COMPUTER ALGORITHMS AND MODELLING

Statistical Model using Numerical Integrations

USMAN BASHARAT
000874782

Table of Contents

Introduction

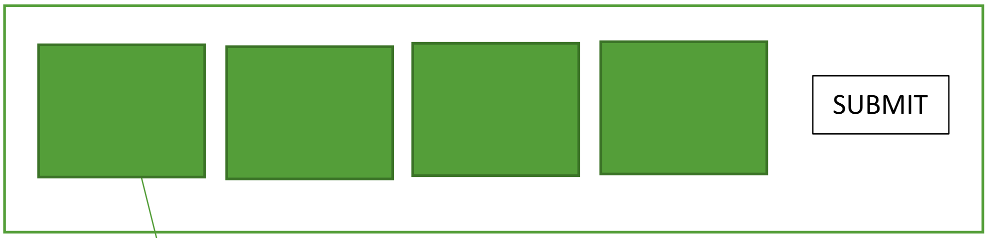
In this report, I will be identifying how I designed the Statistical Model with Numerical Integration. I will be providing screenshots and explaining how I came up with my final program. I will be providing a number of tests to discuss any validations within my program. Lastly, I will be critically evaluating the choice of algorithms used and performance with a reflection of what I have learnt.

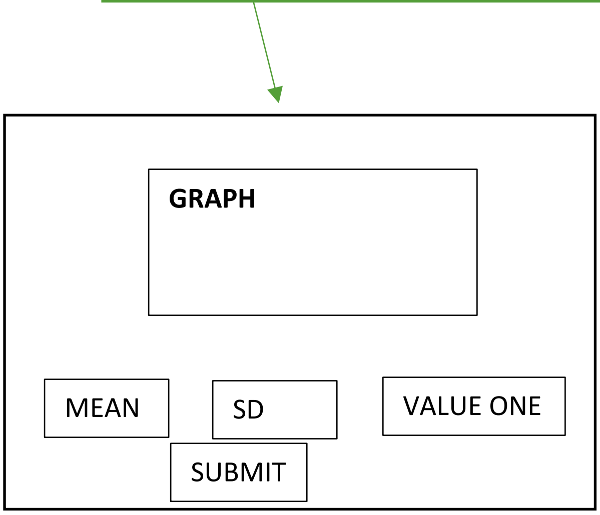
The aim of the *Statistical Model with Numerical Integration* are –

- Data to be directly entered by the user
- User can choose the type of area they are interested in
- Calculate the area and display the area
- Display the graph representation of the area
- Calculate the normal distribution table and display the normal distribution table

Plan

In this section, I will be explaining how I planned how the whole project



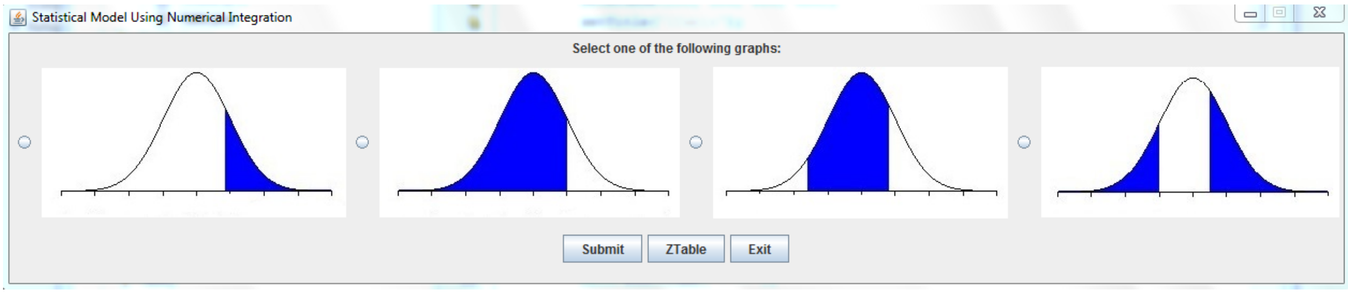


At the start of the project, I had to plan out how to do program each the graphs. Here is a diagram of how I started. Ona piece of paper, I drew these diagrams out. However, I put this in word to show how I started off. I felt that by putting four image graphs within this would make it simpler for the user to use. For each of them, the diagram displayed below it would look roughly like each of them. I found this all out by carefully reading the specification.

