107.330 Statistische Simulation und computerintensive Methoden

Computing a Variance

Alexandra Posekany

WS 2020

Statistical Simulation

Why is statistical simulation relevant? For what do we need computationally intertensive methods?

Buzz Words

- Artificial Intelligence
- Machine Learning
- Big Data
- Data Science

Differences between humans and computers

```
round(1.499999999999999)
```

```
## [1] 1
```

$$\neq$$

```
round(1.4999999999999999)
```

```
## [1] 2
```

What is relevant for simulation?

- programming environment
- implementation of formulae in Software systems, libraries/packages
- usage of memory

Example: Variance Calculation

The Variance σ_x^2 of a random variable x is defined as

$$\sigma_x^2 = E((x - E(x))^2).$$

For a sample $x_1, \ldots, x_n$, depending on the application, it is usually estimates as

$$s_x^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \quad \text{or} \quad s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2,$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the sample mean.

Variance Decomposition

The variance decomposition formula

$$\begin{aligned}
\mathbb{V}(X) &= \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] \\
&= \mathbb{E}\left[X^2 - 2X\mathbb{E}[X] + \mathbb{E}[X]^2\right] \\
&= \mathbb{E}\left[X^2\right] - 2\mathbb{E}[X]\mathbb{E}[X] + \mathbb{E}[X]^2 \\
&= \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2
\end{aligned}$$

results in an alternative expression for the sample variance

$$s_x^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} x_i\right)^2\right)$$

Computing the Finite Sample Variance

But how to compute the estimate with a computer?

Let us consider in the following the unbiased estimate.

For demonstration purposes implement the following four algorithms in R.

Algorithm 1 and Algorithm 2

- Algorithm 1 (two - pass algorithm - variance calculation in R):
  1. Compute the sample mean
  2. Compute the variance as defined above
- Algorithm 2 (one - pass algorithm - previously variance calculation in Excel):
  1. Compute $P_1 = \sum_{i=1}^{n} x_i^2$.
  2. Compute $P_2 = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2$.
  3. Compute $s_x^2 = (P_1 - P_2)/(n-1)$.

Scale Invariance property

The variance is a scale estimate and therefore invariant under location changes, i.e. $\sigma_x^2 = \sigma_{x-c}^2$ for any constant c.

```
x<-c(1:20)
var(x)
```

```
## [1] 35
```

```
y<-x+50
var(y)
```

```
## [1] 35
```

```
z<-x-100
var(z)
```

```
## [1] 35
```

Algorithm 3

- Algorithm 3 (shifted one - pass algorithm):
  1. Compute $P_1 = \sum_{i=1}^{n}(x_i - c)^2$.
  2. Compute $P_2 = \frac{1}{n}(\sum_{i=1}^{n}(x_i - c))^2$.
  3. Compute $s_x^2 = (P_1 - P_2)/(n-1)$.

Consider what would be a good value for c?

Take for the beginning $c = x_1$ as the initialisation.

Algorithm 4

The above algorithms are all so called **batch** algorithms which assume that all data is available. The following algorithm is a so called **online** algorithm which assumes that the data is either arriving one observation after another or that not all observations can be saved at the same location.

Denote $\bar{x}_k$ as the mean estimate when the first k observations are available and s_k^2 as the corresponding variance estimate.

Updating equations are then:

- $\bar{x}_n = \frac{(n-1)\bar{x}_{n-1}+x_n}{n} = \bar{x}_{n-1} + \frac{x_n - \bar{x}_{n-1}}{n}$
- $s_n^2 = \frac{n-2}{n-1} s_{n-1}^2 + \frac{(x_n - \bar{x}_{n-1})^2}{n}, n > 1$

Algorithm 4 II

Write an R implementation of the online algorithm where the function input is the n-vector x and starting with the first two observations the estimate is updated successively by including the other observations one after anonther.

