# Solve integral Problems Using MATLAB

*Matthew Shih, Andrew Minh Nguyen*
*Affiliation of Authors:*
*Revelle College, Astrophysics major*
*Revelle College, Computer science major*

## 1. Statement of Problem:

College-level math classes require students to solve complex equations. In Math 20B, or Calculus 2, students solve integrals with trigonometry and support their answers with proofs which may require several pages. Certain questions also ask students to graph complex functions such as $\sin(x)\,cos^2(x)$ which can be hard to visualize and graph precisely. MATLAB's built-in functions such as ***dsolve***, ***integral*** and ***fplot*** can all be used to integrate and plot complex mathematical functions but require the user to re-write script for different functions and recall all the different commands which can oftentimes be time consuming and inefficient.

## 2. Method:

In our script, we combine the different MATLAB built in functions used to solve integration problems with ***while-loop*** and ***if-elseif-else statements*** to create a script which prompts the user to enter any function with respect to x and they will be given a selection tab with the options: Option 1 – returns the indefinite integral of the inputted function. Option 2 – prompts the user to enter the upper and lower bound for their inputted function and then returns the area of the definite value. Option 3 – prompts the user to enter the left and right domain of their input function and then returns a graph of the function within those bounds. Option 4 – allows the user to change the function they inputted.

Since we want our program to continue allowing the user to solve functions after their first input, we used a ***while-loop*** with the condition "while true" followed later on by another while-loop with the condition "selection ~= [1,2,3,4]". Since the first while statement is always true, after the script finishes a cycle, MATLAB will continue running the script until 'q' is pressed to stop the program. Next our script utilizes 4 ***fprintf*** statements to display the options available to the user and prompt them to input the selection number of their choice. A second ***while-loop***, along with a series of ***if-else-if-else statements***, when a certain selection number is inputted, MATLAB will only execute the statements which satisfy the condition, meaning that only one of the 4 options will be outputted at a given time.

To make sure that MATLAB is able to read the input from the user, we use the MATLAB built-in functions ***str2sym*** and ***matlabFunction*** to convert the users input (originally stored as a string) into a symbolic expression and into a function which can be used later in MATLAB built-in functions such as ***dsolve*** and ***integral***. In the case where the user enters a function which cannot be read by MATLAB (such as the function '2x'), we used the MATLAB built-in functions ***try*** and ***catch*** within the first while-loop to check if the function is valid. If the function is not valid, MATLAB will proceed to 'catch' the error, print a warning message (indicating that the function is not valid) and prompt the user to input a new function.

The full script can be found in the Appendix of this document.

## 3. Results and Discussion:

By running our script with the input function being $\sin(x)\,cos^2(x)$, we get the following outputs for the respective options (for options 2 and 3, we decided to set the upper bound and lower bound to be equal to 0 and $2\pi$):

Command Window

```
Type 'q' to quit
Please type in your functions using x:
sin(x)*(cos(x))^2
Option 1: Indefinite integral
Option 2: Definite integral
Option 3: Graph of function
Option 4: Definite integral graph
Option 5: Change function
Input number for desired option: 1
selection =
     1
zint =
C1 - (2*(3*tan(x/2)^4 + 1))/(3*(tan(x/2)^2 + 1)^3)
```

Command Window

```
Type 'q' to quit
Please type in your functions using x:
sin(x)*(cos(x))^2
Option 1: Indefinite integral
Option 2: Definite integral
Option 3: Graph of function
Option 4: Definite integral graph
Option 5: Change function
Input number for desired option: 2
selection =
     2
Input the lower bound for your integral: 0
a =
     0
Input the upper bound for your integral: 2*pi
b =
    6.2832
area =
  -1.0408e-16
```

Command Window

```
Type 'q' to quit
Please type in your functions using x:
sin(x)*(cos(x))^2
Option 1: Indefinite integral
Option 2: Definite integral
Option 3: Graph of function
Option 4: Definite integral graph
Option 5: Change function
Input number for desired option: 3
selection =
     3
Input left domain of the graph: 0
x1 =
     0
Input right domain of the graph: 2*pi
x2 =
    6.2832
```