Design

In this section, I will be discussing how to use the program.

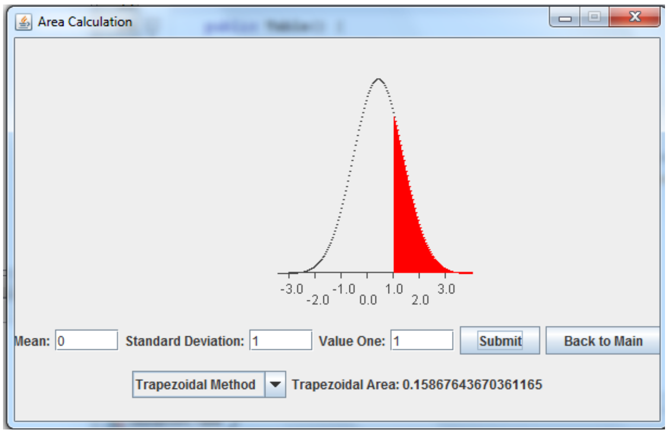
Referring to Figure 1, the objective of the program is to allow the user to choose between four different graphs of which they would like to find out the area. Figure 1 clearly shows different images of graphs by labelling them so that the user knows which one they are have chosen. Once they have chosen which one they want by selecting the radio button, they must click submit to refer them over to the next step of the program. The performance used within this is simple. I feel like having four graphs make the user feel as though it is not hard.



Statistical Model Using Numerical Integration

Select one of the following graphs:

