A formal methodology for the comparison of results from different software packages

Alexander Zlotnik

Department of Electronic Engineering. Technical University of Madrid. & Ramón y Cajal University Hospital
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- Closed source vs open source
- An example
- Summary
- Multi-core performance
Why?

- Multidisciplinary and multi-location research is becoming the norm for high-impact scientific publications.

- Sometimes one part of your team uses one software package (R, Python, MATLAB, Mathematica, etc) and the other part uses a different one (Stata, SAS, SPSS, etc).

- After a lot of frustrating testing, you realize that you are getting different results with the same operations and the same datasets.
Why?

- In other situations, you simply want to develop your own functionality for an existing software package or a full independent software package with some statistical functionality.

- Since you know Stata well, you decide to compare the output of your program with the one from Stata and **guess** what …?
Hi there

I have a STATA do file that analyses data from an excel spreadsheet and runs a few calculations and then outputs the results to a new excel spreadsheet.

I was looking for someone to convert this STATA do file into php or C++ code as quickly as possible.

thank you for your help

Sign in or Register to see more

Desired Skills

PHP, C++, Statistical Computing

Source: elance.com

This seems doable, but there are likely to be many hidden surprises when comparing results for both implementations.
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- Are the same numerical methods used?
- *Closed source vs open source*
- An example
- Summary
- Multi-core performance
Is the output really different?

- Are you using the same dataset?

- Is the data converted in the same way?

- Are there any issues with numerical precision?
Is the output really different?

- Are you using the same dataset?
- Sometimes someone changes the dataset but doesn’t rename it.

- When in doubt,
  - Is the number of records the same?
  - Compare the files side-by-side (in text files).
  - Perform an MD5 hash on both files. Is it identical? (http://onlinemd5.com)
MD5 file hash

Two files are identical if they have the same MD5 hash

OnlineMD5

MD5 & SHA1 Hash Generator For File

Generate and verify the MD5/SHA1 checksum of a file without uploading it.

Click to select a file, or drag and drop it here (max. 4GB).

Filename: No File Selected
File size: 0 Bytes
Checksum type: MD5, SHA1, SHA-256
File checksum:
Compare with:
Process:

[Compare, Pause, Stop buttons]
### Merge/diff Software

These software packages can be used to compare text file databases (CSV, etc).
Is the output really different?

- Is the data converted in the same way in both programs?
  
  - Dates
    - Usually (under the scenes or explicitly) they are converted into numerical values. Which ones?
    
    - Are there any *seasonality* variables? How are they defined?

- SPSS f.ex. is very different from Stata.
Is the output really different?

- Is the data converted in the same way in both programs?

  - Categorical / numerical variables
    - Some software may interpret a variable as numeric and other software may decide to make it categorical or allow both options.

```
regress intvar
regress i.intvar
```
Is the output really different?

- Is the data converted in the same way in both programs?

  - Categorical / numerical variables
    - How are categorical variables encoded?
      Is it “1” = “category A” or “2” = “category A”?
    - Which is the reference category?
      Stata allows this syntax:
      `logistic bx.cat_variable`

      But in other software packages, reference categories may be automatically defined.
Is the output really different?

- Compare outputs in these scenarios, in this order:
  - 1 numeric,
  - Several numeric,
  - 1 categorical,
  - Several categorical,
  - Mix of several categorical and numeric

This should give you an idea about which data conversion is working differently, if any.
Is the output really different?

- Issues with numerical precision
  - Aren’t current computers powerful and capable of very precise calculations?
    - Yes, but for performance reasons (related with computer architecture), allowable minimum and maximum values are limited.
  - Some programming languages do not have a proper implementation of the standards (IEEE 754).
Is the output really different?

- Issues with numerical precision

  - If you turn a double into a float, some precision may be lost. In some cases, you may even get a completely different result if there is an overflow (change of sign, for example).
Is the output really different?

- Issues with numerical precision
  - Another funny thing: how are computations done internally in a program? Are they using “double” variables or maybe special, higher precision variables (“long double” for example)?
  - But really, this should not be relevant, right?
Is the output really different?

- Issues with numerical precision

A

-----

B

B = small number, which may or may not be zero depending on numerical precision

Or maybe it should?
Is the output really different?

- Issues with numerical precision

Ariane 5 launch

failure
Is the output really different?

- **Ada** (a real-time programming language used for the control of the rocket’s inertial guidance system) did not have a proper implementation of the **IEEE Standard 754 for Binary Floating-Point Arithmetic**.

- Upon launch, sensors reported acceleration so strong that it caused **Conversion-to-Integer Overflow in software** intended for recalibration of the rocket’s inertial guidance (...)

