

# USER MANUAL - FINAL PROJECT

## **Numerical Analysis**

---

JUAN DIEGO GUTIERREZ MONTOYA

JAIRO ALONSO CARVAJAL OCHOA

KATHERIN VALENCIA CORREA

SANTIAGO HINCAPIE MURILLO

**2020**

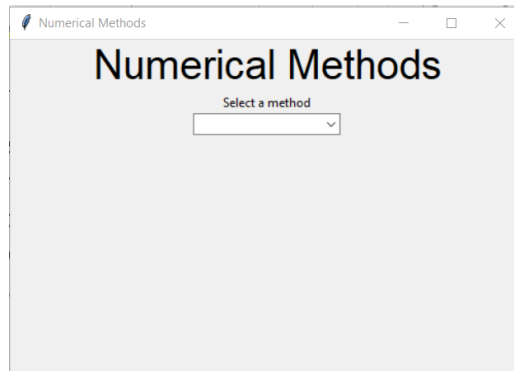
# Contenido

INTRODUCTION .....	3
METHODS.....	4
SINGLE-VARIABLE EQUATIONS.....	4
GENERAL CONSIDERATIONS.....	4
BISECTION .....	5
FALSE POSITION .....	5
FIXED-POINT ITERATION.....	6
NEWTON .....	6
SECANT .....	7
MULTIPLE ROOTS.....	7
EQUATION SYSTEMS.....	8
GENERAL CONSIDERATIONS.....	8
GAUSSIAN ELIMINATION .....	8
PARTIAL PIVOTING .....	8
LU FACTORIZATION.....	9
CHOLESKY.....	9
CROUT .....	10
DOOLITTLE .....	10
JACOBI.....	11
GAUSS-SEIDEL .....	11
INTERPOLATION.....	12
GENERAL CONSIDERATIONS.....	12
VANDERMONDE .....	12
LAGRANGE .....	12
DIVIDED DIFFERENCES (NEWTON) .....	13
EXTRAS .....	13
GENERAL CONSIDERATIONS.....	13
TRAPEZOIDAL RULE .....	14
SIMPSON 1/3 .....	14
SIMPSON 3/8 .....	15

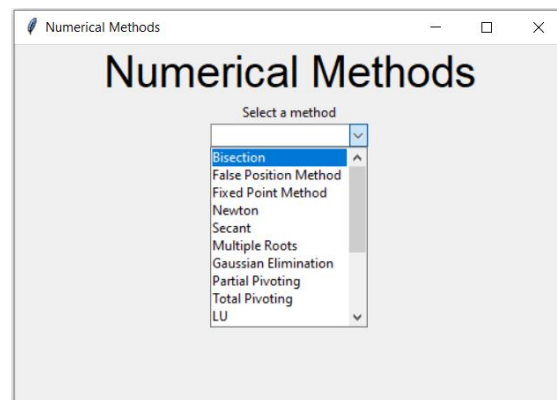
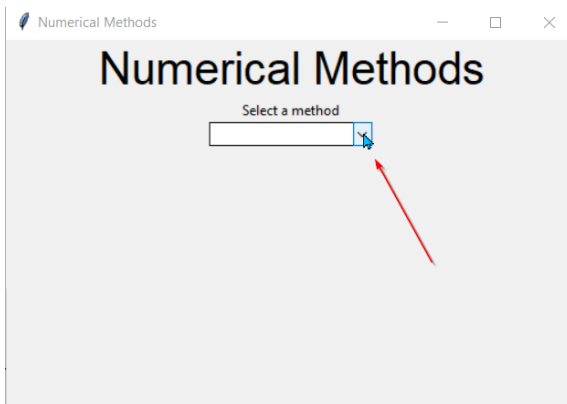
# INTRODUCTION

This user manual shows how to manage the graphical interface depending on the selected numerical method, it specifies what the user must enter for each of the methods and how to do it in order to reach the expected result. It shows the parameters to enter and the way in which to do it, such as how to write the functions or matrices in each specific case, the objective of this manual is to make the use of our application easier and more efficient.

