Scientific Computing and Data Management

Marcello Vichi

Department of Oceanography
marcello.vichi@uct.ac.za

Applied Ocean Sciences

Outline

Oceanography is an observational and computational science

Fact

Ocean scientists collect data or access data collected by others

- Many scientists choose to work with a specific software for data analysis and visualization but soon realize that much of the work is not only writing computationally intensive number-crunching loops
- Very often programming is about **shuffling data in and out of different tools, converting one data format to another, extracting numerical data from a text, and administering experiments involving a large number of data files and directories**. (HP Langtangen, Python Scripting for Computational Science, 2008)
- This workflow must be properly organized, no matter which operating system or software you have chosen

The data<->science workflow

The aim of this course is to increase the productivity of your data-to-science workflow, by presenting you with a few examples of methodologies and tools

1. Production/Collection *(Investigation and Meta-search)*
2. Retrieval *(Transfer)*
3. Data Management *(Short-term archival and meta-data)*
4. Processing *(Programming and Revision control)*
5. Analysis *(Programming and Revision control)*
6. Publication or Higher Level Production *(Programming, Revision control and meta-data)*
7. Data Management *(Long-term archival and meta-data)*

Data Management (DM)

Adapted from the lecture notes by R. Roman and D. Byrne

Data management comprises the following items:

- data collection
- organisation
- storage
- preservation
- publication

Why do we need DM?

- Government and research data, collected at public expense, must be properly managed in order to realize their full potential and justify their considerable production and maintenance costs
- Your institution's and funding agency's expectations and policies
- To ensure research integrity and validation of results.
- To increase research efficiency
- To facilitate data security and minimise the risk of data loss
- To ensure wider dissemination and increased impact
- To enable research continuity through secondary data use (permit new and innovative research to be built on existing information)
- Uniqueness of observations => you can't measure a 2001 temperature next year!
- Journals now request data to be made available on line for the study to be validated (Retraction watch) https://retractionwatch.com/

Preventing data loss

- - human error
- - natural disaster
- - facilities infrastructure failure
- - storage failure
- - server hardware/software failure
- - application software failure
- - format obsolescence [floppy and stiffy drives]
- - legal encumbrance
- - malicious attack
- - loss of staffing competencies
- - loss of institutional commitment
- - loss of financial stability

Principles of good DM

- Define a research data management policy (University of Cape Town data policy: http://www.digitalservices.lib.uct.ac.za/dls/rdm-policy)
- Data ownership (clear identification who has managerial and financial control; legal rights along with copyright and intellectual property rights)
- Data documentation and metadata compilation [metadata applicable to your field]
- Data quality and standardisation
- Data life-cycle control
- Data custodian (not a person)
- Data access and dissemination

Definition: Research Data

Research data is defined as **recorded factual material** commonly retained by and accepted in the scientific community as necessary to validate research findings
http://www.epsrc.ac.uk/about/standards/researchdata/scope

Forms of research data

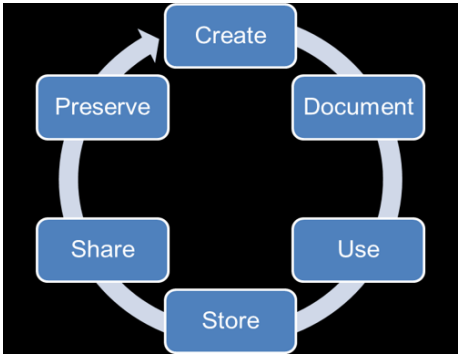
1. Observational (data captured in real-time. For example, sensor data, survey data, sample data)
2. Experimental (data from lab equipment, often reproducible)
3. Simulations (data generated from models. For example climate and oceanic models)
4. Derived/compiled (data is reproducible, text and data mining)
5. Reference (collection of peer-reviewed datasets that is published and curated)

Data Management Plans

Most of the collected data in Ocean Sciences will ultimately become **digital data**.
A DMP is always needed and considerations must be done prior to collecting any data.

