Illustrating Theorems from Calculus

Instructions

1. Open the template file: username_illustrating_calc.mlx. Save this template into your chosen MATLAB folder for this course and replace 'username' with your username.
2. Complete the four problems outlined in this document. You are welcome and encouraged to discuss these with your classmates and instructor!
3. Suppress all numerical output by terminating commands with `;`. The only displayed results should be the relevant plots and tables.
4. Any mathematical notation in your discussion must be typeset with the equation editor.
5. When finished, run and save your .mlx file first, then export it as a pdf.

Useful Code to Get Started

Creating Vector Variables

```
% Rows and Columns
row = [1 2 3]; % row vector
col = [1; 2; 3]; % column vector
% Vectors with Incremental Elements
% Option 1: vector starting at 0, ending at 10, elements in increments of 0.5
X = 0:0.5:10;
N = numel(X); % N = number of elements in X
% Option 2: vector starting at 10, ending at 50, with N equally spaced elements
Y = linspace(10,50,N);
```

Basic Plotting

```
figure; % create new figure
plot(X,Y) % X and Y must be vectors of same size
xlabel 'x', ylabel 'y' % add axis labels
title('Example of a Plot') % add a title
```

1 Plotting Functions

Anonymous Functions

- a function that stores a single executable line of code producing one array of output.
- basic syntax: `myFunc = @(input) (single executable line of code)`

Example: Let's do several calculations with the function $g(x) = x\sqrt{1+\ln(x)}$.

- Create a function handle `>> g = @(x) x.*sqrt(1 + log(x));`
- Use `g(input)` to evaluate g on `input` values (e.g. `g(1)` or `g([1,2])`).
- **Notice element-wise multiplication.**
  The function acts *element-wise* on vector inputs.
- Check $g([1,2]) = [g(1), g(2)]$ in the command line.

Exercise 1

Consider the function $f(x) = e^{-x} + 2x^2\sin(x)$ on the interval $[-2, 3]$.

1. Create an anonymous function handle for f.
   - Make sure you use *element-wise multiplication* in your function formula.
   - Also, remember the exponential function is `exp` in MATLAB.
2. Create a variable called `X1` of 100 equally spaced points between -2 and 3 using the command `linspace`.
3. Create a variable called `Y1` of f evaluated at each point in `X1`.
4. Check your workspace to see that `X1` and `Y1` are of the same size.
5. Plot `(X1,Y1)` using command `plot`.
   - Make your plot blue.
   - Do not use markers (you want a smooth-looking plot).
   - Use command `axis` to set the axis bounds $[-2, 3]$ for x and $[-2, 10]$ for y.
   - Include axis labels and title your plot "Plot of f(x)."
6. Discussion: Explain where in the formula for f you must use element-wise arithmetic and why.

2 Illustrating the Extreme Value Theorem

In this section, you will create an illustration for the Extreme Value Theorem in MATLAB using the same function $f(x) = e^{-x} + 2x^2 \sin(x)$ on the same interval $[-2, 3]$. We will do this by investigating the MATLAB commands `max` and `min`, as well as a `table`.

Maximum (`max`) and Minimum (`min`) in MATLAB

- Commands: `max` and `min`
- These functions return the maximum/minimum value(s) of an array.
- Basic syntax: `M = max(A)`.
- When A is a vector, it returns the maximum/minimum value of A.
- `[M,I] = max(A)` gives you the *both* the maximal value of the vector A in the variable `M` *and* tells you its index, `I`, so that `A(I) = M` (and similar for `min`).

Making a table with `table`

- Basic syntax (for a table with 2 columns):

```
table(Vector1,Vector2,'RowNames',{'Row1Name','Row2Name'}, ...
    'VariableNames',{'Col1Name','Col2Name'})
```

- `Vector1` and `Vector2` should be *column* vectors.
- The entries in `'RowNames'` and `'VariableNames'` must equal the number of rows and columns respectively.

Exercise 2

1. Copy and paste your code from Exercise 1 into this section in your live script. Change the title of the plot to "Illustration of the Extreme Value Theorem."
2. Use the `max` command to find the max value (output `M`) of `Y1` and its index (output `I`). Repeat for the minimum value (using `min`), but use the output variables `[L,J]`.
3. Add the max and min points in our plot.
   - Use the command `hold on` and execute another plot command to plot the max point, `(X1(I),M)` and min point, `(X1(J),L)`.
   - Use an open circle as a marker and the color red.

4. Look up the command `text` in your command line.
   Use the `text` command to label the max and min points on your plot as "max" and "min."

5. Recreate this table displaying the max and min values and locations using the command `table`. Make sure the "Location" is the x-value of the extremal point (not the index value).

		Max	Min
1	Location		
2	Value		

   To make this table, first you need to first make two vectors, `MaxVector` and `MinVector`, that store the x and y values of the extrema.

6. Discussion: Write a paragraph explaining how your plot illustrates the Extreme Value Theorem as stated in the live script.

3 Illustrating the Intermediate Value Theorem

Next, you will illustrate the Intermediate Value Theorem. We will use the same function and variables as we did in the previous section to create a new illustration.

Exercise 3

1. Copy and paste your code from Exercise 1 into this section in your live script.
   Change the title of the plot to "Illustration of the Intermediate Value Theorem."

2. Add the following MATLAB commands to your code.

```
V = L + rand(1,1)*(M-L);
fMinusV = @(x) f(x)-V;
c1 = fzero(fMinusV,0);
hold on;
yline(V,'-r','y=V')
plot(c1,V,'ro')
```

3. Run this section several times and take note of what happens.

4. Add annotations to the above commands using `%` explaining what each line is doing.
   Remember, you can use `>> help commandName` to quickly look up new commands.

5. Discussion: Write a paragraph sentences explaining how running this section over and over illustrates the Intermediate Value Theorem as stated in this section of your live script.

4 Illustrating the Mean Value Theorem

In this final exercise, you will create an illustration for the Mean Value Theorem, similar to what you did for the other Value theorems in lab 0. We will use the same function as in the previous sections.

Exercise 4

1. Copy and paste your code from Exercise 1 into this section. Change the title of the plot to "Illustration of the Mean Value Theorem."

2. Calculate the slope of the secant line connecting $(-2, f(-2))$ to $(3, f(3))$. Save this calculation in the variable `m`.

3. Create an anonymous function handle for the equation of the secant line. Call this function S (you should use `m` in your formula).

4. Find $f'(x)$ on paper. Make an anonymous function handle for $f'(x)$ (call it `df`). Remember to use element-wise arithmetic!

5. Add the following MATLAB commands to your code.

```
dfMinusm = @(x) df(x)-m;
x0 = 0;
c2 = fzero(dfMinusm,x0);
```

6. Run the code and add annotations to each line of the above code explaining what the code does.

7. Create an anonymous function handle for the tangent line at point `c2`. Call the function `L`.

8. Add the secant line and tangent line to your plot.

   - Use a solid red line for the secant line and a dashed red line for the tangent line.
   - Include a Legend for your plot by using the `legend` command (use `>> help legend` to see how to use this command).

9. Discussion: Write a paragraph explaining how this section illustrates the Mean Value Theorem as stated in this section of the live script.