Journal of the Nigerian Association of Mathematical Physics
***Volume* 25 (November, 2013), pp 303 – 306**
© J. of NAMP

Performance of a Written Matlab Function to an Inbuilt Function for a Set of Examination Scores

Arowolo O. T., Salawu S. O. and Kareem R. A.

**Department of Mathematics,
Lagos State Polytechnic, Ikorodu, Nigeria**

Abstract

In this work, we write a function which asks the user to enter a set of exam scores. We use MATLAB program to calculate and display the arithmetic mean, standard deviation, median, mode, upper quartile, lower quartile, range and interquartile range of the data. A function $[\ ] = st1(x)$ is defined for the program to calculate these values and the performance of this function is compared with the performance of the inbuilt MATLAB functions.

1.0 Introduction

In the day to day activities of individual student, there is always arise the need to make decisions to evaluate past activities with a view to determine performance or seek improvement and projection into the future. Successful achievement of any of the above would depend to a large extend on availability of necessary facts [1]. Entering large amounts of data is very tedious and would not be productive. To circumvent this, MATLAB has some inbuilt functions which can be used to produce sets of pseudo-random numbers. MATLAB is a computer program for people doing numerical computation and its strength includes cutting-edge algorithms, enormous data handling abilities [2]. This availability also provides an opportunity to easily conduct numerical experiments and to tackle realistic and more complicated problems [3, 4]. In this work, a program is written in MAATLAB to enter a set of examination scores. The program is written to calculate and display the arithmetic mean, standard deviation, median, mode, upper quartile, lower quartile, range and interquartile range of the data. We compare our own function with the performance of the inbuilt MATLAB functions.

2.0 How the Program Works

Assuming x is the set of examination scores. The program is evaluated for two cases:

Case 1: If the examination score is odd, i.e. $x = 23,40,25,20,25$

Case 2: If the examination score is even, i.e. $x = 23,40,25,20$

In any of the two cases above, the mean can be calculated as follow

Case 1: The user should enter the set of exam scores one after the other. Sum them up and divide by the total number. i.e.

$$x = 23,40,25,20,25$$

$$x = \frac{23 + 40 + 25 + 20 + 25}{5} = \frac{133}{5} = 26.6$$

Then the output of the program will be, the mean is 26.6

Hence the program has carried out the following

Input: $x = 23,40,25,20,25$

Processing: $x = \frac{23 + 40 + 25\_20 + 25}{5} = \frac{133}{5} = 26.6$

Output: The mean is 26.6

Considering the median for the two cases

Corresponding author: ***Kareem R. A.,*** E-mail: -, Tel.: +2348032018520

Case 1: $x = 23,40,25,20,25$

The scores must be arranged in ascending or descending order before determining the median. i.e.

$$x = 20,23,25,25,40$$

The median is the middle number or score, the median is 25

Input: $x = 23,40,25,20,25$

Processing: sort (ascending order) $20,23,25,25,40$

Output: the median is 25

Case 2: $x = 23,40,25,20$

The scores should be arranged in ascending or descending order. i.e.

$$x = 20,23,25,40$$

Here 23 and 25 fall into the middle. So, we add the two scores and divide by 2

$$\text{Median } (x) = \frac{23+25}{2} = \frac{48}{2} = 24$$

The median is 24.

Input: $x = 23,40,25,20$

Processing: sort $20,23,25,40$

$$\frac{23+25}{2} = \frac{48}{2} = 24$$

Output: the median is 24

If the scores are odd it will be sorted in the right order and the middle score will be picked as the median score.

For the mode the function returns more than one solution if the students have the same scores and the mode is the number with the highest frequency. It implies that all the exam scores are mode. That mean that no student have the same score and returns only one mode if there is a particular score with the highest frequency.

The program executes each variable as defined in the function of the program.

This is how the program works for other variables in the program.

3.0 Program Variable Definition

The variables used to calculate these values are:

Mean = Mean of the exam scores

SD = Standard deviation of the exam scores

Median = Middle value of the exam series

Mode = Mode of the exams

Lower Quartile = Lower quartile of the exam

Upper Quartile = Upper quartile of the exam scores

Interquartile Range = Interquartile range of the exam scores

Range = Range of the exam scores

This program attempts to calculate the arithmetic mean, standard deviation, median, mode, upper quartile, lower quartile, range and interquartile range of a set of examination scores.