- Algorithm 4 (online algorithm):
  1. Compute $\bar{x}_2$ and s_2^2 using the original definition.
  2. For i in $3, \ldots, n$ update $\bar{x}_i$ and s_i^2.
  3. Return s_n^2

Comparison

Compare the 4 algorithms with the results from the R function `var` for the following two data sets.

```
set.seed(1)
x1 <- rnorm(100)
set.seed(1)
x2 <- rnorm(100, mean=1000000)
```

Get familiar with the "functions" `==`, `identical` and `all.equal` and use them to compare the results.

Comparison II

Use the function `microbenchmark` from the `microbenchmark` package to compare the computation times of the five different functions when using `x1` as the input data.

Visualize the computation times using boxplots.

Would you know another way in R to compare computing times?

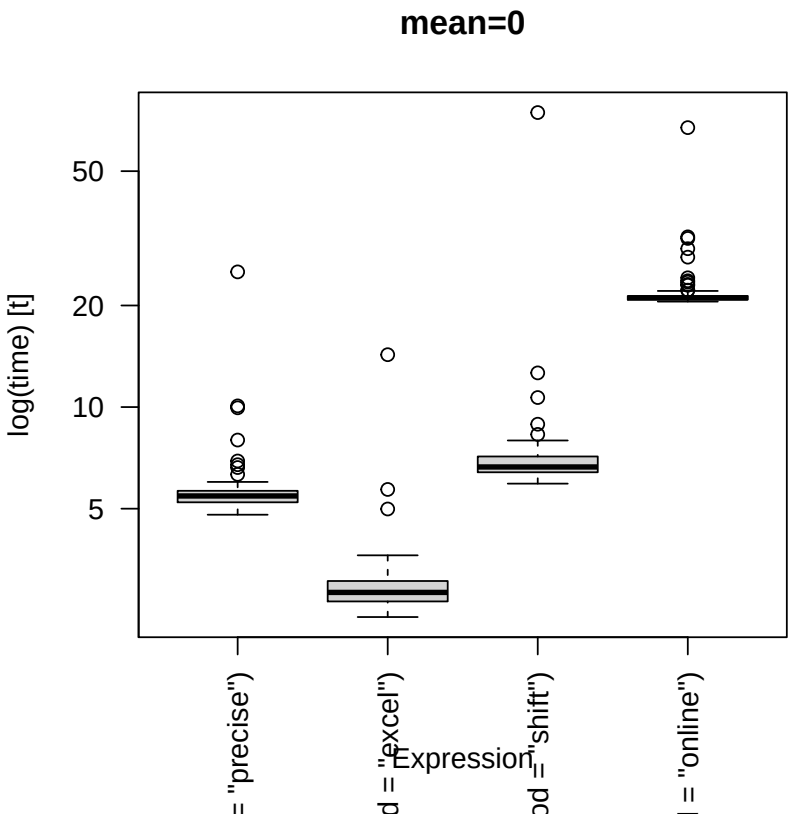
Comparison III

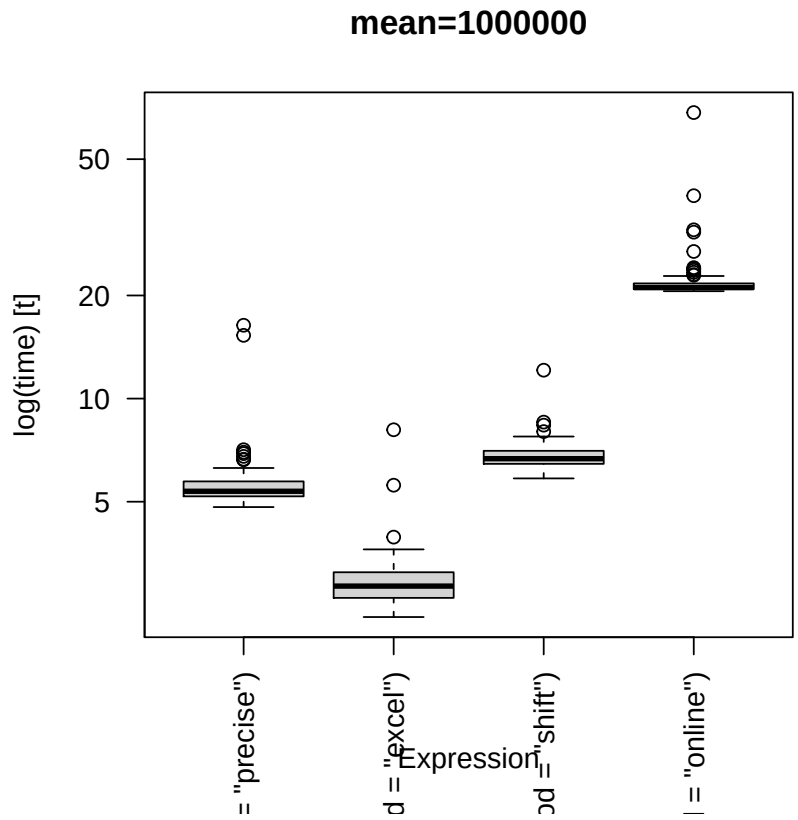
	precise		excel	shift	online
mean=0	0.81		0.83	0.81	0.81
mean=1000000	0.81	20202024601	51.81	0.81	0.81

	min	lq	mean	median	uq	max	neval
precise	5.41	6.86	13.27	9.08	13.28	57.42	100.00
excel	2.62	3.98	7.07	5.59	7.90	43.35	100.00
shift	6.99	8.57	17.69	11.42	20.19	100.09	100.00
online	21.35	23.55	38.01	26.28	39.62	165.02	100.00

	min	lq	mean	median	uq	max	neval
precise	5.10	5.87	7.54	6.64	8.45	16.94	100.00
excel	2.43	2.92	3.80	3.39	4.03	16.92	100.00
shift	6.27	7.47	10.42	8.66	11.01	88.67	100.00
online	20.80	21.86	25.48	23.23	25.89	94.04	100.00

Comparison IV





Take Home Message

Speed of calculation is not the only measure for quality!

The old Excel method is the fastest, but least robust with the greatest error made!

Condition Number

