HW # 05 Solutions: Supplement

1. Why is linear independence important? Why not just check a determinant?

   Answer: Linear independence of column vectors is not limited to square matrices. It also works for rectangular matrices and allows you to test for uniqueness of solutions to $Ax = b$ for rectangular A. That should seem empowering!

   For square systems of equations and hence for square matrices, we can just check the determinant. Yeah, we get it, that is much easier, but it is leaving out what is really going on. Linear independence is at least one-layer deeper into the existence question!

   When A is not square, you can check linear independence of its columns by checking $\det(A^\top A)$. This is our Pro tip!

2. How to do regression computations by hand?

   Answer: Carefully? We know, bad humor again. You hate regression by hand and so do we! We do them all in Julia or Matlab. The Drill Problems are meant to work on the method. Next time, I will make sure that Problem 6 on HW05 Drill Problems says that working it with Julia is fine. Thanks for pointing this out.

3. Checking linear independence in code?

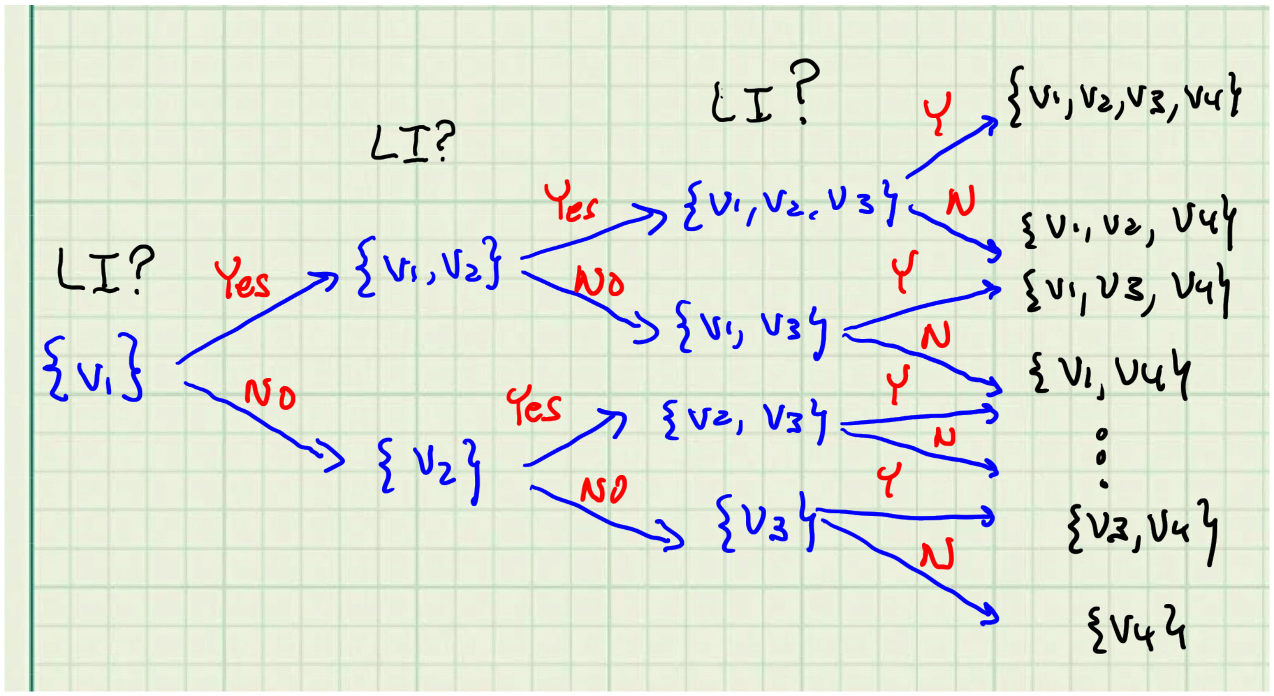

Figure 1: Checking Linear Independence From Left to Right

Answer: first we give an interesting matrix. Then we implement the above.

```
#Special M giving diag U does not tell the whole story
M=[-0.79847     0.742622    1.69606    1.07523    0.0398835;
 -0.919013    0.478715   -1.321       0.56191   -0.871412;
  0.634727   -0.0513059  -2.16552     0.451831   0.125808;
  0.0162196  -0.445659    0.917731  -0.194856  -1.2868;
 -3.25932    -1.79361    -1.08994     0.720373   1.84247;
 -1.27088     0.281362    0.936592  -0.551119   0.438452;
 -1.97054    -0.306892   -1.5218      0.786302   0.543086]
```

```
B=[ 0.466987  0.79299    0.115564;
 0.350669  0.547496   0.45875;
 0.788564  0.0784635  0.0969156]
A=[M[:,2:4]*B M]
```

