Smit Pateliya
40202779

SOEN 6011-Summer 2022
1st July 2022

1 Introduction of Beta Function B(p,q)

There are two most popular functions in Mathematics: Gamma and Beta functions. Gamma function is a single variable function, while Beta is two-variable function. Here, we will talk about Beta function in detail. Beta function is often as known as the first type of Euler's integrals. We often use β notation to represent it.

1.1 Beta Function Definition

Beta Function is one kind of a function that is classified as the first type of Euler's integrals. It is defined in the real numbers domain. It is represented by B(p,q), where p and q are real numbers.

1.2 Beta Function Formula

The Beta function is derived by as follows:[2]

$$B(p,q) = \int_0^1 t^{p-1}(1-t)^{q-1}dt$$

The domain of the function are $p \in \mathbb{R} : p > 0$ and $q \in \mathbb{R} : q > 0$. The co-domain of the function are $p \in \mathbb{R}$: $p > 0$ and $q \in \mathbb{R} : q > 0$.
Because of its relation with Gamma Function and Factorial function, it is very important in Mathematics, Calculus and Analysis. Many complex integrals can be deduced and reduced to similar expressions using Beta Function.

2 Properties of Beta Function

- Beta Function is Symmetric function

$$B(p,q) = B(q,p)$$

- Beta Function can be expressed in the different form.[1]

$$B(p,q) = 2\int_0^{\frac{\pi}{2}} \sin^{2p-1}\theta \cos^{2q-1}\theta d\theta$$

$$B(p,q) = \int_0^{\infty} \frac{t^{p-1}}{(1+t)^{p+q}}dt$$

$$B(p,q) = \int_0^{1} \frac{t^{p-1}+t^{q-1}}{(1+t)^{p+q}}dt$$

- Beta can be expressed with the help of the Gamma function.

$$B(p,q) = \frac{\Gamma p \Gamma q}{\Gamma(p+q)}$$

- Beta Function also possesses the Recurrence Relationship Property.

$$B(p+1,q) = B(p,q)\frac{p}{p+q}$$

- Beta Function can be expressed with the help of the Factorial.

