機械学習講習会

[2]「勾配降下法」

2024/06/25

traP Kaggle班

まとめ

- アイスの売り上げを予測するには, 気温から売り上げを予測する「関数」を構築するのが必要であった.
- いったん, 今回は関数の形として $f(x) = ax + b$ (一次関数) に限って,関数を決めることにした.
- この関数は, パラメータとして (a, b) をもち, (a, b) を変えることで性質が変わるのがわかった
- モデルの「よさ」のめやすとして, 「損失関数」を導入した
- パラメータを変えることで損失関数を最小化する過程のことを「学習」と呼ぶ

前回到達したところ...

a, b を動かすことで....

$\mathcal{L}(a,b) = \dfrac{1}{n}\displaystyle\sum_{i=0}^{n-1} (y_i - f(x_i;a,b))^2$ を小さくしたい 🥺

「関数の最小化」を考える

問題

最小化してください.

$$f(x) = x^2 + 4x + 6$$

「関数の最小化」を考える

問題

最小化してください.

$$f(x) = x^2 + 4x + 6$$

解答

$$f(x) = x^2 + 4x + 6 = (x+2)^2 + 2$$

$$\therefore\ x = -2 \text{ のとき最小値}$$

どう「解けた」？？

- 簡単な数式の操作で解けた！
- 機械的に書くなら

  「$ax^2 + bx + c$ を最小にする x は $x = -\dfrac{b}{2a}$ 」という公式を使った

プログラムに起こすと…

```python
# ax^2 + bx + c を最小にする x を返す関数.
def solve(a, b, c):
    return -b / (2 * a)
```

第二問

最小化してください.

$$f(x) = x^2 + e^{-x}$$

第二問

$f'(x) = 2x - e^{-x}$ なので, 最小値であることの必要条件 $f'(x) = 0$ を調べると...

$$2x - e^{-x} = 0$$

を満たす x を考えると.......

?


Google
Google 検索
I'm Feeling Lucky


Google
wolfram alpha
wolfram alpha
wolfram alpha 積分
wolfram alpha 微分方程式
wolfram alpha 微分


WOLFRAM言語とMATHEMATICAの開発元による
WolframAlpha
solve 2x - e^{-x} = 0
自然言語
数学入力
拡張キーボード
例を見る
アップロード
ランダムな例を使う

WOLFRAM言語とMATHEMATICAの開発元による

WolframAlpha

solve 2x - e^{-x} = 0

自然言語 | 数学入力 | 拡張キーボード | 例を見る | アップロード | ランダムな例を使う

入力解釈

$2x - e^{(-x)} = 0$ を解く

結果

$$x = W_n\left(\frac{1}{2}\right), \quad n \in \mathbb{Z}$$

$W_k(z)$ は乗積対数関数の解析接続です

$\mathbb{Z}$ は整数の集合です

実数解

表示桁数を増やす

ランベルトのW関数

文A 20の言語版

ページ　ノート　　閲覧　編集　履歴表示　ツール

出典: フリー百科事典『ウィキペディア（Wikipedia）』

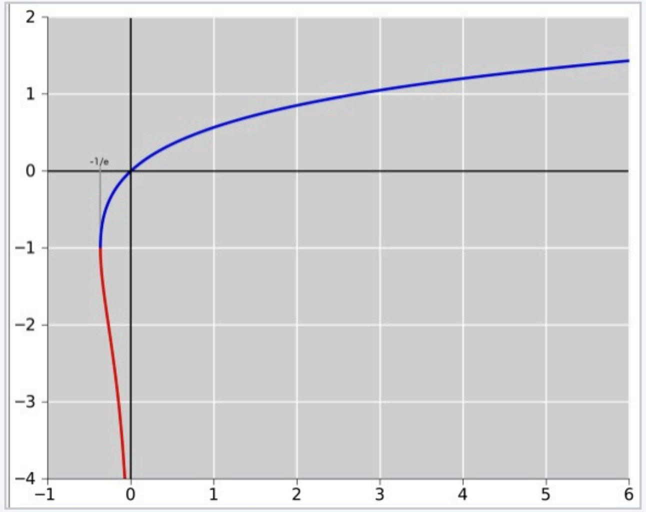
ランベルトのW関数（ランベルトのWかんすう、英: *Lambert W function*）あるいは**オメガ関数** (*ω function*)、対数積（*product logarithm*; 乗積対数）は、函数 $f(z) = ze^z$ の逆関係の分枝として得られる函数 W の総称である。ここで、e^z は指数函数、z は任意の複素数とする。すなわち、W は $z = f^{-1}(ze^z) = W(ze^z)$ を満たす。

