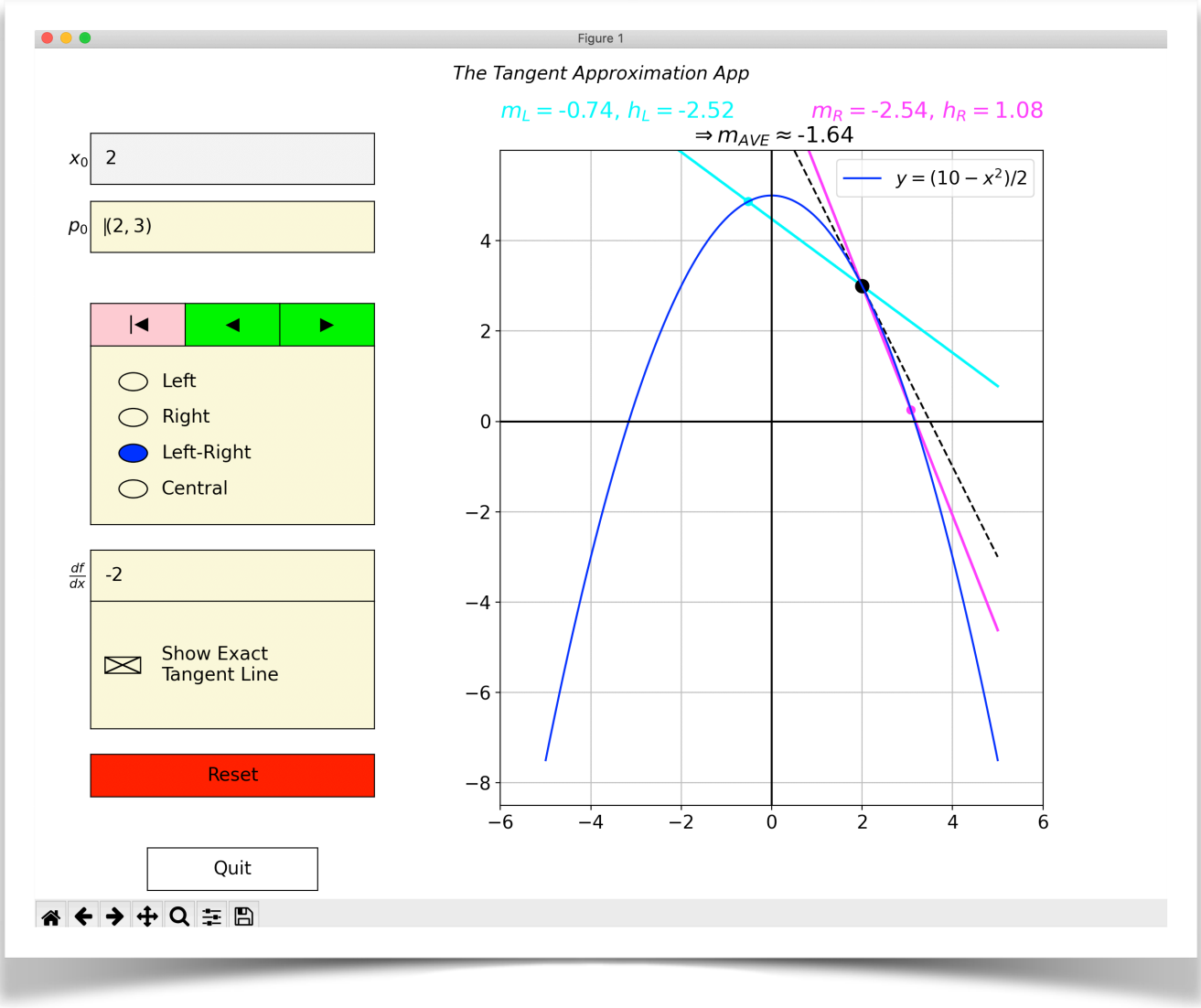
UMass | Dartmouth

The ACCOMPLISH Project
The Tangent Approximation App

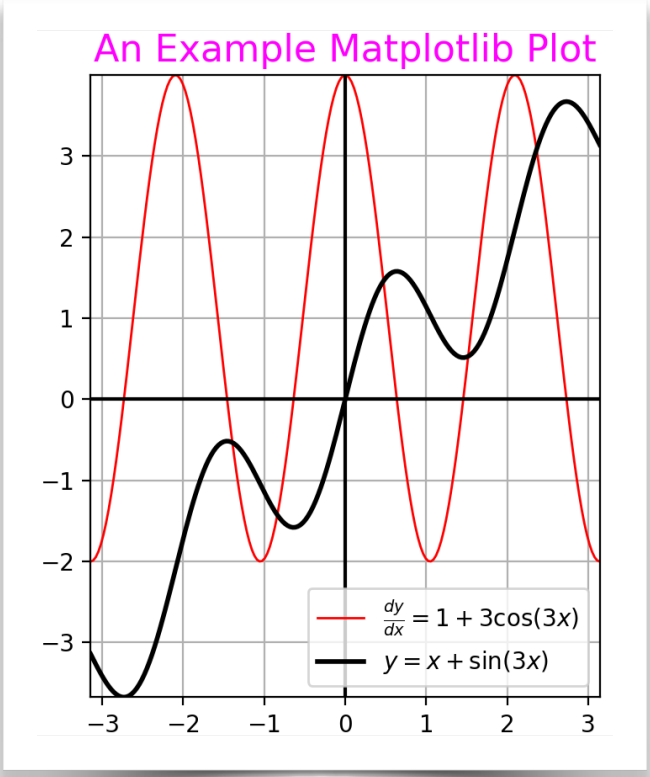


This module will introduce students to methods for approximating the tangent to the graph of a function. For this module, you will complete a Python 3 - Matplotlib application that uses secants to approximate the tangent line to the graph of a function $y = f(x)$ at a point. The entire program is about 470 lines of Python 3 code but you will need to only need to write about 20 lines of code. The app was written using the Anaconda Python Distribution using the Spyder IDE. The app runs under Mac OS Catalina, Mac OS Big Sur, Ubuntu 20.04, and Windows 10.

This module folder (**TheTangentApproximation)** contains the following files and subfolders:

- This **READ-ME-The-Tangent-Approximation-Module.pdf**
- The **docsAndVideo** subfolder containing
  a) The module video: **TheTangentApproximationModule.mp4**
  b) A video demonstrating the app: **TheTangentApproximationAppDemo.mp**4
- The **code** subfolder containing
  a) The subfolder **instructor_tangemtApproximation** containing code files for the instructor.
    i) The **READ_ME_Matplotlib_ExamplePlot.py** program file.
    ii) The main program: **tangentApproximationAppV1.py**
    iii) The completed user library: **tangentApproximationLib.py**
    iv) Examples of how to use the completed app with different functions: **example0.py**, **example1.py**, **example2.py**, **example3.py**, **example4.py, example5.py**, and **example6.py**
  b) The subfolder **student_tangemtApproximation** containing code files for the student.
    i) The READ_ME_Matplotlib_ExamplePlot.py