## 1) Main view of the program.



Selecting a specific method:



## METHODS

## SINGLE-VARIABLE EQUATIONS

## GENERAL CONSIDERATIONS

All methods in single-variable equations must follow the following considerations:

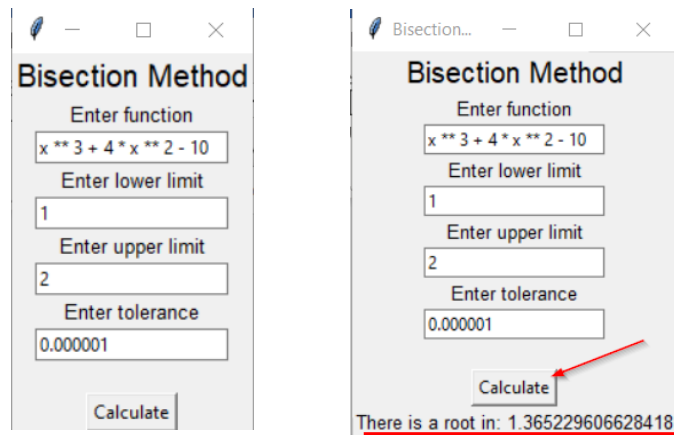
- The user can only enter the variable 'x' in the function. No other variable is allowed (a, z, g, etc.).
- If the user wants to enter an **exponential function**, he/she must use the following format:  
*np.exp(expression here)*  
Example of valid exponential function: *np.exp(-x) + 2*
- $\Pi$  (PI) must be entered as: *np.pi*
- Natural logarithm** must be entered as: *np.log(expression here)*
- Tolerances entered must be between: [0.1 , 0.00000000000000000000 ... 1]
- Use period (".") for decimal numbers
- If it is required to make use of a power within the function to be evaluated, use "\*\*\*".  
Example: *x\*\*2* for the power  $x^2$

## BISECTION

For this method, the user must enter as parameters:

- The function
- The limits of the interval (lower and upper) where the user wants to evaluate the function
- The tolerance.

Example: We want to know if a root exists within the function  $x^3 + 4x^2 - 10$ , between [1,2] with a tolerance of 0.000001



The image shows two screenshots of a software window titled "Bisection Method". The left screenshot shows the input fields: "Enter function" with the text  $x^3 + 4x^2 - 10$ , "Enter lower limit" with the value 1, "Enter upper limit" with the value 2, and "Enter tolerance" with the value 0.000001. A "Calculate" button is at the bottom. The right screenshot shows the same interface after clicking the "Calculate" button, with the output "There is a root in: 1.365229606628418" displayed at the bottom. A red arrow points to the "Calculate" button.

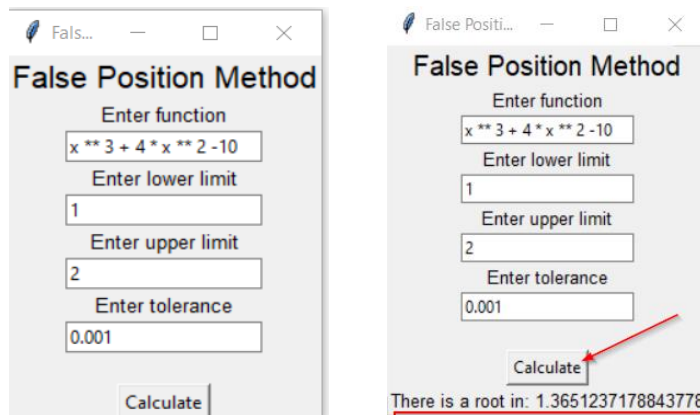
As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## FALSE POSITION

For this method, the user must enter as parameters:

- The function
- The limits of the interval (lower and upper) where the user wants to evaluate the function
- The tolerance.

Example: We want to know if a root exists within the function  $x^3 + 4x^2 - 10$ , between [1,2] with a tolerance of 0.000001



The image shows two screenshots of a software window titled "False Position Method". The left screenshot shows the input fields: "Enter function" with the text  $x^3 + 4x^2 - 10$ , "Enter lower limit" with the value 1, "Enter upper limit" with the value 2, and "Enter tolerance" with the value 0.001. A "Calculate" button is at the bottom. The right screenshot shows the same interface after clicking the "Calculate" button, with the output "There is a root in: 1.3651237178843778" displayed at the bottom. A red arrow points to the "Calculate" button.