上記の方程式で、$z' = ze^z$ と置きかえれば、任意の複素数 z' に対する W 函数（一般には W 関係）の定義方程式

$$z' = W(z')e^{W(z')}$$

を得る。

函数 f は単射ではないから、関係 W は（0 を除いて）多価である。仮に実数値の W に注意を制限するとすれば、複素変数 z は実変数 x に取り換えられ、関係の定義域は区間 $x \geq -1/e$ に限られ、また開区間 $(-1/e, 0)$ 上で二価の函数になる。さらに制約条件として $W \geq -1$ を追加すれば一価函数 $W_0(x)$ が定義されて、$W_0(0) = 0$ および $W_0(-1/e) = -1$ を得る。それと同時に、下側の枝は $W \leq -1$ であって、$W_{-1}(x)$ と書かれる。これは $W_{-1}(-1/e) = -1$ から $W_{-1}(-0) = -\infty$ まで単調減少する。



$W(x)$ のグラフの $W > -4$ および $x < 6$ の部分。$W \geq -1$ なる上の枝を主枝 W_0 と言い、$W \leq -1$ なる下側の分枝を W_{-1} という。

ランベルト W 関係は初等函数では表すことができない[1]。ランベルト W は組合せ論において有用で、例えば木の数え上げに用いられる。指数函数を含む様々な方程式（例えばプランク分布、ボーズ–アインシュタイン分布、フェルミ–ディラック分布などの最大値）を解くのに用いられ、また$y'(t) = ay(t-1)$ のような遅延微分方程式（英語版） の解としても生じる。生化学において、また特に酵素動力学において、ミカエリ

?

一般の関数の最小化

いいたかったこと

✅ **このレベルの単純な形の関数でも, 解をよく知っている形で書き表すことは難しい**

もう一度目的を整理する

われわれの目標…

誤差 $\mathcal{L}(a, b)$ を最小化したかった.

効いてくる条件①

Q. 厳密な最小値を得る必要があるか？

効いてくる条件①

A. No. 厳密に最小値を得る必要はない

数学の答案で最小値 1 になるところを 1.001と答えたら当然 🙅

一方，「誤差 1」が「誤差1.001」になってもほとんど変わらない

効いてくる条件②

$\mathcal{L}$ は非常に複雑になりうる

第一回では **話を簡単にするために** $f(x) = ax + b$ の形を考えたが...

(特にニューラルネットワーク以降は) **非常に複雑になりうる**

$$\mathcal{L}(\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \cdots, \mathbf{W}^{(\mathbf{n})}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \cdots, \mathbf{b}^{(\mathbf{n})}) = \frac{1}{n}\sum_{i=0}^{n-1}\left(y_i - {W^{(n)}}^T\sigma\left(\cdots\sigma\left({W^{(1)}}^T x_i + b^{(1)}\right)\cdots + b^{(n-1)}\right)\right)^2, \ \sigma(x) = \frac{1}{1+e^{-x}}$$

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{\sum_{p\in S}\left|f(p;\boldsymbol{\theta}) - \mathcal{F}(p)\right|^2 \cdot \omega_p}{\sum_{p\in S}\omega_p}$$

