- **The user is able to:**

  Enter an augmented matrix of the system to be solved.

  Choose a method to be used

  Choose a stopping criterion for an iterative method

  Enter a threshold parameter for a stopping criterion

  Enter a starting approximation for an iterative method or use a starting approximation created by default in the program.

- **The program returns:**

  The roots of the matrix

  The mean true error

The GUI of the program was made programmatically by using features such as uitextareas and uitextlabels to allow for more control over the gui elements

Example:

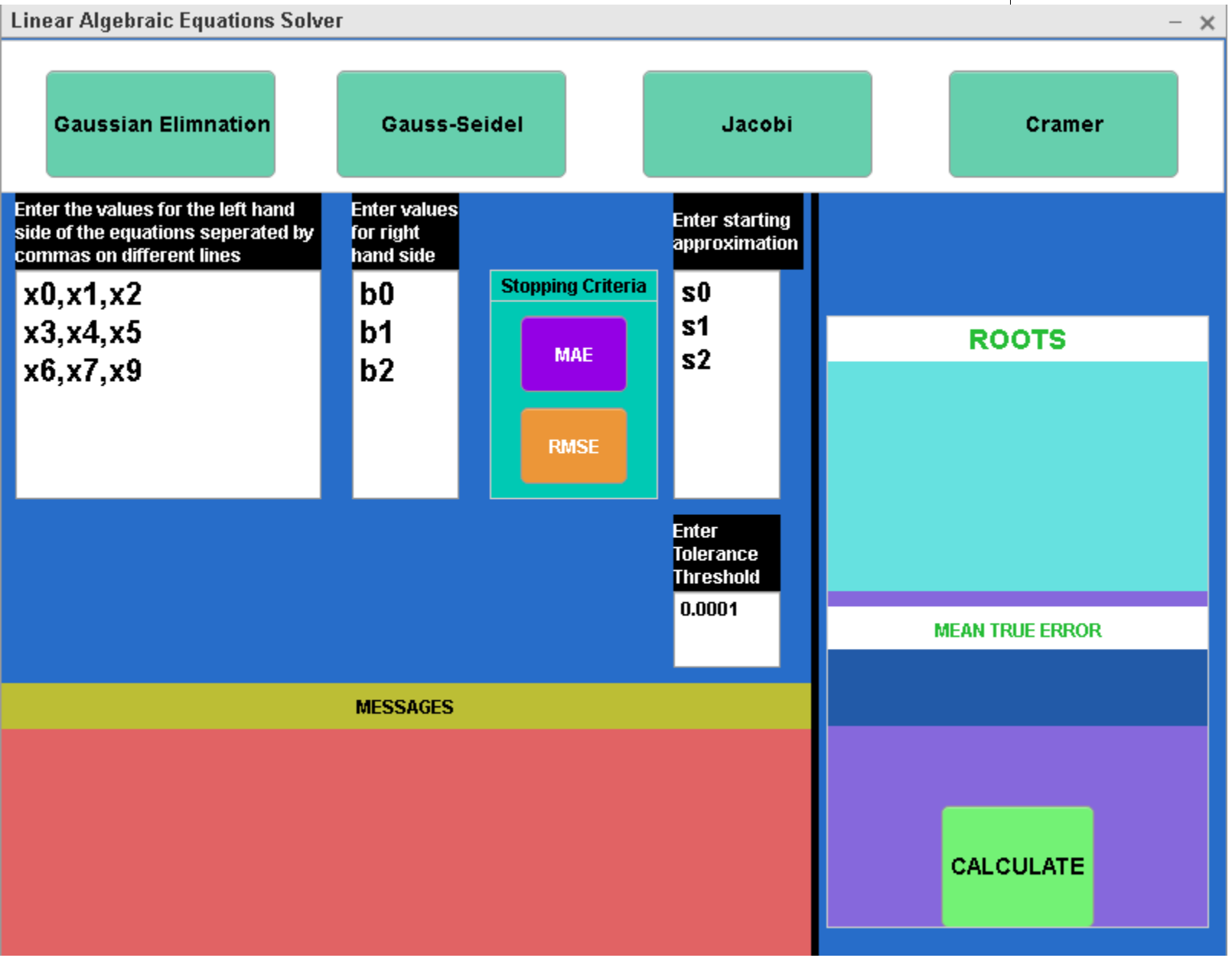
```
right_label = uilabel(f, "Position", [230 450 70 50], "Text", "Enter values for right hand side"
right_label.FontColor = "#ffffff";
right_label.FontWeight = "Bold";
right_label.WordWrap = "on";
right_label.BackgroundColor = "#000000";

right_textbox = uitextarea(f,"Position",[230 300 70 150], "Placeholder",sprintf("b0\nb1\nb2"));
right_textbox.ValueChangedFcn = @(right_textbox,event) get_matrix_right(right_textbox);
right_textbox.FontSize = 20;
right_textbox.FontWeight = "Bold";

stop_criteria = "";
stop_c_panel = uipanel(f,"Position",[320 300 110 150], "Title","Stopping Criteria", ...
    "FontWeight","bold","BackgroundColor","#21C8B3", "TitlePosition","centertop");
mae_button = uibutton(stop_c_panel, "push","InnerPosition",[20 70 70 50], ...
    "Text","MAE", "FontColor", "#ffffff", "BackgroundColor", "#8A27EB", "FontWeight", "Bold");
mae_button.ButtonPushedFcn = @(mae_button,event) s_criteria(mae_button.Text);

rmse_button = uibutton(stop_c_panel, "push","InnerPosition",[20 10 70 50], ...
    "Text","RMSE", "FontColor", "#ffffff", "BackgroundColor", "#EB9427", "FontWeight", "Bold");
rmse_button.ButtonPushedFcn = @(rsme_button,event) s_criteria(rsme_button.Text);


th_label = uilabel(f, "Position", [440 240 70 50], "Text", "Enter Tolerance Threshold");
th_label.FontColor = "#ffffff";
th_label.FontWeight = "Bold";
th_label.WordWrap = "on";
th_label.BackgroundColor = "#000";
```

