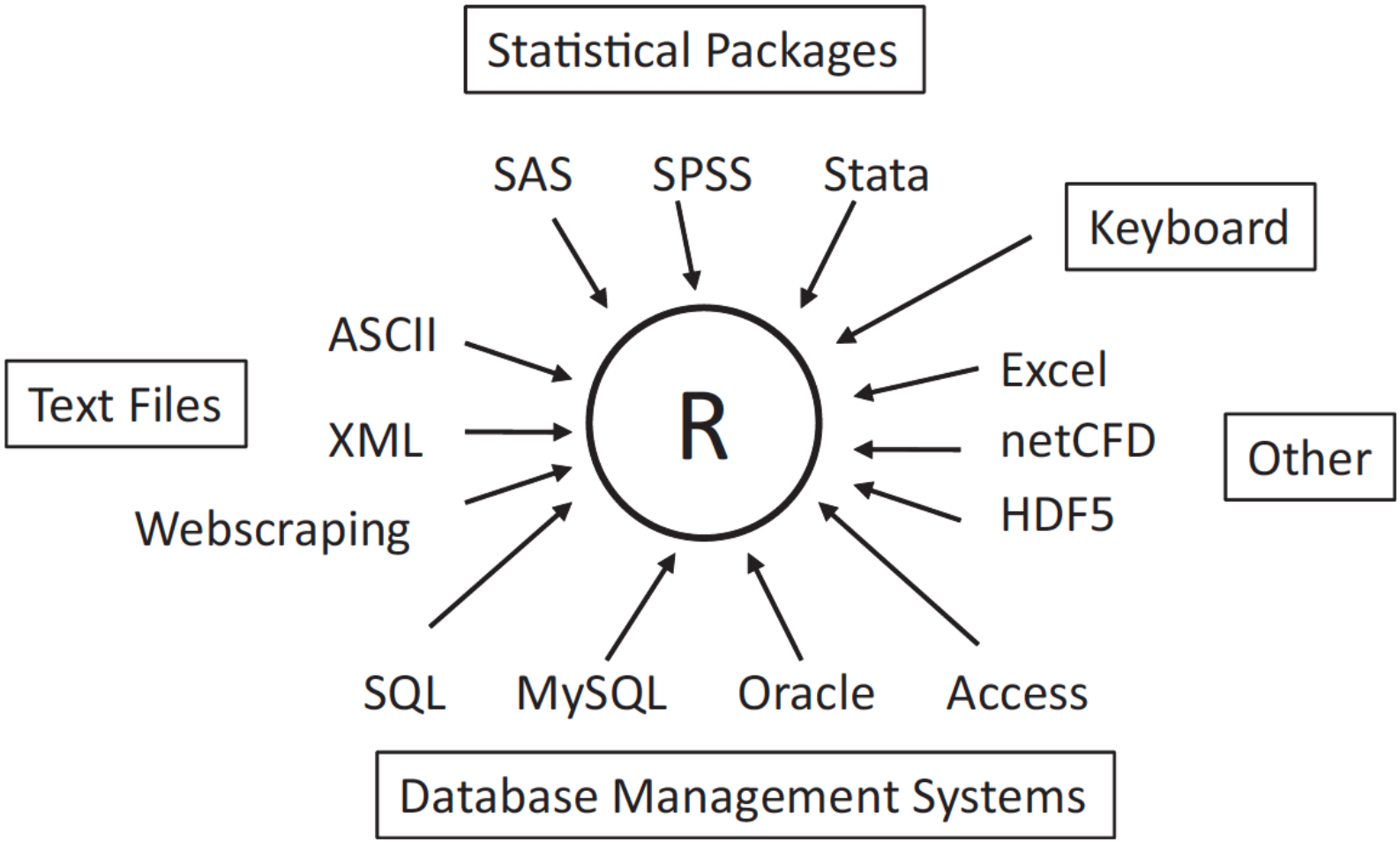
Subsection 3

Importing Data

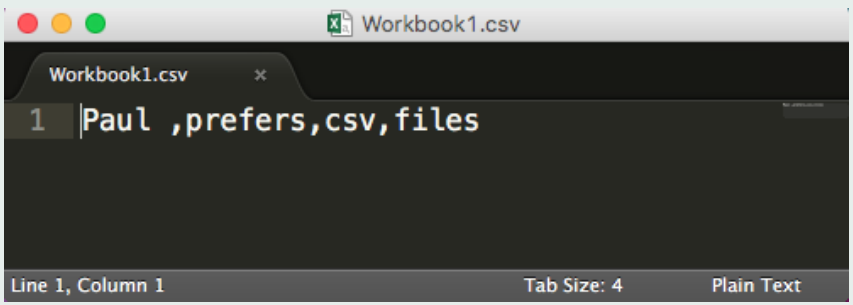
R can Import Data from a Multitude of Sources



Why csv Files?

How large is a file that contains the four separate words "*Paul prefers csv files*"?

