研究室ローテーション 第２回

担当：和田山 正・中井彩乃

本日の学習目標

- 深層学習の概要
- 勾配法によるパラメータ調整
- Google Colab(演習課題)

深層学習の概要

イヌ・ネコ分類問題

こんな関数を作りたい！

・入力された画像について、写っているのがイヌなのかネコなのか判別する関数を構成したい (クラス分類問題)

$f : \mathbb{R}^n \to \mathbb{R}$

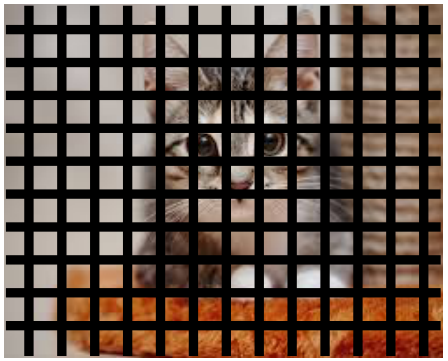
0: 猫

$f : \mathbb{R}^n \to \mathbb{R}$

1 : 犬

イヌ・ネコ分類問題

$$f : \mathbb{R}^n \to \mathbb{R}$$



・ピクセルに分割

・例ではカラーだけど白黒

各ピクセルは実数だと思う

・一次元に並べ替える

・n個の実数の組として、画像が表現される

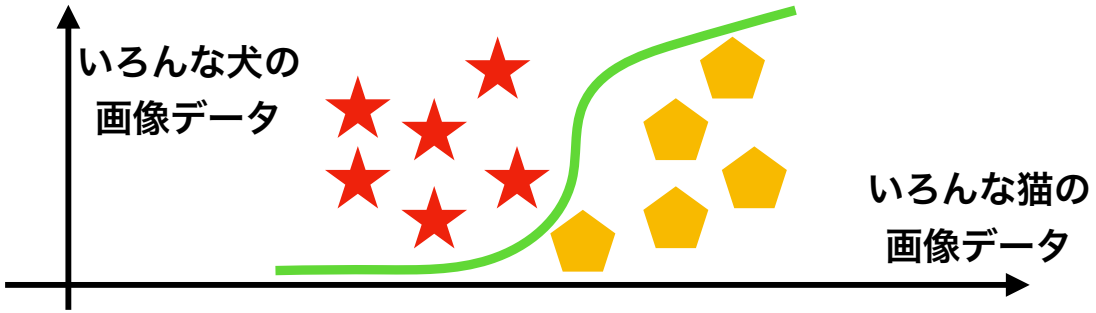
$$(x_1, x_2, \ldots, x_n) \to f(x_1, x_2, \ldots, x_n)$$

・関数fを適用する

・0 または 1 の値が返る

イヌ・ネコ分類問題

こんな
イメージ



うまくデータを
分ける関数の形
を作りたい

$f : \mathbb{R}^n \to \mathbb{R}$

0: 猫

$f : \mathbb{R}^n \to \mathbb{R}$

1 : 犬

関数の形状を制御するパラメータを導入する

パラメトリックモデル

$$f_\Theta : \mathbb{R}^n \to \mathbb{R}$$

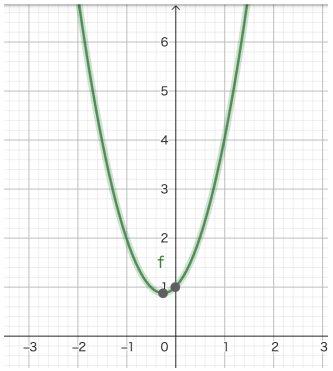
Θ：関数の形状をコントロールするパラメータ群

例：2次関数に基づくパラメトリックモデル

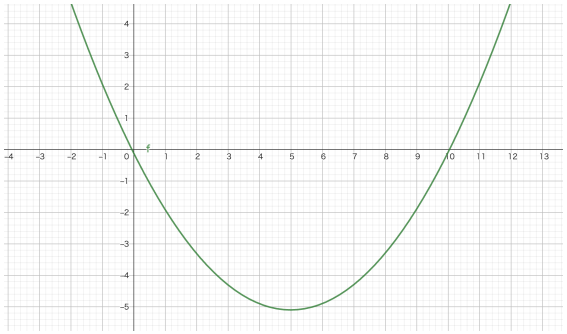
$$f_\Theta(x) = ax^2 + bx + c$$

$$\Theta = \{a, b, c\}$$

2次関数でいうa, b, cがパラメータ.
値を変えることで関数の形が変わる.