***Figure 1**– shows the main program*



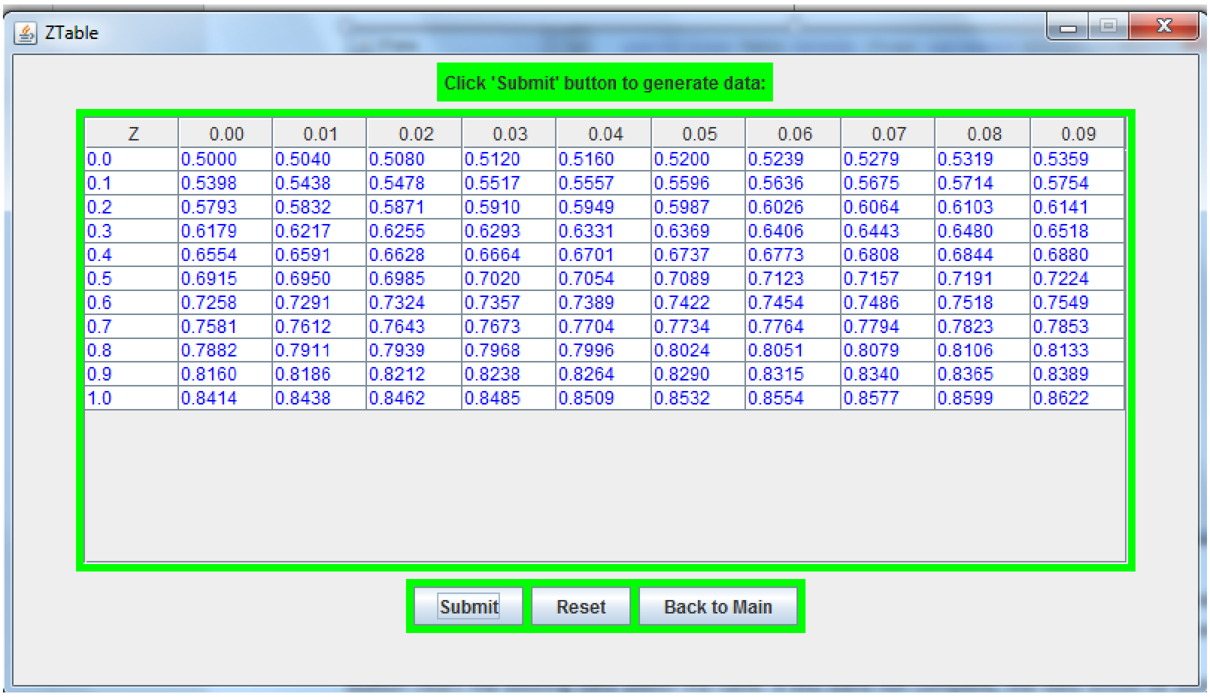
Area Calculation

Mean: 0 Standard Deviation: 1 Value One: 1

Trapezoidal Method Trapezoidal Area: 0.15867643670361165

Figure 2- shows values the user needs to input to display the graph

Assuming that the user has shown the first graph in Figure 1, the next chose is for the user is to enter the values. Referring to Figure 2, I have given the user to choose between four different inputs. The inputs are shown above. The dropdown box, the user needs to choose which area method he or she would like to choose from. I have given the user two to choose. Once the user has selected all four of the options, the graph will be displayed and the area will be displayed as soon as the submit button is clicked. As evident, this is the same display as any other graphs the user would have chosen. However, it would be different according to. For example, if the third one is clicked, it would be shaded in from two values entered from 1 to 2. if outside is selected with 1 and 2, it would shade the outside parts of 1 and 2.



Z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5200	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5754
0.2	0.5793	0.5832	0.5871	0.5910	0.5949	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6369	0.6406	0.6443	0.6480	0.6518
0.4	0.6554	0.6591	0.6628	0.6664	0.6701	0.6737	0.6773	0.6808	0.6844	0.6880
0.5	0.6915	0.6950	0.6985	0.7020	0.7054	0.7089	0.7123	0.7157	0.7191	0.7224
0.6	0.7258	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7518	0.7549
0.7	0.7581	0.7612	0.7643	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7853
0.8	0.7882	0.7911	0.7939	0.7968	0.7996	0.8024	0.8051	0.8079	0.8106	0.8133
0.9	0.8160	0.8186	0.8212	0.8238	0.8264	0.8290	0.8315	0.8340	0.8365	0.8389
1.0	0.8414	0.8438	0.8462	0.8485	0.8509	0.8532	0.8554	0.8577	0.8599	0.8622

Figure 3 – demonstrates the normal distribution table with an example of the user inputs

Referring to Figure 3, this shows the normal distribution table. As you can see, the normal distribution table allows the user to enter the values to see of what they want to view. If they want choose another set of values within the table, they would need to press the reset button. This button clears the existing data within the table. If this were not complete, the data would be added to the existing data at the end of it. This would confuse the user.

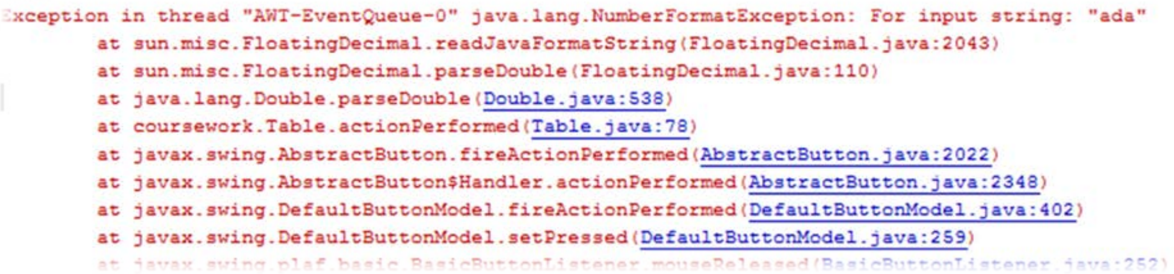
Testing

During this section, I will be discussing the tests and any problems I faced is explained whilst I was doing this.