Here is a very naive solution. It relies on our our unreliable friend, the matrix determinant.

```
# let's find the find the linearly independent columns
# and place them in a matrix called Indep
# For fun, we'll keep track of the columns that we discard
(n,m)=size(A)
Indep=Array{Float64,2}(undef, n, 0)
discardColumns=Vector{Int64}(undef, 0)
epsilon=1e-8
for j=1:m
    v=A[:,j]
    temp=[Indep v]
    test=det(temp'*temp)
    if  !isapprox(test,0, atol=epsilon)
        # Keep the vector because it is independent of the previous vectors
        Indep=[Indep v]
    else
        # reject the vector
        # keep track of the column
        discardColumns=[discardColumns;j]
        println("Remember: the determinant is not a reliable friend")
    end
end

@show discardColumns
Indep

\textbf{}Output}
Remember: the determinant is not a reliable friend
Remember: the determinant is not a reliable friend
Remember: the determinant is not a reliable friend
discardColumns = [5, 6, 7]
7×5 Array{Float64,2}:
  1.78944     1.60184    0.968094  -0.79847     0.0398835
  0.203422   -0.299537  -0.496229  -0.919013   -0.871412
 -0.427042   -1.19085   -0.955572   0.634727    0.125808
 -0.0399536   0.133762   0.350622   0.0162196  -1.2868
 -0.651741   -1.96253   -0.637471  -3.25932     1.84247
  0.0252336   0.692655   0.408765  -1.27088     0.438452
 -0.0569132  -1.01485   -0.657386  -1.97054     0.543086
```

Here is better way, but it is highly inefficient because it does so many LU Factorizations!

```
# let's find the find the linearly independent columns
# and place them in a matrix called Indep
# For fun, we'll keep track of the columns that we discard
(n,m)=size(A)
Indep=Array{Float64,2}(undef, n, 0)
discardColumns=Vector{Int64}(undef, 0)
epsilon=1e-10
```

```
for j=1:m
    v=A[:,j]
    temp=[Indep v]
    F=lu(temp'*temp, check=false)
    test=minimum(abs.(diag(F.U)))
    if  !isapprox(test,0, atol=epsilon)
        # Keep the vector because it is independent of the previous vectors
        Indep=[Indep v]
    else
        # reject the vector
        # keep track of the column
        discardColumns=[discardColumns;j]
        println("Remember: LU is a reliable friend")
    end
end

@show discardColumns
Indep
```

Best Method: See our work with LU in the book!

4. How to keep track of the conditions for checking linear independence?

   Answer: Make sure you include this in your cheat sheet for HW07.

5. Concept of using linear combinations to solve $Ax = b$ is confusing.

   Answer: When considering $Ax = b$, lets' first make sure we understand what is Ax?

   We write

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix} \text{ and } \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}. \tag{1}$$

   Then, using our column times row form for multiplication, we have

$$Ax = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} x_1 + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{bmatrix} x_2 + \cdots + \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix} x_m \tag{2}$$

   If the above is not clicking for you, then reach out to us, because without the above formula, Chapters 7 and 9 are off limits!

   The next step is to move the x_i in front of the column vectors of A. That should be straightforward.

$$Ax = x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{bmatrix} + \cdots + x_m \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix}.$$

   Consider again $Ax = b$, which we write as

$$Ax = b \iff x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{bmatrix} + \cdots + x_m \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

We see from the above that for a solution to exist, b must a linear combination of the columns of A.

6. Why are we using $P \cdot A = L \cdot U$? Why is $A = P^\top \cdot L \cdot U$?

   Answer: Some matrices do not have an LU Factorization without permutations. We need you to attempt to compute an LU Factorization for
$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$
   without using permutations. If you do not do it yourself, then it just will not click.

   A beautiful property of permutation matrices is that $P^\top \cdot P = P \cdot P^\top = I$. Hence,
$$P \cdot A = L \cdot U \implies A = P^\top \cdot L \cdot U.$$

7. Proof of the Pro Tip was hard to follow.

   Answer: The key steps were summarized here.

Don't miss seeing the forest for the trees!

Don't let the last details of the proof distract you too much. The main steps of the Pro Tip are

- [rectangular system of equations] $A\alpha = 0 \iff A^\top \cdot A\alpha = 0$ [square system of equations].
- The square system of equations $A^\top \cdot A\alpha = 0$ has a unique solution of $\alpha = 0$, the 0-vector in $\mathbb{R}^m$, if, and only if, $\det(A^\top \cdot A) \neq 0$.
- Hence, $A\alpha = 0$ has a unique solution of $\alpha = 0$, the 0-vector in $\mathbb{R}^m$, if, and only if, $\det(A^\top \cdot A) \neq 0$.
- Our final results is, $A\alpha = 0$ has a unique solution of $\alpha = 0$, the 0-vector in $\mathbb{R}^m$, if, and only if, the columns of A are linearly independent, where the last implication uses the **definition of linear independence**.

8. When is the triangle inequality useful?

   Answer: The triangle inequality is useful for having a bound on the norm of a sum of two vectors. You know, in ROB 101, we may not use it, so maybe it should not have been stated!

