情報管理

第３回：時系列データの可視化と相関

今回の講義内容

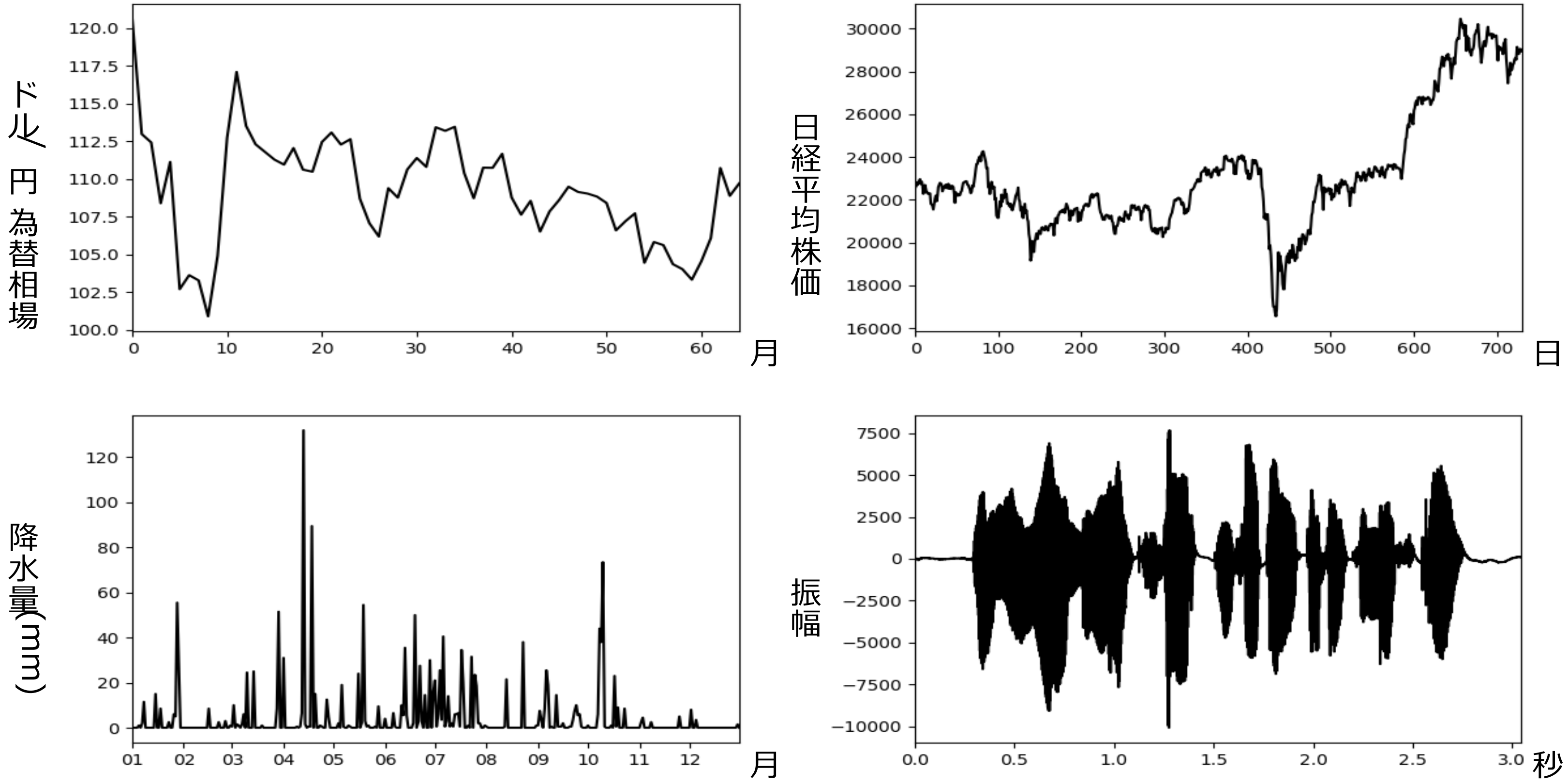
前回まではpython の使い方説明が中心でしたが，今回からは本格的にデータの解析方法について勉強していきます。

今回は以下の内容を学んでもらいます。

- 時系列データの可視化
  - 長期的な変化と短期的な変化の可視化：移動平均と階差
  - 変化傾向の数値化：直線近似と曲線近似
- データ間の関係性の分析
  - 相関係数
  - 相互相関関数

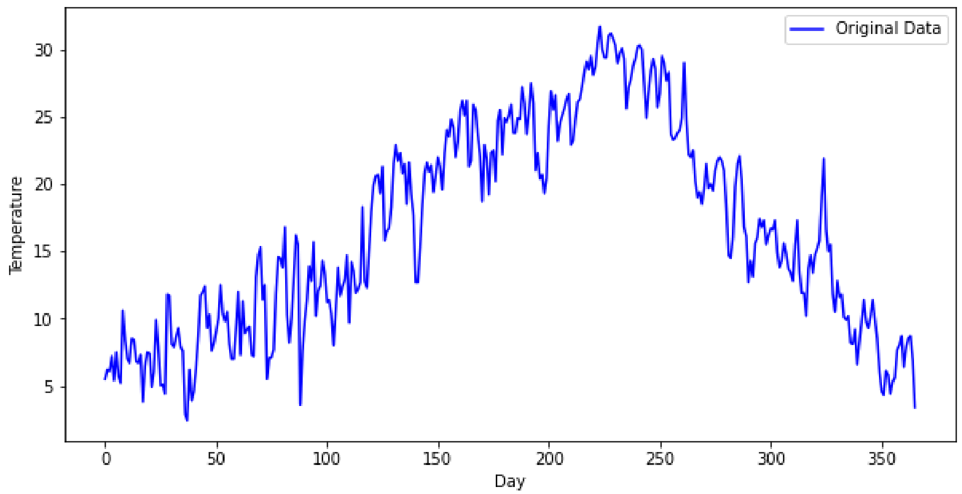
時系列データとは？

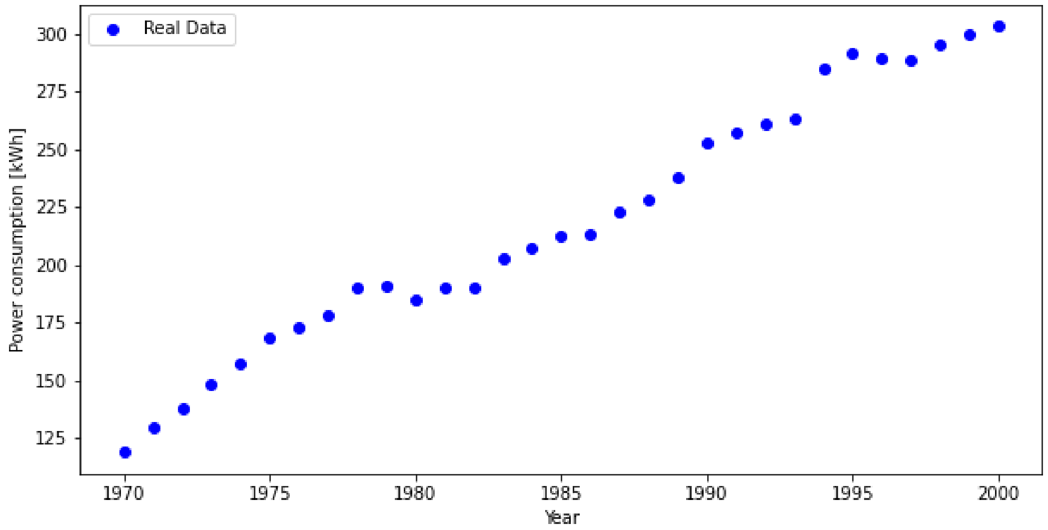
時間毎に変化する現象を一定間隔で記録したものを**時系列データ**と呼びます。
前回扱った気温データは，１日間隔で記録した時系列データとなります。
為替相場，株価のような経済データや気温・降水量のような気象データ，
音声や心電図のような生体情報も時系列データです。



