Project 7 - Numeric Computations with Taylor Polynomials

Ryan Scott and Diego Guerra

Grand Canyon University

CST-305: Principles of Modeling and Simulation Lecture & Lab

Professor Citro

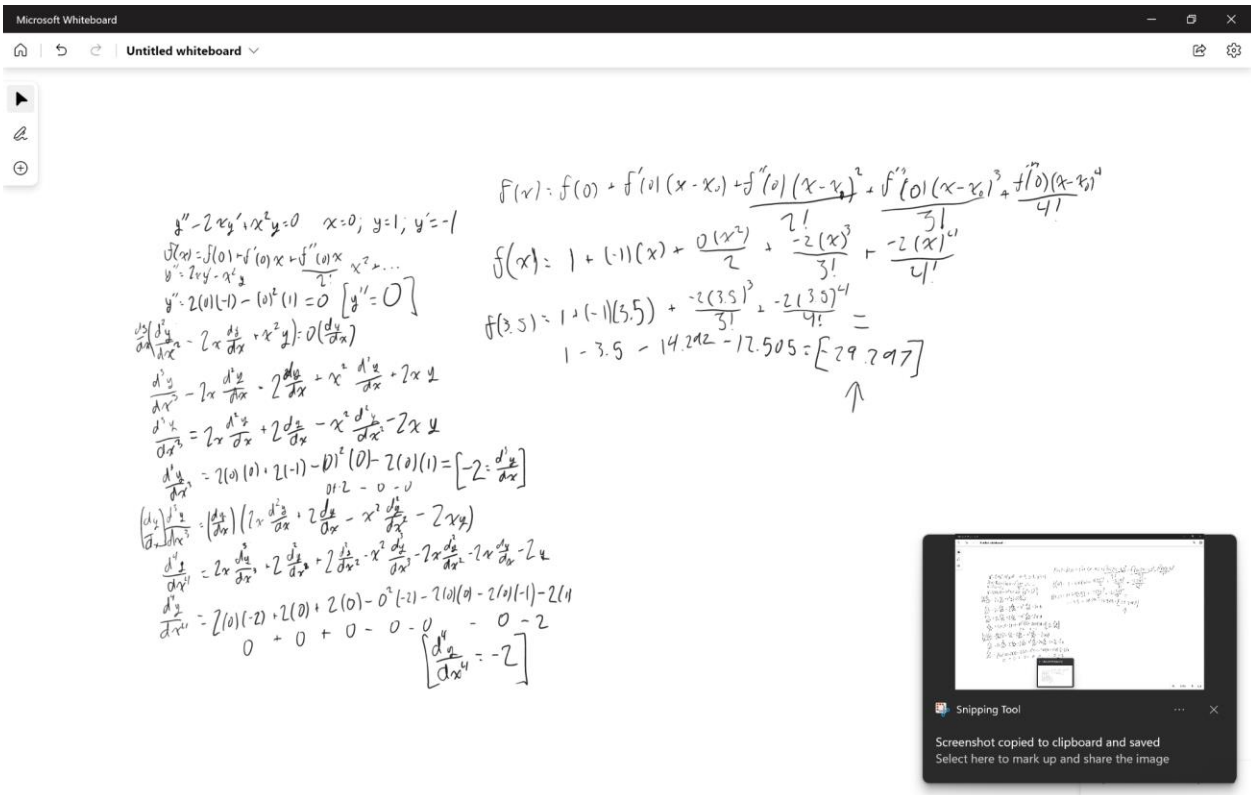
November 25th, 2022

Part 1

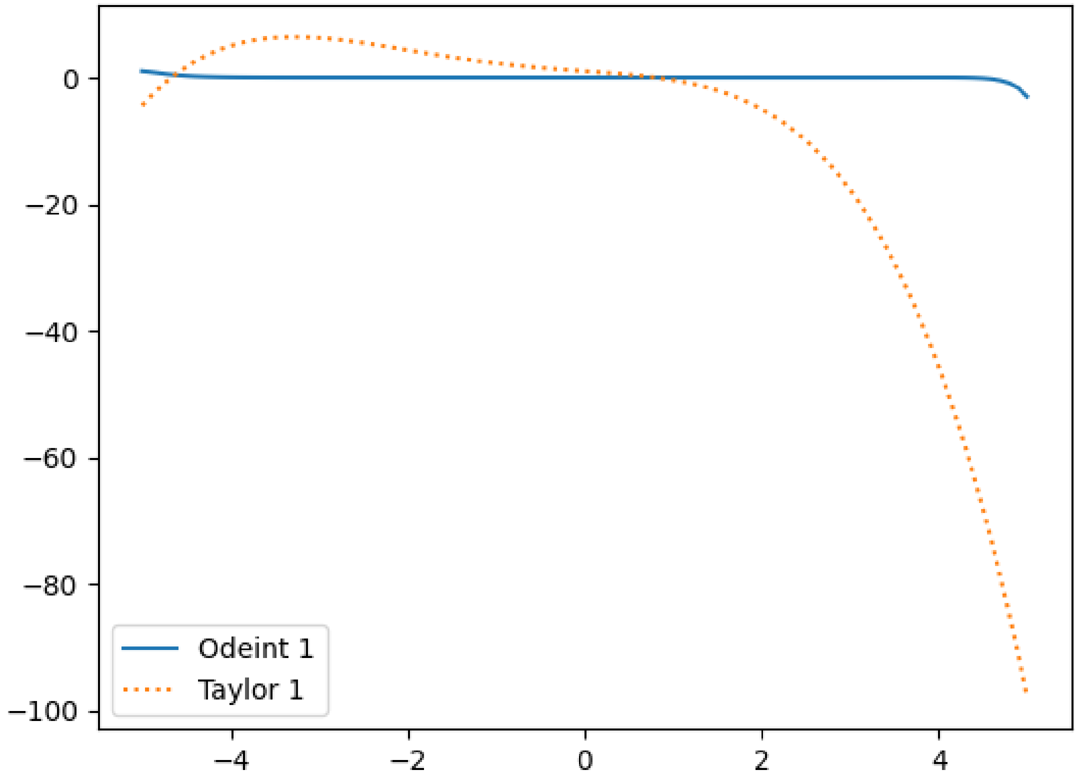
a)

We are to find the Taylor expansion of y’’ - 2xy’ + x^2y = 0 up to n <= 4. Along with that, we are to find the value of y(3.5). The initial conditions of the DE are x = 0, y = 1, y’ = -1.

Here is the handwritten solution, as well as the value of y(3.5).



Here is the equation graphed with odeint next to the Taylor polynomial.

