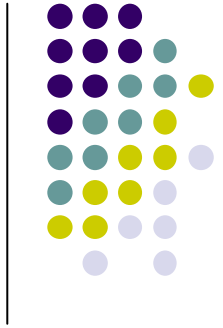
2014 Spanish Stata Users Group meeting

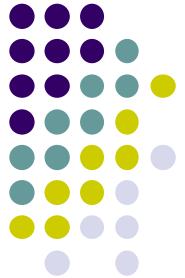


A formal methodology for the comparison of results from different software packages

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

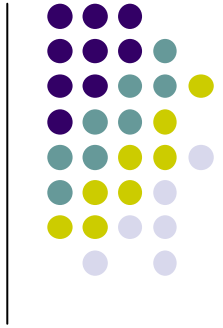
Why?

- 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 …?**

Convert STATA do file to php or C++

Sign in to Add to Watch List

IT & Programming > Other IT & Programming

Posted: Sun, Sep 14, 2014
Time Left: Closed
Location: Anywhere
Start: Immediately

Hourly Rate: Not Sure
Hrs/wk: Not Sure | Duration: 1-2 week:
Work View™ Payment Protection
W9 Not Required

Less Detail

Sign in to view client's details

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

Job ID: 62189981 Report Violation

Sign in to Elance and start working on jobs today.

Sign in to view more of the job details and submit a proposal. Once registered, you'll have access to thousands of jobs online or through email.

Submit a Proposal

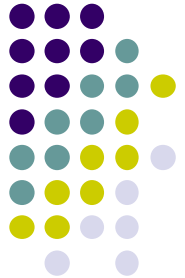
Are you ready to post a job like this one?

Post a Similar Job »

Source: elance.com

This seems doable, but there are likely to be many hidden surprises when comparing results for both implementationsc

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?

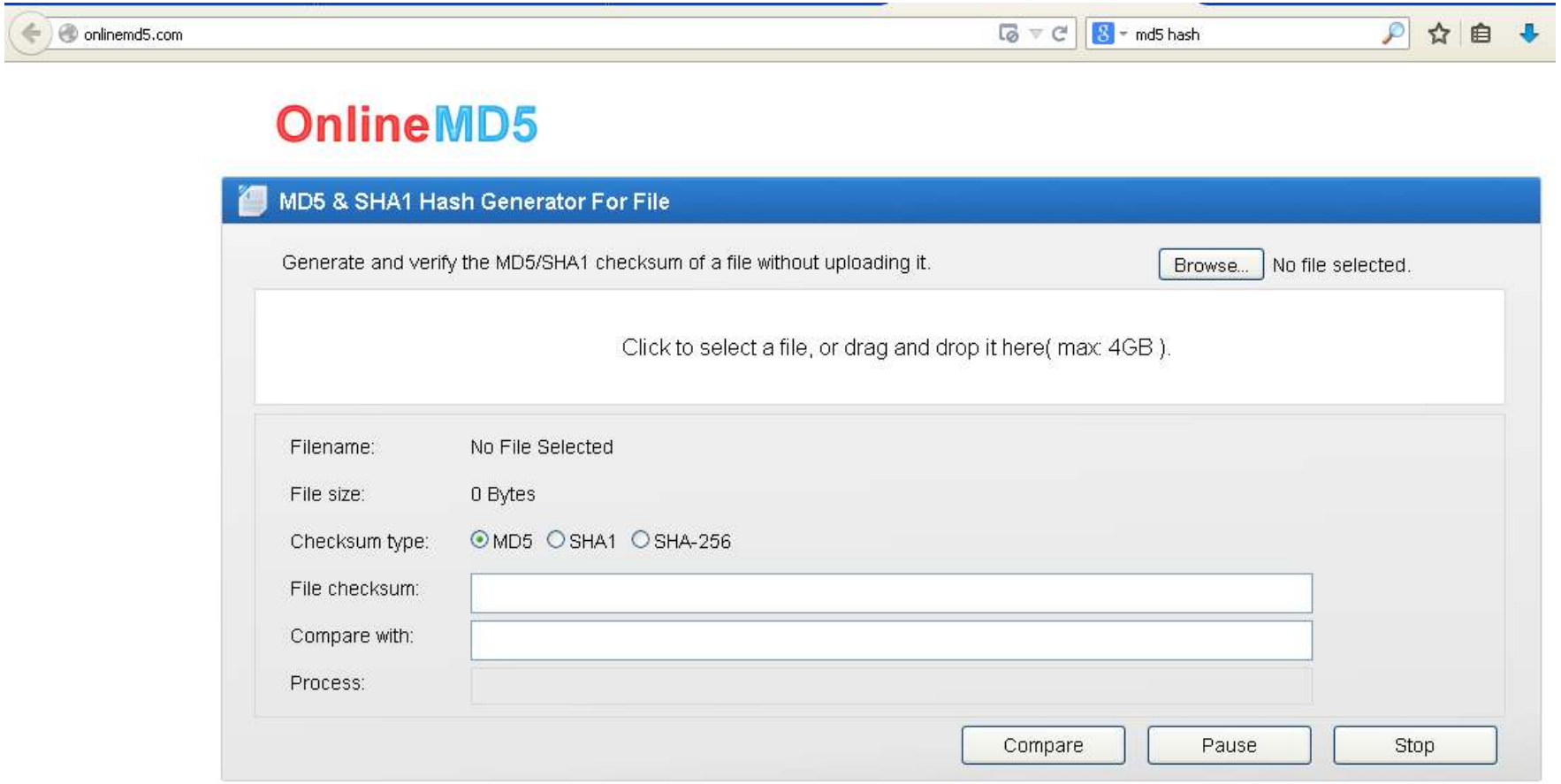
- Are you using the same dataset?
- Is the data converted in the same way?
- Are there any issues with numerical precision?