$\{2,1,1\}$



$\{0.2, -2, -0.1\}$

訓練データを準備する

学習パラメータをチューニング（調整）するために訓練データを準備する

入力値　　教師ラベル

{  , 1}　{  , 1}　{  , 1}

{  , 0}　{  , 0}　{  , 0}

- 画像はn次元のベクトルとして表現しておく
- 良い認識結果を得るには、一般に多数の訓練データが必要

パラメータを更新する(訓練・学習)

出力と教師ラベルとの間の食い違いがなるべく小さくなるように学習パラメータを更新する

$f_\Theta : \mathbb{R}^n \to \mathbb{R}$

近い値？

$\hat{y}$ y

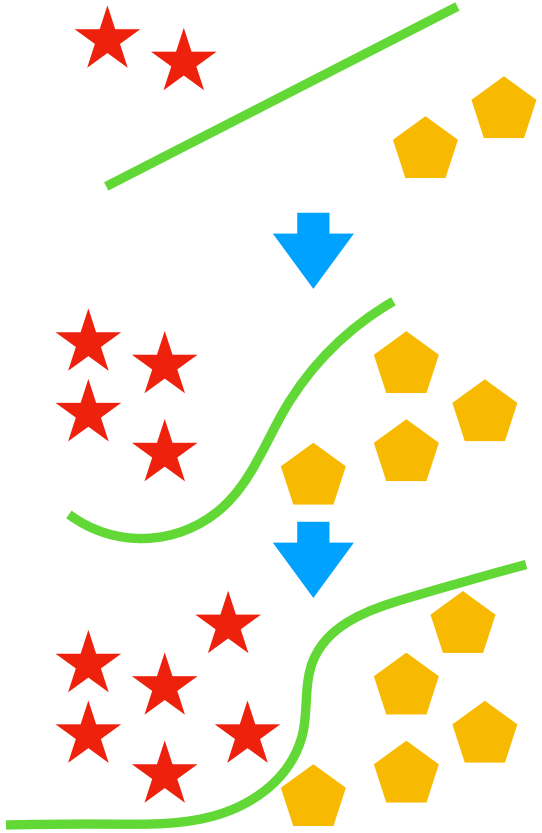
{ , 1}

{ , 1}

{ , 0}

大量の訓練データを与える

データを与えて
関数の形を変えていく



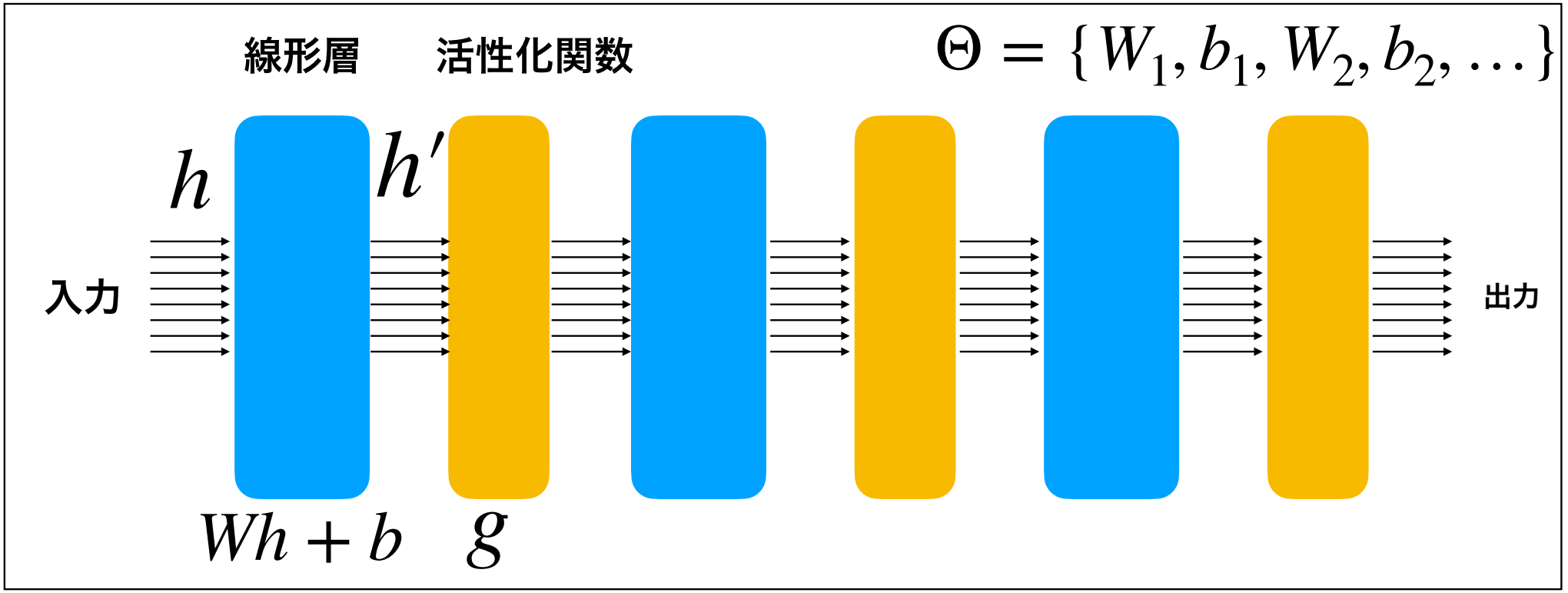
深層ニューラルネットワークモデル

：柔軟性が高い関数の形状.

以下のように関数を順番に合成する

$$f_\Theta : \mathbb{R}^n \to \mathbb{R}^m$$

$$\Theta = \{W_1, b_1, W_2, b_2, \ldots\}$$



ネットワークという名前だがただの（合成）関数，ただし複雑な形状

アフィン変換と活性化関数

アフィン変換
(W,b: 学習パラメータ)

