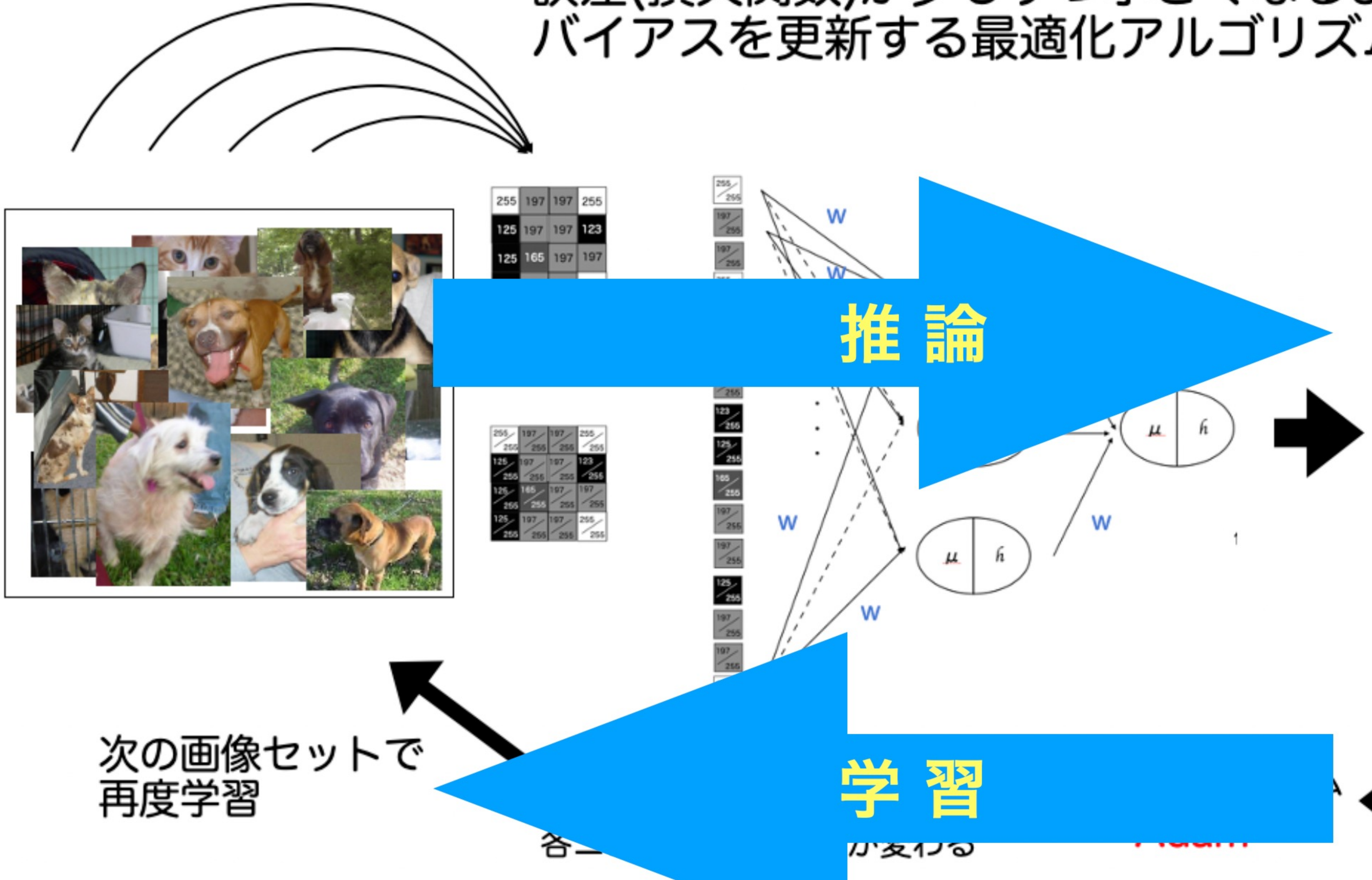
医療とAI・ビッグデータ応用
学習の仕組み

本スライドは、自由にお使いください。
使用した場合は、このQRコードからアンケートに回答をお願いします。

統合教育機構
須藤毅顕

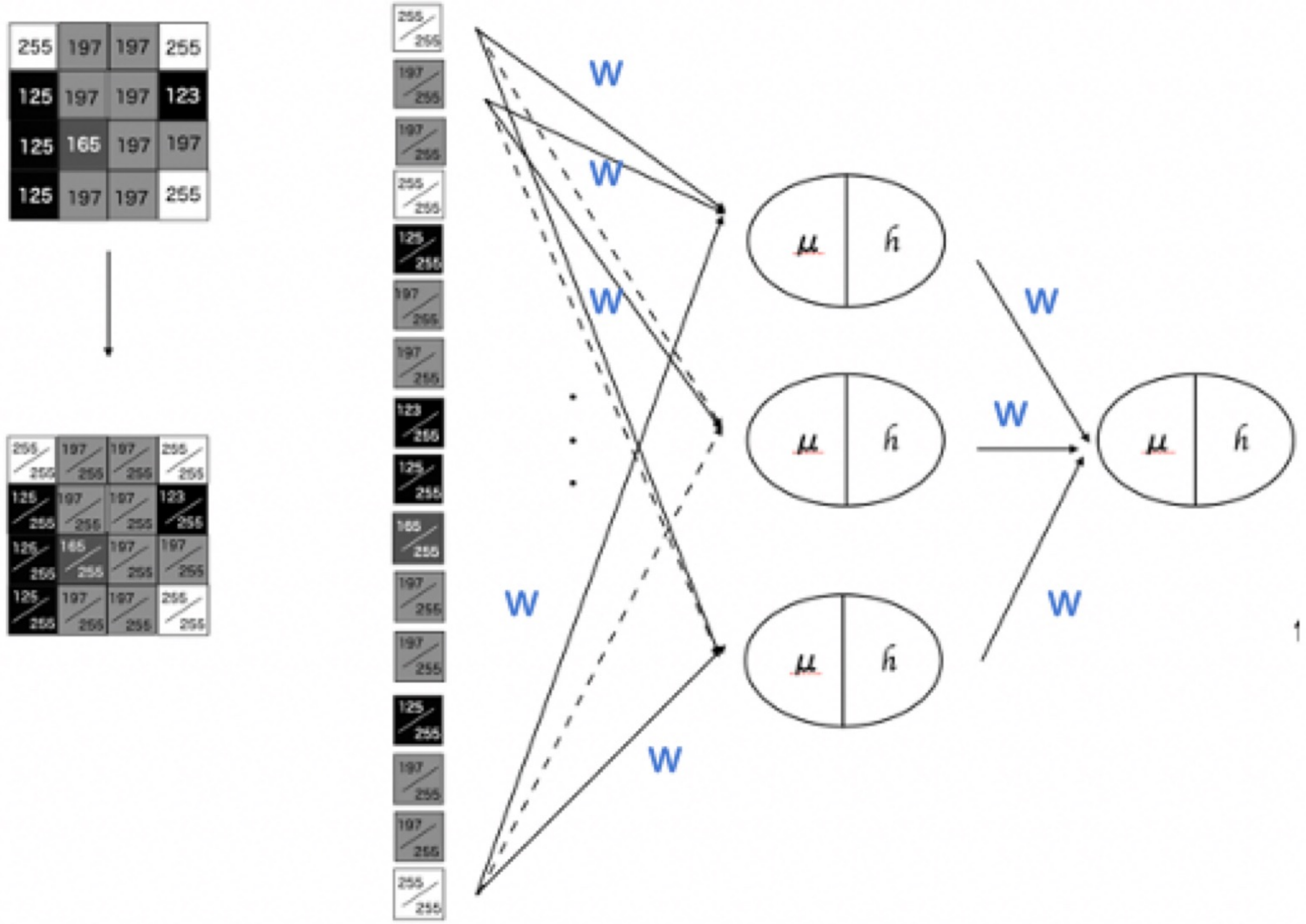
誤差(損失関数)が少しずつ小さくなるように重みと
バイアスを更新する最適化アルゴリズムを設定する



No.	出力	正解
1	0.5	1
2	0.3	1
3	0.6	1
4	0.7	1
5	0.5	0
6	0.3	0
7	0.7	0
8	0.5	0

誤差L = 0.8

この誤差（正解と出力の差）はどのように計算する？



No.	出力	正解
1	0.5	1
2	0.3	1
3	0.6	1
4	0.7	1
5	0.5	0
6	0.3	0
7	0.7	0
8	0.5	0

誤差L = 0.8

最適化アルゴリズム
Adam

重みとバイアスを更新
各ニューロンのwとbが変わる

次の画像セットで
再度学習

誤差について

①完璧

出力	正解	正解 – 出力
1	1	0
1	1	0
0	0	0
0	0	0

②誤差小

出力	正解	正解 – 出力
0.7	1	0.3
0.8	1	0.2
0.2	0	0.2
0.1	0	0.1

③誤差大

出力	正解	正解 – 出力
0.5	1	0.5
0.3	1	0.7
0.6	0	0.4
0.7	0	0.7

この誤差をどのように計算するか？

（正解 – 出力）の和？

①完璧

出力	正解	正解 – 出力
1	1	0
1	1	0
0	0	0
0	0	0

②誤差小

出力	正解	正解 – 出力
0.7	1	0.3
0.8	1	0.2
0.2	0	-0.2
0.1	0	-0.1

③誤差大

出力	正解	正解 – 出力
0.4	1	0.6
0.3	1	0.7
0.6	0	-0.4
0.7	0	-0.7

この4回の誤差を計算する場合、（正解 – 出力）の和にすると

①： $0 + 0 + 0 + 0 = 0$
②：$0.3 + 0.2 + (-0.2) + (-0.1) = 0.2$
③：$0.6 + 0.7 + (-0.4) + (-0.7) = 0.2$

となり、②と③の誤差が等しくなってしまう（ので不適当）

残差平方和

①完璧

出力	正解	正解 – 出力
1	1	0
1	1	0
0	0	0
0	0	0

②誤差小

出力	正解	正解 – 出力
0.7	1	0.3
0.8	1	0.2
0.2	0	-0.2
0.1	0	-0.1

③誤差大

出力	正解	正解 – 出力
0.4	1	0.6
0.3	1	0.7
0.6	0	-0.4
0.7	0	-0.7

(正解 – 出力)^{2}の和にすると

①： $0^2 + 0^2 + 0^2 + 0^2 = 0$
②：$0.3^2 + 0.2^2 + (-0.2)^2 + (-0.1)^2 = 0.09 + 0.04 + 0.04 + 0.01 = 0.18$
③：$0.6^2 + 0.7^2 + (-0.4)^2 + (-0.7)^2 = 0.36 + 0.49 + 0.16 + 0.49 = 1.50$

となり、正しく誤差を① < ② < ③と評価できます。

残差平方和

これを一般化した式で表すと誤差E（＝残差平方和）は

正解t	予測y	予測y – 正解t
t_1	y_1	y_1-t_1
t_2	y_2	y_2-t_2
・・・	・・・	・・・
t_n	y_n	y_n-t_n

$$L = \sum_{i=1}^{n} (y_i - t_i)^2$$

と表すことが出来る

色んな損失関数

残差平方和

$$L = \sum_{i=1}^{n} (y_i - t_i)^2$$

二乗和誤差

$$L = \frac{1}{N} \sum_{i=1}^{n} (y_i - t_i)^2$$

交差エントロピー誤差

$$L = -\sum_{i=1}^{n} t_k \log(y_k)$$

回帰問題で使用

yは具体的な値

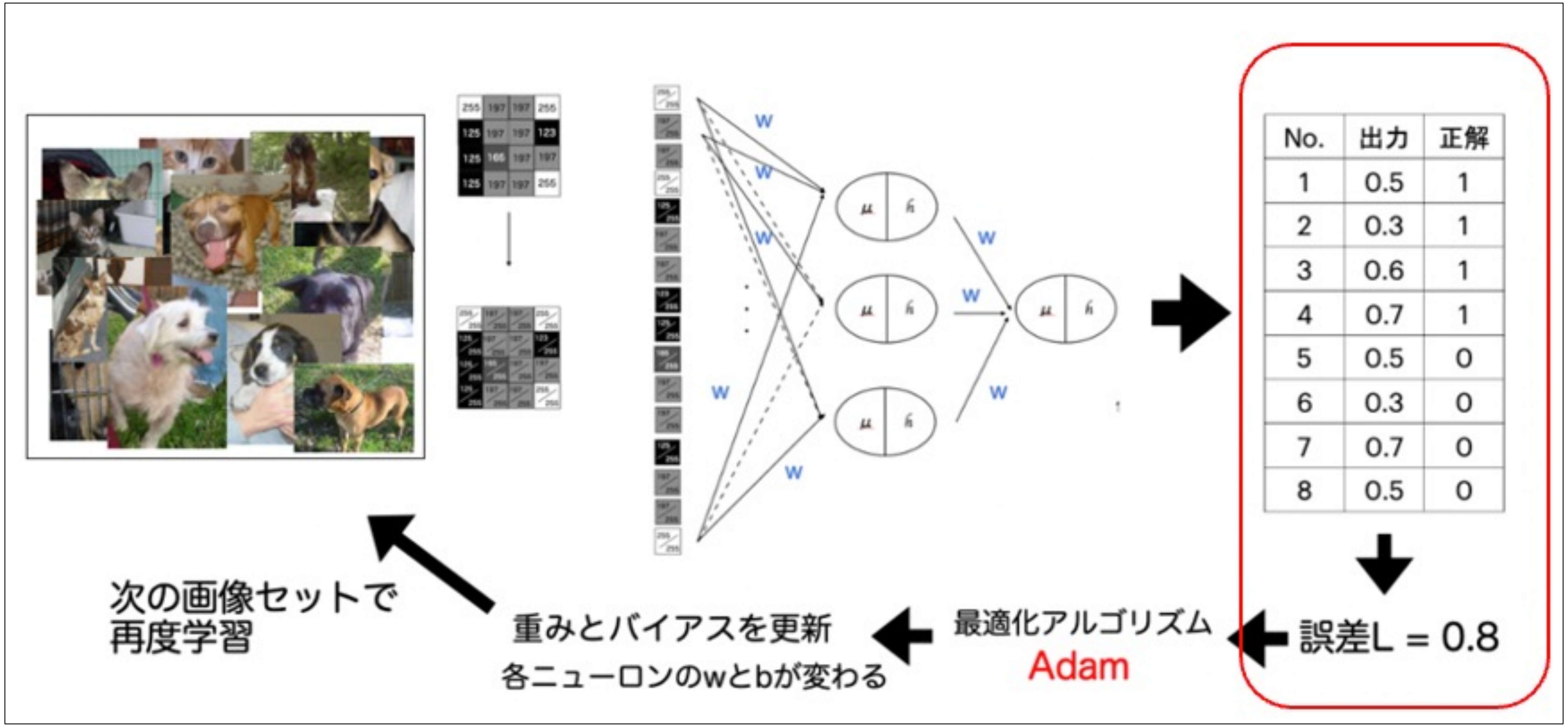
分類問題で使用

yは0~1の確率的な値

今回は考えやすいように二乗和誤差で考える

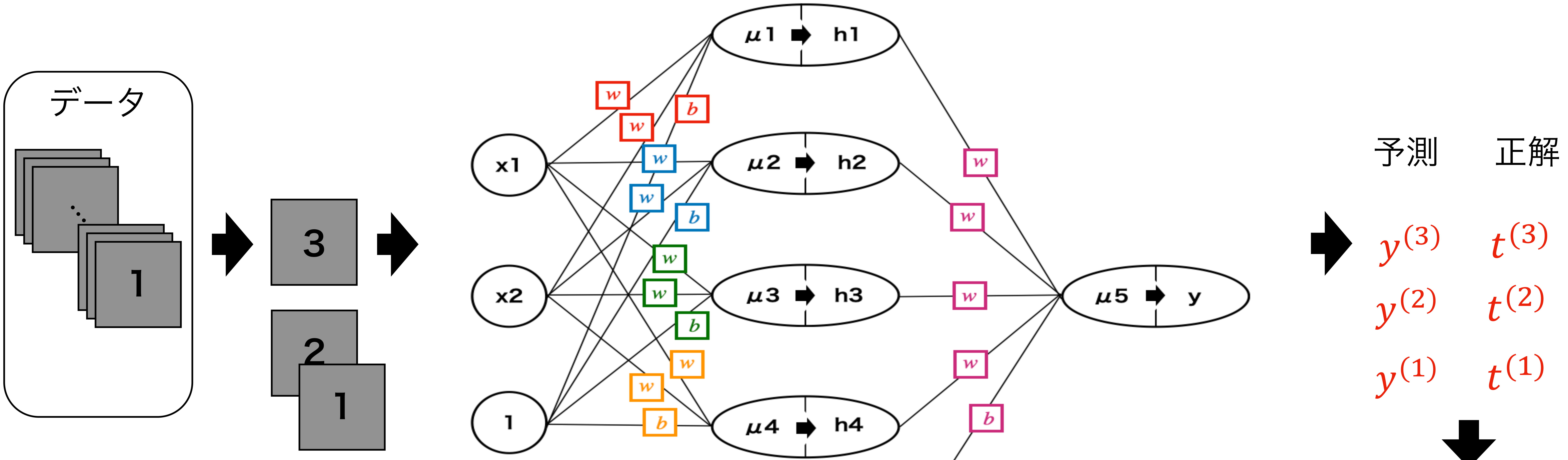
推論の結果得られる誤差Lを小さくしていきたい

$$L = \frac{1}{N}\sum_{i=1}^{n}(y_i - t_i)^2$$



このLが最小の時はどんなとき？
そもそも変数は何か？

損失関数



[[0.5 , 0.3] 画像1(1)
[0.2 , 0.4] 画像2(0)
[0.6 , 0.2]]画像3(0)

