Data cleaning and management

Data wrangling

- Data wrangling is the art of getting your data into Python in a useful form for visualisation and modelling. Data wrangling is very important: without it you can’t work with your own data!
- The three main parts are import, tidy, transform.
- Data transformation is slightly different to what you might think! You will learn to select important variables, filter out key observations, create new variables, and compute summaries.

Import data

First you must **import** your data into Python. This typically means that you take data stored in a file, database, or web API, and load it into a data frame in Pandas in Python. If you can’t get your data into Python, you can’t do data science on it!

Tidy data

Once you’ve imported your data, it is a good idea to **tidy** it.

Tidying your data means storing it in a consistent form that matches the semantics of the dataset with the way it is stored.

In brief, when your data is tidy, each column is a variable, and each row is an observation.

Tidy data is important because the consistent structure lets you focus your struggle on questions about the data, not fighting to get the data into the right form for different functions.

Transform data

Once you have tidy data, a common first step is to **transform** it.

Transformation includes narrowing in on observations of interest (like all people in one city, or all data from the last year), creating new variables that are functions of existing variables (like computing speed from distance and time), and calculating a set of summary statistics (like counts or means).

Together, tidying and transforming are called **wrangling**, because getting your data in a form that's natural to work with often feels like a fight!

When you have tidy data

Once you have tidy data with the variables you need, there are two main engines of knowledge generation: **visualisation** and **modelling**.

These have complementary strengths and weaknesses so any real analysis will iterate between them many times.

Visualising data

Visualisation is a fundamentally human activity. A good visualisation will show you things that you did not expect, or raise new questions about the data.

A good visualisation might also hint that you’re asking the wrong question, or you need to collect different data.

Visualisations can surprise you, but don’t scale particularly well because they require a human to interpret them.

Modelling data

Models are complementary tools to visualisation. Once you have made your questions sufficiently precise, you can use a model to answer them. Models are a fundamentally mathematical or computational tool, so they generally scale well. Even when they don’t, it’s usually cheaper to buy more computers than it is to buy more brains! But every model makes assumptions, and by its very nature a model cannot question its own assumptions.

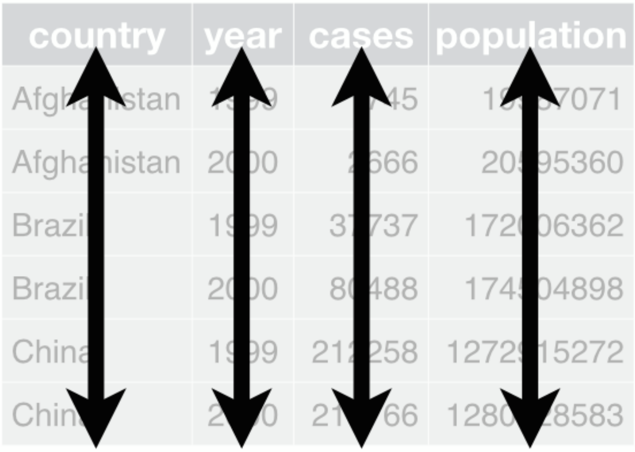
Communicating results

The last step of data science is **communication**, an absolutely critical part of any data analysis project.

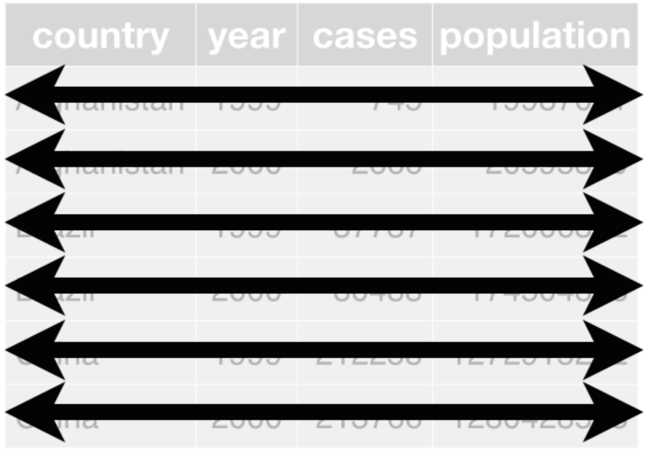
It doesn’t matter how well your models and visualisation have led you to understand the data unless you can also communicate your results to others!