Test Number	**Item to test**	**Code used**	**Expected Result**	**Actual Result**	**Pass/Fail**

1.	**Input Validation**	`JFormattedTextField meanTxt = new JFormattedTextField (new DecimalFormat());`
		`meanTxt.setValue(new Double(0));`
		`meanTxt.setColumns(5);`
			Expecting any other character to be put in and be set back to its default value	**I entered letters and it set the value back to 0.**	**Pass**

For this test, before I was using **JTextField meanTxt = new JTextField ();** and when I entered any value, a NumberFormatException came up as an error. I felt that this needed to be changed as I only need to enter decimal values within the textboxes that is needed. Any other values entered within JTextField causes errors within the program. By setting it to DecimalFormat, it allows any decimal values to be entered. Any other one would set it back to its original value, which is 0 in this case.

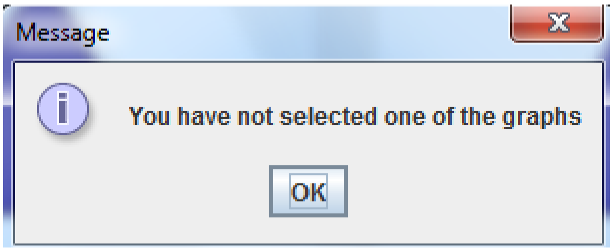


```
xception in thread "AWT-EventQueue-0" java.lang.NumberFormatException: For input string: "ada"
	at sun.misc.FloatingDecimal.readJavaFormatString(FloatingDecimal.java:2043)
	at sun.misc.FloatingDecimal.parseDouble(FloatingDecimal.java:110)
	at java.lang.Double.parseDouble(Double.java:538)
	at coursework.Table.actionPerformed(Table.java:78)
	at javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:2022)
	at javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2348)
	at javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:402)
	at javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:259)
	at javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:252)
```

Figure 4 - shows an exception when entered any other input apart from values in JTextField form

2.	**RadioButtons**	`JRadioButton below = new JRadioButton();`
		`ButtonGroup bG = new ButtonGroup();`
		`bG.add(below);`
		`middle.add(below);`
			Expecting to select each of the buttons by itself when selecting.	**I selected each of them by itself and it was grouped together so it didn't press two together at the same time.**	**Pass**

For this test, when I added the radio buttons by itself, I realised as soon as that when I clicked two at the same time, they were being selected together. This is not how I wanted to do this. Therefore, what I did is I grouped the radio buttons together so it would enable me to press one radio button at one time. Referring to Figure 5, I added a message to the main frame so that if none of the radio buttons has been selected and the person clicks submit, this message has appeared until the user has selected one of them. I feel that this is very important as part of the testing so that it does not appear as an error again.



You have not selected one of the graphs

Figure 5 - message display

3.	Images	`ImageIcon iconOne = new ImageIcon("Images/two.jpg");`
		`JLabel labelOne = new JLabel(iconOne);`
		` middle.add(labelOne);`
			Expecting the images to appear whenever I open the program and wherever I open the program	The images appeared whenever and wherever I left.	Pass

Adding images is necessary to identify which radio button is which. When I started to add images during my program, I came across a problem that I set the ImageIcon inside the label. For example, **labelOne.setIcon("new ImageIcon(Images/two.jpg");** and then I added this to the frame. I did this and it showed up once, but when I ran it on another computer, it did not show the image. Therefore, I had to change the style I was doing it, even though this was correct. I ended up changing the way I did it as shown the code above and it was working fully.

4.	Table	` main.addColumn("Z");`
		`main.addColumn("0.00");`
		`scroll = new JScrollPane(table);`
		`scroll.setPreferredSize(new Dimension(700, 300));`
			Expecting the columns to be added to the jTable.	The columns were added to the jTable as expected.	Pass

When I started to add each column, I added a jTable to the class and all I did was add the columns. This did not work. I was trying to figure out what went wrong and I soon realised that I did not add a scroll pane to the column. As soon as I entered this, I added the sizes of the scroll pane and it added the columns as I expected. In this problem, I just did not realise to add the scroll pane and it did not work at all. For future references, I need to the scroll pane if I was to use this again.

