Statistics: Changes since I was an undergrad

Part 1: Increased easy of computation

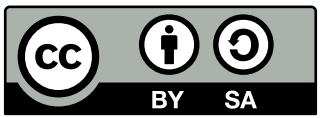
Pedro J. Aphalo

Department of Biosciences, University of Helsinki

17 November 2014

©2014 by Pedro J. Aphalo
Department of Biosciences, University of Helsinki, Finland.
http://blogs.helsinki.fi/aphalo/

Statistics: 'Changes since I was an undergrad. Part 1: Increased easy of computation' by Pedro J. Aphalo is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.



Lectures in this series

Changes in statistics: 1975–2014

This lecture is the first in a series of four related lectures.

1. Computing capacity has dramatically increased and programming languages have evolved.
2. New statistical methods have been developed.
3. Statistical software is dramatically more powerful and varied.
4. Accountability has become a requirement in scientific research.

Outline

What has changed?

- The changes:
  - Much more data is available.
  - Much more complex statistical analyses are possible.
  - Fewer and much more flexible assumptions are required for analyses.
- The not-so-new and new buzz-words:
  - Bioinformatics: biological and genetic data analysis.
  - Data science: mainly analysis of data about people's (reading, shopping, etc.) behaviour in the internet (Google, Amazon, Tweeter, e-mail reading tracking, user-location tracking in Android, etc.).
  - Big data: what *Data Science* and *Bioinformatics* study.

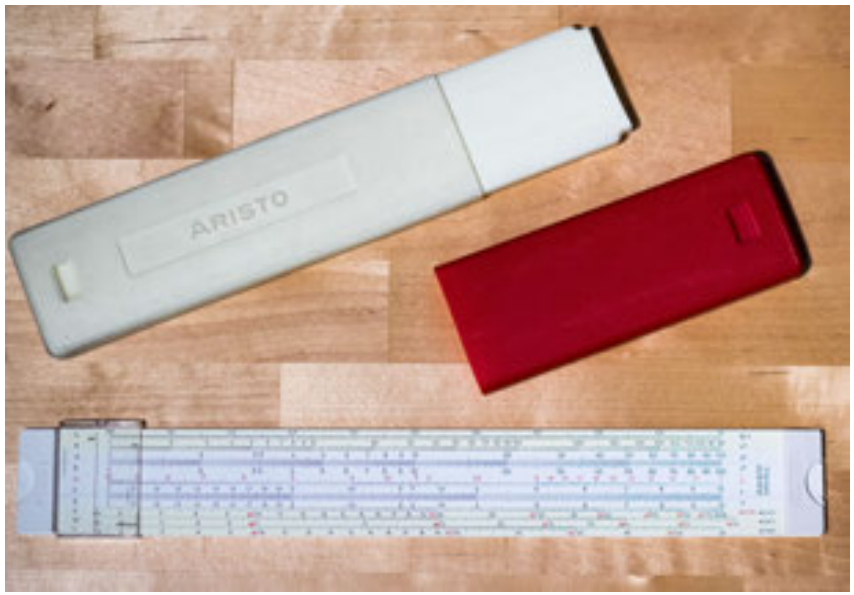
What has made the change possible?

- Communications infrastructure.
- Computing hardware.
- Computing software.
- Numerical methods.
- Statistical theory and (numerical) algorithms.

  Progress was interdependent.

Calculation devices

Mechanical devices e.g. Slide rule. Used 1625–197x. Replaced by scientific calculators.



Analogue electronic computers Used 194x–197x.

Digital electronic devices Calculators and computers.

Calculation devices

Mechanical devices e.g. Slide rule. Used 1625–197x. Replaced by scientific calculators.

Analogue electronic computers Used 194x–197x.

Digital electronic devices Calculators and computers.

Calculation devices

Mechanical devices e.g. Slide rule. Used 1625–197x. Replaced by scientific calculators.

Analogue electronic computers Used 194x–197x.

Digital electronic devices Calculators and computers.