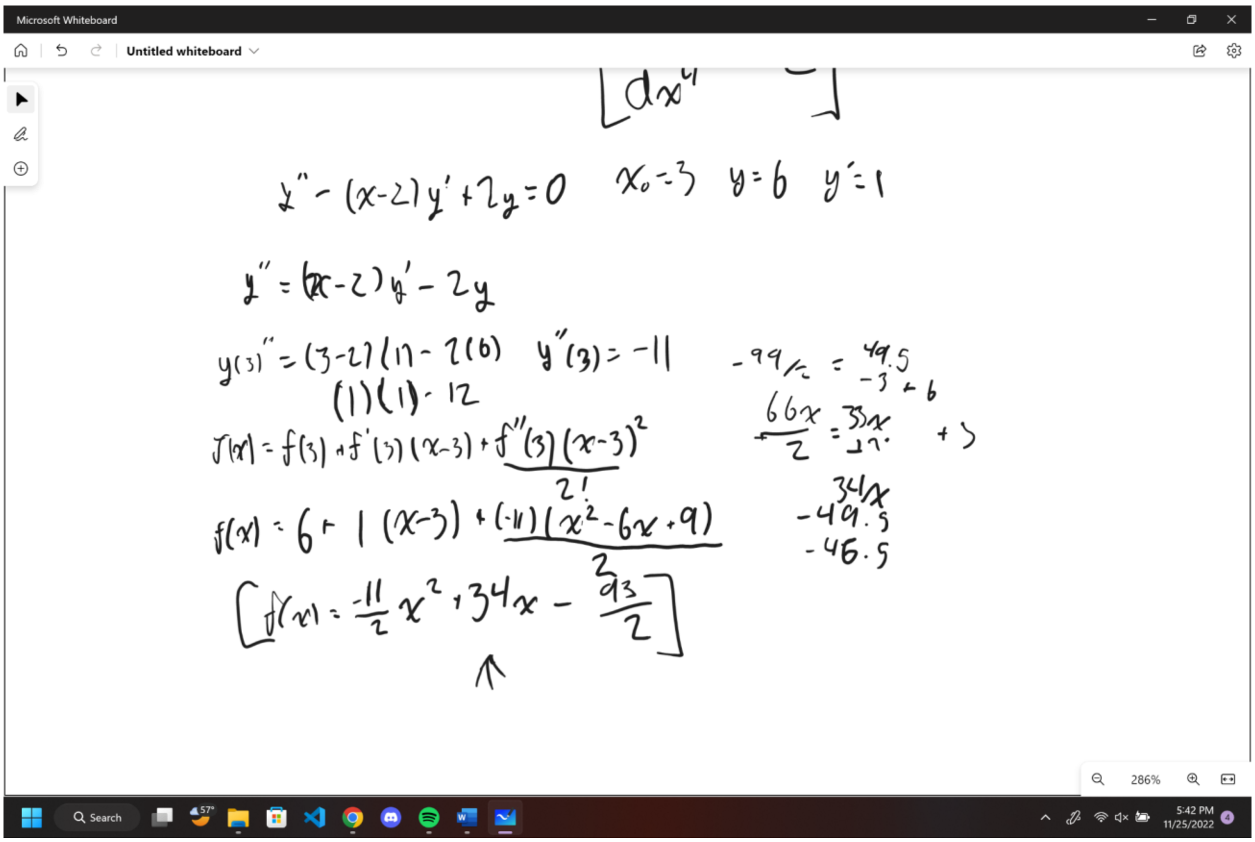


A general aim of the Taylor method is to achieve an accurate representation of the real solution around a certain point. As a result, the graphs tend to converge at that point. For this problem, x = 0 is the point of convergence. At x values far from the point of convergence, these graphs become increasingly distant from each other. It is at the point x = 6 that the Odeint line takes an extremely sharp turn downward, causing the Taylor line to appear flat when it is shown all together.

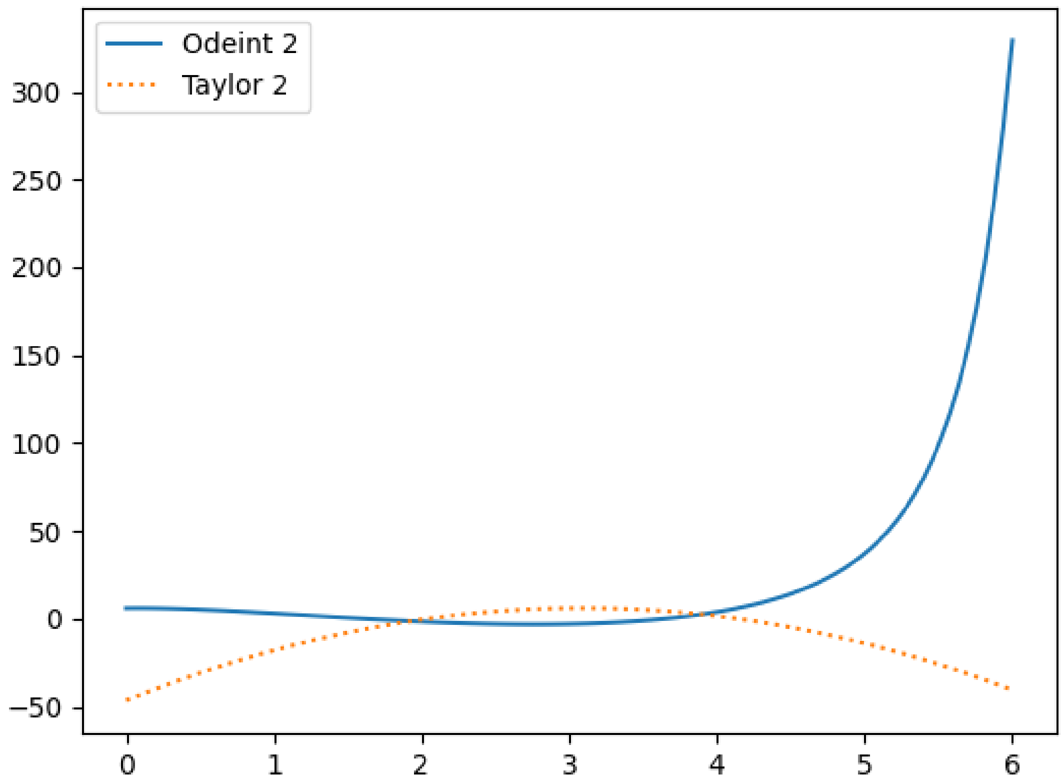
b)

We are to find the Taylor expansion of y'' - (x - 2)y' + 2y = 0 up to n <= 2. The initial conditions of the DE are x = 3, y = 6, y' = 1.

Here is the handwritten solution.



Here is the equation graphed with odeint next to the Taylor polynomial.



There is a point of convergence for this Taylor polynomial around x = 3. In comparison to the first Taylor polynomial, this one is significantly less accurate. It is over 300 y units off and only 3 x units away from convergence. This is due to the fact that this polynomial has a lower degree than the previous one. This one has a degree of 2, while the previous one had a degree of 4. The Odeint line continues to climb sharply until it reaches x = 7, at which point the Taylor line appears flat in comparison.