Is the output really different?

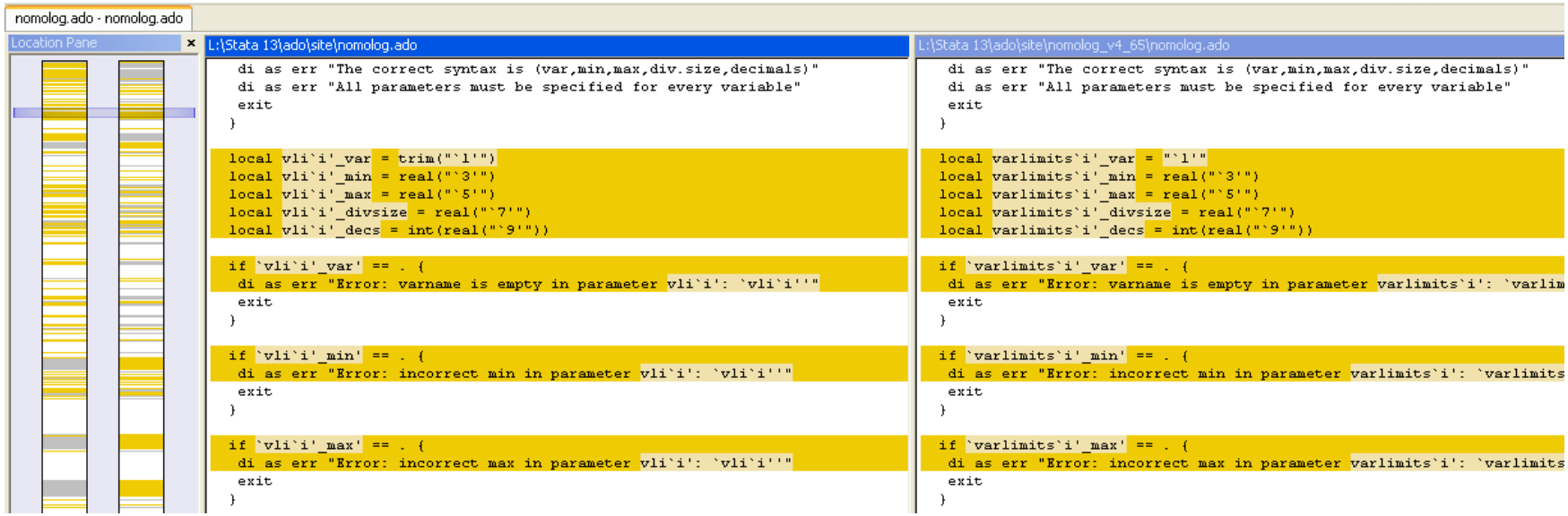
- **Are you using the same dataset?**
- Sometimes someone changes the dataset but doesn't rename it.
- When in doubt,
  - Is the number of records the same?
  - Compare the files side-by-side (in text files).
  - Perform an MD5 hash on both files. Is it identical? (http://onlinemd5.com)

MD5 file hash

Two files are identical if they have the same MD5 hash



“Merge/diff” software

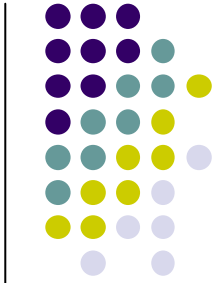


These software packages can be used to compare text file databases (CSV, etc)

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?

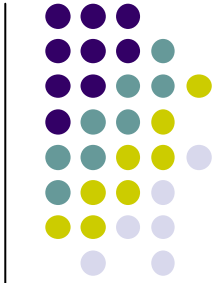


- **Is the data converted in the same way in both programs?**
  - Categorical / numerical variables
    - Some software may interpret a variable as numeric and other software may decide to make it categorical or allow both options.

      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?

