


パラメータの最適化

ぱうえる (けんた) 

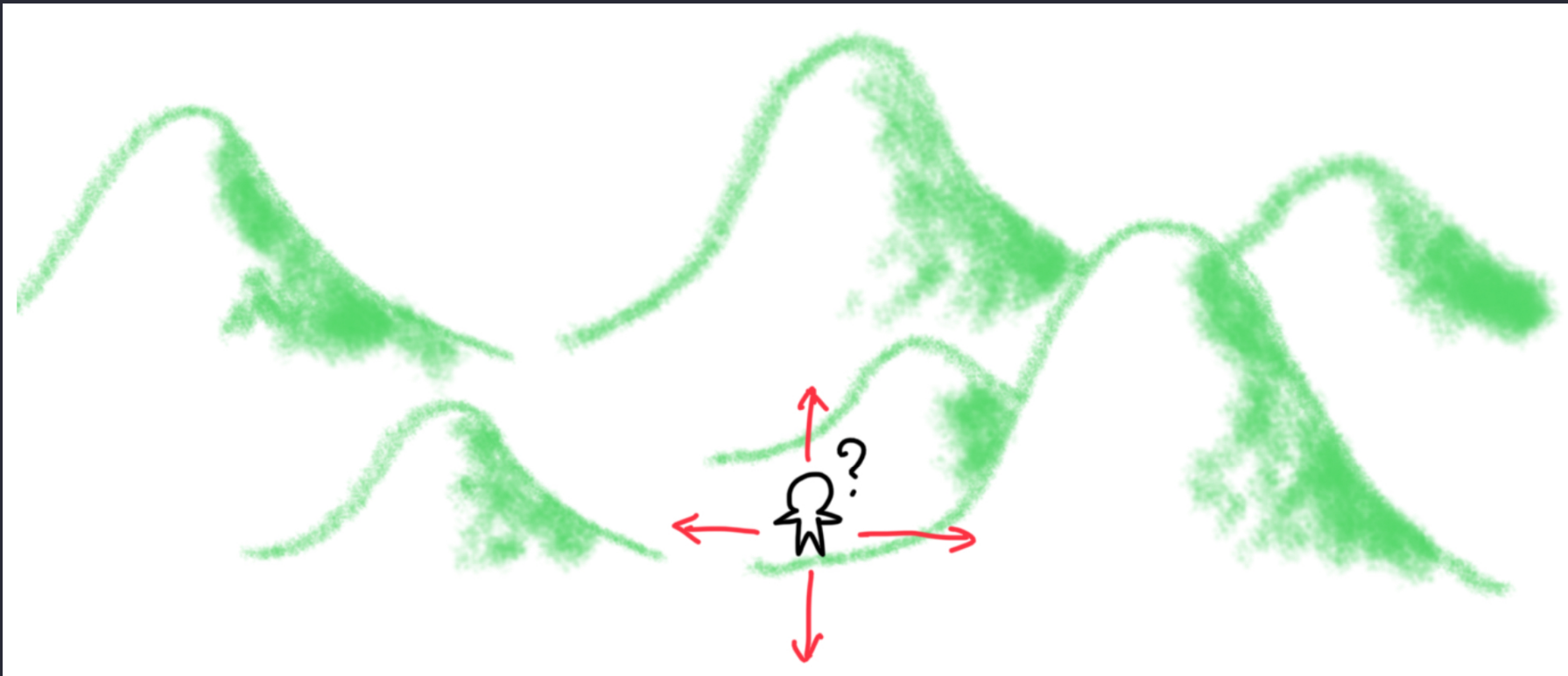
パラメータとは

- 結果に影響を与える変数のこと
- ニューラルネットワークには、膨大な数のパラメータが存在する
 - 各層の重み
 - 各層のバイアス