Part 2

We were to solve the equation (x^2 + 4)y’’ + y = x using the recurrence formula.

$$(x^2+4)(2a_2 + 6a_3 x + 12a_4 x^2 \cdots) +$$

$$(a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots) - x = 0$$

$$(2a_2 + a_0) + x(6a_3 + a_1) + x^2(12a_4 + a_2)$$

$$x^n\left[(n+2)(n+1)a_{n+2} + a_n\right]$$

$$a_{n+2} = \frac{a_n}{(n+2)(n+1)}$$

$\frac{a_3}{2}]$

$$a_{n+2} = \frac{n^2 - n + 1}{4(n+2)(n+1)} a_n$$

$n=0 \quad a_2 = -\frac{1}{8} a_0$

$n=1 \quad a_3 = \frac{1}{24} a_1$

$n=2 \quad a_4 = \frac{1}{128} a_2$

$n=3 \quad a_5 = \frac{7}{1920} a_3$

$n=4 \quad a_6 = \frac{-13}{15360} a_4$

$n=5 \quad a_7 = \frac{-1}{151760} a_5$

$n=6 \quad a_8 = \frac{407}{3440640} a_6$

$n=7 \quad a_9 = \frac{7...}{4423680} a_7$

$n=8 \quad a_{10} = \ldots a_8$

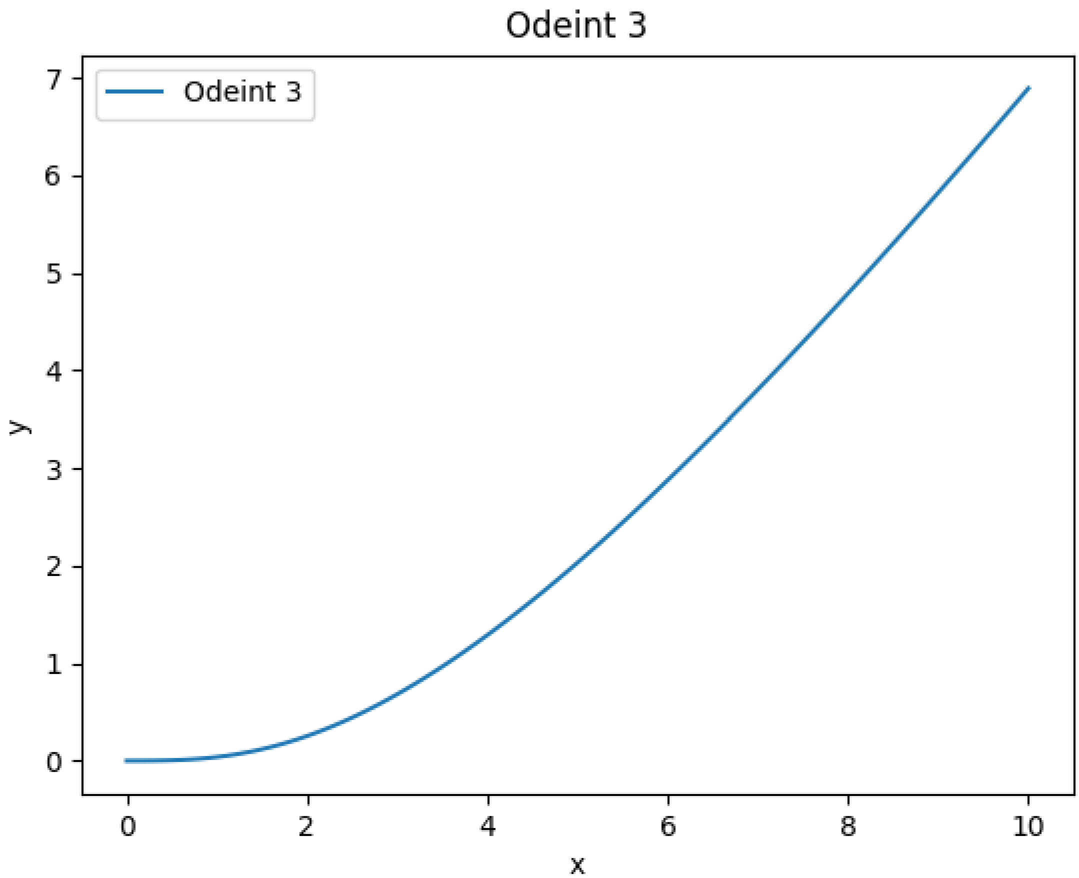
$$y = a_0 y_1(x) + a_1 y_2(x) + \ldots$$

$$a_0 + a_1 x + \tfrac{1}{6} a_0 x^2 + \tfrac{1}{12} a_1 x^3 + \tfrac{1}{20} a_0 x^4 + \tfrac{1}{30} a_1 x^5 + \tfrac{1}{56} a_0 x^6 + \tfrac{1}{72} a_1 x^7 \quad \tfrac{1}{90} a x^8$$

$$y = \left[\tfrac{1}{24} x^3 - \tfrac{7}{1920} x^5 + \ldots\right] + a_0 \left[1 - \tfrac{1}{8} x^2 + \tfrac{1}{128} x^4 + \ldots\right]$$

$$+ \; a_1 \left[x - \tfrac{1}{24} x^3 + \tfrac{7}{1920} x^5 + \ldots\right]$$

Here we have a graph of the original equation using odeint.



