hilb(n);u=ones(n,1);
>> b=A*u; v=A\b; norm(u-v)
ans =
    7.377072953848606e-004
>> n=20;
>> A=hilb(n);u=ones(n,1);
>> b=A*u; v=A\b; norm(u-v)
Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 1.155429e-019.
ans =
    2.011761385588561e+002
```

A MORE GENERAL DEFINITION

Relative condition number

Given a function $x \mapsto f(x)$ and $\varepsilon > 0$, we consider all the relative error of all posible perturbations os size ε or less:

$$\begin{aligned} e(\varepsilon) &:= \sup_{|\delta x| \leq \varepsilon} \frac{\|f(x+\delta x) - f(x)\|}{\|f(x)\|} \frac{\|x\|}{\|\delta x\|} \\ &= \frac{\|x\|}{\|f(x)\|} \sup_{|\delta x| \leq \varepsilon} \frac{\|f(x+\delta x) - f(x)\|}{\|\delta x\|} \end{aligned}$$

and then we take the limit as the size of the perturbation is reduced:

$$\kappa(x) = \lim_{\varepsilon \to 0} e(\varepsilon).$$

Example

The roots of the polynomial

$$(x-2)^2 = x^2 - 4x + 4$$

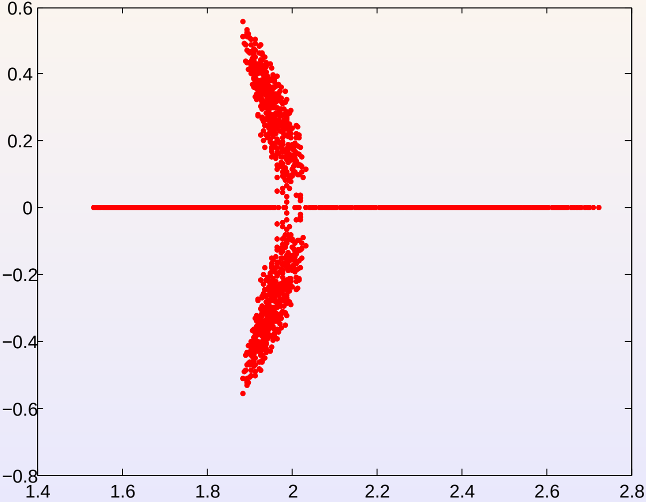
are compared with a polynomial with a small perturbation:

$$(1+\delta a)x^2 + (-4+\delta b)x + (4+\delta c),$$

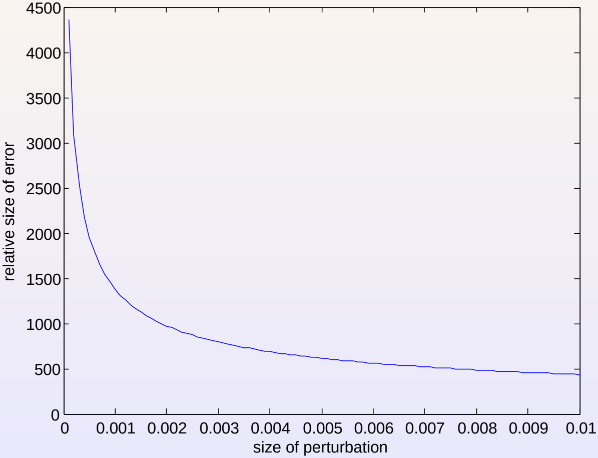
where $\delta a, \delta b, \delta c$ are chosen randomly in $\varepsilon[-\frac{1}{2}, \frac{1}{2}]$. We first show where the roots are when $\varepsilon = 0.1$ and then we look at the function

$$\varepsilon \mapsto \max\left\{ \frac{|r_1(\delta a, \delta b, \delta c) - 2|}{\max\{|\delta a|, |\delta b|, |\delta c|\}} \;:\; (\delta a, \delta b, \delta c) \right\}.$$

These are the roots



... and this is really bad news



FLOATING POINT

An approximate floating point model

Even if numbers are typically stored in a different way, we are going to assume that **floating point** numbers are the following: 0 is a number, and all other numbers are

$$\pm 0.x_1 x_2 \dots x_{16} \times 10^e, \qquad x_1 \neq 0,$$

where the exponent varies between a minimum and a maximum, that we will impose to be

$$-323 \leq e \leq 309.$$

Warning

Once again, numbers are not typically stored like this, but in base 2 decompositions, so the the digits x_j would be zeros and ones, there would be more of them and the exponential part would be 2^e. This model is quite close to what is called *double precision*.