9. Is **min** or **arg min** a real math notation or just Julia syntax?

   Answer: They are math concepts that have existed for a long time, but over the past decade, have taken over graduate-level engineering! Why? Because we can now compute a minimum value of many types of functions in real-time on a robot! Imagine you want to guide a robot from point A to point B. Doing it so that you consume the least amount of battery energy makes a lot of sense! Hence, finding the minimum of energy consumed over all motor commands that take a robot from A to B is super important. In some sense, we care more about the motor commands than the actual amount of energy (as long as it is less than what we have in our battery). The **arg min** function would return the optimal motor commands in this case.

10. Hard to grasp b a linear combination of columns of A?

    Answer: Here we assume you have done the check that $Ax = b$ has at least one solution. If the solution if unique, we know A must be squared the solution can be found from solving $Ly = b$ and $Ux = y$. However, if the solution is not unique, we can still do the LU factorization of $P \cdot A = L \cdot U$, which gives $L \cdot Ux = Pb$. Given that the shape of A is $n \times m$,

    - if $n > m$, then L will be $n \times m$ and U will be $m \times m$. The equation $Ly = b$ has a unique solution. To solve $Ux = y$, you will have a unique solution all diagonal terms are zero, if not, you can assign arbitrary values to variables that correspond to zero diagonal element.
    - if $n < m$, then L will be $n \times n$ and U will be $n \times m$. The solution to equation $Ly = b$ is unique. To solve $Ux = y$, if the diagonal elements of U are non-zero, you can arbitrarily assign values to last $m - n$ variables. If there are zero diagonal elements, you assign arbitrary values to these variables that correspond to the diagonal elements.

11. Why can two zeros on diagonal of U pose an issue?

    Answer: Here is an example of a matrix A where the number of non-zero elements on the diagonal of U does not predict the number of linearly independent columns of A. Consider

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

    A is symmetric, meaning that $A^\top = A$. Moreover, you can check that $A^\top \cdot A = A$. The following is a valid LU Factorization with row permutations

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{P} \cdot \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A^\top \cdot A} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{L} \cdot \underbrace{\begin{bmatrix} \boxed{0} & 1 & 0 \\ 0 & \boxed{0} & 0 \\ 0 & 0 & \boxed{0} \end{bmatrix}}_{U}.$$

    We see that U has one linearly independent column (the same as A), while $\mathrm{diag}(U)$ has three zeros. Hence, the number of non-zero elements on the diagonal of U does not always correspond to the number of linearly independent columns of A and U. If you take more theoretical linear algebra course, you will learn that the eigenvalue zero of U has algebraic multiplicity three and geometric multiplicity two. To be clear, graduate students struggle with the concept of geometric multiplicity of eigenvalues and eigenvectors. If you want to learn more about this, look up Jordan Canonical Forms.

12. What is the point of a vector space?

    Answer: One way to look at is the following: first you learned about whole numbers. You learned a lot about adding and multiplying them. Then you learned about division, and for your set of numbers to be closed under the operation of division, you had to introduce the rational numbers. The rational numbers satisfied all the properties you had for whole numbers, plus division was defined. Then, you made a giant leap to the real numbers, which include things like π and $\sqrt{2}$. But when you made that giant leap, it wasn't too bad because the real numbers satisfied the same rules for addition, multiplication, and division as the rational numbers!

    Our analogy would be that we started working with vectors and matrices. Then we realized that solving $Ax = b$ involves linear combinations and to be closed under linear combinations, we needed the idea of a subspace, but if you are a **sub**-space, you must be a subset of some parent set, and for us, that was $\mathbb{R}^n$. From vectors, to linear equations, to linear combinations of vectors, we landed in the domain of vector spaces.

    You will get really comfortable after awhile working with $\mathbb{R}^n$. And then you will notice or an instructor will point out that all the rules for manipulating vectors that you know also apply to $n \times m$ matrices! And you will discover, to your consternation, that set of $n \times m$ matrices forms a vector space. And then comes graduate school. By then, you've worked with functions a lot more than you have now. And you will notice or an instructor will point out that all the rules for manipulating vectors that you know also apply to functions $f : \mathbb{R}^m \to \mathbb{R}^n$. We know how to add two functions, we know how to multiply a function by a scalar, we know to apply the rules of linear algebra to functions! And you will discover, to your consternation, that set of functions $f : \mathbb{R}^m \to \mathbb{R}^n$ forms a vector space, and this vector space will be amazing because it is infinite dimensional. And you will think that makes it super hard to understand and do anything practical with it, and you will be wrong. It does take a lot more math to do things carefully and correctly, but it is not out of reach of engineers. You can start yourself down this path by targeting MATH 451 before you graduate. Myself (JWG), I took a similar course at UT Austin the summer between undergrad and grad school. It opened my mind to so many things!

    Summarizing: vectors are to whole numbers what vectors spaces are to rational numbers: you are closing the set under a given operation: for vector spaces, you are closed under linear combinations, while for the rational numbers, you are closed under division by non-zero whole numbers. Understanding $\mathbb{R}^n$ is the first step along a very interesting road!

Remaining questions/remarks to be answered

Hard to follow proofs of why least squares works.

Still problems with permutation matrices