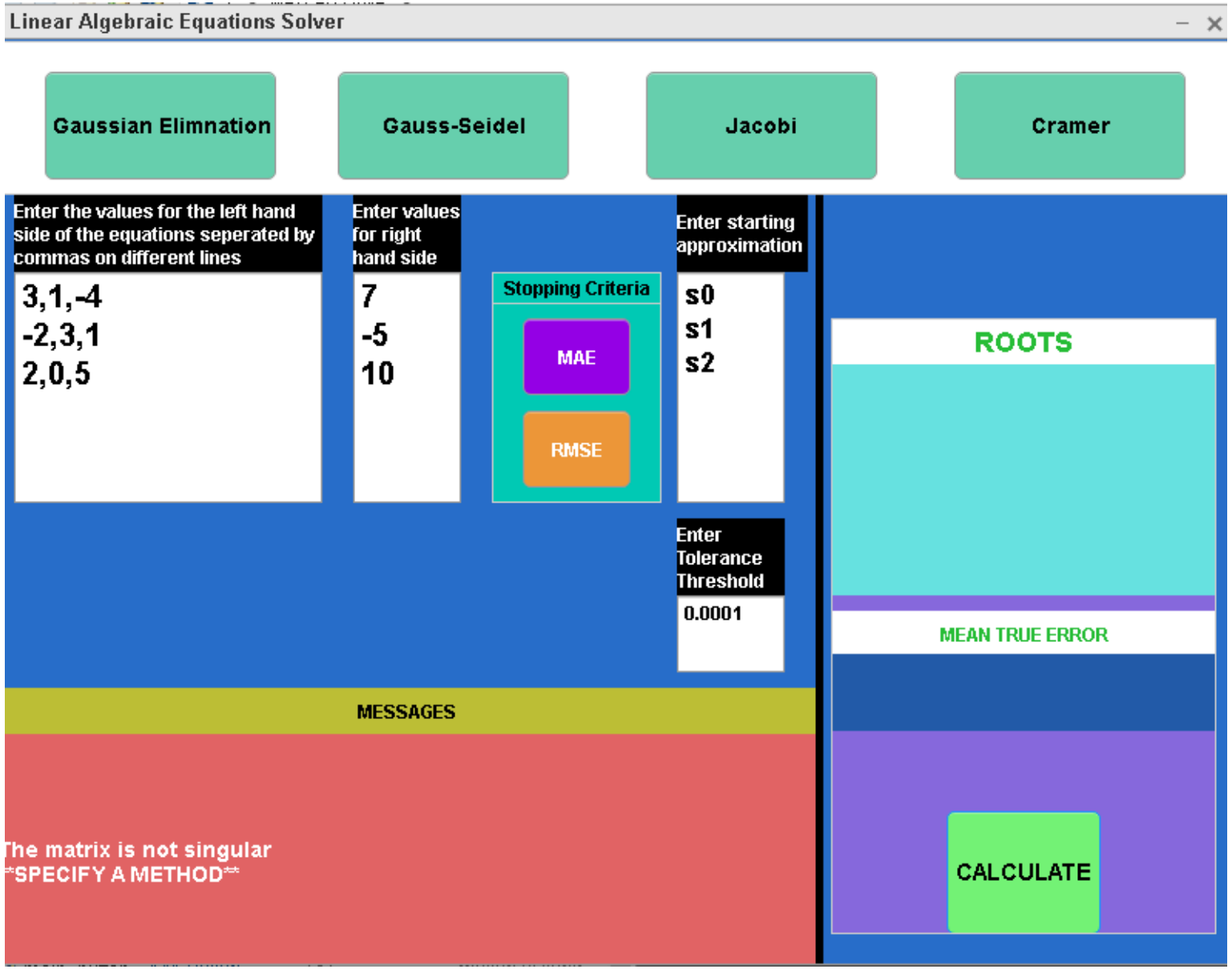


- Default look of the program. The user can choose from 4 different methods to solve the set of linear equations given. There is a message box at the bottom that will display relevant messages to the user, such as the stopping criteria that they have specified.

The stopping criteria, starting approximating and tolerance threshold can be specified when using the Gauss-Seidel or Jacobi Iterative methods.

- The user can enter the left hand side of the equations separated by commas on different lines, and the right hand side can be entered on different lines which correspond to the left hand side. The user can use the “Calculate” button to attain the results.