Source: http://commons.wikimedia.org/wiki/File:TI-59_Half-side_view.jpg

What I have used

1. Scientific calculators: 1977, TI-59.
2. Apple II Plus: 1979, '6502' 8 bit processor @ 1 MHz, 48 kB RAM, no HD, 5.25 inch external floppy (160 kB?). BASIC, UCSD Pascal.
3. IBM PC: 1981, '8088' 16 bit processor (but 8 bit bus) @ 4.7 MHz, 64 kB RAM, no HD, 5.25 inch internal floppy (360 kB capacity DSDD), MS-DOS. Selling price at introduction USD 7000. A socket for a hardware floating-point processor was available. Followed by IBM PC/XT in 1983 with a HD (10 or 20 MB). MS-BASIC, Turbo Pascal.
4. Epson HX-20, the first laptop: 1983, two Hitachi 6301 CPUs at 614 kHz, 16 kB RAM, LCD display, printer, micro-cassette drive, 50 h running time on fully charged battery. EPSON BASIC. The first computer I owned.

The original PC

IBM PC

Source: http://commons.wikimedia.org/wiki/File:IMB_PC-IMG_7270.jpg

First laptop
Epson HX-20

Source: http://commons.wikimedia.org/wiki/File:Epson_HX-20_IMG_4245.jpg

What I have used

1. PC clones became common around 1986. I owned an XT clone with 8088+8087 running at 8 MHz, and a 20 MB hard-disk. Monochrome monitor. MS-DOS. Logitech Modula-2. Literate programming in MWEAVE. First crude user-interfaces using mainly text-based windows. Wrote my M.Sc. thesis using Lotus manuscript and printed it on a dot-matrix printer.
2. In 1989 I used a Unix mainframe for the first time. Heard for the first time about parallel computing... Used e-mail for first time, the internet was starting... In 1991, printed my Ph.D. thesis on a laser printer attached to a Unix mainframe. Used LaTeX to typeset it, working on a PC, but to generate the PostScript file of the whole thesis, I had to use the mainframe computer.

What is available today in our hands

1. My cheap (120 €) mobile phone (Lumia 630) has a 4-core processor running at 1.2 GHz, 512 MB RAM, 8 GB internal Flash memory and supports up to a 128 GB SD flash card.
2. Ubuntu (Linux) phones are expected to be released early next year, with possibility to connect to external monitors and keyboards.
3. R and the IDE RStudio can be run as a server, and accessed remotely through a web browser, even on devices with low computing and memory capacity.
4. Most workstations are just more reliable PCs, we can run multi-user operating systems like Linux and servers on almost any PCs.

What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically.

2. Raspberry Pi, a Linux computer
3. CubieTruck, a Linux or Android mini-PC

What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically. e.g. Arduino UNO (8 bit AVR @ 16 MHz, 2 kB RAM, Arduino IDE), Due (32 bit ARM @ 84 MHz, 96 kB RAM, Arduino IDE), Tre (32 bit ARM @ 1 GHz, 512 MB, Linux), Intel Galileo (32 bit Pentium @ 400 MHz, 256 MB RAM, Arduino IDE). ChipKit, Olimexino, etc. Open-hardware and open-software. Cost 20 to 70 EUR.
2. Raspberry Pi, a Linux computer
3. CubieTruck, a Linux or Android mini-PC

What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically.
2. Raspberry Pi, a Linux computer.

Source: http://commons.wikimedia.org/wiki/File:Front_of_Raspberry_Pi.jpg

3. CubieTruck, a Linux or Android mini-PC

What is available today to hobbists (and researchers)

