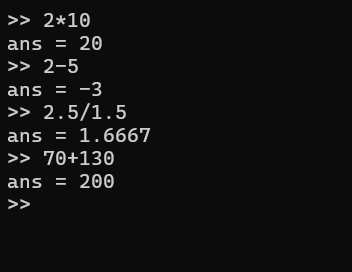
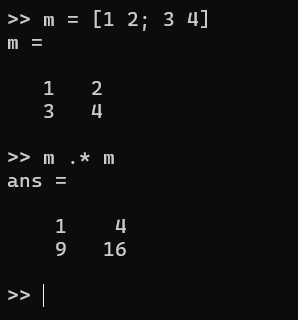
Using DrapsTV playlist on Octave titled “Octave Tutorials”

Comparing Octave with java, there are many differences and I can see how Octave is used for complex numerical computations. The first bits of information I’ve learned so far is just basic arithmetic and how to work with them in Octave. Out of the gate, there is no need for imports or such in order to make formulas that would require the use of powers, just by using ^, Octave knows that we are using powers without having to type out something like math.pow(x,y).



Similarly with arrays and matrices, rather than having a loop in order to multiply specific numbers from an array or matrix, i just need to add the multiply symbol between two matrices in order to print out the result



There is much more that can be done such as using logs or floors, but for now i will focus on re creating the previous programming assignments in this new language

At this point, I decided to swap over to another tutorial as the videos were from a decade ago and my friend recommended a wiki tutorial on how to use the program

<https://en.wikibooks.org/wiki/Octave_Programming_Tutorial/Getting_started>

x = linspace(0, 2\*pi, 100);

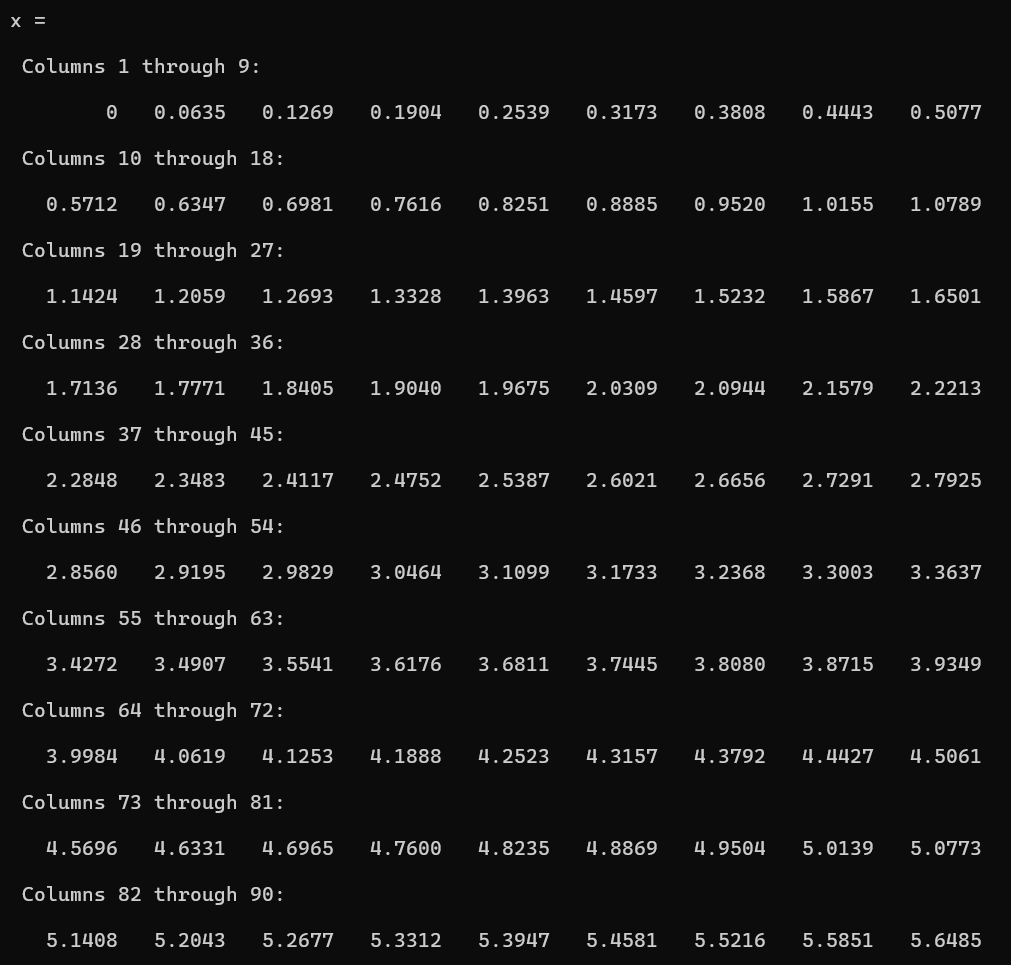
y = sin(x);

plot(x, y);