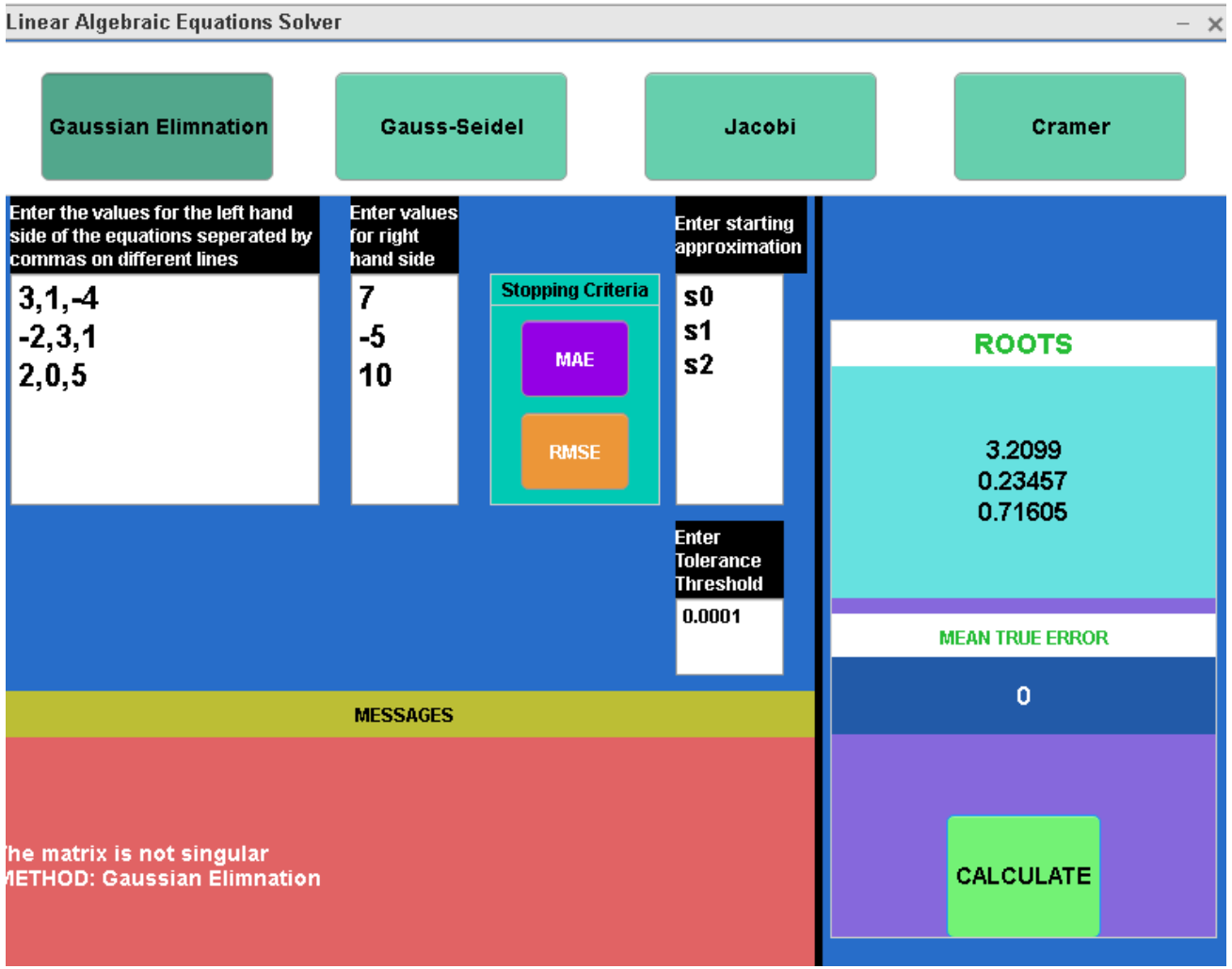
- Since no method was selected after the “Calculate” button was pressed, the messages box indicates that a method must be specified. It is also indicated that the matrix is not singular.



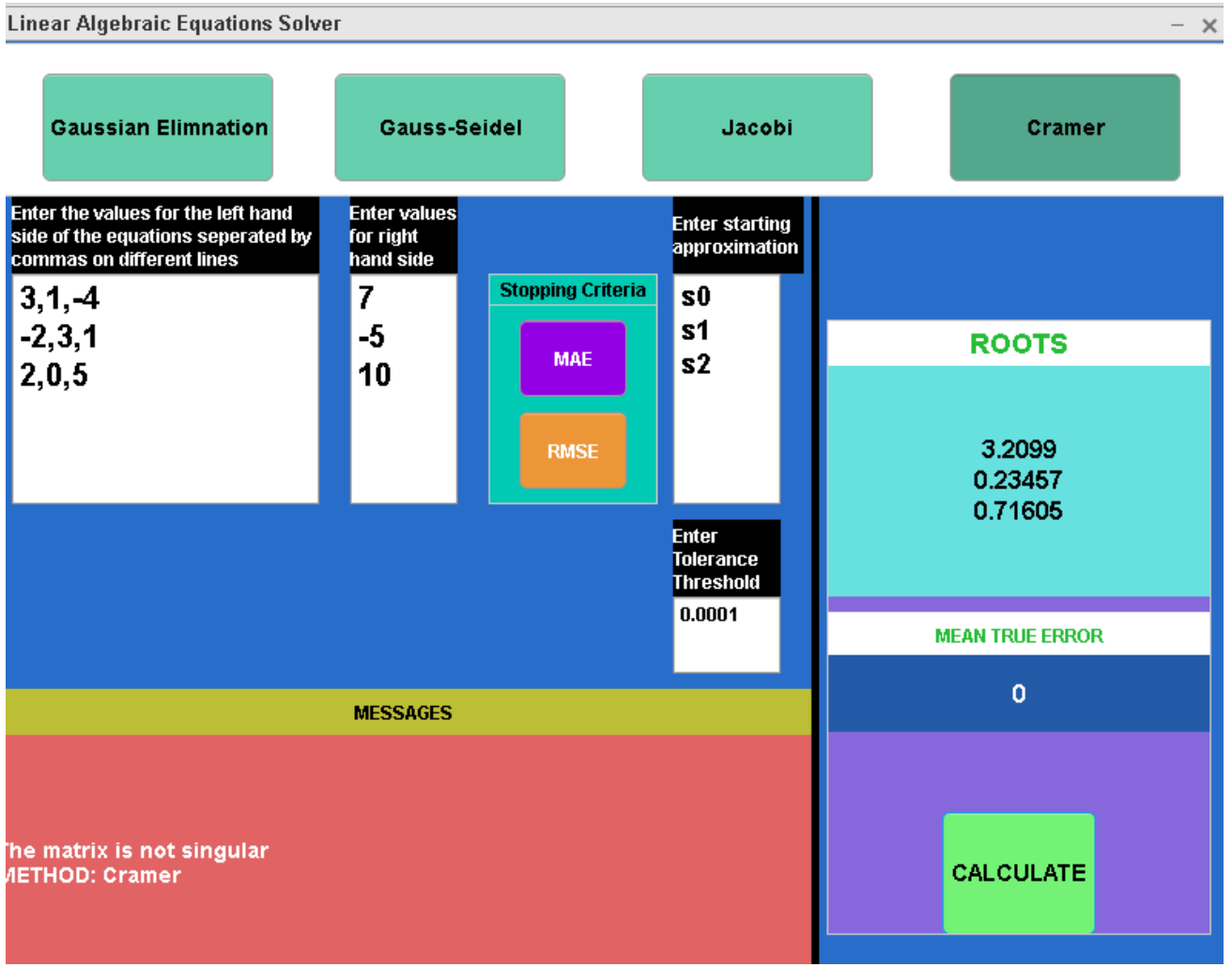
Regular usage with Gaussian or Jacobi Method

Since an iterative method was not chosen, there is no need for a stopping criteria, starting approximation or tolerance threshold to be specified.

