

Numerical Analysis User manual

Maria Sofía Uribe,Santiago Tello,Luis Herrera Chamat,Andrés de Jesús Martinez

November 20, 2020

Contents

1	Repository Address	1
2	Members	1
3	User Manual	1
3.1	Requirements to run the program	1
3.2	Interacting with the main window	2
3.3	Integration methods	3
3.4	Interpolation	6
3.5	Systems of equations	8
3.5.1	Direct methods	8
3.5.2	Iterative methods	11
3.6	Equations of one variable	14
3.6.1	Bracketing methods	14
3.6.2	Open methods	15

1 Repository Address

<https://github.com/msuribec/AnalisisNumerico>

2 Members

- María Sofía Uribe
- Santiago Tello
- Luis Herrera Chamat
- Andrés de Jesús Martinez

3 User Manual

3.1 Requirements to run the program

The interactive graphical user interface for the app was developed in Python 3. It is necessary that python 3 is installed since the division operator behaves differently in previous versions of Python.

Some additional libraries needed are:

- tkinter (some distributions of Python 3 will have this already installed depending on the Operating system that is being used)
- matplotlib
- sympy
- numpy

3.2 Interacting with the main window

The main menu of the app can be accessed by running the **NumericalAnalysisGUI.py** module in the python project folder.

This will prompt the main menu. The main menu has 6 buttons, one for each type of method. The user must choose the type of method they want to try.

