Naming Files

Data Science Workflows

Dr Zak Varty

Naming things is easy, Naming things well is hard

> "There are only two hard things in Computer Science: cache invalidation and naming things."
>
> **Phil Karlton, Netscape Developer**

Part 1: Naming Files

What do we want from file names?

1. Machine Readable

2. Human Readable

3. Order Friendly

Current names:

```
abstract.docx
Effective Data Science's module guide 2022.docx
fig 12.png
Rplot7.png
1711.05189.pdf
HR Protocols 2015 FINAL (Nov 2015).pdf
```

Better names:

```
2015-10-22_human-resources-protocols.pdf
2022_effective-data-science-module-guide.docx
2022_RSS-conference-abstract.docx
fig12_earthquake-timeseries.png
fig07_earthquake-location-map.png
ogata_1984_spacetime-clustering.pdf
```

Machine Readable

What do we mean by machine readable file names?

- Easy to compute on by **deliberate use of delimiters**:
  - `underscores_separate_metadata`, `hyphens-separate-words`.

- Play nicely with **regular expressions** and **globbing**:
  - avoid spaces, punctuation, accents, cases;
  - `rm Rplot*.png`

Machine Readable

Machine readable names are useful when:

- **managing files**: ordering, finding, moving, deleting:

- **extracting information** directly from file names,

- **working programmatically** with file names and regex.

Order Friendly

Start with a number if there is a logical order to your files (e.g steps in an analysis).

Original

```
diagnositc-plots.R
download.R
runtime-comparison.R
...
model-evaluation.R
wrangle.R
```

Refined

```
00_download.R
01_wrangle.R
02_model.R
...
09_model-evaluation.R
10_model-comparison-plots.R
```

Order Friendly

Start with a date for chronologically ordered files (e.g. data, versions, regular reports)

> `2015-10-22_human-resources-protocols.pdf`
>
> `2022-effective-data-science-module-guide.docx`

The ISO 8601 standard for dates was created for a reason. Use it.

YYYY-MM-DD

Human Readable

File names should be meaningful, informative and

```
easilyReadByRealPeople (camelCase)

EasilyReadByRealPeople (PascalCase)

easily_read_by_real_people (snake_case)

easily-read-by-real-people (skewer-case)
```

Bad news for `untitled31.R`, `FinalreportV8.docx` and `7032-185.txt`.

Look into slugs (web URLs, not the animals) for further tips.

Naming Files - Style Guide Summary

1. File names should be meaningful, informative and scripts end in `.r`
2. Stick to letters, numbers underscores (_) and hyphens (–).
3. Pay attention to capitalisation `file.r` ≠ `File.r` on all operating systems.
4. Show order with left-padded numbers or ISO dates.

Original:

```
abstract.docx
Effective Data Science's module guide
2022.docx
fig 12.png
Rplot7.png
1711.05189.pdf
HR Protocols 2015 FINAL (Nov 2015).pdf
```

Refined:

```
2015-10-22_human-resources-protocols.pdf
2022_effective-data-science-module-guide.docx
2022_RSS-conference-abstract.docx
fig12_earthquake-timeseries.png
fig07_earthquake-location-map.png
ogata_1984_spacetime-clustering.pdf
```

Part 2: File Extensions and Where You Work

What comes after the dot?