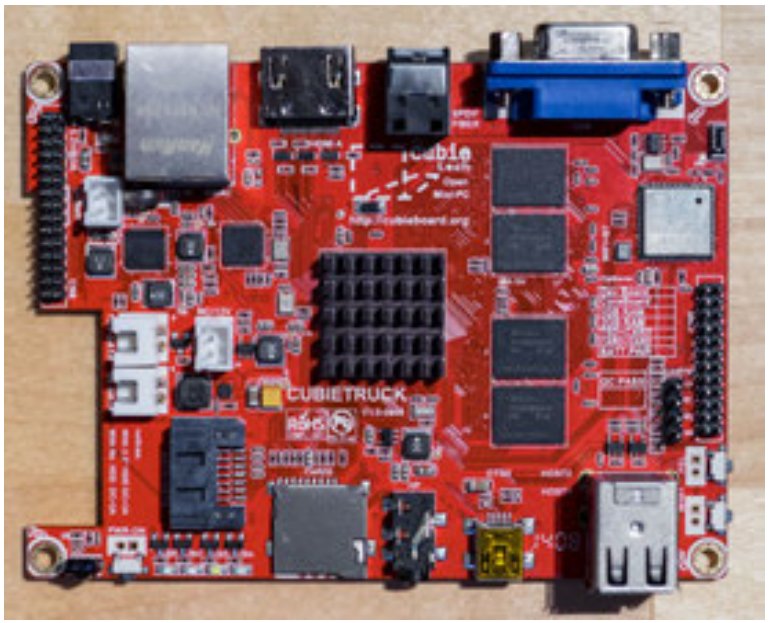
1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically.
2. Raspberry Pi, a Linux computerwith 256 MB RAM, available for 20–40 EUR, several flavours of Linux are supported. R can be made to run on it.
3. CubieTruck, a Linux or Android mini-PC

What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically.
2. Raspberry Pi, a Linux computer
3. CubieTruck, a Linux or Android mini-PC

What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically.
2. Raspberry Pi, a Linux computer
3. CubieTruck, a Linux or Android mini-PC.



What is available today to hobbists (and researchers)

1. Micro-controllers are now easy to use and program, and very cheap. Data acquisition will change radically. e.g. Arduino UNO (8 bit AVR @ 16 MHz, 2 kB RAM, Arduino IDE), Due (32 bit ARM @ 84 MHz, 96 kB RAM, Arduino IDE), Tre (32 bit ARM @ 1 GHz, 512 MB, Linux), Intel Galileo (32 bit Pentium @ 400 MHz, 256 MB RAM, Arduino IDE). ChipKit, Olimexino, etc. Open-hardware and open-software. Cost 20 to 70 EUR.
2. Raspberry Pi, a Linux computerwith 256 MB RAM, available for 20–40 EUR, several flavours of Linux are supported. R can be made to run on it.
3. CubieTruck, a Linux or Android mini-PCwith 2 GB RAM, 8 GB Flash, Cortex A7 2-core @ 1 Ghz, WiFi, Bluetooth, SATA 2.0, USB, HDMI 1080P, VGA, microSD, SDPIF digital audio, analogue audio out. Available for 65 EUR (board), 170 EUR (kit with metal case, SSD and Li-ion battery).

Parallel computing

1. (Not parallel) The 8087 arithmetic co-processor of the IBM-XT achieved 50 kFLOP or 0.000 000 000 050 TFLOP. An IBM-XT cost 3 000 to 7 000 USD in 1983.
2. The supercomputer Cray-2 was until 1990 the fastest computer on Earth at 0.0019 TFLOPS. Cost 32×10^6 USD.
3. CUDA and similar GPU (Graphic Processing Unit) based parallel processing. NVIDIA Tesla K-20 card, 3.5 TFLOP, 2496 cores, 225 Watts, 2700 USD. NVIDIA Quadro 4000 graphics card, 0.5 TFLOP, 256 cores, 450 USD.
4. Parallella III board (16 core processor): sells @ 100 USD and is widely available. Expected performance 0.090 TFLOP (per 64-core), consuming < 5 Watts.
5. However, today's fastest supercomputer is the Chinese Tianhe-2 (MilkyWay-2) at about 34 000 TFLOP! It has 3 120 000 cores and runs Kylin Linux. Cost 390×10^6 USD.

Parallel computing

1. (Not parallel) The 8087 arithmetic co-processor of the IBM-XT achieved 50 kFLOP or 0.000 000 000 050 TFLOP.
2. The supercomputer Cray-2 was until 1990 the fastest computer on Earth at 0.0019 TFLOPS. Cost 32×10^6 USD.
3. CUDA and similar GPU (Graphic Processing Unit) based parallel processing. NVIDIA Tesla K-20 card, 3.5 TFLOP, 2496 cores, 225 Watts, 2700 USD. NVIDIA Quadro 4000 graphics card, 0.5 TFLOP, 256 cores, 450 USD.
4. Parallella III board (16 core processor): sells @ 100 USD and is widely available. Expected performance 0.090 TFLOP (per 64-core), consuming < 5 Watts.
5. However, today's fastest supercomputer is the Chinese Tianhe-2 (MilkyWay-2) at about 34 000 TFLOP! It has 3 120 000 cores and runs Kylin Linux. Cost 390×10^6 USD.