$$\vdots$$

複雑そうな式を気分で乗せただけなのであまり意図はありません

われわれに必要な道具

✅　非常に広い範囲の関数に対して

そこそこ小さい値を探せる方法

われわれに必要な道具

勾配降下法

微分のおさらい

微分係数

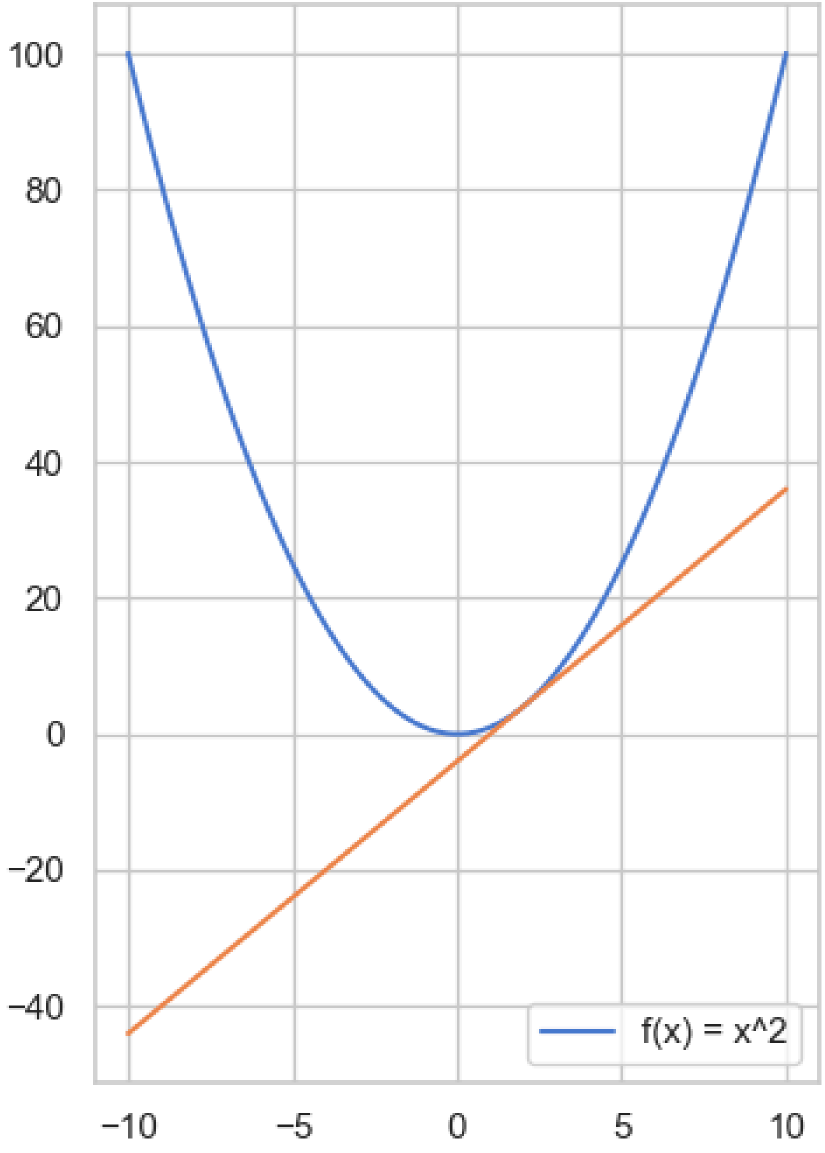
関数 f の x における微分係数

$$\lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

微分は「傾き」

微分係数

$f'(x)$ は, x における接線の傾き



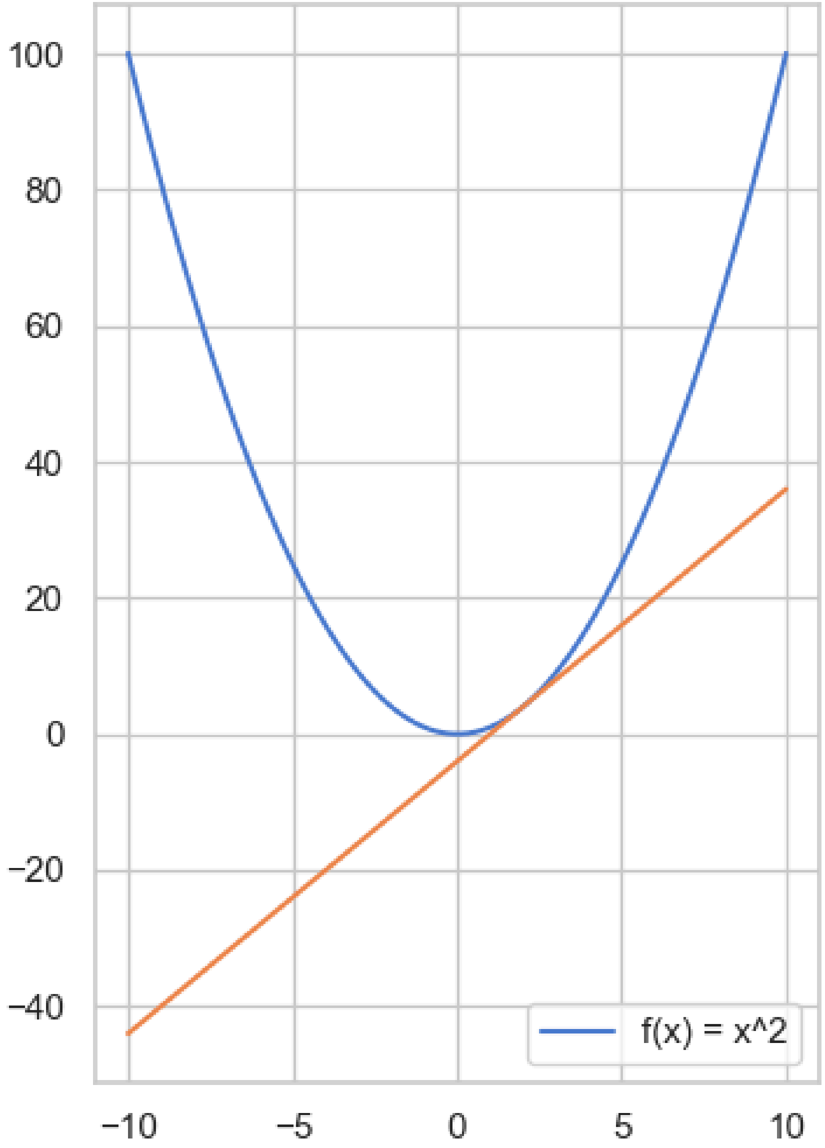
微分は「傾き」

微分係数

$f'(x)$ は, x における接線の傾き

⬇

$-f'(x)$ 方向に関数を
すこし動かすと,関数の値は
すこし小さくなる



「傾き」で値を更新してみる

例) $f(x) = x^2$