An approximate floating point model (2)

$$\pm 0.x_1 x_2 \ldots x_{16} \times 10^e, \qquad x_1 \neq 0, \qquad -322 \leq e \leq 309$$

- The sign is separated from the number. There are as many positive numbers as there are negative numbers.
- When a real number is too close to zero ($e < -322$), the model goes to *underflow*. In Matlab, the underflow is identified with zero.
- When a real number is too large ($e > 309$), the model goes to *overflow*. Matlab associated overflow with the number `Inf`.
- The smallest number and largest numbers (in absolute value) are

$$0.1\underbrace{0\ldots\ldots 0}_{\text{15 zeros}}\times 10^{-322} = 10^{-323} \qquad 0.\underbrace{9\ldots\ldots 9}_{\text{16 nines}}\times 10^{309}.$$

An approximate floating point model (2)

$$\pm 0.x_1 x_2 \dots x_{16} \times 10^e, \qquad x_1 \neq 0, \qquad -322 \leq e \leq 309$$

The following number to the right of

$$1 = 0.10\dots0 \times 10^1$$

is

$$0.1\underbrace{0\dots0}_{14}1 \times 10^1 = 1 + 10^{-15}.$$

Therefore, the distance between 1 and all the numbers to its right before we reach 10 is 10^{-15}. These numbers are equally spaced. Once we get to

$$10 = 0.10\dots0 \times 10^2$$

the next number is 10^{-14} to the right.

For you to think...

1. How many positive numbers can we store with this model?
2. In this model all integers up to 10^{16} can be stored exactly. What happens after that?
3. What is the closest number to $1/3$ that is stored in the model?
4. Repeat the previous question with $1/6$.
5. If $10^m \leq x < 10^{m+1}$, what is the distance between x and the closest number in the model?

Some Matlab experiments

Remember that we are actually working in base 2, so we are not going to observe the model we have discussed, but something similar.

```
>> 1+1e-16
ans =
     1
>> 1+1e-15
ans =
   1.000000000000001
>> 1+5.551116e-16   % the answer is inbetween
ans =
   1.000000000000001
>> 9999999999999999   % I typed 16 nines
ans =
    1.000000000000000e+016
```

Some Matlab experiments (2)

Remember that we are actually working in base 2, so we are not going to observe the model we have discussed, but something similar.

```
>> 1/6
ans =
   0.166666666666667
>> 1/6-0.166666666666666   % I changed the last digit
ans =
    6.661338147750939e-016
>> eps      % this is what matlab
            % recognizes as machine epsilon
ans =
    2.220446049250313e-016
```

A working model for theory

The issues of magnitude (upper and lower limits for exponents) are typically ignored when we deal with stability. Real numbers x will be assigned a floating point representation $\mathrm{fl}(x) \in \mathbf{F}$. The four arithmetic operators $\{+, -, \times, /\}$ will have floating point correspondents $\{\oplus, \ominus, \otimes, \oslash\}$, which act on floating point numbers. There's a small number called the **machine epsilon** $\epsilon_{\text{machine}}$. We admit the following axioms:

- For all $x \in \mathbb{R}$,

$$\mathrm{fl}(x) = x(1 + \epsilon), \qquad |\epsilon| \le \epsilon_{\text{machine}}.$$

- For all $x, y \in \mathbf{F}$ and $* \in \{+, -, \times, /\}$,

$$x \circledast y = (x * y)(1 + \epsilon) \qquad |\epsilon| \le \epsilon_{\text{machine}}.$$

BACKWARD STABILITY

Landau's big O symbol

For notational purposes, we will write things like

$$\varphi(t) = O(\psi(t)) \qquad \text{as } t \to 0, \text{ or } t \to \infty, \ldots$$

meaning that there exists C such that

$$|\varphi(t)| \le C\psi(t) \qquad \text{as } t \to 0, \text{ or } t \to \infty, \ldots$$

When there are other parameters and C does not depend on them, we will say the $\varphi = O(\psi)$ **uniformly in** these other parameters.

