CPSC 303 HOMEWORK 2 SOLUTIONS, SPRING 2020

JOEL FRIEDMAN

(1) Let

$$A(n) = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n$$

(a) Consider the following MATLAB code for computing some values of $A(n)$:

```
A = [ 1 , 1 ; 1 , 0 ]
for n=1:8, n, A^n , end
```

Write down an exact formula for A^n based on what you see.

(b) What does the following MATLAB code do?

```
A = [ 1 , 1 ; 1 , 0 ]
for n=1:8, n, A=A^2 , end
```

(c) Show that the equation $x_{n+2} = x_{n+1} + x_n$ is equivalent to the equation

$$\begin{bmatrix} x_{n+2} \\ x_{n+1} \end{bmatrix} = A \begin{bmatrix} x_{n+1} \\ x_n \end{bmatrix}$$

(d) Describe the exact value of

$$A^n \mathbf{v}, \quad \text{where} \quad v = \begin{bmatrix} \left(1 - \sqrt{5}\right)/2 \\ 1 \end{bmatrix}$$

(e) Describe the MATLAB computation of the sequence $A^n v$ with the following code:

```
A = [ 1 , 1 ; 1 , 0 ]
v = [ (1-sqrt(5))/2 ; 1 ]
for n=1:80, n, v = A*v  , end
```

How does the output behave for various values of n?

(f) Describe the output of the following MATLAB computation, and explain (roughly) why you see these results.

Research supported in part by an NSERC grant.

```
A = [ 1 , 1 ; 1 , 0 ]
v = [ (1-sqrt(5))/2 ; 1 ]
for n=1:80, v = A*v  ; v_ratio(n) = v(1)/v(2); end
v_ratio
```

Solution:

(a) You should see Fibonacci numbers in these powers and observe the pattern

$$A(n) = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1}, \end{bmatrix}$$

where $F_0 = 0$, $F_1 = F_2 = 1$, etc., are the Fibonacci numbers as described in the handout "Recurrence Relations and Finite Precision."

(b) This code repeatedly squares A, printing out n and A^{2^n} for $n = 1, \ldots, 8$.

(c)

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_{n+1} \\ x_n \end{bmatrix} = \begin{bmatrix} x_{n+1} + x_n \\ x_{n+1} \end{bmatrix}$$

Hence

$$A \begin{bmatrix} x_{n+1} \\ x_n \end{bmatrix} \begin{bmatrix} x_{n+2} \\ x_{n+1} \end{bmatrix} \iff \begin{bmatrix} x_{n+1} + x_n \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} x_{n+2} \\ x_{n+1} \end{bmatrix} \iff x_{n+2} = x_{n+1} + x_n$$

(d) Since $\mathbf{v} = (r, 1)$ where r is the Golden Ratio Conjugate, we have $A^n\mathbf{v} = (r^{n+1}, r^n)$ (see class notes `01_10` or Section 4 of "Recurrence Relations and Finite Precision"). [Alternatively, you could prove this by verify that $A\mathbf{v} = r\mathbf{v}$, and hence A scales $\mathbf{v}$ by r (i.e., $\mathbf{v}, r$ are an eigenvector and eigenvalue pair), and hence $A^n\mathbf{v} = r^n\mathbf{v}$.]

(e) For small n, $\mathbf{v}$ looks like $r^n\mathbf{v}$ (where r is the Golden Ratio Conjugate); for n between roughly 35 and 45 the ratio of v_1/v_2 changes from r to the Golden Ratio, where it stays for larger n (i.e., $n \geq 45$).

(f) Roughly speaking (both in the part and the previous one), what you are seeing are consecutive solutions to the Fibonacci recurrence,

$$x_n = C_1 \left(\frac{1+\sqrt{5}}{2} \right)^n + C_2 \left(\frac{1-\sqrt{5}}{2} \right)^n \tag{1}$$

where $C_2 = 1$, but instead of the $C_1 = 0$ you are seeing C_1 a very small, non-zero quantity.

This code prints out the first component of the numerical value of $\mathbf{v}_n = A^n\mathbf{v}$, numerically computed via the recurrence $\mathbf{v}_n = A\mathbf{v}_{n-1}$. By examining this ratio it is easier to see when the numerical solution transitions from the C_2 term in (1) is dominant to when the C_1 term is.