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

変数が多いので一度整理　x, w, b, μ, h, y, L, t

損失関数

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

二乗和誤差で考えてみます。

$t^{(k)}$は定数(0,1など)なので、変数は予測結果 $y^{(k)}$ のみです

入力データx　　正解t

[[0.5 , 0.3] 画像1(1)
[0.2 , 0.4] 画像2(0)
[0.6 , 0.2]]画像3(0)

```
result = model.predict(data)
result
```

```
1/1 [==============================] - 0s 192ms/step
array([[0.54587066],
       [0.54487884],
       [0.53991485]], dtype=float32)
```

予測結果y

$$L = \frac{1}{3}\sum_{k=1}^{3}(y^{(k)} - t^{(k)})^2 = \frac{1}{3}((0.54587066 - 1)^2 + (0.5448884 - 0)^2 + (0.53991485 - 0)^2)$$

Lが小さくなるにはy(k)が変わるしかない

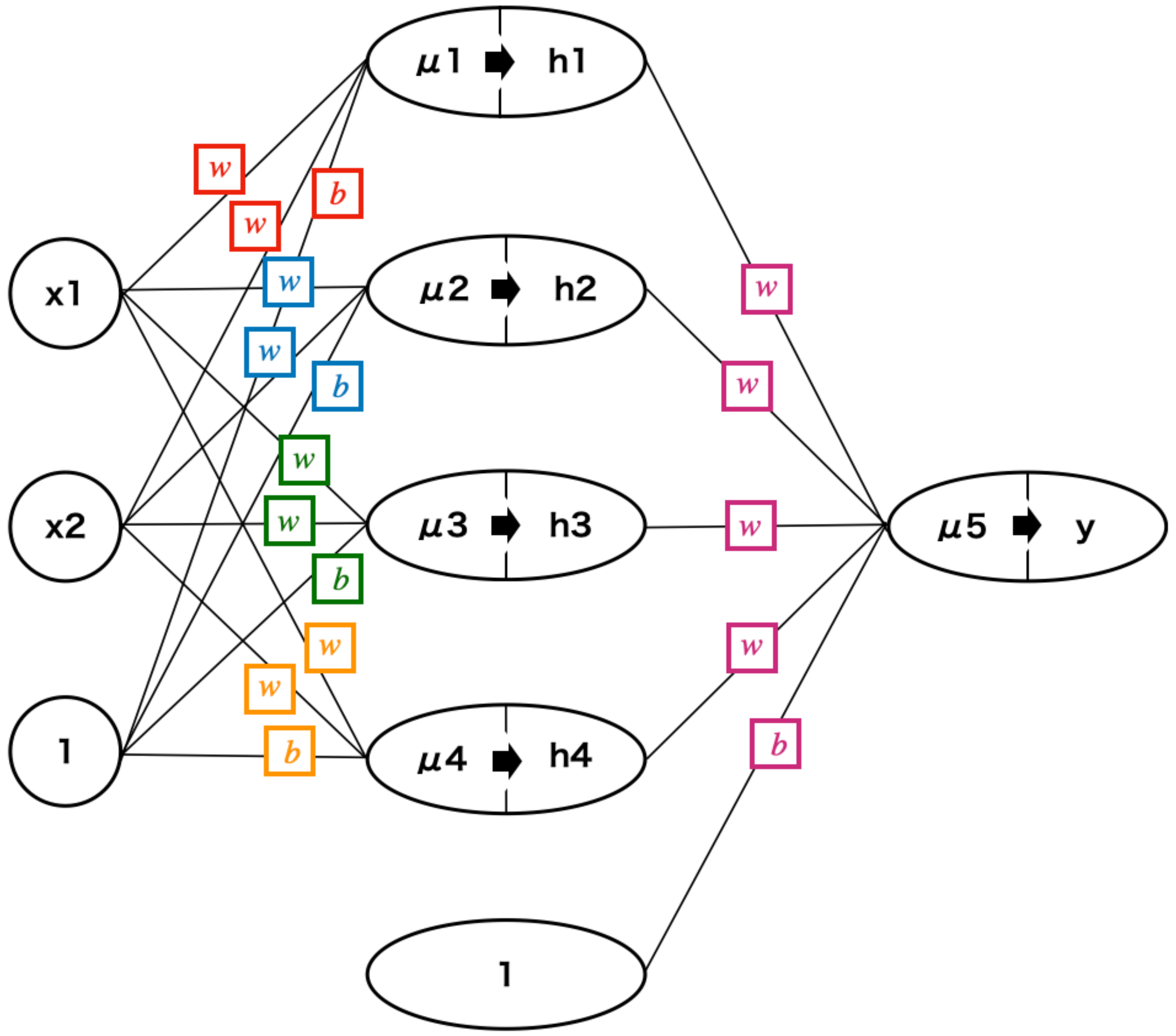
損失関数

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

二乗和誤差で考えてみます。

$t^{(k)}$は定数(0,1など)なので、変数は予測結果 $y^{(k)}$ のみです

推論でやったように、$y^{(k)}$ はモデル内の全ての重みとバイアスで表せます



$$y = sigmoid(\mu5)$$

$$\mu5 = w9 \times h1 + w10 \times h2 + w11 \times h3 + w12 \times h4 + b5$$

$$h = relu(\mu) \qquad h: h1 \sim h4, \mu: \mu1 \sim \mu4$$

$$\mu = w1 \times x1 + w2 \times x2 + b1 \qquad \mu: \mu1 \sim \mu4$$

推論時はwとbを固定するので

$$y = f(x)$$ **となる**

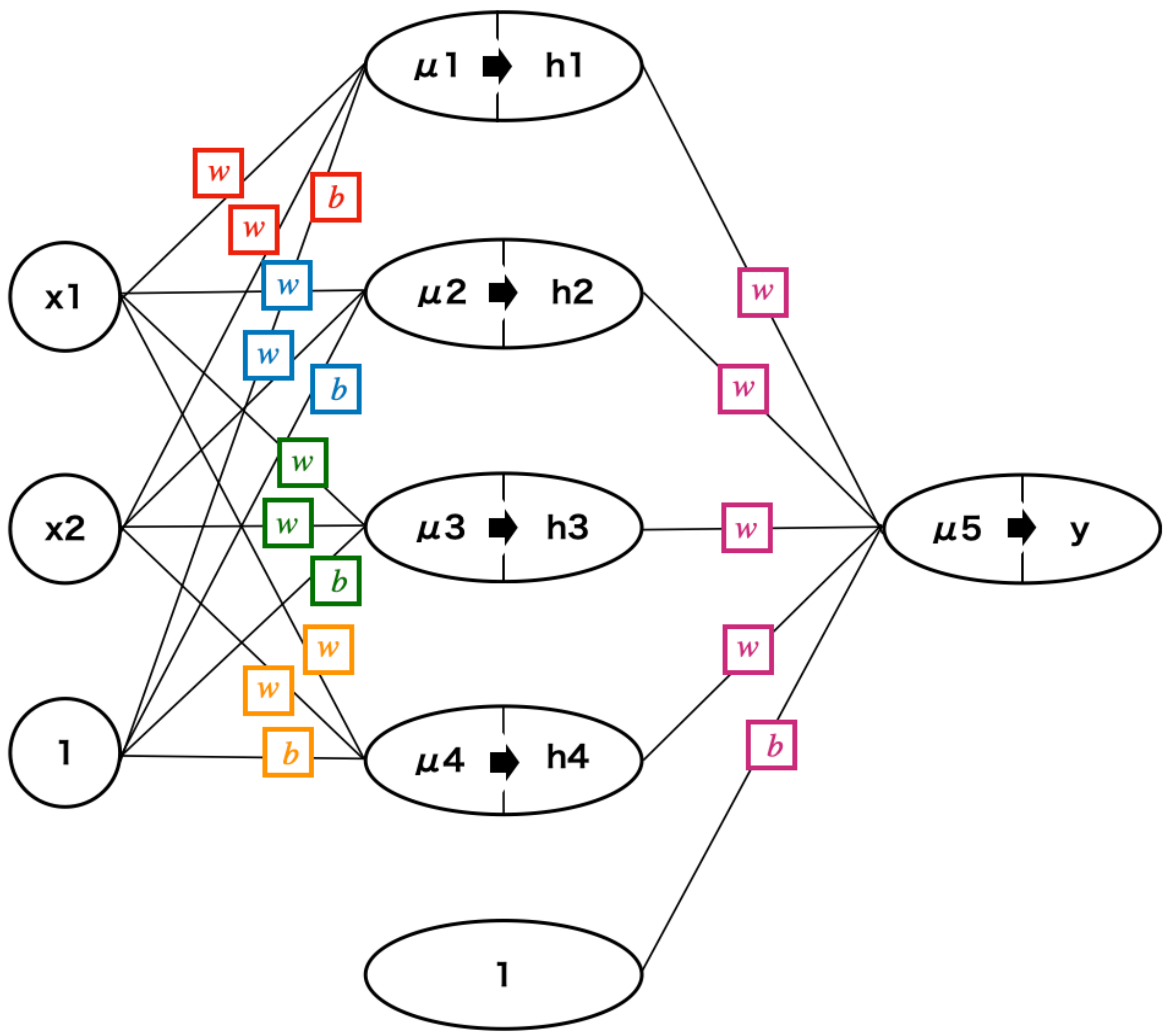
損失関数

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

二乗和誤差で考えてみます。

$t^{(k)}$は定数(0,1など)なので、変数は予測結果 $y^{(k)}$ のみです

推論でやったように、$y^{(k)}$ はモデル内の全ての重みとバイアスで表せます



学習はwとbを変化させて、Lを小さくしたいと考える

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

$$= g(w, b)$$

xを固定していれば、wとbが変わると、
yが変わってLが変わる

学習

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

L は、重みとバイアスの関数 $L(w, b)$ になる

$$L = g(w, b)$$

学習

$$L = \frac{1}{N}\sum_{k=1}^{n}(y^{(k)} - t^{(k)})^2$$

L は、重みとバイアスの関数 $L(w, b)$ になる

$$L = g(w, b)$$

勾配降下法という手法を用います

(これまで出てきたAdamはこれを改良したものです)

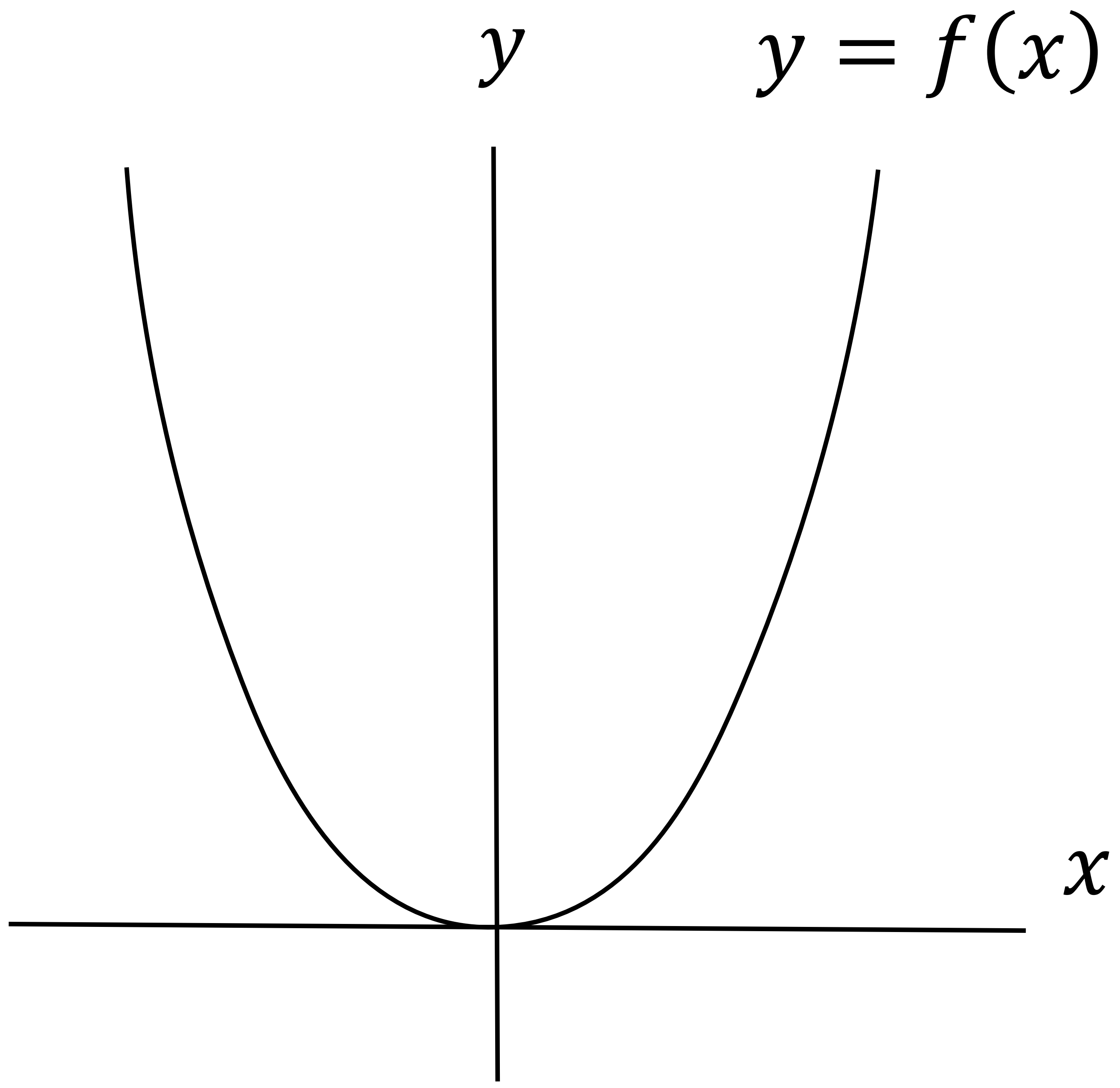
$$w \leftarrow w - \alpha \frac{\partial L}{\partial w} \qquad b \leftarrow b - \alpha \frac{\partial L}{\partial b}$$

