# *pandas and data analysis*

*Ben Bolker*

*10:09 23 March 2015*

- pandas cheat sheet

`pandas` stands for **pan**el **da**ta **s**ystem. It's a convenient and powerful system for handling large, complicated data sets.

Download US measles data from Project Tycho.

- `read_csv` reads a CSV file as a **data frame**; it automatically interprets the first row as headings
- `df.iloc[]` indexes the result as though it were an array
- `df.head()` shows just at the beginning; `df.tail()` shows just the end

**Data frames**: * rectangular data structure, a lot like an array. * can have columns (**Series**) of different types * can index by labels as well as positions * handles **missing data** * convenient plotting * fast operations with keys

```
import pandas as pd
v = "MEASLES_Cases_1909-2001_20150322001618.csv"
p  = pd.read_csv(v,skiprows=2,na_values=["-"])  ## read in data
print(p.iloc[:,0:3].head())                     ## look at the first little bit
```

```
##    YEAR  WEEK  ALABAMA
## 0  1909     1      NaN
## 1  1909     2      NaN
## 2  1909     3      NaN
## 3  1909     4      NaN
## 4  1909     5      NaN
```

## *Selecting*

- Pandas doc, indexing and selecting
- Choosing specific columns of a data frame
- `df["NAME"]`: extract one series (column)
- `df.NAME` (*attribute* operator)
- `df.loc[:,"MASSACHUSETTS":"NEVADA"]` (index by *label*; includes endpoint)
- `df.iloc[:,range]` (index by *integer*)

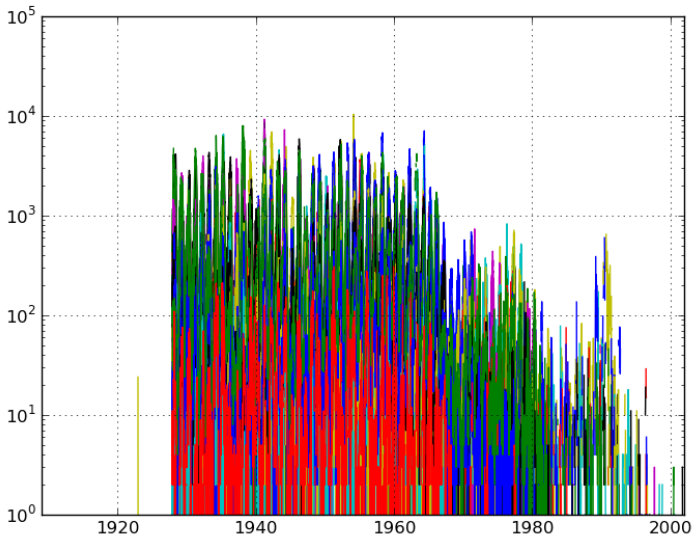
*Filtering*

Choosing specific rows of a data frame; `&`, `|` ,`~` correspond to `and`, `or`, `not` (individual elements *must* be in parentheses)

```
ariz = p.ARIZONA                          ## pull out a column (attribute)
ariz[(p.YEAR==1970) & (ariz>50)]          ## *must* use parentheses!
```

*Basic plotting*

```
pp = p.drop(["YEAR","WEEK"],axis=1)
pp.index = p.YEAR+(p.WEEK-1)/52           ## assign index
pp.plot(legend=False,logy=True)           ## plot method (non-Pythonic)
plt.savefig("pix/measles1.png")
```



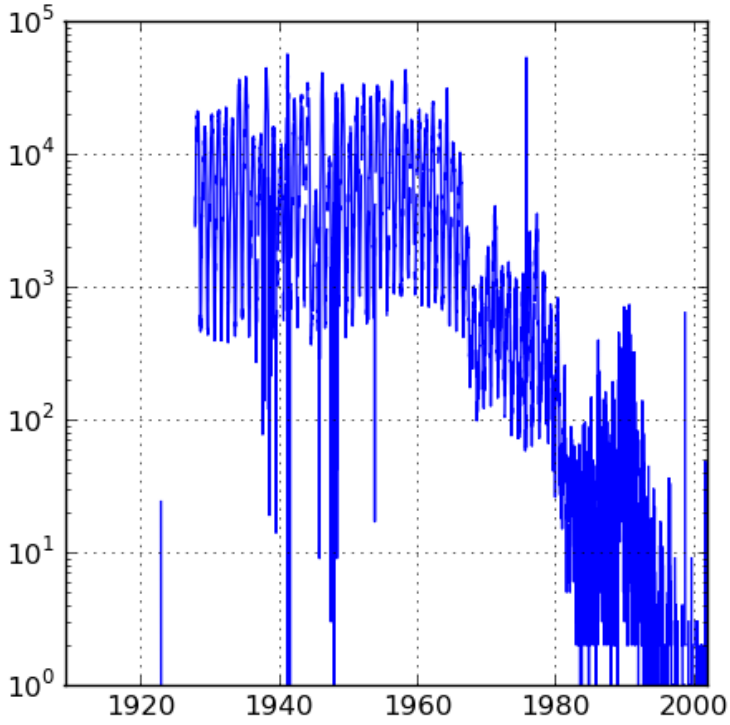

```
fig = plt.figure()
ax = fig.add_subplot(1,1,1)
ax.scatter(pp.index,np.log10(pp.ARIZONA))
```

*Column and row manipulations*

- totals by week

```
ptot = pp.sum(axis=1)
```

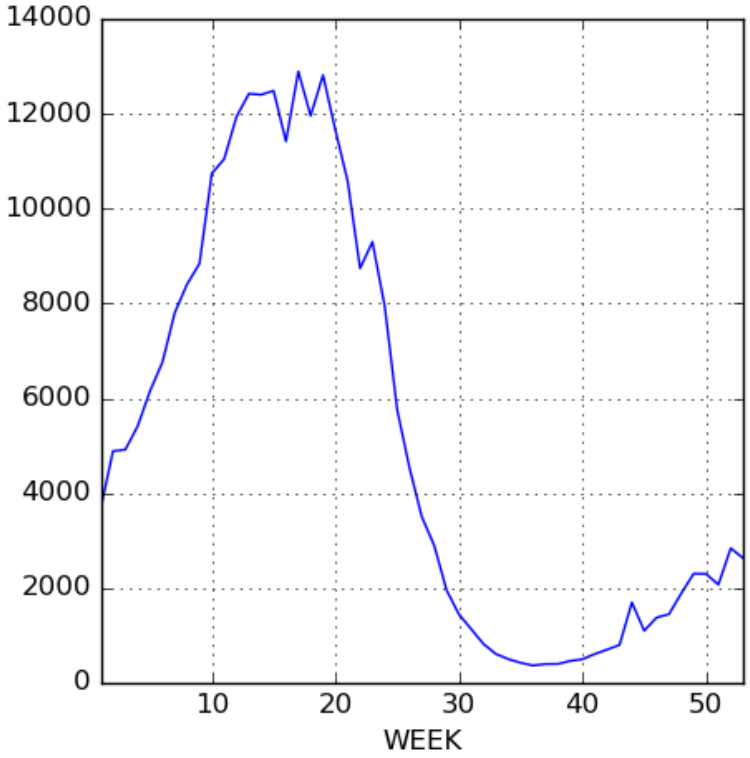
- `df.min`, `df.max`, `df.mean` all work too ...




*Aggregation*

```
ptotweek = ptot.groupby(p.WEEK)
ptotweekmean = ptotweek.aggregate(np.mean)
ptotweekmean.plot()
```

- early, mid, late periods?


14000
12000
10000
8000
6000
4000
2000
0
10
20
30
40
50
WEEK