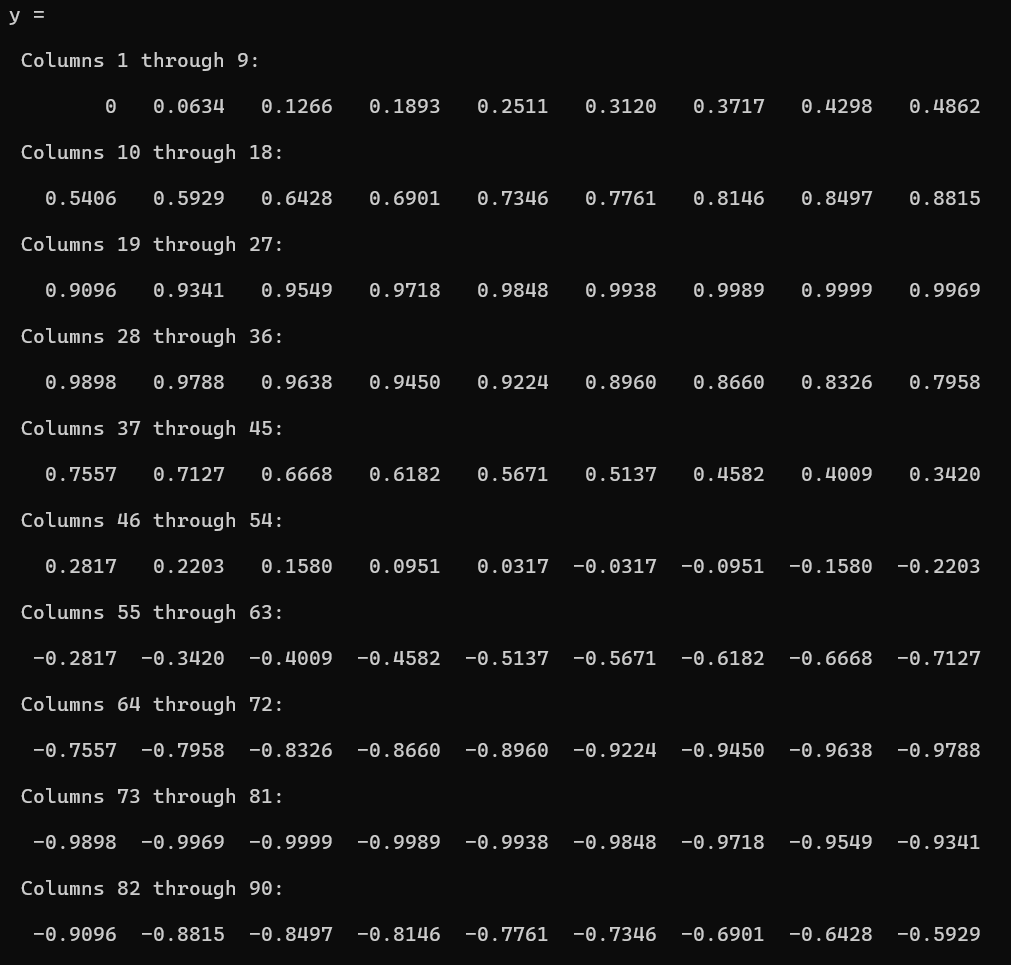
figure;

This is the first instance of plotting in the tutorial and I was surprised at first how compact this is compared to our previous java code which required lengthy code to convert to a csv and plot it

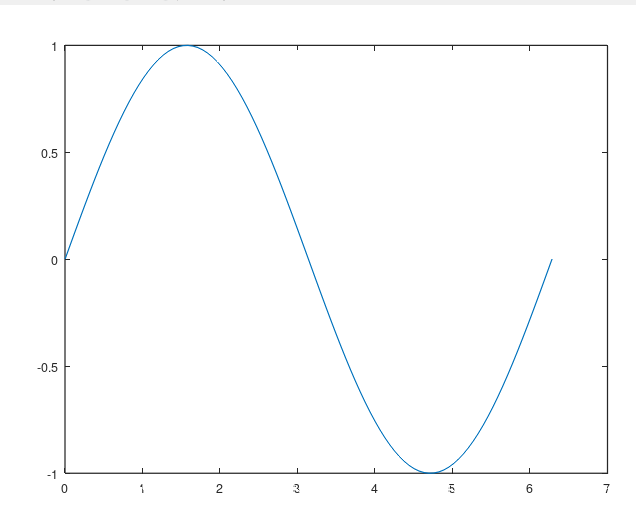
Going over from the top, x = linspace(0, 2\*pi, 100); This creates a vector of 100 values starting from 0 and ending at 2pi (I cannot fit all 100 values into a screenshot)