Part 3

Our task is to develop a differential equation that models computer system performance. The values displayed by this equation are not grounded in reality, but are merely theoretical in nature. I propose the following equation:

$$(1/x^2)y'' + (1/x)y' + y = x$$

Where:

x = cost

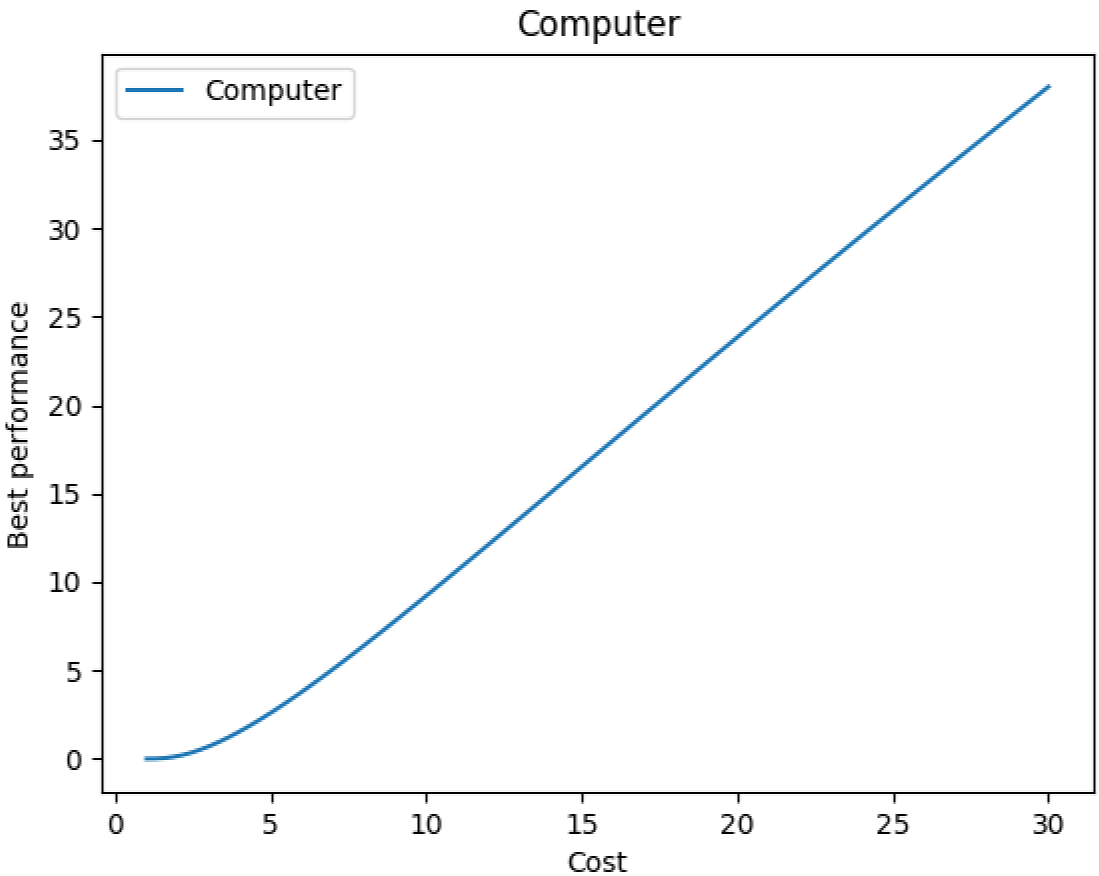
y'' = cpu speed

y' = memory speed

y = memory size

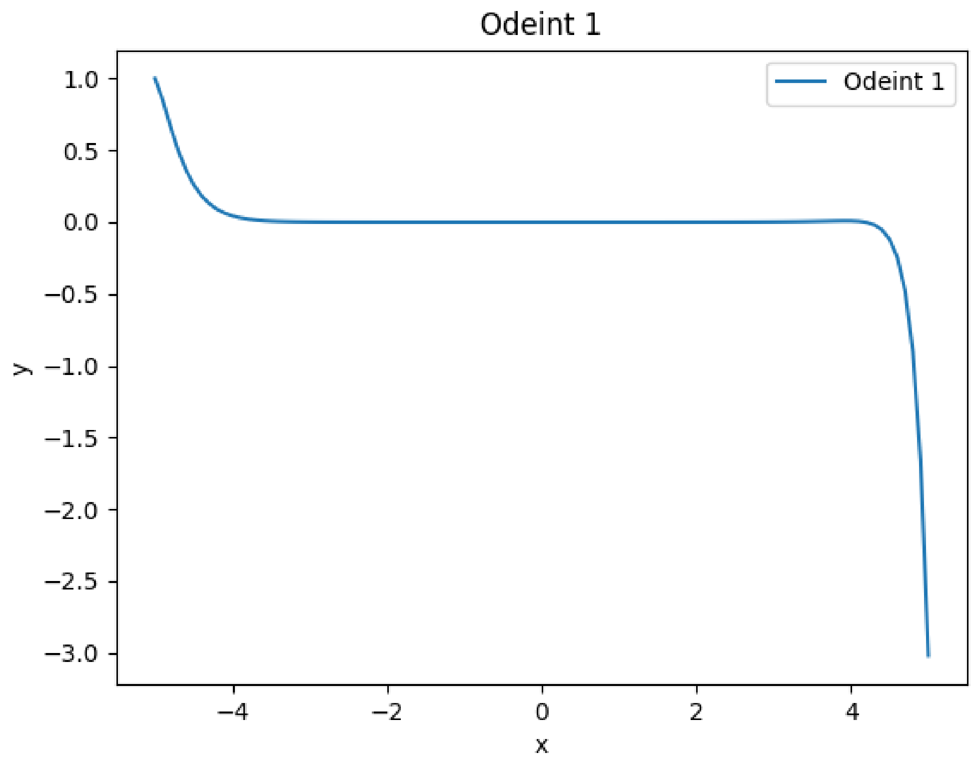
The equation was generated by isolating x on the right-hand side, making it clear that as the cost increases, the computer parts will also increase. As far as performance goes, I considered the CPU to be the most important