勾配降下法は上のような式によって重みとバイアスを更新します

ちょっとだけ微分の復習

$y = x^2$ の微分

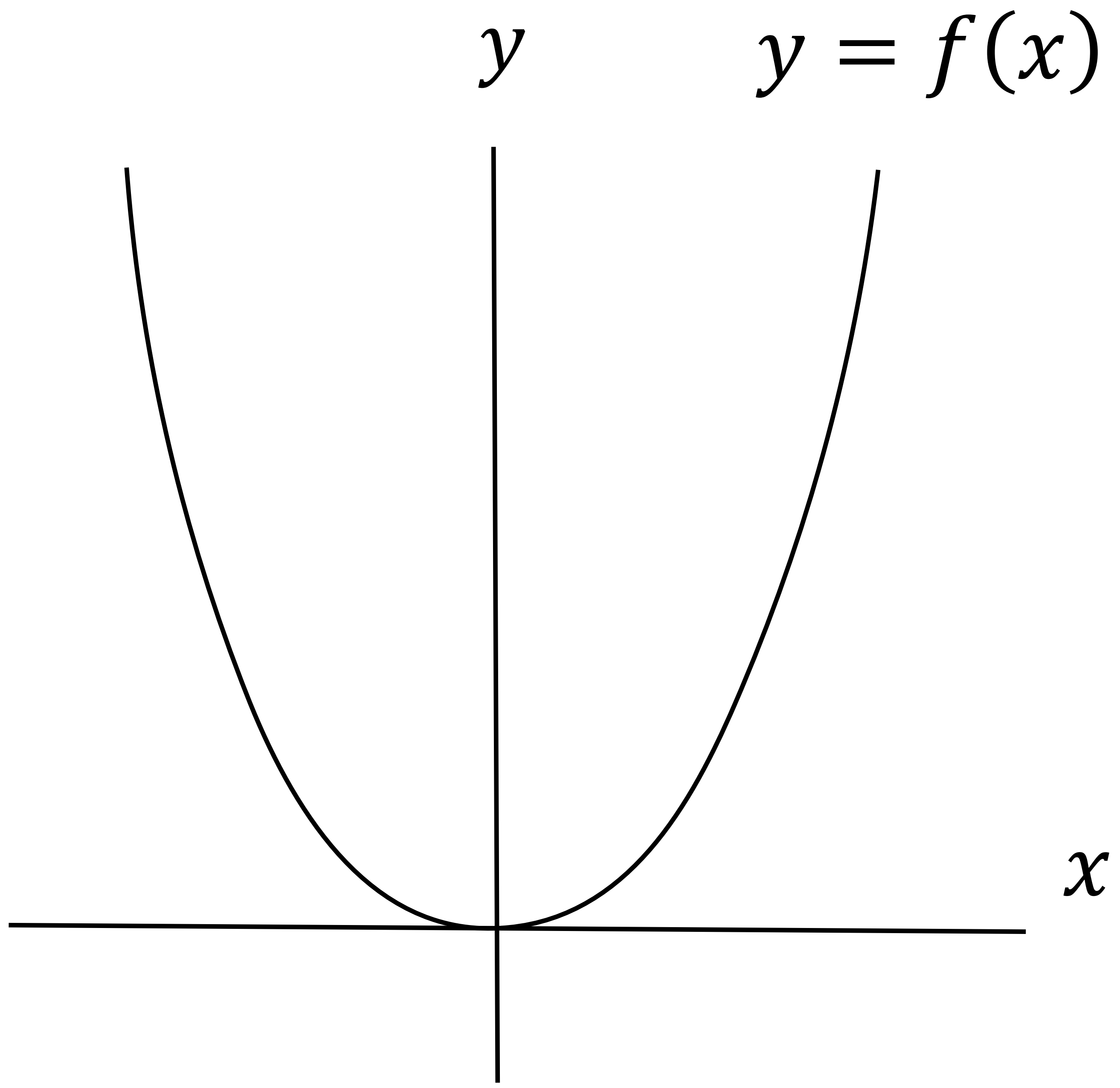
xが少しだけ動いた時のyの変化量 = 傾き



ちょっとだけ微分の復習

$y = x^2$ の微分

xが少しだけ動いた時のyの変化量 = 傾き



$$f'(x) = 2x$$

$$y' = 2x$$

$$\frac{dy}{dx} = 2x$$

$y = f(x)$
とすれば

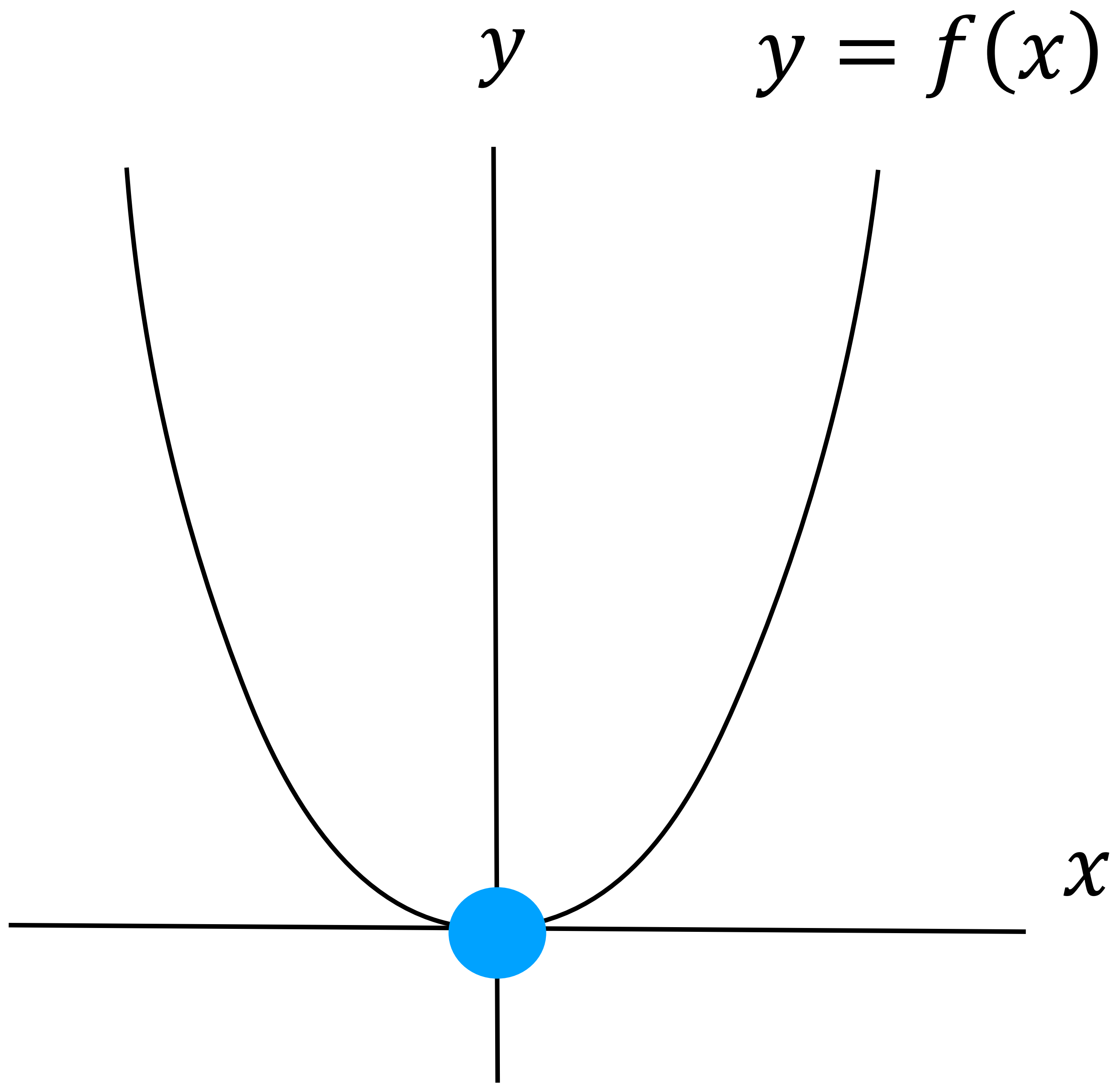
$$\frac{df(x)}{dx} = 2x$$

$$\frac{d}{dx}f(x) = 2x$$

ちょっとだけ微分の復習

$y = x^2$ の微分

xが少しだけ動いた時のyの変化量 = 傾き



$f'(x) = 2x$

$y' = 2x$

$\frac{dy}{dx} = 2x$

$y = f(x)$ とすれば

$\frac{df(x)}{dx} = 2x$

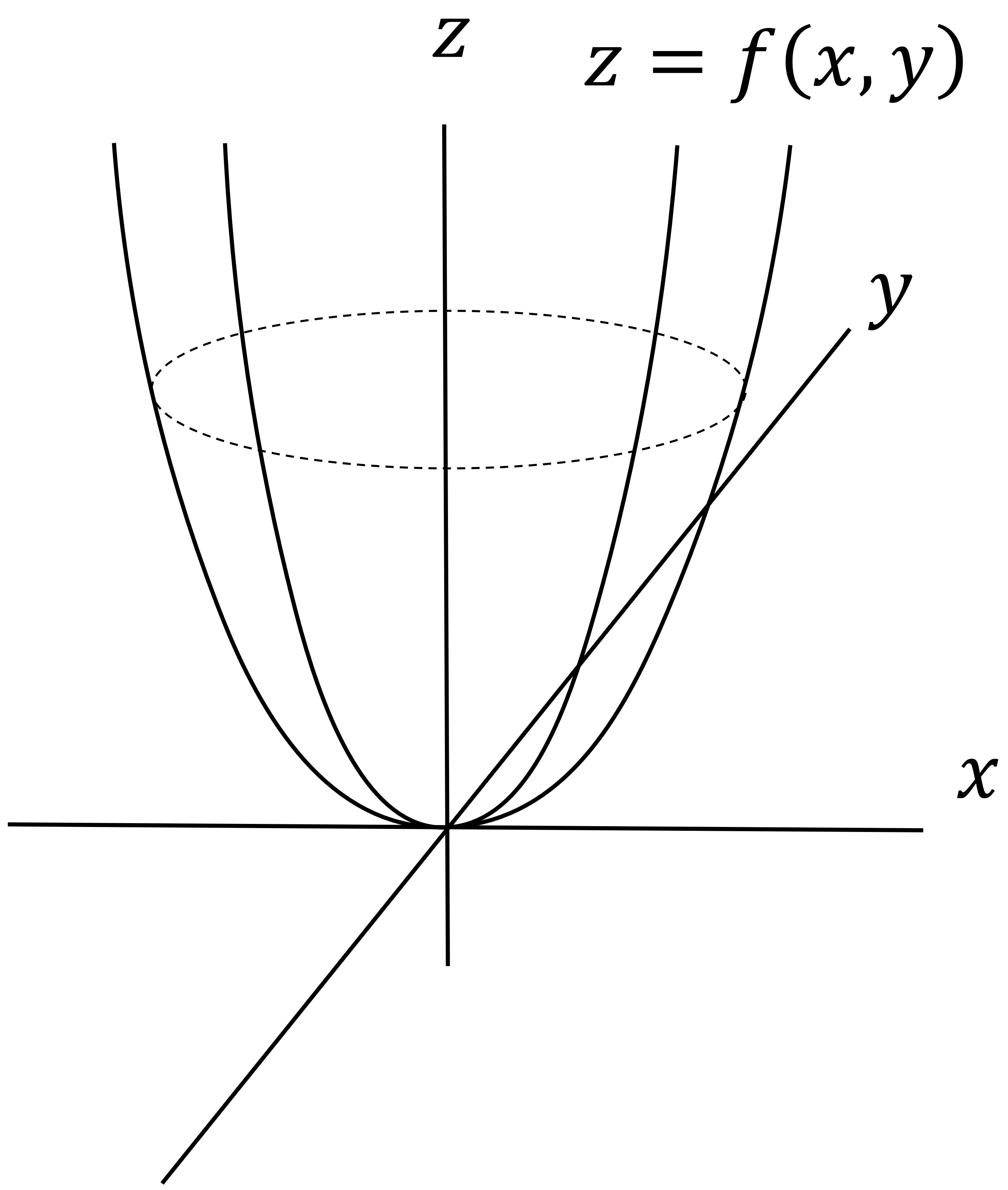
$\frac{d}{dx}f(x) = 2x$

$\frac{dy}{dx} = 2x$ は x=0 の時が傾きが0で y(=f(x)) が最小

ちょっとだけ微分の復習

$z = x^2 + y^2$ の偏微分

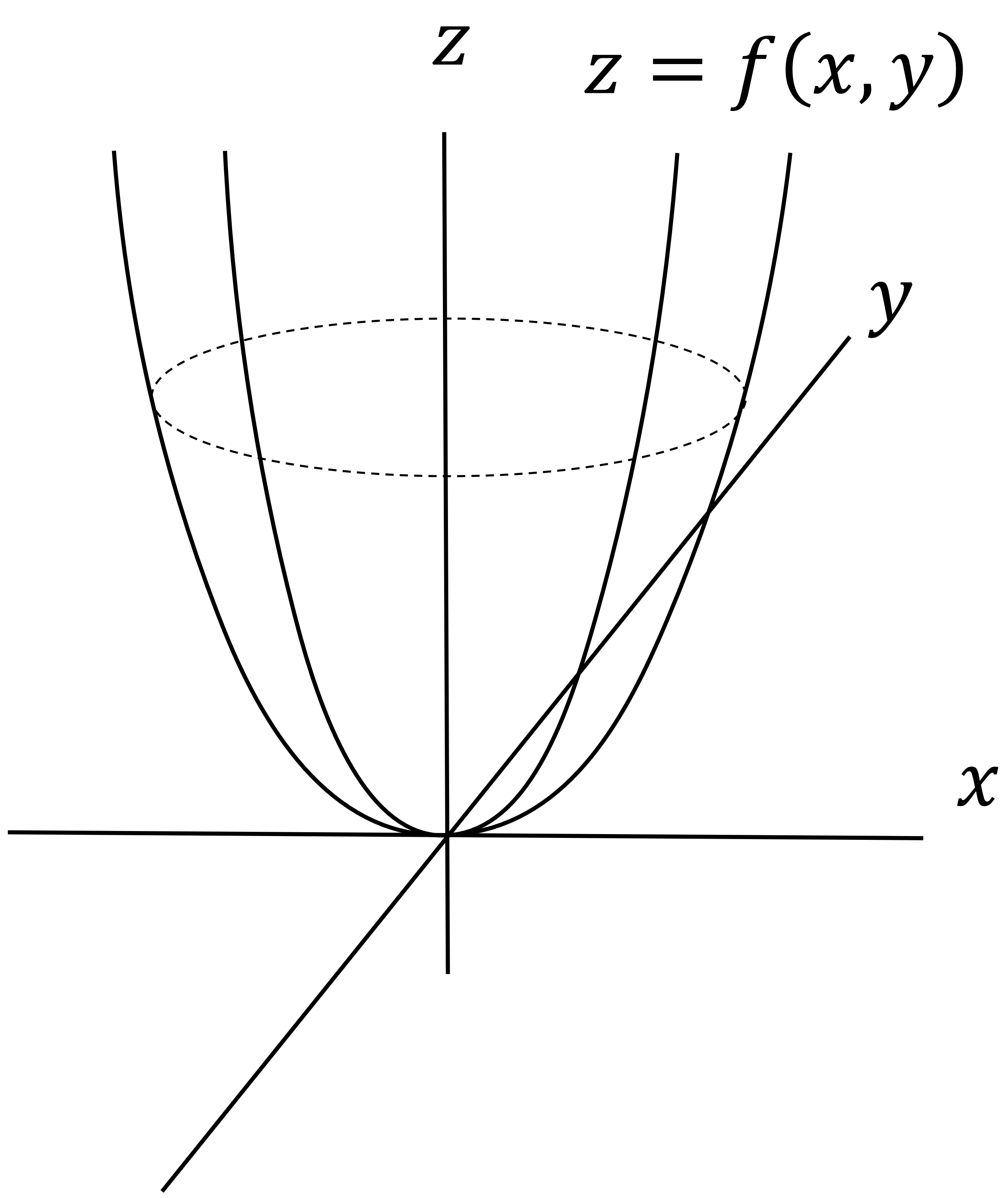
yを固定してxが少しだけ動いた時のzの変化量 = 傾き
xを固定してyが少しだけ動いた時のzの変化量 = 傾き