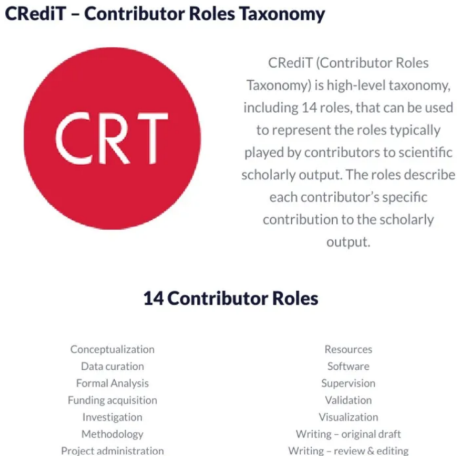
- Types of data
- Data sharing, access & security,
- Metadata standards,
- re-use, archiving & long-term preservation.

Research data life cycle (Digital Curation Centre, www.dcc.ac.uk)



Data curation and visualization are recognized scholarly roles

- Visualization - Preparation, creation and/or presentation of the published work, specifically visualization/data presentation.
- Data curation - Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later re-use.

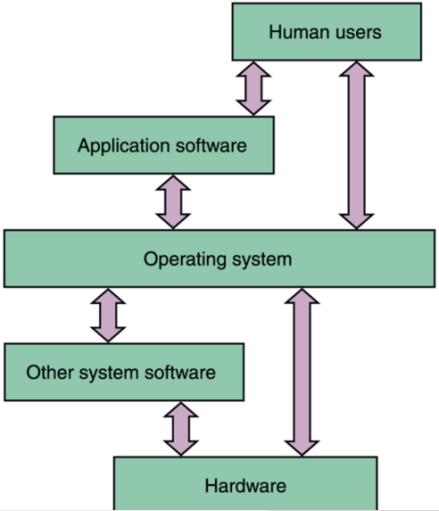


The computer interface

- The "computer" is a logical processing unit that must be instructed to perform operations (programmed)
- It deals with information in a **digital** format, using electrical signals that are temporarily stored in a physical space called memory
- The "inventors" of the computer *idea* (Turing and von Neumann) created the mathematical methods to perform operations (program) and to solve an *infinite* set of problems. This developed the concept of the **Central Processing Unit** of programmable computers
- This was the first giant leap: what was needed was creating the technical possibility to do things according to this theoretical model. Translating the operations in a machine language and returning the results in a human-understandable language. The first working programmable computer was done by Konrad Zuse in 1941.
- How do we interface with a computer?

The operating system (OS)

- The operating system is the agent that allows the interaction between hardware (the components), software (the programs) and the user `http://faculty.salina.k-state.edu/tim/ossg/Introduction/OSrole.html`



What the OS does

The operating system is the **software interface** that appears after the boot has completed, and runs underneath all other programs - without it nothing would happen at all. In simple terms, an operating system is a manager. It manages all the available resources on a computer, from the CPU, to memory, to network and hard disk accesses.

Tasks the operating system must perform:

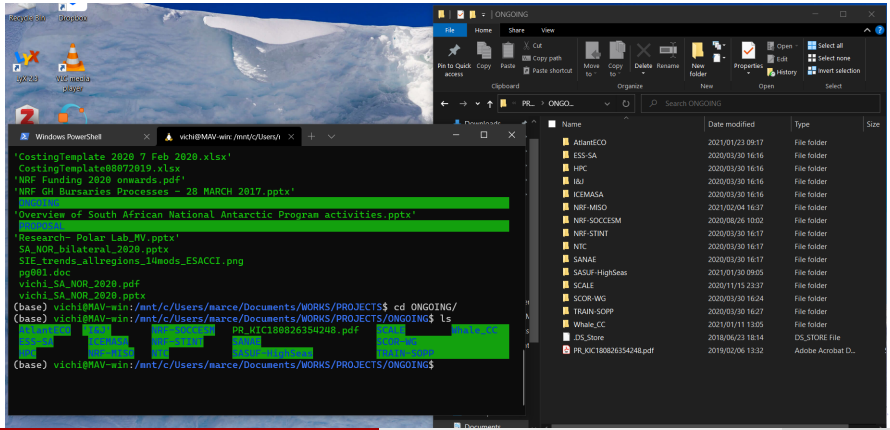
- Control Hardware - The operating system controls all the parts of the computer and attempts to get everything working together.
- Run Applications - Another job the OS does is run application software. This would include word processors, web browsers, games, etc...
- Manage Data and Files - The OS makes it easy for you to organize your computer. Through the OS you are able to do a number of things to data, including copy, move, delete, and rename it. This makes it much easier to find and organize what you have