factor, followed by memory speed and memory size. When increasing the cost of the parts, I placed fractions that were proportional to the cost in front of them, causing the parts to increase less when the cost was increased. Since all of the different factors are interdependent and could bottleneck if one part is not upgraded, I believe it is important to balance these y values, as spending a lot of money on one thing may not be the best way to improve performance.

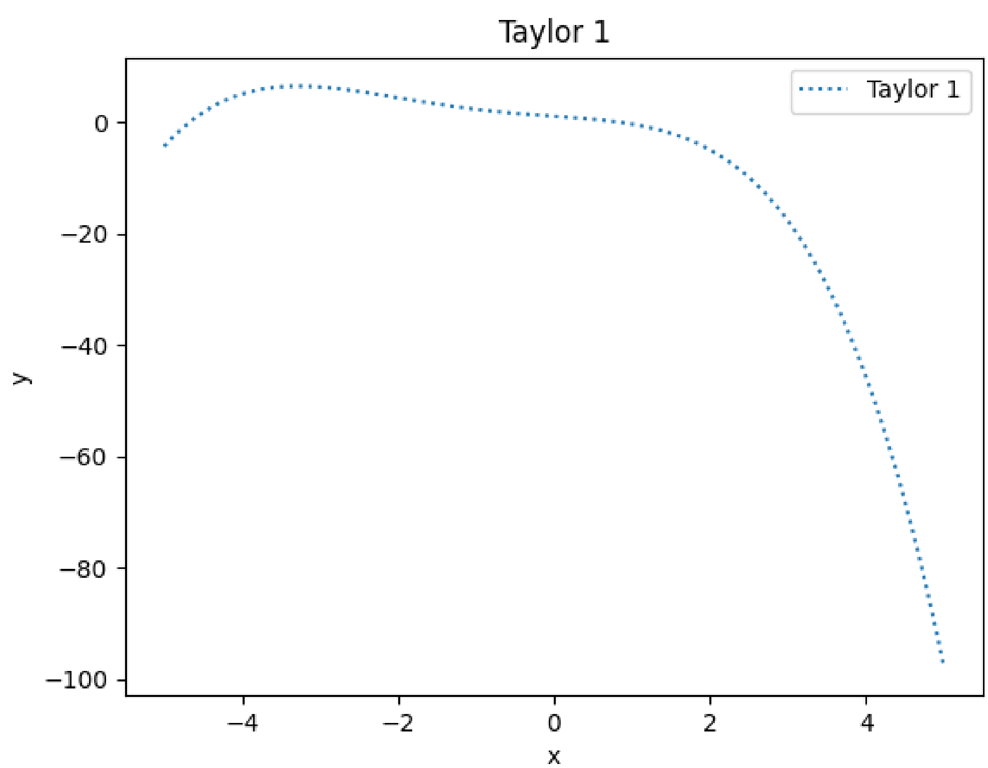
A graph was created in Python for this equation to illustrate the increased performance if the cost is distributed as described in the equation.