出力 h' = 重み行列 W 入力 h + バイアス b

活性化関数
：複雑な構造
を生み出す

シグモイド関数

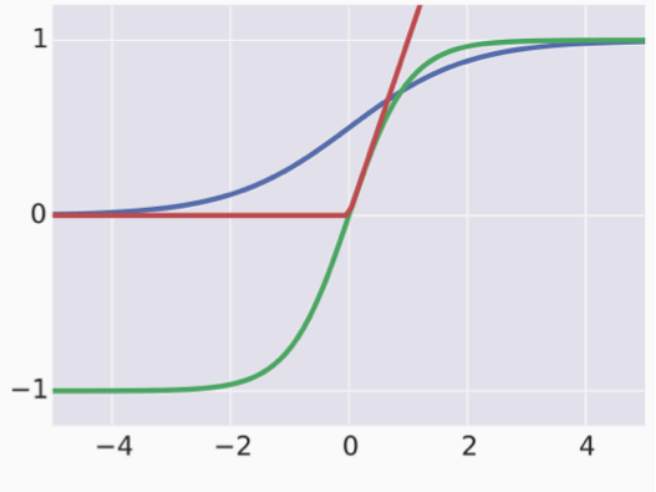
$$\sigma(u) = \frac{1}{1 + e^{-u}}$$

tanh関数

$$\tanh(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}}$$

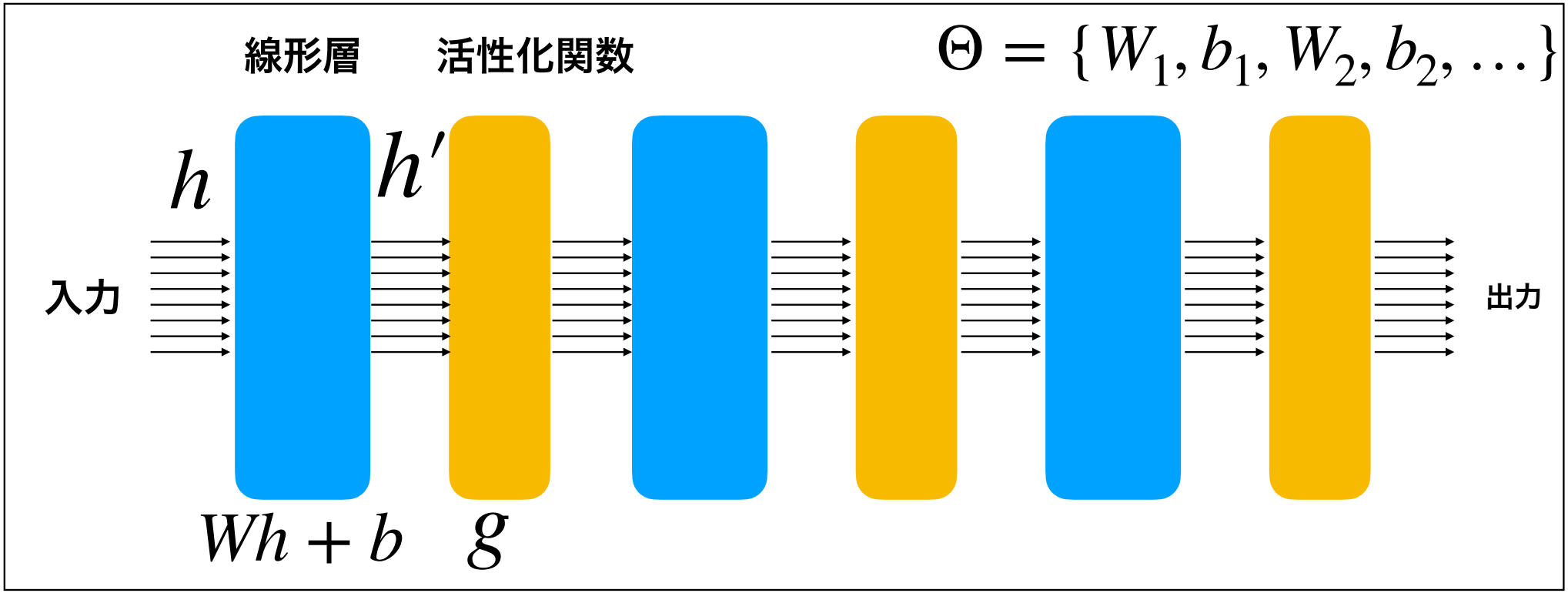