時系列データの可視化

時系列データをそのままプロットするだけでも分かる情報はありますが，
様々な処理を行うことで，特定の情報を際立たせて見やすくできます。



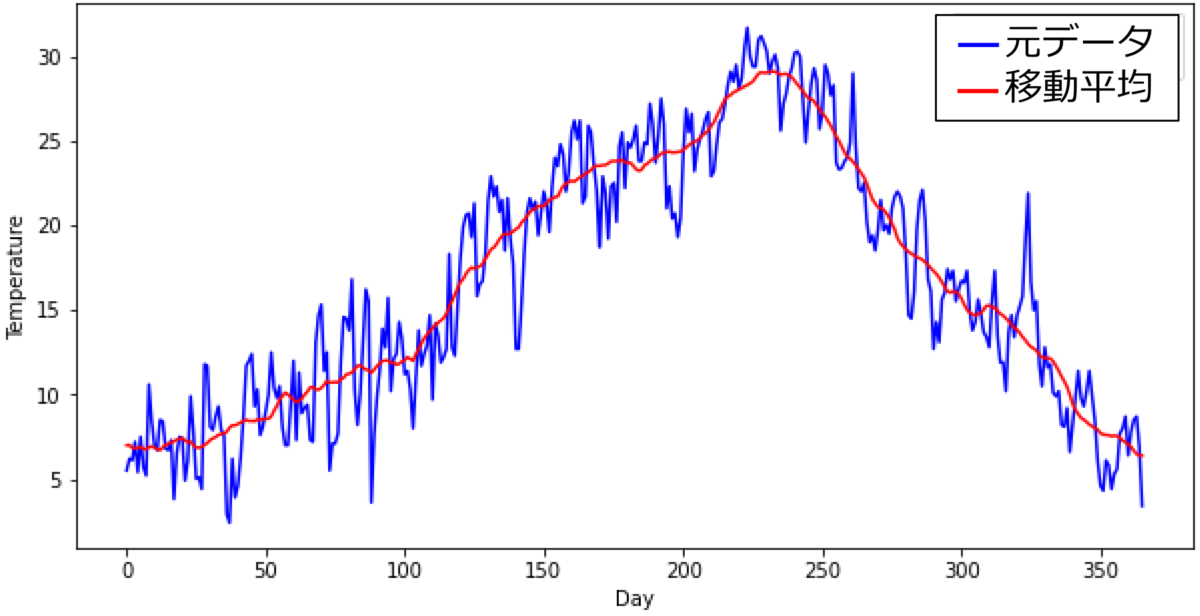


大まかにはどのように変化している？
→長期的変動の可視化
どれくらい小刻みに変化している？
→短期的変動の可視化

どれくらいの割合で増加している？
→ 傾きの数値化

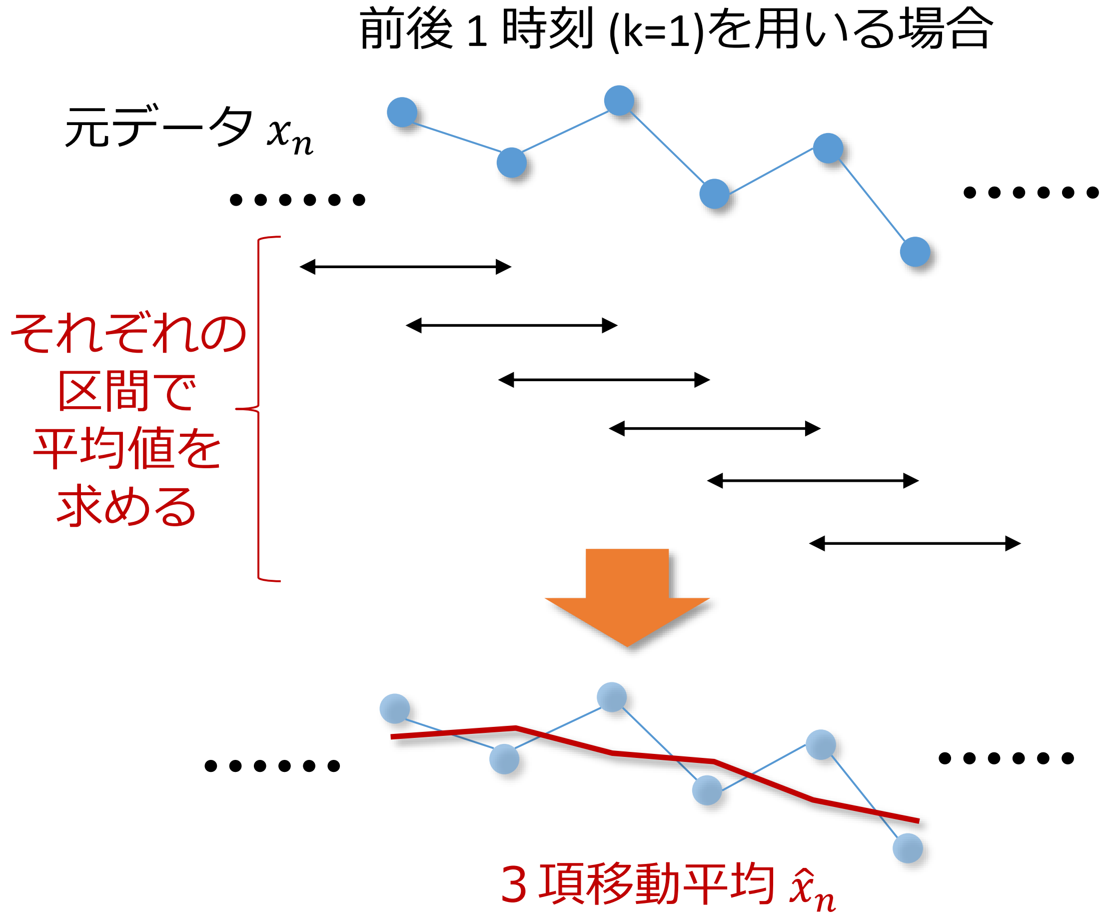
時系列データの可視化方法 1-1：移動平均

細かい変動成分を抑える（平滑化）ことで，大まかな変化傾向（＝長期的変動）を見やすくする方法です。



時系列データの可視化方法 1-1：移動平均

ある時刻に対して，その前後いくつかの時刻の値を用いて，平均を計算します。



$$\hat{x}_n = \frac{1}{2k+1}\underbrace{(x_{n-k} + \cdots + x_n + \cdots + x_{n+k})}$$