これらをどのように修正していけば、損失関数の値を最小にできる？

パラメータの最適化

- 頂上に辿り着くため、あなたはどこに向かうべき？



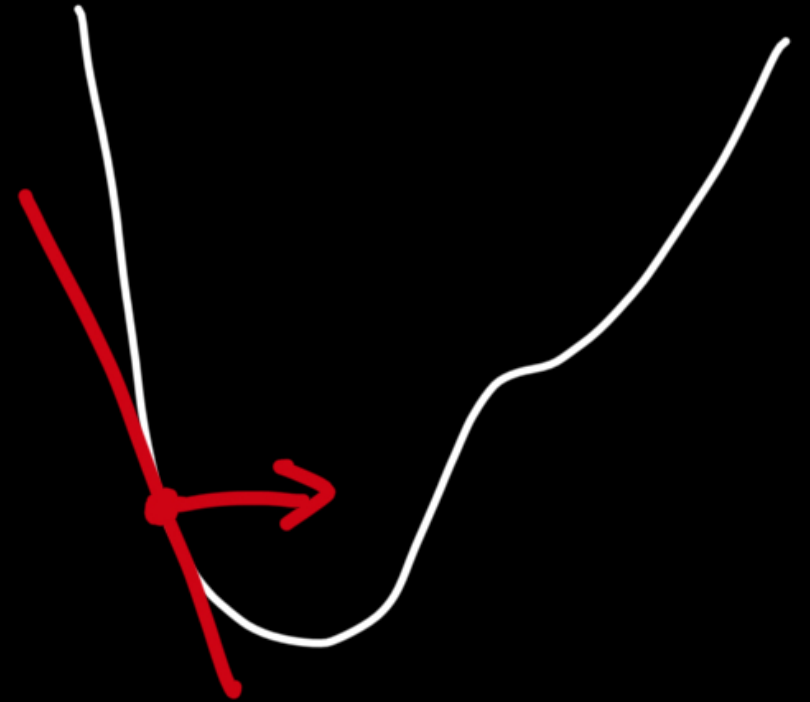
今までの方法(SGD)

確率的勾配降下法

(Stochastic Gradient Decent)

