

時系列データの取得と整形

Acquire and format Time-series data

本日の目標: データを整形するスキルを身につける

Today's Goal: master a skill of data cleansing

- 本日使うデータ: 大気汚染データ / Data to be used today: Air pollution
<http://soramame.taiki.go.jp/>
- 次のようにしてダウンロードできる / You can download like:
http://soramame.taiki.go.jp/Download/201701/201701_00.zip
- 2017年のデータをダウンロードする。 / Download data in 2017
 - csh

```
% foreach m ( 01 02 03 04 05 06 07 08 09 10 11 12 )  
foreach? wget http://soramame.taiki.go.jp/Download/2017${m}/2017${m}_00.zip  
foreach? end
```
 - sh

```
$ for m in 01 02 03 04 05 06 07 08 09 10 11 12  
> do  
> wget http://soramame.taiki.go.jp/Download/2017${m}/2017${m}_00.zip  
> done
```
- データはZIP圧縮されたファイル / The data is compressed as ZIP file

本日の目標: データを整形するスキルを身につける

Today's Goal: master a skill of data cleansing

- データはZIP圧縮されたファイル。展開する必要がある。
The data is compressed as ZIP file. You must uncompress them.
 - csh

```
% mkdir data
% foreach f ( *.zip )
  foreach? unzip $f -d data
foreach? end
```
 - sh

```
$ mkdir data
$ for f in *.zip
> do
> unzip $f -d data
> done
```
- "data" ディレクトリの中に展開される。 / data will be stored in "data" directory.
- 564個 (12*47個) のファイルが得られる。 / You may have 564 (12*47) files.
 - 47は都道府県の数。 / 47 is the number of prefectures.

本日の目標: データを整形するスキルを身につける

Today's Goal: master a skill of data cleansing

- しかし、展開されたデータも圧縮されている。
But expanded data is also compressed.
- これらも展開する。 / You must expand them too.

– csh

```
% mkdir csv
% foreach f ( data/*.zip )
foreach? unzip $f -d csv
foreach? end
```

– sh

```
$ mkdir csv
$ for f in data/*.zip
> do
> unzip $f -d csv
> done
```

- 36425個のファイルが得られる。You may have 36425 files.

– csv/14/201701_14_14205040.csv

YYYYMM 県番号 観測局番号
Pref No. Station No.

データの概観

Outline of data

- ファイルはいくつある? / How many files did you get?

- "ls csv/*/ * | wc -l" ではだめ。なぜ?

You cannot get the result using "ls csv/*/ * | wc -l". Why?

```
% find csv -print | wc -l
```

- 観測局毎にいくつのデータがあるか?

How many record per station?

```
% cat csv/14/*_14204010.csv | wc -l
```

- 結果にはcsvヘッダを含むので注意。The result includes csv headers.

- データの頭の方、終わりの方 / head and tail of data

```
% cat csv/14/*_14204010.csv | head -5 | nkf
```

```
% cat csv/14/*_14204010.csv | tail -5 | nkf
```

- "nkf"は文字コード変換の為。 / "nkf" is used to convert character code.

Colaboratoryを使ってデータをプロットしてみる (1)

Plot the data using Colaboratory (1)

- 1地点のデータを1つに結合する。 / concatenate data


```
% ( head -1 csv/14/201701_14_14204010.csv ;  
      nkf csv/14/*_14_14204010.csv | grep -v S02 ) >  
2017_14204010.csv
```

- Colaboratory

```
# upload a file  
from google.colab import files  
uploaded = files.upload()
```

```
# read csv  
import pandas as pd  
import io  
df = pd.read_csv(io.StringIO(uploaded['2017_14204010.csv'].decode('Shift_JIS')))
```

```
# add new column YYYYMMDD-HH:00:00  
df = pd.concat([df, df["日付"].str.split("/", expand=True)], axis=1)  
df.rename(columns={0: "YYYY", 1: "MM", 2: "DD"}, inplace=True)  
df["日時"] = df["YYYY"] + "-" + df["MM"] + "-" + df["DD"] + " " + (df["時"]-  
1).astype(str)+ ":00:00"  
df["日時"] = pd.to_datetime(df["日時"])  
df = df.set_index("日時")
```



記述統計量

Description Statistics

- 平均値

`df.mean()`

- 中央値

`df.median()`

- 分散

`df.var()`

- 標準偏差

`dv.std()`

Colaboratoryを使ってデータをプロットしてみる (2)

Plot the data using Colaboratory (2)

- Colaboratory

```
# plot
import matplotlib.pyplot as plt;
plt.scatter(df.index, df["PM2.5(ug/m3)"])
plt.show()
```


Colaboratoryを使ってデータをプロットしてみる (3)

Plot the data using Colaboratory (3)