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## FIXED-POINT ITERATION

For this method, the user must enter as parameters:

- $x = g(x)$
- The starting point: Random number
- The tolerance

Example: We want to know if a root exists within the function  $\ln(x^2 - 2x + 2)$  between  $[1,2]$  with a tolerance of 0.000001

Fixed Point Method

Enter  $x = g(x)$

`np.log(x ** 2 - 2 * x + 2)`

Enter starting point

1

Enter tolerance

0.0001

Calculate

There is a root in: 0.35121863316287405

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## NEWTON

For this method, the user must enter as parameters:

- The function
- Function derivative
- Starting point
- The tolerance

Newton Method

Enter function

`x ** 3 + 4 * x ** 2 - 10`

Enter function derivative

`3 * x ** 2 + 8 * x`

Enter starting point

1

Enter tolerance

0.000001

Calculate

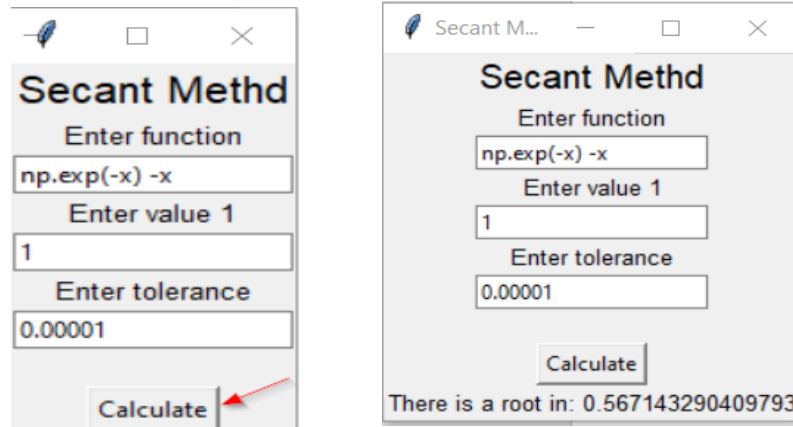
There is a root in: 1.3652300134140969

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## SECANT

For this method, the user must enter as parameters:

- The function
- Value 1
- Tolerance



Secant Method

Enter function  
`np.exp(-x) -x`

Enter value 1  
1

Enter tolerance  
0.00001

Calculate

There is a root in: 0.567143290409793

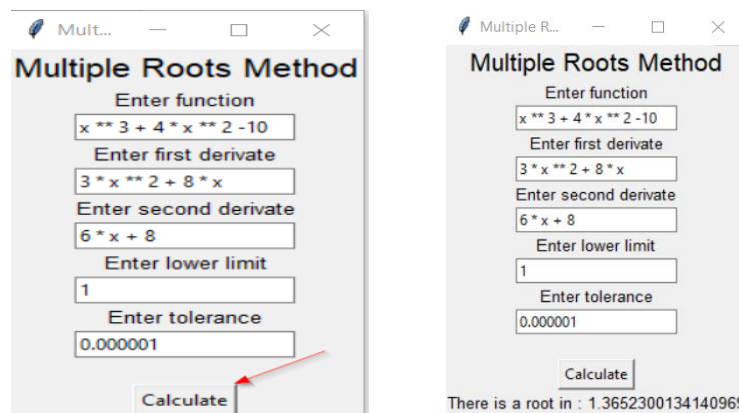
As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

*Note: The original method receives two values, but in our implementation only the first one is entered. The second value is calculated by adding the first value plus 4 times the tolerance.*

## MULTIPLE ROOTS

For this method, the user must enter as parameters:

- The function
- First function derivate
- Second function derivate
- Lower limit
- Tolerance



Multiple Roots Method

Enter function  
`x ** 3 + 4 * x ** 2 - 10`

Enter first derivate  
`3 * x ** 2 + 8 * x`

Enter second derivate  
`6 * x + 8`

Enter lower limit  
1

Enter tolerance  
0.000001

Calculate

There is a root in : 1.3652300134140969

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## EQUATION SYSTEMS

### GENERAL CONSIDERATIONS

All methods of equation systems must follow the following considerations:

For Partial Pivoting, Jacobi and Gauss-Seidel Method:

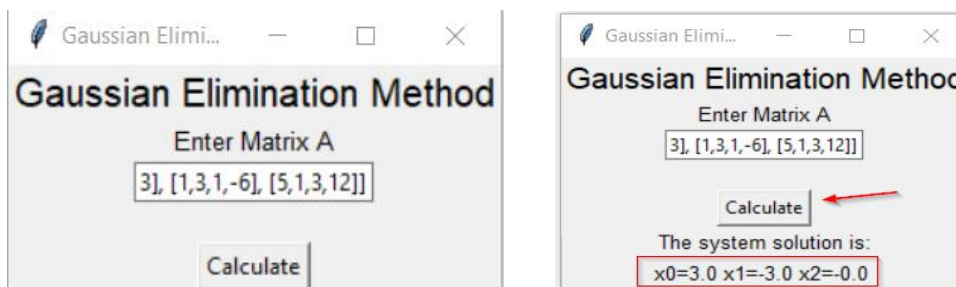
- Same consideration as Gaussian Elimination Method