Parallel computing

1. (Not parallel) The 8087 arithmetic co-processor of the IBM-XT achieved 50 kFLOP or 0.000 000 000 050 TFLOP.
2. The supercomputer Cray-2 was until 1990 the fastest computer on Earth at 0.0019 TFLOPS. Cost 32×10^6 USD.
3. CUDA and similar GPU (Graphic Processing Unit) based parallel processing. In the case of R, much easier under Linux than under Windows. NVIDIA Tesla K-20 card, 3.5 TFLOP, 2496 cores, 225 Watts, 2700 USD. NVIDIA Quadro 4000 graphics card, 0.5 TFLOP, 256 cores, 450 USD.
4. Parallella III board (16 core processor): sells @ 100 USD and is widely available. Expected performance 0.090 TFLOP (per 64-core), consuming < 5 Watts.
5. However, today's fastest supercomputer is the Chinese Tianhe-2 (MilkyWay-2) at about 34 000 TFLOP! It has 3 120 000 cores and runs Kylin Linux. Cost 390×10^6 USD.

Parallel computing

1. (Not parallel) The 8087 arithmetic co-processor of the IBM-XT achieved 50 kFLOP or 0.000 000 000 050 TFLOP.
2. The supercomputer Cray-2 was until 1990 the fastest computer on Earth at 0.0019 TFLOPS. Cost 32×10^6 USD.
3. CUDA and similar GPU (Graphic Processing Unit) based parallel processing. NVIDIA Tesla K-20 card, 3.5 TFLOP, 2496 cores, 225 Watts, 2700 USD. NVIDIA Quadro 4000 graphics card, 0.5 TFLOP, 256 cores, 450 USD.
4. Parallella III board (16 core processor): sells @ 100 USD and is widely available. Expected performance 0.090 TFLOP (per 64-core), consuming < 5 Watts. It has an ARM processor and up to a 64 core Epiphany IV Multi-core Accelerator, 1 GB RAM. Clustering up to 64×64 cores. Future plans for up to 1028 cores per chip yielding 1.4 TFLOP at 35 GFLOP per Watt. Can accelerate R through the r-opencl package...
5. However, today's fastest supercomputer is the Chinese Tianhe-2 (MilkyWay-2) at about 34 000 TFLOP! It has 3 120 000 cores and runs Kylin Linux. Cost 390×10^6 USD.

Cray-2 & Parallella

Sources: http://commons.wikimedia.org/wiki/File:Cray-2-IMG_0515.jpg and

http://www.parallella.org/board/

Open hardware

1. (IBM PC)
2. Arduino, ChipKit, etc.
3. 3D printers
4. ODAC ("Objective" DAC for music)
5. Parallella

What is available today through the net

1. Unix mail is from 1972, uucp (Unix to Unix copy) from 1978.
2. Internet as we know it today was born in 1982, when protocols were standardized.
3. The World Wide Web started in 1990. Has any of you used the browser Mosaic? or the character based browser Lynx?
4. I have had a web site on Plant Photobiology continuously on-line since 1995.
5. Currently: a huge amount of data are available on-line, such as maps, climatological time-series, genetic data, biological classification data, images, and gene-expression data. Open access scientific literature.
6. Future: open and reproducible science. Requirement for open access to raw data and data analysis methods used.

Open software

- Richard Stallman and Gnu
- Copyleft, GPL and FDL
- Linus Tordvalds and Linux
- Donald Knuth and TEX + Leslie Lamport and LATEX
- Sourceforge, Github, Bitbucket, etc.

Operating systems

- Unix: mainframe ← mini-computer → workstation → OS-X (Mac).
- Linux + Gnu: clusters/mainframes ← workstation ← PC → tablet/phone
  Red Hat; Ubuntu; Android; Sailfish (Jolla)
- MS-DOS: ⇝ Windows ⇝ Windows NT → → → Windows 8.
- O/S 2; CP/M; VAX; (dead or dying?)