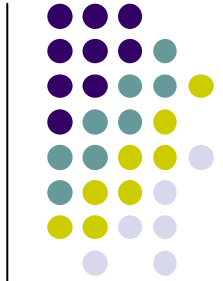
- Compare outputs in these scenarios, in this order:
  - 1 numeric,
  - Several numeric,
  - 1 categorical,
  - Several categorical,
  - Mix of several categorical and numeric

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**

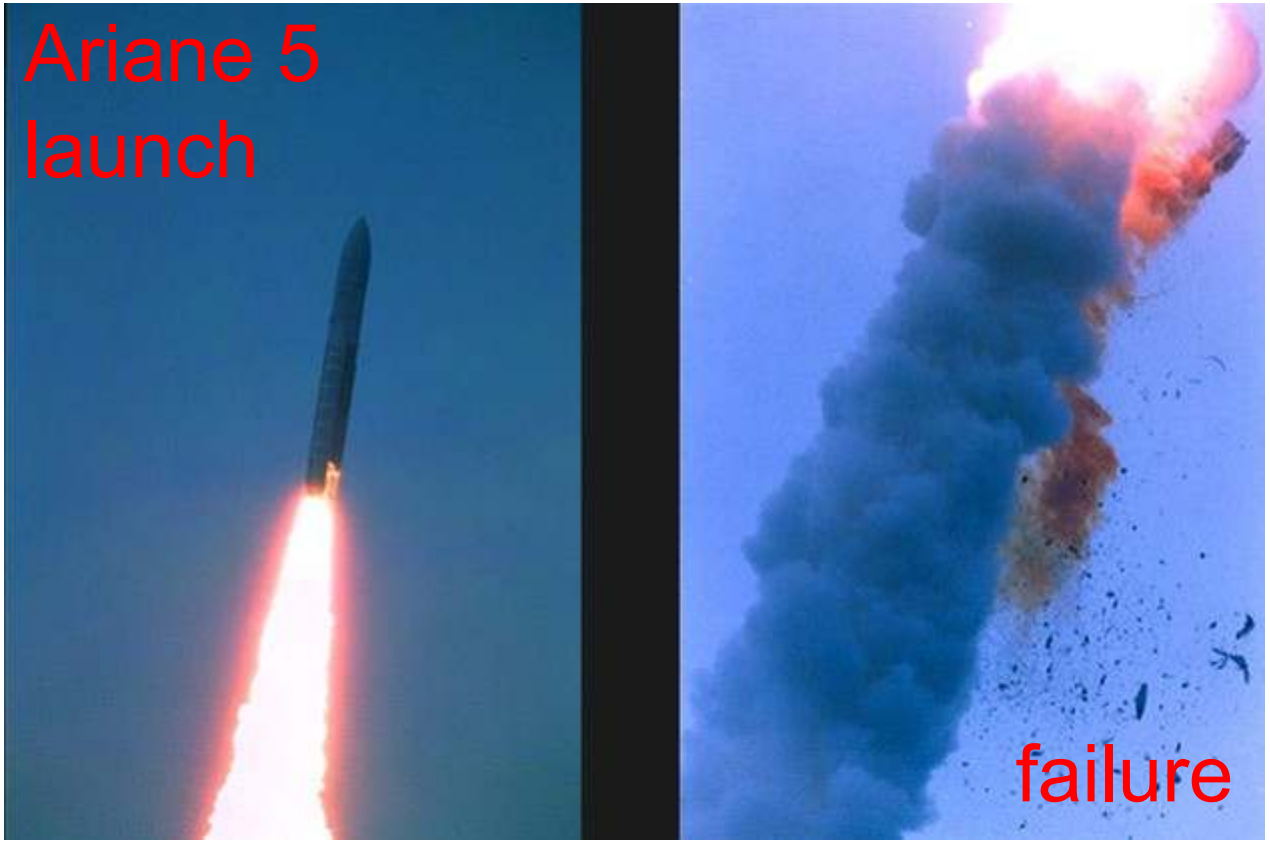
$$\frac{A}{B}$$

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

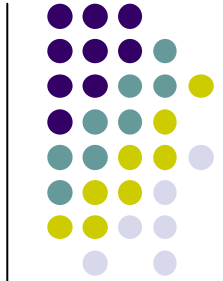
Or maybe it should?

Is the output really different?

- **Issues with numerical precision**



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 (...)

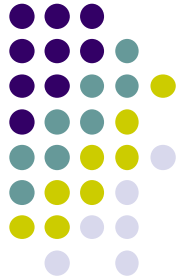
- Sources:
  - https://www.ima.umn.edu/~arnold/disasters/ariane5rep.html
  - How Java's Floating-Point Hurts Everyone Everywhere, Prof. W. Kahan and Joseph D. Darcy, 2006

Станислав Евграфович Петров
Stanisláv Evgráfovich’ 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