$x = 3$ で $f(3) = 9,\ f'(3) = 6$

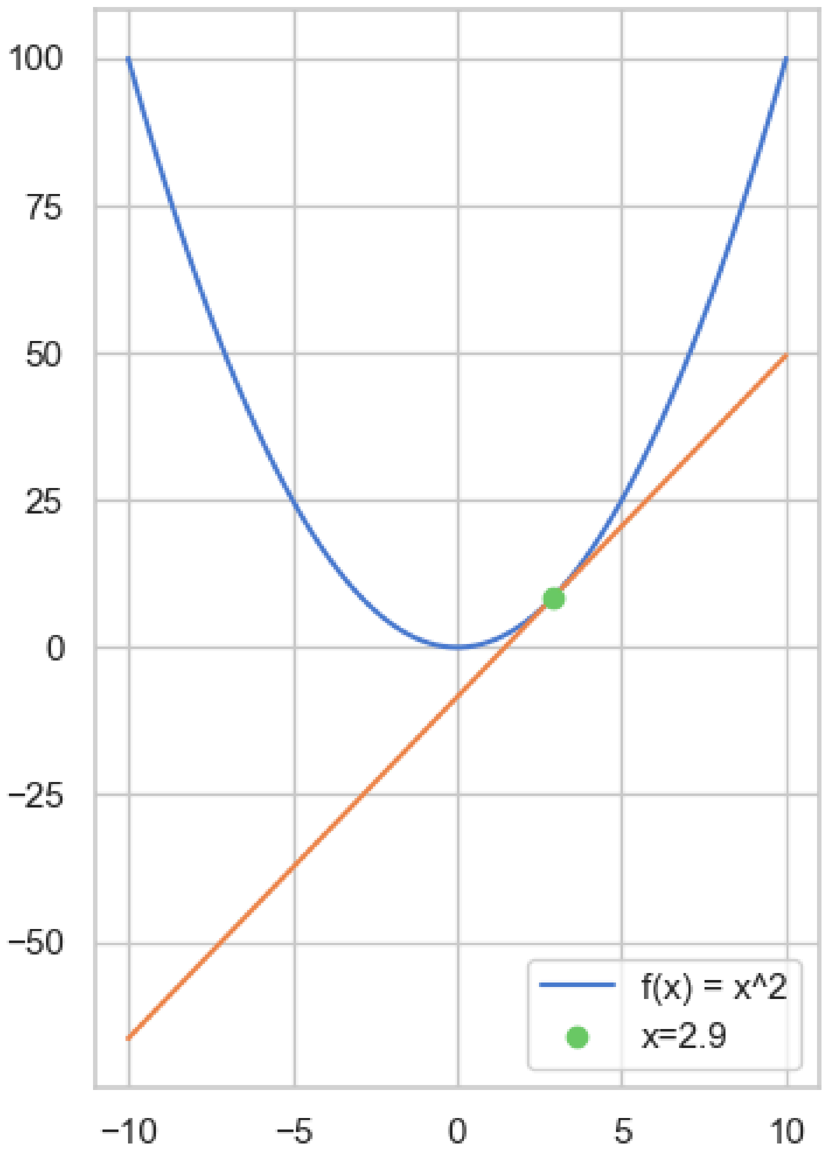
$\therefore -f'(x)$ は負の方向

⬇

すこし負の方向に x を動かしてみる

$f(2.9) = 8.41 < 9$

✅ 小さくなった



「傾き」で値を更新してみる

例) $f(x) = x^2$

$x = 2.9$ で

$f(2.9) = 8.41,\ f'(2.9) = 5.8$

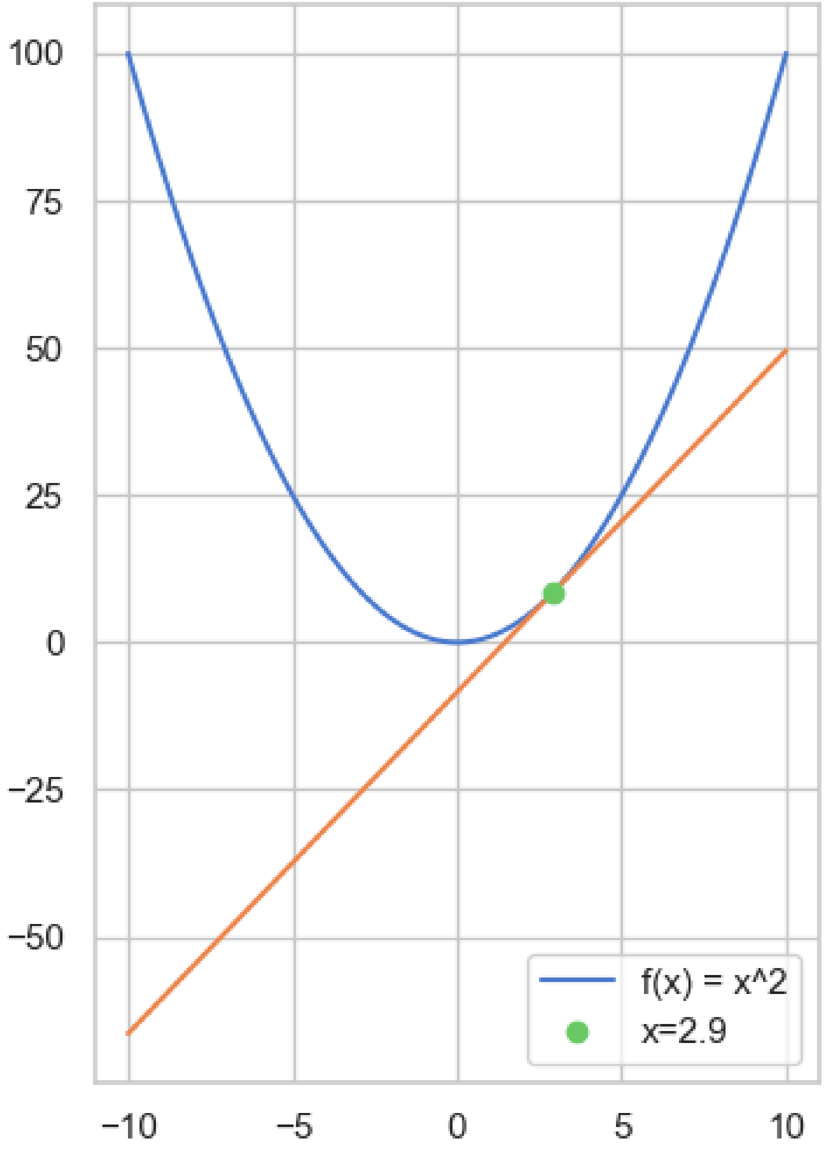
$\therefore -f'(x)$ は負の方向

⬇

すこし負の方向に x を動かしてみる

$f(2.8) = 7.84 < 8.41$

✅ 小さくなった



「傾き」で値を更新してみる

これを繰り返すことで小さい値まで到達できそう！

勾配降下法

勾配降下法

関数 $f(x)$ と,初期値 x_0 が与えられたとき,