現在の時刻 n に対して，
前後 k 時刻の値を用いて
平均を計算する。

⭐**前後２方向にk項！**

前後 k 時刻を用いた場合，
K = 2k+1 個のデータで平均計算することになる。
→ **K項移動平均**と呼ぶ。

時系列データの可視化方法 1-1：移動平均

移動平均は計算方法によって，さらに以下の３種類に分けられます。

- 前後の時刻を用いる場合：中央移動平均

  前ページで紹介した方法。既に収集完了した区間のデータに対して
  よく用いられる。

- 後ろの時刻のみを用いる場合：前方移動平均

- 前の時刻のみを用いた場合：後方移動平均

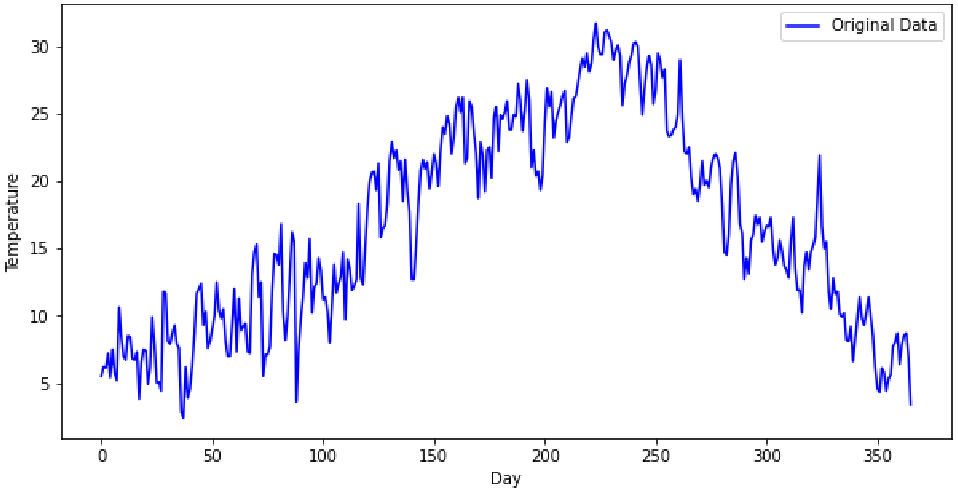
  後ろの時刻が存在しない＝現在進行形で収集・分析をしている場合において
  よく用いられる（例：COVID-19感染者の分析）。

時系列データの可視化方法 1-1：階差

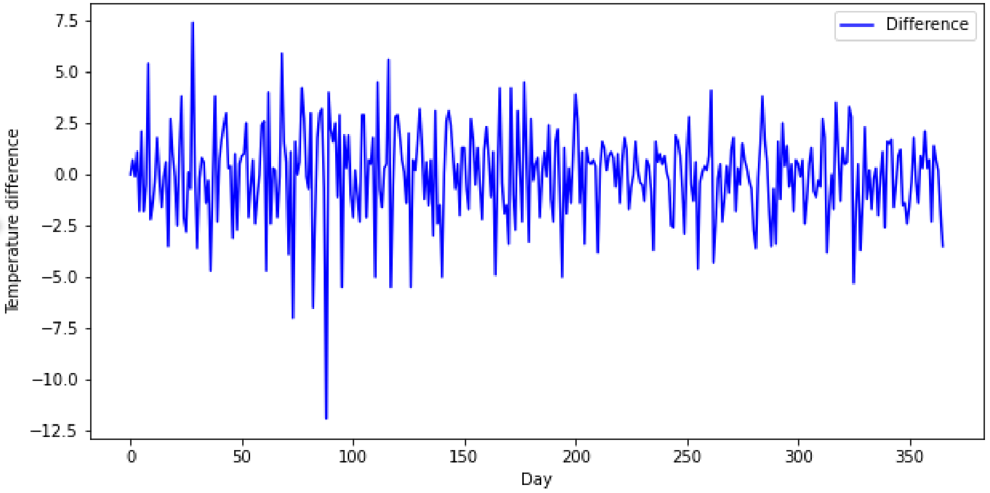
現在の時刻から一つ前の時刻の値との差を求めることで，細かい変化（＝短期的変動）を見やすくする方法です。