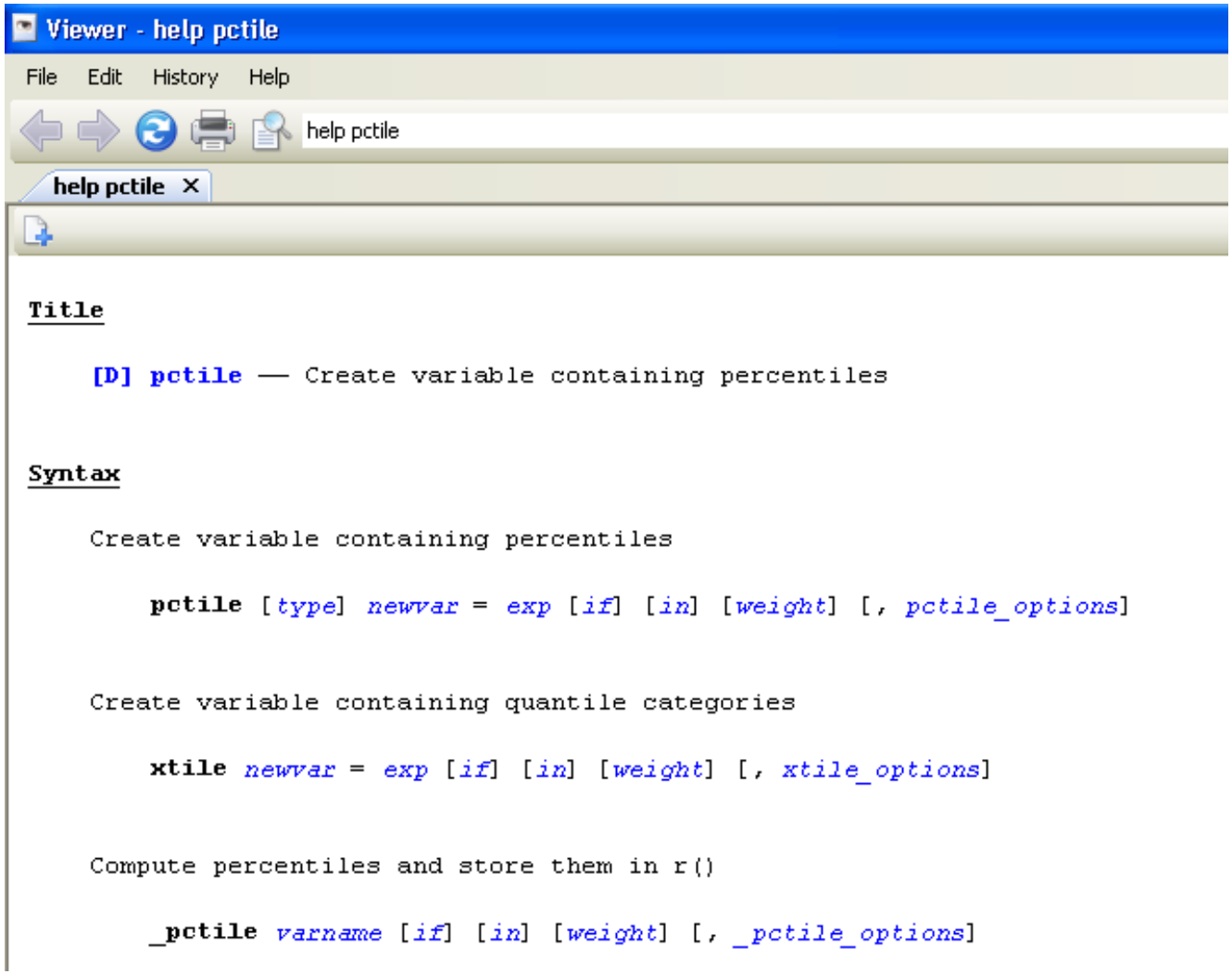


- Why?
- Is the output really different?
- Is the algorithm documented?
- Are the same numerical methods used?
- *Closed source* vs *open source*
- An example
- Summary
- Multi-core performance

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?

Is the algorithm documented?



Is the algorithm documented?

Options

Main

`nquantiles(#)` specifies the number of quantiles. It computes percentiles corresponding to percentages $100\,k/m$ 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*$_p$`)` (`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 pth percentile, which we will denote as $x_{[p]}$, let $P = Np/100$, and let

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

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

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

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 pth 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.

+ Show All

Applies to:
Excel 2003

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.

IF QUART EQUALS	QUARTILE RETURNS
0	Minimum value
1	First quartile (25th percentile)
2	Median value (50th percentile)
3	Third quartile (75th percentile)
4	Maximum value

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.[1] The most comprehensive breadth of methods is available in the R and GNU Octave programming languages, which include nine sample quantile methods.[2][3] SAS includes five sample quantile methods, SciPy includes eight,[4] STATA includes two, and Microsoft Excel includes one.

In effect, the methods compute Q_p, the estimate for the kth 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 hth 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:

Type	h	Q_p	Notes
R-1, SAS-3	$Np + 1/2$	$x_{\lceil h - 1/2 \rceil}$	Inverse of empirical distribution function. When $p = 0$, use x_1.
R-2, SAS-5	$Np + 1/2$	$\frac{x_{\lceil h - 1/2 \rceil} + x_{\lfloor h + 1/2 \rfloor}}{2}$	The same as R-1, but with averaging at discontinuities. When $p = 0$, x_1. When $p = 1$, use x_N.
R-3, SAS-2	Np	$x_{\lfloor h \rceil}$	The observation numbered closest to Np. Here, $\lfloor h \rceil$ indicates rounding to the nearest integer, choosing the even integer in the case of a tie. When $p \leq (1/2) / N$, use x_1.
R-4, SAS-1, SciPy-(0,1)	Np	$x_{\lfloor h \rfloor} + (h - \lfloor h \rfloor)(x_{\lfloor h \rfloor + 1} - x_{\lfloor h \rfloor})$	Linear interpolation of the empirical distribution function. When $p < 1 / N$, use x_1. When $p = 1$, use x_N.

Source: Wikipedia

Is the algorithm documented?

