14

Data: Small and Big

14.1 Data files and formats

For checking purposes it is advantageous to create human readable output, that is, plain text. Plain text is also a highly portable file format; except, the end of line encoding can vary between operating system, which can be a nuisance, until one discovers one of the tools that convert them automatically. The line-ending for Unix, Linux, and Mac is `\n`; that for Windows & DOS is `\r\n`.

File Size. It is possible to at least estimate file size. When data are stored in text format, as they often are, each character takes up one byte. Delimiters and blank spaces also count as characters. The number `1.23456E-04`, without leading or trailing blanks, takes up 11 bytes on the storage medium. If an invisible carriage return is at the end of the number, then it consumes an additional byte. A single-precision number, like `1.23456E-04`, typically takes up four bytes of memory. As a rule of thumb, a set of stored data takes up more disk space than the same set in memory.

Compression. A large file containing mostly numbers uses only a small part of the full character set and can hence be substantially compressed into a file of smaller size. Number-only files typically compress, with conventional utilities, to around 40% of their original size. If repetitive patterns are present in the file, the compression will be even stronger.

Binary files are typically smaller than plain text files. They are portable, except for a flip in endian-ness on different machines; some computers write groups of bytes left-to-right and others right-to-left.

XML (EXtensible Markup Language) was designed to describe data. (HTML was designed to display data, and CSS formats data.) This format has widespread support for creation, reading, and decoding. An alternative to XML is JSON (JavaScript Object Notation). Although originally derived from the JavaScript scripting language, JSON is a language-independent data format. Code for parsing and generating JSON data is readily available in many programming languages. JSON is promoted as a low-overhead alternative to XML.

14.2 Text processing utilities

Data, whether real or from computer simulations, come in different formats, are created on different operating systems, have different symbols for comment lines, and different placeholders for invalid entries. Handy tools for file manipulation are operating system utilities and scripting languages. Sophisticated automated manipulations can be done quickly with text processing languages such as awk, perl, sed, and others.

To illustrate the capabilities of text processing tools, here are a few useful one-liners:

- Print all lines where the entry in the first column is larger than zero:

  ```
  awk ‘{if ($1>0.) print}’ yourfile
  ```

- Print the first column and the sum of the third and fourth column of a comma separated table, and replace commas with spaces:

  ```
  awk -F, ‘{print $1,$3+$4}’ yourfile.csv
  ```

- Display all lines except whose that contain ’NaN’:

  ```
  grep -v NaN yourfile
  ```

- Replace the word “NaN” with “-9999” everywhere:

  ```
  sed ‘s/NaN/-9999/’ yourfile
  ```

- replace line breaks with spaces:

  ```
  tr "\n" " " < yourfile
  ```

- Flip order of rows:

```
perl -e ‘print reverse <>’ yourfile
```

`sed` is a *stream editor* used for non-interactive editing. `awk` is a programming language designed for text processing and often used as a data extraction tool. `Perl` is a bigger language than `awk` and `sed` and can, in principle, replace both. The command-line Unix utility `grep` is an excellent data filter.

A `regular expression` is a special text string that describes a search pattern. For example, a wildcard `*` matches any string, `?` matches any one character, `[09]` matches 0 or 9, [0-9] any number from 0 to 9, and [0-9a-fA-F] matches a single digit of a hexadecimal number. Unix utilities and other tools often recognize regular expressions (e.g. awk, grep, and Perl).

`wget` and `curl` are command line tools that can download content from web addresses without user interaction.

14.3 Big data

“Big data” is a broad and vague term that refers to data so large or complex that traditional data processing methods or technology become inadequate. They may require distributed storage, new numerical methods, and enhanced curation.

The following problems may arise when working with large data sets:

1. Data don’t fit in main memory: This is common and the method has to consider *data locality* that minimizes the number of time data is read from the hard drive instead of minimizing the number of floating point operations. An example of such a numerical algorithm is the tiling for matrix multiplication in sec. 10.4.

2. Data don’t fit on the local drive, but can be streamed through: In addition to data local processing, it becomes important that data are formatted rigorously, because otherwise dealing with no-longer rare exceptions will amount to a significant effort.