- **Sources:**
  - [https://www.ima.umn.edu/~arnold/disasters/ariane5rep.html](https://www.ima.umn.edu/~arnold/disasters/ariane5rep.html)
  - How Java’s Floating-Point Hurts Everyone Everywhere, Prof. W. Kahan and Joseph D. Darcy, 2006
Станислав Евграфович Петров
Stanisláv Evgráfovič’ Petróv
Is the output really different?

- Issues with numerical precision
  
  - How can we detect this?
    
    - Examine extreme (min, max) values in a given dataset in both programs. Are they different?
    
    - Compute some simple operations on both software platform with extreme values and see if the result is the same.
    
    - If there are differences (rounding, for example), decide if they might affect some calculations.
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- Are the same numerical methods used?
- *Closed source vs open source*
- An example
- Summary
- Multi-core performance
Is the algorithm documented?

- Let’s use one apparently simple example.

- The calculation of percentiles should be very simple, right?

- Then, why, when using Excel I get one value and when I am using Stata I am getting a different one?
Title

[D] pctl — Create variable containing percentiles

Syntax

Create variable containing percentiles

```
  pctl [type] newvar = exp [if] [in] [weight] [, pctl_options]
```

Create variable containing quantile categories

```
xtile newvar = exp [if] [in] [weight] [, xtile_options]
```

Compute percentiles and store them in r()

```
  _pctl varname [if] [in] [weight] [, _pctl_options]
```
Is the algorithm documented?

Options

- nquantiles(#) specifies the number of quantiles. It computes percentiles corresponding to percentages $100 \frac{k}{m}$ for $k = 1, 2, \ldots, m - 1$, where $m = #$. For example, nquantiles(10) requests that the 10th, 20th, ..., 90th percentiles be computed. The default is nquantiles(2); that is, the median is computed.

- genp(newvar, ) (pctile only) specifies a new variable to be generated containing the percentages corresponding to the percentiles.

- altdef uses an alternative formula for calculating percentiles. The default method is to invert the empirical distribution function by using averages, $(x_i + x_{i+1})/2$, where the function is flat (the default is the same method used by summarize; see [R] summarize). The alternative formula uses an interpolation method. See Methods and formulas at the end of this entry. Weights cannot be used when altdef is specified.

- cutpoints(varname) (xtile only) requests that xtile use the values of varname, rather than quantiles, as cutpoints for the categories. All values of varname are used, regardless of any if or in restriction; see the technical note in the xtile section below.

- percentiles(numlist) (_pctile only) requests percentiles corresponding to the specified percentages. Percentiles are placed in r(r1), r(r2), ..., etc. For example, percentiles(10(20)90) requests that the 10th, 30th, 50th, 70th, and 90th percentiles be computed and placed into r(r1), r(r2), r(r3), r(r4), and r(r5). Up to 1,000 (inclusive) percentiles can be requested. See [P] numlist for the syntax of a numlist.
Is the algorithm documented?

Methods and formulas

The default formula for percentiles is as follows: Let $x_{(j)}$ refer to the $x$ in ascending order for $j = 1, 2, \ldots, n$. Let $w_{(j)}$ refer to the corresponding weights of $x_{(j)}$; if there are no weights, $w_{(j)} = 1$. Let $N = \sum_{j=1}^{n} w_{(j)}$.

To obtain the $p$th percentile, which we will denote as $x_{[p]}$, let $P = N p / 100$, and let

$$W_{(i)} = \sum_{j=1}^{i} w_{(j)}$$

Find the first index, $i$, such that $W_{(i)} > P$. The $p$th percentile is then

$$x_{[p]} = \begin{cases} \frac{x_{(i-1)} + x_{(i)}}{2} & \text{if } W_{(i-1)} = P \\ x_{(i)} & \text{otherwise} \end{cases}$$

When the altdef option is specified, the following alternative definition is used. Here weights are not allowed.

Let $i$ be the integer floor of $(n + 1) p / 100$; that is, $i$ is the largest integer $i \leq (n + 1) p / 100$. Let $h$ be the remainder $h = (n + 1) p / 100 - i$. The $p$th percentile is then

$$x_{[p]} = (1 - h)x_{(i)} + hx_{(i+1)}$$

where $x_{(0)}$ is taken to be $x_{(1)}$ and $x_{(n+1)}$ is taken to be $x_{(n)}$. 
Is the algorithm documented?

**Excel 2003**

**QUARTILE**

Returns the quartile of a data set. Quartiles often are used in sales and survey data to divide populations into groups. For example, you can use QUARTILE to find the top 25 percent of incomes in a population.