Before Office 2010

	Excel	R	SAS	Minitab	Maple	JMP
Method 4 (Linear CDF)		type=4	PCTLDEF=1		method=3	
Method 5 (Piecewise Linear)		type=5			method=4	
Method 6 (N+1)		type=6	PCTLDEF=4	Default	method=5	Default
Method 7 (N-1)	QUARTILE	type=7			method=6	
Method 8 (N+1/3)		type=8			method=7	
Method 9 (N + 1/4)		type=9			method=8	

Source: Bacon Bits blog

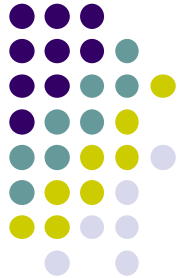
Is the algorithm documented?

After Office 2010

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

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

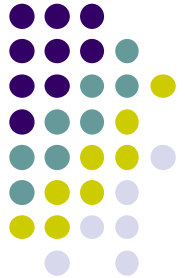
- Open source:
  - Usually it's free to use, at least for non-profit purposes.
  - You can modify the software, right to the core. This is sometimes very handy (ex: eliminate performance bottleneck for a specific problem, introduce new features, *et cetera*).
  - You can see exactly what the program 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.*

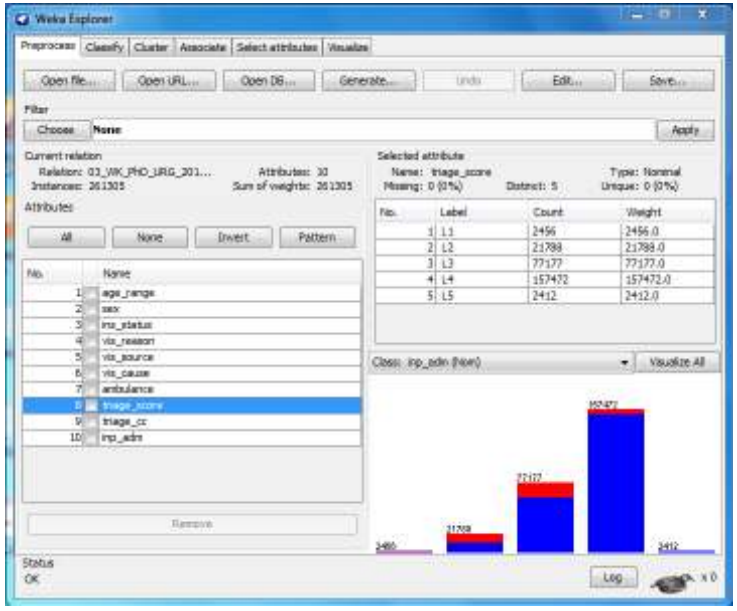
Structure of Presentation

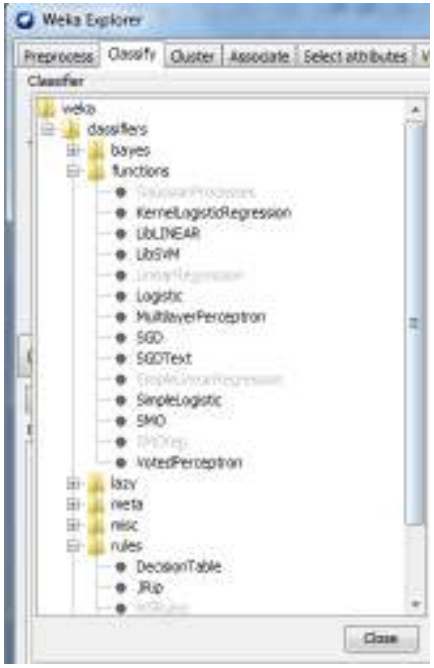


- Why?
- Is the output really different?
- Is the algorithm documented?
- *Closed source* vs *open source*
- An example
- Summary
- Multi-core performance

An example

- Implementation of Hosmer Lemeshow deciles of risk for the Weka data mining software for binary problems.