### GAUSSIAN ELIMINATION

- In this method the diagonal cannot contain zeros (0) (Just in this method)
- The matrix is delimited by brackets [ ].
- Rows format is: [ # . , # . , # . ] inside of the matrix brackets.
  - Example of 1 row: [ [ 1., 2., 3.] ]
  - Example of 2 rows: [ [ 1., 2., 3.] , [ 4. , 5. , 6.] ]

For this method, the user must enter as parameters:

- The matrix A: The matrix must be Ab matrix. (Matrix and coefficients)  
Example here: [ [ 3, 2, 3, 3], [ 1, 3, 1, -6], [ 5, 1, 3, 12] ]

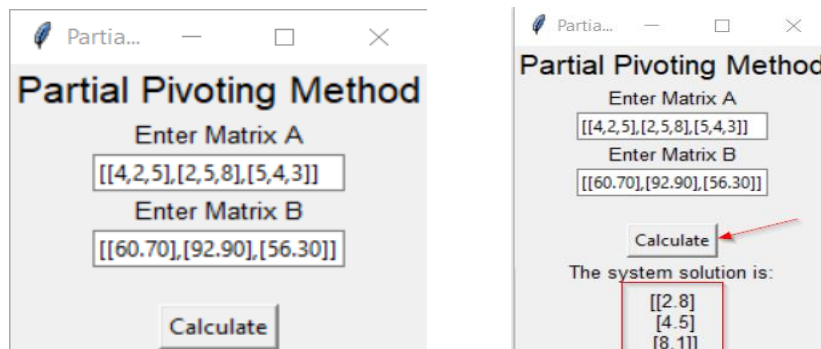


As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

### PARTIAL PIVOTING

For this method, the user must enter as parameters:

- Matrix A: Square Matrix
- Matrix B: Coefficient matrix



As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.



## LU FACTORIZATION

For this method, the user must enter as parameters:

- Augmented matrix: The system's augmented matrix includes the independent terms, use commas to separate rows and semicolons to separate columns.

Example: If you want to enter this matrix augmented,

$$\begin{pmatrix} 7 & 3 & -1 & 2 \\ 3 & 8 & 1 & -4 \\ -1 & 1 & 4 & -1 \\ 2 & -4 & -1 & 6 \end{pmatrix}$$

it must be entered like this 7,3,-1,2;3,8,1,-4;-1,1,4,-1;2,-4,-1,6

LU Gaussian Factorization Method

Enter Augmented Matrix

4,-1, 1, 4, -1;2, -4, -1, 6

Calculate

LU Gaussian Factorization Method

Enter Augmented Matrix

4,-1, 1, 4, -1;2, -4, -1, 6

Calculate

Solution X:

[-0.9798161292585468, 1.1803090206366118, -0.23068077865207284, 1.6864096749811024]

Solution Z:

[ 0.28571429 -0.59574468 -0.28143713 3.18095238]

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## CHOLESKY

For this method, the user must enter as parameters:

- Matrix A: Square Matrix
- Matrix B: Coefficient matrix
- Value of n: Size of the square matrix

*Note: Use commas to separate rows and semicolons to separate columns.*

Cholesky Factorization Method

Enter Matrix A

7, 3, -1;3, 8, 1;-1, 1, 4

Enter the vector b of independent terms

2,-4,-1

Enter value of n

3

Calculate

Cholesky Factorization Method

Enter Matrix A

7, 3, -1;3, 8, 1;-1, 1, 4

Enter the vector b of independent terms

2,-4,-1

Enter value of n

3

Calculate

The solution to factoring is:

0.6167664670658681 -0.7425149700598802 0.08982035928143708

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## CROUT

For this method, the user must enter as parameters:

- Matrix A: Square Matrix
- Matrix B: Coefficient matrix
- Value of n: Size of the square matrix