This time, “Gaussian Elimination” was selected and the “Calculate” button was clicked again, so the roots and the mean true error are displayed to the right and the messages box shows what method was used.

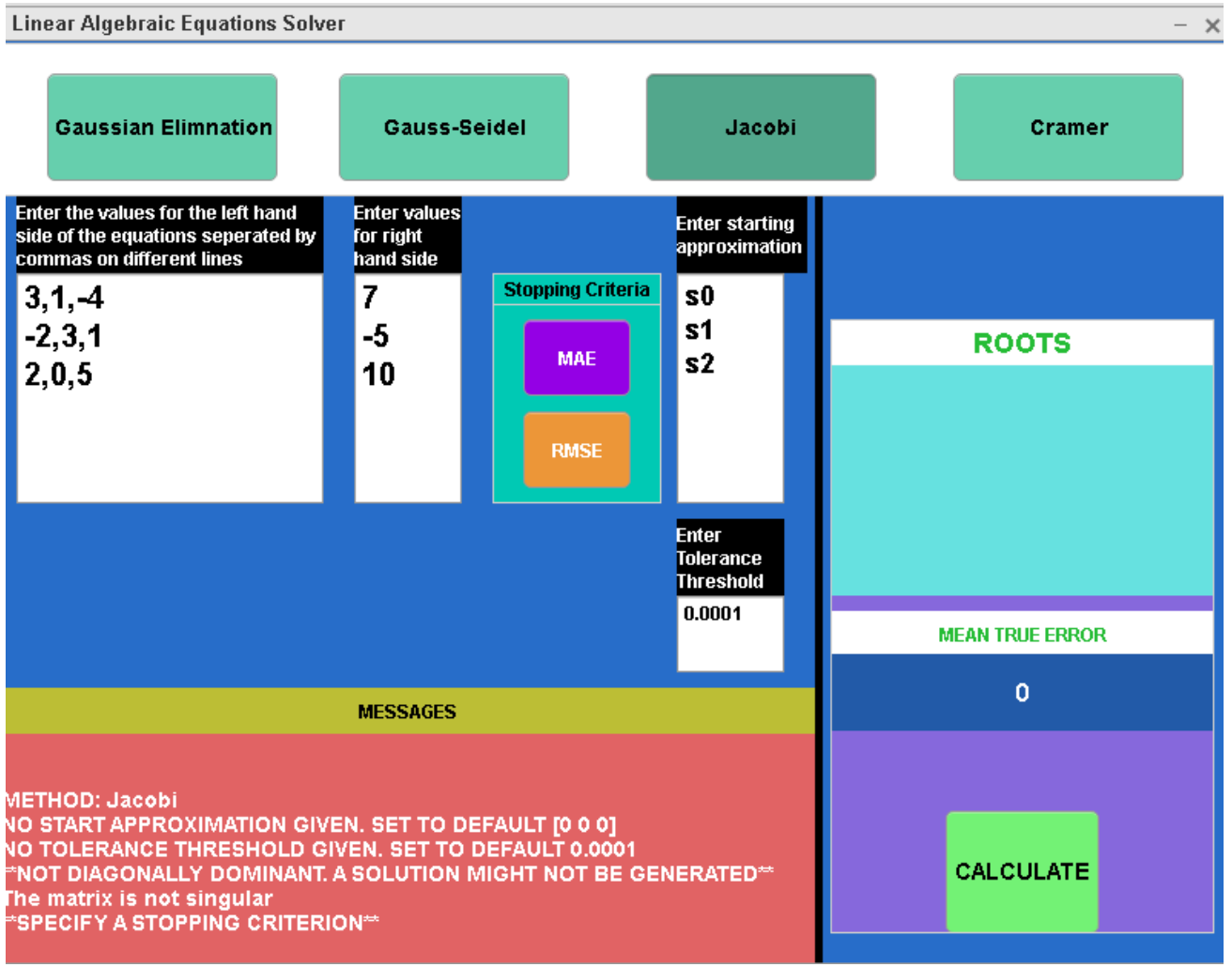


Same as previous example but with the Cramer method



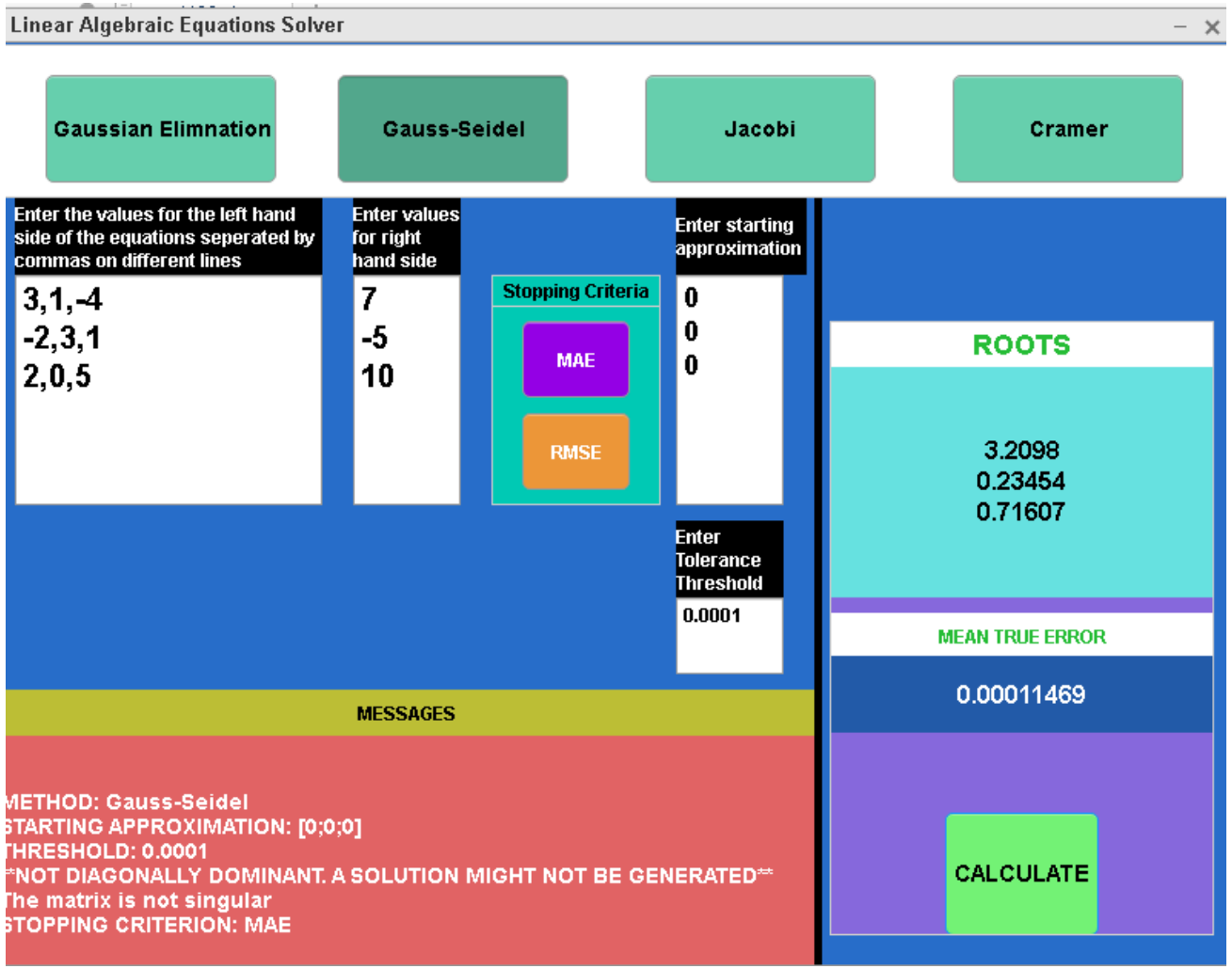
Iterative method chosen with nothing specified

The Jacobi iterative method was chosen. The messages box shows that since no starting approximation or tolerance threshold was specified, default values were given to them. Furthermore, because no stopping criteria was chosen, that is also shown in the message box and the user must choose one of the criteria to find the roots.