Good data management

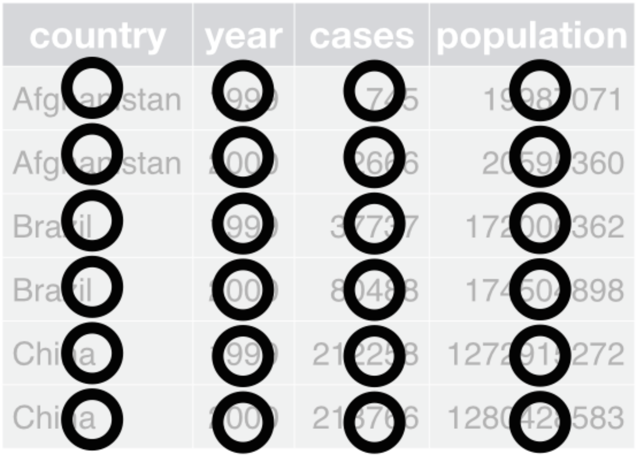
- Observations in rows, variables in columns.
- One column for one variable.



variables



observations



values

Good data management

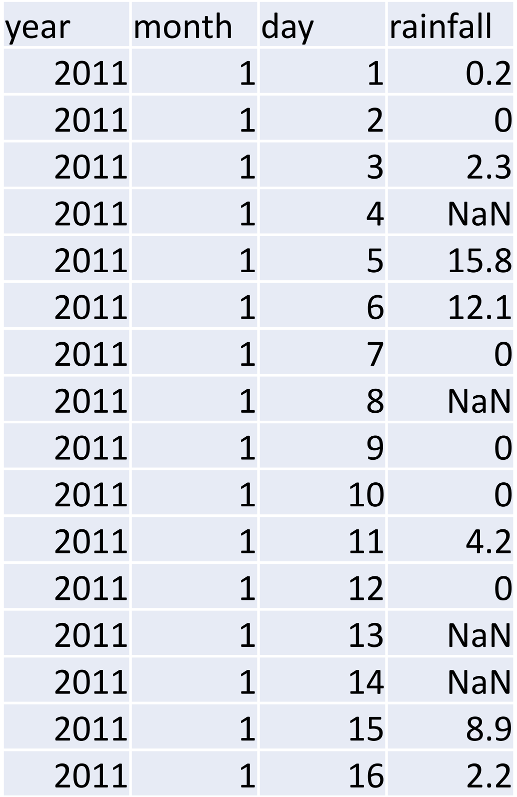
- Meaningful column names without spaces (use underscores), and consistent letter case.

year	month	net_profit
2011	1	123
2011	2	234
2011	3	424
2011	4	176
2011	5	454
2011	6	439
2011	7	391
2011	8	280
2011	9	302
2011	10	408
2011	11	515
2011	12	523

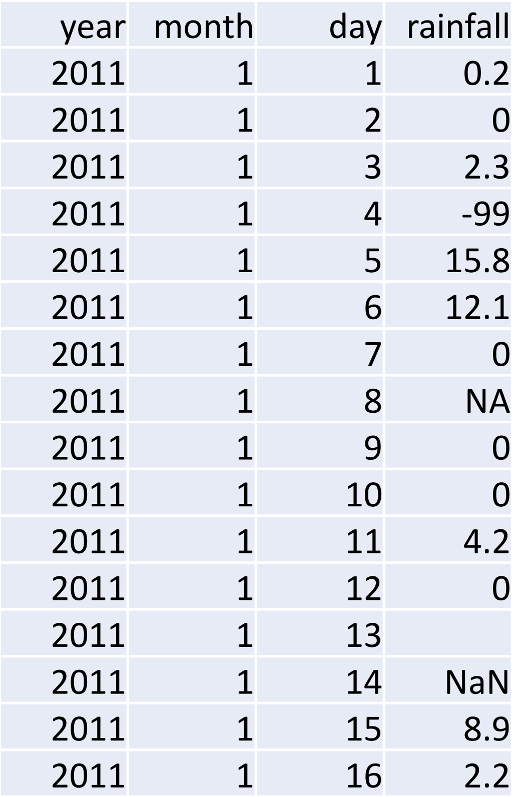
y	m	val net
2011	1	123
2011	2	234
2011	3	424
2011	4	176
2011	5	454
2011	6	439
2011	7	391
2011	8	280
2011	9	302
2011	10	408
2011	11	515
2011	12	523

Good data management

- Consistent missing values convention (best to use NaN in Python). Do not leave gaps.