次の式で $\{x_k\}$ を更新するアルゴリズム

$$x_{k+1} = x_k - \eta f'(x_k)$$

(η は**学習率**と呼ばれる定数)

正確にはこれは最急降下法と呼ばれるアルゴリズムで,「勾配降下法」は勾配を使った最適化手法の総称として用いられることが多いと思います.
(そこまで目くじらを立てる人はいないと思いますし, 勾配降下法あるいは勾配法と言われたらたいていの人がこれを思い浮かべると思います.)

勾配降下法

マイナーチェンジが大量！
(実際に使われるやつは第五回で予定)

$$x_{n+1} = x_n - \eta f'(x_n)$$

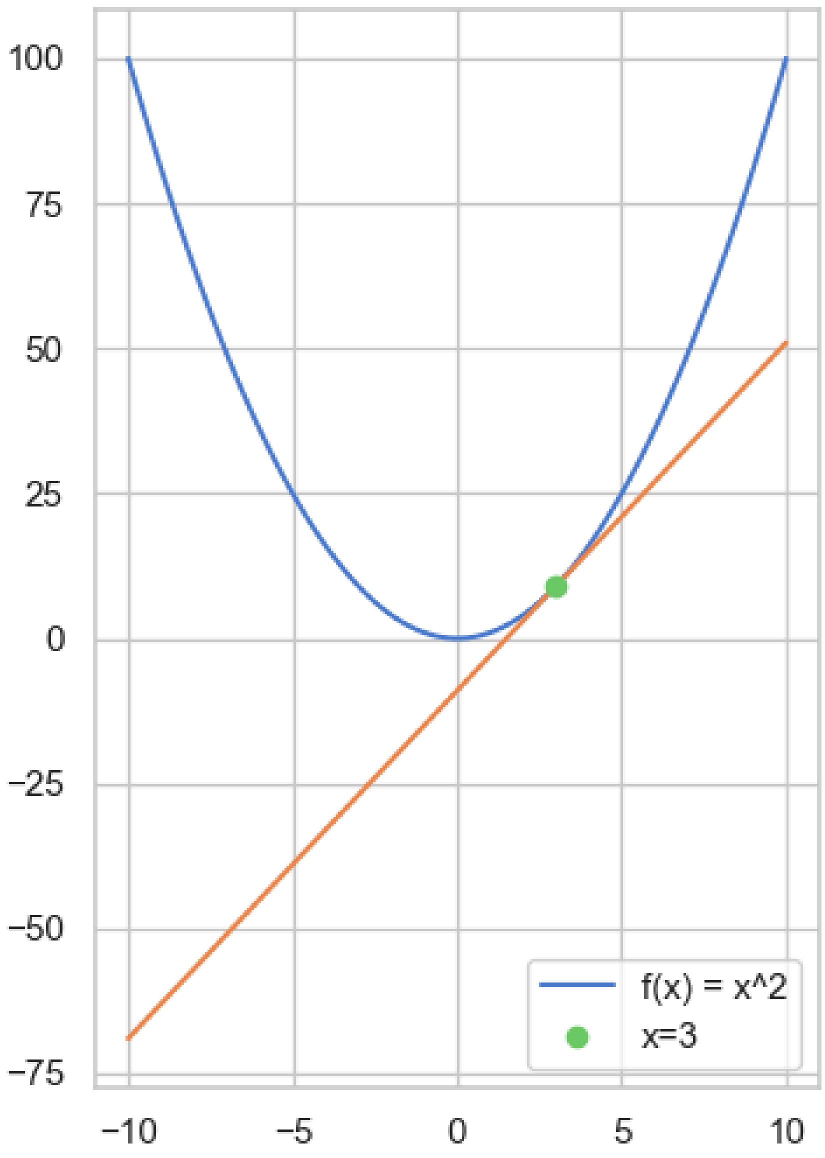
抑えてほしいこと 👀

1. 値が $-f'(x)$ の方向に更新される
2. 学習率によって更新幅を制御する

勾配降下法のお気持ち

値が $-f'(x)$ の方向に更新される

(さっきの説明の通り)