$$B(p,q) = \frac{(p-1)!(q-1)!}{(p+q-1)!}$$

3 The context of Use Model for Beta Function

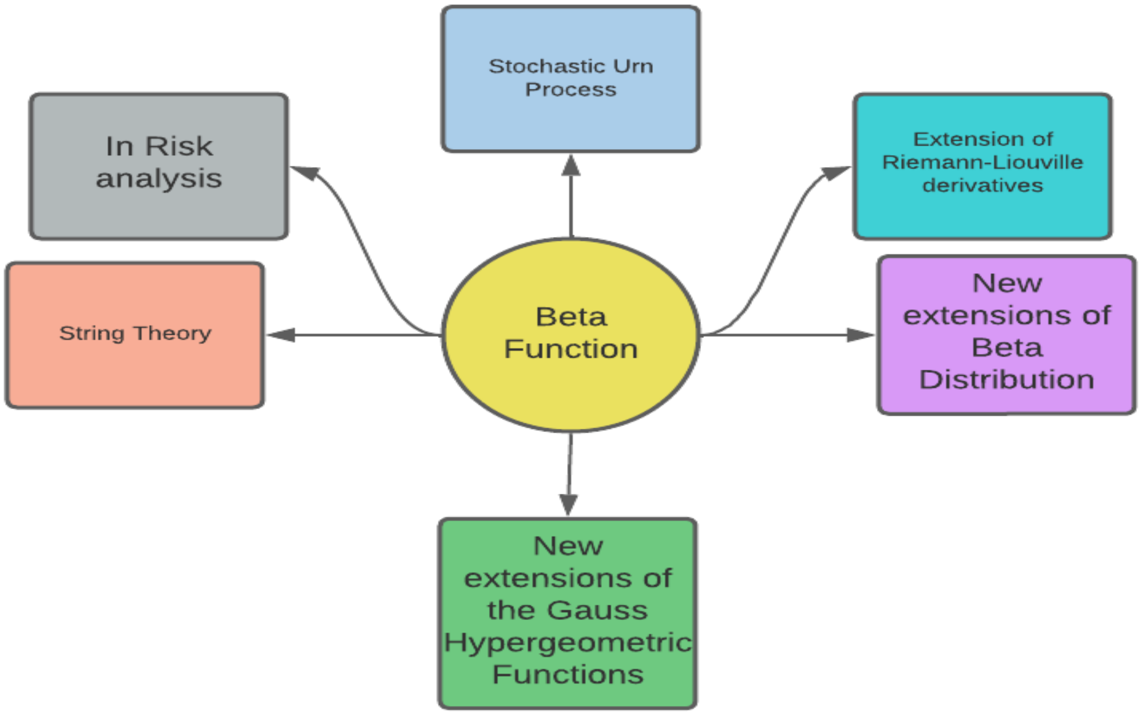

Stochastic Urn Process
In Risk analysis
Extension of Riemann-Liouville derivatives
String Theory
Beta Function
New extensions of Beta Distribution
New extensions of the Gauss Hypergeometric Functions

Figure 1: Context Of Use Model

4 Requirements of Beta Function

4.1 Identifier - R1

- **Identifier:** R1
- **Type:** Functional Requirements
- **Description:** The function needs two arguments p and q to evaluate function.
- **Rationale:** p and q

4.2 Identifier - R2

- **Identifier:** R2
- **Type:** Functional Requirements
- **Description:** The two variable p and q which we have defined in the R1, needs to be positive real numbers.
- **Rationale:** $p \geq 0$ and $q \geq 0$

4.3 Identifier - R3

- **Identifier:** R3
- **Type:** Functional Requirements
- **Description:** The co-domain of function is $\mathbb{R}+$.
- **Rationale:** $B(p,q) \geq 0$

4.4 Identifier - R4

- **Identifier: R4**
- **Type:** Functional Requirements
- **Description:** If the domain belongs to $\mathbb{Z}^+$, then we can evaluate beta function with the help of the Gamma function.
- **Rationale:** $\{\forall p, q \in \mathbb{Z}^+ \mid B(p,q) = \frac{\Gamma p \Gamma q}{\Gamma(p+q)}\}$[3]

4.5 Identifier - R5

- **Identifier: R5**
- **Type:** Functional Requirements
- **Description:** We need a supporting function to calculate the value of X raised to the power Y, if we need result of the Beta Function for positive real numbers as input. Therefore, we need to create power function $power(x,y)$ to calculate X^Y.

4.6 Identifier - R6

- **Identifier: R6**
- **Type:** Functional Requirements
- **Description:** The two variable $pandq$ for beta function can be equal or cannot be equal.
- **Rationale:** $p = q$ or $p \neq q$

4.7 Identifier - R7

- **Identifier: R7**
- **Type:** Functional Requirements
- **Description:** We need a supporting function to calculate the factorial of X.

4.8 Identifier - R8

- **Identifier: R8**
- **Type:** Functional Requirements
- **Description:** We need a supporting function to calculate the natural logarithmic to find power of fractional power value.

4.9 Identifier - R9

- **Identifier: R9**
- **Type:** Functional Requirements
- **Description:** We need a supporting function to calculate the square root of X.

4.10 Identifier - R10

- **Identifier: R10**
- **Type:** Non-functional Requirements
- **Description:** The method which use to calculate the Beta Function, should be able to calculate result in efficient way for large positive inputs for p and q.

4.11 Identifier - R11

- **Identifier: R11**
- **Type:** Non-functional Requirements
- **Description:** We need a way to store large decimal values for calculating the value of Beta Function accurately.

4.12 Identifier - R12

- **Identifier: R12**
- **Type:** Non-functional Requirements
- **Description:** The method which use to calculate the Beta Function, should be able to calculate result without considering input values and hardware requirements.

4.13 Identifier - R13

- **Identifier: R13**
- **Type:** Functional Assumption
- **Description:** To calculate the value of Beta Function, we can take approximate value of the Definite Integral using Numerical Methods.