*Note: Use commas to separate rows and semicolons to separate columns.*

### Crout Factorization Method

Enter Matrix A

Enter the vector b of independent terms

Enter value of n

### Crout Factorization Method

Enter Matrix A

Enter the vector b of independent terms

Enter value of n

The solution to factoring is:

0.6167664670658682 -0.7425149700598802 0.0898203592814371

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## DOOLITTLE

For this method, the user must enter as parameters:

- Matrix A: Square Matrix
- Matrix B: Coefficient matrix
- Value of n: Size of the square matrix

*Note: Use commas to separate rows and semicolons to separate columns.*

### Doolittle Factorization Method

Enter Matrix A

Enter the vector b of independent terms

Enter value of n

### Doolittle Factorization Method

Enter Matrix A

Enter the vector b of independent terms

Enter value of n

The solution to factoring is:

0.6167664670658682 -0.7425149700598802 0.0898203592814371

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## JACOBI

For this method, the user must enter as parameters:

- Matrix A: Square Matrix
- Matrix B: Coefficient Matrix

Jacobi Method

Enter Matrix A

Enter Matrix B

Jacobi Method

Enter Matrix A

Enter Matrix B

The system solution is:

Error:

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## GAUSS-SEIDEL

For this method, the user must enter as parameters:

- Matrix A: square matrix
- Matrix B: Coefficient matrix
- Initial vector: Whatever number
- The tolerance

Gauss Seidel Method

Enter Matrix A

Enter Matrix B

Enter initial vector

Enter tolerance

Gauss Seidel Method

Enter Matrix A

Enter Matrix B

Enter initial vector

Enter tolerance

The system solution is:

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## INTERPOLATION

### GENERAL CONSIDERATIONS

All methods of equation systems must follow the following considerations:

- Use period (".") for decimal numbers
- The values of  $x$  and  $fx$  must be entered separated by commas

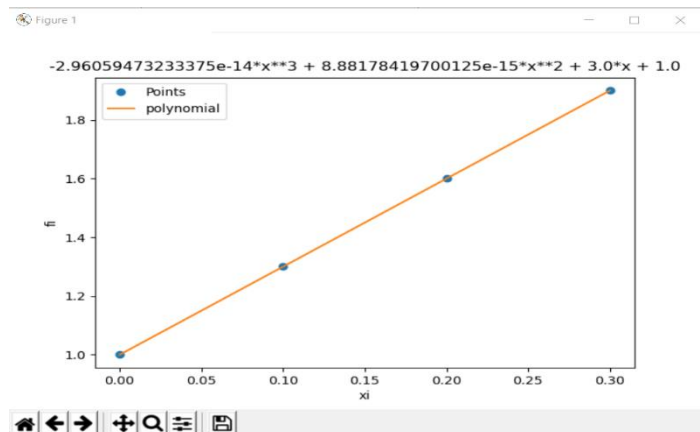
### VANDERMONDE

For this method, the user must enter as parameters:

- Values of  $x$ :
- Values of  $fx$ :



A screenshot of a software window titled "Vandermonde". It contains two input fields: "Enter the values of x" with the text "0,0.1,0.2,0.3" and "Enter the values of fx" with the text "1,1.3,1.6,1.9". Below these fields is a "Calculate" button, which is highlighted with a red arrow.

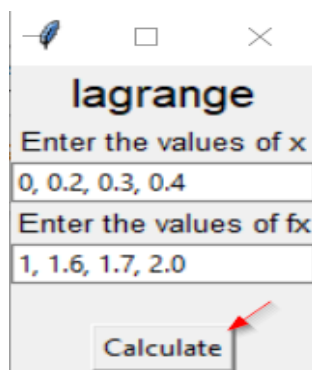


As you saw before, after clicking the button "Calculate", if all the values are correct, the output is shown. (Graphic output)

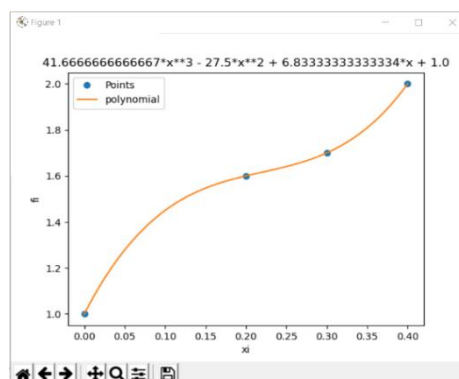
### LAGRANGE

For this method, the user must enter as parameters:

- Values of  $x$ :
- Values of  $fx$ :



A screenshot of a software window titled "lagrange". It contains two input fields: "Enter the values of x" with the text "0, 0.2, 0.3, 0.4" and "Enter the values of fx" with the text "1, 1.6, 1.7, 2.0". Below these fields is a "Calculate" button, which is highlighted with a red arrow.