1.1. Program 1

```
Function [ ]=st1(x)
Tic
[n,p]=size(x);
N=n*p;
ssum=0;

For i=1:N;
    ssum=ssum+ x (I,i);
end
mean=ssum/N;
display(sprint( ‘The mean is %d’,mean))
wsum=0;
for i=1:N
```

```
wsum=wsum+( x (1,i) – mean)^2;
end
SD=sqrt (wsum/ (N – 1))

y=sort (x);
Med = (N+1) / 2;

if Med==floor (Med);
    Median=y(1,Med);
else
    Med1=ceil (Med);
    Med2=floor (Med);
    Median=(y(1,Med1)+y(1,Med2)) / 2;
end
display (sprint( ‘ The median is %e’,Median))

i=1
z=zeros (1,(N – 1));
count=0
for i=1:N – 1;
    if y(1, i+1)==y(1,i);
            count=count+1;
            z(1,i)=count;
    else
            count=0;
    end
end
for i=1:N – 1;
    if z(1,i)==max(z);
            modee=y(1,i)
    end
end

y=sort (x);
[n,p]=size(x);
N=n*p;
lqp=0.25*N;
If lqp / lqp==1
    lqp;
else
    round (lqp);
end
e=round(lqp)+1;
Lower_Quartile=y(1,e)

u=sort(x);
[n,p]=size(x);
N=n*p;
Uqp=0.75*N;
If uqp / uqp==1
    Uqp;
else
    round (uqp);
end
f=round (uqp)+1;
Upper_Quartile =u(1,f)
Inter_range=u(1,f) – y(1,e)
```

```
Range=max(u) – min(u);
display (sprint ( ‘ The range is %f’,Range))
toc
end
```

1.2. Program 2

```
tic
Mean=mean (x)
Sigma=std (x)
Median=median (x)
Mode=mode (x)
f=dataset (x);
summary (f)
InterquartileRange=iqr (x)
Range=range (x)
Toc
```

1.3. Program 3

```
N=150;
x-rand (1,N);
```

Table 1: Elapsed time

Elapse time for program 1	Elapse time for program 2
0.004184	0.349750
0.002923	0.025382
0.003262	0.011884
0.002959	0.012467
0.009145	0.010487
0.002841	0.010663
0.002863	0.012797
0.002895	0.010937
0.002867	0.011798
0.002904	0.012719

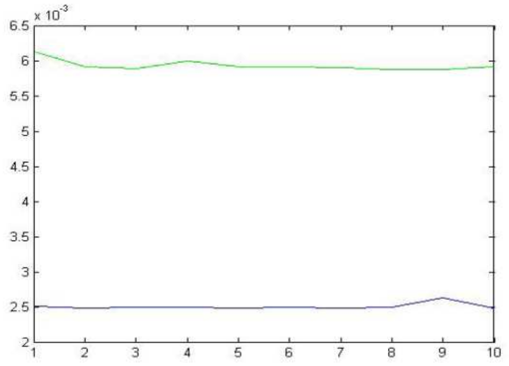

Figure 1: Comparison of Elapsed Time

A plot of these 10 random numbers in Figure 1 where the lower line is for program 1 and the upper line is for program 2 shows the performance of our code to be faster than that of the inbuilt MATLAB function.

4.0 Conclusion

The use of Tic and Toc gives the total elapsed time of our program to be 0.000729 and that of the inbuilt function is given to be 0.009124. This shows that it takes our program shorter time to calculate all these values compared to the MATLAB inbuilt function.

The program is tested for random numbers 150 using ‘rand’ as defined in program 3 above and it gives accurate solutions with a shorter time to that of the inbuilt function. This is evident in Table 1 above from the first 10 random numbers.

Therefore the program has successfully calculated the values of the mean, standard deviation, median, mode, upper quartile, lower quartile, range and interquartile range of the data, within a short period of time compared to that of the inbuilt function.

Reference

[1] R.A. Kareem and P.K. Olowu (2006): Descriptive Statistics (The Modern Approach).
[2] Staff of the Indian University Stat / Math center.
[3] D. Houcque, Introduction to MATLAB for Engineering students, Northwestern University (version 1.2, August 2005).
[4] The Math Works, Inc. Statistics Toolbox, for used with MATLAB, User’s Guide version 4, Natick, MA01760-2098