Python for scientific research

Data visualisation with `seaborn`

Bram Kuijper

University of Exeter, Penryn Campus, UK

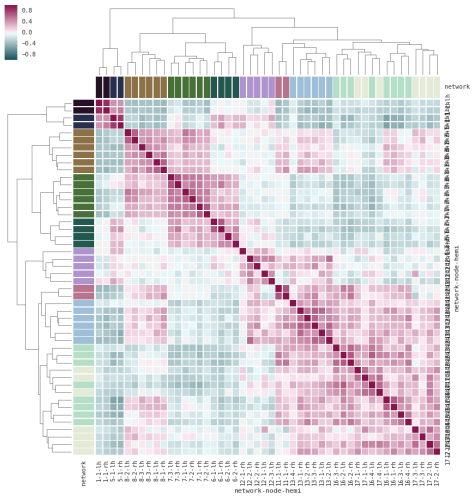
March 3, 2020

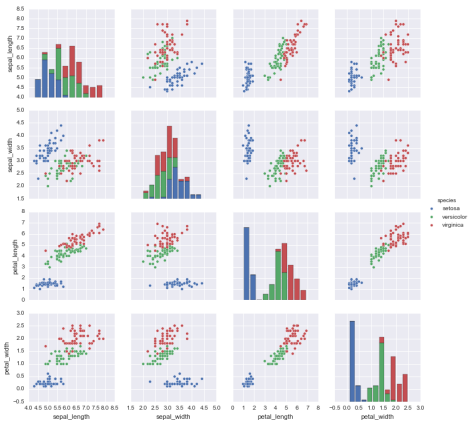
What we've done so far

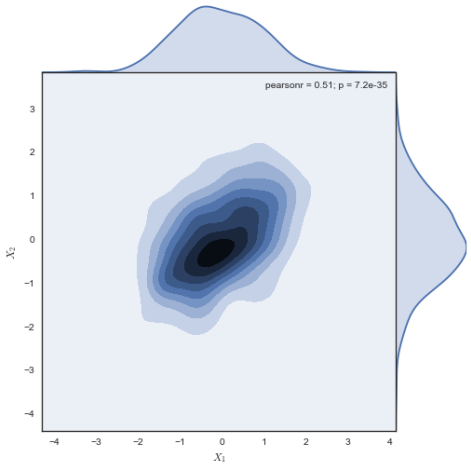
1. Declare variables using built-in data types and execute operations on them
2. Use flow control commands to dictate the order in which commands are run and when
3. Encapsulate programs into reusable functions, modules and packages
4. Use string manipulation and regex to work with textual data
5. Interact with the file system
6. Number crunching using `NumPy/SciPy`
7. Publication-ready graphs with `Matplotlib`
8. Working with data using `pandas`
9. **Next**: Introducing `Seaborn`, an advanced plotting library

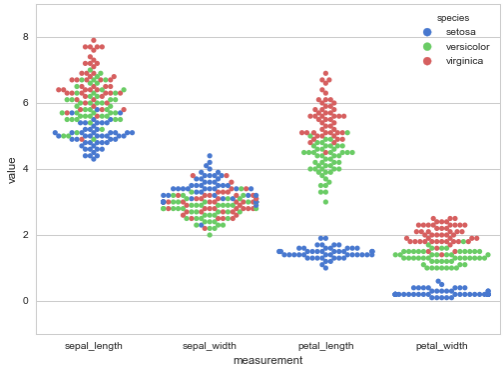
Introduction

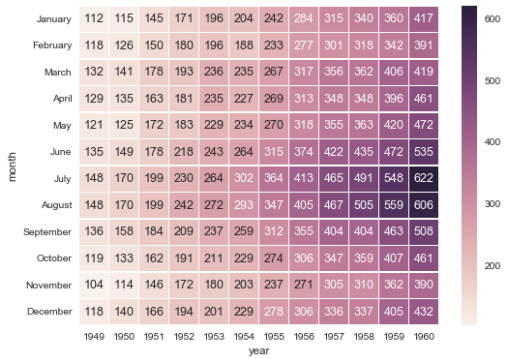
- `Seaborn` is a library built on top of `Matplotlib` for making attractive and informative statistical graphics
- It supports `Numpy` and `Pandas` data structures

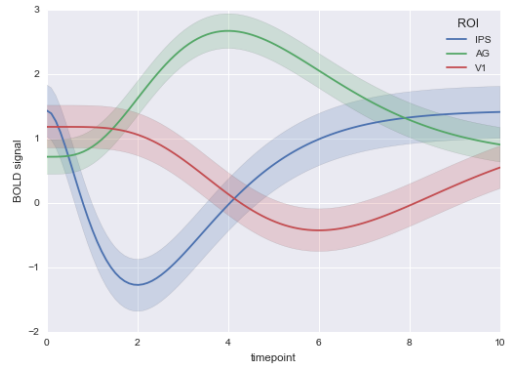












Reading data files: Wine

```
1 import pandas as pd
2
3 # Chemical analysis of wines grown in the same region in
      Italy but
4 # from three different cultivars
5 df = pd.read_csv("wine.csv", header=0)
6 df.head()
```

```
  WineType  Alcohol  MalicAcid   Ash  AlcalinityAsh   Magnesium  \
0        A    14.23       1.71  2.43           15.6         127
1        A    13.20       1.78  2.14           11.2         100
2        A    13.16       2.36  2.67           18.6         101
3        A    14.37       1.95  2.50           16.8         113
4        A    13.24       2.59  2.87           21.0         118

   TotalPhenols   Flavanoids  NonflavanoidPhenols   Proanthocyanins   \
0          2.80         3.06                 0.28              2.29
1          2.65         2.76                 0.26              1.28
2          2.80         3.24                 0.30              2.81
3          3.85         3.49                 0.24              2.18
4          2.80         2.69                 0.39              1.82

   ColorIntensity  Hue   OD280_OD315  Proline
0            5.64  1.04         3.92     1065
1            4.38  1.05         3.40     1050
2            5.68  1.03         3.17     1185
3            7.80  0.86         3.45     1480
4            4.32  1.04         2.93      735
```