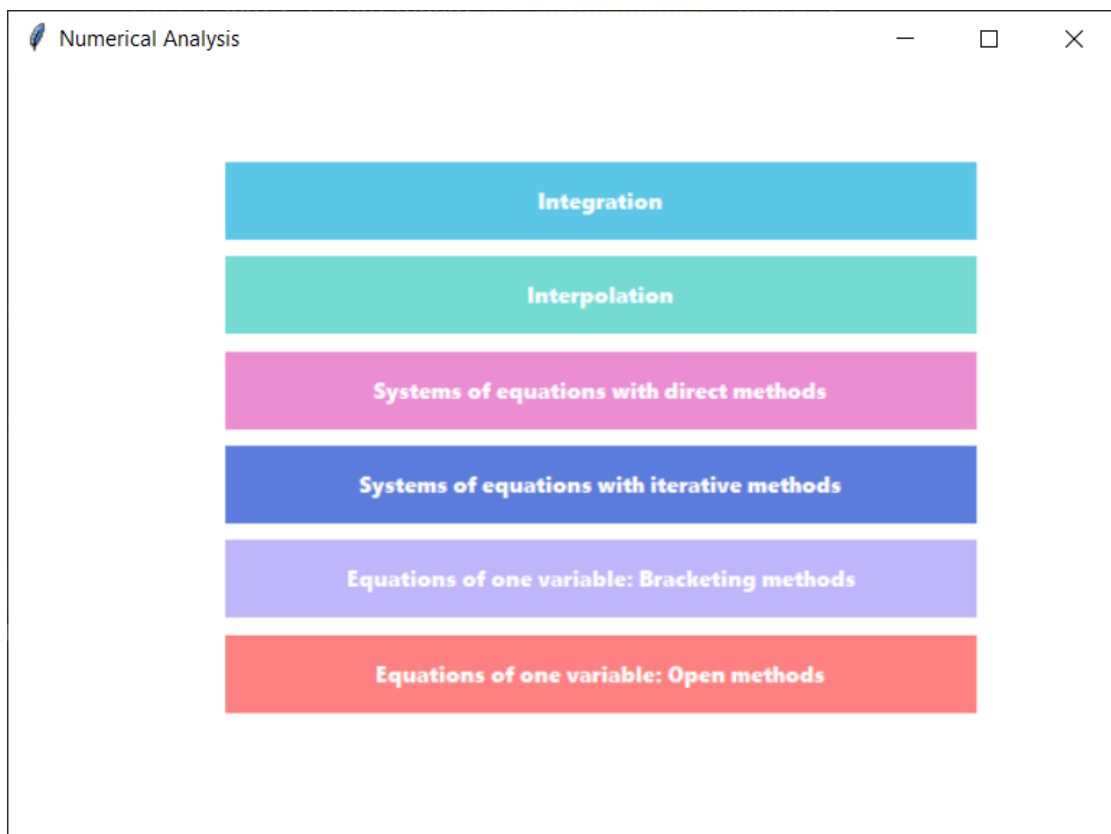


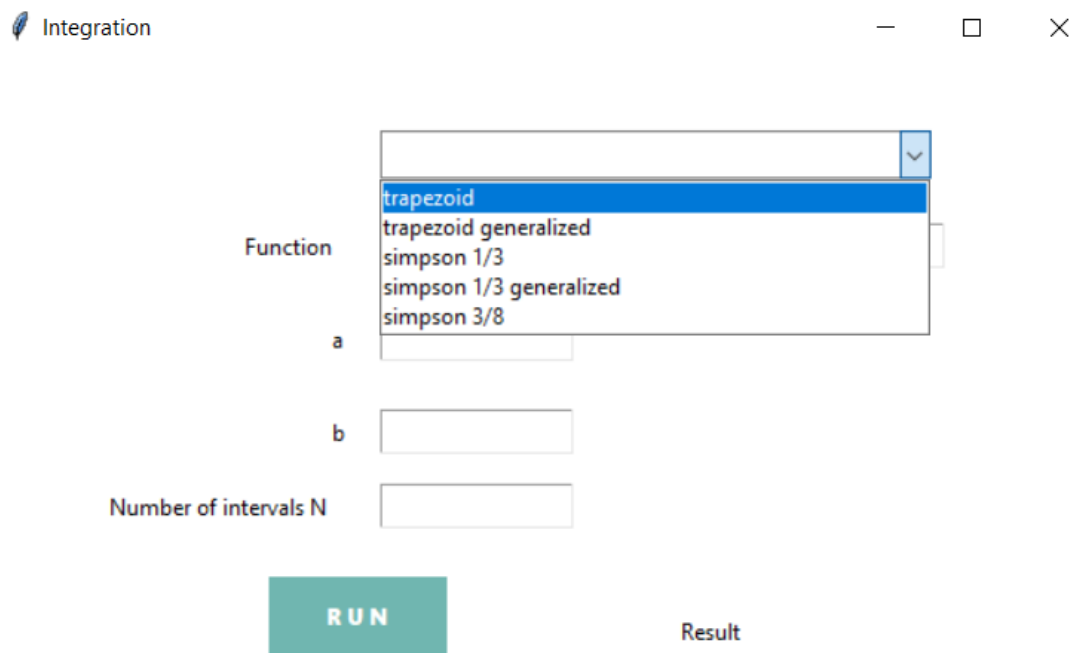
Figure 1: Main menu

3.3 Integration methods

Clicking the Integration button on the main menu will prompt the window for the integration methods. The user must choose the name of the method they wish to run. The options are trapezoid method, generalized trapezoid method, simpson 1/3 method, generalized simpson 1/3 and simpson 3/8.

For each method the user must input a function and the limits of integration (a is the lower limit and b is the upper limit)

If the chosen method is one of the generalized methods, the user must choose the number of intervals (must be an integer and in the case of generalized simpson 1/3 it must be even).



The screenshot shows a window titled "Integration" with standard window controls (minimize, maximize, close). Inside the window, there is a dropdown menu for selecting an integration method. The dropdown is open, showing the following options: "trapezoid", "trapezoid generalized", "simpson 1/3", "simpson 1/3 generalized", and "simpson 3/8". Below the dropdown, there are three input fields: "a", "b", and "Number of intervals N". To the left of the "a" and "b" fields is the label "Function". Below the input fields is a green button labeled "RUN". To the right of the "RUN" button is the label "Result".

