How to be a DRY lecturer

Paul Murrell
The University of Auckland

July 12 2010

Introduction

- It will be **assumed** that statistics students need to be taught computing skills.

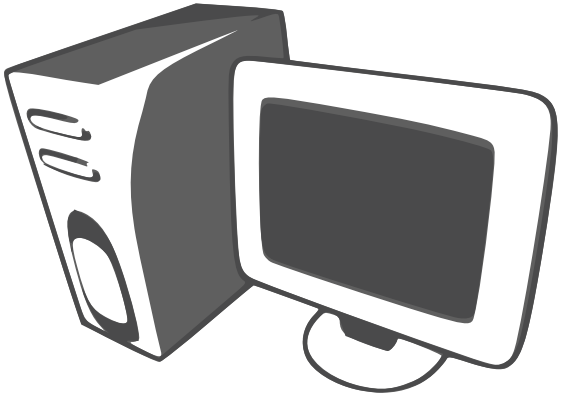
- The main question addressed here is:

  What computing skills should we teach?

- A related question is:

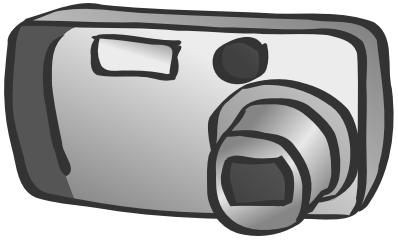
  How should we teach these computing skills?

A Clustering Quiz

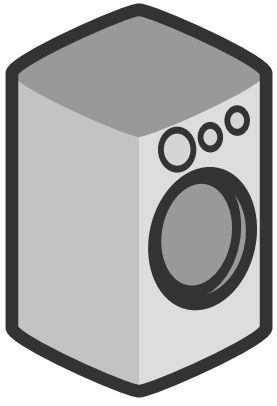


Personal Computer

Gaming Console

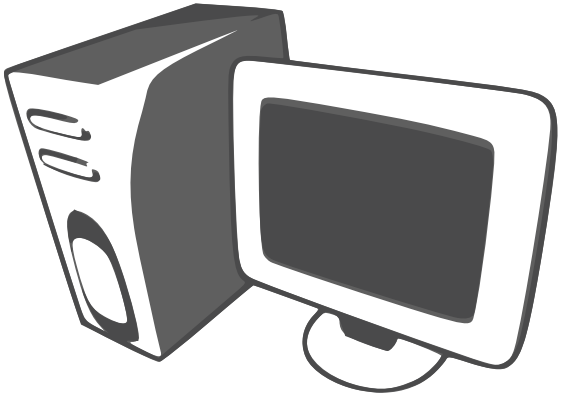


Digital Camera



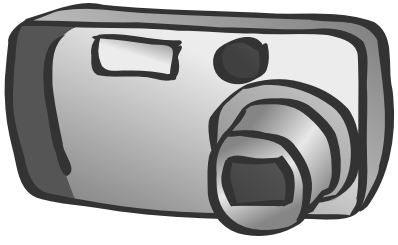
Washing Machine

The Answer ?

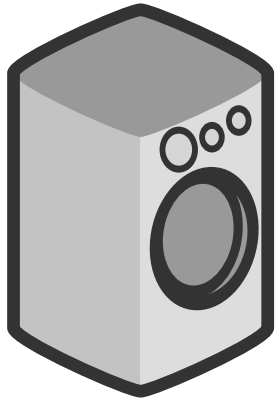


Personal Computer

Gaming Console

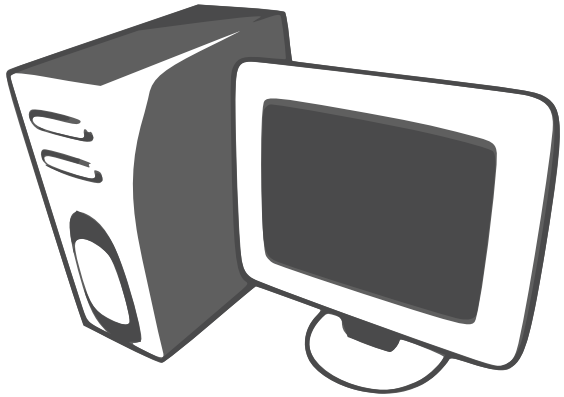


Digital Camera



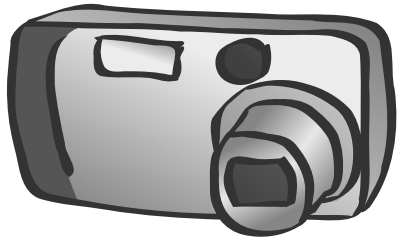
Washing Machine

The Answer ?