5 Algorithm for Implementing Beta Functions

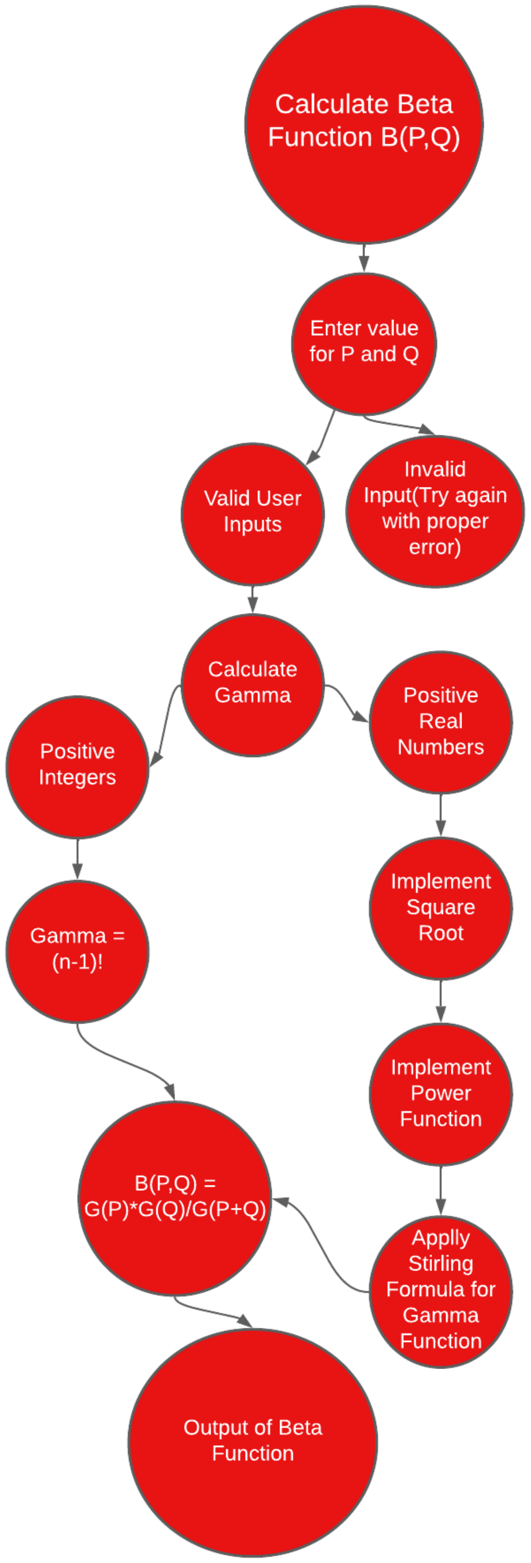


Figure 2: Mind Map for Beta Function

5.1 Algorithm - 1

In this algorithm, Beta function uses Gamma function to evaluate its value. Moreover, Gamma function uses factorial method to calculate its value. Hence, the factorial of the negative numbers is not possible therefore, the domain of beta function is all positive integers. i.e. $\forall p, q \in \mathbb{Z}^+$.

$$B(p,q) = \frac{\Gamma p \Gamma q}{\Gamma(p+q)}$$

$$\Gamma p = (p-1)!$$

5.1.1 Advantages

1. This algorithm is easier to implement and debug as programmer needs to implement only one factorial function to calculate beta function.
2. This way of calculating beta function gives more accurate answer compare to other algorithm. The second algorithm depends on the value e which is infinite number.
3. This algorithm is faster to execute and performs better then other algorithm.

5.1.2 Disadvantages

1. This algorithm can only be used for positive real integers, as the factorial of the fraction numbers and negative number can not possible.

Algorithm 1 Calculate Beta Function with the help of Factorial

```
Require: $p > 0$ and $q > 0$                                   ▷ i.e. $p, q \in \mathbb{Z}^+$
Result: $B(p, q)$
  procedure CalculateFactorial($value$)
      $result \leftarrow 1$
      for $i \leftarrow 2$ to $value$ do
          $result \leftarrow result * i$
      end for
      return $result$                                  ▷ Return Factorial of the value
  end procedure

  procedure CalculateGamma($value$)
      $result = $ CalculateFactorial($value - 1$)
      return $gamma$                                   ▷ It returns the gamma value
  end procedure

  procedure CalculateBeta($p, q$)
      $value1 \leftarrow$ CalculateGamma($p$)
      $value2 \leftarrow$ CalculateGamma($q$)
      $r \leftarrow p + q$
      $value3 \leftarrow$ CalculateGamma($r$)
      $beta \leftarrow \frac{value1 * value2}{value3}$
      return $beta$                                    ▷ It returns the beta value
  end procedure

  $result \leftarrow$ CalculateBeta($p, q$)                ▷ Final result of $Beta(p, q)$
```