As you saw before, after clicking the button "Calculate", if all the values are correct, the output is shown. (Graphic output)

## DIVIDED DIFFERENCES (NEWTON)

For this method, the user must enter as parameters:

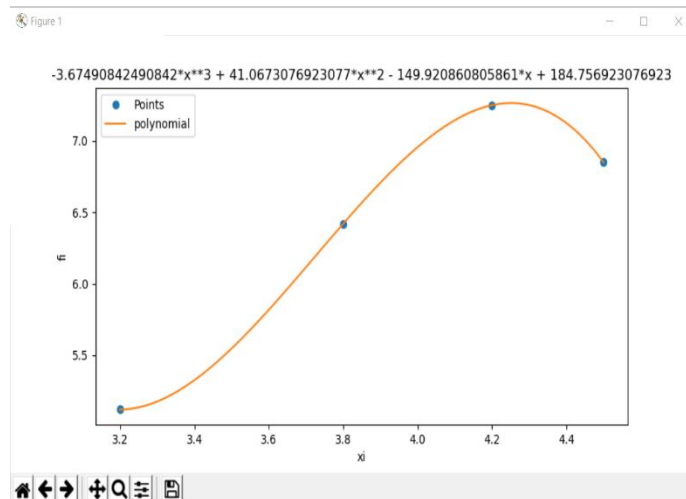
- Values of  $x$ :
- Values of  $fx$ :

 |   

## Divided differences

Enter the values of x

Enter the values of fx



As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## EXTRAS

## GENERAL CONSIDERATIONS

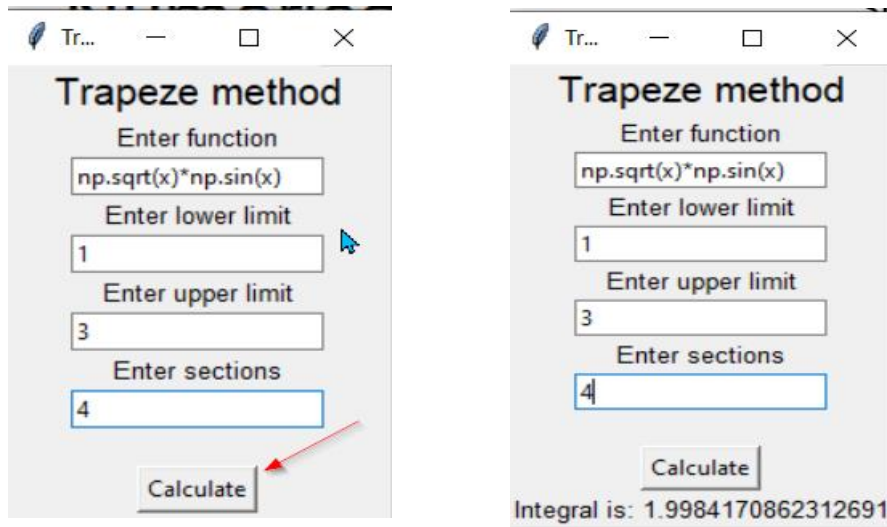
All methods in extras must follow the following considerations:

- The user can only enter the variable 'x' in the function. No other variable is allowed (a, z, g, etc.).
- If the user wants to enter an **exponential function**, he/she must use the following format:  
*np.exp(expression here)*  
Example of valid exponential function: *np.exp(-x) + 2*
- $\Pi$  (PI) must be entered as: *np.pi*
- Natural logarithm** must be entered as: *np.log(expression here)*
- Tolerances entered must be between: [0.1 , 0.00000000000000000000 ... 1]
- Use period ("." ) for decimal numbers
- If it is required to make use of a power within the function to be evaluated, use "\*\*\*".  
Example: *x\*\*2* for the power  $x^2$

## TRAPEZOIDAL RULE

For this method, the user must enter as parameters:

- The function
- Lower and upper limit
- Sections



**Trapeze method**

Enter function

Enter lower limit

Enter upper limit

Enter sections

**Trapeze method**

Enter function

Enter lower limit

Enter upper limit

Enter sections

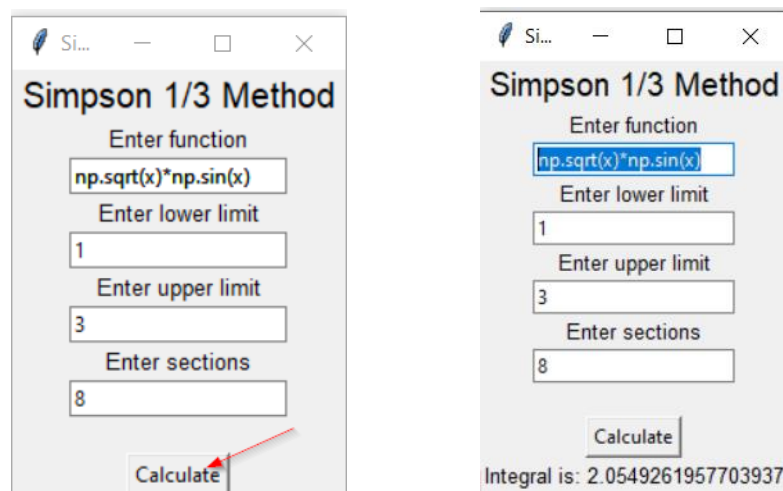
Integral is: 1.9984170862312691

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## SIMPSON 1/3

For this method, the user must enter as parameters:

- The function
- Lower and upper limit
- Sections: Must be pair



**Simpson 1/3 Method**

Enter function

Enter lower limit

Enter upper limit

Enter sections

**Simpson 1/3 Method**

Enter function

Enter lower limit

Enter upper limit

Enter sections

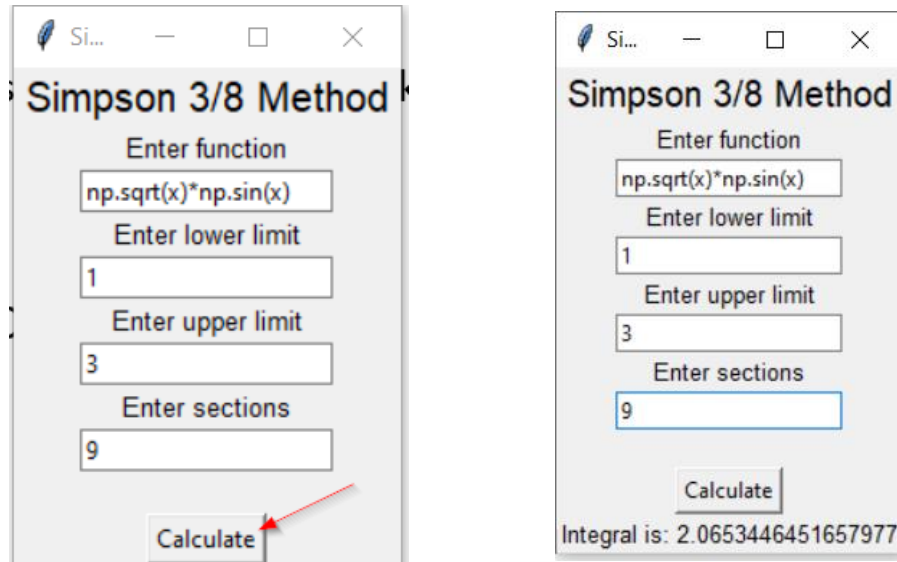
Integral is: 2.0549261957703937

As you saw before, after clicking the button “Calculate”, if all the values are correct, the output is shown.

## SIMPSON 3/8

For this method, the user must enter as parameters:

- The function
- Lower and upper limit
- Sections: Must be an odd number multiple of 3



The image displays two screenshots of a software interface titled "Simpson 3/8 Method".

The left screenshot shows the input fields:

- Enter function: `np.sqrt(x)*np.sin(x)`
- Enter lower limit: `1`
- Enter upper limit: `3`
- Enter sections: `9`
- A red arrow points to the "Calculate" button.

The right screenshot shows the same interface after calculation:

- Enter function: `np.sqrt(x)*np.sin(x)`
- Enter lower limit: `1`
- Enter upper limit: `3`
- Enter sections: `9`
- The "Calculate" button is highlighted.
- The output is displayed: "Integral is: 2.0653446451657977".

As you saw before, after clicking the button "Calculate", if all the values are correct, the output is shown.