The condition number is an application of the derivative, and is formally defined as the value of the asymptotic worst-case relative change in output for a relative change in input, where f denotes the original problem and $\tilde{f}(x)$ denotes the algorithm to solve it. It measures the robustness or stability of an algorithm with respect to input values and their perturbances.

$$\lim_{\varepsilon \to 0} \sup_{\|\Delta x\| \leq \varepsilon} \frac{\|f(x) - \tilde{f}(x)\|}{\|\Delta x\|}$$

Condition Number for the Variance

The condition number for the variance is defined as

$$\kappa = \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{S}} = \sqrt{1 + \frac{\bar{x}^2 n}{S}},$$

where $S = \sum_{i=1}^{n}(x_i - \bar{x})^2$.

If S is "small" and $\bar{x}$ nonzero this is approximately

$$\kappa \approx \bar{x}\sqrt{\frac{n}{S}} = \frac{\bar{x}}{s_n}.$$

It always holds that $\kappa >= 1$.

Relate the comparisons of the algorithms above to the condition number.

Condition Number for Shifted Variance

Note that adjusting the condition number for the shifted algorithm we have

$$\tilde{\kappa} = \sqrt{1 + \frac{n}{S}(\bar{x} - c)^2},$$

hence if $|c - \bar{x}| < |\bar{x}|$ we see $\tilde{\kappa} < \kappa$. And hence $c = \bar{x}$ gives the best condition number.

Conclusion

Implementing statistical procedures requires understanding of the method to implement and knowledge about how computers perform their calculations.

The mathematically most convenient form to express a method might not be necessarily the best way to implement and compute it:

The best conditioned algorithm need not be the best performing one in pratice. Therefore, adding actual simulations and run-time performances to the spectrum of properties applied for evaluating the quality of an algorithm.