**Syntax**

`QUARTILE(array,quart)`

- **Array** is the array or cell range of numeric values for which you want the quartile value.
- **Quart** indicates which value to return.

<table>
<thead>
<tr>
<th>IF QUART EQUALS</th>
<th>QUARTILE RETURNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Minimum value</td>
</tr>
<tr>
<td>1</td>
<td>First quartile (25th percentile)</td>
</tr>
<tr>
<td>2</td>
<td>Median value (50th percentile)</td>
</tr>
<tr>
<td>3</td>
<td>Third quartile (75th percentile)</td>
</tr>
<tr>
<td>4</td>
<td>Maximum value</td>
</tr>
</tbody>
</table>

No information about exact procedure
Is the algorithm documented?

- It turns out, there are several ways...

**Estimating the quantiles of a population**  [edit]

There are several methods for estimating the quantiles. The most comprehensive breadth of methods is available in the R and GNU Octave programming languages, which include nine sample quantile methods. SAS includes five sample quantile methods, Scipy includes eight, and STATA includes two, and Microsoft Excel includes one.

In effect, the methods compute \( Q_p \), the estimate for the \( \theta \)-th \( q \)-quantile, where \( p = \) \( k / q \), from a sample of size \( N \) by computing a real-valued index \( h \). When \( h \) is an integer, the \( h \)-th smallest of the \( N \) values, \( x_h \), is the quantile estimate. Otherwise a rounding or interpolation scheme is used to compute the quantile estimate from \( h \), \( x_{\lfloor h \rfloor} \), and \( x_{\lceil h \rceil} \). (For notation, see floor and ceiling functions.)

Estimate types include:

<table>
<thead>
<tr>
<th>Type</th>
<th>( n )</th>
<th>( Q_p )</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-1, SAS-3</td>
<td>( Np + 1/2 )</td>
<td>( x_{\lfloor h - 1/2 \rfloor} )</td>
<td>Inverse of empirical distribution function. When ( p = 0 ), use ( x_1 ).</td>
</tr>
<tr>
<td>R-2, SAS-5</td>
<td>( Np + 1/2 )</td>
<td>( x_{\lfloor h - 1/2 \rfloor} + x_{\lfloor h + 1/2 \rfloor} / 2 )</td>
<td>The same as R-1, but with averaging at discontinuities. When ( p = 0 ), use ( x_1 ). When ( p = 1 ), use ( x_N ).</td>
</tr>
<tr>
<td>R-3, SAS-2</td>
<td>( Np )</td>
<td>( x_{\lfloor h \rfloor} )</td>
<td>The observation numbered closest to ( Np ). Here, ( \lfloor h \rfloor ) indicates rounding to the nearest integer, choosing the even integer in the case of a tie. When ( p \leq (1/2) / M ), use ( x_1 ).</td>
</tr>
<tr>
<td>R-4, SAS-1, Scipy(0,1)</td>
<td>( Np )</td>
<td>( x_{\lfloor h \rfloor} + (h - \lfloor h \rfloor)(x_{\lfloor h \rfloor+1} - x_{\lfloor h \rfloor}) )</td>
<td>Linear interpolation of the empirical distribution function. When ( p &lt; 1 / M ), use ( x_1 ). When ( p = 1 ), use ( x_N ).</td>
</tr>
</tbody>
</table>

Is the algorithm documented?

<table>
<thead>
<tr>
<th>Before Office 2010</th>
<th>Excel</th>
<th>R</th>
<th>SAS</th>
<th>Minitab</th>
<th>Maple</th>
<th>JMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 4 (Linear CDF)</td>
<td></td>
<td>type=4</td>
<td>PCTLDEF=1</td>
<td></td>
<td>method=3</td>
<td></td>
</tr>
<tr>
<td>Method 5 (Piecewise Linear)</td>
<td></td>
<td>type=5</td>
<td></td>
<td></td>
<td>method=4</td>
<td></td>
</tr>
<tr>
<td>Method 6 (N+1)</td>
<td></td>
<td>type=6</td>
<td>PCTLDEF=4</td>
<td>Default</td>
<td>method=5</td>
<td>Default</td>
</tr>
<tr>
<td>Method 7 (N-1)</td>
<td>QUARTILE</td>
<td>type=7</td>
<td></td>
<td></td>
<td>method=6</td>
<td></td>
</tr>
<tr>
<td>Method 8 (N+1/3)</td>
<td></td>
<td>type=8</td>
<td></td>
<td></td>
<td>method=7</td>
<td></td>
</tr>
<tr>
<td>Method 9 (N + 1/4)</td>
<td></td>
<td>type=9</td>
<td></td>
<td></td>
<td>method=8</td>
<td></td>
</tr>
</tbody>
</table>

Source: Bacon Bits blog
Is the algorithm documented?