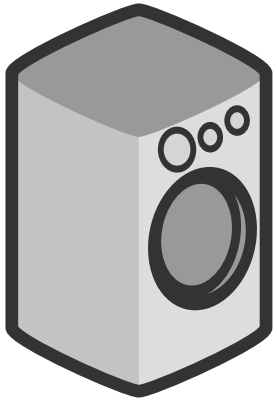


Personal Computer

Gaming Console

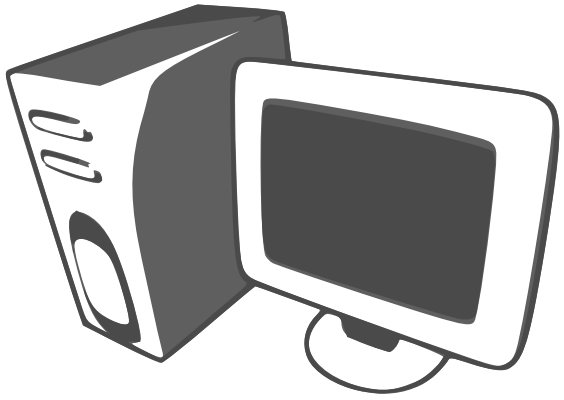


Digital Camera



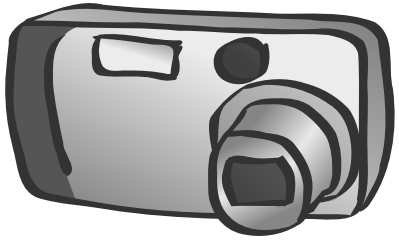
Washing Machine

The Answer !

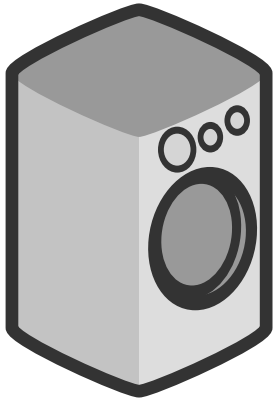


Personal Computer

Gaming Console



Digital Camera



Washing Machine

The Importance of Being Open

- The **Personal Computer** is an **Open** system

  With an open system, the user has the **Freedom** to put the system to any use.

- The others are **Closed** systems

  A closed system **Constrains** the user to only a subset of the possible uses.

The Importance of Being Open

Examples of the evils of being closed ...

- My automatic washing machine only allows **three** water levels!
  and only one sequence of events (wash, rinse, spin)
- My digital camera only offers **three** photo resolutions!
  and all of them use some form of (lossy) compression
- My playstation allows me to play amazing games ...
  but I cannot **develop my own**!

The Importance of Being Open

An example of the glory of being open ...

- *The Term Test and The Missing Clock:*

```
library(grid)
while (TRUE) {
    grid.newpage()
    grid.text(format(Sys.time(), format="%H:%M:%S"),
              gp=gpar(cex=10))
}
```

With a tip of the hat to Thomas Lumley

The Importance of Being Open

11:21:47

The Importance of Being Open

- If you only use Microsoft Windows and a mouse, the Personal Computer looks like a Closed system.
- Once you realise that the Personal Computer is Open, you can begin to **take control** of your computing environment (rather than the other way around).
- You do **not** have to be(come) a **Programmer** to start taking control.

Big Ideas in Statistical Computing

The most important things that I can teach a lot of my students are big ideas like this.

- The importance of being open
  (also discuss Open Standards for file formats)
- Learning what is **possible**
  (as well as **how** to do it)
- The importance of **writing code**
  (properly)
- The **Don't Repeat Yourself** (DRY) Principle
  (CSS and HTML, Database normalisation, Writing functions)

STATS 220 Data Technologies

The following computing topics are covered:

- HTML (for writing code)
- Computer memory and File formats
- XML (for data storage)
- Relational databases
- SQL (SELECT statement)
- R
    - Data structures
    - Data import/export
    - Data manipulation
    - Text processing
    - Regular expressions

The **motivation** and **context** for these topics are always **data handling** and **data processing**.

STATS 220 Data Technologies

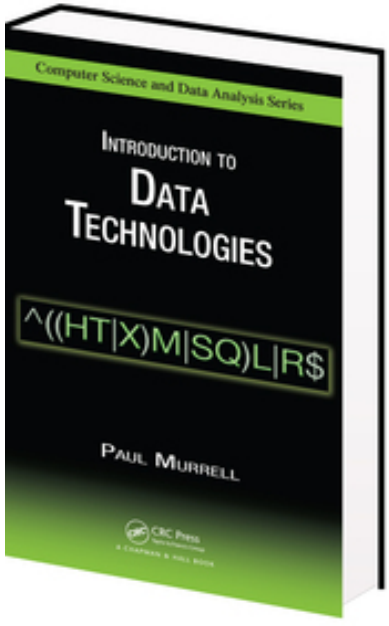
The **practical** component of the course is vital ...

- Two lectures per week
- One three-hour lab per week (assessed)
- Three assignments

... but because submission of coursework is electronic, there must also be assessment under controlled conditions ...