$$\Delta x_n = x_n - x_{n-1}$$

元データ x_n



階差 Δx_n

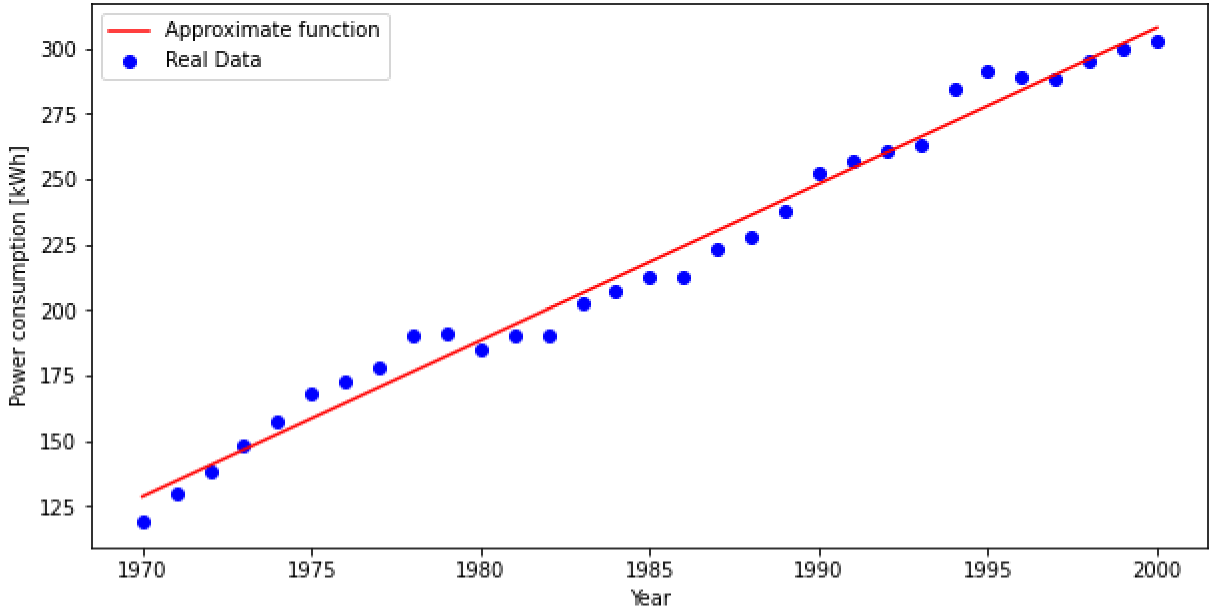


気温データに対して移動平均と階差を計算しよう

03_01_moving_average.ipynb を動かして，気温データに対する移動平均と階差を計算し，どのようなことが分かるかを見てみましょう。

時系列データの可視化方法2-1：直線近似

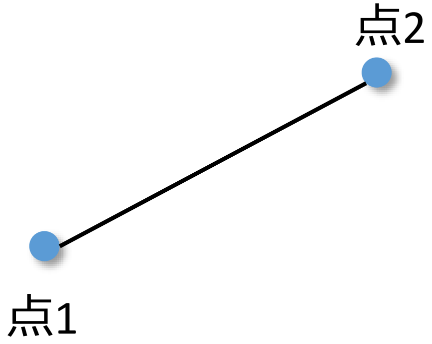
時系列データを直線（線形関数：$y = ax + b$）で近似してみましょう。
直線近似することで，時系列データの増減傾向や全体的な大きさが
傾き a と切片 b によって数値化できます。



時系列データの可視化方法2-1：直線近似

中学・高校数学では，「点1と点2を通る直線を求めよ」という問題で，
連立方程式を解くことで直線を求めていたと思います。
しかし，点が沢山ある場合，全てを通る直線を引くことは通常不可能です。
そこで，できるだけ**全ての点に近い場所を通る直線を引く**ことを考えます。

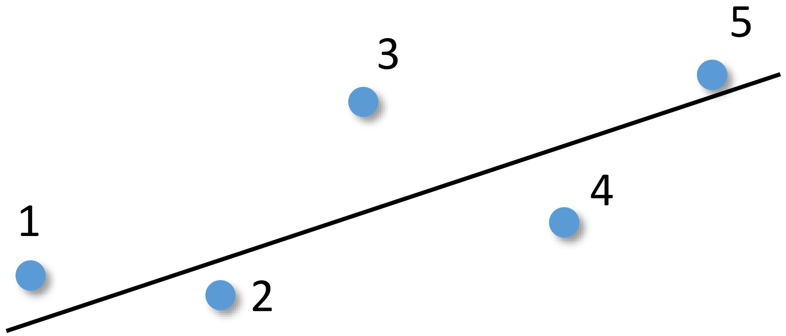
「2点を**通る**直線を求めよ」



$$\begin{cases} y_1 = ax_1 + b \\ y_2 = ax_2 + b \end{cases}$$

連立方程式を解けば良い

「全ての点の**近くを通る**直線を求めよ」



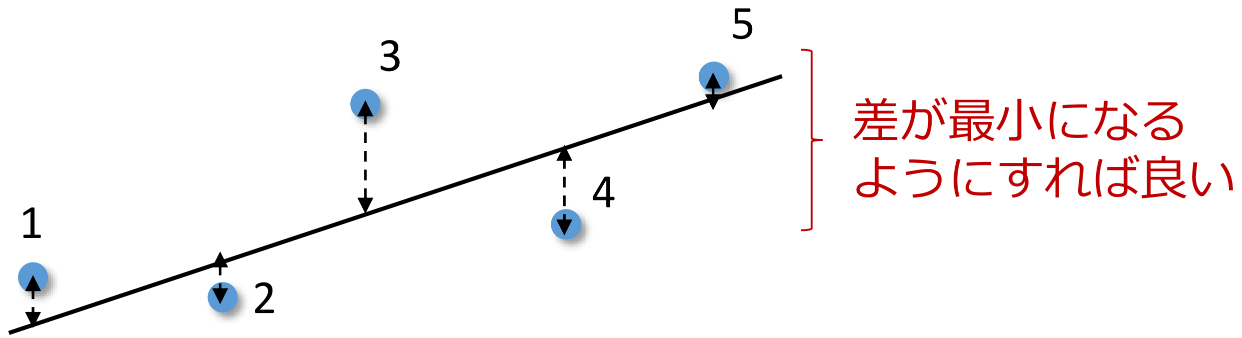
$$\left. \begin{aligned} y_1 &= ax_1 + b \\ y_2 &= ax_2 + b \\ y_3 &= ax_3 + b \\ y_4 &= ax_4 + b \\ y_5 &= ax_5 + b \end{aligned} \right\}$$

全てを満たす
a, b は存在しない！

時系列データの可視化方法2-1：直線近似

できるだけ全ての点に近い場所を通る直線を引く

→直線 $ax+b$ が計算した y と，実際のデータの y との誤差が小さければよい。



つまり，以下の式を満たす a, b を求めればよい。

$$\sum_{\underbrace{n}_{\text{データ数で総和}}} \big(\underbrace{y_n}_{\text{実際の}y} - \underbrace{(ax_n + b)}_{\text{直線が計算した}y}\big)^2 \rightarrow min$$

誤差の二乗を最小化することから，これを **最小二乗法** と呼びます。

時系列データの可視化方法2-1：直線近似

最小二乗法を用いて，傾き a と切片 b を求めます。

$$L = \sum_n \big(y_n - (ax_n + b)\big)^2 = \sum_n (y_n^2 + a^2x_n^2 + b^2 - 2ax_ny_n + 2abx_n - 2by_n) \rightarrow min$$

L は a に対する下に凸の2次関数であり，また b に対しても下に凸の2次関数である。
よって，最小値を求めるにはa, b で偏微分し，=0を解けばよい。

L を a に対して偏微分し，= 0とおく

$$\frac{\partial L}{\partial a} = 2a\sum_n x_n^2 - 2\sum_n x_ny_n + 2b\sum_n x_n = 0$$

$$\rightarrow a\sum_n x_n^2 - \sum_n x_ny_n + b\sum_n x_n = 0 \quad \cdots\cdots \quad ①$$

L を b に対して偏微分し，= 0とおく

$$\frac{\partial L}{\partial b} = 2Nb + 2a\sum_n x_n - 2\sum_n y_n = 0$$

$$\rightarrow a\sum_n x_n + Nb - \sum_n y_n = 0 \quad \cdots\cdots \quad ②$$

時系列データの可視化方法2-1：直線近似

$$a\sum_n x_n^2 - \sum_n x_n y_n + b\sum_n x_n = 0 \quad \cdots\cdots \text{①}$$

$$a\sum_n x_n + Nb - \sum_n y_n = 0 \quad \cdots\cdots \text{②}$$

上記の連立方程式を解くと，以下の結果が得られます（導出は省略）

直線近似の傾きと切片

x_n: n番目のデータのｘ軸の値
y_n: n番目のデータのｙ軸の値
N: データの総数

$$a = \frac{N\sum_n x_n y_n - \sum_n x_n \sum_n y_n}{N\sum_n x_n^2 - (\sum_n x_n)^2}$$

$$b = \frac{\sum_n x_n^2 \sum_n y_n - \sum_n x_n y_n \sum_n y_n}{N\sum_n x_n^2 - (\sum_n x_n)^2}$$