An example

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

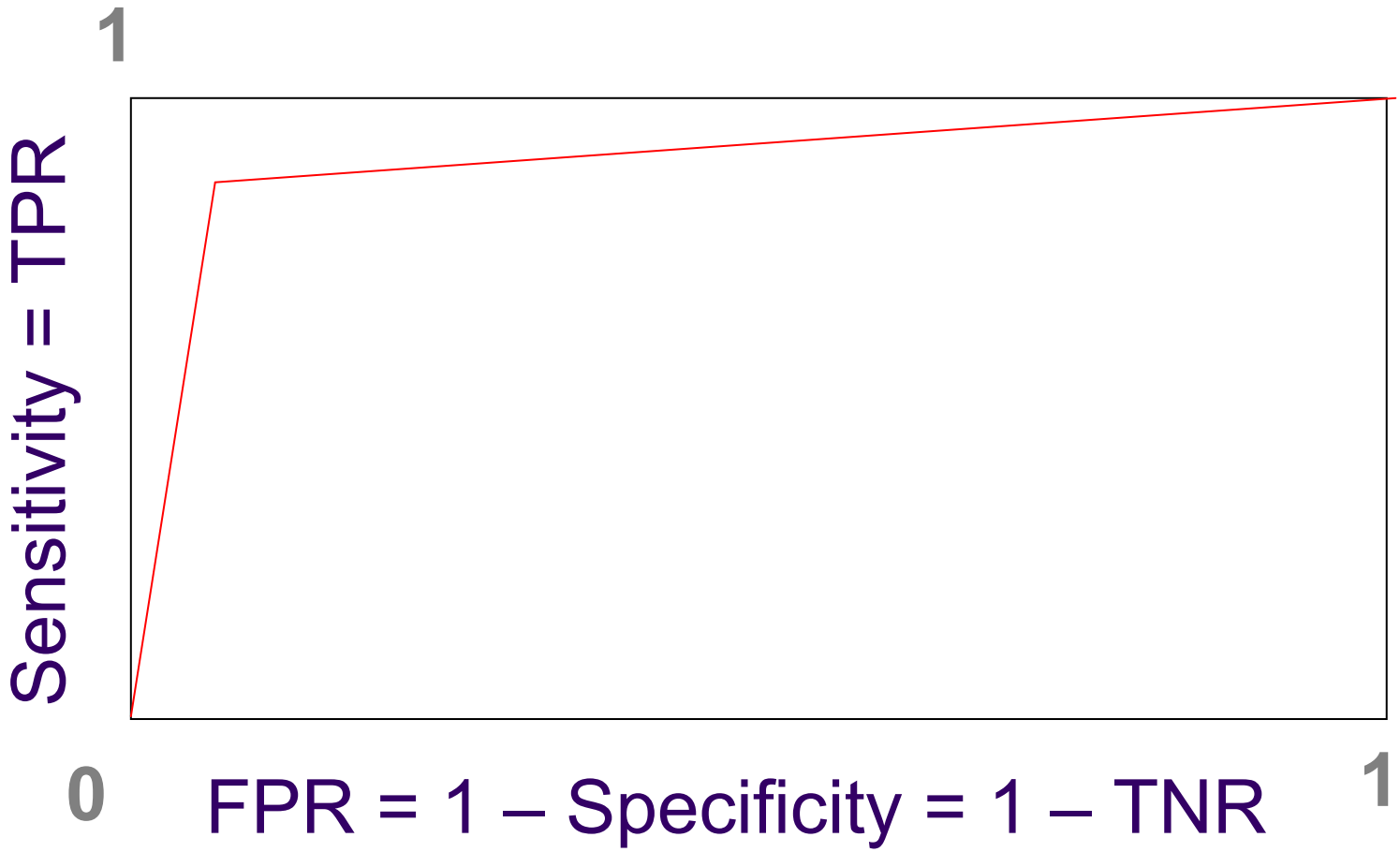
- Problem: for some weird reason it lacks easy to interpret calibration metrics!

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

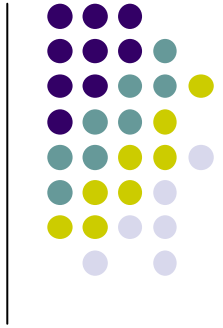
Area under the ROC curve



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



$\chi 2 = \Sigma\ [(\text{observed} - \text{expected})^2/\text{expected}]$

$\chi 2 = (1 - 0.5)^2 / 0.5 + (1 - 1.4)^2 / 1.3 + (3 - 3.0)^2 / 3.0$

Predicted (expected)		*Actual (observed)*	
0.0		0	
0.2		1	
0.3	sum of group = 0.5	0	sum = 1
0.4		0	
0.5		0	
0.5	sum of group = 1.4	1	sum = 1
0.5		1	
0.8		1	
0.8		0	
0.9	sum of group = 3.0	1	sum = 3

Hosmer-Lemeshow "deciles of risk"

o_{1k} observed frequencies in group k when the outcome variable is 1

e_{1k} expected frequencies in group k when the outcome variable is 1

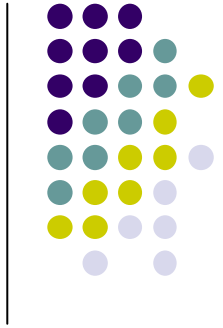
o_{0k} observed frequencies in group k when the outcome variable is 0

e_{0k} expected frequencies in group k when the outcome variable is 0

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

$$\text{p-value} = 1 - \mathrm{P}(\chi^2 \leq C)$$

g = number of groups



```
=== Summary ===

Correctly Classified Instances       78634               88.5079 %
Incorrectly Classified Instances     10210               11.4921 %
Kappa statistic                          0.3058
Mean absolute error                      0.1798
Root mean squared error                  0.3003
Relative absolute error                 77.9345 %
Root relative squared error             88.3541 %
Coverage of cases (0.95 level)          97.7601 %
Mean rel. region size (0.95 level)      69.3868 %
Total Number of Instances            88844

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.985    0.766    0.893      0.985    0.937      0.363    0.799     0.946     N
                 0.234    0.015    0.710      0.234    0.352      0.363    0.799     0.416     Y
Weighted Avg.    0.885    0.666    0.869      0.885    0.859      0.363    0.799     0.875
```

```
=== Calibration metrics ===
Group    Cut_Point Exp(Y=1)     Obs(Y=1)     Exp(Y=0)     Obs(Y=0)        Total
   1       0.7826  22479.1      22533         9467.9       9414          31947
   2       0.7826        0          0              0          0              0
   3       0.7826        0          0              0          0              0
   4       0.9636  53618.1      53588         2376.9       2407          55995
   5       0.9636        0          0              0          0              0
   6       0.9636        0          0              0          0              0
   7       0.9636        0          0              0          0              0
   8       0.9636        0          0              0          0              0
   9       0.9636        0          0              0          0              0
  10       1           893.3        877            8.7         25            902
```