3. Data are too big to be downloaded: Storage is easier than transfer, hence analysis has to be where the data are not where the user is. This is a game changer, as now the responsibility for data analysis infrastructure lies with the data host. "Cloud computing" primarily deals with this issue.

   Analyzing data on somebody else's computer also implies that the researcher is limited to the software provided by the host. SQL (Structured Query Language) is a query language for requesting information from a database. It is not only a special-purpose programming language, but an old rudimentary version of SQL is an official standard, and its syntax and concepts have become widely used. Many derivatives and extended versions of SQL are in use today. Apache Hadoop is an open-source software framework for distributed storage and distributed processing of very large data sets on computer clusters.

4. Data don't fit anywhere, or the data is produced in bursts and cannot be stored fast enough (e.g., particle collider experiments, arrays of radio telescopes). In other words, the data move too fast. This situation has to be dealt with with on-the-fly data analysis, also known as "stream processing". Floating point is much faster than writing to disk, so valuable analysis can be done on the fly. Hardware accelerators, such as GPUs (sec. 9.3), are natural candidates for such a situation.

5. Big data is not necessarily large data. Data that are complex but unstructured or insufficiently structured may require too much time to be properly analyzed.

Example of a Big data tool: JPEG-2000 (or JP2) is an image format that incorporates "smart" decoding for work with very large images. It is possible to smoothly pan and zoom with decompression of only a portion of the compressed data. And downsampled versions of the image can be viewed without adding to the file size. The JPEG2000 standard also provides an internet protocol (JPIP – JPEG 2000 Interactive Protocol) for efficient transmission of JP2 files over the network. JPIP makes it possible to pan and zoom gigapixel images in real time over the network.

For example, JPIP is used to view lossless JP2 images from the Mars Reconnaissance Orbiter on the HiRISE website.

14.4 Network and storage technologies

Ethernet can transfer up to 10 Gigabits per second (Gbps) and can be up to several hundreds meter long. Optical fibers have higher transfer rates (commercially currently up to about 100 Gbps on a single fiber) and can be used over longer distances. They are more expensive, not because of the cost of the cable, but because opto-electrical converters are needed at the beginning and end of each line. Wireless (radio) is nowadays implemented with low-cost semiconductor technology, and commonly provides up to 100 Mbps.

Assuming an actual average download speed of 0.1 Gbps (already 3 times of what I get at my research institution on a weekend in 2015), downloading 1 TB takes 22 hours. Data of the size of 1 PB are hence, for all practical purposes, physically stuck in place, which gives rise to the concept that analysis has to be where the data are not where the user is.

When data sets are very large, they have to be distributed over many physical storage units. And since every piece of hardware has a failure probability, failure is commonplace in a large cloud or data center. Modern file systems can automatically handle this situation, without interruption to normal operations, by maintaining replica of all data. A rule of thumb is to keep two copies for every piece of data, one nearby (e.g. on the same rack) and another remote.

RAID (Redundant Array of Independent Disks) is a storage system that distributes data across multiple physical drives to achieve data redundancy, fast data access, or a bit of both.

HDFS (Hadoop Distributed File System) is designed to hold very large amounts of data. Data nodes can talk to each other to move copies around, but coordination is through a master node.

14.5 Data archiving

"you have no idea the number of excuses people come up with

to hang onto their data and not give it to you"

Tim Berners-Lee

Archiving of data enhances their scientific value. In particular, it allows for combining data from different sources. The lower the barrier to finding, obtaining, and understanding data, the better. That said, data need to be *curated*, and that requires extra effort by the data creator. Today data by themselves, without any scientific conclusion derived from them, can be published and cited, giving the data creators the credit they deserve.

The following features serve the function of an archive: Long-term availability, open accessibility (without pre-approval), standardized formats and file types, independent peer review, documentation, and citability.

Longevity of file formats: As technology changes, both hardware and software may become obsolete. And if the past is any guide, they really do. File formats more likely to be accessible in the future are nonproprietary, open, documented standards commonly used by the research community. Where file size permits, uncompressed plain text (ASCII or Unicode) is an excellent choice.

□ Good file format choices for data archiving include: Plain text (Ascii/UTF-8) or Open Document Format (not Word), CSV (not Excel), JPEG2000 (not JPG), TIFF or PNG (not GIF), MPEG-4 (not Quicktime), and XML.