時系列データの可視化方法2-1：直線近似

Q. なぜ誤差の二乗を最小化するのか？

A. いくつか理由があります。

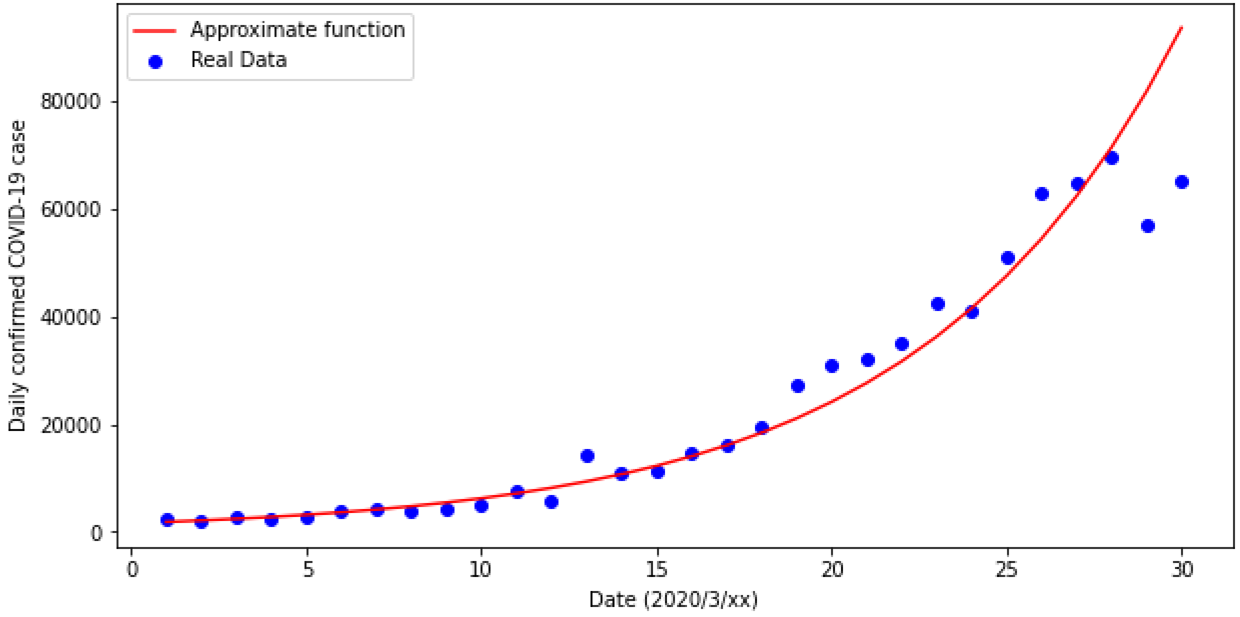
- 誤差を二乗すればかならずゼロ以上の値になるので，誤差の正負を気にしなくて良い。
- 最適化関数が下に凸の二次関数になるため，偏微分して =0 を求める方法が取れるため，使い勝手が良い。

誤差の二乗のほかに，誤差の絶対値を取る方法などもあります。

最小二乗法は今後も出てきますので，理解しておいてください。

時系列データの可視化方法2-1：曲線近似

時系列データを曲線で近似する場合，いくつか種類が考えられます。
（例：2次関数，3次関数，…，n次関数，対数関数など）
ここでは，指数関数 $y = be^{ax}$ を使用します。



時系列データの可視化方法2-1：曲線近似

指数関数で近似する場合，データ全体に対数を計算することで，直線近似と同じ問題にすることができます。

$$y = be^{ax}$$

両辺の対数を取る

$$\begin{aligned} \log_e y &= \log_e be^{ax} \\ &= \log_e b + \log_e e^{ax} \\ &= \log_e b + ax \\ &= B + ax \end{aligned}$$

つまり，データ y に対して対数を取ったデータ $\log_e y$ に対して，
$B + ax$ 直線近似を行う。その後， $b = e^B$ を計算すれば， a, b が求まる。

直線近似を動かしてみよう

03_02_approximate_function.ipynb を動かして，直線近似の様子をみてみましょう。

[補足] どの程度うまく近似できているか？

近似関数の値と実際の値との誤差を計算することで，近似がどの程度うまくできているかを評価できます。

平均二乗誤差（Mean square error: MSE）

実際の値　近似関数の値

$$\mathrm{MSE} = \frac{1}{N}\sum_{n=1}^{N}(y_n - \tilde{y}_n)^2$$

最小二乗法において最小化している式そのもの。

平方根平均二乗誤差 (Root mean square error: RMSE)

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$

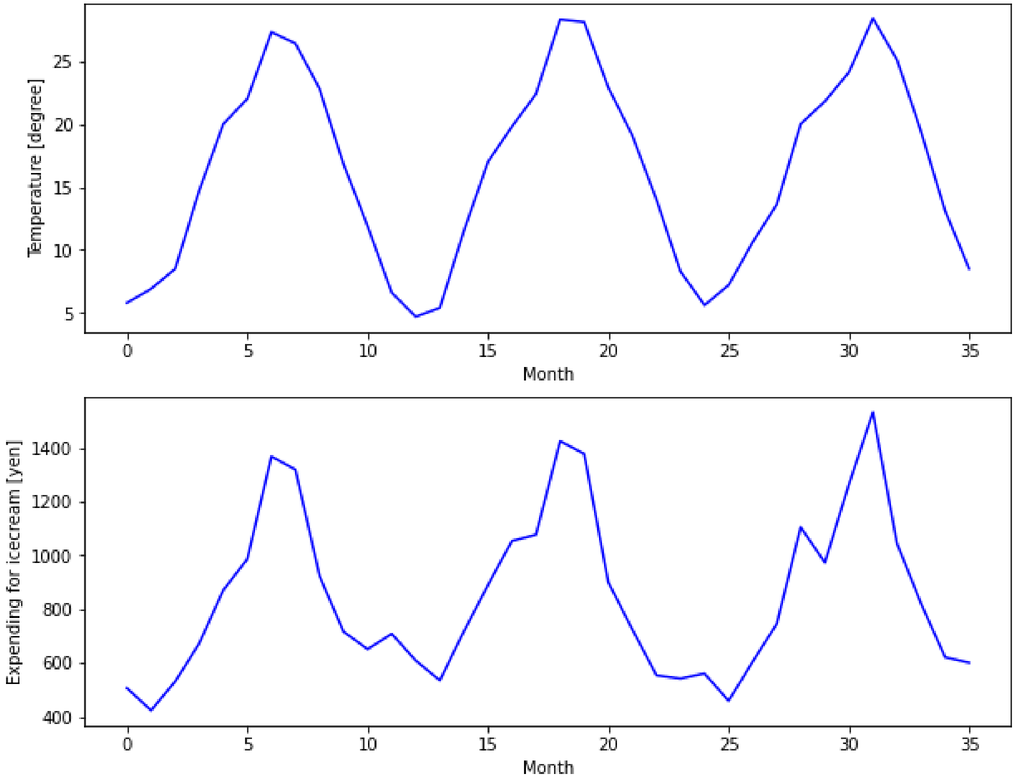
平均二乗誤差の平方根を取ることで，目的変数と同じ単位で誤差が測れる。

データ間の関係性１：相関係数

アイスクリームは夏の暑い時期に良く売れ，逆に寒い冬の時期にはあまり売れません。

このことから，アイスクリームの売り上げと気温には強い関係があると想像できます。

3年間の気温の変化(上)とアイスの売上(下)