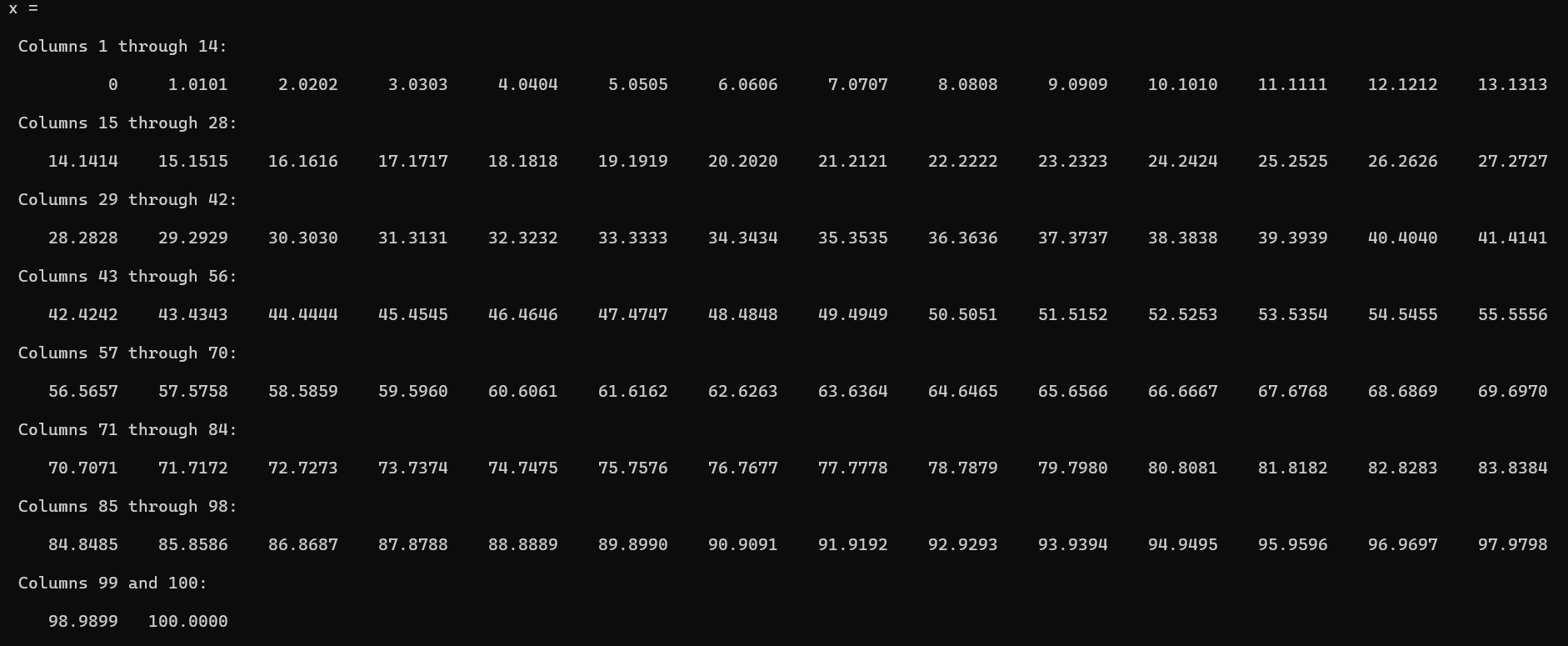
From the second, y=sin(x), y takes a value from the vector x, and converts it to a sin value. This is repeated for all 100 values that is stored in x



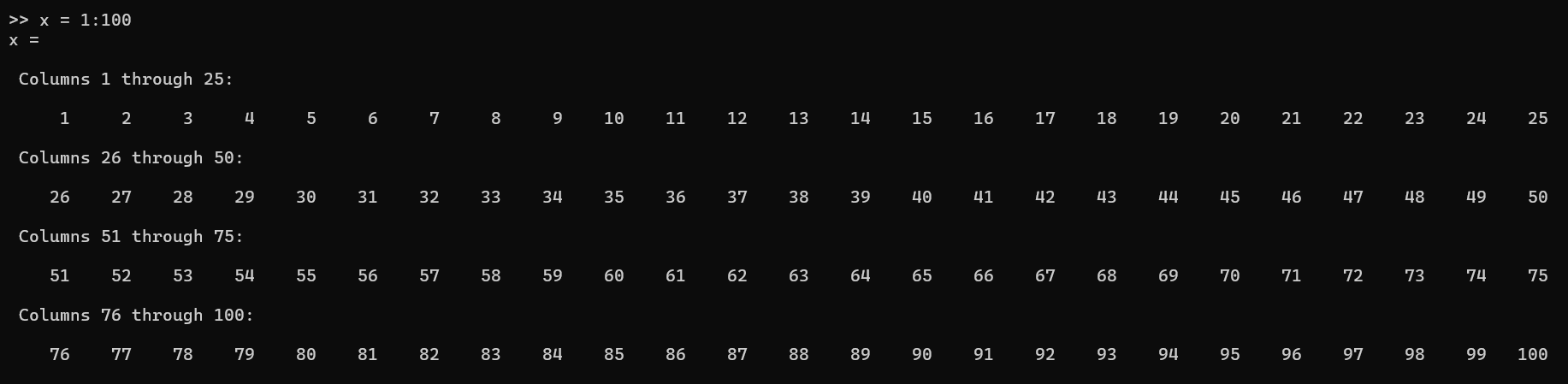
It is then plotted by the line, plot(x,y) and then is output as a graph