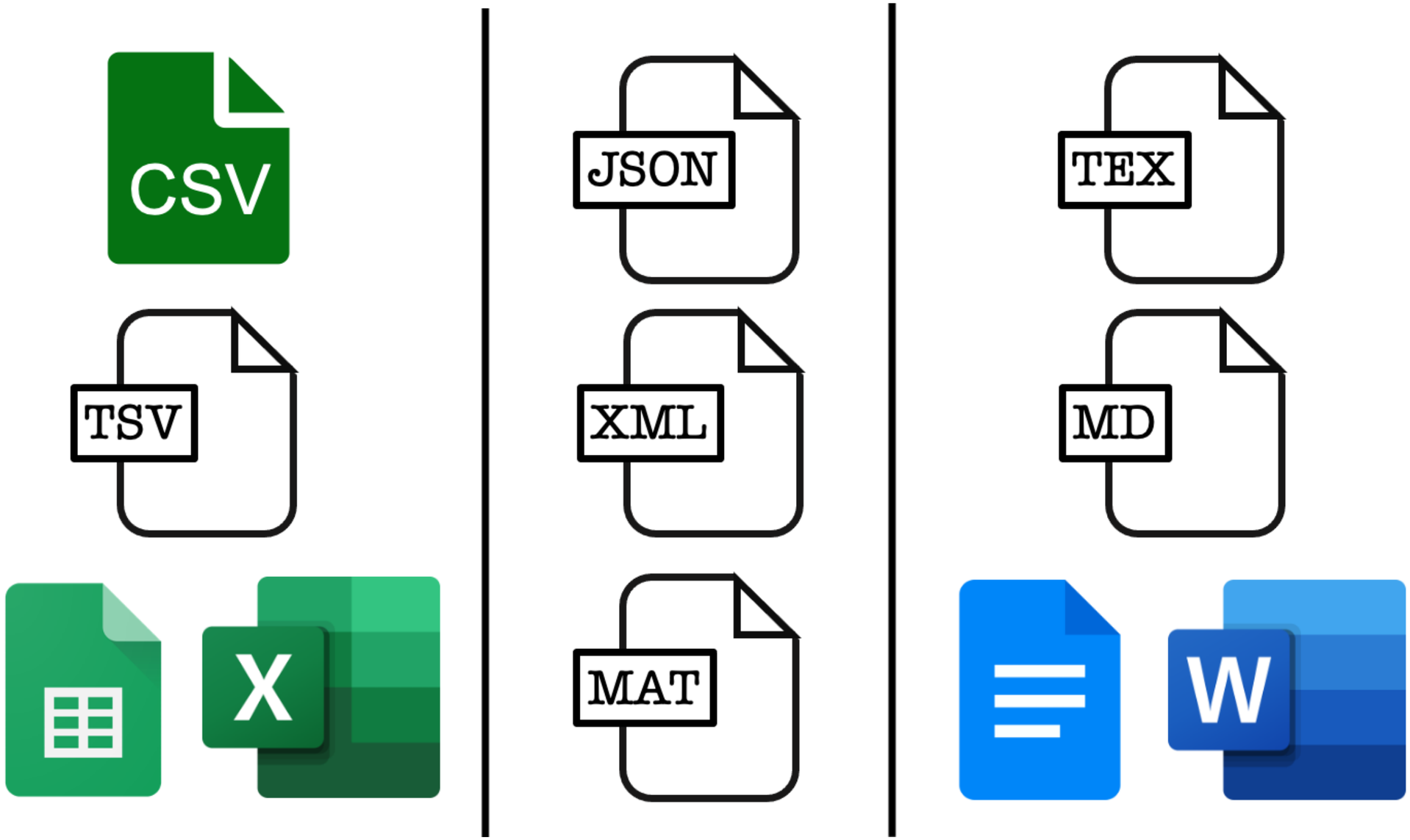
So far we have focused entirely on what comes before the dot, the file name.

Equally, if not more, important is what comes after the dot, the file extension.

```
2022-10-31_earthquake-data.csv
01-download.R
2022_annual_report.docx
```

The file extension describes how information is stored in that file and determines what software can be used view or run it.

Open Source vs Proprietary File Types



Inside a CSV file

Commas separate values, one line per record.

```
1 library(readr)
2 read_file(file = "images/example.csv")
```

```
[1] "Name,Number\r\nA,1\r\nB,2\r\nC,3"
```

```
1 Name,Number
2 A,1
3 B,2
4 C,3
```

Inside a TSV file

Tabs separate values, one line per record.

```
1 library(readr)
2 read_file(file = "images/example.tsv")
```

```
[1] "Name\tNumber\r\nA\t1\r\nB\t2\r\nC\t3"
```

```
1 Name    Number
2 A   1
3 B   2
4 C   3
```

Inside an Excel File

Issues caused by potential for formatting and multiple sheets.

Carrying around a lot more data than is needed. (8.7 KB vs 29B)

```
504b 0304 1400 0600 0800 0000 2100 62ee
9d68 5e01 0000 9004 0000 1300 0802 5b43
6f6e 7465 6e74 5f54 7970 6573 5d2e 786d
6c20 a204 0228 a000 0200 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
.... .... .... .... .... .... .... ....
0000 0000 0000 0000 ac92 4d4f c330 0c86
ef48 fc87 c8f7 d5dd 9010 424b 7741 48bb
2154 7e80 49dc 0fb5 8da3 241b ddbf 271c
1054 1a83 0347 7fbd 7efc cadb dd3c 8dea
.... .... .... .... .... .... .... ....
```

Inside a JSON file

Each record is a collection of `key:value` pairs.

Can be nested to store `list`-like or `dictionary`-like objects.

```
[{
    "Name": "A",
    "Number": "1"
}, {
    "Name": "B",
    "Number": "2"
}, {
    "Name": "C",
    "Number": "3"
}]
```

Inside a XML file

Also a collection of `key:value` pairs, but formatted differently.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <row>
    <Name>A</Name>
    <Number>1</Number>
  </row>
  <row>
    <Name>B</Name>
    <Number>2</Number>
  </row>
  <row>
    <Name>C</Name>
    <Number>3</Number>
  </row>
</root>
```

A note on notebooks

- There are two and a half notebook formats that you are likely to use to: `.rmd` (alternatively `.qmd`) and `.ipynb`.
- R markdown documents `.rmd` are plain text files, so are very human friendly.
- JuPyteR notebooks have multi-language support but are not so human friendly (JSON in disguise).
- Quarto documents offer the best of both worlds and more extensive language support. Not yet as established as a format.

File extensions and where you code

Property	Notebook	Script	Command Line
reproducible	~	✓	X
readable	~	✓	~
self-documenting	✓	X	X
in production	X	✓	~
ordering / automation	~	✓	~

Summary

Name files so that they are:

- Machine Readable,
- Human Readable,
- Order Friendly.

Use document types that are:

- Widely accessible,
- Easy to read and reproduce,
- Appropriate for the task at hand.