year	month	day	rainfall
2011	1	1	0.2
2011	1	2	0
2011	1	3	2.3
2011	1	4	NaN
2011	1	5	15.8
2011	1	6	12.1
2011	1	7	0
2011	1	8	NaN
2011	1	9	0
2011	1	10	0
2011	1	11	4.2
2011	1	12	0
2011	1	13	NaN
2011	1	14	NaN
2011	1	15	8.9
2011	1	16	2.2



year	month	day	rainfall
2011	1	1	0.2
2011	1	2	0
2011	1	3	2.3
2011	1	4	-99
2011	1	5	15.8
2011	1	6	12.1
2011	1	7	0
2011	1	8	NA
2011	1	9	0
2011	1	10	0
2011	1	11	4.2
2011	1	12	0
2011	1	13	
2011	1	14	NaN
2011	1	15	8.9
2011	1	16	2.2

Good data management

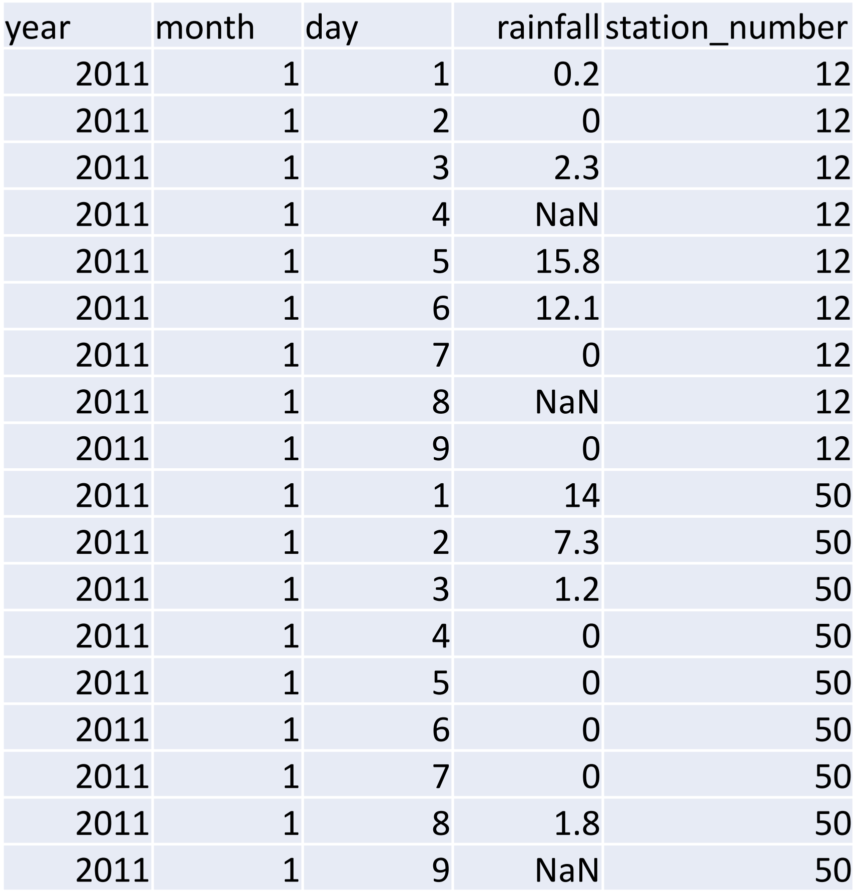
- Usually best to have continuous dates, and include missing dates with NaNs elsewhere.

year	month	day	rainfall
2011	1	1	0.2
2011	1	2	0
2011	1	3	2.3
2011	1	4	NaN
2011	1	5	15.8
2011	1	6	12.1
2011	1	7	0
2011	1	8	NaN
2011	1	9	0
2011	1	10	0
2011	1	11	4.2
2011	1	12	0
2011	1	13	NaN
2011	1	14	NaN
2011	1	15	8.9
2011	1	16	2.2