一方が増えると他方も増え，

一方が減ると他方も減るというように，

増減の仕方が似ているデータ同士を

「**相関**がある」と呼びます。

データ間の関係性1：相関係数

相関の強さを数値化したものとして，**相関係数**があります。

相関係数Rは以下の式で計算されます。

x, y それぞれの平均値

$$R = \frac{\sum_n (x_n - \bar{x})(y_n - \bar{y})}{\underbrace{\sqrt{\sum_n (x_n - \bar{x})^2}}_{x\text{の標準偏差}}\ \underbrace{\sqrt{\sum_n (y_n - \bar{y})^2}}_{y\text{の標準偏差}}}$$

Rは-1から1までの値を取り，Rの値によって以下のように呼びます。

- Rが正の時：正の相関
- Rが負の時：負の相関
- Rが0の時：無相関

相関係数の直感的説明

$$R = \frac{\sum_n (x_n - \bar{x})(y_n - \bar{y})}{\sqrt{\sum_n (x_n - \bar{x})^2}\sqrt{\sum_n (y_n - \bar{y})^2}}$$

x_nとy_nが両方増加 → $(x_n - \bar{x})$，$(y_n - \bar{y})$どちらも正 → $(x_n - \bar{x})(y_n - \bar{y})$は正

x_nとy_nが両方減少 → $(x_n - \bar{x})$，$(y_n - \bar{y})$どちらも負 → $(x_n - \bar{x})(y_n - \bar{y})$は正

→ x と y が同じ増減傾向をしていれば，$\sum_n (x_n - \bar{x})(y_n - \bar{y})$は大きい正の値になる。

一方，

x_nとy_nの一方が増加，他方が減少 → $(x_n - \bar{x})$，$(y_n - \bar{y})$の一方は正，他方は負 → $(x_n - \bar{x})(y_n - \bar{y})$は負

→ x と y が逆の増減傾向をしていれば，$\sum_n (x_n - \bar{x})(y_n - \bar{y})$は大きい負の値になる。

また，

x と y が同じ増減，逆の増減が入り混じっている場合，つまり統一性が無い場合，

→ $\sum_n (x_n - \bar{x})(y_n - \bar{y})$は 0 になる。

なお，分母の標準偏差は，R を -1 から 1 のレンジに入れる（正規化）ための項である。

データ間の関係性 2：相互相関関数

時系列データ同士で相関を調べる場合，お互いの増減関係に時間差が存在している場合があります。

例：広告を出した数と売上の関係

通常，広告の効果は遅れてやってくるため，広告数の増減と比べて売上は遅れて増減するはず。

時間差を考慮しながら相関関係を調べる方法として，**相互相関関数**があります。

データ間の関係性２：相互相関関数

相互相関関数は以下の式で計算されます。

$$R(m) = \sum_n x_n \underline{y_{n+m}}$$

y_n を 時刻 m ずらしている

相関係数

$$R = \frac{\sum_n (x_n - \bar{x})(y_n - \bar{y})}{\sqrt{\sum_n (x_n - \bar{x})^2}\sqrt{\sum_n (y_n - \bar{y})^2}}$$

相関係数との違いは

- 分母が無い，また分子に対しても平均で引く処理が無い（標準化しない）
- R が m の関数である（m の数だけRを計算している）。
- y が 時刻 m だけずれている。

つまり，一方のデータ(ここでは y) の時刻を m ずらしつつ，相関(ただし標準化なし)を計算している。

[補足] 正規化について

単位（スケール）が異なるデータを同一に扱うと，想定通りの分析ができない場合があります。

	気温 [℃]	アイス支出額 [円]
1	5.8	506
2	6.9	423
3	8.5	531

スケールを合わせるため，データ毎に平均が0，標準偏差が1になるように正規化します。（これを標準化と呼びます。）

$$\boldsymbol{x}_{norm} = \frac{x - (x\text{の平均})}{(x\text{の標準偏差})}$$

相関係数の式は，標準化したデータ同士の内積に相当します。

相互相関関数を計算する際にも事前に標準化しておくと，相関係数のように扱うことができます。

（03_03_correlation.ipynb でも標準化しています。）

相関係数

$$R = \frac{\sum_n (x_n - \bar{x})(y_n - \bar{y})}{\sqrt{\sum_n (x_n - \bar{x})^2}\sqrt{\sum_n (y_n - \bar{y})^2}}$$

相関係数と相互相関関数を求めてみよう

03_03_correlation.ipynb を動かして，相関係数と相互相関関数を求めてみましょう。

おわりに

今回は，主に時系列データの可視化方法と，相関について解説しました。

特に最小二乗法は，以降でもパラメータ推定方法として出てきますので，
しっかり理解しておきましょう。

レポート課題

（ゴールデンウィークを挟むので２題出します。）

課題１：直線近似と曲線近似

課題２：音源方向推定

前回のレポートと同じく， convert_report.ipynb を使って html ファイルに変換してから提出してください。

※課題１，２のそれぞれ個別にBEEFの提出リンクを貼っています。

レポート提出期限：5/10(火) AM10:30

レポート課題１：直線近似と曲線近似

ファイル一式に入っている「covid19.csv」を読み込んでください。

このデータは，2020年3月1日から2020年3月30日までの，全世界のCOVID-19の新規感染者数を日ごとに記録したデータです。

横軸(x)＝日，縦軸(y) ＝新規感染者数としてプロットしたデータに対して，

一次関数 $y = ax + b$ および指数関数 $y = be^{ax}$ を用いてそれぞれ近似し，

その結果を比較・考察しなさい。

ヒント：

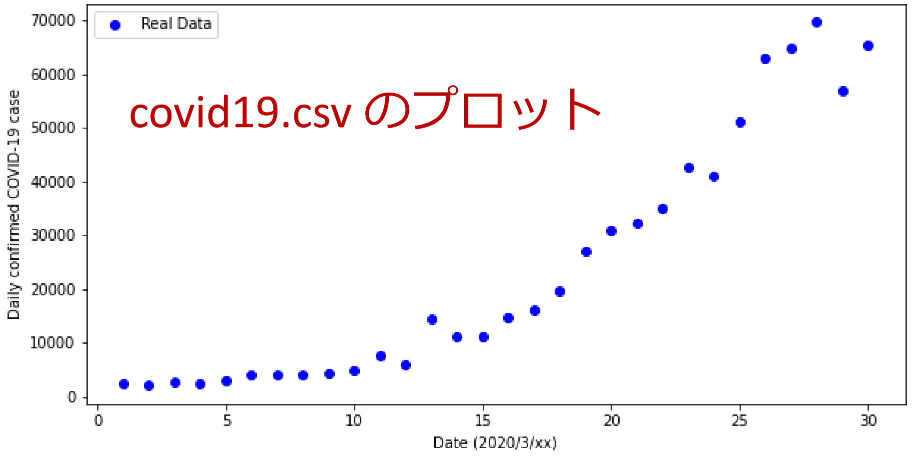
対数 $y = \log(x)$ および指数 $y = e^x$ はそれぞれ numpy の log，exp関数で計算できます。

```
import numpy as np

log_x = np.log(x)
exp_x = np.exp(x)
```