ReLU関数

$$\mathrm{ReLU}(u) = \max(0, u)$$



深層ニューラルネットワークモデル

$$f_\Theta : \mathbb{R}^n \to \mathbb{R}^m$$

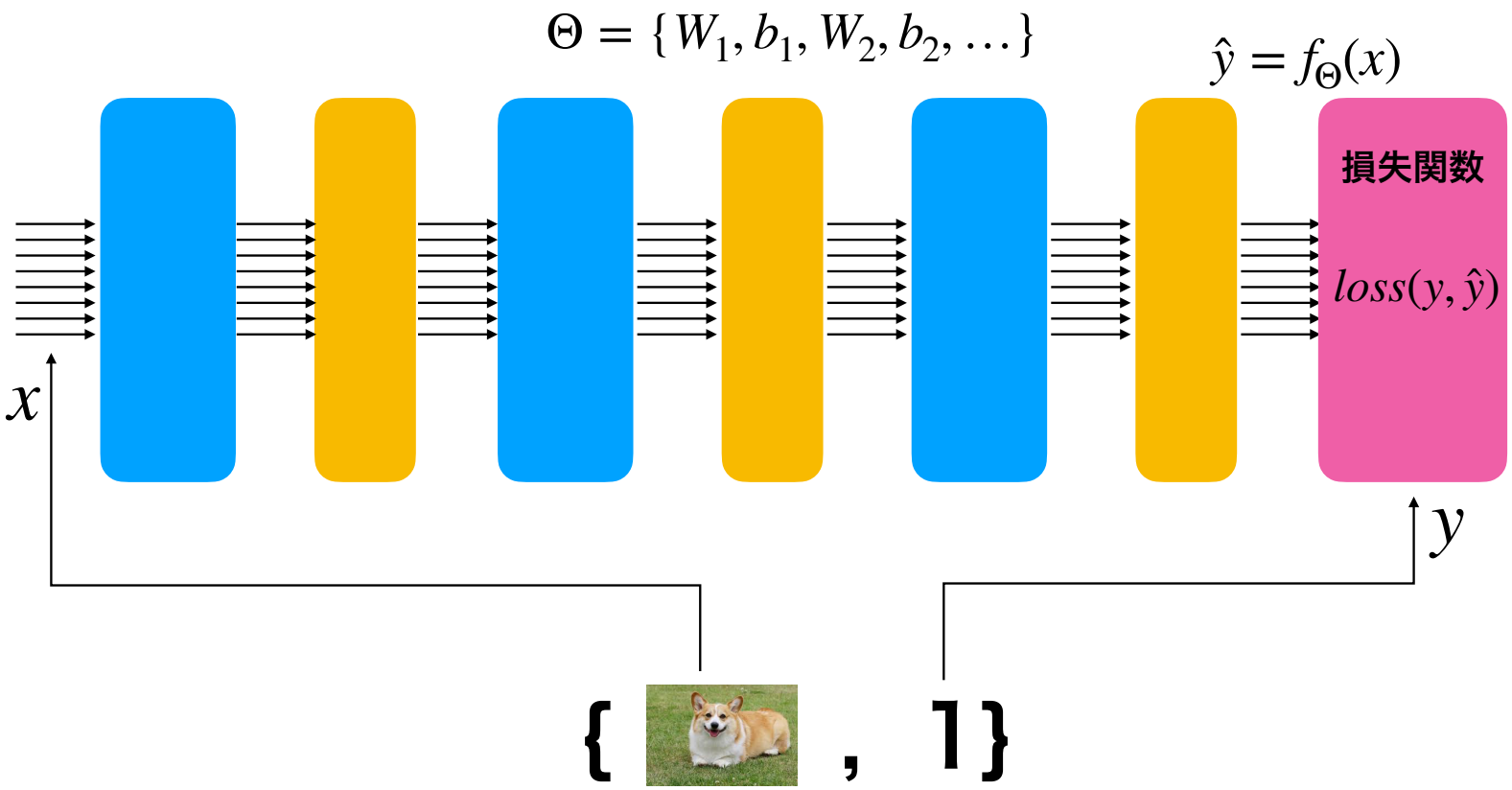


アフィン変換と活性化関数の合成関数で複雑な形状をつくる

学習プロセス

- ニューラルネットワークなどのモデルの学習 = 学習パラメータの調整(最適化)