5.2 Algorithm - 2

We can implement beta function with the help of Stirling's approximation for factorials. Stirling's approximation is approximation method. This method is also for accurate results for small value of p.
The sterling's approximation equation is represented as:

$$B(p,q) = \frac{\Gamma p \Gamma q}{\Gamma(p+q)}$$

$$\Gamma p = \sqrt{\frac{2\pi}{p}}(\frac{p}{e})^p$$

5.2.1 Advantages

1. The algorithm can compute beta function for all the positive real numbers i.e. $p > 0$ and $q > 0$.
2. The algorithm can evaluate result of beta function for all the positive integers i.e. $\forall p, q \in \mathbb{Z}^+$.
3. The algorithm give a approximation value for the integration formula. So, we can able to compute beta function without computing integration.

5.2.2 Disadvantages

1. This algorithm can give only accurate results for the integration formula.
2. This algorithm is quite complex compare to first algorithm even though we are not calculating integration formula.
3. There is more different between actual results and the required result for the small values when the algorithm is implemented. However, for all larger values, the difference between both values becomes narrower.

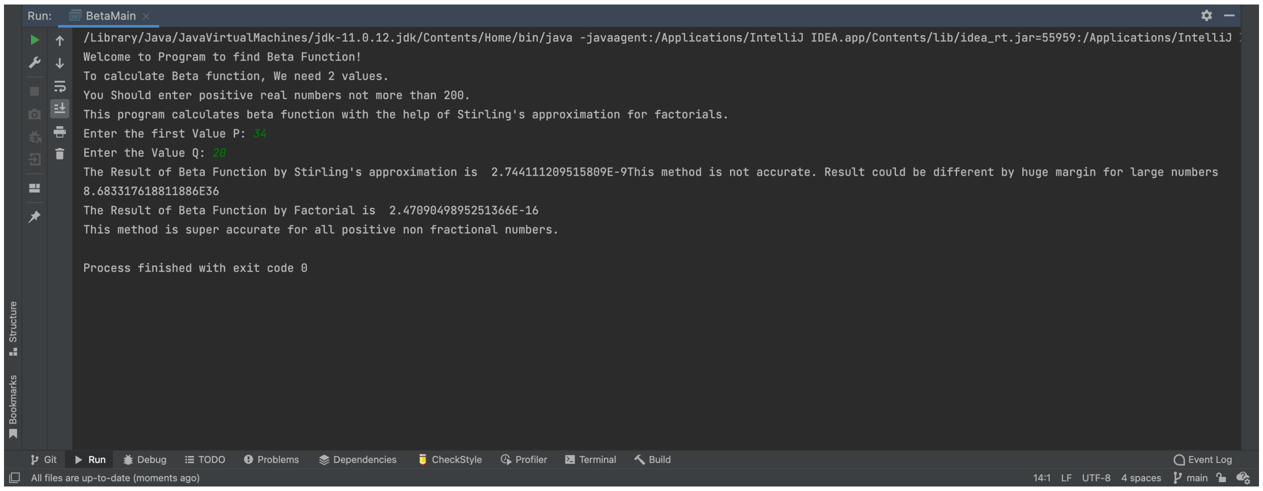
Algorithm 2 Calculate Beta Function with the help of Stirling's Approximation

```
Require: $p > 0$ and $q > 0$                                   ▷ i.e. $p, q \in \mathbb{Z}^+$
Result: $B(p, q)$
  procedure CalculateSquareRoot($value$)
      $squareRoot \leftarrow value/2$
      repeat
          $result \leftarrow squareRoot$
          $result = (result + (value/result))/2$
      until $(result - squareRoot) \neq 0$
      return $squareRoot$                                     ▷ Return Square Root
  end procedure

  procedure CalculatePower($value$, $power$)
      $result \leftarrow 1$
      for $i \leftarrow 1$ to $power$ do
          $result \leftarrow result * value$
      end for
      return $result$                                         ▷ Return base to the power
  end procedure

  procedure CalculateGamma($value$)
      $intermediateValue1 \leftarrow$ CalculatePower($\frac{value}{e}$, $value$)
      $intermediateValue2 \leftarrow$ CalculateSquareRoot($\frac{2\pi}{value}$)
      $gamma = intermediateValue1 * intermediateValue1$
      return $gamma$                                          ▷ It returns the gamma value
  end procedure

  procedure CalculateBeta($p, q$)
      $value1 \leftarrow$ CalculateGamma($p$)
      $value2 \leftarrow$ CalculateGamma($q$)
      $r \leftarrow p + q$
      $value3 \leftarrow$ CalculateGamma($r$)
      $beta \leftarrow \frac{value1*value2}{value3}$
      return $beta$                                           ▷ It returns the beta value
  end procedure

  $result \leftarrow$ CalculateBeta($p, q$)                   ▷ Final result of $Beta(p, q)$
```