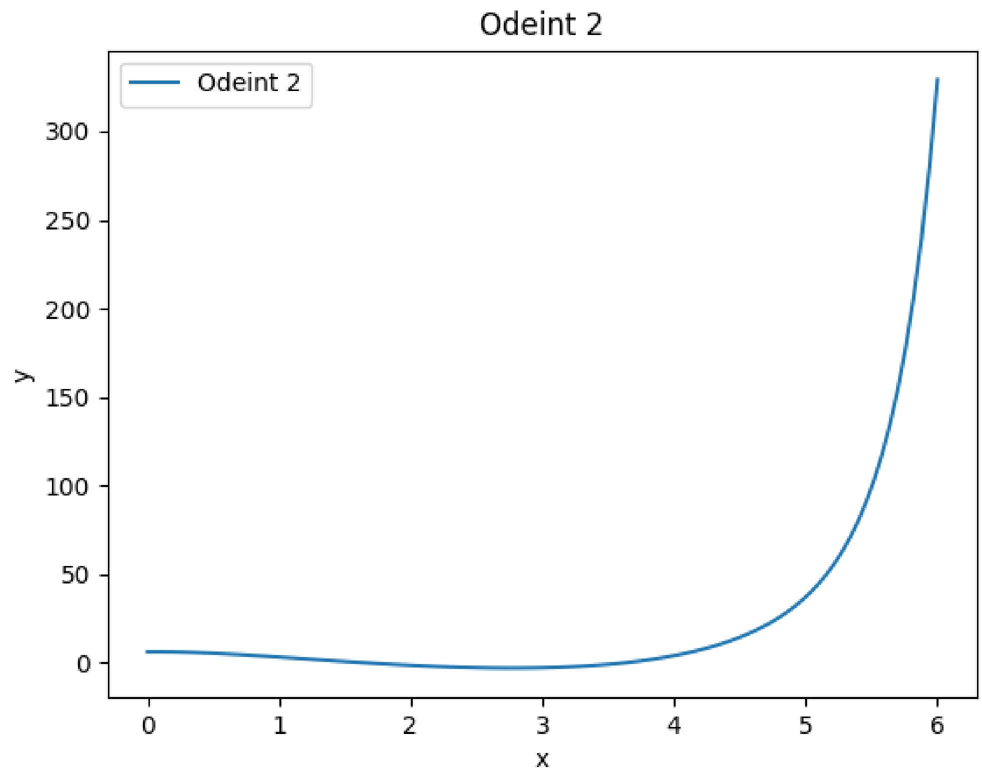


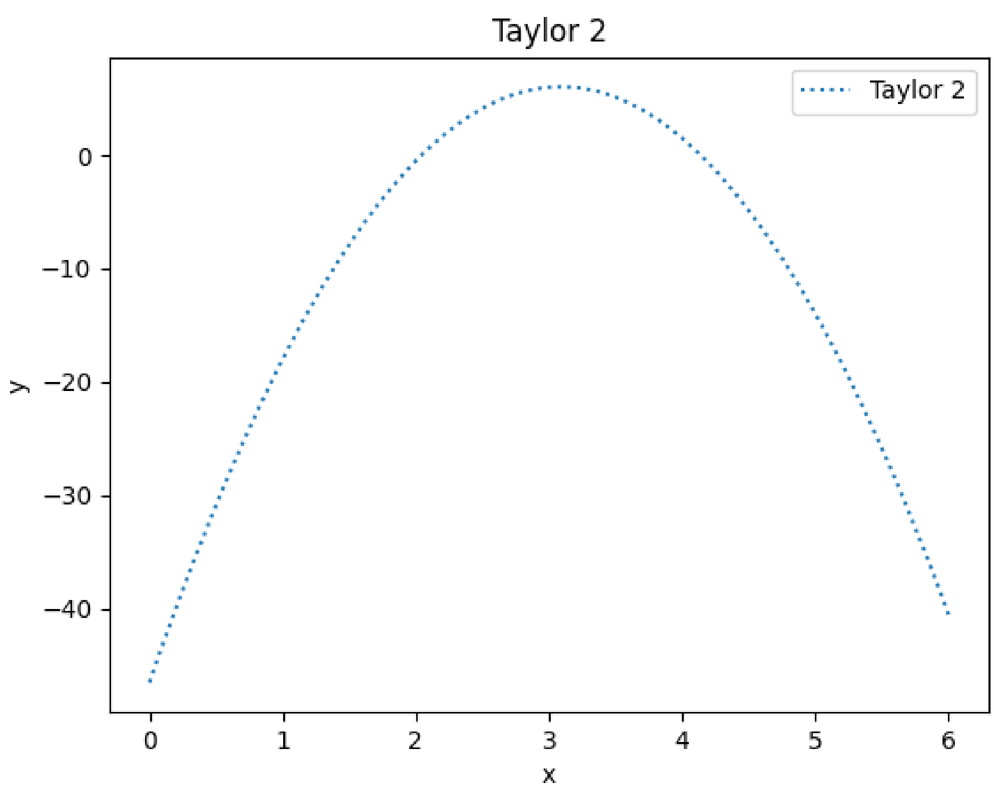
As the cost of the machine increases, the performance of the machine increases as well. Performance increases slowly at first, but once the system begins working well, it continues to improve at a reasonable rate.

