Linear Algebra Demo Addendum: Breaking Down The Vectorized Code

By Jaisohn Kim (jais0hn@vt.edu)

In the 05b video, the following statements were used to numerically obtain the weight of the box such that T_1 didn't exceed $T_{max} = 200\ N$:

```
T1_max = max(T1_vec(T1_vec <= T_max))
W_max = max(W_vec(T1_vec <= T_max))
```

The best way to read this is starting from the inside and working your way out. The two statements are nearly identical except the second statement uses `W_vec` instead of `T1_vec` right after the "max" command, so I'll focus on explaining the first line.

If we're starting on the inside and working our way out, the first statement to inspect is `(T1_vec <= T_max)`. Copy/paste this statement into the Command Window. As seen in Figure 1, you should get a 1-by-26 logical array with 1's occupying the first 15 slots and 0's occupying the rest. You can assign this statement to a dummy variable to make it appear in the Workspace.

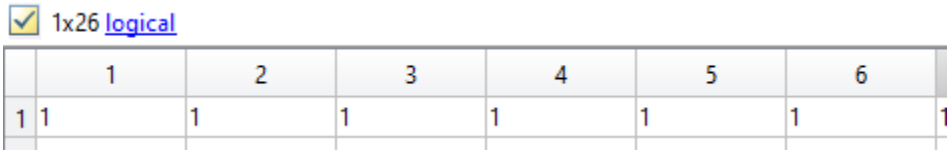

Figure 1: Result from the statement `(T1_vec <= T_max)`

This means that MATLAB compared every element in each of the two vectors and outputted a 1 if `T1_vec <= T_max` and 0 otherwise. This logical array has 26 entries because there are 26 entries each in `T1_vec` and `T_max`. (I highly recommend keeping track of variables' dimensions; it can be incredibly useful when deciphering code.)

Next, copy/paste the statement `T1_vec(T1_vec <= T_max)` into the Command Window. You will get another logical array, but this time it's 1-by-15. This statement tells MATLAB to only retrieve the elements of `T1_vec` where `(T1_vec <= T_max)`. In other words, go through the entire `T1_vec` and output its value if the corresponding output of `(T1_vec <= T_max)` is 1. Do nothing if the corresponding output of `(T1_vec <= T_max)` is 0. The results are printed in Figure 2.

1x15 double

	1	2	3	4	5	6
1	0	14.1421	28.2843	42.4264	56.5685	70.7107

Figure 2: Results from the statement `T1_vec(T1_vec <= T_max)`

When this second statement is executed, MATLAB goes element-by-element through `(T1_vec <= T_max)` and `T1_vec` simultaneously. It outputs the value of `T1_vec` at that index if there is a 1 in `(T1_vec <= T_max)` at the same index. Only the first 15 elements of `(T1_vec <= T_max)` contain a 1, so `T1_vec((T1_vec <= T_max))` contains the first 19 elements of `T1_vec` and ignores the rest. This is essentially a fancy way of saying `T1_vec(1:15)`. Note that this is a *double array* instead of a *logical array*. This indicates we've returned actual numerical values instead of just 1's and 0's.

The outermost statement, `max(T1_vec((T1_vec <= T_max)))`, returns the maximum value of the 1-by-15 double array (Figure 2). The maximum value just so happens to be the last (15^{th}) element of the vector (this should make sense if you understand linear systems). This returns a single number, which represents the maximum `T1` that doesn't exceed `T_max`.

```
T1_max =

   197.9899
```

We know what the maximum force without exceeding the threshold is, but at what weight of the box does that occur? That's where the statement `W_max = max(W_vec((T1_vec <= T_max)))` comes in. The same logic applies, but we are instead looking through the `W_vec` vector to find what the maximum weight is and where it occurs.

```
W_max =

    140
```

Finally, you can extend this principle to T_2 as well. What if we wanted to find what value T_1 holds when T_1 is maximized? Enter this statement into the Command Window:

```
T2_max = max(T2_vec(T1_vec <= T_max))
```

The output is:

```
T2_max =

    140
```

`T2_max` and `W_max` have the same value. This coincidence is due to the properties of the statics problem. See if you can figure out why this happens!