- Term test 20%
- Exam 60%

Data Technologies Book



http://www.stat.auckland.ac.nz/~paul/ItDT/

STATS 380 Statistical Computing

The original plan was to teach students some ideas of **software development** (e.g., object-oriented programming), but that proved **hopelessly optimistic**.

The current approach is an **extension** of (the R section of) **STATS 220**.

- Motivated by data processing tasks
- Larger tasks lead to larger bodies of code ...
- ... which lead to more sophisticated code (**algorithms**) ...
- ... and more sophisticated use of data processing tools (**writing** your own **functions**) ...
- ... and more sophisticated computing tools (**debugging**)

STATS 380 Statistical Computing

The course material has four main components:

- A single large **motivating example**
- **Side-tracks** to explore individual functions in more detail
- **Recaps** to emphasize concepts and re-orientate ourselves within the large project
- **Summaries** that list all functions and concepts dealt with in a section of the course.

380 Large motivating example

1. Harvest the data: read a set of web pages into R
2. Process the data: extract data from the web pages
3. Plot the data: produce displays of the data
4. Output the results: produce a report containing the plots

380 Large motivating example: Generating file names

```
> month <- 1:12
> monthString <- sprintf("%02d", month)
> year <- 2005:2009
> outer(year, monthString, paste, sep="-")
```

```
     [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
[1,] "2005-01" "2005-02" "2005-03" "2005-04" "2005-05" "2005-06" "2005-07"
[2,] "2006-01" "2006-02" "2006-03" "2006-04" "2006-05" "2006-06" "2006-07"
[3,] "2007-01" "2007-02" "2007-03" "2007-04" "2007-05" "2007-06" "2007-07"
[4,] "2008-01" "2008-02" "2008-03" "2008-04" "2008-05" "2008-06" "2008-07"
[5,] "2009-01" "2009-02" "2009-03" "2009-04" "2009-05" "2009-06" "2009-07"
     [,8]      [,9]      [,10]     [,11]     [,12]
[1,] "2005-08" "2005-09" "2005-10" "2005-11" "2005-12"
[2,] "2006-08" "2006-09" "2006-10" "2006-11" "2006-12"
[3,] "2007-08" "2007-09" "2007-10" "2007-11" "2007-12"
[4,] "2008-08" "2008-09" "2008-10" "2008-11" "2008-12"
[5,] "2009-08" "2009-09" "2009-10" "2009-11" "2009-12"
```

Example 380 side-track

The `sprintf()` function:

- `print` a `f`ormatted `s`tring.
- Why does this function exist?
- What do the funny format codes mean?

Example 380 Recap

1. Harvest the data: read a set of web pages into R
   - Generate a character vector of file names
   - Use `lapply()` to ...
     - Read each HTML file using `readLines()`

   ... and return a list of character vectors

Example 380 Summary

`read.csv()`	Read a CSV text file and generate a data frame.
`readLines()`	Read a text file and generate a character vector.
`sprintf(f, x)`	`print` a value, `x`, in a particular `format` and return the result as a `string`.
`outer(x, y, FUN)`	Call the function `FUN` on all possible combinations of `x` and `y`.
`paste()`	Join several character values together to make a single character value.
`as.character()`	Convert an R object to a character vector (if possible).
`t()`	Transpose a matrix.
`numeric(n)`	Generate a numeric vector of length `n` (filled with zeroes).
`vector(type, n)`	Generate a vector of a given `type` of length `n`.

Conclusions

- Students who have only been computer **users** need help to become computer **literate**.
- There is a **lot** that computer users **do not know** about how computers work; and what they **do know** is at the **wrong level of abstraction**.

- Students do **not** have to become **programmers** to learn useful skills.
- A basic knowledge of **data structures**, basic **discipline** in **writing code**, and knowledge of basic **data processing** tools can go a looooooong way.

- Data handling and data processing provide an excellent **motivation** and **context** for teaching computing skills

Acknowledgements

- The image of the Personal Computer is a modified version of `http://openclipart.org/people/Anonymous/Anonymous_gis-computer.svg`, a Public Domain image from the Open Clip Art Library.
- The image of the Gaming Console is a modified version of `http://commons.wikimedia.org/wiki/File:Gamepad.svg`, an LGPL image from Wikimedia Commons.
- The image of the Digital Camera is a modified version of `http://openclipart.org/people/nicubunu/nicubunu_Digital_camera_(compact).svg`, a Public Domain image from the Open Clip Art Library.
- The image of the Washing Machine is a modified version of `http://openclipart.org/people/jcartier/jcartier_Washing_machine.svg`, a Public Domain image from the Open Clip Art Library.
- The image of the digital camera hack is from `http://chdk.wikia.com/wiki/File:Histo_R_G_B.jpg`, a Creative Commons (CC-BY-SA) licensed image from the Wiki of the Canon Hack Development Kit project.