Figure 2: Window for integration methods

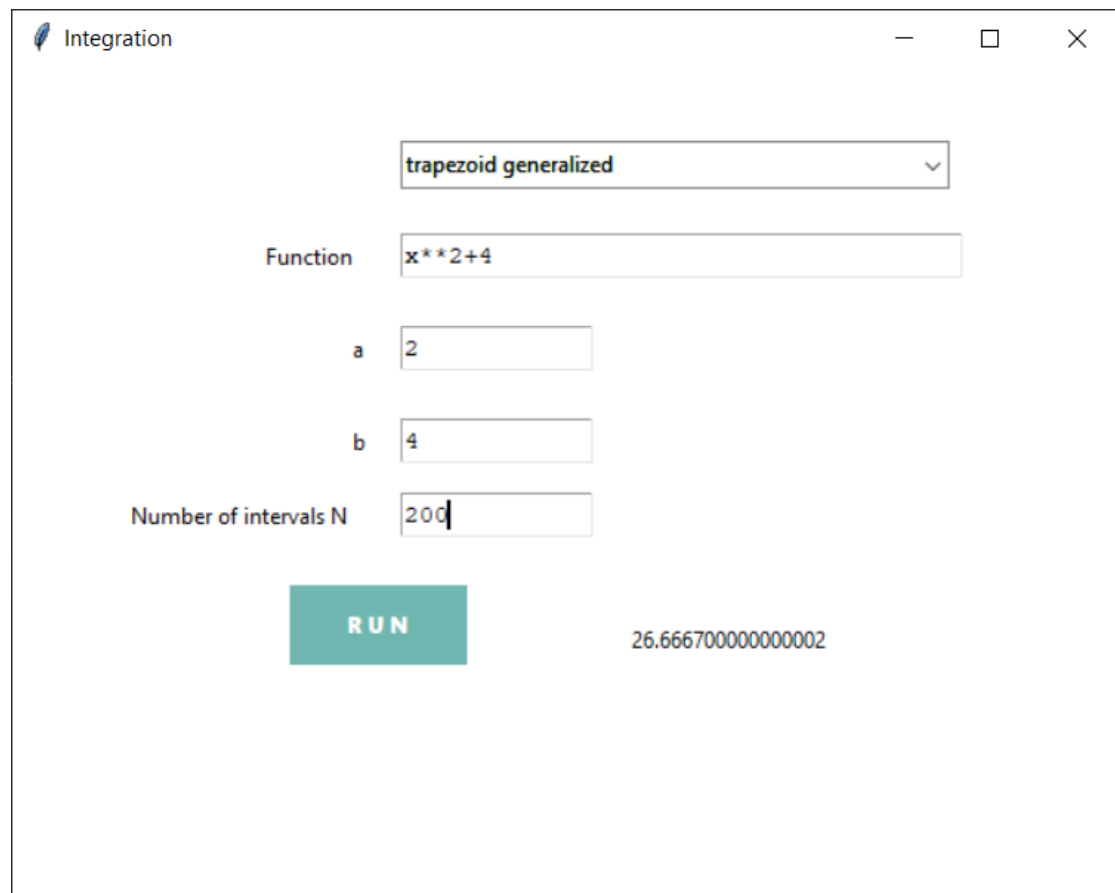
Examples of functions are

- e^x is written as `exp(x)`
- $x^2 - 4$ is written as `x**2 -4`

The screenshot shows a window titled "Integration" with the following elements:

- A dropdown menu at the top set to "trapezoid".
- A text input field labeled "Function" containing the expression `x**2+4`.
- Two input fields for the interval: "a" with the value 2, and "b" with the value 4.
- An input field labeled "Number of intervals N" which is currently empty.
- A green button labeled "RUN".
- The numerical result "28.0" displayed to the right of the "RUN" button.

Figure 3: Example of input for the trapezoid method, the function is $f(x) = x^2 - 4$



