

Algorithms for Standard Deviation

1. Pseudo code of Traditional (Iterative) approach

Algorithm 1 Traditional Algorithm

```
1: procedure STANDARDDEVIATION( )
2:    $Sum \leftarrow 0$ 
3:    $Mean \leftarrow 0$ 
4:    $Var \leftarrow 0$ 
5:    $SD \leftarrow 0$ 
6:    $Number \leftarrow 0$ 
7:    $Size \leftarrow Array.count()$ 
8:    $Sum \leftarrow Sum + Sum$ 
9:   for  $i \leftarrow 0, size$  do
10:     $Sum+ = Array$ 
11:   end for
12:    $SD = squareRoot(SD/Size)$ 
13:   Return SD
14: end procedure
```

Which one to choose, Traditional or N-Pass?

- Standard deviation can be calculated more quickly and with considerably more efficiency using the n-pass technique, which can accept more data inputs in the form of files. Other than that, the problem is that because it employs recursion rather than iteration, the memory stack would fill up quickly.
- The standard deviation can be calculated using the traditional approach, but it can take some time if the input data is huge and the frequency of the numbers is high. In addition, it is more prone to errors. As a result, the Multi pass or n-pass algorithm is more faster and more effective than the Traditional approach for standard deviation.

Algorithms for Standard Deviation

2. Pseudo code of N-Pass approach

Algorithm 2 N-Pass Algorithm

```
1: procedure STANDARDDEVIATION( )
2:    $Size \leftarrow Array.count()$ 
3:   for  $i \leftarrow 0, size$  do
4:      $Sum+ = Array$ 
5:   end for
6:    $Mean \leftarrow Sum/Size$ 
7:   for  $i \leftarrow 0, size$  do
8:      $SD = SD + power((Array[i] - Mean), 2)$ 
9:   end for
10:   $Variance = SD/Size$ 
11:   $SD = SquareRoot(Variance)$ 
12:  Return SD
13: end procedure

14: procedure POWER(x,y)
15:   for  $i \leftarrow 1, y$  do
16:      $Result = Result * x$ 
17:   end for
18:   Return Result
19: end procedure

20: procedure SQUAREROOT(x)
21:    $temp$ 
22:    $sqrt \leftarrow \frac{x}{2}$ 
23:    $temp \leftarrow sqrt$ 
24:    $sqrt \leftarrow \frac{temp + \frac{x}{temp}}{2}$ 
25:   while  $temp - sqrt \neq 0$  do
26:      $temp \leftarrow sqrt$ 
27:      $sqrt \leftarrow \frac{temp + \frac{x}{temp}}{2}$ 
28:   end while
29:   Return  $temp$ 
30: end procedure
```

Mind Map

The below given mind map shows the flow of the pseudo code for finding Standard Deviation. The algorithm will start from Main() method by taking input of data values. It will then call the CalculateSum() procedure to calculate sum of the data elements. With mean and array size in hand, CalculateVariance() method will be called. Internally this method calls Power() function to compute power and atlast SquareRoot() is used to find Standard Deviation.

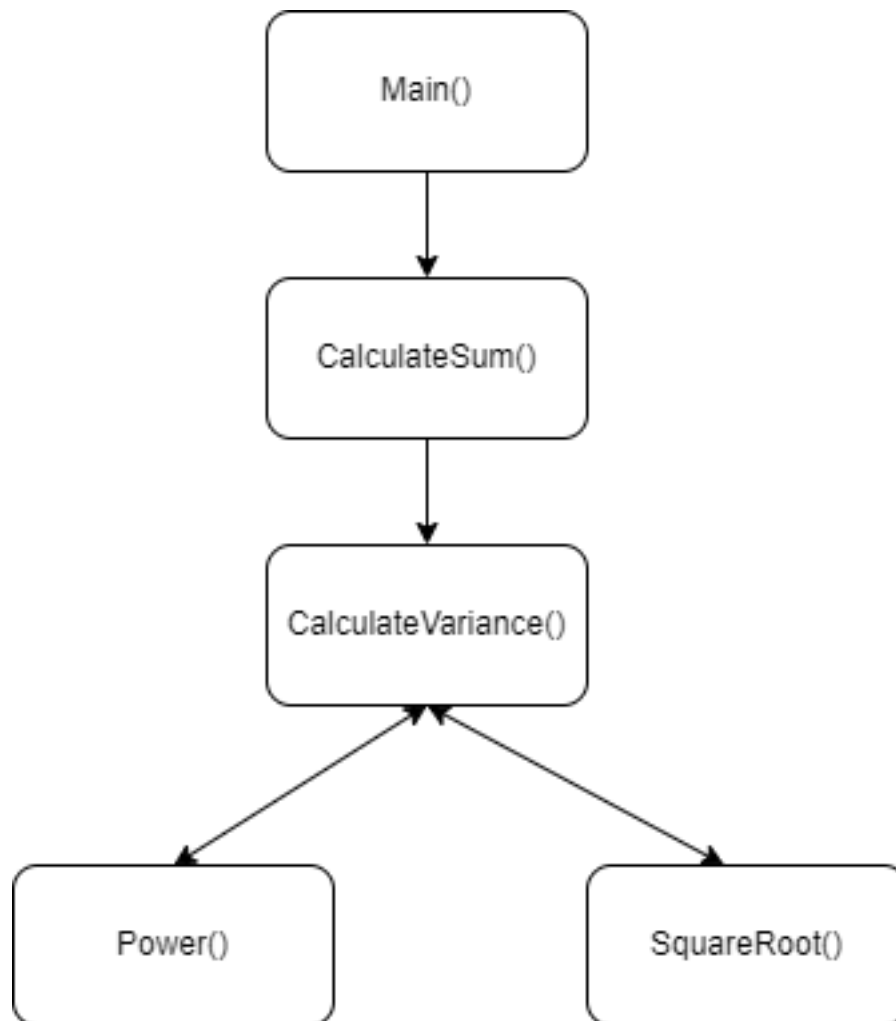


Figure 1: Mind Map

Pros of Traditional Algorithm

- This method does not use the stack so it's faster.
- Traditional algorithm consumes less memory.
- The code is easy to understand and debug.

Cons of Traditional Algorithm

- The iterative alternative is to repeatedly dynamically allocate or resize memory blocks. On many platforms automatic allocation is much faster, to the point that its speed bonus outweighs the speed penalty and storage cost of recursive calls.
- It makes the code longer.
- Sometimes execution and calculation time becomes higher.

Pros of N-Pass Algorithm

- It is faster and much more efficient to use because it makes use of memory up to the mark.
- Can take large set of data to find standard deviation.
- It makes the code smaller.

Cons of N-Pass Algorithm

- The inbuilt garbage collector begins to operate when the memory stack becomes mostly full, deleting the unused memory that has filled the stack.

References

- [1] <https://benpfaff.org/writings/clc/recursion-vs-iteration.html>
- [2] <https://www.tutorialspoint.com/what-are-the-differences-between-recursion-and-iteration-in-java>
- [3] https://influentialpoints.com/Training/variance_and_standard_deviation-principles-properties-assumptions