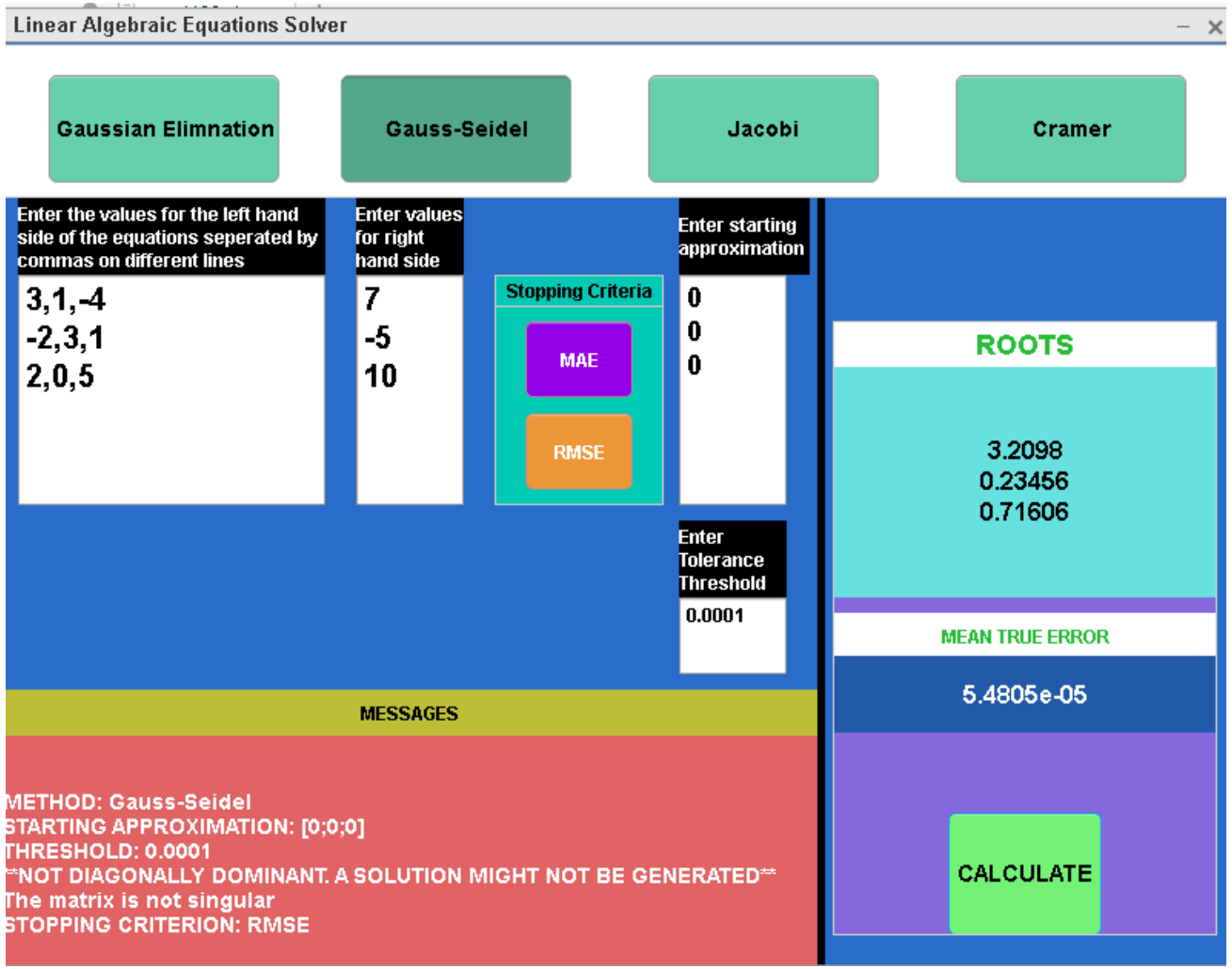


Iterative method with specified values

The Gauss-Seidel method was chosen and a stopping criteria (MAE), starting approximation and tolerance threshold were specified. As a result, the roots produced by the method along with the mean true error are shown. The details of what the user has specified are shown in the messages box.

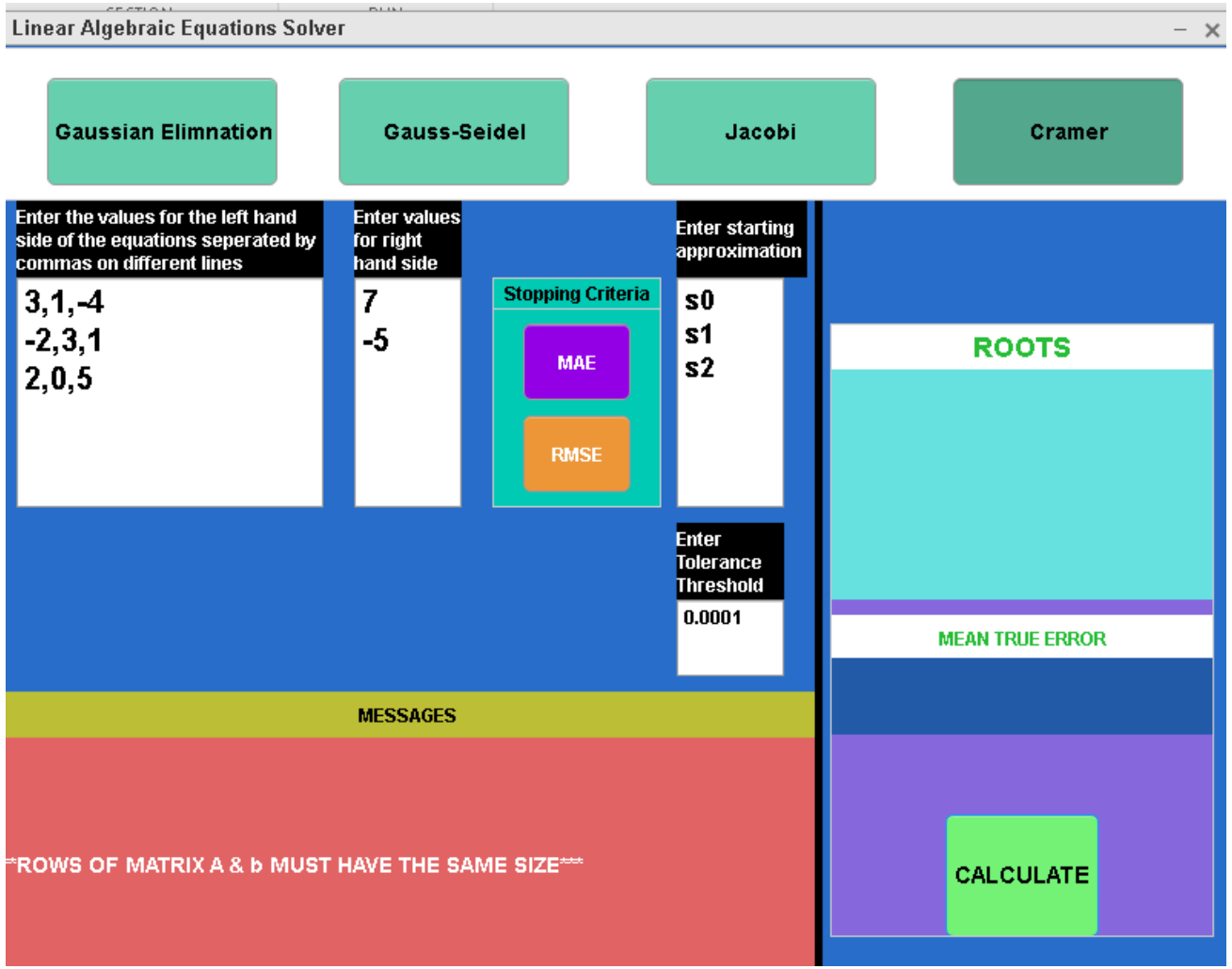


Same as previous but with the stopping criteria changed to RMSE. We can observe that switching to this stopping criteria yielded a smaller mean true error.