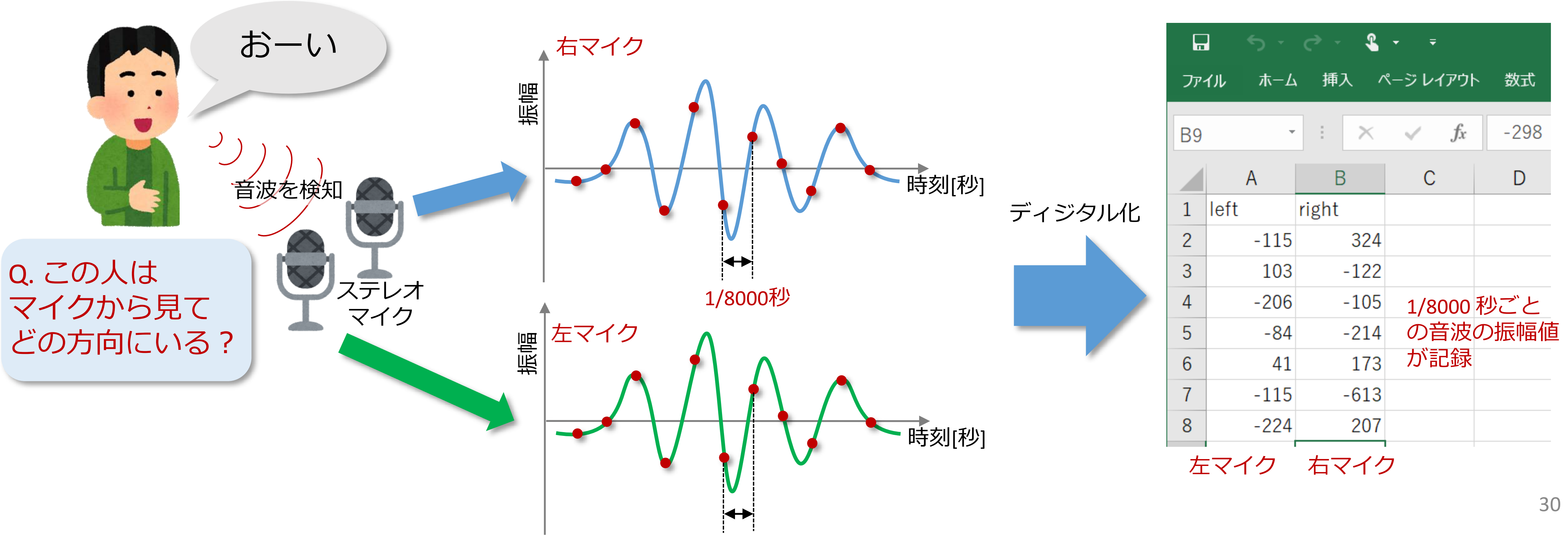
$\log(x)$を計算

e^xを計算



レポート課題２：音源方向推定

report03_input.csv というデータを読み込んでください。
このデータは，左右ステレオのマイクで収録された音声信号です。
左右それぞれのマイクにおける音の振幅値を，1/8000秒間隔で記録しています。
次以降のページを読んで，この音がどの方向からやって来たかを推定してください。

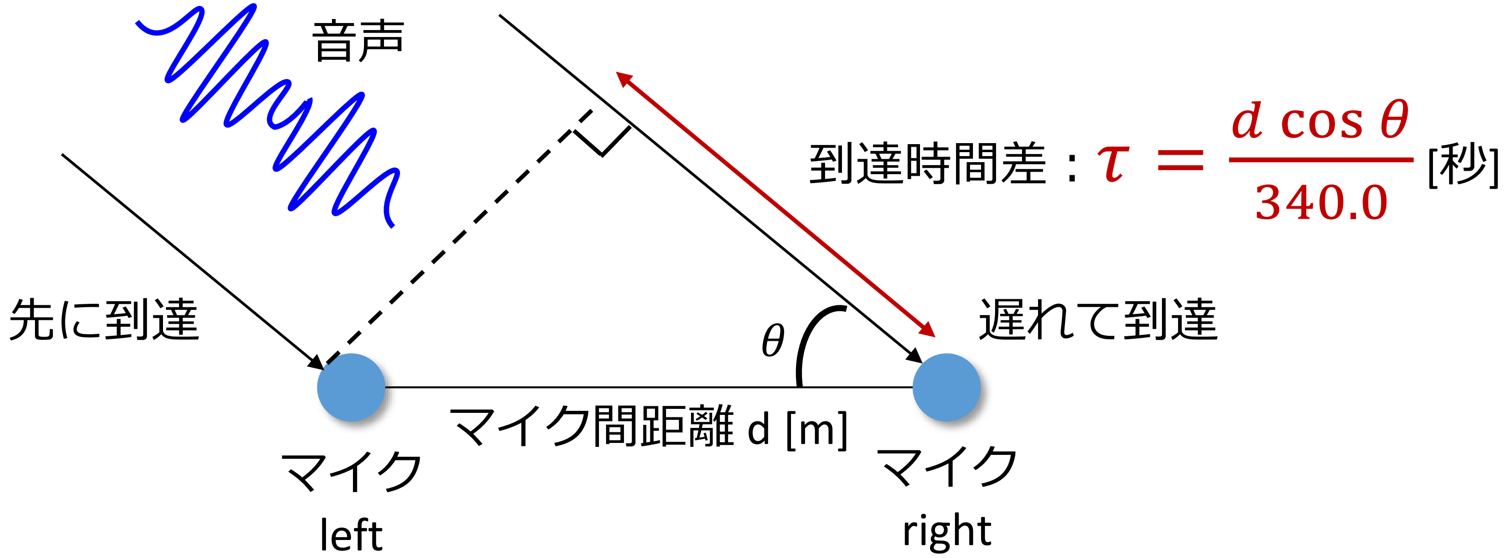


レポート課題２：音源方向推定

- 2個のマイクで音声を収録したとき，音声がマイクに到達する時間差が生じる。
- マイク間距離を d [m]，音源方向を θ，音速を 340.0 [m/秒] としたとき，マイク間の音の到達時間差は $\tau = d\cos\theta\, /\, 340.0$ [秒]である。
- つまり，マイク間の時間差 τ [秒] が分かれば，音源方向は以下の式で得られる。