Programming paradigms

- Machine: Assembler.
- Stack based: Forth.
- Functional: Fortran, Basic, Cobol.
- Structured: Pascal, C.
- Modular: Modula-2.
- List: Lisp.
- Object oriented: C++, Java.

Application specific languages

- Discrete simulation models: Simula
- System dynamics models: Dynamo
- System dynamics models (graphical): Stella, Vensim
- Mixed paradigm models (graphical): Simile
- Process/command ('shells'): sh, ksh, bash, command, etc..
- Text manipulations: awk, Perl.
- Typesetting mathematics: TEX, LATEX.
- Data analysis: S (R = Gnu S).
- Ruby, Phyton.
- Real time graphical (DSP, Robotics, instrumentation): LabView, FlowStone, etc.

Statistical software

- Systems not based on a well defined programming language, but with scripting and job control: Systat, SPSS, SAS, Minitab, Origin.
- Systems based on a programming language → boundary between user code and system code gets more flexible → they can be expanded by anybody: S, S-Plus, R.
- I have been using **R** for the past 15 years or so.
- Earlier I have used extensively **Systat**, **SPSS**, and **Origin**, and software developed by myself.
- Earlier I have used occasionally **S-Plus**, **SAS**, **Minitab**, **Gnuplot**, and others.

Text processing and typesetting

- WYSWYG: what you see is (approximately) what you get.
  - Word Perfect, MS-Word, Apache OpenOffice Writer, ...
  - Lotus Manuscript (A word processor for science from 1986).
- Typesetting systems: involve a "compilation" stage.
  - Unix's `troff`
  - TEX and LATEX, BibTEX
  - Metafont, `dvips`
  - pdfTEX, LuaTEX, XeTEX, Omega...
  - MetaPost, biber, arara...

Literate programming

> *Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to humans what we want the computer to do. — Donald E. Knuth, Literate Programming, 1984*

WEB, CWEB, FWEB, MWEB combine Pascal, C, FORTRAN, and Modula-2 with TEX to achieve this goal. Later, less extreme approaches are for example noweb, and the Java documentation system. WEB consisted in two main programs weave and `tangle`, plus `tie` for applying more than one change file.

An example from my PhD thesis

Literate programming

> *Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to humans what we want the computer to do. — Donald E. Knuth, Literate Programming, 1984*

WEB, CWEB, FWEB, MWEB combine Pascal, C, FORTRAN, and Modula-2 with TEX to achieve this goal. Later, less extreme approaches are for example noweb, and the Java documentation system. WEB consisted in two main programs weave and `tangle`, plus `tie` for applying more than one change file.

An example from my PhD thesis

Open science: An effervescent field

There is a parallel between the current Open Science and Reproducible Science movements and the earlier Literate Programming attempts. One aspect of doing research that is now becoming crucial is documenting data analyses in a human-readable- and understandable way.
At the same time, nowadays data analysis methods used tend to be more complex than they used to be. The best way of documenting such analyses is using some variation of Literate Programming with scripts and programs used in data analyses, plus making all 'validated' raw data openly accessible through public on-line repositories.

LaTeX + R = Sweave

Until recently the easiest way to achieve this was to use a system called sweave (remember WEB mentioned above) included as part of the R distribution. Output can be as PDF, PS and it depended on LaTeX.

Exercises as given before the class

Exercises as given after the class

Both .pdf files compiled from the same 'source code' file.

LaTeX + R = Sweave

Until recently the easiest way to achieve this was to use a system called sweave (remember WEB mentioned above) included as part of the R distribution. Output can be as PDF, PS and it depended on LaTeX.

Exercises as given before the class

Exercises as given after the class

Both .pdf files compiled from the same 'source code' file.