Integration

trapezoid generalized

Function `x**2+4`

a 2

b 4

Number of intervals N 200

RUN

26.666700000000002

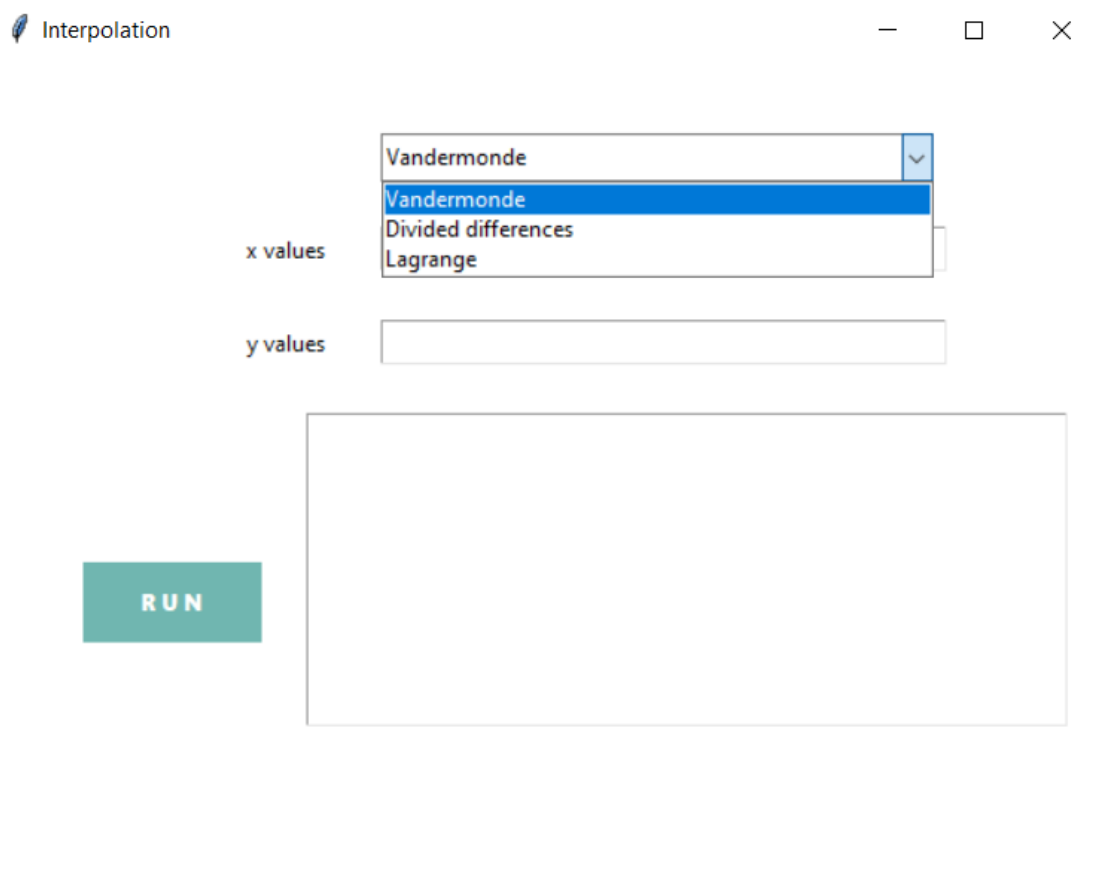
Figure 4: Example of input for the generalized trapezoid method, the function is $f(x) = x^2 - 4$

3.4 Interpolation

Clicking the Interpolation button on the main menu will prompt the window for the interpolation methods. The user must choose the name of the method they wish to run. The options are the Vandermonde method, the divided differences method and the Lagrange method.

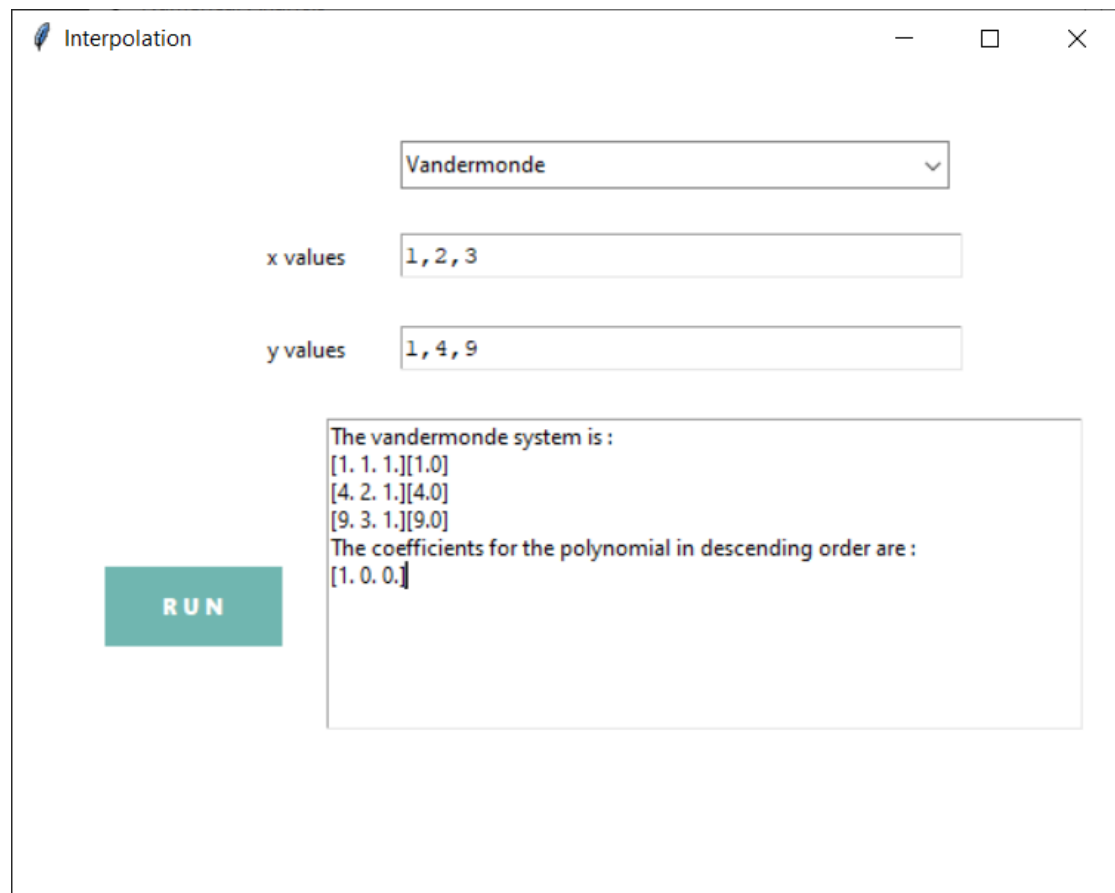
For each method the user must input the x values and the y values separated by a comma. e

