1. With a formula for the derivative function, this code requests the step size and initial coordinate to calculate the point the user wishes to approximate. While calculating using Euler’s method, it also graphs the function and approximating process. While the project seemed intimidating at first, it was less complicated than anticipated, mostly because of the class assignment we had of demonstrating Newton’s Method for approximations. In that project, the idea was about the same where inputs were given and the output was an answer and a graph displayed the answer, but what was especially helpful in reviewing that code was observing how certain quantities were reassigned variables within a for loop. For graphing, looking back at the Fourier series code we did as a class really helped me understand the steps it takes to graph progress that occurs in a for loop. My next step in this process would be to display the actual graph of the function described in order to show the accuracy between different step sizes. One thing I would hope to improve this project with is allowing the derivative function to be an input instead of a given in the code. This would enable the program to be used without editing it to the user’s needs, which is necessary currently. Overall, this project was, despite being confusing at first, interesting and exciting to complete. However, in the end, I believe that this project has shown me important skills that are extremely applicably to my endeavors in college regarding math in computer science.
2. I found the real GDP in the United States of America between 1990 and 2018 through Statistica. Then with the data, I imported the excel sheet to MATLAB with the year as one column vector and the real GDP as another. With both of these as column vectors I was able to make a figure with the year on the x axis and GDP on the y. Those steps were difficult mostly because finding reliable data and analyzing it was difficult. Working on mostly multivariable calculus in MATLAB was, unfortunately, fairly unhelpful for this project. Once I had plot the data points, my next step was to create a trend line. This was the most difficult part of the project, and it was the most time consuming activity in the entire portfolio. This was due to my MATLAB not being fully installed (something I could not fix), and thus I could not use the pre-installed programs. I attempted transposing one of the column vectors to use a different tactic, but that also refused to work. In the end, I found the formula for a least-square regression line in my multivariable textbook, and included it in my program to calculate the line’s slope and y-intercept, which I then combined and plotted on the graph. I don’t think there’s much I could do to improve my project, but if given more time and resources I would provide an approximation feature where one could input a year and using the calculated trend line find the predicted GDP at that specific time.
3. The first portion of this project is problem21, where I used the mesh functions along with linspace to graph an elliptic cone in a 3d coordinate plane. I also included names for the overall graph and the axes. Then, in problem41, I created three variables with different functions using the variable *t*. I implemented fplot3 with my three variables to make a 3d vector graph, labeling the axes and giving a title. I repeated these steps to also make a second parametric vector graph.In problem57 I made a system of three equations and then solved to various moments of Inertia using the solve function. With each moment of inertia’s (Ixy,Ixz, & Iyz) calculated, the code then goes through an if statement to see if either inertia is equal. If it is, then the code states that the moments of inertia is verified. In problem66, given formulas to show the ocean floor, I was able to graph what the ocean floor would look like using mesh functions once again, finally labelling the axes accordingly. I believe after using MATLAB for homework questions I am better at working with its systematics. Moving from using java to MATLAB is a little tricky because of the lack of semicolons, brackets, and minor but significant differences that lead to a large portion of time spent on debugging. The most difficult thing to adjust to in MATLAB is the conditions to be met for a for loop, as in java the syntax semicolons between mathematical statements and in MATLAB colons are used to demonstrate a range of values. Nonetheless,learning MATLAB was important for me as it showed me how to translate my skills in one language to another.