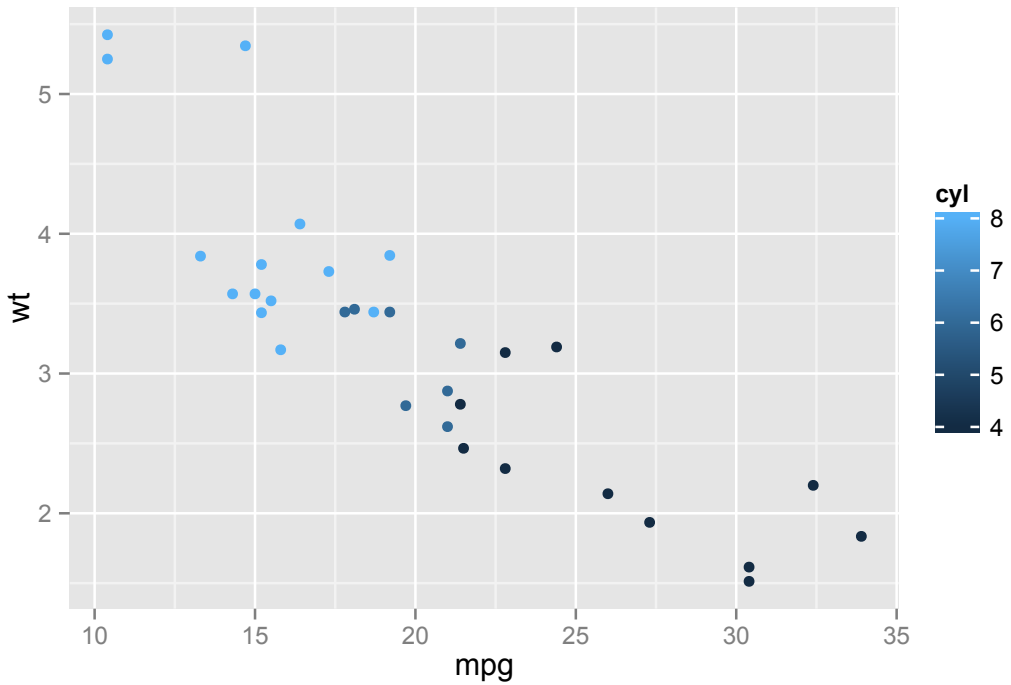
LaTeX + R = knitr

At the moment the easiest way to achieve this is to use a system called knitr (remember WEB mentioned above). Output can be as PDF, HTML, etc. and other 'typesetting engines' can be used in addition to LaTeX.
It is possible to use Markdown to document R code.
The best IDE for this is currently RStudio, which natively supports this type of approach for writing scripts and also for including R code and/or the results of calculations, including figures, in any report, or as you will see in the next slide, even presentations like this one.

knitr example

The code producing the figure below, with syntax highlighted in color.

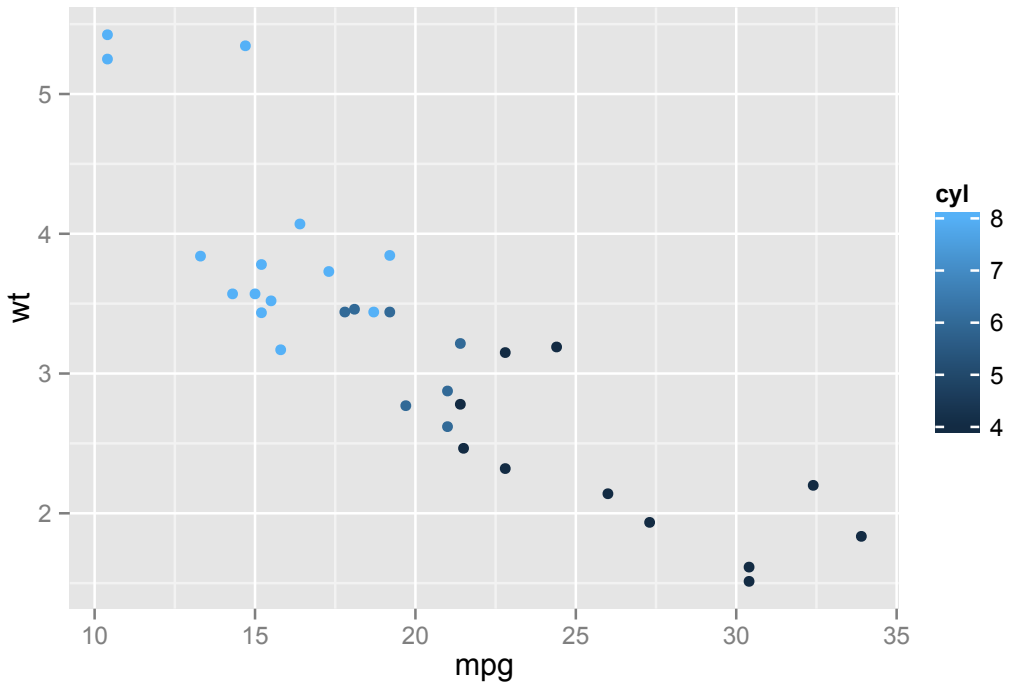
```
qplot(mpg, wt, data=mtcars, colour=cyl)
```



