Lab 01: Equations and -scripts

**Handout date:** Wednesday, August 28, 2019

**Due date:** Wednesday, September 11, 2019 at the beginning of the lecture as hardcopy

*This lab counts 4 % toward your total grade*

**Objectives:** In this lab   
[a] you will practice your understanding of the structure of mathematical equations and subsequently properly typesetting these equations,   
[b] you will practice your  skills.

**Format of answer:** Your answers (graphs and verbal description) should be handed in as ***hard-copy*** in ***one*** document. Add a running title into the header of the document with the following information: ***your name***, ***GISC6301-Lab01*** and ***page numbers***. Label each answer properly starting with its task number. Maintain the sequence of questions. Format any code and computer output properly before inserting it into the document with your answer. R-code and text output need to be in a ***monospaced*** font (i.e., fixed-pitch font) such as Courier New so proper spacing and alignments are preserved. Excessive, but irrelevant, output will lead to a deduction of your accumulated points.

## Task 1: Equation Editor Exercise (1 point)

Typeset the derivation of the computational equation of the variance equation in line (7) from its definitional counterpart in line (1). Make sure that your formatting follows exactly the one given in the screen shot below. In particular, focus on [a] the change from the *italics* equation mode to the upright text mode in equation (1), [b] proper nesting of the employed mathematical templates, [c] the alignment of all equation at the “”-sign, and [d] the use of underbraces in equations (3) and (4). You do not need to replicate the *explanations* and *line numbers* in red. Deviations from the formatting in the screen shot will lead to a loss of partial points.

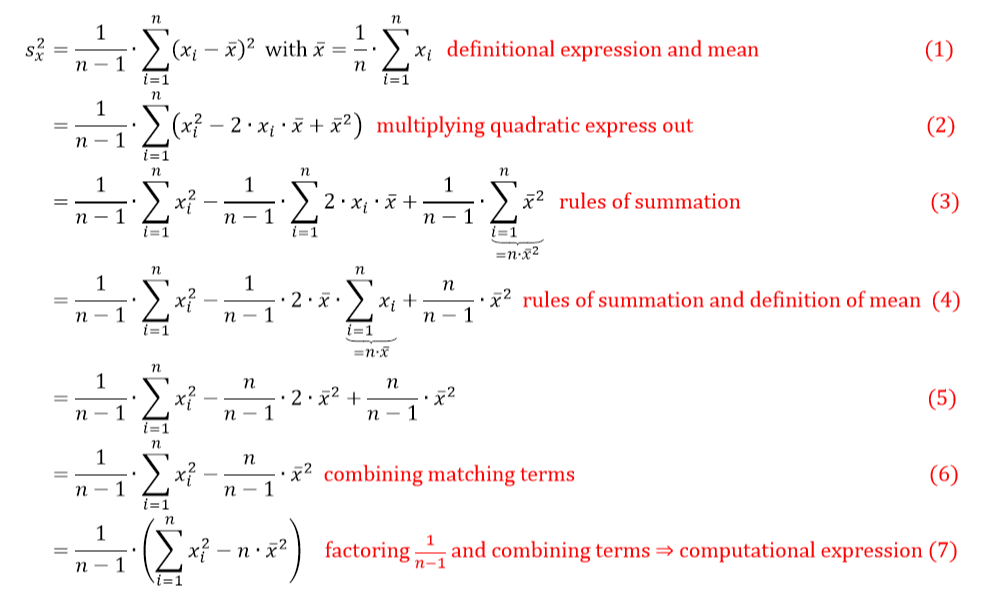


Figure 1: Definitional (1) and computational (7) expressions of the empirical variance estimators

Both formulas will be used in Task 3 to demonstrate speed and numerical stability issues of the definitional and the computational algorithms.

## Task 2: Summation rules applied in Task 1 (1 point)

Appendix 3a in Burt, Barber & Rigby (pp 148-149) reviews the use of the ***sigma*** summation-notation

The transformations in Task 1 make use of some algebraic rules. Assign the relevant rules to the equation lines. The learning objective here is that you understand the algebraic transformation

List of algebraic rules:

1. None of the given rules has been applied
2. with being a constant
3. with being a constant
4. A transformation of the arithmetic mean:

Which algebraic rules were applied to get to from one equation to the next? Justify your answer.