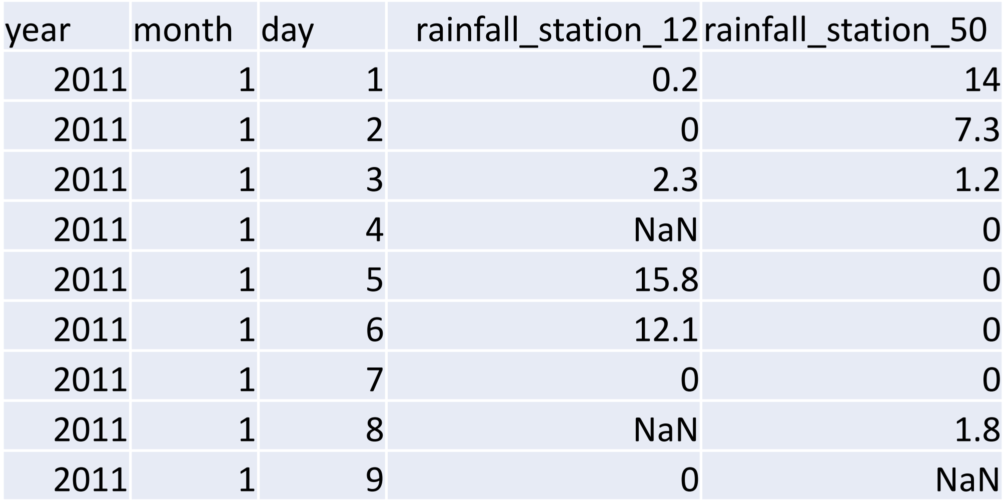
year	month	day	rainfall
2011	1	1	0.2
2011	1	2	0
2011	1	3	2.3
2011	1	5	15.8
2011	1	6	12.1
2011	1	7	0
2011	1	9	0
2011	1	10	0
2011	1	11	4.2
2011	1	12	0
2011	1	15	8.9
2011	1	16	2.2

Good data management

- Have an indicator column showing levels of factors such as year and groups.



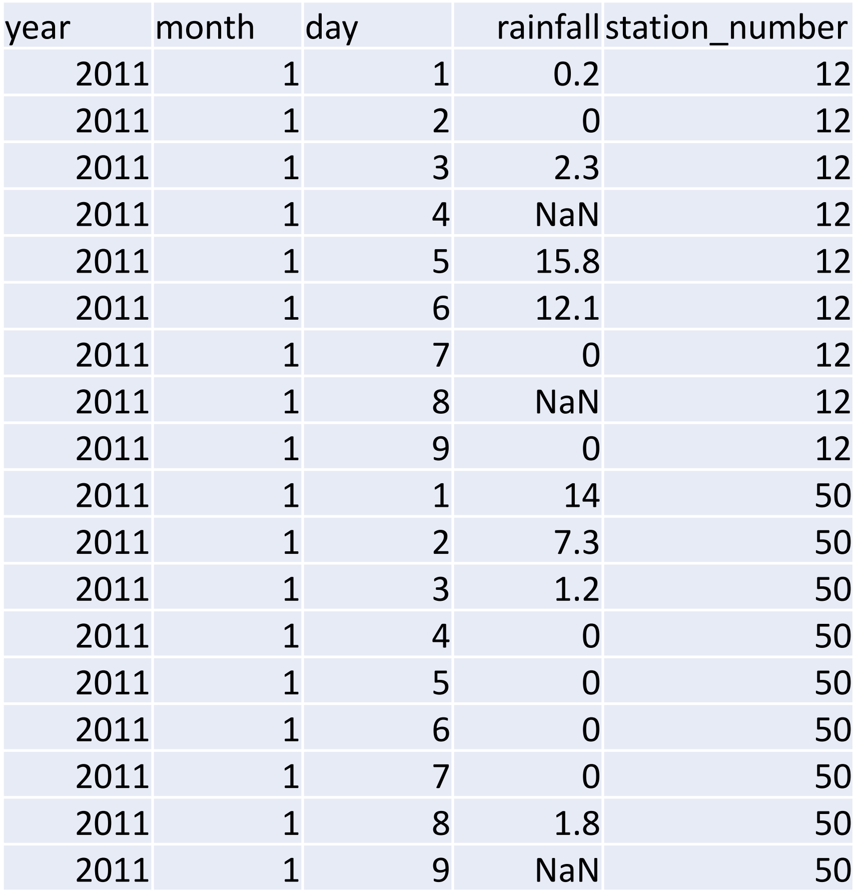
year	month	day	rainfall	station_number
2011	1	1	0.2	12
2011	1	2	0	12
2011	1	3	2.3	12
2011	1	4	NaN	12
2011	1	5	15.8	12
2011	1	6	12.1	12
2011	1	7	0	12
2011	1	8	NaN	12
2011	1	9	0	12
2011	1	1	14	50
2011	1	2	7.3	50
2011	1	3	1.2	50
2011	1	4	0	50
2011	1	5	0	50
2011	1	6	0	50
2011	1	7	0	50
2011	1	8	1.8	50
2011	1	9	NaN	50