To the five digits of precision that MATLAB reports by default, the change from -0.6180 to 1.6180, i.e., from the Golden Ratio Conjugate to the Golden Ratio, occurs from $n = 28$ to $n = 52$ [and the most dramatic changes are more between $n = 31$ or $n = 36$ to $n = 42$ or $n = 46$]. Note: there is no single correct answer, because **we have not precisely defined** "the range of transition."

(2) Set $x_0 = 1$, $x_1 = 1/8$, and set

$$\mathbf{v}_n = \begin{bmatrix} 9/8 & -1/8 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} x_1 \\ x_0 \end{bmatrix}$$

(a) What is the value of $\mathbf{v}_n$ in exact arithmetic?
(b) What happens with the MATLAB floating point computation of $\mathbf{v}_n$ below?

```
A = [ 9/8 , -1/8 ; 1 , 0 ]
v = [ 1/8 ; 1 ]
for n=1:400, n, v = A*v ,  end
```

Solution:

(a) Reasoning as with the Fibonacci recurrence (Problem 1(c) above), we have that the 2×2 matrix equation is equivalent to $\mathbf{v}_n = (x_{n+1}, x_n)$, where the sequence x_n, $n \in \mathbb{Z}$ is given by the recurrence

$$x_{n+2} = (9/8)x_{n+1} - (1/8)x_n.$$

The solution to this recurrence is given as

$$x_n = C_1 r_1^n + C_2 r_2^n$$

where r_1, r_2 are the roots to

$$r^2 = (9/8)r - (1/8),$$

i.e., $r = 1, 1/8$. It follows that if $x_0 = 1$ and $x_1 = 1/8$, then the general solution in exact arithmetic is $x_n = (1/8)^n$, and hence

$$\mathbf{v}_n = \begin{bmatrix} x_{n+1} \\ x_n \end{bmatrix} = \begin{bmatrix} (1/8)^{n+1} \\ (1/8)^n. \end{bmatrix}$$

[As in Problem 1, you could reach the same conclusion by noting that $(1/8, 1)$ is an eigenvector of the 2×2 matrix in this problem, with eigenvalue $1/8$.]

(b) Because you are dividing by 8 in double precision, which is done base 2, the calculation has no roundoff/truncation error, and at $n = 1, 2, \ldots, 357$ you get what seems like the exact answer, and surely where the ratio of the first to the second component is $1/8$, down to 9.9e-323 times $(0.0500, 0.4000)$. for $n = 358$ the first component is reported as 0, and for $n \geq 359$, $\mathbf{v}_n$ is reported as $(0, 0)$.

The following are additional remarks not required on the homework:

I am a bit suspicious of the $n = 357$ answer because of the 00's there, and since these reported numbers are very close to the limits of double precision's smallest non-zero numbers. . .; I certainly believe up to $n = 350$, and possibly a few more, since the smallest non-zero positive number in double-precision is 2^{-1074}, or roughly 4.94×10^{-324}.

The real story is a bit more involved: Double precision usually writes numbers in base 2 scientific notation as

$$s \ \times \ 1.b_1b_2\dots b_{52} \times 2^m$$

where $s = \pm$, $b_1,\dots,b_{52} \in \{0,1\}$ are bits (binary digits), and $m = -1022,\dots,1023$; such a number is called a *normal number.* The 52 bits $b_1,\dots,b_{52}$ gives you "53 bits of precision" (at least in the way we speak of 3.45 having "3 digits of precision") for *normal numbers.* There are 2046 possible values of m, from -1022 to 1023, which takes 11 bits (binary digits) and leaves $2^{11} - 2046 = 2$ special values:

(i) one of these two special values (of the 2048 possible values) is for values like Inf, -Inf, and Nan (this special value has a sign, s, plus 52 bits $b_1,\dots,b_{52}$ to describe the particular special value you mean);

(ii) the other special value is for *subnormal numbers*, where when double precision understands that you mean the number

$$\pm\ 0.b_1b_2\dots b_{52} \times 2^{-1022}.$$

In this way you can express numbers as small as

$$0.\underbrace{000\dots000}_{51\text{ 0's}}1 \ \times \ 2^{-1022},$$

as a *subnormal numbers* in double precision, which is $2^{-52} \times 2^{-1022} = 2^{-1074}$; of course, for this number you only have one bit (binary digit) of precision; you can only count on a full 53-bits of precision if your number is 2^{-1022} or larger, and the smaller a *subnormal* number is, the more precision you will lose.