For example, to interpolate the points (1, 1), (2, 4), (3, 9) the x values must be entered as 1, 2, 3 and the y values as 1, 4, 9



The screenshot shows a window titled "Interpolation" with standard window controls (minimize, maximize, close). Inside the window, there is a dropdown menu for selecting an interpolation method. The dropdown is open, showing three options: "Vandermonde" (selected), "Divided differences", and "Lagrange". To the left of the dropdown is the label "x values". Below the dropdown is a text input field for "y values". To the left of the input fields is a green button labeled "RUN". Below the input fields is a large, empty rectangular box for the output.

Figure 5: Window for interpolation methods



The screenshot shows a window titled "Interpolation" with standard window controls (minimize, maximize, close) in the top right corner. Inside the window, there is a dropdown menu set to "Vandermonde". Below this, there are two input fields: "x values" containing "1, 2, 3" and "y values" containing "1, 4, 9". To the left of a large text area is a green button labeled "RUN". The text area contains the following output:

```
The vandermonde system is :  
[1. 1. 1.][1.0]  
[4. 2. 1.][4.0]  
[9. 3. 1.][9.0]  
The coefficients for the polynomial in descending order are :  
[1. 0. 0.]
```

Figure 6: Example of input for the Vandermonde method

3.5 Systems of equations

3.5.1 Direct methods

Clicking the Systems of equations with direct methods button on the main menu will prompt the window for the Systems of equations with direct methods. The user must choose the name of the method they wish to run. The options are Gaussian elimination, Gaussian elimination with partial pivoting, Gaussian elimination with total pivoting, LU factorization with Gaussian elimination, LU factorization with Gaussian elimination and partial pivoting, Cholesky factorization, Crout factorization and Doolittle factorization

For each method the user must input the coefficient matrix A (all elements separated by a comma) and the vector of independent terms b to solve the system $Ax = b$ (all elements separated by a comma)

For example, the matrix

$$A = \begin{bmatrix} 129 & 163 & 70 \\ 195 & 142 & 156 \\ 357 & 300 & 276 \end{bmatrix}$$