- モデル出力と教師信号の食い違い = 損失関数 (二乗誤差関数, クロスエントロピーなど)
- 学習プロセス = 多数の訓練データに基づく確率的勾配法による損失関数値の最小化

深層ネットワークの訓練



訓練・学習過程では，損失関数値を最小化するように
学習パラメータを最適化（調整）する。

損失関数

入力信号　x

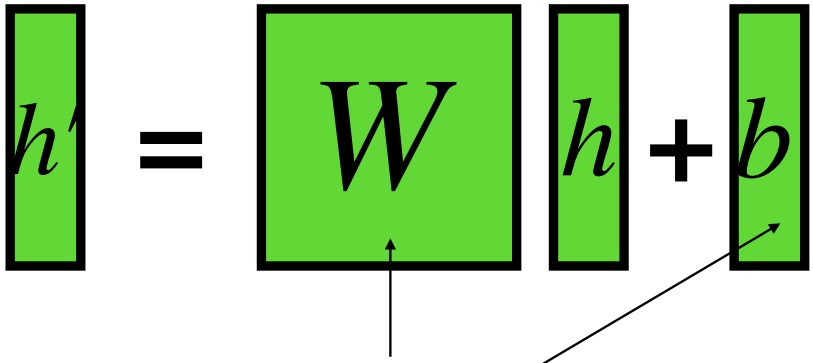
推定出力　$\hat{y} = f_\Theta(x)$

教師信号　y

二乗誤差関数(典型的な損失関数)