学習率による更新幅の制御

✅ 微分はあくまで「**その点**の情報」

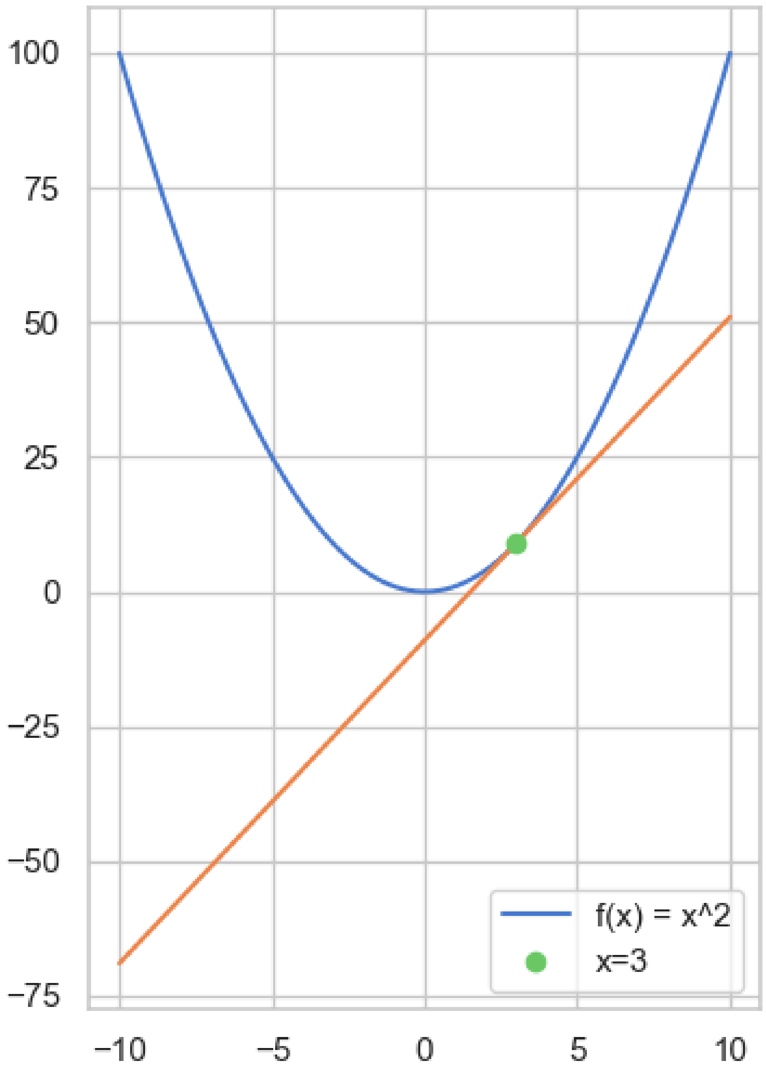
傾向が成り立つのはその周辺だけ

⬇

少しずつ更新していく必要がある

⬇

小さな値 **学習率** η をかけることで
少しずつ更新する



実際にやってみる

$$f(x) = x^2$$

初期値として,$x_0 = 3$
学習率として,$\eta = 0.1$ を設定.(この二つは自分で決める！)

$$x_1 = x_0 - \eta f'(x_0) = 3 - 0.1 \times 6 = 2.4$$
$$x_2 = x_1 - \eta f'(x_1) = 2.4 - 0.1 \times 4.8 = 1.92$$
$$x_3 = x_2 - \eta f'(x_2) = 1.92 - 0.1 \times 3.84 = 1.536$$
$$\cdots$$
$$x_{100} = 0.0000000006111107929$$

✅ **最小値を与える $x = 0$ に非常に近い値が得られた！**

勾配降下法のココがすごい！

✅ その式を (解析的に) 解いた結果が何であるか知らなくても，
導関数さえ求められれば解を探しにいける

実際にやってみる2

第二問

最小化してください.

$$f(x) = x^2 + e^{-x}$$

実際にやってみる2

$$f'(x) = 2x - e^{-x}.$$

初期値として $x = 3$, 学習率として $\eta = 0.01$ を設定.

$$x_0 = 3$$
$$x_1 = 2.940497870683678 6$$
$$\vdots$$
$$x_{1000} = 0.35173371125366865$$