:

:

:

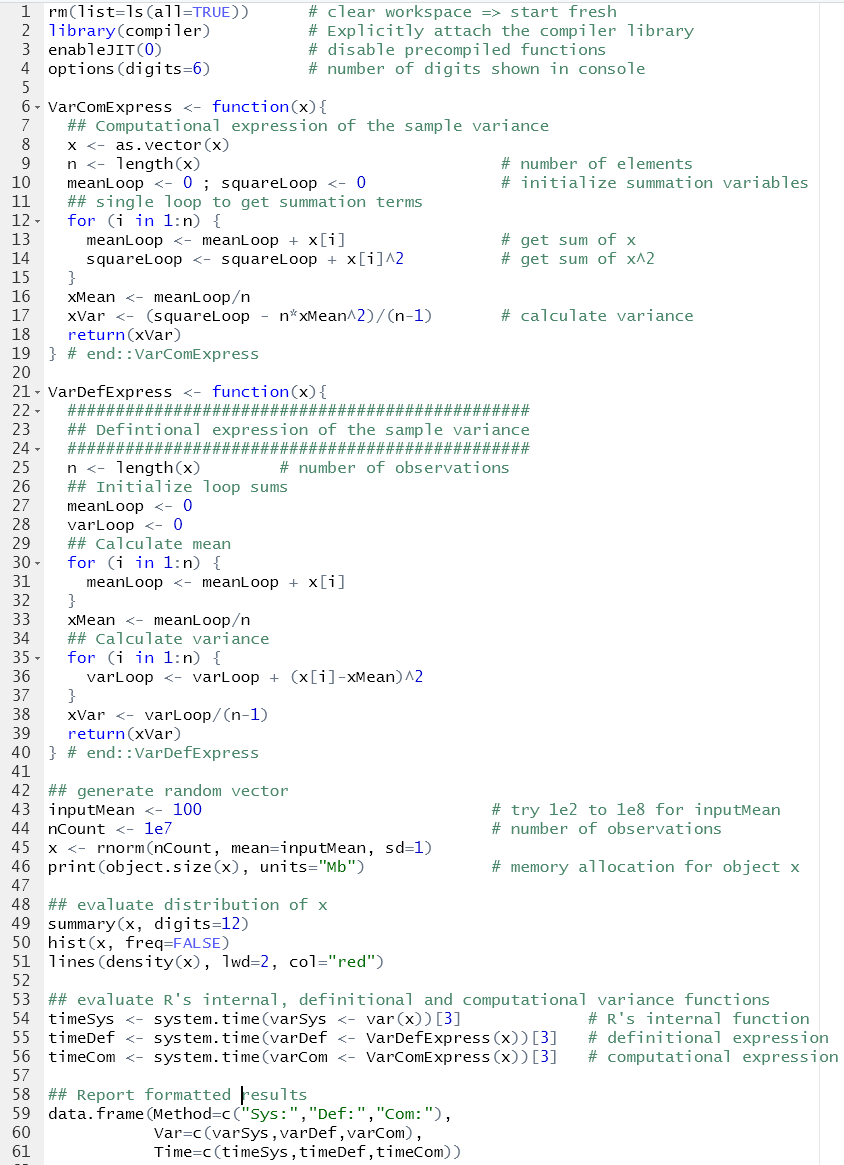
:

:

:

## Task 3: Computational Properties of Different Implementations of the Sample Variance Estimator (2 points)

You find below the  code for calculating the ***definitional expression*** and the ***computational expression*** of the variance including some code which tests these functions:



[a] Enter this code properly formatted into an  script. Show your properly formatted script. (1 point)

You can study loops, control statements and user functions in greater depth by looking at Lander’s *Chapter 8: Writing R functions*, *Chapter 9: Control Statements* and *Chapter 10: Loops, the Un-R Way to Iterate*.

[b] Why is it required to initialize the variables **meanLoop**, **squareLoop** and **varLoop** with zero in the functions **VarDefExpress( )** and **VarComExpress( )**? (0.1 points)

[c] What is the **rnorm( )** function doing? (0.1 points)

[d] What is the **system.time( )** function doing? (0.1 point)

[e] Run the script once with the variable **inputMean** set to 100 (line 43) and report the run times and calculated variances of all three variance functions in a table. (0.1 points)

[f] Run the script again with the variable **inputMean** set to 100,000,000 (line 43) and report the run times and calculated variances of all three variance functions in a table. (0.1 points)

[g] Why will the ***computational expression*** have a speed advantage? Think in terms of how many times the functions have to pass over the data. (0.25 point)

[h] In the ***computational expression*** both variables **xMean**, which was derived from **meanLoop** in line 16, and **squareLoop** are of different magnitudes on the floating point scale. Why does this have the potential to lead to rounding errors when combining both variables in line 17? (0.25 point)