Types of OS

- Windows (originally DOS)
- Mac OSX (based on UNIX)
- UNIX and Linux
- Android and iOS
- Virtual OS: host and guest (VirtualBox, VMware, etc)

Giving instructions

Operating systems have evolved considerably, from terminal prompts to desktop managers. All modern OS allows you to give instructions both through gestures (menu-click-drag-drop) and command line terminals

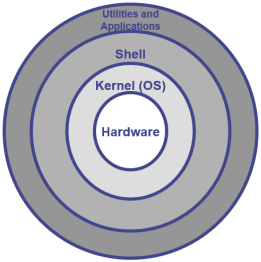


UNIX: a prototype for all OS

The UNIX operating system was born in the late 1960s. It originally began as a one-man project led by Ken Thompson of Bell Labs, and has since grown to become the most widely used OS.

- In the time since UNIX was first developed, it has gone through many different generations and even mutations.
- Some differ substantially from the original version, like the Berkeley Software Distribution (BSD, used on Mac OS) or Linux.
- Other OSs still contain major portions that are based on the original source code.
- An interesting and rather up-to-date timeline of these variations of UNIX can be found at http://www.levenez.com/unix/history.html

Structure of the UNIX OS



There are many standard applications:

- file system commands
- text editors
- compilers
- text processing

- The Kernel - handles memory management, input and output requests, and program scheduling. Technically speaking, the kernel is the OS. It provides the basic software connection to the hardware. The kernel is very complex and deals with the inner workings of these things, and is beyond the scope of this course.
- The Shell and Graphical User Interfaces (GUIs) - the *shell* is a program that provides a "command line" interface where the user can type commands. The GUI is similar, but commands are substituted by graphical actions (buttons, menus, etc). These commands are translated by the shell into something the kernel can comprehend, and then executed by the kernel.
- The Built-in System Utilities - are programs that allow a user to perform tasks which involve complex actions. Utilities provide user interface functions that are basic to an operating system, but are too complex to be built into the shell. Examples of utilities are programs that let us see the contents of a directory, move & copy files, remove files, etc...
- Application Software & Utilities – these are not part of the operating system. They are additional programs that are bundled with the OS distribution, or available separately. These can range from additional or different versions of basic utilities, to full scale commercial applications

Testing the command line

- All operating systems have a command line interface. The commands are typed by the user and executed using the enter key. To obtain the list of files in a folder, test the following:
  - Windows app: Command prompt. Launch the application and type the command `dir`.
  - Mac and Linux app: Terminal. Launch the application and type the command `ls`.
- You will obtain the list of files in that folder. Some of them are files of various types and some other are folders. Note that a folder is also a file on the operating system: a file that contains a list of other files

Basic data types

- All machine-readable data are digital, which means that they are made of coded **discrete** information
- Data must have a format to be understood by the operating system
  - **Binary data**: contain the bits and bytes of the information without a specific coding. They can be interpreted by a software that knows already what to look for.
  - **ASCII data**: American Standard Code for Information Interchange. An ASCII code is the numerical representation of a character, which is still written in binary code but software know how to handle the coding. 01101000 and 01101001 are equal to the decimal numbers 104 and 105, which in ASCII corresponds to the letters "hi"
- A text produced with MS Word is a binary file, even if it contains ASCII characters. This is because it includes formatting information on how the document looks like. A text produced with a *text editor* (e.g. notepad) is instead an ASCII file (often called a **flat file**).
- A MS Excel "table" is not a flat file. Only if you save it in ASCII format (e.g. using the Comma Separated Values format, csv) it will be accessible by other ASCII applications.