ヨシ！😺

Pythonによる実装

```python
from math import exp

x = 3
# (注意: $\eta$ は, 学習率 (learning rate) の略である lr としています.)
lr = 0.0005

# 最小化したい関数
def f(x):
  return x ** 2 + exp(-x)

# f の x での微分係数
def grad(x):
    return 2 * x - exp(-x)
```

Pythonによる実装

$x_{n+1} = x_n - \eta f'(x_n)$ をコードに起こす

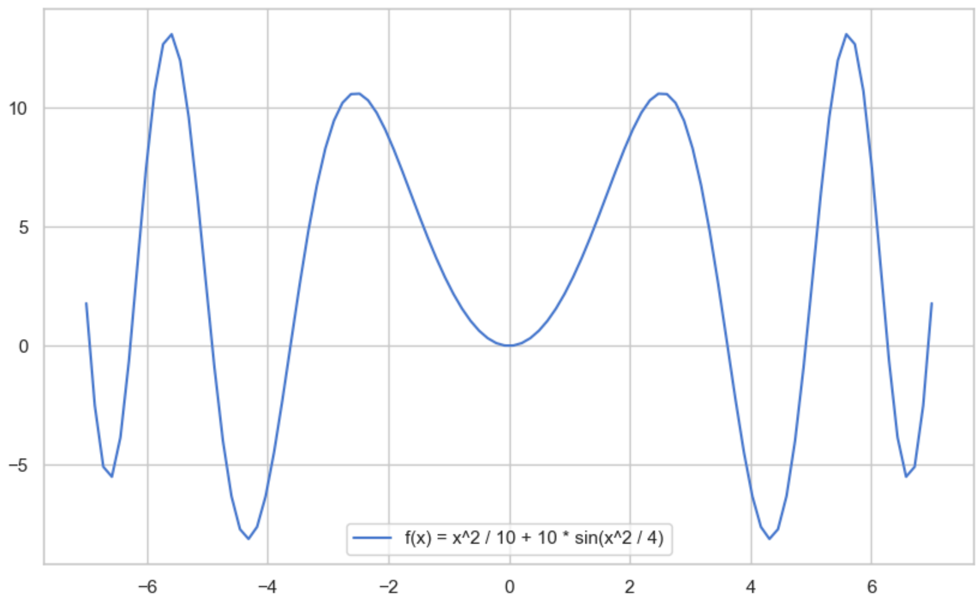
```
for i in range(10001):
    # 更新式
    x = x - lr * grad(x)
    if i % 1000 == 0:
        print('x_', i, '=', x , ', f(x) =', f(x))
```

```
x_ 0 = 2.997024893534184 , f(x) = 9.032093623218246
x_ 1000 = 1.1617489280037716 , f(x) = 1.6625989669983947
x_ 2000 = 0.5760466279295902 , f(x) = 0.8939459518186053
x_ 3000 = 0.4109554481889124 , f(x) = 0.8319008499233866
...
x_ 9000 = 0.3517515401706734 , f(x) = 0.8271840265571999
x_ 10000 = 0.3517383210080008 , f(x) = 0.8271840261562484
```