• Colaboratory

```
# make daily data
daily = pd.DataFrame(df["PM2.5(ug/m3)"].resample("D").mean())
- M, D, H, min, ...
- mean, max, min, median, sum, first, last, ...

# plot daily data
plt.scatter(daily.index, daily["PM2.5(ug/m3)"])
plt.show()
```

気象データ

Weather data

- 次のURLから気象データを取得することができる。
You can download weather data from following URL
 - <http://www.data.jma.go.jp/gmd/risk/obsdl/index.php>
- 試しに「辻堂」のデータをダウンロードしてみる。
Try downloading "Tsujiido" data.
 - 地点を選ぶ: 神奈川県→辻堂
 - 項目を選ぶ: 日別データ、日平均気温、日最高気温、日最低気温、降水量の日合計、日照時間
 - 期間を選ぶ: 2017年1月1日から2017年12月31日
 - オプション: 利用上注意が必要なデータの扱い=値を表示（格納）しない。、観測環境などの変化の前後で、値が不均質となったデータの扱い=観測環境などの変化前の値を表示(格納)しない。
- データの概観 / Outline of data

```
% wc -l data.csv
% nkf data.csv | head
% nkf data.csv | tail
```

気象データと環境データの統合

Joining Weather data and Environmental data

- 気象データをクレンジング / Cleansing weather data

```
% ( nkf data.csv | grep "年月日" | nkf -s ; tail -365 data.csv ) > weather.csv
```

- 気象データをColaboratoryにアップロードして、結合する。
Upload weather data and join it to air pollution data.

```
# upload the weather data  
uploaded = files.upload()
```

```
# read and format weather data
```

```
weather = pd.read_csv(io.StringIO(uploaded['weather.csv'].decode('Shift_JIS')), dtype = None)  
weather["年月日"] = pd.to_datetime(weather["年月日"])  
weather = weather.set_index("年月日")
```

```
# join the data  
data = daily.join(weather)
```

```
# rainfall and PM2.5
```

```
plt.scatter(data.index, data["PM2.5(ug/m3)"])  
plt.scatter(data.index, data["降水量の合計(mm)"])  
plt.show()
```

```
# 相関? / correlation?  
data.corr()
```

データの結合 (keyを使う場合) Join data (use key)

- データの作成 / Make data

```
df1 = pd.DataFrame({"key": ["a", "b", "c"], "data1": range(3)})
```

```
df2 = pd.DataFrame({"key": ["a", "b", "a", "b", "d", "a"], "data2": range(6)})
```

- Inner Join

```
pd.merge(df1, df2)
```

```
pd.merge(df1, df2, on="key")
```

```
# pd.merge(df1, df2, left_on='key1', right_on='key2')
```

- Left Outer Join

```
pd.merge(df1, df2, how="left")
```

- Right Outer Join

```
pd.merge(df1, df2, how="right")
```

- Outer Join

```
pd.merge(df1, df2, how="outer")
```

データの結合 (keyを使わない場合) Join data (don't use key)

- Join: indexを使って結合 / Join: use index

```
df1 = pd.DataFrame({"key": ["a", "b", "c"], "data1": range(3)})  
df1 = df1.set_index("key")  
df2 = pd.DataFrame({"key": ["a", "b", "a", "b", "d", "a"], "data2": range(6)})  
df2 = df2.set_index("key")
```

```
df1.join(df2, how="left")  
df1.join(df2, how="right")  
df1.join(df2, how="outer")  
df1.join(df2, how="inner")
```

- Concat: Uniqueなindexを持つデータの連結 / concatenation of data which have unique index

```
df1 = pd.DataFrame({"key": ["a", "b", "c"], "data1": range(3)})  
df1 = df1.set_index("key")  
df2 = pd.DataFrame({"key": ["a", "b", "d", "e"], "data2": range(4)})  
df2 = df2.set_index("key")
```

```
pd.concat([df1, df2])           // 縦に繋ぐ  
pd.concat([df1, df2], axis=1)  // 横に繋ぐ
```

課題1

Assignment 1

- 大気汚染データと天候データを使ってデータ分析をし、レポートを作成しなさい。

Analyze and make a report about air pollution data and weather data.

- <http://soramame.taiki.go.jp/>
- <http://www.data.jma.go.jp/gmd/risk/obsdl/index.php>

- ① 解析の目的について述べなさい。 / What is your purpose of analysis?
- ② 解析の手順について述べなさい。 / Process of Analysis
- ③ わかったことを述べなさい。 / What do you find in the analysis?
- ④ やってみた感想を述べなさい。 / What kind of impression did you have?

- 注意 / Notation

- 必ず、氏名・学籍番号を記載のこと。 / You MUST put your name and student id number.
- いわゆるレポートの形で作成すること。(スライド不可) / Slide is not acceptable. Make report.

- 締め切り: 5/18 (金) 23:59 / Deadline: 18th May (Fri) 23:59