The Cooper Union Department of Electrical Engineering
Prof. Fred L. Fontaine
ECE211 Signal Processing & Systems Analysis
Problem Set VI: State-Space Analysis
April 9, 2019

1. Given:

$$H(z) = \frac{5z^2 + 6z + 7}{10z^2 - 3z - 4}$$

   (a) Sketch the direct form II transposed realization.

   (b) Find the $\{A, B, C, D\}$ state-space representation. Take the convention that the delay element on top is state-variable $x_1[n]$, and the one below is $x_2[n]$. *Note:* Sometimes the path from one input or state variable to an output or state variable goes through more than one multiplier and/or adder; in this case you have to trace out the full path to get the proper coefficients in A, B, C, D.

   (c) In MATLAB, use *ss2tf* (not the symbolic toolbox) to convert your state-space realizations back to a transfer function, and verify that you get back the original $H(z)$.

   (d) Obtain $\{A', B', C', D'\}$ where $A' = A^T$, $B' = C^T$, $C' = B^T$ and $D' = D^T$. Apply $ss2tf$ to check that $H(z)$ has not changed. **Note:** This will turn out to be the direct form II realization, you don't have to draw it.

   (e) Now call $tf2ss$ with $H(z)$. Does MATLAB return either of the two realizations you have so far, or something else?

   (f) Start with the $\{A, B, C, D\}$ realization MATLAB $tf2ss$ returns from $H(z)$. Apply the following transformation of state: $x' = Tx$, which results in $A' = TAT^{-1}$, $B' = TB$, $C' = CT^{-1}$, $D' = D$. Take:

$$T = \begin{bmatrix} 5 & 2 \\ 7 & 3 \end{bmatrix}$$

   Compute $\{A', B', C', D'\}$, and use the symbolic toolbox to compute the transfer function (using the formula for $H(z)$ in terms of the state-space matrices), and verify that it is the same $H(z)$.

2. The following information about an analog state-space realization $\{A, B, C, D\}$ is known: D is a 2×3 matrix, and the eigenvalues of A are: -3 with multiplicity 3, $-2 \pm j5$ (simple) and $+0.5$ (simple).

   (a) Specify the number of inputs, outputs and state variables, and the dimensions of $A, B, C, H(s)$.

   (b) Write the general form of an entry in e^{At} (as a sum of real time-domain functions with several unspecified constants).

   (c) Is it possible for the transfer function matrix $H(s)$ to be stable? If so, what is the condition for this to occur?

3. Given:

$$A = \begin{bmatrix} 139.3 & -263 & -135 \\ 83 & -156.7 & -80.4 \\ -18 & 34 & 17.5 \end{bmatrix}$$

(a) (Make sure you typed in the MATRIX **PRECISELY** as given, in MATLAB; this has been carefully crafted to produce fun results!! You're welcome.) Use MATLAB to check that the eigenvalues are 0.3 with multiplicity 2, and -0.5 (simple).

(b) Use the symbolic toolbox to compute $z\left(zI - A\right)^{-1}$. Then use *iztrans* to compute $\Phi\left[n\right] = A^n$.

(c) Take $N = 40$. With initial condition $\vec{x}\left[0\right] = \left[1\ 1\ 1\right]^T$, compute $\vec{x}\left[n\right]$ for $0 \leq n \leq N$ using the iteration $\vec{x}\left[n+1\right] = Ax\left[n\right]$. (Do NOT directly compute using A^n).

(d) Here, for a vector $\vec{v}$, let $|\vec{v}|$ denote its length. Graph $|\vec{x}\left[n\right]|$ on a stem plot, with the value given in *decibels* (should it be $20\log_{10}$ or $10\log_{10}$?). [If $\vec{x} \to 0$, what do you expect to happen to the length $|x\left[n\right]|$ in *decibels?]*

(e) Look at your graph of $|\vec{x}\left[n\right]|$ *in decibels.* For large n, this seems to converge to a straight line with a negative slope. Why? What would the slope be, and why?

(f) Again with $N = 40$, you can evaluate A^N directly in MATLAB, and you can also evaluate it using your symbolic representation of $\Phi\left[n\right]$; if *phi* is the name you gave to the symbolic state transition matrix, then the following will evaluate it at N:

$$phiN = double(subs(phi,' n', N));$$

Do this, and compare your computed A^N with $\Phi\left[N\right]$ found as above by computing the following: find the maximum absolute error between the elements of the two computed matrices.

4. Here we continue the example from the previous problem. Look at your graph for $|x\left[n\right]|$. You should make the following observation: although $\vec{x} \to 0$, it does not decrease monotonically; that is, it is possible for $|A\vec{v}| \geq |\vec{v}|$ even though its eigenvalues have magnitude less than 1. The operator norm of A (even if A is nonlinear) is defined as:

$$\|A\| = \max_{|\vec{v}| \neq 0} \frac{|A\vec{v}|}{|\vec{v}|}$$

In MATLAB, *norm(A)* returns the operator norm of the matrix A.