<table>
<thead>
<tr>
<th>Method</th>
<th>Excel</th>
<th>R</th>
<th>SAS</th>
<th>Minitab</th>
<th>Maple</th>
<th>JMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method 4</td>
<td></td>
<td>type=4</td>
<td>PCTLDEF=1</td>
<td></td>
<td>method=3</td>
<td></td>
</tr>
<tr>
<td>(Linear CDF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method 5</td>
<td></td>
<td>type=5</td>
<td></td>
<td>.method=4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Piecewise Linear)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method 6</td>
<td>QUARTILE.EXC</td>
<td>type=6</td>
<td>PCTLDEF=4</td>
<td>Default</td>
<td>method=5</td>
<td>Default</td>
</tr>
<tr>
<td>(N+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method 7</td>
<td>QUARTILE</td>
<td>type=7</td>
<td></td>
<td>method=6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(N-1)</td>
<td>INC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method 8</td>
<td>QUARTILE</td>
<td>type=8</td>
<td></td>
<td>method=7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(N+1/3)</td>
<td>INC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method 9</td>
<td>QUARTILE</td>
<td>type=9</td>
<td></td>
<td>method=8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(N + 1/4)</td>
<td>INC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: Bacon Bits blog
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- *Closed source vs open source*
- An example
- Summary
- Multi-core performance
Closed source vs Open source

- Open source:
  - Usually it’s **free to use**, at least for non-profit purposes.
  - You can **modify the software**, right to the core. This is sometimes very handy (ex: eliminate performance bottleneck for a specific problem, introduce new features, *et cetera*).
  - You can see exactly what the program is doing examining the source code, which is useful because **documentation is often very basic**.
Closed source vs Open source

- Closed source:
  - Most often not free.
  - Usually better usability.
  - Usually much better documentation. However, if the documentation is lacking, you are at the expense of support personnel.

- Some can be described as a hybrid case: the most important Stata functions are written in C/C++ and source code is not available for these; however all the stuff written in Stata language and Mata is readily available.
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- *Closed source vs open source*
- An example
- Summary
- Multi-core performance
An example

An example

- Weka: open source, Java-based, implements many machine learning algorithms (ANNs, SVMs, bayesian networks, regression & classification trees, et cetera).

- Problem: for some weird reason it lacks easy to interpret calibration metrics!
Why is calibration important?

- Let’s suppose we have a classifier for a binary outcome with a reasonably good discrimination (area under ROC = 0.87 for example).

- But this classifier produces only two outputs: 0 and 1
Area under the ROC curve

Sensitivity = TPR

FPR = 1 – Specificity = 1 – TNR
Why is calibration important?

- For most binary decision support systems we need a **probabilistic output** (estimation of event probability).

- Is the **predicted** probability of 90% the **real** probability of 90%?

- Ideally, calibration should compare **predicted probabilities** with **real underlying probabilities**.

- As the latter are usually unknown, the most common approach to assess calibration is to compare **expected** (predicted) and **observed** (actual) outcomes by groups.
**Calibration**

\[ \chi^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}} \]

\[ \chi^2 = \frac{(1 - 0.5)^2}{0.5} + \frac{(1 - 1.4)^2}{1.3} + \frac{(3 - 3.0)^2}{3.0} \]

<table>
<thead>
<tr>
<th>Predicted (expected)</th>
<th>Actual (observed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0</td>
</tr>
<tr>
<td>0.2</td>
<td>1</td>
</tr>
<tr>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>0.8</td>
<td>1</td>
</tr>
<tr>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>1</td>
</tr>
</tbody>
</table>
Hosmer-Lemeshow “deciles of risk”

\[ C = \sum_{k=1}^{g} \left[ \frac{(o_{1k} - e_{1k})^2}{e_{1k}} + \frac{(o_{0k} - e_{0k})^2}{e_{0k}} \right] \]

p-value = 1 - P(\chi^2 \leq C)

\( g = \) number of groups
An example of bad calibration

Classifier = C4.5
An example of better calibration

Classifier = logistic regression
But our results were different from those obtained in Stata...

- Reason 1: Weka did not implement decile computation. We had to use the *Apache Commons Math* library, which implements percentile calculation in a different way.