6 Notes about the Java Implementation

I have implemented this algorithm with the help of the simple Command line interface. Also, I have catched all the possible run time errors in the catch block. Moreover, I have given suggestion to avoid the run time exceptions.

I have created all the mathematical functions from scratch i.e. Power Function, Log function, Factorial Function, Square-root Function.



```
/Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55959:/Applications/IntelliJ
Welcome to Program to find Beta Function!
To calculate Beta function, We need 2 values.
You Should enter positive real numbers not more than 200.
This program calculates beta function with the help of Stirling's approximation for factorials.
Enter the first Value P: 34
Enter the Value Q: 20
The Result of Beta Function by Stirling's approximation is  2.744111209515809E-9This method is not accurate. Result could be different by huge margin for large numbers
8.683317618811886E36
The Result of Beta Function by Factorial is  2.4709049895251366E-16
This method is super accurate for all positive non fractional numbers.

Process finished with exit code 0
```

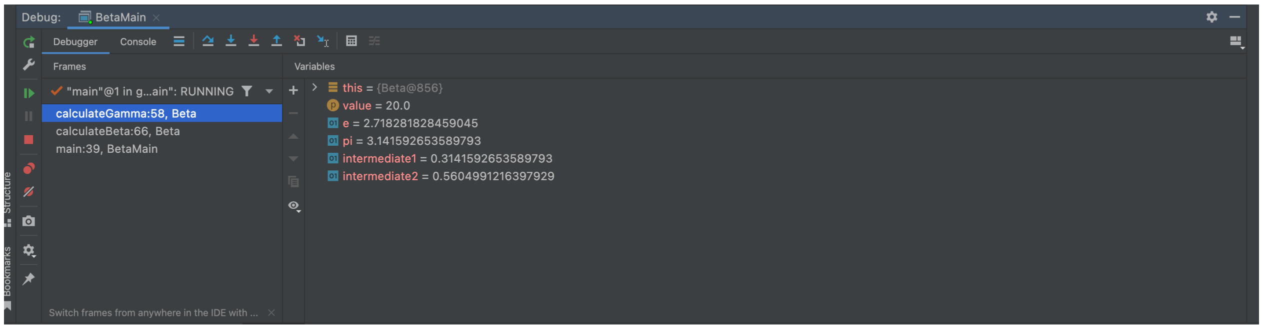
Figure 3: Output of the Main Program

7 Information about the Debugger

I have used the IntelliJ IDE for implementing Java code for this function. So, I have used IntelliJ degguer for debugging the code.

IntelliJ is one of the most widely use IDE for Java language. It also has one of the best debugger with plenty of different features for simplicity of programmer. I have mentioned some of the features below.

- Programmer can check value of the different variable at the run time, if run-time exception occurs.
- It provides well defined graphical user interface for checking values of the variables.
- Programmer can create different breakpoints and watchpoints to evaluate particular expressions.
- It is providing multiple debugger sessions for multiple codes or same code.