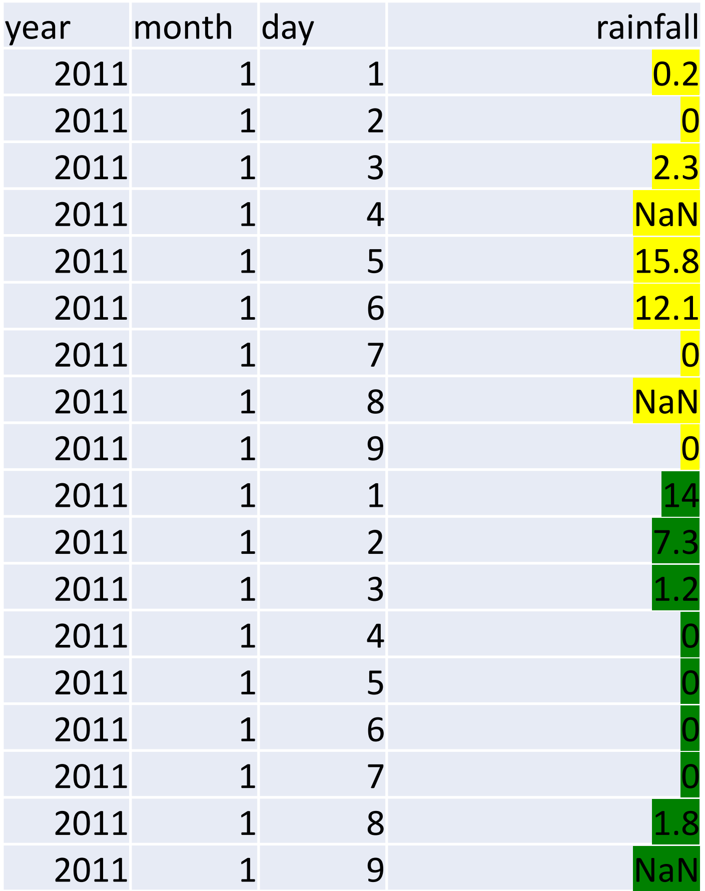
year	month	day	rainfall_station_12	rainfall_station_50
2011	1	1	0.2	14
2011	1	2	0	7.3
2011	1	3	2.3	1.2
2011	1	4	NaN	0
2011	1	5	15.8	0
2011	1	6	12.1	0
2011	1	7	0	0
2011	1	8	NaN	1.8
2011	1	9	0	NaN

Good data management

- Do not use colour to distinguish factor levels!