5.	ComboBox	`String[] list = {" ","Trapezoidal Method", "Rectangle"};`
		`JComboBox comboBox = new JComboBox(list);`
		`comboBox.setSelectedIndex(1);`
		`comboBox.addActionListener(this);`
		`bottom.add(comboBox);`
			Expecting the ComboBox to be added with the arrays listed.	The ComboBox added the arrays within it.	Pass

By adding this ComboBox, it gives another option for the user to choose which method the user would like to choose. Before I only added the labels to display both methods at once. However, since it is the user's choice, I felt like the user can choose which method the user would want to choose. For this, before doing this, all I did is type in **ComboBox.addItems(Labels Here);**. As soon I realised that an error has appeared in the ComboBox, I felt that something was wrong. I only added the items, but I did not specific where. Therefore, I added a String Array added within the ComboBox. Therefore, anything within the String Array that is listed, it is inside the ComboBox option.

6.	Adding Graph to Class	`JPanel addToGUI = aboveGraph;`
		`addToGUI.setLayout(new BoxLayout(aboveGraph, BoxLayout.LINE_AXIS));`
		`addToGUI.add(Box.createHorizontalGlue());`
		`addToGUI.add(Box.createRigidArea(new Dimension(250, 0)));`
		`addToGUI.setBorder(BorderFactory.createEmptyBorder(250, 0, 0, 0));`
		`add(aboveGraph, BorderLayout.CENTER);`
			Expecting this to add the graph and display it where I placed it within the GUI.	The graph appeared where I wanted it to be.	Pass

Whilst doing this at the very start, I found problems to how I should display it. At first, I added the graph to the frame by adding JPanel addToGUI = new JPanel();, addToGUI.setVisible();. Later on, I realised that this did not work and it needed it is only visible within the Graph's class and it needs its own "space". Therefore, I realised that it needs this space by creating a layout for this to visible in the current class.

7.	Graph	`if ("1".equals(option)) {`
		`double xInterAbove = (62 + (above - start) / xInc);`
		`if (i >= xInterAbove) {`
		`g2D.setColor(Color.red)`
		`g2D.draw(new Line2D.Double(xValue, 210, xValue, c));`
		`}`
		`}`
			Expecting "1" to be shaded once called upon.	**Once "1" was called upon, the program was run and it shaded the specific area that is needed.**	**Pass**

During my time of making this graph, I came across multiple struggles whilst doing this. One of the struggles that I found during this is putting the labels on the x-axis. I knew first thing that I needed a loop for this to be complete. I kept putting 'g.drawLine (0, 0, 0, 0);'. Soon enough, this did not work. Later on, I realised that I need to draw a line and updates the values too. I realised that I needed the mean to be added with each other for each one and run in a loop. This is what I did and it worked.

8.	Buttons	`JButton submit = new JButton("Submit");`
		`top.add(submit);`
		`submit.addActionListener(this);`
		`if(e.getSource() == submit) { …}`
			Expecting the button to respond to what is typed within the IF statement	**Button responded to what was typed in and did exactly what I want it do.**	**Pass**

Whilst I was doing this, I added 'back to Main' button so that once user runs the application, they can have access to going back to the main menu once they have pressed one of the graphs. So how it runs is once one of the graphs has been chosen, the main menu disappears and lets the user do the area on the chosen graph. If the user has chosen a wrong one, they can go back by using the 'Back to Main' button. They can exit the application too. Also, an important point is that this stops the user from selecting multiple forms of graphs. It would confuse and maybe crash the application.

9.	Accessing the JAVA Application	`SHIFT + F6`	**Expecting the program to run the GUI**	**The program ran without any errors by displaying the GUI.**	**Pass**