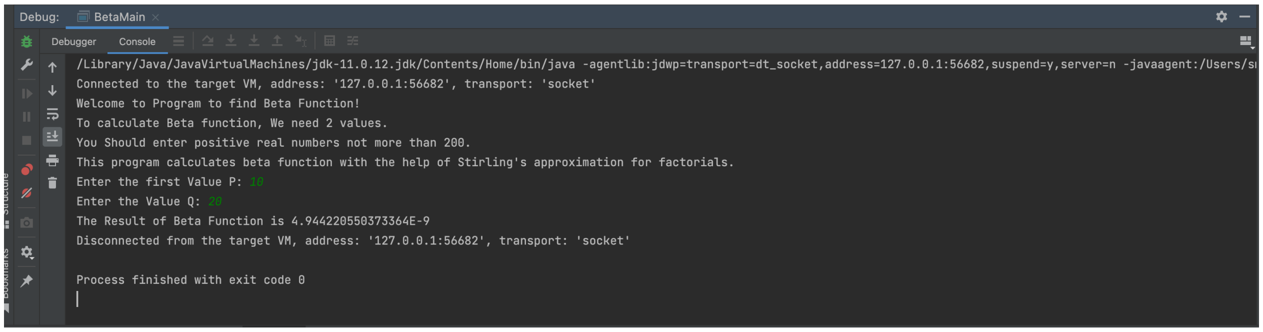
```
"main"@1 in g...ain": RUNNING
calculateGamma:58, Beta
calculateBeta:66, Beta
main:39, BetaMain
Switch frames from anywhere in the IDE with ...
```

```
this = {Beta@856}
value = 20.0
e = 2.718281828459045
pi = 3.141592653589793
intermediate1 = 0.3141592653589793
intermediate2 = 0.5604991216397929
```

Figure 4: Debugger

7.1 Advantages

- Programmer can simply evaluate mathematical functions.
- It is providing plenty of different customization for the breakpoints and watchpoints: add or remove breakpoints at the run-time; skip some breakpoints; add conditions etc.
- Programmer can get different info about the particular steps.
- Programmer can easily debug recursive code.



```
/Library/Java/JavaVirtualMachines/jdk-11.0.12.jdk/Contents/Home/bin/java -agentlib:jdwp=transport=dt_socket,address=127.0.0.1:56682,suspend=y,server=n -javaagent:/Users/s
Connected to the target VM, address: '127.0.0.1:56682', transport: 'socket'
Welcome to Program to find Beta Function!
To calculate Beta function, We need 2 values.
You Should enter positive real numbers not more than 200.
This program calculates beta function with the help of Stirling's approximation for factorials.
Enter the first Value P: 10
Enter the Value Q: 20
The Result of Beta Function is 4.944220550373364E-9
Disconnected from the target VM, address: '127.0.0.1:56682', transport: 'socket'

Process finished with exit code 0
```

Figure 5: Debugger

7.2 Disadvantages

- For the first time user, it is quite steep learning curve.
- It is providing a lot features for debugging, it is overwhelming and inefficient to use over time.

8 Qualities of the Program

8.1 Efficiency of the code

I have implemented all the mathematical functions with the help of the single loop. So the code is very efficient and gives result in near perfect real time.

8.2 Usability of the code

I have provided all the instructions to calculate the Beta function at the start of the run-time. So, any user is able ti calculate beta function. Moreover, code has different error catching mechanism to eliminate run time errors.

For programmers, I have written Java doc all the functions. So, any programmer can modify codes for his or her purposes.

8.3 Robustness of the code

I have provides different messages for different expected errors. So, with the help of this message, user can solve errors straightforwardly.

8.4 Maintainability of the code

I have implemented the code in the different classes and different methods for maintainability purposes.

8.5 Correctness of the code

I have computed the beta function with the help of two different methods.

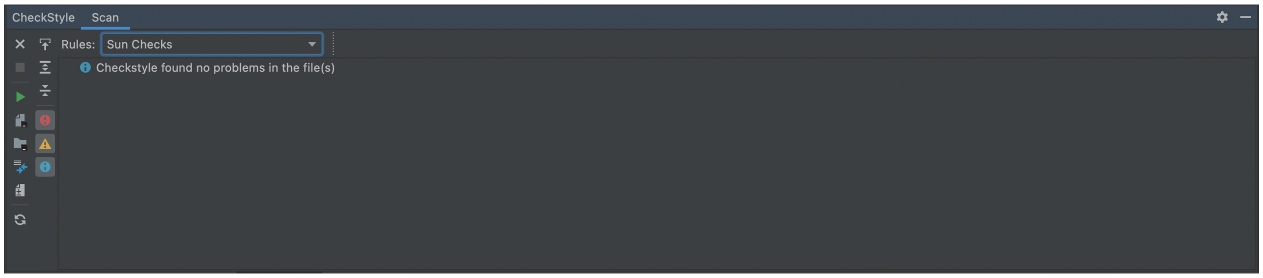
1. By Stirling's approximation for Gamma Function
2. Calculating with the help of the Factorial for Gamma Function

If we can use Stirling's approximation, then we need to calculate power function for real numbers, which is very inaccurate. So, the result of beta function is very inaccurate. If we use factorial method for finding Gamma function, then we can get quite accurate result for the beta function.