ちょっとだけ微分の復習

$z = x^2 + y^2$ の偏微分

yを固定してxが少しだけ動いた時のzの変化量 = 傾き
xを固定してyが少しだけ動いた時のzの変化量 = 傾き



$\frac{\partial z}{\partial x}$ (または $\frac{\partial z}{\partial y}$) は y (またはx) を定数とするので、

$$\frac{\partial z}{\partial x} = 2x$$

$$\frac{\partial z}{\partial y} = 2y$$

$z = f(x, y)$
とすれば

$$\frac{\partial f(x, y)}{\partial x} = 2x$$

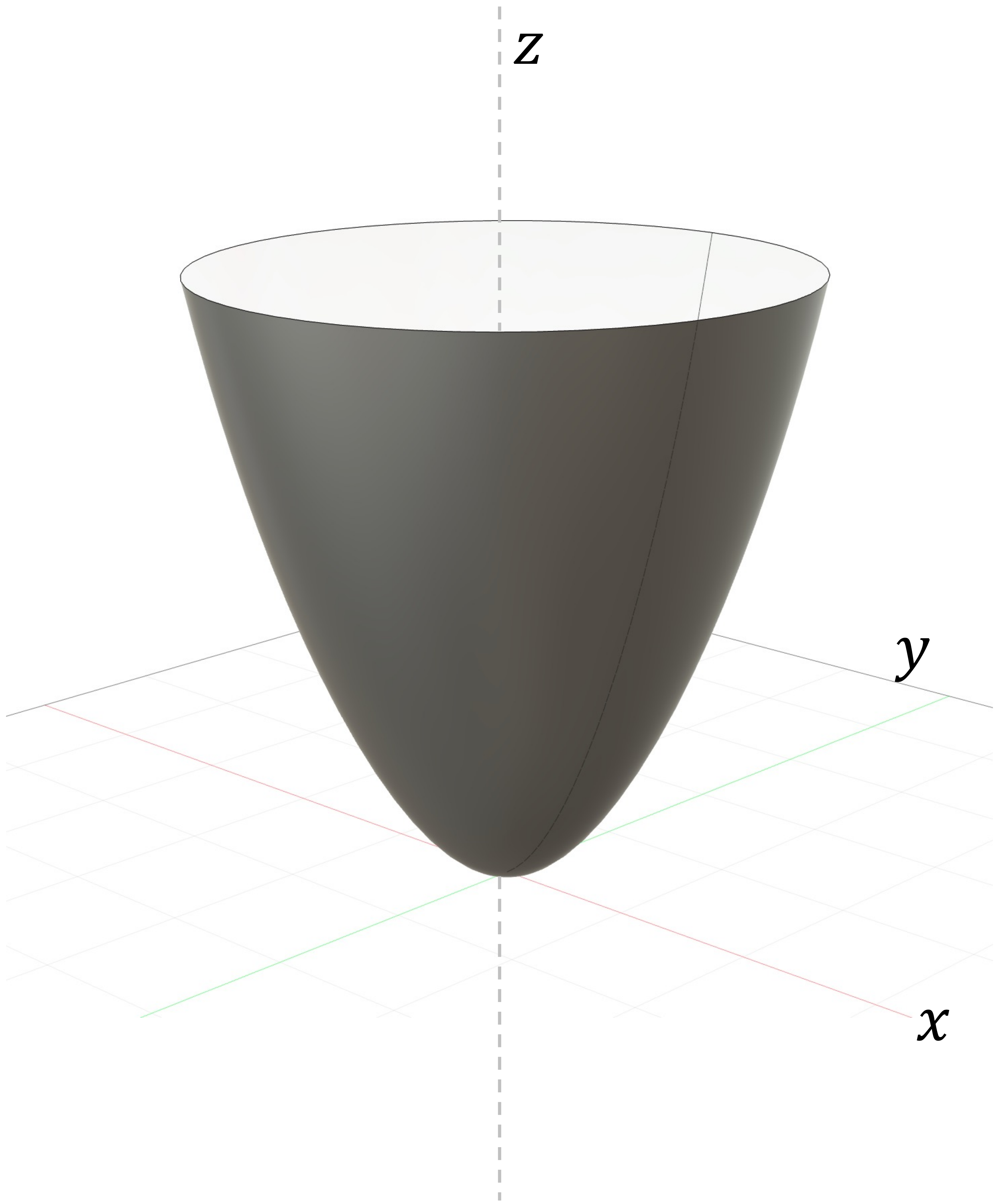
$$\frac{\partial f(x, y)}{\partial y} = 2y$$

この場合、xもyも0の時が、傾き0でzは最小

ちょっとだけ微分の復習

$z = x^2 + y^2$ の偏微分

yを固定してxが少しだけ動いた時のzの変化量 = 傾き
xを固定してyが少しだけ動いた時のzの変化量 = 傾き



$\frac{\partial z}{\partial x}$ (または $\frac{\partial z}{\partial y}$) は y (またはx) を定数とするので、

$z = f(x, y)$
とすれば

$$\frac{\partial f(x,y)}{\partial x} = 2x$$

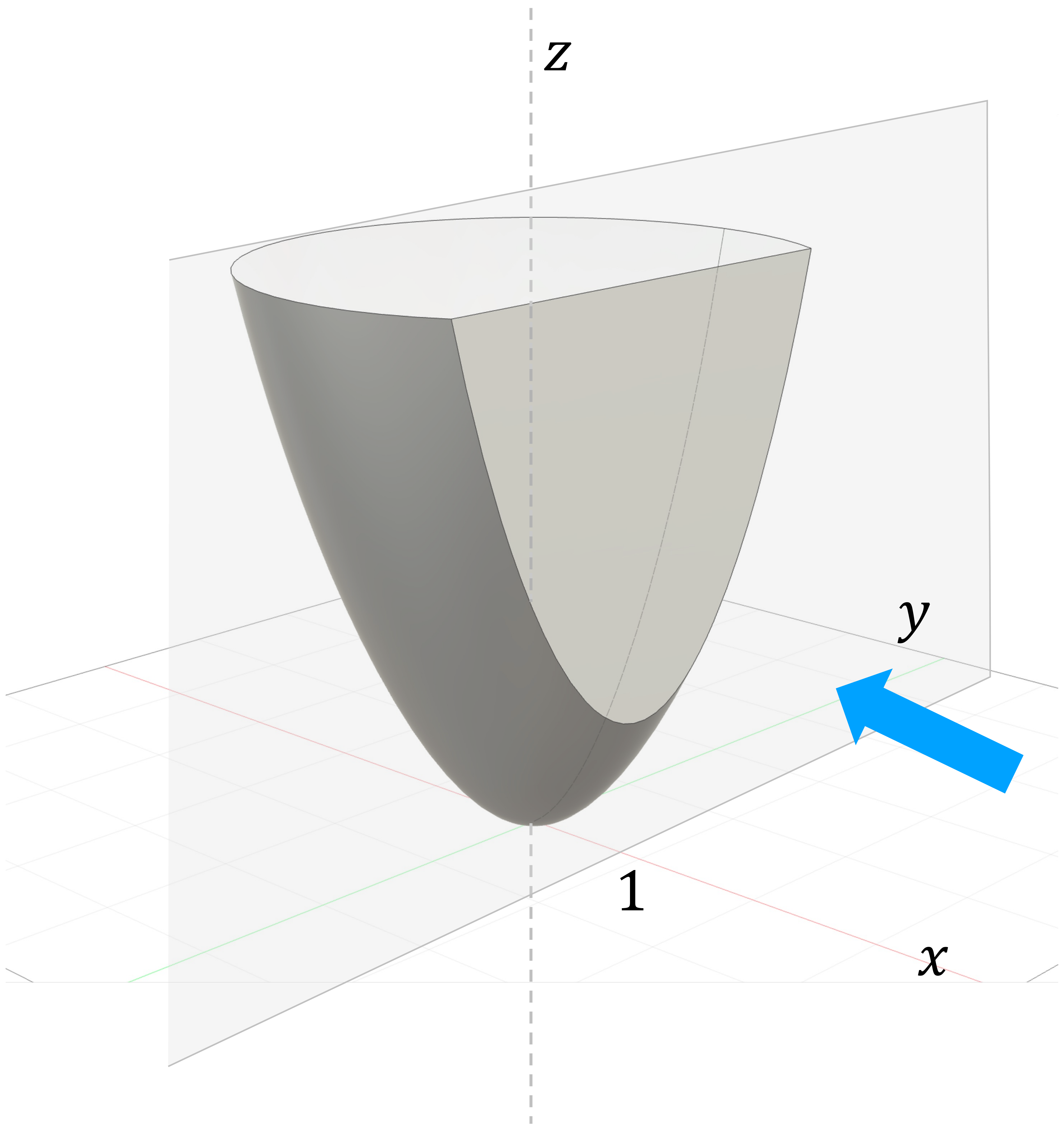
$$\frac{\partial f(x,y)}{\partial y} = 2y$$