- パラメータをもとに勾配を求め、
勾配が小さくなるように修正

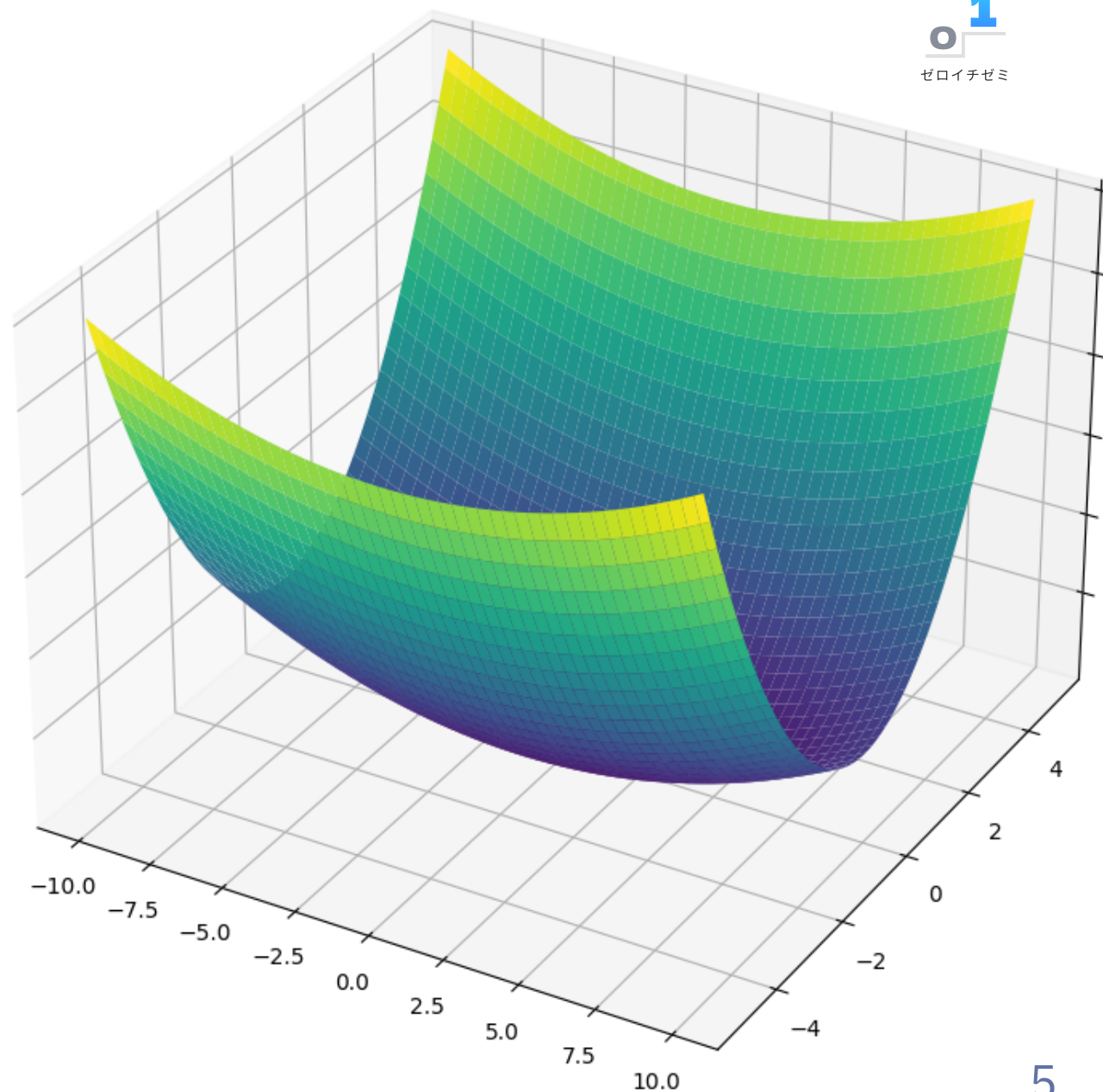
$$W \leftarrow W - \eta \frac{\partial L}{\partial W}$$