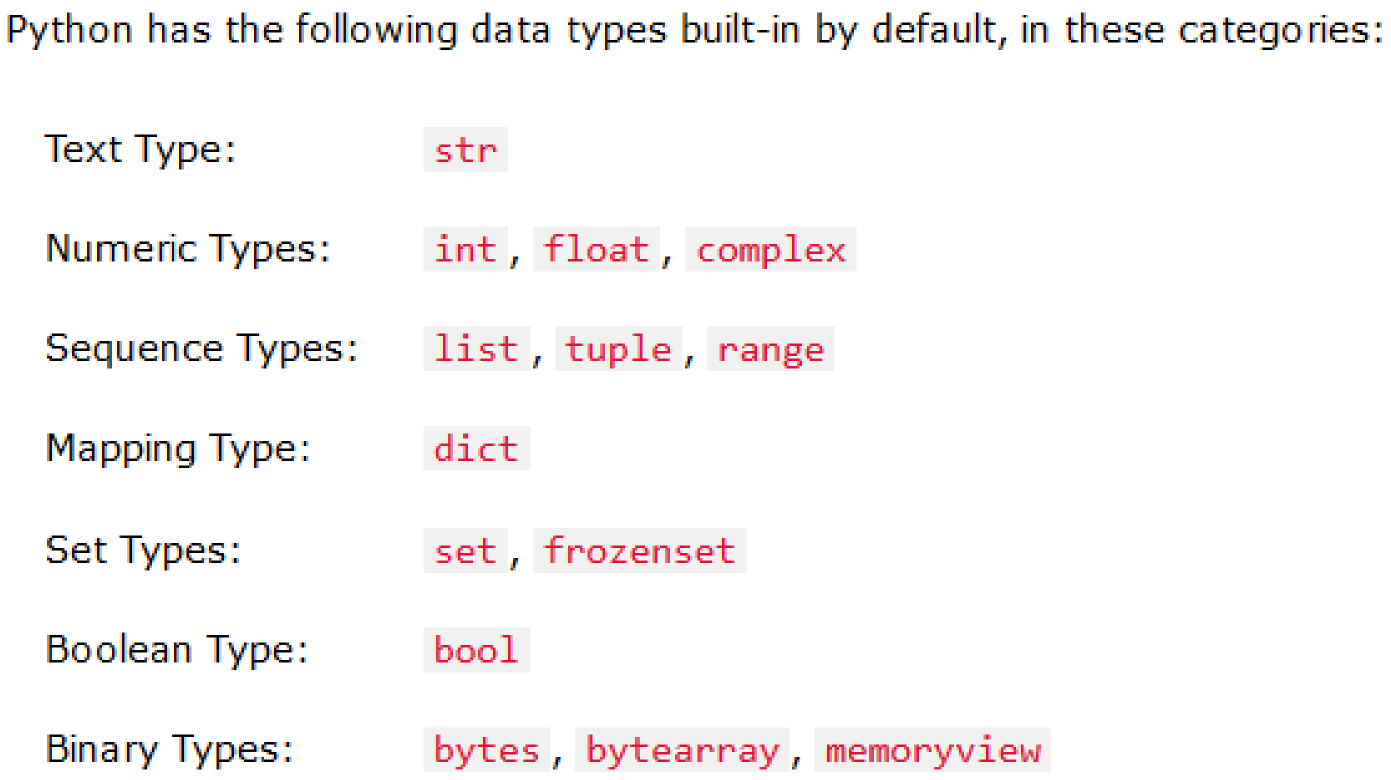
year	month	day	rainfall	station_number
2011	1	1	0.2	12
2011	1	2	0	12
2011	1	3	2.3	12
2011	1	4	NaN	12
2011	1	5	15.8	12
2011	1	6	12.1	12
2011	1	7	0	12
2011	1	8	NaN	12
2011	1	9	0	12
2011	1	1	14	50
2011	1	2	7.3	50
2011	1	3	1.2	50
2011	1	4	0	50
2011	1	5	0	50
2011	1	6	0	50
2011	1	7	0	50
2011	1	8	1.8	50
2011	1	9	NaN	50



year	month	day	rainfall
2011	1	1	0.2
2011	1	2	0
2011	1	3	2.3
2011	1	4	NaN
2011	1	5	15.8
2011	1	6	12.1
2011	1	7	0
2011	1	8	NaN
2011	1	9	0
2011	1	1	14
2011	1	2	7.3
2011	1	3	1.2
2011	1	4	0
2011	1	5	0
2011	1	6	0
2011	1	7	0
2011	1	8	1.8
2011	1	9	NaN

Good data management

- Have your columns in the appropriate data type. Use df.dtypes to see the data type of each column in a Pandas DataFrame called df.



Python has the following data types built-in by default, in these categories:

Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`

Good data management

- Keep a master file that is the raw data file eg. football_scores_master_20201204
- Have a good file naming convention for any edited files, including the date and using underscores eg. football_scores_edit1_20191208.

Good data management

- Use metadata and documentation to make your dataset accessible to all users without you needing to explain it.

- https://www.met.ie/climate/available-data/long-term-data-sets
- This zip file contains data and metadata for the Long term (1850-2010) Island of Ireland Precipitation (IIP) network.
- There are 25 files, NAME.csv, containing monthly rainfall totals for each station in the network, each of these files also contains information on the station location and altitude.
- The file IIP_National series.csv contains the rainfall averaged over the 25 stations.
- The file IIP station metadata.pdf contains station metadata and details of how the series were constructed.

Gridded rainfall exercise on data management

- Read in the gridded data from Moodle (IRL_MON_RR_2011_grid and IRL_MON_RR_2012_grid). They contain the total monthly rainfall value for each of a grid of coordinates (given in eastings and northings) in 2011 and 2012 respectively.
- Is the data in an appropriate format for analysis?
- Tidy the data into an appropriate format.
- The final dataset should be sorted by year, then northing, then easting.
- Please work together and ask me if you have any questions.