The (current) Wikipedia article on "Double-precision floating-point format" has a good explanation of this with examples; for example, the largest number in double precision is

$$1.\underbrace{1111\dots1111}_{52\text{ 1's}} \times 2^{1023} \approx 2^{1024},$$

and this is *normal* number—the kind you *should* be working with—since you get a full 53 bits of precision, because *normal* numbers are written as 1. followed by 52 more bits (binary digits). The smallest positive normal number is 2^{-1022}.

Since we are working with powers of 1/8, each power of 8 looks like

$$1.\underbrace{000\dots000}_{52\text{ 0's}} \times 2^m$$

for $-1022 \le m \le 1023$, which are normal numbers, but for $m \le -1023$ these powers of 2 (or 8) are the special *subnormal numbers* that look like

$$0.\underbrace{000\dots000}_{\text{some 0's}}1\underbrace{000\dots000}_{\text{more 0's}} \times 2^{-1022}$$

So in binary arithmetic, the smallest positive number is $2^{-1074} = 8^{-358}$, so I'll (probably) trust the numerical computation as exact until the division by 8 in the recurrence dips below 8^{-358}.

Note that textbook [A&G] **does not mention subnormal numbers**. Similarly, if you type `realmin` into MATLAB, it will return `2.2251e-308`, since you can't count on 53 bits of precision for smaller positive numbers, i.e., subnormal numbers (and you should realize this caveat in working with smaller positive numbers). However, MATLAB will report subnormal numbers without telling you this, which explains why you can see numbers as small as $2^{-1074} \approx 4.94 \times 10^{-324}$.

(3) Set $x_0 = 1$, $x_1 = 1/7$, and set

$$\mathbf{v}_n = \begin{bmatrix} 8/7 & -1/7 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} x_1 \\ x_0 \end{bmatrix}$$

(a) What is the value of $\mathbf{v}_n$ in exact arithmetic?
(b) Explain what you see in the MATLAB floating point computation of $\mathbf{v}_n$ below, and (roughly) why you see it.

```
A = [ 8/7 , -1/7 ; 1 , 0 ]
v = [ 1/7 ; 1 ]
for n=1:40, n, v = A*v ,  end
```

Solution:

(a) As before (in the previous problems), this 2×2 system is equivalent to solving the recurrence $x_{n+2} = (8/7)x_{n+1} - (1/7)x_n$. So we solve $r^2 = (8/7)r - (1/7)$ to get $r = 1, 1/7$. It follows that for either value of r, we have

$$\begin{bmatrix} 8/7 & -1/7 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} r \\ 1 \end{bmatrix} = \begin{bmatrix} r^{n+1} \\ r^n \end{bmatrix}$$

(b) The general solution to

$$\begin{bmatrix} 8/7 & -1/7 \\ 1 & 0 \end{bmatrix}^n \begin{bmatrix} x_1 \\ x_0 \end{bmatrix},$$

for general x_0, x_1 is of the form

$$C_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + C_2 \begin{bmatrix} (1/7)^{n+1} \\ (1/7)^n, \end{bmatrix} \tag{2}$$

since the vectors this generates is equivalent to producing (x_{n+1}, x_n) where the sequence x_n satisfies the recurrence $x_{n+2} = (8/7)x_{n+1} - (1/7)x_n$. Although the solution in exact arithmetic is $C_2 = 1$ and $C_1 = 0$ for $(x_1, x_0) = (1/7, 1)$, due to roundoff/truncation the observed solution will be $C_2 = 1$ but C_1 very close to 0 (but not exactly 0), because of the divisions by 7 in the base 2 format of double precision. Specifically, for small values of n we observe only the C_2 term in (2), but for large $n \geq 26$ numerically observe a solution in (2) where only the C_1 term dominates, with C_1 roughly 0.5480e-17. The additional line of code

```
v = [ 1/7 ; 1 ]
for n=1:40, v = A*v ; v_ratio(n) = v(1)/v(2); end
v_ratio
```

makes it easier to identify the transition from the ratio of 1/7 to 1, which takes place from $n = 15$ to $n = 25$ as reported by MATLAB (to five places of precision), with more dramatic changes around $n = 18$ to $n = 21$. The relevant transition values are:

```
Columns 15 through 21

  0.1428    0.1427    0.1418    0.1351    0.0857   -0.5231    1.4159

Columns 22 through 28

  1.0420    1.0058    1.0008    1.0001    1.0000    1.0000    1.0000
```

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA.
E-mail address: jf@cs.ubc.ca
URL: http://www.cs.ubc.ca/~jf