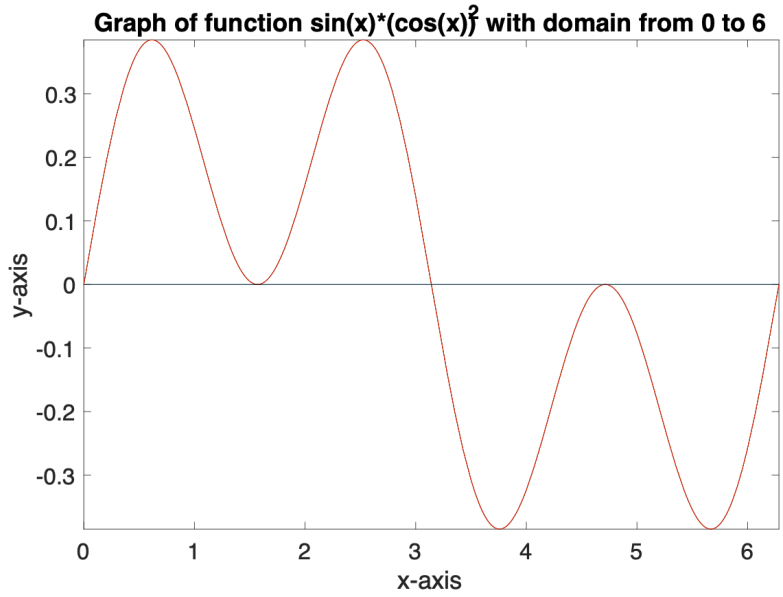


**Figure 1:** Graph plotted by MATLAB built-in function ***fplot***

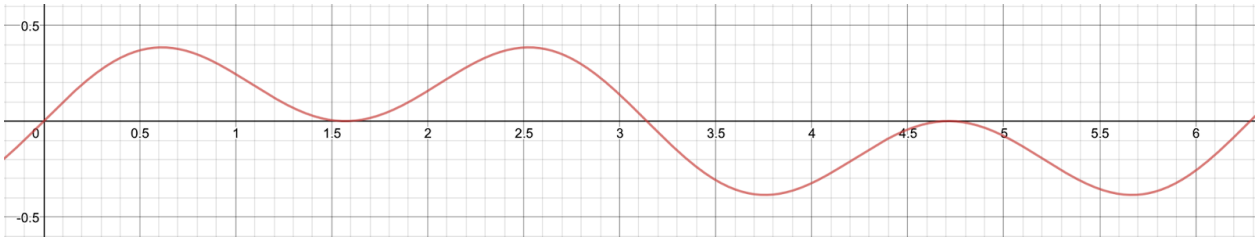


**Figure 2:** Graph plotted by Desmos.com
*source:* https://www.desmos.com/calculator

To check these answers, we used the integral calculator from *https://www.integral-calculator.com/* and the graphing website https://www.desmos.com/ to verify our answers and the shape of our graph obtained from our script. From the integral calculator, the results for the definite integral of between 0 and is 0 which matches the value calculated from our MATLAB script (the small discrepancy is due to rounding errors within MATLAB which can be neglected) and the indefinite integral is equal to  which can be obtained by simplifying the indefinite integral obtained from our MATLAB script. This shows that our calculated values are consistent with values from other sources.

## 4. Conclusion:

In summary, we have presented a way of combining ***while-loop*** and ***if-else-if*** statements to create a program which allows the user to perform multiple commands on one or more inputs in a short span of time. This work indicates that it is possible to convert the users input function (stored as a string) into a function which MATLAB can then solve and plot. Our script presents a possibility for researchers or students to solve multiple complex and time-consuming problems in a short time frame whilst only having to change their input.

## 5. Acknowledgement:

The authors acknowledge the MATLAB license provided by the University of San Diego California and MATLAB Documentation. We also acknowledge the integral calculator from *https://www.integral-calculator.com/* and the graphing option from *https://www.desmos.com/calculator* to provide us with the reference values for our calculations.

## 6. Appendix:

```
% Program for evaluating and graphing integrals
clear all
clc
syms y(x) x;
Dy = diff(y);

while true
    % selection reverts back to 0 to prevent infinite loop
    selection = 0;
    % func_string takes the user's input and
    % turns the input into a string
    disp("Type 'q' to quit")
    func_string = input("Please type in your functions using x:\n",'s');
    % Used to stop the program
    if func_string == 'q'
        break
    end
    % tests if the function is graphable
    % checks func_string for string 'q'
    try
        % func will be the function used for plot
        % e.g. Selection 3: 2*x will be plotted as y = 2*x
        func = str2sym(func_string);
        % indef_func will be used for printing results from function
        % e.g Selection 1: 2*x = x^2 + C1
        indef_func = matlabFunction(func);
    % if not, then returns back to top and prints warning
    catch
       warning("Function %s is invalid. Please try again", func_string)
        continue
    end
    % 4 fprintf statements below display options for user to input
    fprintf("Option 1: Indefinite integral\n")
    fprintf("Option 2: Definite integral\n")
    fprintf("Option 3: Graph of function\n")
    fprintf("Option 4: Change function\n")
    % loops until user inputs a number from vector below.
    while selection ~= [1,2,3,4]
        selection = input("Input number for desired option: ")
% Returns indefinite integration of function
        if selection == 1
            zint = dsolve(Dy == indef_func)

% Returns value of definite integral from a to b
        elseif selection == 2
            a = input('Input the lower bound for your integral: ')
            b = input('Input the upper bound for your integral: ')
            area = integral(indef_func,a,b)

% Graphs function with domain [x1.x2]
        elseif selection == 3
           x1 = input('Input left domain of the graph: ')
           x2 = input('Input right domain of the graph: ')

           %opens up a new window for every graph
           figure
           hold on
           fplot(func,[x1 x2],'-r')
           f = @(x)0*x;
           fplot(f,[x1 x2],'-k')
           xlabel('x-axis')
           ylabel('y-axis')
           title(['Graph of function ',func_string,' with domain from ',int2str(x1), ...
               ' to ',int2str(x2)])
           hold off

            %Breaks selection loop and rerturns to inputting function
        elseif selection == 4
            break

            %Returns back to inputting selection
        else
            fprintf("Number %d is not an option\n", selection)
        end
    end
end
```