must be entered as

129, 163, 70, 195, 142, 156, 357, 300, 276

and the vector

$$b = \begin{bmatrix} 10 \\ 33 \\ 43 \end{bmatrix}$$

must be entered as

10, 33, 43

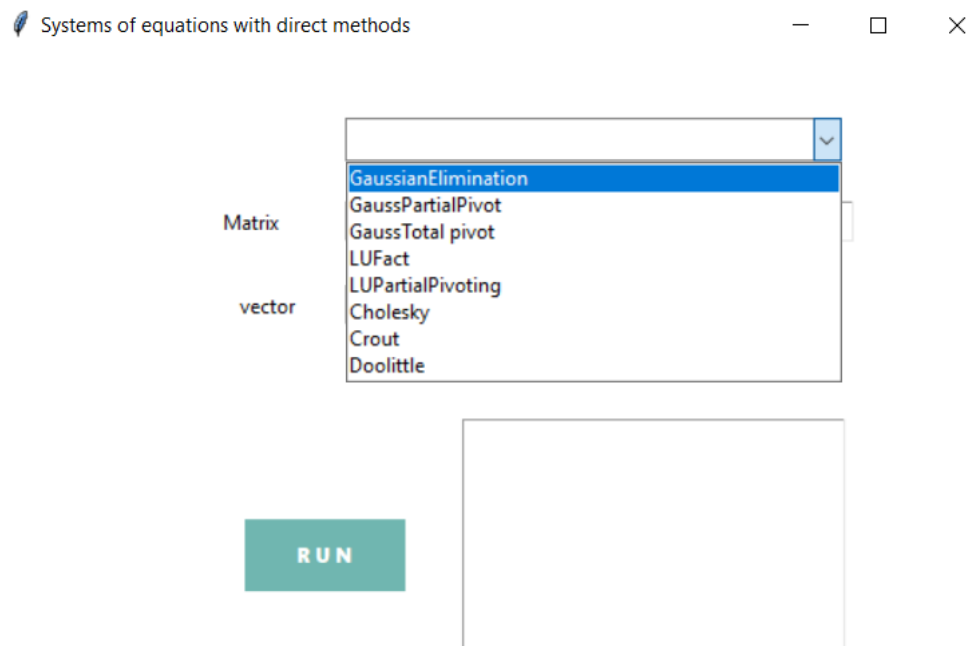


Figure 7: Window for Systems of equations with direct methods

Systems of equations with direct methods

Cholesky

Matrix

vector

RUN

[0. 0. 1.15470054]]
solving $Lz=b$ by forward substitution, w
 $y = [0.70710678 \ 2.04124145 \ 4.90747729]$
Solving $Ux=z$ by backward substitution,
 $x = [2.75 \ 4.5 \ 4.25]$

Figure 8: Example of input for the Cholesky method

3.5.2 Iterative methods

Clicking the Systems of equations with iterative methods button on the main menu will prompt the window for the Systems of equations with iterative methods. The user must choose the name of the method they wish to run. The options are Jacobi and Gauss Seidel.

For each method the user must input the coefficient matrix A (all elements separated by a comma) and the vector of independent terms b to solve the system (all elements separated by a comma) $Ax = b$. Additionally the user must enter the number of maximum iterations (must be an integer) and the tolerance (float with dot as a decimal separator).

For example, the matrix

$$A = \begin{bmatrix} 129 & 163 & 70 \\ 195 & 142 & 156 \\ 357 & 300 & 276 \end{bmatrix}$$

must be entered as

129, 163, 70, 195, 142, 156, 357, 300, 276

and the vector

$$b = \begin{bmatrix} 10 \\ 33 \\ 43 \end{bmatrix}$$

must be entered as

10, 33, 43

Iterative methods for systems of equations

Matrix A

vector b

initial solution vector

Max iterations

tolerance

RUN

Figure 9: Window for Systems of equations with iterative methods

Iterative methods for systems of equations

Matrix A:

vector b:

initial solution vector:

Max iterations:

tolerance:

	2.743164062	4.491210938	4.243164062	4.9e-03
	2.745605469	4.493164062	4.245605469	2.4e-03
	2.746582031	4.495605469	4.246582031	2.4e-03
	2.747802734	4.496582031	4.247802734	1.2e-03
	2.748291016	4.497802734	4.248291016	1.2e-03
	2.748901367	4.498291016	4.248901367	6.1e-04
	2.749145508	4.498901367	4.249145508	6.1e-04
	2.749450684	4.499145508	4.249450684	3.1e-04

The solution is
[2.74945068 4.49914551 4.24945068]
Se alcanzó una aproximación con tol 0.0005000000000000

Figure 10: Example of input for the jacobi method

3.6 Equations of one variable

Clicking the Equations of one variable equations with bracketing methods button on the main menu will prompt the window for the one variable equations with bracketing methods. The user must choose the name of the method they wish to run. The options are bisection and regula falsi.

For each method the user must input a function and the values of a and b, the number of maximum iterations (must be an integer) and the tolerance (float with dot as a decimal separator).