$$loss(y, \hat{y}) = \frac{1}{2} \sum_{i=1}^{n} |y_i - \hat{y}_i|^2$$

学習プロセス



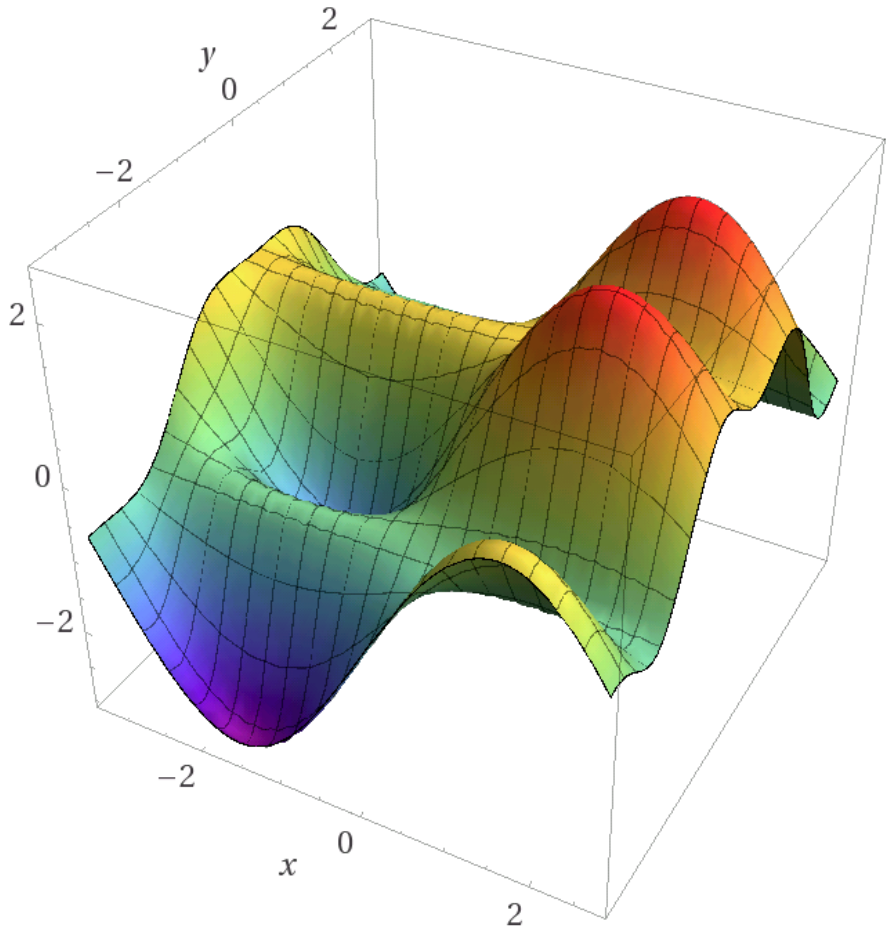
二乗誤差関数を最小化するように学習パラメータを動かせばよい

$$loss(y, \hat{y}) = \frac{1}{2} \sum_{i=1}^{n} |y_i - \hat{y}_i|^2$$

勾配法による数値最小化

学習プロセス=学習パラメータの調整

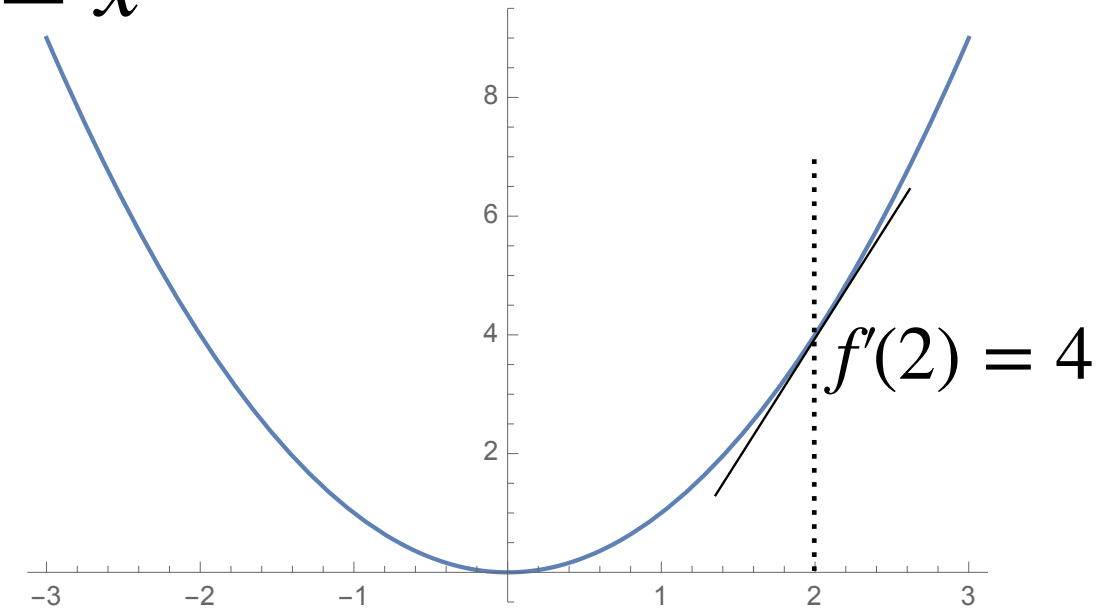
誤差関数は一般に
でこぼこな形状



どうやって誤差を最小にすればよいの？

微積の力を借りる！

$y = x^2$

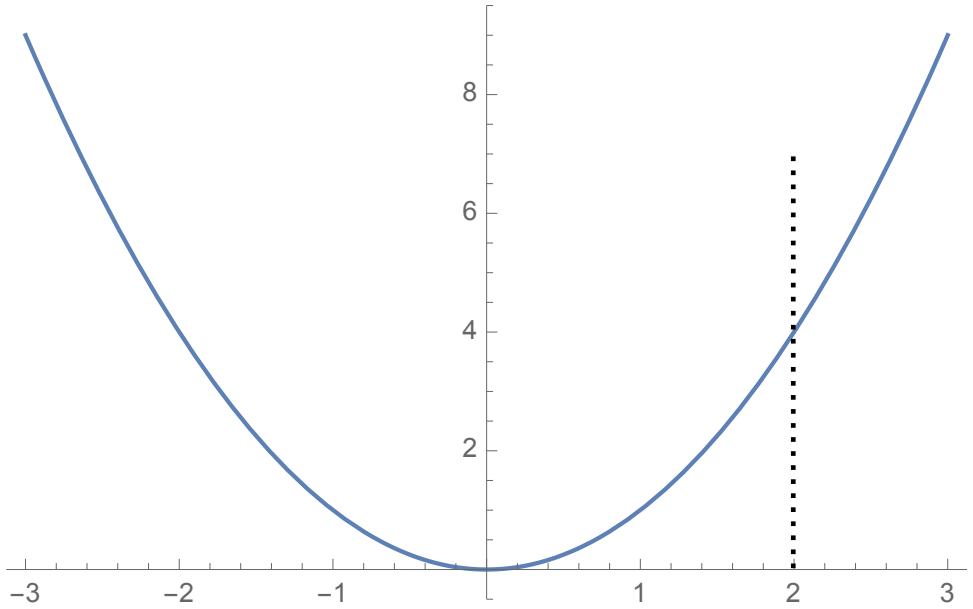