Even if $\epsilon_{\text{machine}}$ is fixed, we will admit expressions like

$$\text{something we have computed} = O(\epsilon_{\text{machine}})$$

assuming that $\epsilon_{\text{machine}}$ is allowed to go to zero.

Terminology

A problem is a function

$$f : X \to Y \qquad X, Y \text{ are vector spaces (copies of } \mathbb{R}^n\text{)}$$

and an algorithm is an actual approximation of the function

$$\widetilde{f} : X \to Y.$$

An algorithm is **backward stable** if for every x, there exists $\widetilde{x}$ such that

$$\frac{\|x - \widetilde{x}\|}{\|x\|} = O(\epsilon_{\text{machine}}) \qquad \text{and} \qquad f(\widetilde{x}) = \widetilde{f}(x).$$

A simple example

Multiplication of floating point numbers:

$$f(x_1, x_2) = x_1 \times x_2, \qquad \widetilde{f}(x_1, x_2) = \mathrm{fl}(x_1) \otimes \mathrm{fl}(x_2).$$

The spaces are $X = \mathbb{R}^2$ and $Y = \mathbb{R}$. Recall the axioms:

$$\begin{aligned} \mathrm{fl}(x_j) &= x_j(1+\epsilon_j), & |\epsilon_j| &\leq \epsilon_{\text{machine}} \\ x \otimes y &= (x \times y)(1+\epsilon_3), & |\epsilon_3| &\leq \epsilon_{\text{machine}}. \end{aligned}$$

Then

$$\begin{aligned} \widetilde{f}(x_1, x_2) &= \mathrm{fl}(x_1) \otimes \mathrm{fl}(x_2) \\ &= \underbrace{x_1(1+\epsilon_1)}_{\widetilde{x}_1} \underbrace{x_2(1+\epsilon_2)(1+\epsilon_3)}_{\widetilde{x}_2} \\ &= f(\widetilde{x}_1, \widetilde{x}_2). \end{aligned}$$

A simple example (cont'd)

We have shown that

$$\widetilde{f}(x_1, x_2) = f(\widetilde{x}_1, \widetilde{x}_2)$$

with

$$\widetilde{x}_1 = x_1(1 + \epsilon_1), \qquad \widetilde{x}_2 = x_2(1 + \epsilon_2)(1 + \epsilon_3)$$

satisfying

$$\frac{|\widetilde{x}_1 - x_1|}{|x_1|} = |\epsilon_1| \leq \epsilon_{\text{machine}} = O(\epsilon_{\text{machine}})$$

$$\frac{|\widetilde{x}_2 - x_2|}{|x_2|} = |\epsilon_2 + \epsilon_3 + \epsilon_2\epsilon_3| \leq 2\epsilon_{\text{machine}} + \epsilon_{\text{machine}}^2 = O(\epsilon_{\text{machine}}).$$

So, there we go! Floating point multiplication is backward stable.

Stable vs backward stable

$f : X \to Y$ is the problem, $\widetilde{f} : X \to Y$ is the algorithm.

An algorithm is **backward stable** if for every x, there exists $\widetilde{x}$ such that

$$\frac{\|x - \widetilde{x}\|}{\|x\|} = O(\epsilon_{\text{machine}}) \qquad \text{and} \qquad f(\widetilde{x}) = \widetilde{f}(x).$$

An algorithm is **stable** if for every x, there exists $\widetilde{x}$ such that

$$\frac{\|x - \widetilde{x}\|}{\|x\|} = O(\epsilon_{\text{machine}}) \qquad \text{and} \qquad \frac{\|f(\widetilde{x}) - \widetilde{f}(x)\|}{\|f(\widetilde{x})\|} = O(\epsilon_{\text{machine}}).$$

Clearly, backward stable implies stable. The reverse statement is false.