    ii) The main program: **tangentApproximationAppV1.py**
    iii) The completed user library: **tangentApproximationStudentLib.py**

The following figure is an example of plotting a function and its derivative using **Python 3**, **Matplotlib** and **Numpy**. The code is shown after the figure.



```
from matplotlib import pyplot as plt
import numpy as np
# Set the domain
xMin = -np.pi; xMax = np.pi
# Use numpy to generate 200 x-value
x = np.linspace(-np.pi, np.pi, 200)
# Generate the function's values
y = x + np.sin(3*x)
# Generate the derivative's values
dydx = 1+ 3*np.cos(3*x)
# Determine the range minimum and maximum
yMin = np.min([y, dydx]); yMax = np.max([y, dydx])
# Begin the plot
fig, ax = plt.subplots()
# Plot the function's derivative with a LaTeX label
ax.plot(x, dydx, color='red', linewidth=1,
        label= '$\\frac{dy}{dx}=1+3\cos(3x)$')
# Plot the function with a LaTeX label
ax.plot(x, y, color='black', linewidth=2,
        label= '$y=x+\sin(3x)$')
# Show the legend
ax.legend()
# Set graph's x-axis and y-axis limits
ax.axis([xMin, xMax, yMin, yMax])
# Make the axes scaling equal
ax.set_aspect('equal')
ax.grid(True, which='both')
# Highlight the x-axis and y-axis
ax.axhline(y=0, color='black')
ax.axvline(x=0, color='black')
# Add a title to the plot
ax.set_title("An Example Matplotlib Plot", fontsize=16,
             color='magenta')
# Show the plot
plt.show()
```

Hopefully, this example along with the project overview video will give you some hints on how to complete the skeleton functions in the **tangentApproximationLibStudent.py** file**.**

Prof. Adam O. Hausknecht
Department of Mathematics
UMass Dartmouth