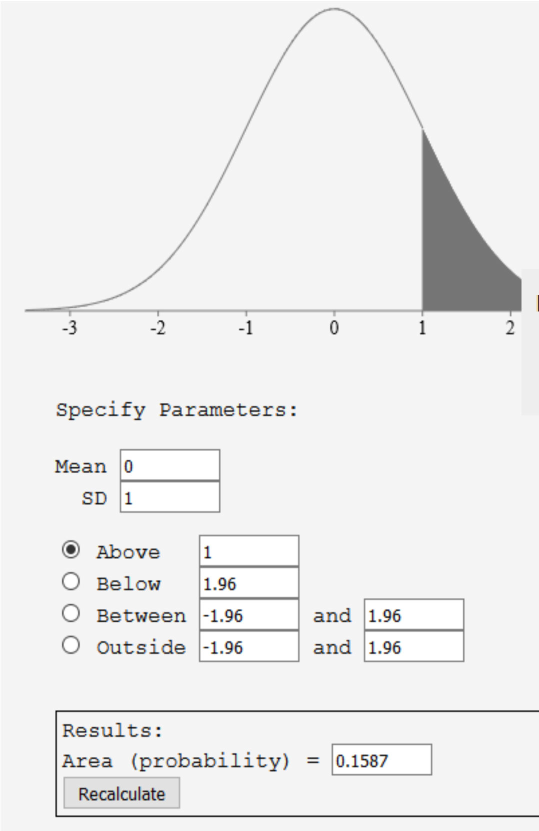
10.	Displaying the area and ensuring it is correct	`area.setText("Trapezoidal Area:" + " " + (1 - ModelCalculations.trapRule(secondValue, valueOne, strips, meanValue, sdValue)));`	**Expecting the area to be displayed**	**The area was displayed as expected and it displayed the correct answer**	**Pass**

During my time of calculating the area, I did it a separate way before. I started to do the calculations separately without any loops. When I did this, the area that was displayed was wrong, but the concept was correct. Therefore, at a later stage, I thought by using a loop would be much more accurate. Therefore, what I did was I added times all of the strips by delta by putting it in a loop and using the probability density function of it. This worked as an accurate answer, but I was missing something. Each time, I got an answer, I was close every time. Referring to **(M.Lane D),** I compared the answer to the online source. Once it was correct, I knew what to do for the rest of the graphs.

Evaluation

*Critical evaluation of the **choice** and **performance** of the **data structures** and **algorithms** used*

Three methods were associated with the algorithms used for numerical integration. Four methods were given. Those four are True and Relative True Error, Rectangle Rule, Trapezoidal Rule and Simpson's rule. These methods are used for numerical integration used within java. This means that doing integration within java can be complicated. Therefore, this is used to make it easier to do the integration of PDF (Probability Density Function). I decided to use Trapezoidal Rule and Rectangular Rule, because of the accuracy of the answers that were given are near enough correct. Referring to Figure 6, this shows an online example of the area calculated. The user entered mean 0, standard deviation 1, and above value as 1. The answer as shown is 0.1587 for the online version. Referring to Figure 6 and Figure 7, I entered with the same values and the results were the same. However, this is comparing the Trapezoidal Method that I selected which is accurate as you can see.



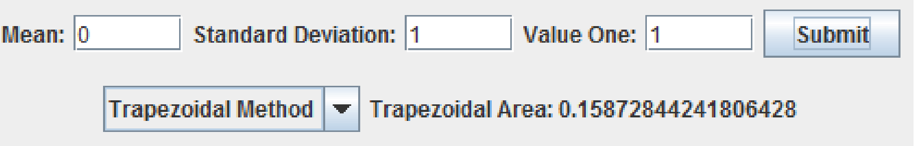
Specify Parameters:

Mean 0
SD 1

● Above 1
○ Below 1.96
○ Between -1.96 and 1.96
○ Outside -1.96 and 1.96

Results:
Area (probability) = 0.1587

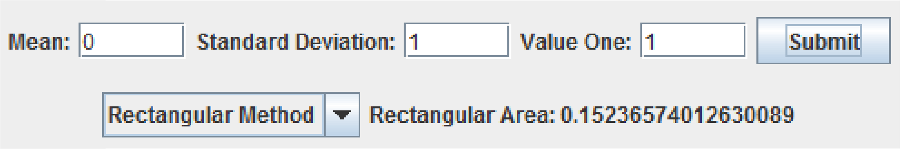
Figure 6 - shows the online answer



Mean: 0 Standard Deviation: 1 Value One: 1

Trapezoidal Method Trapezoidal Area: 0.15872844241806428

Figure 7 - shows the answer I got when I entered the same values

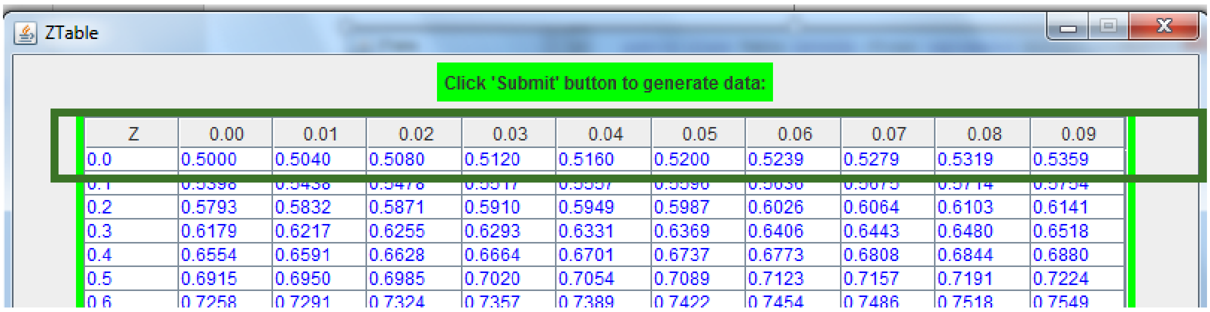


Mean: 0 Standard Deviation: 1 Value One: 1

Rectangular Method Rectangular Area: 0.15236574012630089

Figure 8 - shows the example of another method being entered with the same values

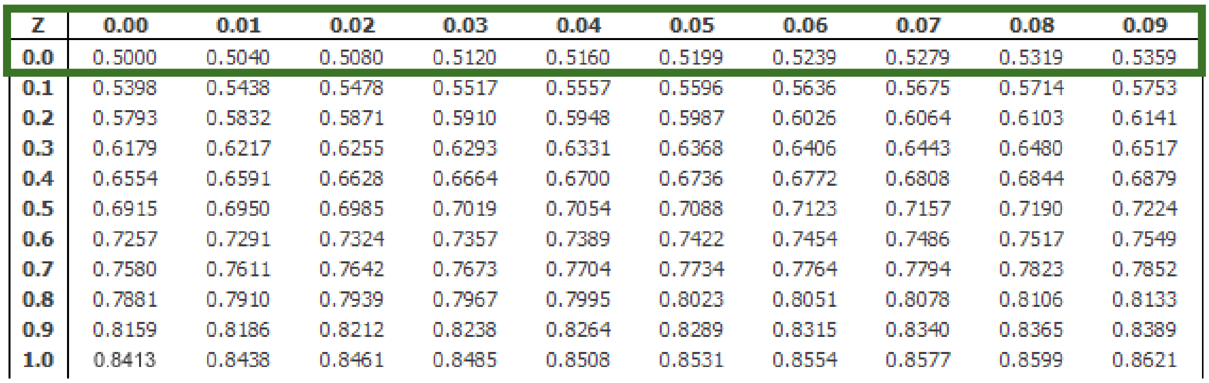
However, when I entered the values with the Rectangular Method, there is a not that much of a difference with the answer. This is due to how accurate the answer is. Different methods of calculating numerical integration can lead to having different accuracy of answers. As displayed, it has near enough if **0.6%** difference between the two. As you can see the difference is **0.1587 – 0.1523 = 0.0064.**



Click 'Submit' button to generate data:

Z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5200	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5754
0.2	0.5793	0.5832	0.5871	0.5910	0.5949	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6369	0.6406	0.6443	0.6480	0.6518
0.4	0.6554	0.6591	0.6628	0.6664	0.6701	0.6737	0.6773	0.6808	0.6844	0.6880
0.5	0.6915	0.6950	0.6985	0.7020	0.7054	0.7089	0.7123	0.7157	0.7191	0.7224
0.6	0.7258	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7518	0.7549