SGDの弱点

こんな感じのフィールドを考えてみよう

$$z = \frac{1}{20}x^2 + y^2$$

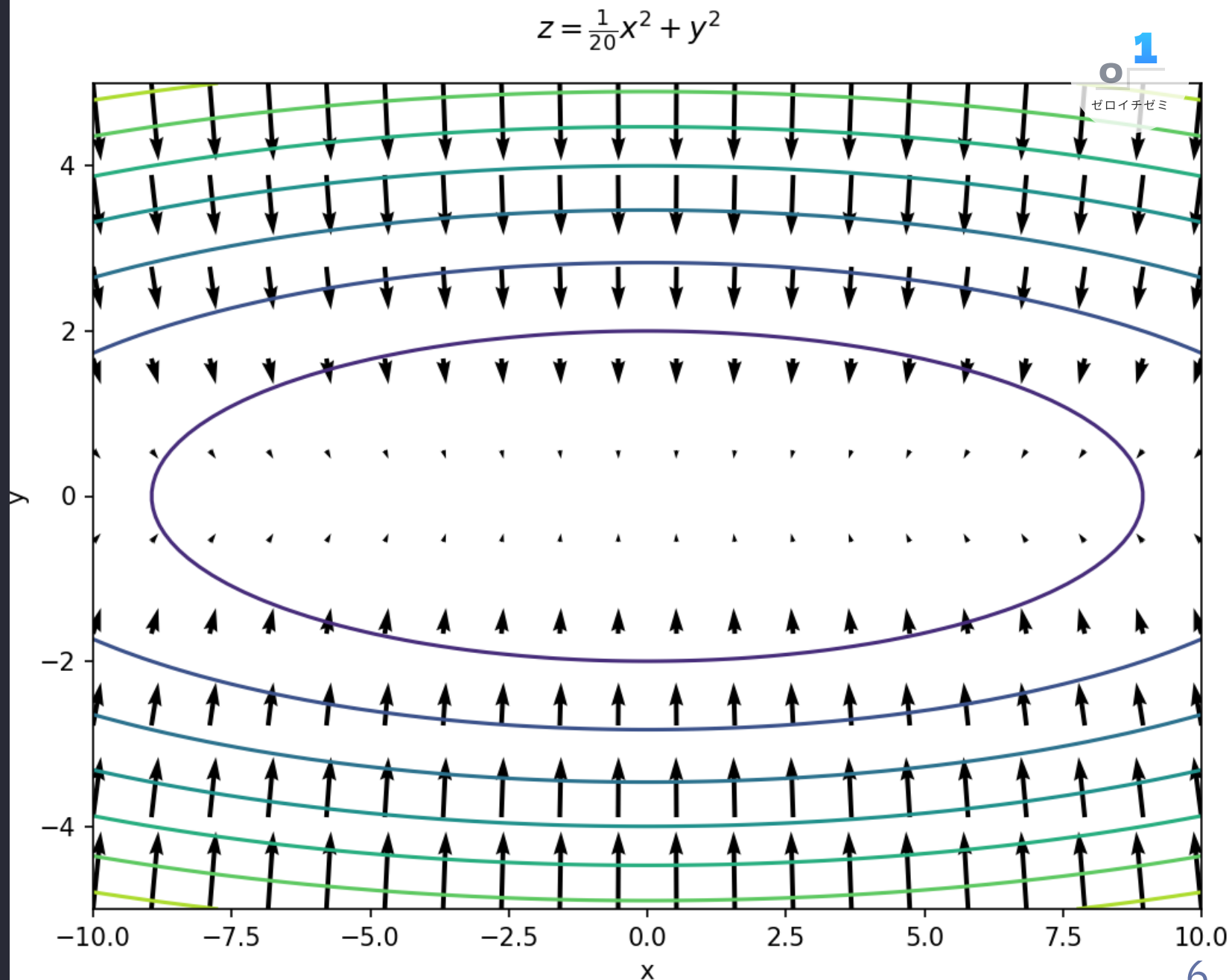


SGDの弱点

勾配はこんな感じ

- x軸の勾配→小
- y軸の勾配→大

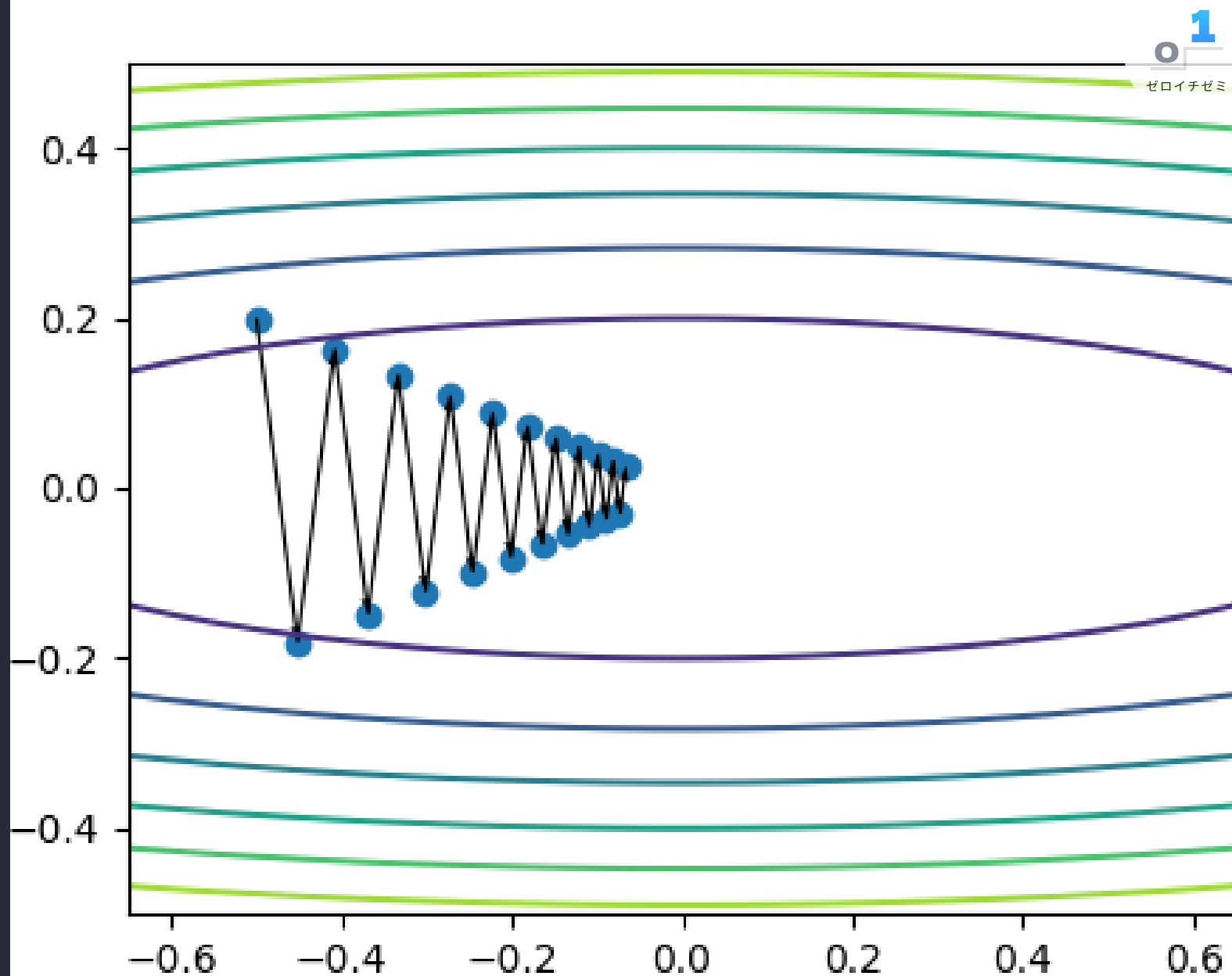
→どんな動き？



SGDの弱点

こんな動き！

y軸の勾配が大きい
いため、無駄な動
きをしている



試してみよう！

`optimization.ipynb` を使ってビジュアライズします👏

- **visualizer.Field**
 - 勾配を持ったフィールドです
 - フィールドのプロットもできます
- **visualizer.Adventurer**
 - 実際に動く人
 - たどったルートを記録します

ビジュアライザの使い方

```
from visualizer import Field, Adventurer

def f(x, y):
    return x^2 + y^2

field = Field(f, (-1, 1), (-1, 1)) # フィールド
adventurer = Adventurer(-0.5, 0.2) # 冒険者（スタート地点を設定）

adventurer.optimize(field, method) # adventurerに冒険をさせます（methodは最適化手法:SGDなど）
route = adventurer.route           # ルートが記録される

field.plot(route=route)            # どんなルートを辿ったか見られます
```

SGDの実装

```
class SGD:
    def __init__(self, lr=0.1):
        self.lr = lr

    def update(self, params, grads):
        # ここを考えよう！
```

↓ヒント

```
params = {
    "x": -0.5, # x座標
    "y": 0.2, # y座標
}
```

新たな手法 "Momentum"

- "運動量"を表す言葉
- その名の通り、坂を転がり落ちていくような動き

計算式

$$\begin{aligned} v &\leftarrow \alpha v - \eta \frac{\partial L}{\partial W} && \dots \text{ (速度の更新)} \\ W &\leftarrow W + v && \dots \text{ (パラメータの更新)} \end{aligned}$$

Momentum の動き

グネグネ動く
↓
無駄な動きが
発生しにくい！