この場合、xもyも0の時が、傾き0でzは最小

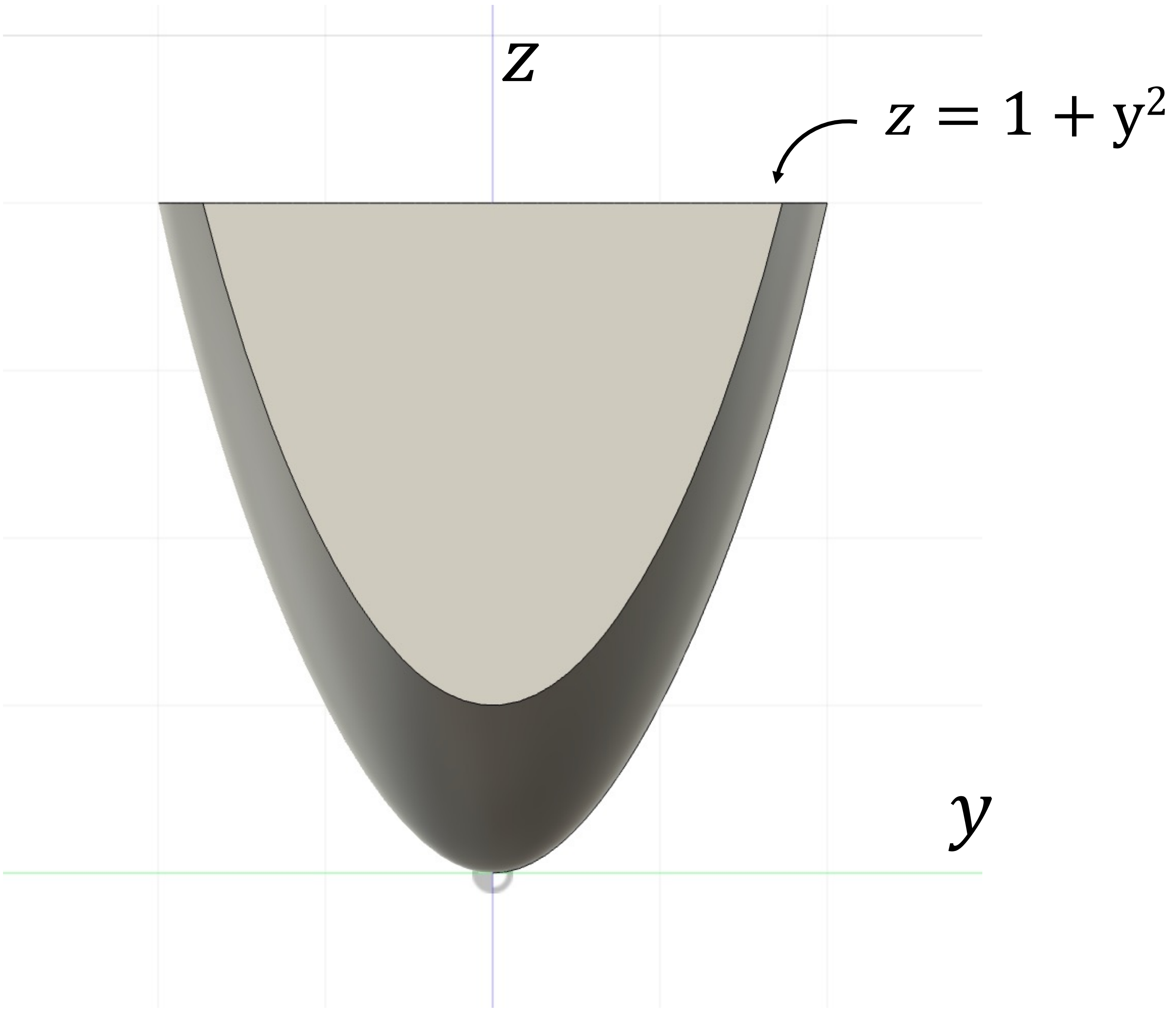
ちょっとだけ微分の復習

$z = x^2 + y^2$ の偏微分

yを固定してxが少しだけ動いた時のzの変化量 = 傾き
xを固定してyが少しだけ動いた時のzの変化量 = 傾き



x = 1の時は　$z = 1 + y^2$　で平面になる

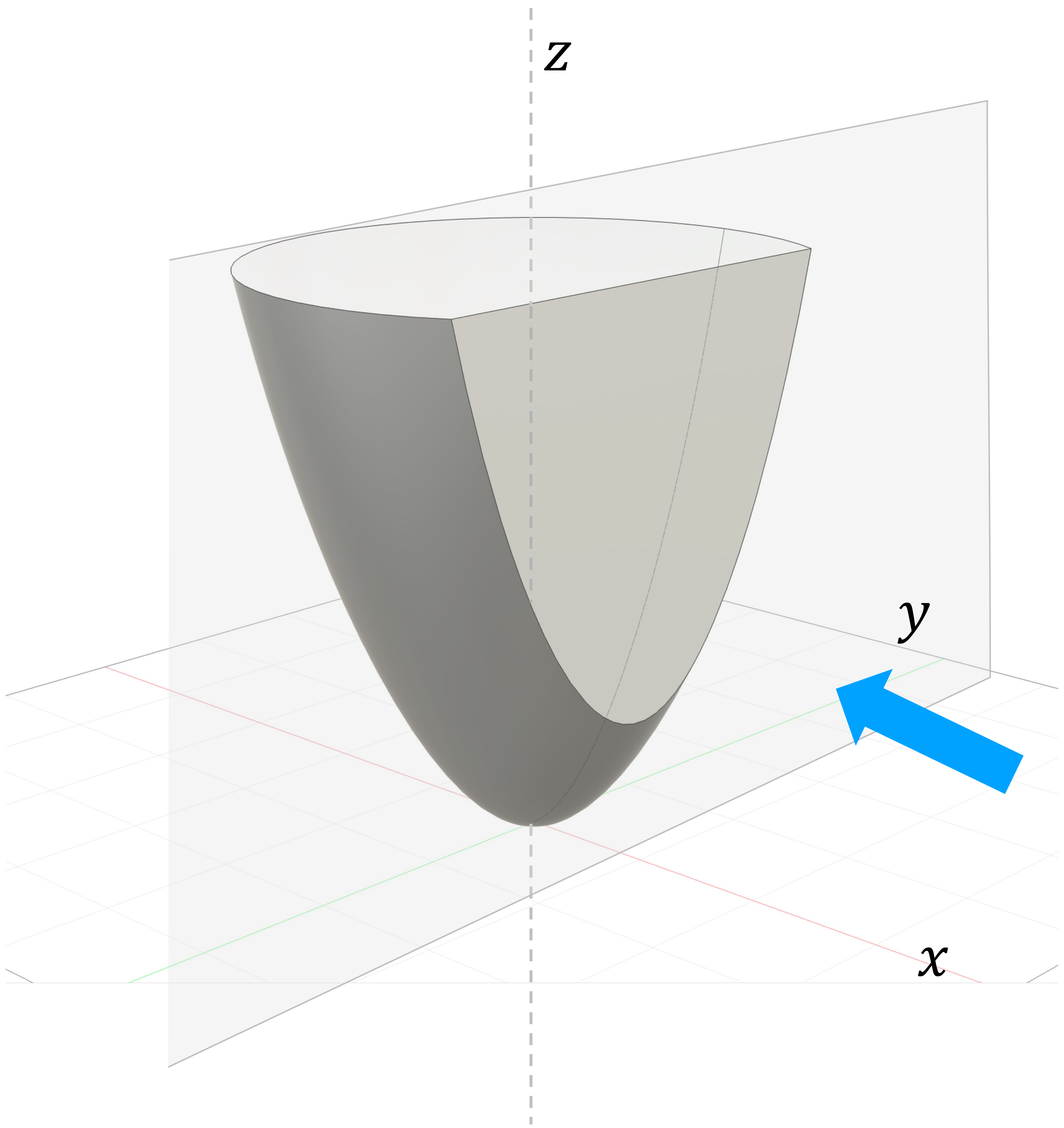


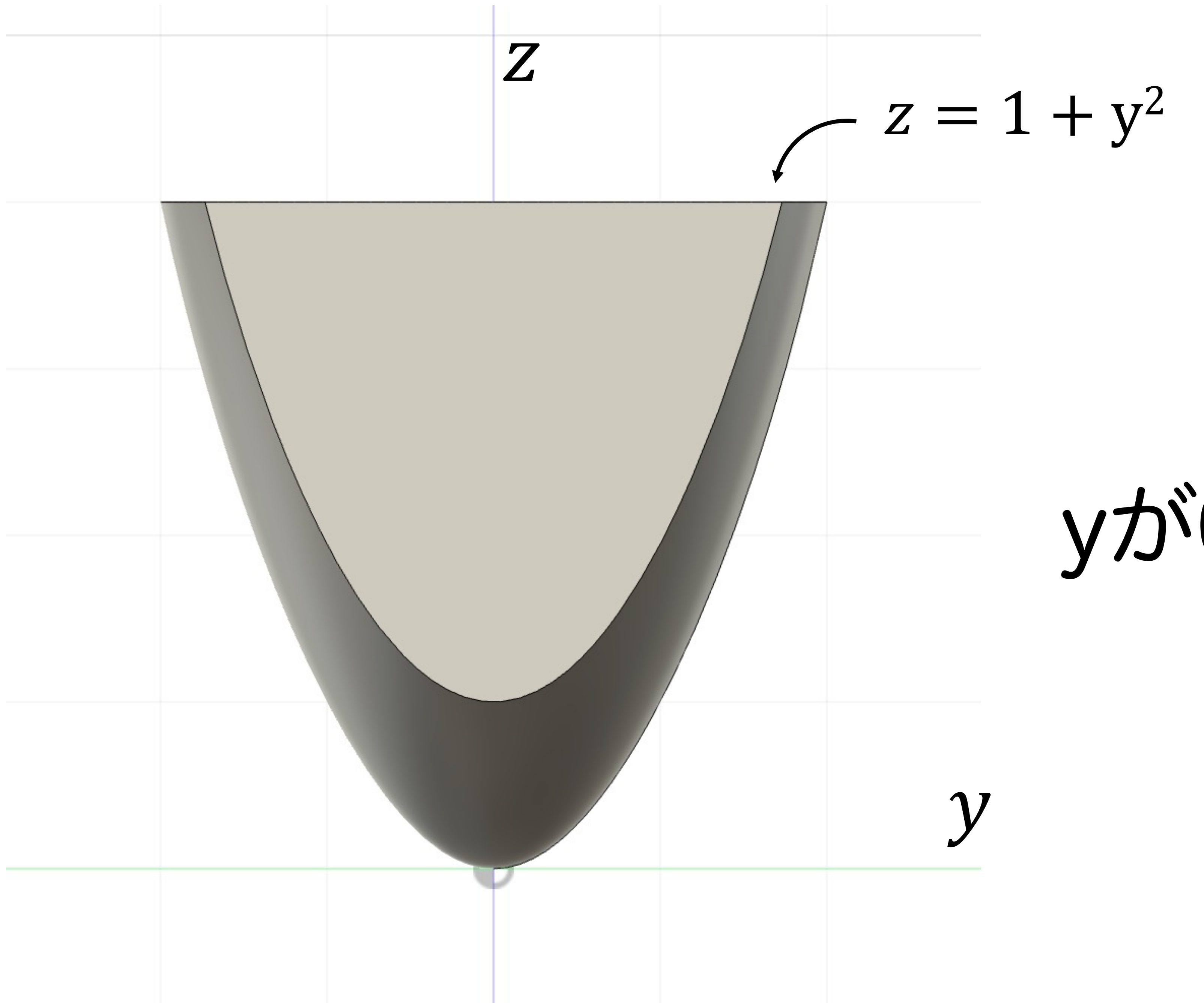
ちょっとだけ微分の復習

$z = x^2 + y^2$ の偏微分

yを固定してxが少しだけ動いた時のzの変化量 = 傾き
xを固定してyが少しだけ動いた時のzの変化量 = 傾き

x = 1の時は　$z = 1 + y^2$　で平面になる





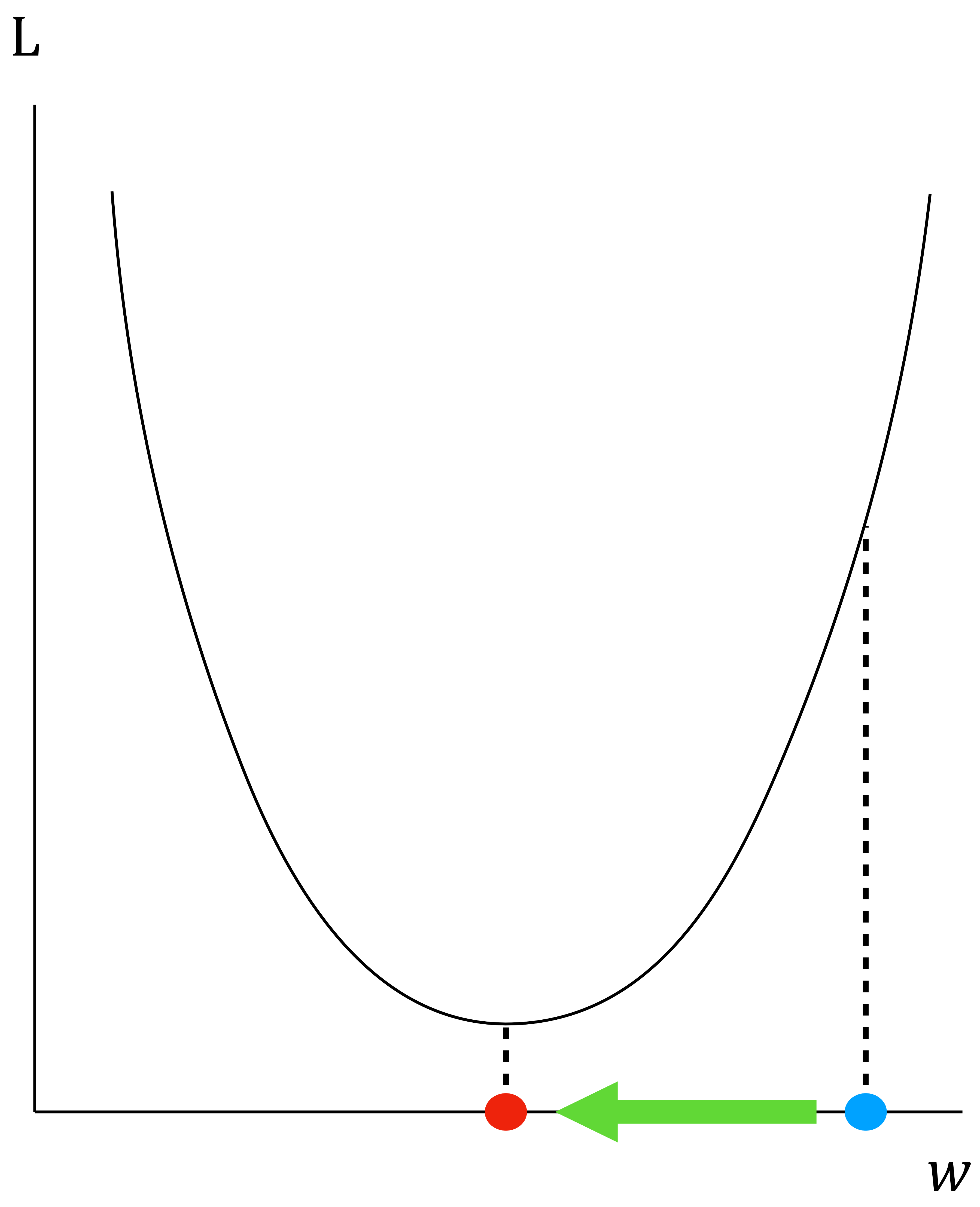
$$\frac{\partial z}{\partial y} = 2y$$

yが0の時、傾き0でzは最小

xがどの値でも同様
yも同様

変数が複数あっても偏微分で他の変数を
固定すればzの微小変化が分かる

勾配降下法



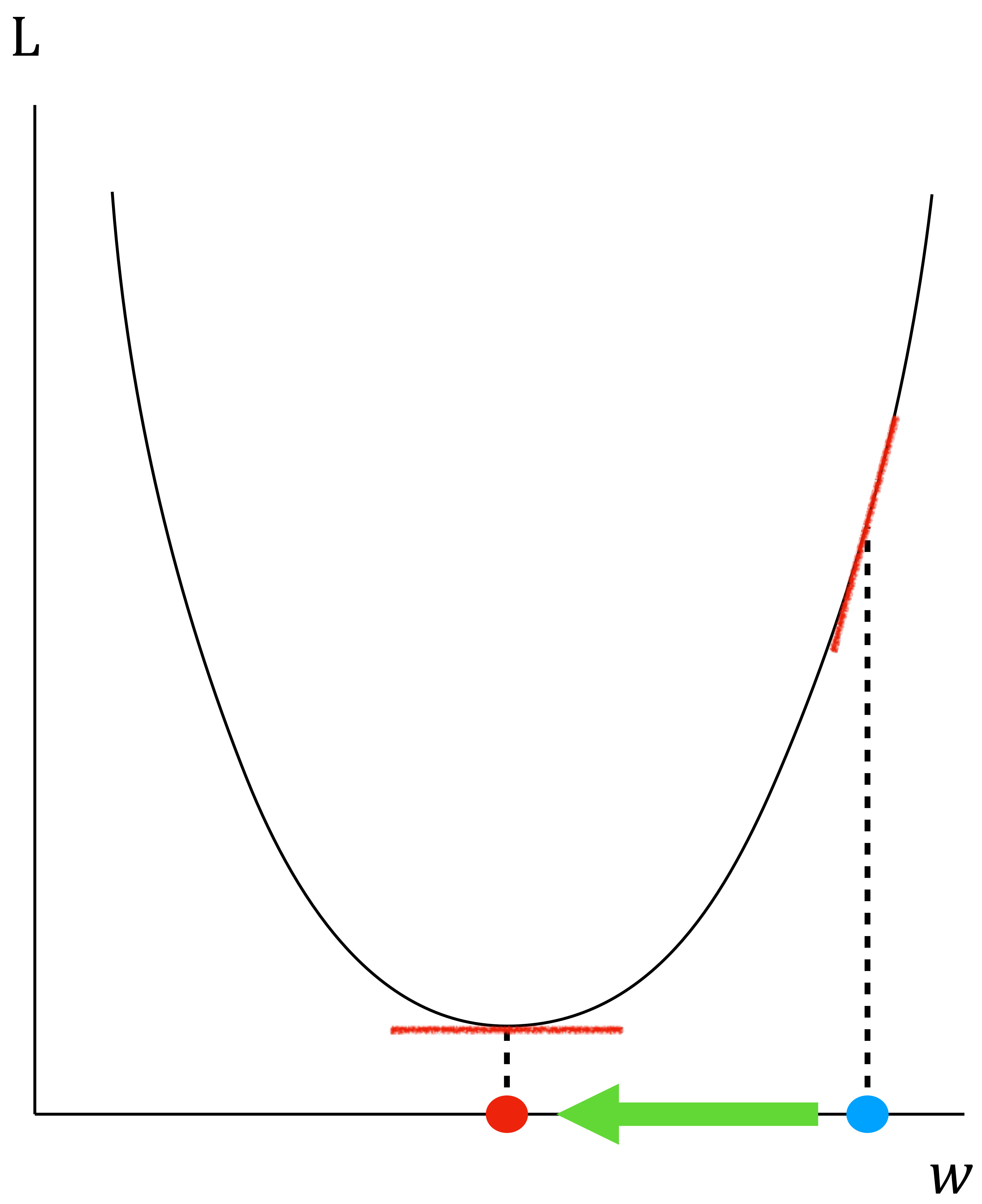
w,bを変えてL(w,b)を小さくしたい

bを固定してwだけでLの変化を考える

左の図のように縦軸Lと横軸に1つのwを考えたとき、

wを ● から ● に移動させるとLは最小になる

勾配降下法



w,bを変えてL(w,b)を小さくしたい

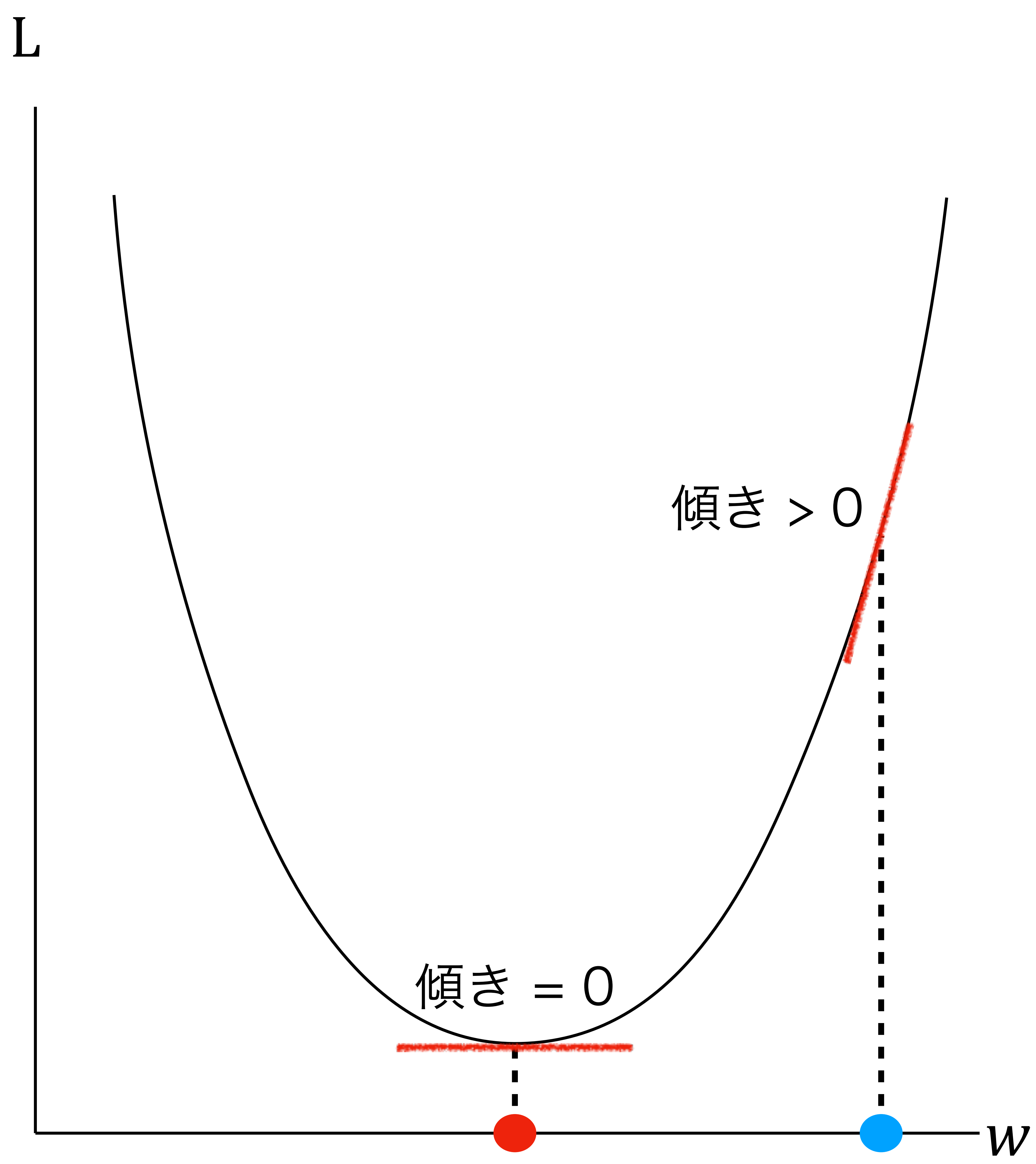
bを固定してwだけでLの変化を考える

左の図のように縦軸Lと横軸に1つのwを考えたとき、