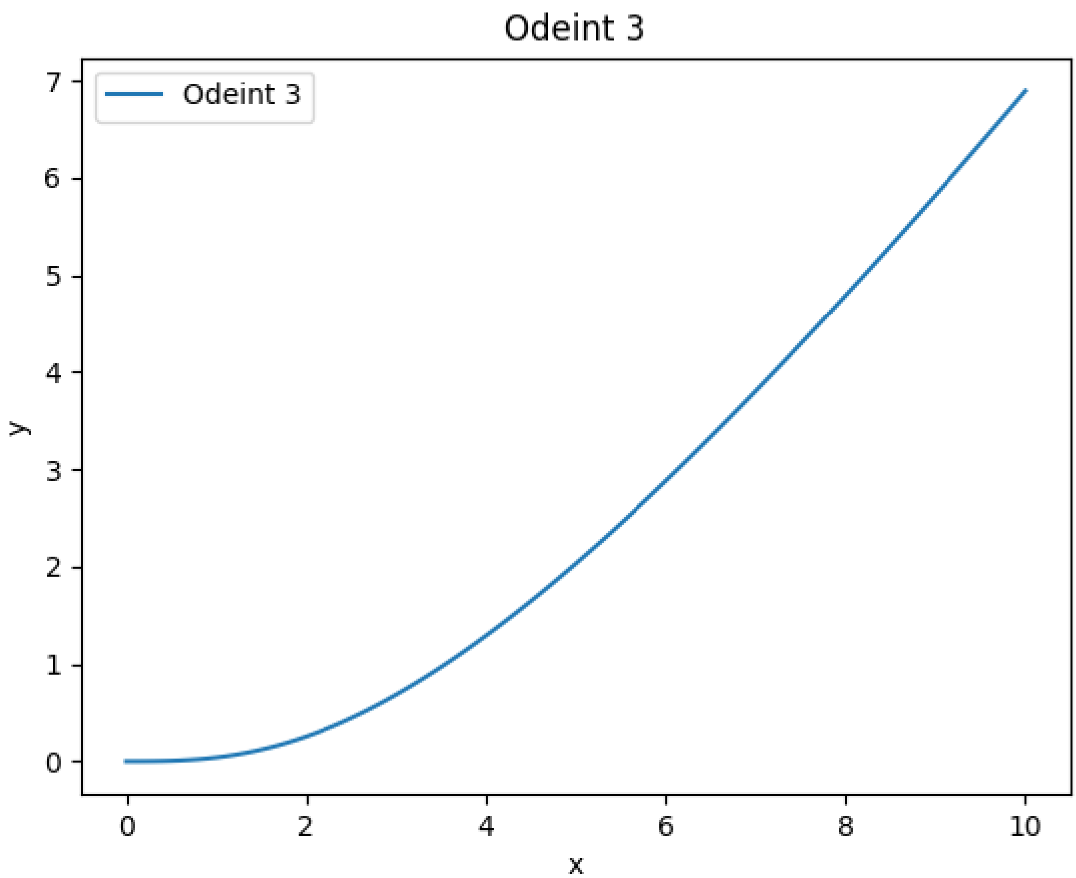
All Odeint and Taylor Graphs:






Odeint 2
Odeint 2
300
250
200
150
100
50
0
y
0
1
2
3
4
5
6
x


Taylor 2
Taylor 2
0
−10
−20
−30
−40
y
0
1
2
3
4
5
6
x



Odeint 3
Odeint 3
y
x
0
1
2
3
4
5
6
7
0
2
4
6
8
10