Data transfer and protocols

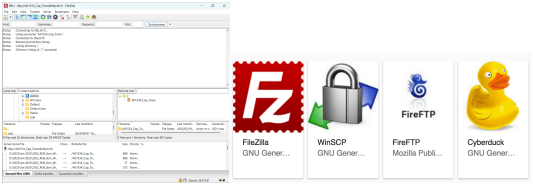
- Data must be on a support that is readable through a peripheral. Nowadays we use files on *drives* or through the network. Originally (until the late 70's), the data were passed through paper, the **punch cards**, then through magnetic *floppy* disks
- On the Internet, data needs to be transferred in a coded way. These methods are called protocols
  - **HTTP**(S): hyper text transfer protocol (the main protocol of browsers, written in HTML)
  - **FTP**: file transfer protocol (needs a terminal or apps)

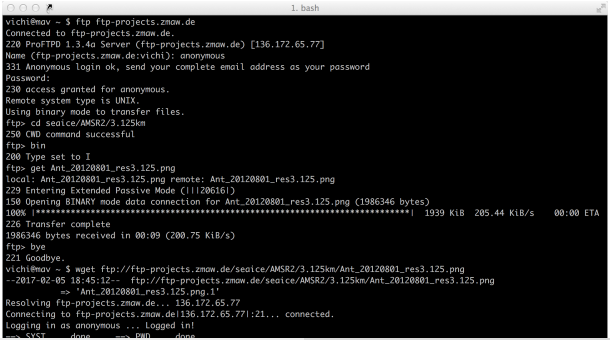
Punch Card in
Punch Card Machine

http://www.computerhope.com

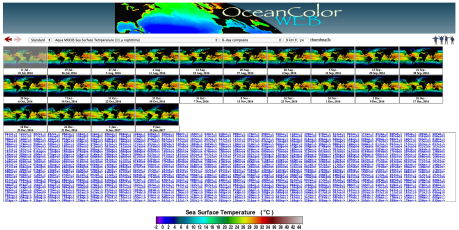
Transfer and Storage: Apps or command line?

- Data (file) transfer must be handled by specific applications that know how to work with the protocols
- This can be done through browsers, clients and command line applications
- Several public data archive allows to access their data via several methods and for most of them it is possible to click on the data we want
- But what if we want more than one data file? We need to use an application that allows this **batch** handling or to write an ad hoc script executed through the command line

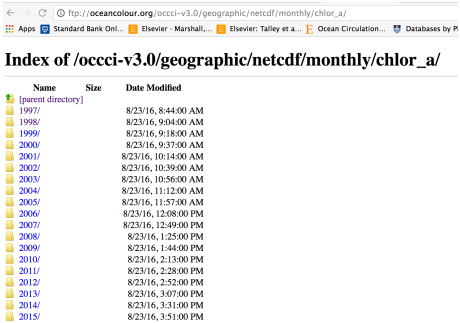




Issues with multiple data transfer



https://oceancolor.gsfc.nasa.gov



https://oceancolour.org

Giving instructions to the OS: the shell

Commercial software is not sufficient to perform all the multiple processing operations needed in ocean sciences. We often have to

- download multiple data from different repositories (often at specific time intervals, e.g. every day)
- perform the same operation on different files (temporal and spatial averages, extractions, etc.)
- rename, cut, concatenate files and convert between different formats

All these operations require more than a GUI. This is where the shell (the language of the terminal) becomes relevant because allows you to run batch operations (*multiple commands that can be sequentially executed by the computer, often in the background*). Your computer, if not a unix OS, is not naturally based on a shell environment. You then need to prep it and install the proper tools as explained on the instructions online.

Scripting and programming: the key to success

http://www.nature.com/naturejobs/science/articles/10.1038/nj7638-563a?WT.mc_id=FBK_NatureNews

There are basically 3 choices to combine the computational capabilities of modern programming languages with processing and analysis of multidisciplinary ocean data

- *Matlab* (commercial software, open-source version *octave*)
- *R* (open source)
- *Python* (open source)

They all have pros and cons and they are all interpreted scripting languages (very high level programming languages; more in the next lectures). They allow to handle the typical objects of earth scientists and georeferenced data

- Python is the language of choice in this course, but we will also learn a few shell scripting commands to manipulate files
- Self-learning environment to guide you through the first steps with python
- on-line assignments will contribute to the final mark of the module