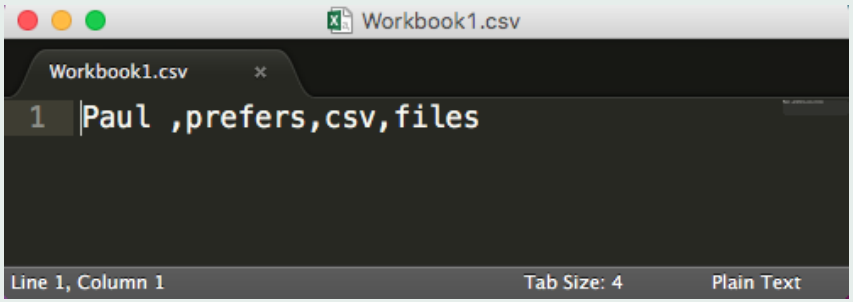
csv File (.csv)
23 bytes



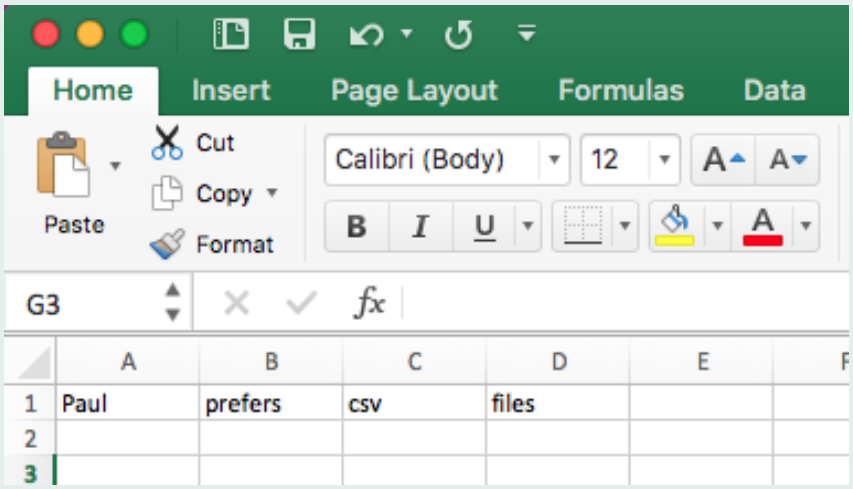
Why csv Files?

How large is a file that contains the four separate words "*Paul prefers csv files*"?

csv File (.csv)
23 bytes



Excel File v15 macOS (.xlsx)
22,687 bytes



Importing Data

- You can import data directly from an Excel file if you like using `read.xlsx()` from the `xlsx` package, or from SQL using the `RODBC` package
- The cleanest (easiest?) and most universally-portable way to move data from one system/technology to another is using delimited text files
- A comma-separated value (`csv`) file is one type of text-delimited file, where the delimiter is a comma

n.b. Don't be fooled: even though the file has a `.csv` extension, this *is* a text file that can be opened and edited with any text editor

- Other common text-delimited file types include tab- (`.tsv`) and space-delimited files
  - these files may alternatively have a `.txt` file extension

read.table()

- Perhaps the most flexible way to read a text-based file into R
- The default separator for `read.table()` is white space, i.e., `sep = ""` looks for one or more spaces, tabs, newlines or carriage returns to identify a new entry
- One has the flexibility to set `sep` to whatever separator the file uses, e.g., `sep = ","` for commas
- `read.table()` also accepts `url` addresses
- the `header` option, `FALSE` by default, indicates whether the file has a header
- `colClasses` can be used to create a of classes to be assumed for the columns
- There are **many** more options for `read.table()` that you should explore by reading the documentation

Stylized Variants of `read.table()`

- `read.csv()` is the equivalent of `read.table()`, but the default separator is `sep = ","`, which saves you having to type that extra bit of code
- `read.csv2()` is the equivalent of `read.table()`, but the default separator is `sep = ";"` and the default character assumed for decimal points is `dec = ","`
- `read.delim()` is the equivalent of `read.table()`, but the default separator is `sep = "\t"`
- `read.delim2()` is the equivalent of `read.table()`, but the default separator is `sep = "\t"` and the default character assumed for decimal points is `dec = ","`

readr

Part of the tidyverse, the readr package offers the ability to read data into R far more quickly and conveniently than base R functions

- read_csv() reads comma-delimited files
- read_csv2() reads semicolon-separated files
- read_tsv() reads tab-delimited files
- read_delim() reads files in any types of delimiter
- read_tsv() reads fixed-width files
- These functions have been shown to be $\sim 10\times$ faster than their base R counterparts, and also produce tibbles instead of data frames (and therefore don't automatically convert character vectors to factors)
- Need even more speed? Try fread() from the data.table package

Beyond Delimited Files

- `haven` is a great package to read SPSS, Stata and SAS files (which can get messy)
- An alternative package to read in Excel files is `readxl`
- `DBI` facilitates writing SQL queries for a backend database
- `jsonlite` is great when working with `JSON` data