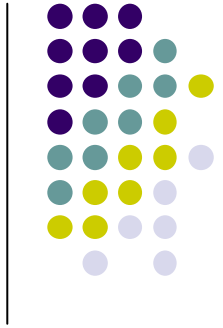
```
Chi-square = NaN
p-value = NaN
Class Y=1: N
Class Y=0: Y
```

An example of bad calibration

Classifier = C4.5

```
=== Confusion Matrix ===

     a     b   <-- classified as
 75865  1133 |     a =  N
  9077  2769 |     b =  Y
```



```
=== Summary ===

Correctly Classified Instances       78703               88.5856 %
Incorrectly Classified Instances     10141               11.4144 %
Kappa statistic                          0.361
Mean absolute error                      0.1666
Root mean squared error                  0.2887
Relative absolute error                 72.1808 %
Root relative squared error             84.9268 %
Coverage of cases (0.95 level)          98.9228 %
Mean rel. region size (0.95 level)      74.7681 %
Total Number of Instances            88844

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.975    0.696    0.901      0.975    0.937      0.394    0.864     0.973     N
                 0.304    0.025    0.655      0.304    0.415      0.394    0.864     0.540     Y
Weighted Avg.    0.886    0.607    0.868      0.886    0.867      0.394    0.864     0.915
```

```
=== Calibration metrics ===
Group     Cut_Point Exp(Y=1)      Obs(Y=1)      Exp(Y=0)      Obs(Y=0)          Total
   1        0.5926   3790.3        3874          5103.7        5020             8894
   2        0.7582   6204.3        6029          2806.7        2982             9011
   3        0.8752   7219          7241          1529          1507             8748
   4        0.9258   8229          8209           853           873             9082
   5        0.9501   9209.8        9231           578.2         557             9788
   6        0.9666   9731.4        9749           400.6         383            10132
   7        0.9764   8107          8126           229           210             8336
   8        0.9826   6945.3        6935           139.7         150             7085
   9        0.9921   8818.4        8807           115.6         127             8934
  10        0.9996   8793.6        8797            40.4          37             8834
```

```
Chi-square = 25.46972556142363
p-value = 0.001293668753559074 (<0.05)
Class Y=1: N
Class Y=0: Y
```

An example of better calibration

Classifier = logistic regression

```
=== Confusion Matrix ===

     a     b   <-- classified as
 75107  1891 |     a =  N
  8250  3596 |     b =  Y
```

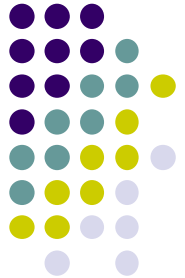
But our results were different from those obtained in Stata…

- 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.

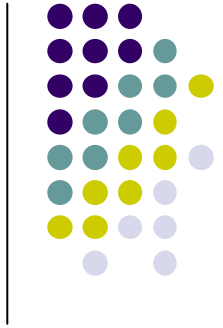
- *However, logistic regression coefficients are the same in both programs (if both datasets are carefully examined and identical variable encoding is used).*

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)

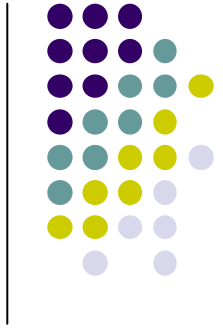


- Is the output really different?
  - MD5, merge / diff, visual inspection
  - Date -> Numeric conversion
  - String -> Categorical conversion
  - Categorical variable encoding
  - Integer vs Float vs Double vs Other numeric formats (f.ex. Long Double)

Summary (2/4)

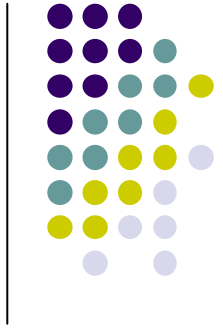
- Compare outputs in the two programs with different datasets:
  - 1 numeric,
  - several numeric,
  - 1 categorical,
  - several categorical,
  - mix of several categorical and numeric.

Summary (3/4)



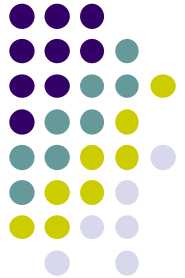
- Is the algorithm documented?
  - Check documentation
    - Formal: help files, official support
    - Informal: forums, blogs
  - If possible, check the source code

Summary (4/4)



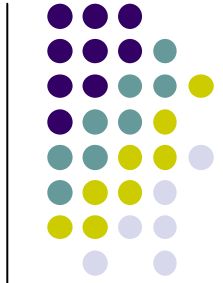
- Generally,
  - If the differences are very large, you are most likely doing something wrong:
    - different dataset
    - different variable transformation, encoding, *et cetera*
    - different algorithm
    - numerically unstable problem (f.ex. usage of perfect predictors)