Examples of functions are

- e^x is written as `exp(x)`
- $x^2 - 4$ is written as `x**2 -4`

3.6.1 Bracketing methods

Bracketing methods

Function:

a:

b:

Max iterations:

Tolerance:

Figure 11: Window for bracketing methods

Bracketing methods

Method: **bisection**

Function: **exp(-x) - x**

a: **0**

b: **1**

Max iterations: **1000**

Tolerance: **0.005**

RUN

```

i: 001 xi: +0.0000000000 xm: +1.0000000000 xf: +0.5000000000 f(xi): +0.106530
i: 002 xi: +0.5000000000 xm: +0.1065306597 xf: +0.7500000000 f(xi): -0.2776334
i: 003 xi: +0.5000000000 xm: +0.1065306597 xf: +0.6250000000 f(xi): -0.0897385
i: 004 xi: +0.5000000000 xm: +0.1065306597 xf: +0.5625000000 f(xi): +0.007282
i: 005 xi: +0.5625000000 xm: +0.0072828247 xf: +0.5937500000 f(xi): -0.0414975
i: 006 xi: +0.5625000000 xm: +0.0072828247 xf: +0.5781250000 f(xi): -0.0171758
i: 007 xi: +0.5625000000 xm: +0.0072828247 xf: +0.5703125000 f(xi): -0.0049637
i: 008 xi: +0.5625000000 xm: +0.0072828247 xf: +0.5664062500 f(xi): +0.001155
i: 009 xi: +0.5664062500 xm: +0.0011552020 xf: +0.5683593750 f(xi): -0.0019053
0.568359 es aproximación a una raíz con tolerancia 0.005000

```

Figure 12: Example of input for bracketing methods

3.6.2 Open methods

Clicking the Equations of one variable equation with open methods button on the main menu will prompt the window for the one variable equation with open methods. The user must choose the name of the method they wish to run. The options are Fixed point, Newton, Secant and multiple roots. Additionally, there is an option to run the incremental searches method (this is not an open method for root finding but is included in case the user wants to use it to find an initial approximation for the other methods)


For each method the user must input a function f , the value of the initial approximation, x_0 , the number of maximum iterations (must be an integer) and the tolerance (float with dot as a decimal separator).

For the fixed point method, there is an additional function g that is needed. For the secant method an additional initial value x_1 , is needed.

These fields will only be available when the respective methods are chosen, otherwise they will be disabled to avoid confusion

Examples of functions are

- e^x is written as `exp(x)`
- $x^2 - 4$ is written as `x**2 -4`

 Open methods — □ ×

Function f

Function g

Fixed Point

Newton

Secant

Multiple roots

Incremental searches

x0

x1

Max iterations

Tolerance

RUN

Figure 13: Window for equations of one variable with open methods

Open methods

Fixed Point

Function f: $\exp(-x) - x$

Function g: $\exp(-x)$

x0: 0 x1:

Max iterations: 1000

Tolerance: 0.005

RUN

```

i:000 x: 0.0000000000 fx: 1.0000000000 error abs: 1.0050000000
i:001 x: 1.0000000000 fx: -0.6321205588 error abs: 1.0000000000
i:002 x: 0.3678794412 fx: 0.3243211864 error abs: 0.6321205588
i:003 x: 0.6922006276 fx: -0.1917271270 error abs: 0.3243211864
i:004 x: 0.5004735006 fx: 0.1057700345 error abs: 0.1917271270
i:005 x: 0.6062435351 fx: -0.0608477491 error abs: 0.1057700345
i:006 x: 0.5453957860 fx: 0.0342165495 error abs: 0.0608477491
i:007 x: 0.5796123355 fx: -0.0194968741 error abs: 0.0342165495
i:008 x: 0.5601154614 fx: 0.0110276537 error abs: 0.0194968741
i:009 x: 0.5711431151 fx: -0.0062637677 error abs: 0.0110276537
i:010 x: 0.5648793474 fx: 0.0035493776 error abs: 0.0062637677
i:011 x: 0.5684287250 fx: -0.0020139919 error abs: 0.0035493776
0.568429 Es aproximación a una raíz con una tolerancia de 0.005000

```

Figure 14: Example of input for the fixed point method

Open methods

Secant

Function f $\exp(-x) - x$

Function g $\exp(-x)$

x0 0 x1 1

Max iterations 1000

Tolerance 0.005

RUN

```
i:001 x: 1.0000000000 fx: -0.6321205588 error abs: 1.0050000000 den: -1.632120
i:002 x: 0.6126998368 fx: -0.0708139479 error abs: 0.3873001632 den: 0.5613066
i:003 x: 0.5638383892 fx: 0.0051823545 error abs: 0.0488614476 den: 0.07599630
i:004 x: 0.5671703584 fx: -0.0000424192 error abs: 0.0033319693
0.567170 es aproximación a una raíz con tolerancia 0.005000
```

Figure 15: Example of input for the secant method