Figure 9 - shows the normal distribution table for the data that I have entered in my program

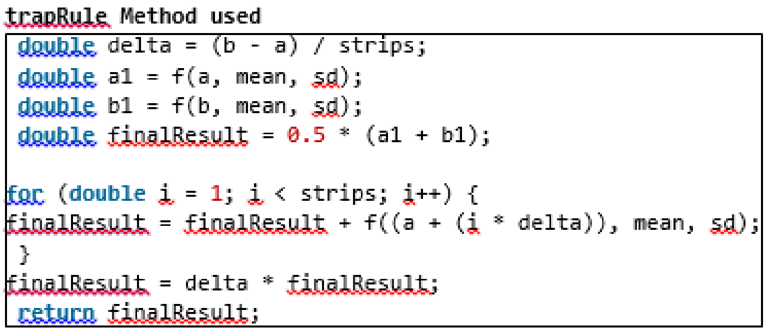


Z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621

Figure 10 - shows the online version of the normal distribution table

Referring to Figure 9, this is the normal distribution table that I have calculated and is currently being displayed. Referring to Figure 10, this is an image online. As you can see, both of the results are roughly the same. The first thing I look at to see if the results are accurate is the blue box being highlighted above. They are the same result and on the same line, if you follow it along and compare both of them, it is the same.

Referring to Figure 11, Trapezoidal Rule has been used below. I feel that the performance of the algorithms below is necessary to name it *public* – so that other classes can see it and name it *static* – so that other classes can call it according to the name of the class. This makes it easier to call these methods. All you need to do to call for Figure 11 is by typing the class name and its method you want to use e.g. **ModelCalculations.trapRule(….);.**



```
trapRule Method used
 double delta = (b - a) / strips;
 double a1 = f(a, mean, sd);
 double b1 = f(b, mean, sd);
 double finalResult = 0.5 * (a1 + b1);

for (double i = 1; i < strips; i++) {
finalResult = finalResult + f((a + (i * delta)), mean, sd);
 }
finalResult = delta * finalResult;
 return finalResult;
```

Figure 11 - shows Trapezoidal Rule method that I have used within my program

Reflection

During this course, I felt that I have got a better understanding of how to take the next approach to programming. Being able to look through the next stage of the programming let me enhance the skills that I acquired from first year. I feel that this part of the course has given me confidence to practice more. I feel that being able to practice regularly enhances my skills further. Having time doing this was a real bonus. In that time, we got to learn what we had to do from scratch. Even though it sounded hard, by attending lectures and tutorials, it gave a bonus to get better at it and practice each week.

Improvements

This section includes improvements that I acknowledge in my program

The improvements that I realise would make a huge difference to the program is not letting the user reset the normal distribution table. Referring to Figure 3, I can improve this if the user would want to press the data again, the table would realise that and clear the table again by itself. This would not add any confusion to the user. It would make it simpler for the user.

Another improvement that I felt is necessary is adding a Y-axis to Figure 2 and updating the Y-axis according to the standard deviation. I felt that during the time I spent on this, I was too focused on the x-axis updating according to the mean, I did not spend time on the Y-axis being updated.

Another improvement I feel can be optional for further improvement is being for the user to use the mouse and being able to drag the shaded part to the part the user would like it to be. This would make it easier and simple for the user to use the program.

Another improvement is not necessary, but just to make aware; I only put in two methods to calculate the area. Simpson's rule is another that I did not add during the time that I spent on this coursework. I feel that by adding another once would make it much better.

Another improvement is as soon as the user has chosen one of the graphs; the graph does not appear as soon as the values have been entered. Before doing anything, the graph needs to be visual beforehand.

References

1. **M.Lane, D. (2016). *Normal Distribution*. [Online] Available at: http://onlinestatbook.com/2/calculators/normal_dist.html**

[Accessed 27 Nov. 2016].

2. **Laerd Statistics, (2016). *Normal Distribution Table*. [Image] Available at: https://statistics.laerd.com/statistical-guides/img/normal-table-large.png**

[Accessed 27 Nov. 2016].