(a) Evaluate *norm(A):* you may be surprised how big it is! Note that $|A\vec{v}| = \sqrt{v^T\left(A^TA\right)v}$; it turns out $\|A\|$ is the square root of the largest eigenvalue of A^TA: use MATLAB to check that this is right. In fact, look at the eigenvalues of A^TA: two are tiny but one is HUGE. If $\vec{v}$ has a component in that eigendirection, $A\vec{v}$ will "explode" but if $\vec{v}$ is $\perp$ to that direction, then $A\vec{v}$ will get tiny. This is significant because, even though $A^n \to 0$ in the limit, it can cause large transients, for example triggering overflows. In fact, this can in turn cause the actual system to become UNSTABLE when quantization is taken into account! Instead

of $\vec{x}[n+1] = A\vec{x}[n]$, we have $\vec{x}[n+1] = Q(Ax[n])$ where Q is the (nonlinear) quantization operation. Thus, over time, we apply the operation QA (multiply by A, then quantize) repeatedly, and can write $\vec{x}[n] = (QA)^n \vec{x}[0]$. Although we have a mathematical guarantee that $A^n \to 0$, meaning this potentially explosive nature of A will eventually not be triggered (look at your graph of $|\vec{x}[n]|$), the perturbation caused by Q might prevent $(QA)^n \to 0$. If this happens, it is called a *limit cycle* and is a form of instability!

(b) Let $Q(\vec{v})$ be a saturation overflow function where each component of the vector is replaced by $+127$ if $v_i > 127$, or -128 if $v_i < -128$. With this, run the iteration:

$$\vec{x}_Q[n+1] = Q(A\vec{x}_Q[n])$$

with $\vec{x}_Q[0] = [1\ 1\ 1]^T$. Run this iteration to compute $\vec{x}_Q[n]$ for $0 \leq n \leq 5$. Look at the contents of x_Q (don't graph $|x_Q[n]|$, look at the actual state vector entries) and you will see what happens! You have hit a limit cycle! [This doesn't necessarily mean it oscillates. It means it doesn't collapse to 0. Actually, in this case, it gets "stuck" pretty far from 0! Obviously with an overflow operation, we don't have to worry about anything blowing up. But this certainly violates *asymptotic* stability!]

Remark: Here is an explanation. No work required, this is just (fun?) reading. We observe that our overflow quantization scheme has the property that $|Q\vec{v}| \leq |\vec{v}|$ for all $\vec{v}$, and in fact we see that the operator norm is $\|Q\| = 1$. In fact, any operator F with the property $\|F\| \leq 1$ is called *nonexpansive* or *passive.* If we have two (linear or nonlinear) operators say F, G, it turns out $\|FG\| \leq \|F\| \|G\|$. From this we get $\|(QA)^n\| \leq \|Q\|^n \|A\|^n = \|A\|^n$. Our problem is although all the eigenvalues of A have magnitude less than 1, which forces $A^n \to 0$, we have $\|A\|$ bigger than the largest eigenvalue magnitude. Let's denote this as $\max|\lambda(A)|$. Because the system is internally stable, we know $\max|\lambda(A)| < 1$. However, this does not force $\|A\| < 1$! In fact, $\|A\| > 1$ in our case! Note that $\|A\|^n$ is an upper bound on the norm of $(QA)^n$. Thus, we MAY have $(QA)^n \to 0$ still, if we are lucky, but it may not (in this case, you have seen it does not!) So, in general, we may or may not have a limit cycle– they can be very difficult to predict. HOWEVER there is a trick! If A is a square matrix and T is square invertible, we know $A' = TAT^{-1}$ has the same eigenvalues of A. There is a mathematical theorem that for every A, we can find T such that $\|A'\| = \max|\lambda(A')|$ which means $\|A'\| = \max|\lambda(A)|$. In this case, if we start with A having $\max|\lambda(A)| < 1$, i.e., $A^n \to 0$, we would get $(QA')^n \to 0$, in fact, it will decay exponentially fast! Thus, for any state-space realization $\{A, B, C, D\}$ that is internally stable, we can always find a transformation of state that *prevents* limit cycles due to passive nonlinearities, such as overflow. Last comment: roundoff can make values LARGER (when you round up); in fact, if Q_R is the roundoff operation, $\|Q_R\| = 2$ (e.g., 0.5 rounds up to 1 if we round to nearest integer). For these types of nonlinearities, there is no systematic way to universally prevent limit cycles- the situation needs to be studied on a case by case basis. Although it might seem that overflow is a "worse" form of quantization than roundoff, in that the size of overflow errors tend to be larger than roundoff errors, the fact that roundoff is not passive can suggest it is a more difficult kind of nonlinearity to contend with.