An example

The problem and the algorithm:

$$f(x) = x + 1, \qquad \widetilde{f}(x) = \mathrm{fl}(x) \oplus 1$$

Looking for backward stability[1]: we compute

$$\begin{aligned} \widetilde{f}(x) &= (x(1+\epsilon_1)+1)(1+\epsilon_2) \\ &= \underbrace{x(1+\epsilon_1)(1+\epsilon_2)+\epsilon_2}_{\widetilde{x}} + 1 = f(\widetilde{x}) \end{aligned}$$

but there's no backwards stability because

$$\frac{|x-\widetilde{x}|}{|x|} = O(\epsilon_{\text{machine}}) + \frac{O(\epsilon_{\text{machine}})}{|x|}.$$

Funny fact! Adding two numbers is backward stable. Adding 1 to another number is not.

[1] every ϵ is assumed to satisfy $|\epsilon| \le \epsilon_{\text{machine}}$

An example (cont'd)

On the other hand

$$
\begin{aligned}
\widetilde{f}(x) &= (x(1+\epsilon_1)+1)(1+\epsilon_2) \\
&= \underbrace{x(1+\epsilon_1)}_{\widetilde{x}} + 1 + \big(\underbrace{x(1+\epsilon_1)}_{\widetilde{x}} + 1\big)\epsilon_2 \\
&= \underbrace{\widetilde{x}+1}_{f(\widetilde{x})} + \big(\underbrace{\widetilde{x}+1}_{f(\widetilde{x})}\big)\epsilon_2 = f(\widetilde{x}) + f(\widetilde{x})\epsilon_2,
\end{aligned}
$$

which gives

$$
\frac{|x-\widetilde{x}|}{|x|} = O(\epsilon_{\text{machine}}) \qquad \frac{|\widetilde{f}(x)-f(\widetilde{x})|}{|f(\widetilde{x})|} = O(\epsilon_{\text{machine}}),
$$

which means this algorithm is stable.

STABILITY AND CONDITIONING

Review

An algorithm $\tilde{f} : X \to Y$ to solve a problem $f : X \to Y$ is backward stable when for all $x \in X$, there exists $\tilde{x} \in X$ such that

$$\tilde{f}(x) = f(\tilde{x}) \qquad \text{and} \qquad \frac{\|x - \tilde{x}\|}{\|x\|} = O(\epsilon_{\text{machine}}).$$

The condition number of the computation $f(x)$ is the limit

$$\kappa(x) = \lim_{\delta \to 0} \left(\sup_{\|\delta x\| \leq \delta} \frac{\|f(x + \delta x) - f(x)\|}{\|f(x)\|} \frac{\|x\|}{\|\delta x\|} \right).$$

For a given x and δx, we can formally write

$$\frac{\|f(x + \delta x) - f(x)\|}{\|f(x)\|} \leq (\kappa(x) + o(1)) \frac{\|\delta x\|}{\|x\|}$$

where $o(1)$ means that there's a quantity converging to zero as $\delta \to 0$.

A simple argument

Let $\widetilde{f}$ be a backward stable algorithm for f. Given x, we can find $\widetilde{x}$ such that

$$\widetilde{f}(x) = f(\widetilde{x}) \qquad \text{and} \qquad \frac{\|x - \widetilde{x}\|}{\|x\|} = O(\epsilon_{\text{machine}}).$$

Then the relative error of the computation satisfies

$$\begin{aligned} \frac{\|\widetilde{f}(x) - f(x)\|}{\|f(x)\|} &= \frac{\|f(\widetilde{x}) - f(x)\|}{\|f(x)\|} \\ &\leq (\kappa(x) + o(1))\frac{\|\widetilde{x} - x\|}{\|x\|} \\ &= O(\kappa(x)\epsilon_{\text{machine}}). \end{aligned}$$