wを ● から ● に移動させるとLは最小になる

$\boxed{\text{曲線の傾き(勾配)に着目する}}$

勾配降下法



w,bを変えてL(w,b)を小さくしたい

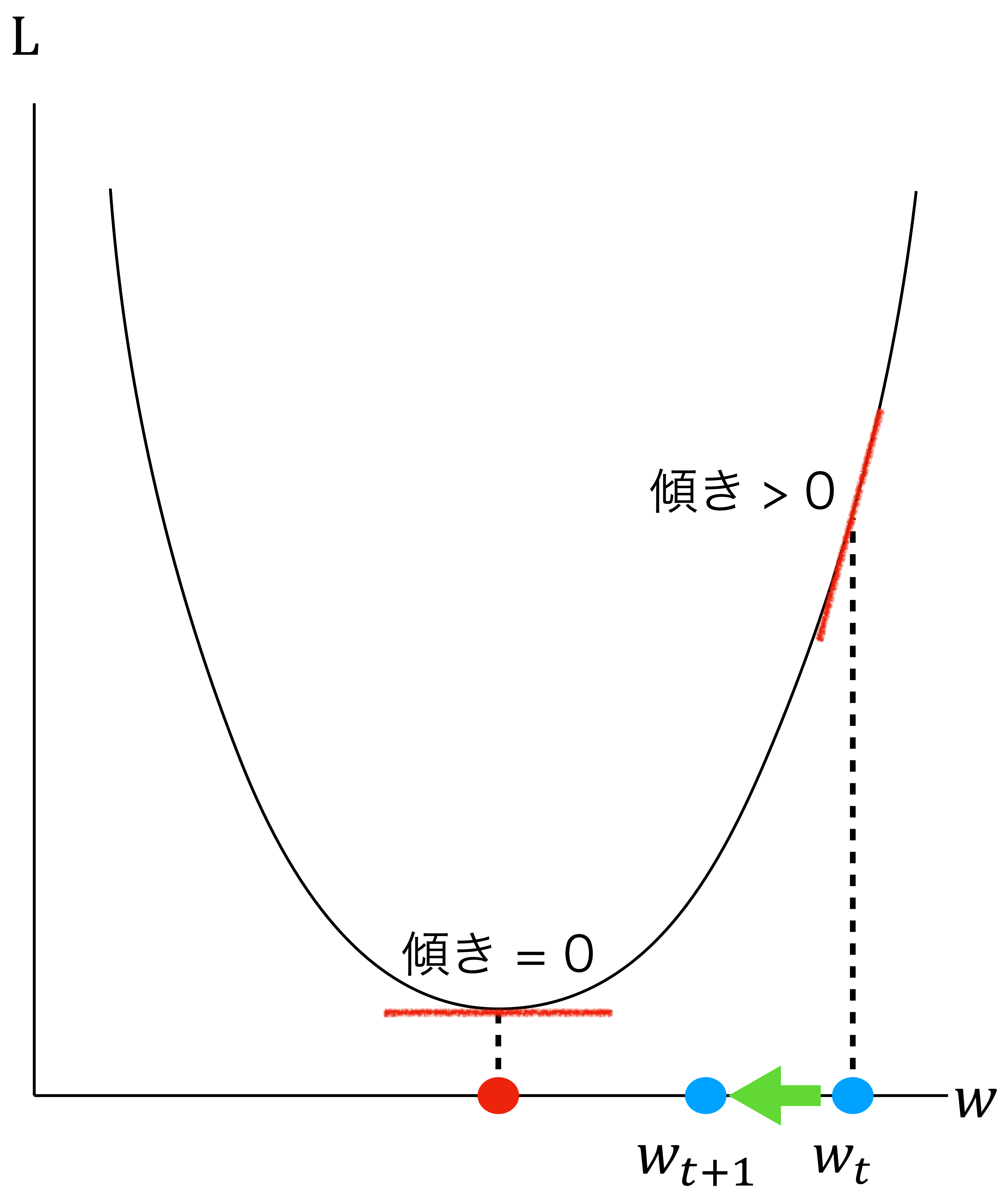
bを固定してwだけでLの変化を考える

左の図のように縦軸Lと横軸に1つのwを考えたとき、

wを から に移動させるとLは最小になる

曲線の傾き(勾配)に着目する

勾配降下法



w,bを変えてL(w,b)を小さくしたい

bを固定してwだけでLの変化を考える

左の図のように縦軸Lと横軸に1つのwを考えたとき、

wを から に移動させるとLは最小になる

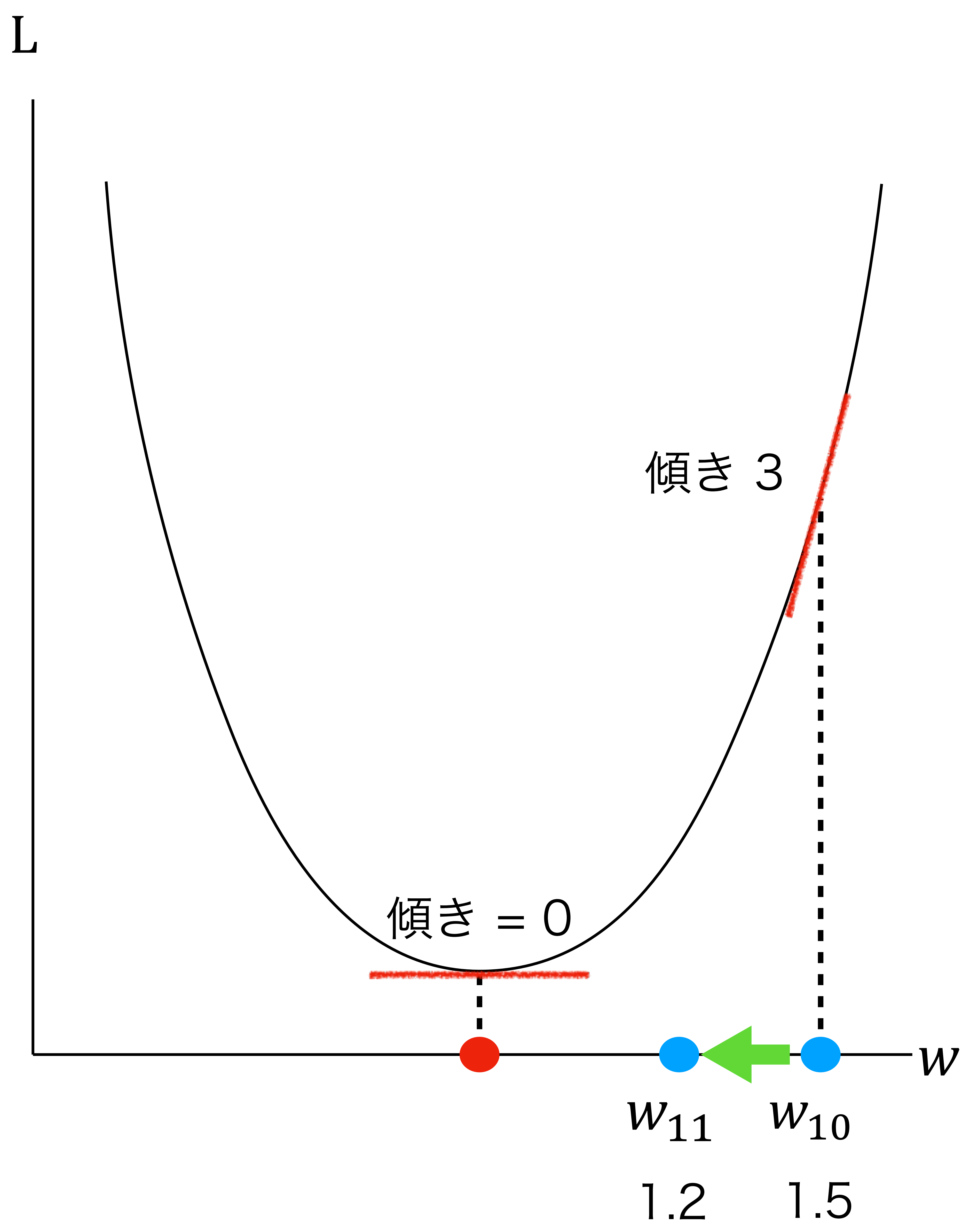
曲線の傾き(勾配)に着目する

L のw_t における傾きは $\frac{dL}{dw_t}$ (偏微分で書くと$\frac{\partial L}{\partial w_t}$)

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

(αは学習率と呼ばれる定数 α=0.1や0.001など)

勾配降下法

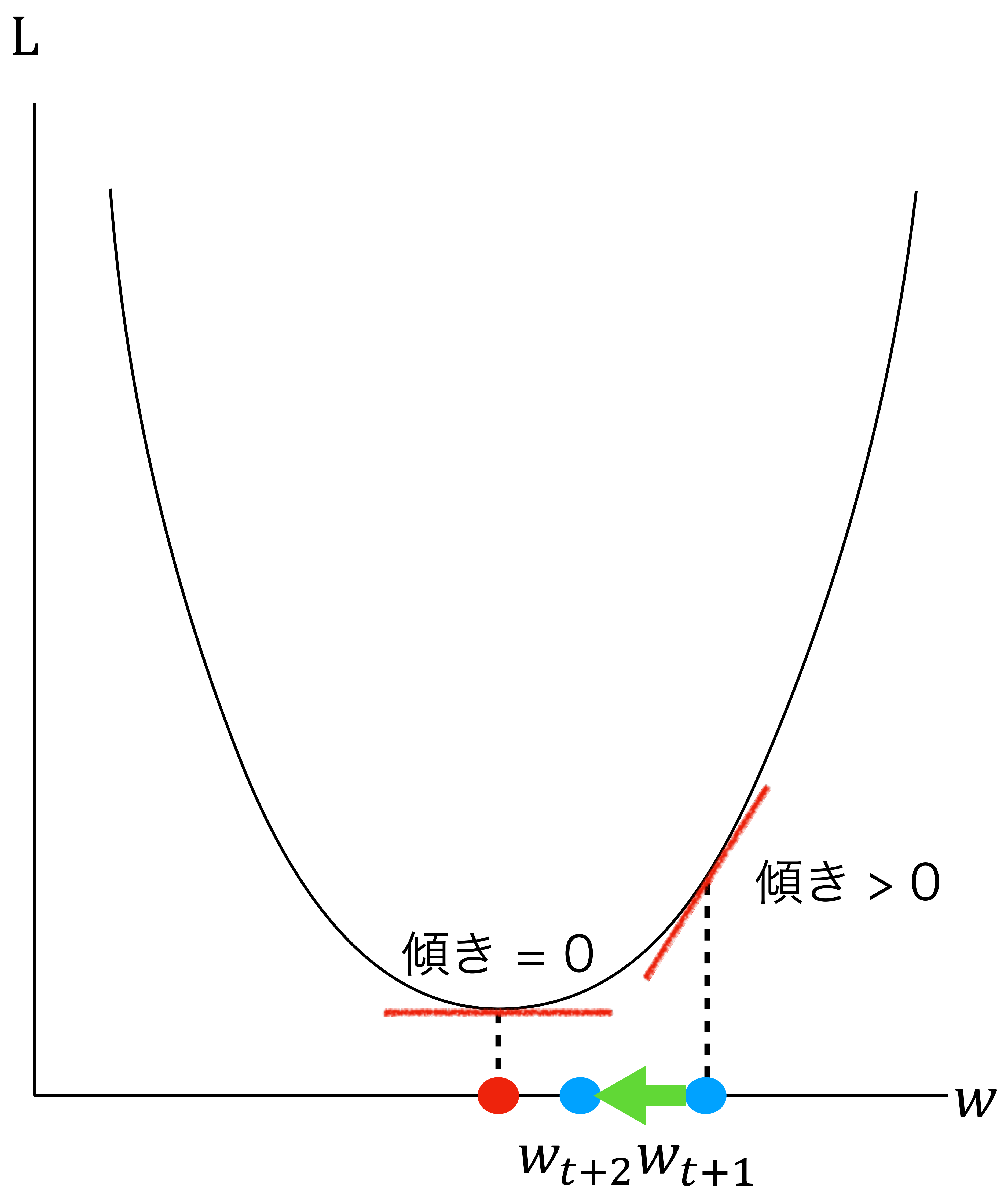


例えばepoch10回目の重みw10が1.5、
その時の傾きを3とすると、

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t} \qquad (\alpha = 0.1)$$

$$\begin{aligned} w_{11} &= w_{10} - 0.1 \times \frac{\partial L}{\partial w_{10}} \\ &= 1.5 - 0.1 \times 3 \\ &= 1.2 \end{aligned}$$

