Repeated Linear Transformations

1 Warm-up Exercise

Before starting the lab, which is on Repeated Linear Transformation, this is a warm-up exercise on the general concepts of transformations.

Write a script following the steps below.

- Ask the user to input an mxn matrix A, where m>1 and n>1, an mx1 column vector **b** (the number of rows of the matrix and vector should be equal), and an mx1 column vector **c**.
- Check if **b** is in the range of the transformation T. If not, throw an error.
- If it is, find a vector **x** that is being transformed into **b** by the matrix A and display it. If **x** is not unique, set the free variables to zero.
- Use fprintf to display whether **x** is unique or not unique.
- Determine if **c** is in the range of the transformation T. Use fprintf to display your answer.

2 Introduction

In this lab we will explore and plot the effects of repeated linear transformations on a vector. To motivate this, recall that in the first case study we wish to solve the set of linear equations x=Ax+d, where A is an mxn matrix and d is an mx1 vector. One way to do this for the particular types of matrices A in the case study is to use an iterative method, which takes an initial value for x (called x_0), and then repeats the calculations $x_{k+1}=Ax_k+d$ for k=0,1,2,etc, until x_{k+1} is sufficiently close to x_k.

For purposes of this lab we will assume that d=0. (Although that does not correspond to a thriving economy in the first case study, it still illustrates what happens when an iterative method is used to solve the set of linear equations.) We can then interpret the iterative method as applying the linear transformation $x \rightarrow T(x) = Ax$ repeatedly, starting with the initial vector x_0. In this lab we will assume that the transformation matrix A is 2x2, so the vector x is in R^2. We will repeatedly apply a transformation to an input vector, plotting the result and comparing the norm of the result to the norm of the original vector after each step.

3 Plotting Vectors

One way to plot a vector in MATLAB is to use the `quiver` command. Here is an example where v is a unit vector:

```
>> v = [1 0]';
>> quiver(0,0,v(1),v(2),0), axis equal
```

The `quiver` command takes five arguments. The first two are the x- and y-coordinates of the origin of the vector. Because we want to plot vectors originating from the origin, these arguments will always be zero. The second two arguments are the x- and y-coordinates of the endpoint of the vector. The final arguments is a scaling factor that should always be zero.

4 Plotting a Circle with Radius Equal to the Norm of a Vector

In the function you will write today, you will need to compare the norm of a transformed vector to the norm of the vector before the transformation. While you will make this comparison numerically, it is also helpful to visualize the transformation by plotting a circle whose radius is equal to the norm of the initial input vector. The following code provides an example of such a plot with the vector `v=[1 1]'`.

```
>> v = [1 1]';
>> initNormv = norm(v);
>> t = [-initNormv:.1e-4:initNormv];
>> plot(t,sqrt(initNormv^2-t.^2)),axis equal, hold on,
>> plot(t,-sqrt(initNormv^2-t.^2)),
>> quiver(0,0,v(1),v(2),0)
```

The preceding code plots the vector **v** along with a circle with radius equal to $||v||$.

5 Exercises

(a) Write a function `TransformVector` that accepts three arguments:
A: a 2x2 transformation matrix
v: a two element column vector
n: the number of times the transformation will be applied to the input vector

Your function will apply the transformation matrix A to the input vector **v**, n times. After each step, plot the resulting vector. If the norm of a resulting vector is greater than ten times the norm of the initial input vector, your function should terminate and print a message stating the transformation has diverged after the k^{th} step. If the norm of a resulting vector is less than a tenth of the norm of the initial input vector, the function should terminate and print a message stating the transformation is approaching zero after the k^{th} step. If the function is able to apply the transformation successfully n times, it should say so upon terminating.

No error checking is required. Here are some hints to help you structure your function:

- Your function will not return any variables. Keep this in mind when writing the function declaration.
- Use the code given in Part 3, above, to plot the initial input vector and the circle whose radius is equal to the vector's norm.
- Use a for loop to transform the vector n times and plot at each step. Specifically, in your loop, you will:

```
1. Apply the transformation, placing the result back in the vector v.
2. Plot the result using the quiver command.
3. Pause the display for 0.1 seconds to make the animation  of transformed
   vectors easy to see. Try pause(0.1).
4. Check if ||v|| is greater than 10 times initNormv and if  true, print
   an appropriate message before breaking out of the loop.
5. Repeat for converging transformations.
6. Check if the loop has run n times and print an appropriate statement
   before terminating.
```

(b) Check that your function works by trying the following commands:

```
>> TransformVector([1 -.2;0 1],[1 1]',25)
>> TransformVector([1 -1;0 1],[1 1]',25)
```

Describe the transformation you observe. Hint: see Lay p.76. Compare the results in the two cases.

(c) Give diagonal and one non-diagonal example of A each that cause the norm of the unit vector [1 0] to (i) diverge, (ii) approach zero, (iii) stay the same. (A total of six A matrices are required.)