- Reason 2: Weka does not drop observations for collinearity reasons (or perfect predictors) when performing a logistic regression. Stata, by default, does.

- However, *logistic regression coefficients are the same in both programs* (if both datasets are carefully examined and identical variable encoding is used).
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- Are the same numerical methods used?
- Closed source vs open source
- An example
- Summary
- Multi-core performance
Summary (1/4)

- Is the output really different?
  - MD5, merge / diff, visual inspection
  - Date -> Numeric conversion
  - String -> Categorical conversion
  - Categorical variable encoding
  - Integer vs Float vs Double vs Other numeric formats (f.ex. Long Double)
Summary (2/4)

- Compare outputs in the two programs with different datasets:
  - 1 numeric,
  - several numeric,
  - 1 categorical,
  - several categorical,
  - mix of several categorical and numeric.
Summary (3/4)

- Is the algorithm documented?
  - Check documentation
    - Formal: help files, official support
    - Informal: forums, blogs
  - If possible, check the source code
Summary (4/4)

- Generally,

  - If the differences are very large, you are most likely doing something wrong:
    - different dataset
    - different variable transformation, encoding, *et cetera*
    - different algorithm
    - numerically unstable problem (f.ex. usage of perfect predictors)
Structure of Presentation

- Why?
- Is the output really different?
- Is the algorithm documented?
- Are the same numerical methods used?
- *Closed source vs open source*
- An example
- Summary
- Multi-core performance
Multi-core performance

Big data (usually) requires Big iron
Multi-core performance

- Multi-core and multi-node processing should speed up operations, right?

- Only if:
  - The operation can be parallelized and is parallelized (Ahmdal’s law)
  - The overhead incurred (child process or thread creation) is not large
Given:
- $n \in \mathbb{N}$, the number of threads of execution.
- $B \in [0, 1]$, the fraction of the algorithm that is strictly serial.

The time $T(n)$ an algorithm takes to finish when being executed on $n$ thread(s) of execution corresponds to:

$$T(n) = T(1) \left( B + \frac{1}{n} (1-B) \right)$$

Therefore, the theoretical speedup $S(n)$ that can be had by executing a given algorithm on a system capable of executing $n$ threads of execution is:

$$S(n) = \frac{T(1)}{T(n)} = \frac{T(1)}{T(1) \left( B + \frac{1}{n} (1-B) \right)} = \frac{1}{B + \frac{1}{n} (1-B)}$$

Short version: Stata MP 12-core isn't running my code any faster than it did when I used Stata IC, and I can't figure out why.

Detailed version: I am running Windows 7 Pro SP1 64-bit on a quad-core machine. I have purchased two Stata licenses. I purchased Intercooled when version 12 was released. I recently purchased MP-12 core to make my Stata code run faster. (I realize I only have four cores so the 12 core is overkill; I want the flexibility to use Amazon's EC2, so I purchased the 12 core version.) Both flavors of Stata are version 12.

Unfortunately, I am finding that MP does not run any faster than IC. Indeed, in all my tests MP is a little slower. To document the issue, I did a fresh install of Stata Intercooled and then I ran benchmark.do (below). I ran it three times, and the average run time was 18.4 seconds. Then I uninstalled Stata completely, installed Stata MP-12 core, and ran benchmark.do again. I ran it three times, and the average was 19.6 seconds. I'm disappointed that MP isn't running faster.

An interesting discussion on Stata List
Dear Ted, I've witnessed many times that MP works much faster the IC. The figures in the report do make sense. No looking at your example: the only parallelizable part here is the "regress mpg weight gear foreign." Two things to notice immediately are the following:

1) the dataset contains 74 observations. The overhead of parallelizing it into 12 CPUs or even 4 CPUs is large relative to the size of the task at hand. You are likely to see the benefits of parallelization when you -expand- your dataset, say 1000000 (10^6) times and perhaps reduce the number of bootstrap iterations.

2) the dataset contains 74 observations. So the _regress command (internal) takes, say, 0.00001 second and with parallelization takes may be 0.00001 second, but then you have 2 seconds of writing the output to the screen and scrolling the output window. That is not parallelized (correct me if I am wrong), though scrolling seems to work much faster in recent versions (THANKS!) So, try disabling the output with -quietly- and you will see more performance gain from MP.

3) finally, Stata's ado files seem to not be parallelizable (you don't write them that way), but only internal commands are. There have been some changes in the most recent versions and the idea is to permit the users to write parallel code. I am yet to see these facilities, but it makes no sense to test parallelization benefits on do/ado code or where such code executes for a significant amount of time. This is also a reason while there is no need to separately benchmark bootstrap commands.