勾配降下法



wを変えてLを小さくしたい

左の図のように縦軸Lと横軸に1つのwを考えたとき、

wを 🔵 から 🔴 に移動させるとLは最小になる

$\boxed{\textbf{曲線の傾き(勾配)に着目する}}$

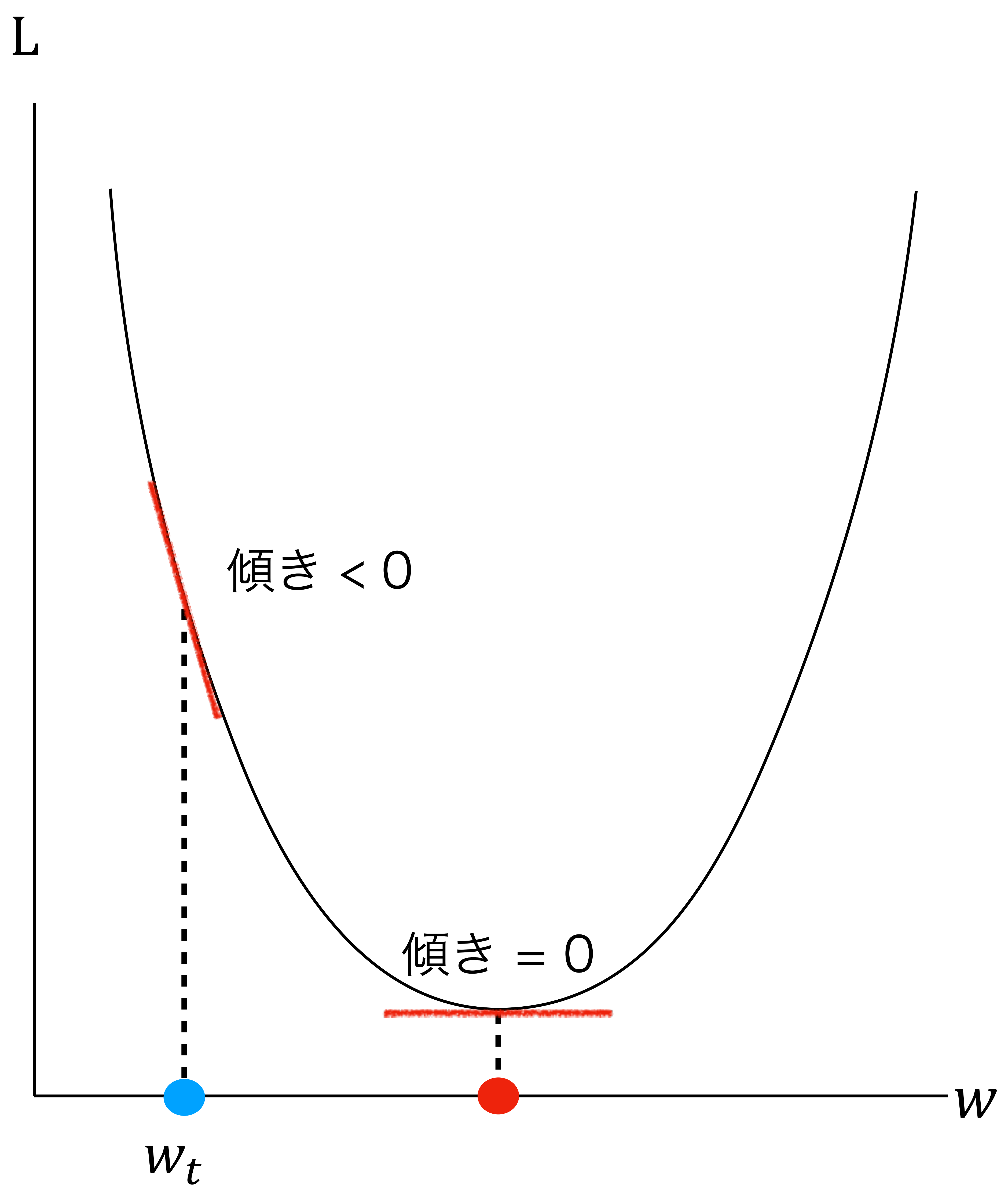
E **の**w_t **における傾きは** $\dfrac{dL}{dw_t}$ **(偏微分で書くと**$\dfrac{\partial L}{\partial w_t}$**)**

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$ （αは定数 α＝0.001など）

$$w_{t+2} = w_{t+1} - \alpha \frac{\partial L}{\partial w_{t+1}}$$

w **はLが小さくなる方向に更新される**

勾配降下法

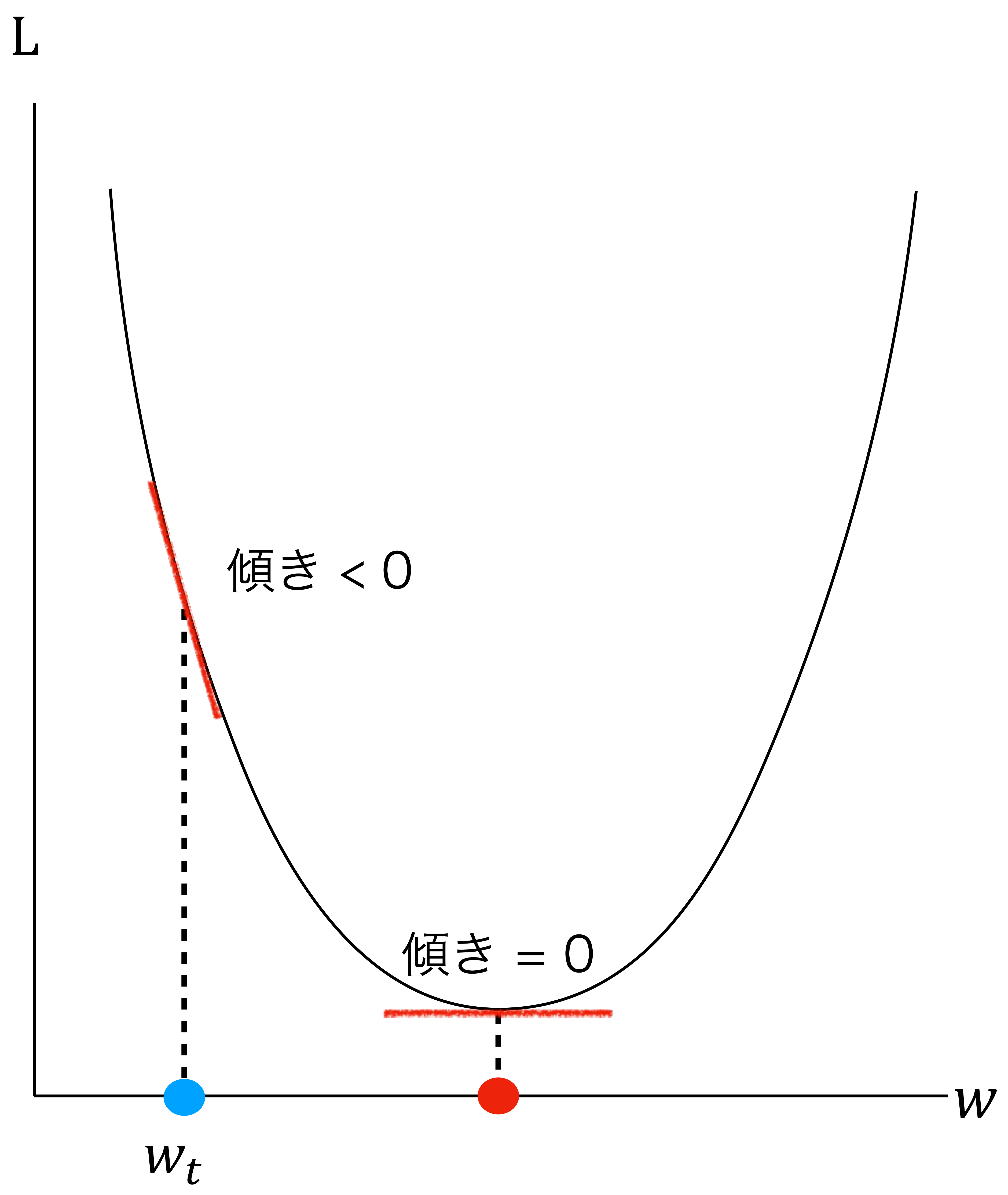


傾きが負でも同様

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

（αは定数　$\alpha = 0.001$など）

勾配降下法



傾きが負でも同様

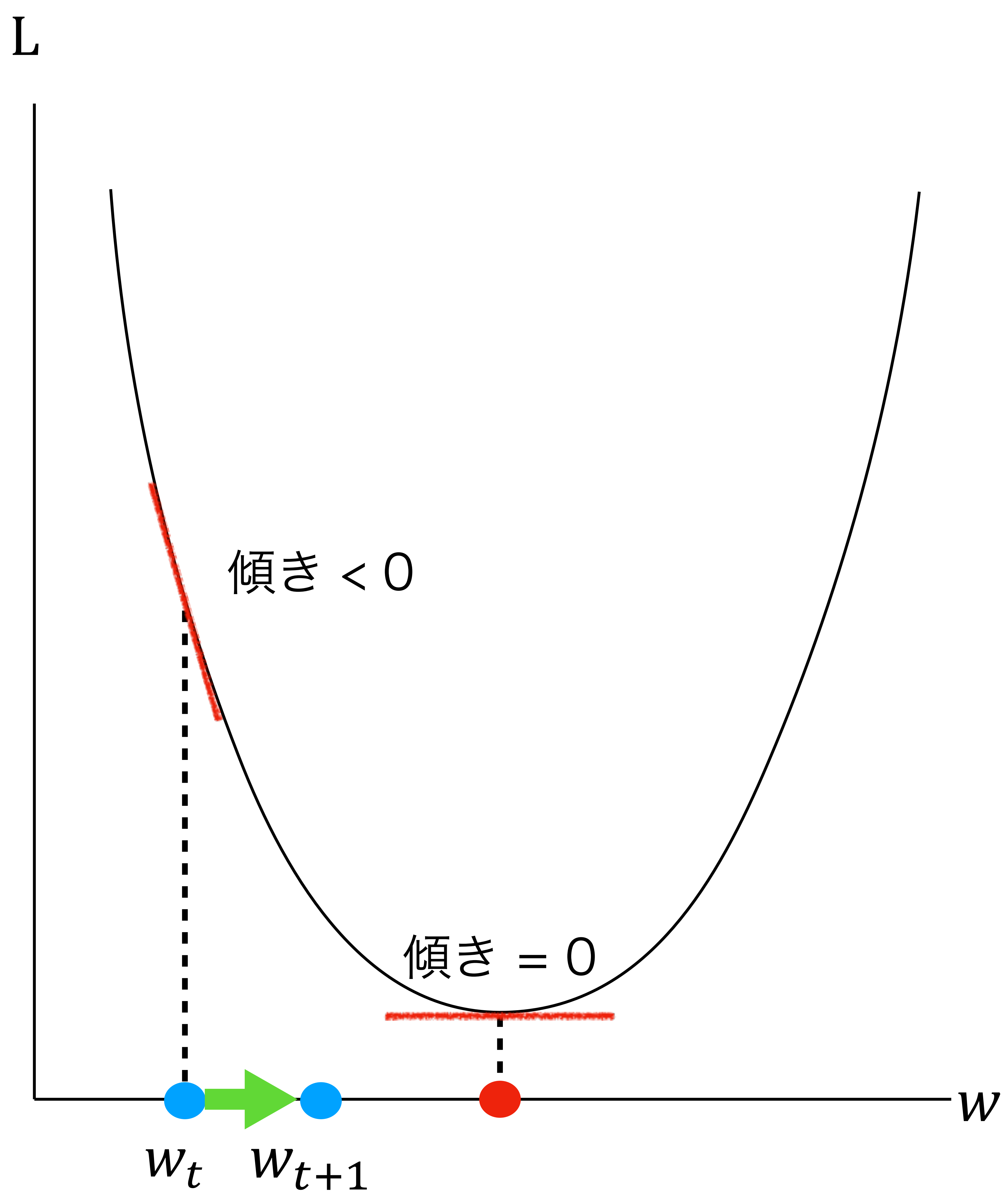
$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

（αは定数 α＝0.001など）

負

正

勾配降下法



傾きが負でも同様

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

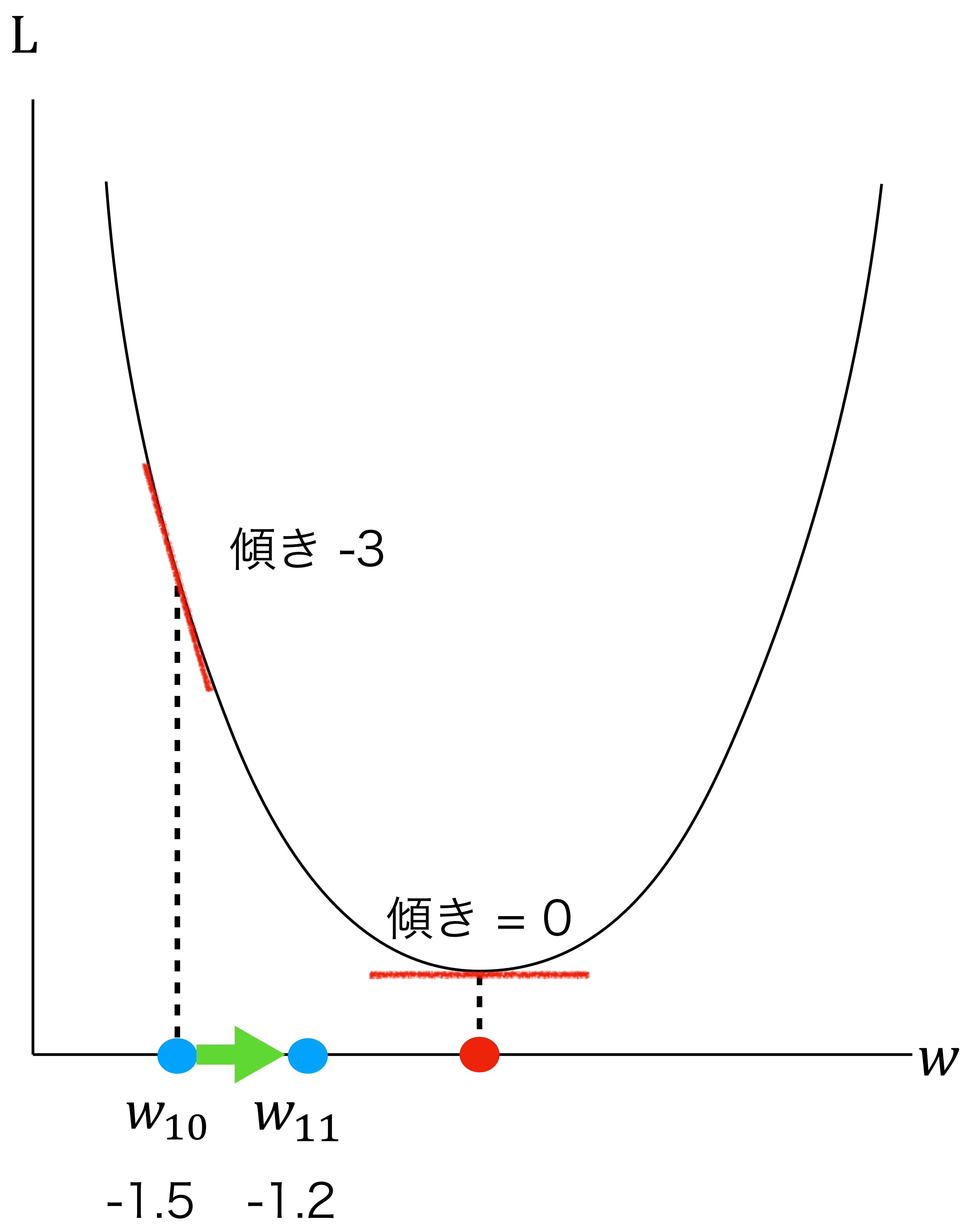
(αは定数 α＝0.001など)