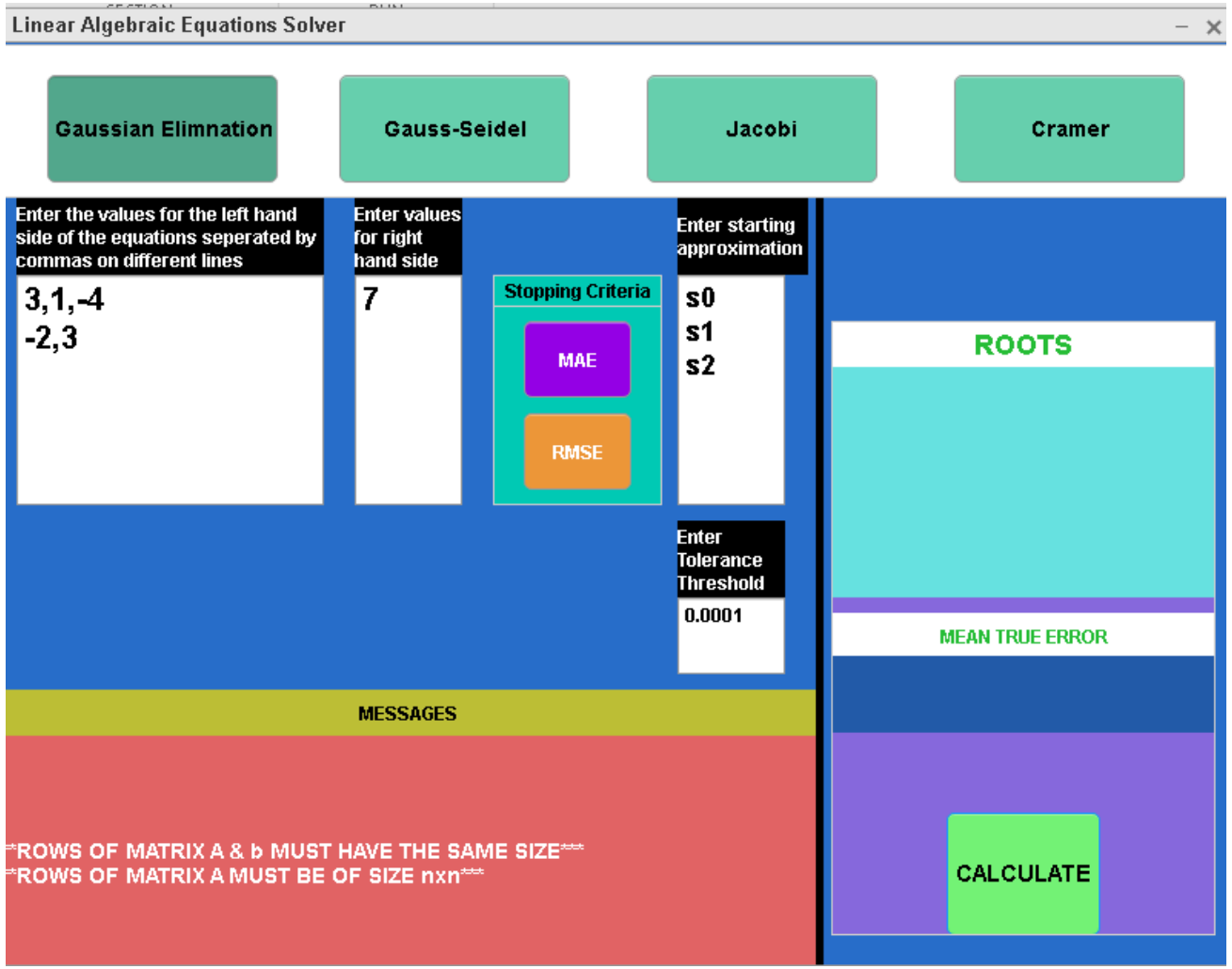
Error checking: Size of A and b are not the same

Since matrix A is a 3x3 matrix and matrix b only has 2 rows, no computation can be done and the messages box relays that the rows of both matrices should be the same