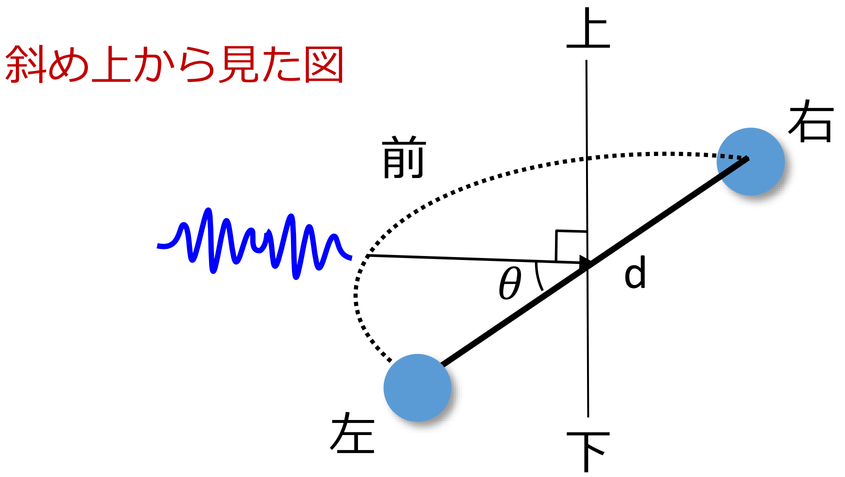
$$\theta = \cos^{-1}\frac{340.0\tau}{d}$$

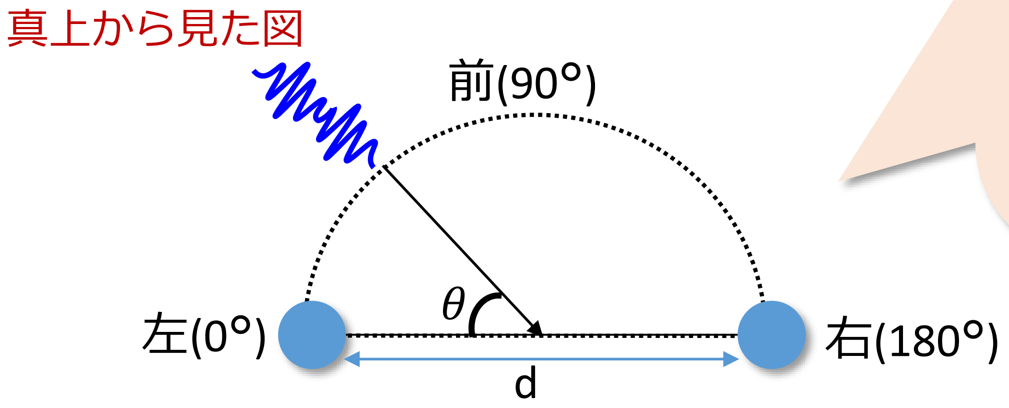
- このデータのマイク間距離は $d = 0.3$ [m] である。
- 音声データの時刻は 1 目盛りにつき 1/8000 秒である。

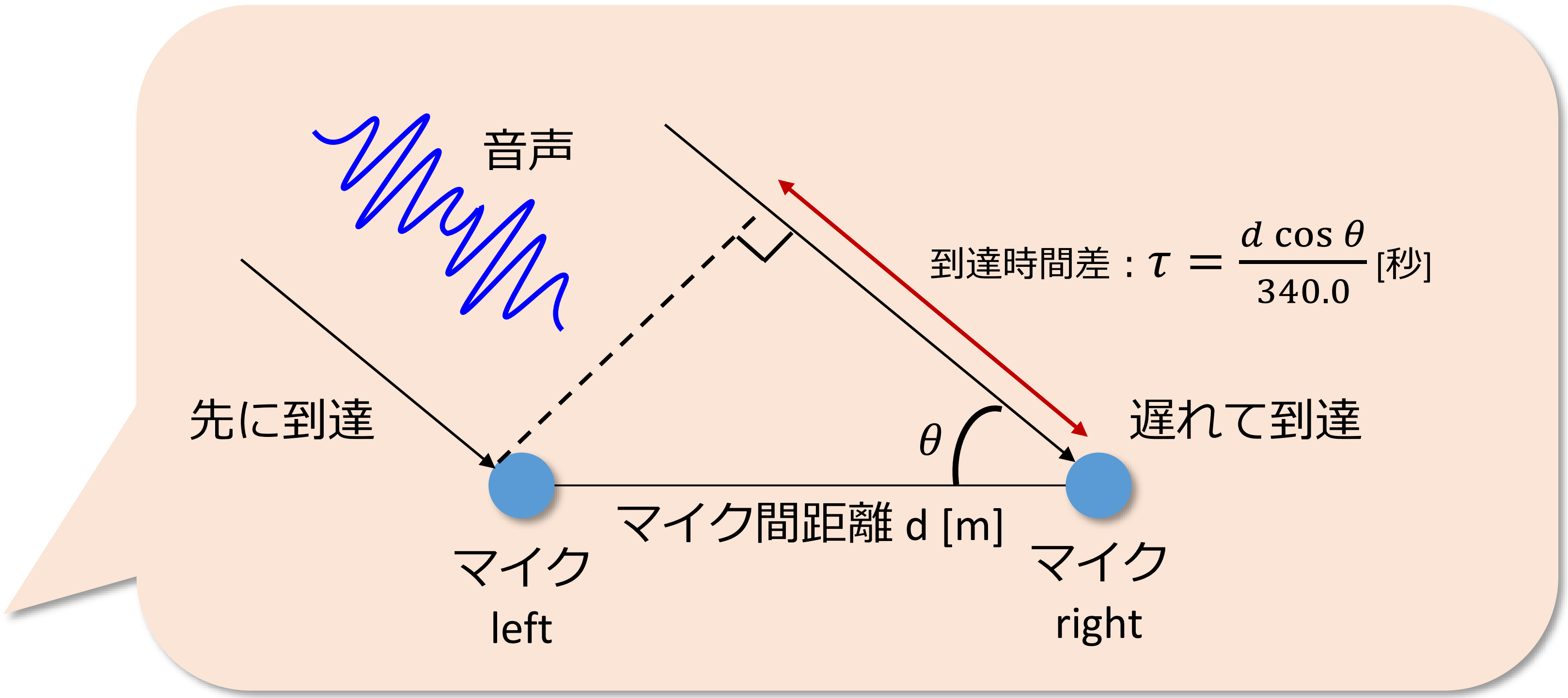


レポート課題２：注意点およびヒント

- 音源方向は下図のように定義する。マイクの真左から到来した場合は 0°，真右から到来した場合は 180°である。
- 推定する角度 θ は「水平角」（前後左右）。高さ方向（仰角）は考慮しなくてよい。（便宜上，マイクと同じ高さから到来しているとする。）







レポート課題２：注意点およびヒント

- 求めた時間差の単位が [秒] に換算されているか注意すること。
  データの時刻の目盛りは 1/8000 秒である。

- $\cos^{-1}(A)$ の計算は，numpyの関数 np.arccos(A) で求められる。
  ただし戻り値の単位は radian である。

- radian から degree への変換は， degree = np.rad2deg(radian) で行える。