knitr example

The code producing the figure below, with syntax highlighted in color.

```
qplot(mpg, wt, data=mtcars, colour=cyl)
```

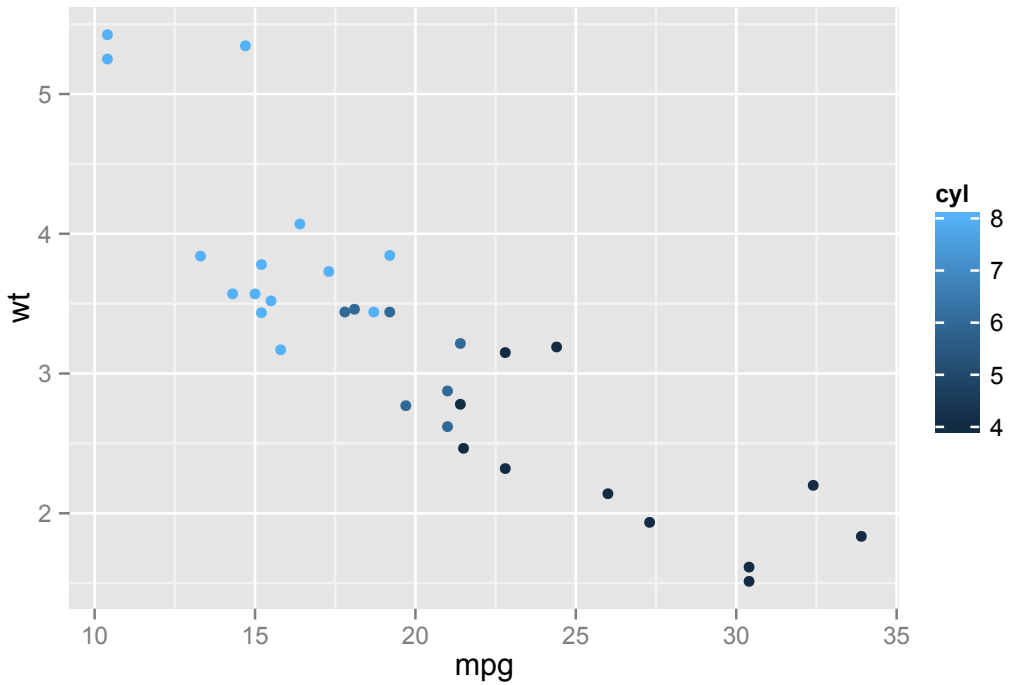


Did you guess by now?

knitr example

The code producing the figure below, with syntax highlighted in color.

```
qplot(mpg, wt, data=mtcars, colour=cyl)
```



Did you guess by now?
This whole presentation was prepared using knitr!

Resources

- The National Museum of Computing, U.K. http://www.tnmoc.org/
- Timeline at Computer Museum http://www.computinghistory.org.uk/cgi/computing-timeline.pl
- Supercomputers ranking http://s.top500.org/
- Donald Knuth's homepage http://www-cs-faculty.stanford.edu/~uno/
- The TEX User's Group (TUG) http://tug.org/
- The R Project http://cran.r-project.org/
- knitr home page http://yihui.name/knitr/

The End

- I prepared this slide presentation with LaTeX and R, on the RStudio IDE. I used Beamer and knitr, ggplot2 and other packages. This is all free open-source software, available for MS-Windows, OS-X, Linux, and Unix.
- This whole presentation (including example) is coded in a single text file (except for photographs and logos).
- To be continued. . . Part 2: Advances in software and theory
- **Thanks for listening!**