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

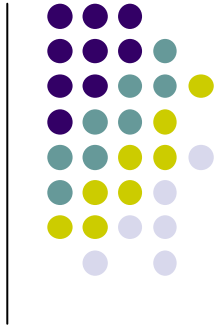
Multi-core performance



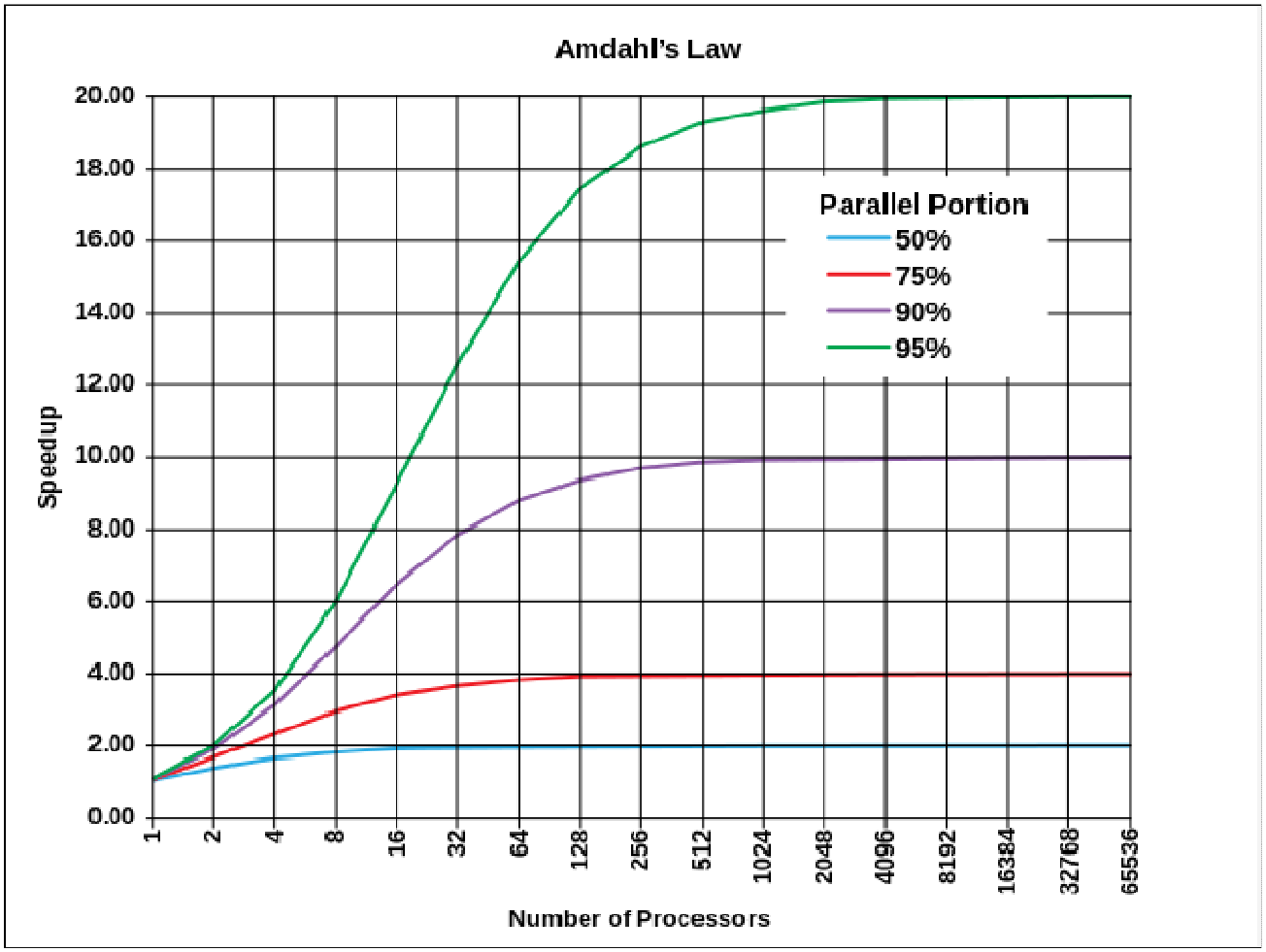
Big data (usually) requires

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:

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

Source: Wikipedia

st: MP running no faster than IC

From Ted Player <ted.player.660@gmail.com>
To statalist@hsphsun2.harvard.edu
Subject st: MP running no faster than IC
Date Mon, 8 Jul 2013 19:41:06 -0600

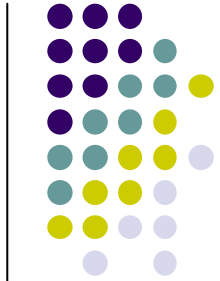
Short version: Stata MP 12-core isn't running my code any faster than it did when I used Stata IC, and I can't figure out why.

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

From Sergiy Radyakin <serjradyakin@gmail.com>
To "statalist@hsphsun2.harvard.edu" <statalist@hsphsun2.harvard.edu>
Subject Re: st: MP running no faster than IC
Date Mon, 8 Jul 2013 23:42:38 -0400



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, 0.00001second and with parallelization takes may be 0.000001 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.