常に上手くいく？

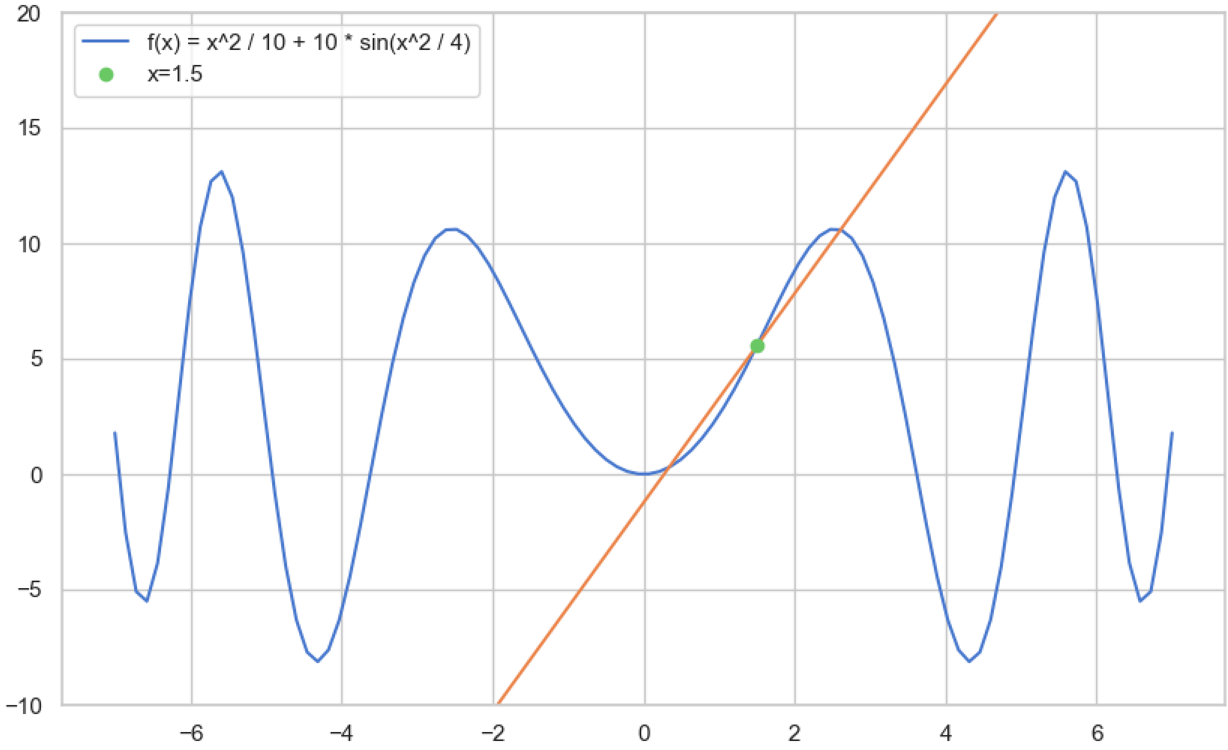
✅ 勾配降下法があまりうまくいかない関数もある

例) $f(x) = \dfrac{x^2}{10} + 10\sin\left(\dfrac{x^2}{4}\right)$



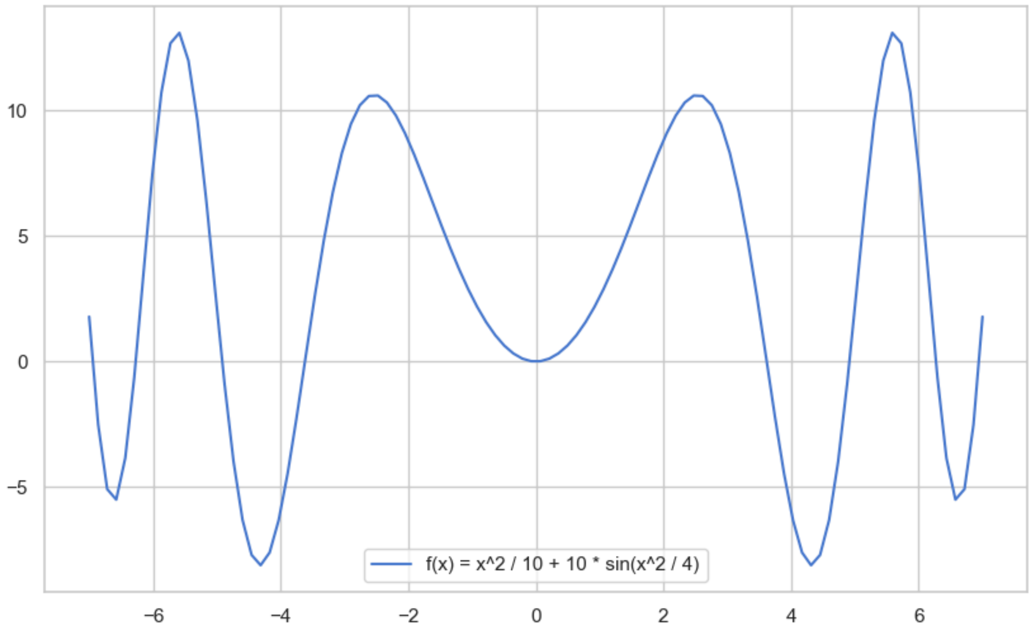
うまくいかない例

$x = 1.5$ あたりから勾配降下法をすると、$x = 0$ に収束する！



局所最適解への収束

局所最適解 ... 付近では最小値 ($x = -6, -4, 0, 4, 6$ あたりのもの全て)
大域最適解 ... 全体で最小値 ($x = -4, 4$ あたりのもの)



マイナーチェンジ

⇨ なるべく局所最適解に **ハマりまくらない** ように色々と工夫 (詳しくは第5回)

- Momentum

$$
\begin{aligned}
v_{n+1} &= \alpha v_n - \eta f'(x_n) \\
x_{n+1} &= x_n + v_{n+1}
\end{aligned}
$$

- AdaGrad

$$
\begin{aligned}
h_{n+1} &= h_n + f'(x_n)^2 \\
x_{n+1} &= x_n - \frac{\eta}{\sqrt{h_{n+1}}} f'(x_n)
\end{aligned}
$$

$$\vdots$$

多変数関数への応用

多変数関数の場合は,微分係数→勾配ベクトル　に置き換えればOK

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \eta \nabla f(\boldsymbol{x}_n)$$

勾配ベクトルとは,各変数の偏微分係数を並べたものです. 例えば,$f(x,y) = x^2 + y^2$ の (x,y) における勾配ベクトルは $(2x, 2y)$ です.
これを$\nabla f(x,y) = (2x, 2y)$ とかきます. 一年生はちょうど微分積分学第一でやるころかと思うので大きくは扱いませんでしたが, 一変数の場合できちんと理解できていれば大丈夫です.

再掲: 一般の関数の最小化

第三問

最小化してください.

$$-\frac{1}{(x^2+1)}\log\left(\frac{1}{1+e^{-x}}+1\right)$$

嫌です.

次回予告

第三回　自動微分