Ive tried doing this for my use case, so from 1 too 100 but it wasnt something i could use



So i would use x = 1:100, which sets up a vector of values from 1 to 100



Now i can square each of these values to receive something similar to what i have outputted and graphed in java and excel

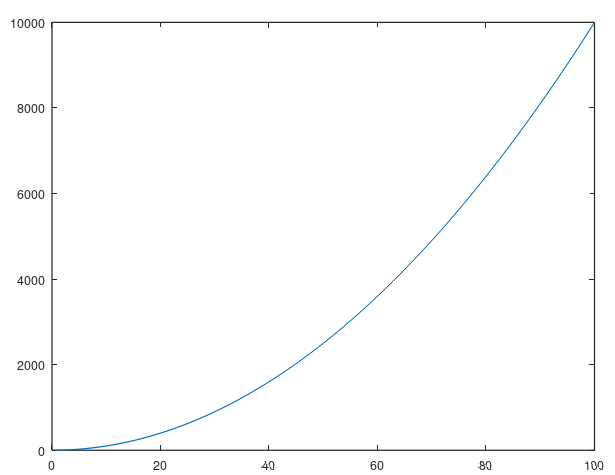
(Forgot to mention that adding ; after the line prevents it from displaying its contents)

So using

x = 1:100;

y = x.^2;

plot(x,y)



In just 4 lines of code i was able to completely graph something i used both lengthy lines in java and my own hands in excel

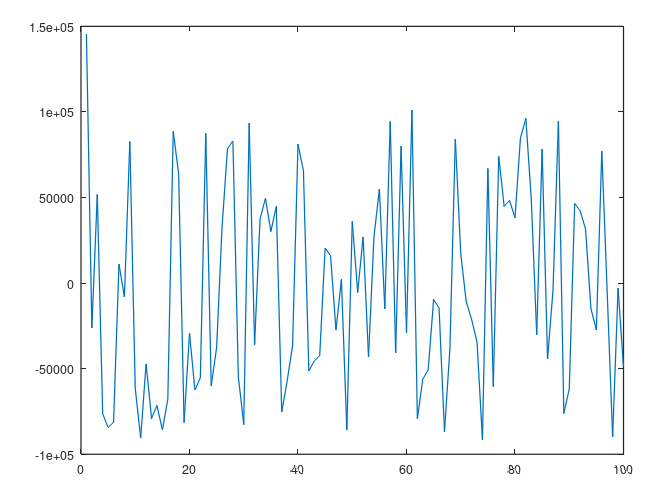
Now onto salting, for this I made use of octaves built in function randi. This generates a random integer from whatever min to whatever max i set. So making use of this and a for loop, i was able to replicate my salting program from java by adding this code to the previous

for i = 1:100;

y(i) = y(i) + randi([-100000,100000]);

endfor

plot(x,y)



Now onto the smoother

for i = 1:100;

start = max(1, i-w);

endVal = min(100, i+w);

y(i) = mean(yValues(start:endVal));

endfor

error: 'yValues' undefined near line 4, column 13

>> plot(x,y)

Finally, time to combine all of these steps together to read and write from a csv

So using these as references

<https://octave.sourceforge.io/octave/function/csvwrite.html>

<https://octave.sourceforge.io/octave/function/csvread.html>

https://docs.octave.org/v4.0.1/Terminal-Input.html

In order to read and write from a csv

I was able to come up with a finished product.

Although it was a little more complicated as this is my first time working with them, i realized how simple it really was. Writing to a csv was as simple as creating a dataset consisting of vectors x and y and then prompting the user using the terminal to write to a csv and thats how i finished the plotter section.

As for the last two, I had to use the csvread function, and although it was tricky at first, i soon realized that it was really similar to the way i read information from my own csv reader class in java as i can create vectors for both x and y by calling

data = csvread(fileExtract);

x = data(:,1);

y = data(:,2)

Similar to creating two arrays from a 2d array

Then i just copy and pasted my previous code and modified it to take user input, and now I am caught up to the first part of PSS, but in a much faster and simpler way