9 Checkstyle

Checkstyle is a tool to help programmers write Java code to following coding standards. Checkstyle has some following features.

- It has naming conventions for functions and variables.
- It is using imports and scope modifiers, etc.



Rules: Sun Checks

```
Checkstyle found no problems in the file(s)
```

Figure 6: Debugger

9.1 Advantages

- Checkstyle gives you the ability to create your own rules and regulations for programs.
- Checkstyle is providing external tooling to integrate it with different external tools.
- It can be used with the different IDEs without any issues. It is providing same interface to every IDEs.

9.2 Disadvantages

- It can be hard to install in the IDEs. It can change the original settings of the IDEs.
- Sometimes, It forces unnecessary rules which won't affect the code and wastes time to reorganize the code.

10 Test Cases

10.1 Test Case - 1

- **Function:** Beta.validateInputs(double)
- **Input:** Beta.validateInputs(34)
- **Expected:** False
- **Result:** Pass
- **Traceability:** R2, R6

10.2 Test Case - 2

- **Function:** Beta.power1(double,double)
- **Input:** Beta.power1(2,3)
- **Expected:** 8
- **Result:** Pass
- **Traceability:** R5

10.3 Test Case - 3

- **Function:** Beta.factorial(double)
- **Input:** Beta.factorial(6)
- **Expected:** 720
- **Result:** Pass
- **Traceability:** R7

10.4 Test Case - 4

- **Function:** Beta.logn(double)
- **Input:** Beta.logn(2)
- **Expected:** 0.693147
- **Result:** Pass
- **Traceability:** R8

10.5 Test Case - 5

- **Function:** Beta.calculatePower(double,double)
- **Input:** Beta.calculatePower(10,0.5)
- **Expected:** 3.162277
- **Result:** Pass
- **Traceability:** R5

10.6 Test Case - 6

- **Function:** Beta.calculateSquareRoot(double)
- **Input:** Beta.calculateSquareRoot(2)
- **Expected:** 1.41
- **Result:** Pass
- **Traceability:** R9

10.7 Test Case - 7

- **Function:** Beta.calculateGammaStirling(double)
- **Input:** Beta.calculateGammaStirling(1.5)
- **Expected:** 0.8389
- **Result:** Pass, not accurate
- **Traceability:** R4

10.8 Test Case - 8

- **Function:** Beta.calculateBetaStirling(double,double,double)
- **Input:** Beta.calculateBetaStirling(10,20,30)
- **Expected:** 4.9494E-9
- **Result:** Pass, not accurate
- **Traceability:** R1, R2, R3, R6

10.9 Test Case - 9

- **Function:** Beta.calculateGammaFactorial(double)
- **Input:** Beta.calculateGammaFactorial(12)
- **Expected:** 3.99168E7
- **Result:** Pass, highly accurate
- **Traceability:** R4

10.10 Test Case - 10

- **Function:** Beta.calculateBetaFactorial(double,double,double)
- **Input:** Beta.calculateBetaFactorial(10,20,30)
- **Expected:** 4.9925087E-9
- **Result:** Pass, highly accurate
- **Traceability:** R1, R2, R3, R6

References

[1] Beta Function Accessed: 02-07-2022 URL: https://mathworld.wolfram.com/BetaFunction.html

[2] Beta Function Accessed: 01-07-2022 URL: https://byjus.com/maths/beta-function/

[3] Beta Function Accessed: 16-07-2022 URL: https://www.efunda.com/math/beta/

[4] Stirling's Approximation Accessed: 16-07-2022 URL: https://en.wikipedia.org/wiki/Stirling's_approximation#Stirling's_formula_for_the_gamma_functio

[5] CheckStyle Accessed:23-07-2022 URL: https://checkstyle.sourceforge.io/

[6] Debug code Accessed:22-07-2022 URL: https://www.jetbrains.com/help/idea/debugging-code.html

[7] Beta Function and its Applications, Riddhi D. Accessed: 06-07-2022 URL:http://sces.phys.utk.edu/ moreo/mm08/Riddi.pdf

[8] Alhassan, Elvis & Albert, L. & Louis, O.. (2015). On some Applications of Beta Function in some Statistical Distributions. Researcher