負

正

w_{t+1} **はLが小さくなる方向に更新される**

勾配降下法



傾きが負でも同様

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

（αは定数 α＝0.001など）

負

正

w_{t+1} はLが小さくなる方向に更新される

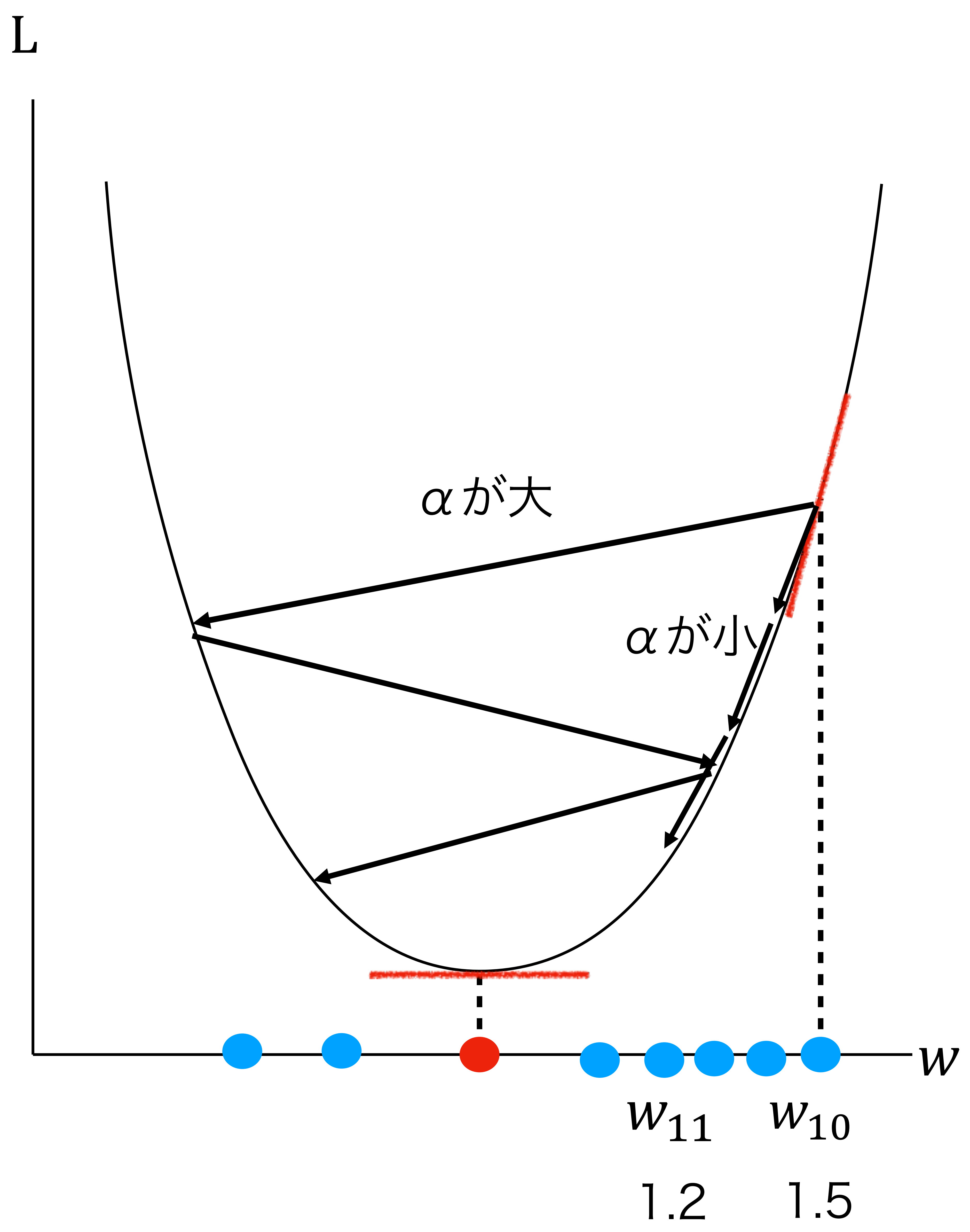
例えばepoch10回目の重みw10が-1.5、
その時の傾きを-3とすると、

$$w_{11} = w_{10} - 0.1 \times \frac{\partial L}{\partial w_{10}}$$

$$= -1.5 - 0.1 \times (-3)$$

$$= -1.2$$

勾配降下法



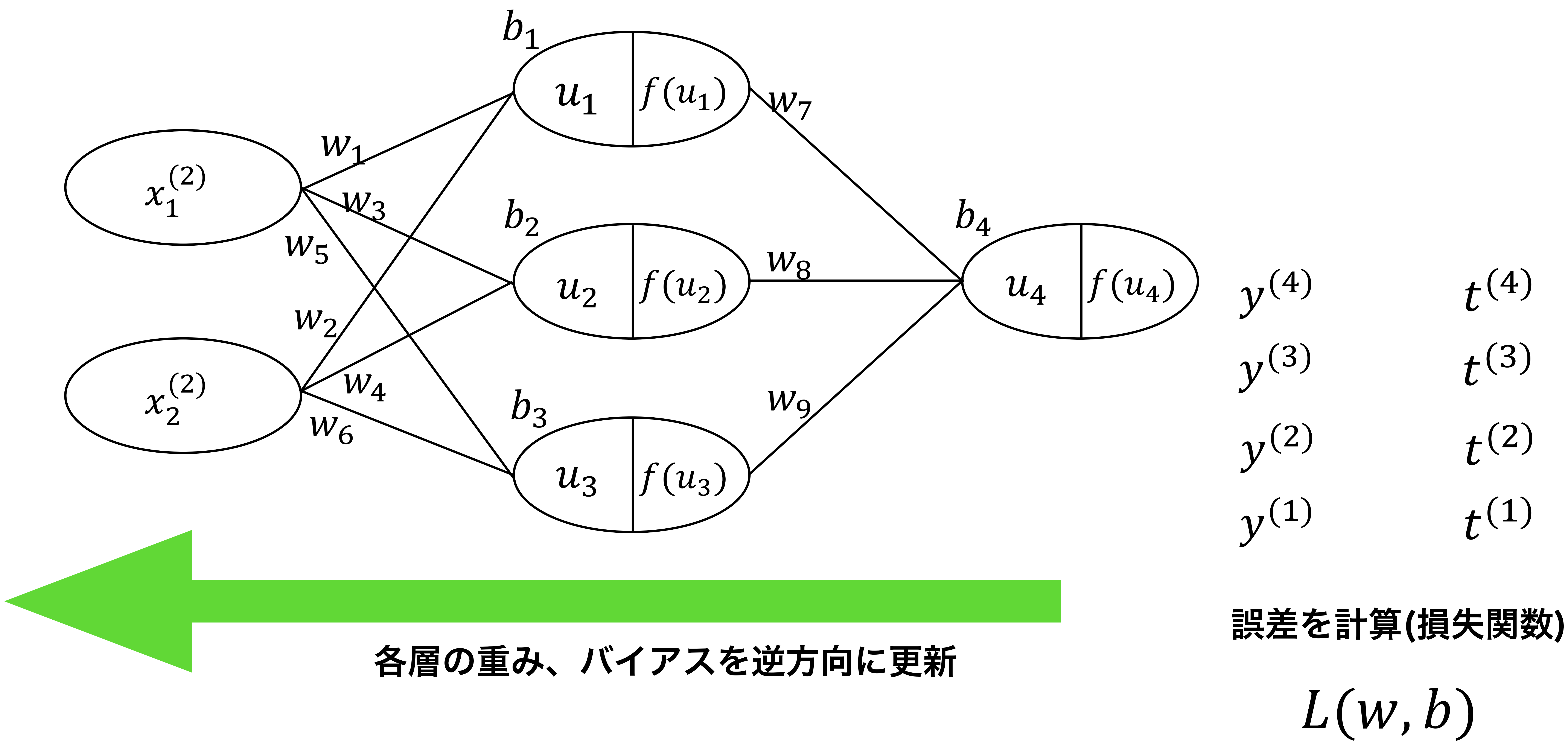
例えばepoch10回目の重みw10が1.5、
その時の傾きを3とすると、

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t} \quad (\alpha = 0.1)$$

$$\begin{aligned} w_{11} &= w_{10} - 0.1 \times \frac{\partial L}{\partial w_{10}} \\ &= 1.5 - 0.1 \times 3 \\ &= 1.2 \end{aligned}$$

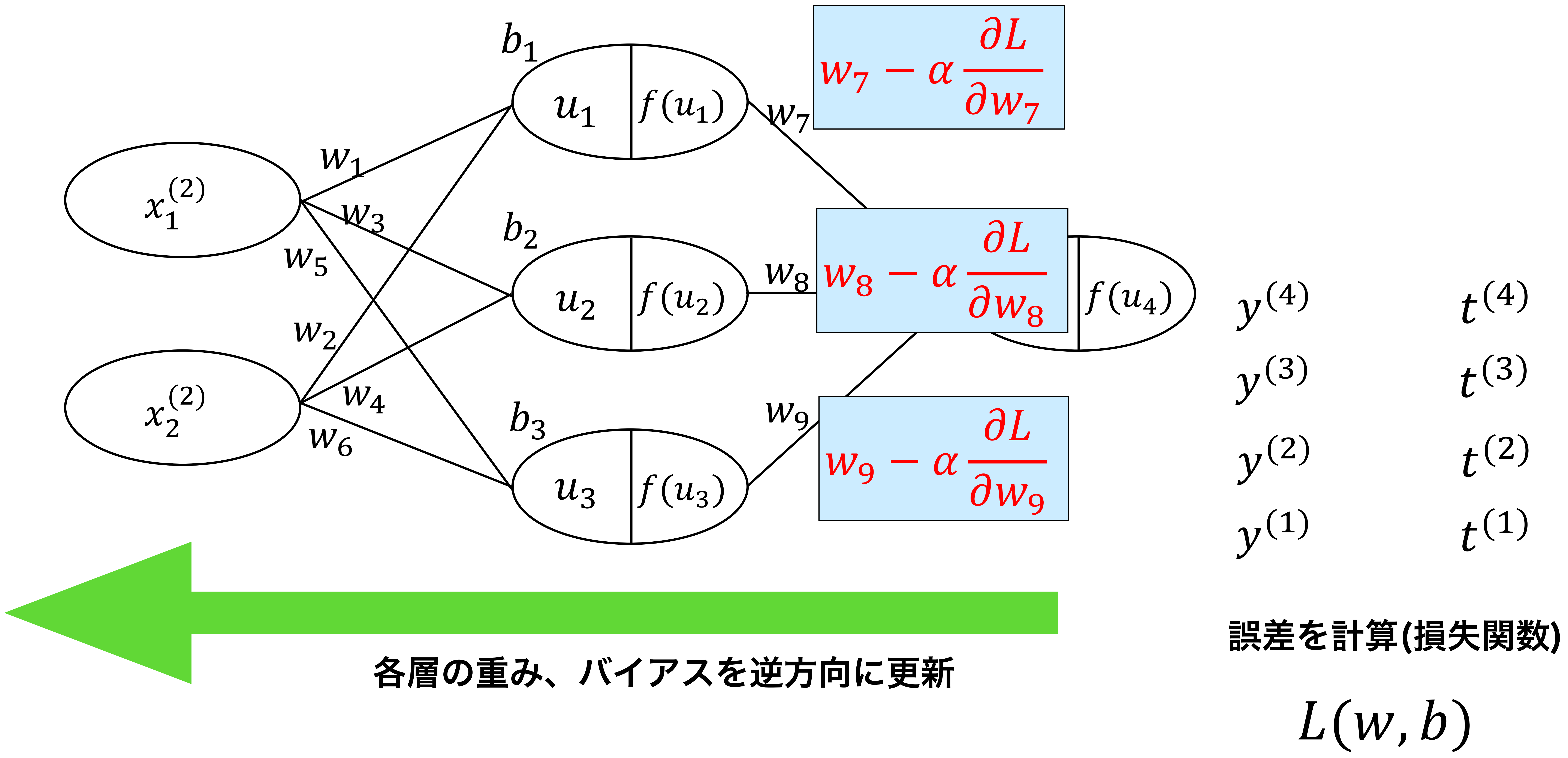
学習率αは１回にどれだけ更新するかを決める
学習率が高すぎると収束しにくく、
学習率が低すぎると収束が遅くなる

学習 ≒ 誤差逆伝播法



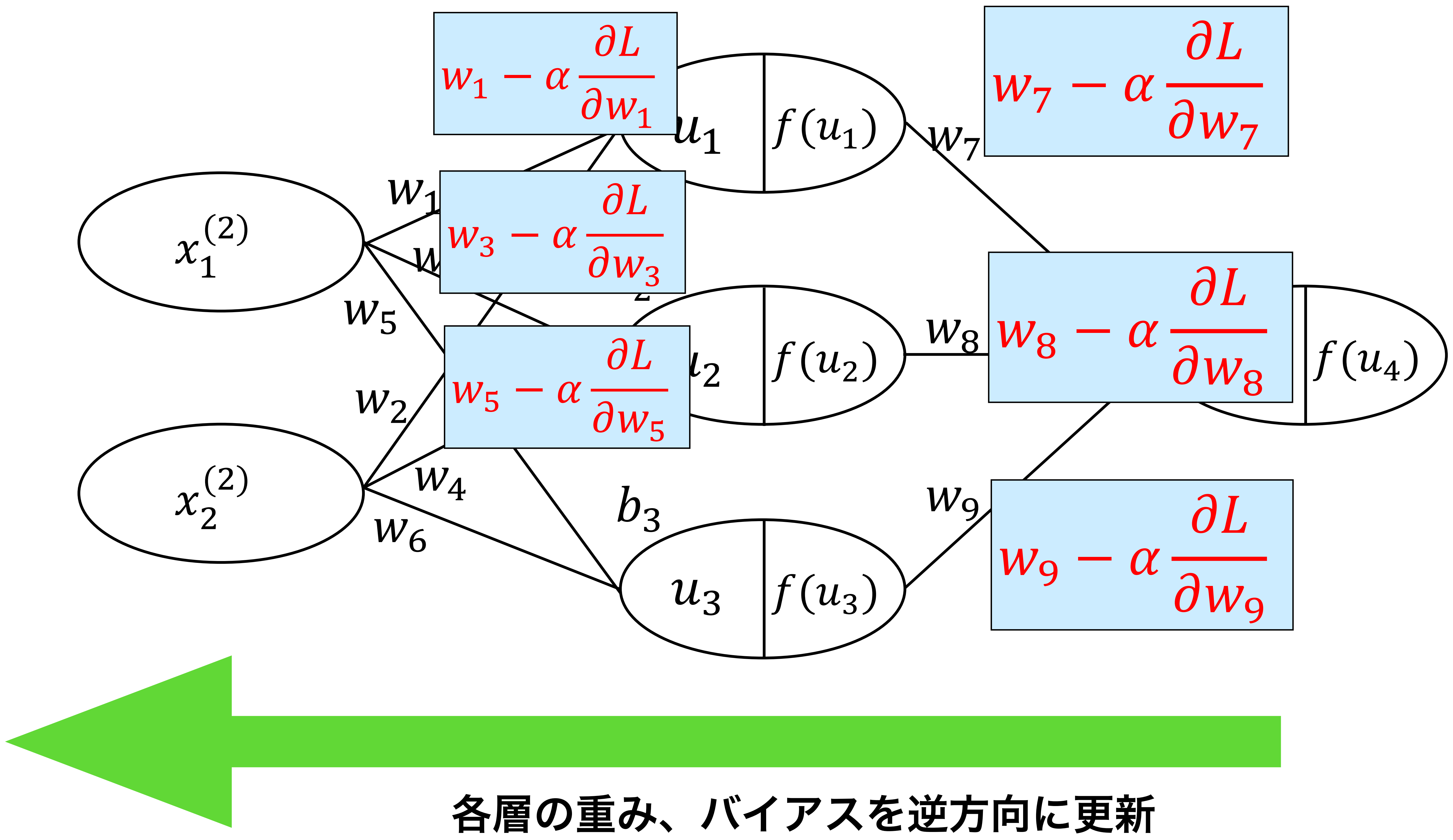
実際に各重みとバイアスを更新する際は、
出力層→中間層→入力層の順に計算することで求めることができます
推論と異なり逆方向に値を求めていくため、この学習の過程を誤差逆伝播法と言います。

学習 ≒ 誤差逆伝播法



実際に各重みとバイアスを更新する際は、
出力層→中間層→入力層の順に計算することで求めることができます
推論と異なり逆方法に値を求めていくため、この学習の過程を誤差逆伝播法と言います。

学習 ≒ 誤差逆伝播法



この偏微分には合成関数の
微分(連鎖律)などが必要
今回は割愛

$y^{(4)}$ $t^{(4)}$

$y^{(3)}$ $t^{(3)}$

$y^{(2)}$ $t^{(2)}$

$y^{(1)}$ $t^{(1)}$

誤差を計算(損失関数)

$$L(w,b)$$

実際に各重みとバイアスを更新する際は、
出力層→中間層→入力層の順に計算することで求めることができます
推論と異なり逆方法に値を求めていくため、この学習の過程を誤差逆伝播法と言います。

```
model.compile(loss='categorical_crossentropy',optimizer='Adam',metrics=['accuracy'])
```

損失関数

交差エントロピー誤差

$$L = -\sum_{i=1}^{n} t_k \log(y_k)$$

最適化アルゴリズム

勾配降下法(SGD)

$$w \leftarrow w - \alpha \frac{\partial L}{\partial w} \quad b \leftarrow b - \alpha \frac{\partial L}{\partial b}$$

Adamはこれの派生系

課題

・WebClassにある課題6をやりましょう

締め切りは1週間後の5/23の23:59です。
締め切りを過ぎた課題は受け取らないので注意して下さい。
(1週間後に正解をアップします)