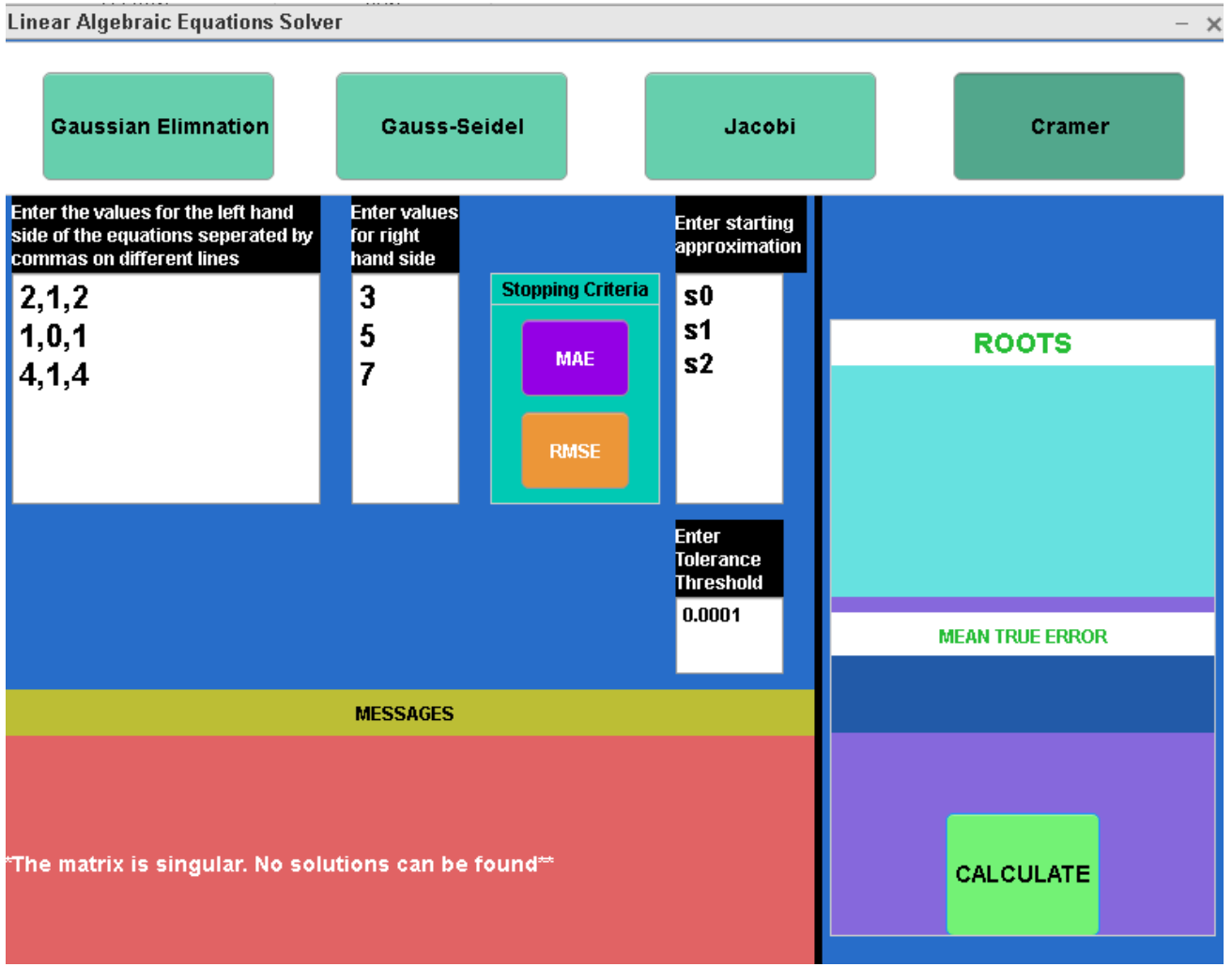
Error checking: Invalid sizes

Since matrix A is not an nxn matrix, and matrix b only has one row while matrix A has 2 rows, the message box lets the user know that the input is invalid.



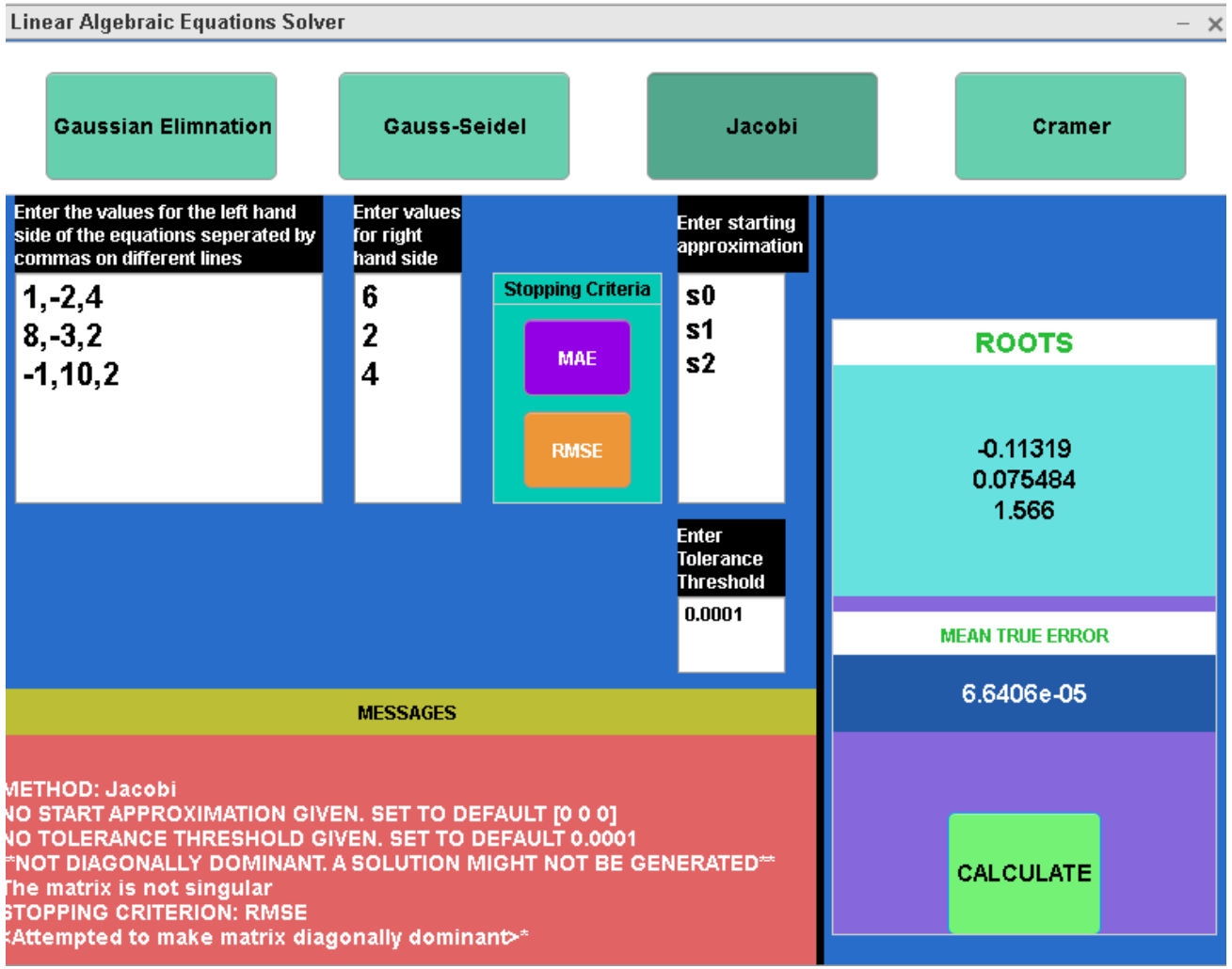
Error checking: Singular matrix

Since the matrix is singular (has a determinant of 0), the solutions cannot be found.



Error checking:
Not diagonally dominant, attempts to make matrix diagonally dominant

In this example, the solution of the matrix could not be found using the Jacobi iterative method because it is not diagonally dominant, the program detected this and attempted to rearrange the matrix so that it became diagonally dominant. The roots of the rearranged matrix are displayed and the message that the program tried to make the matrix diagonally dominant is shown in the message box



Error checking: Bad input

If the user input contains non-numeric characters, or just obviously incorrect input, the message box will instruct the user to check their input.