導関数 $\dfrac{dy}{dx} = f'(x) = 2x$

勾配法(数値的最小化法)

ステップ１：s を初期値にセットする
ステップ２：g := f'(s) とする
ステップ３：s := s - a g とする（a は学習係数と呼ばれる）
ステップ４：ステップ２に戻る

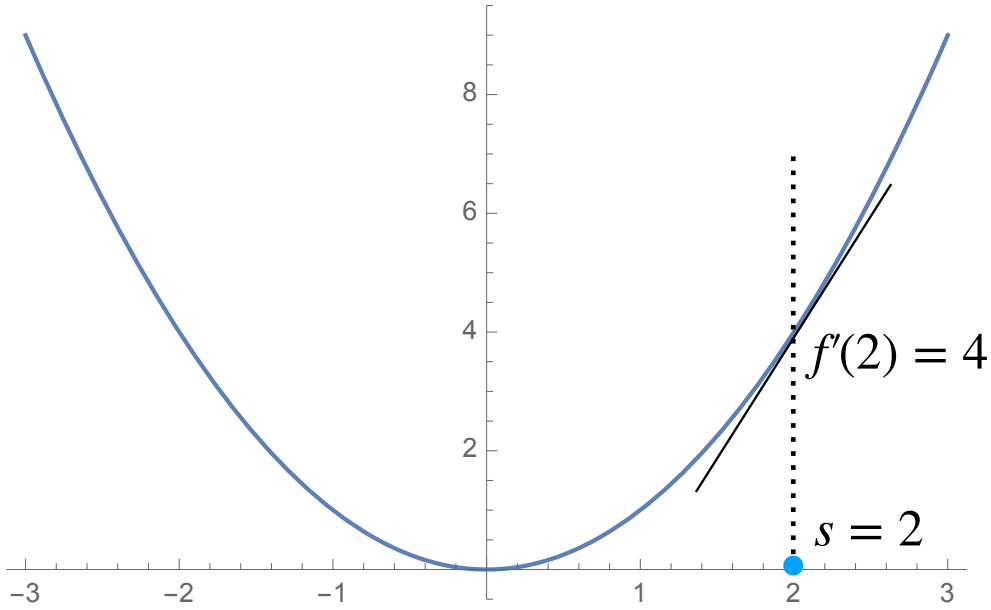
[練習問題] 初期値s = 2, a = 0.2 として上の勾配法のステップを実行せよ。



勾配法(数値的最小化法)

ステップ１：s を初期値にセットする
ステップ２：g := f'(s) とする
ステップ３：s := s - a g とする（a は学習係数と呼ばれる）
ステップ４：ステップ２に戻る

[練習問題] 初期値s = 2, a = 0.2 として上の勾配法のステップを実行せよ。

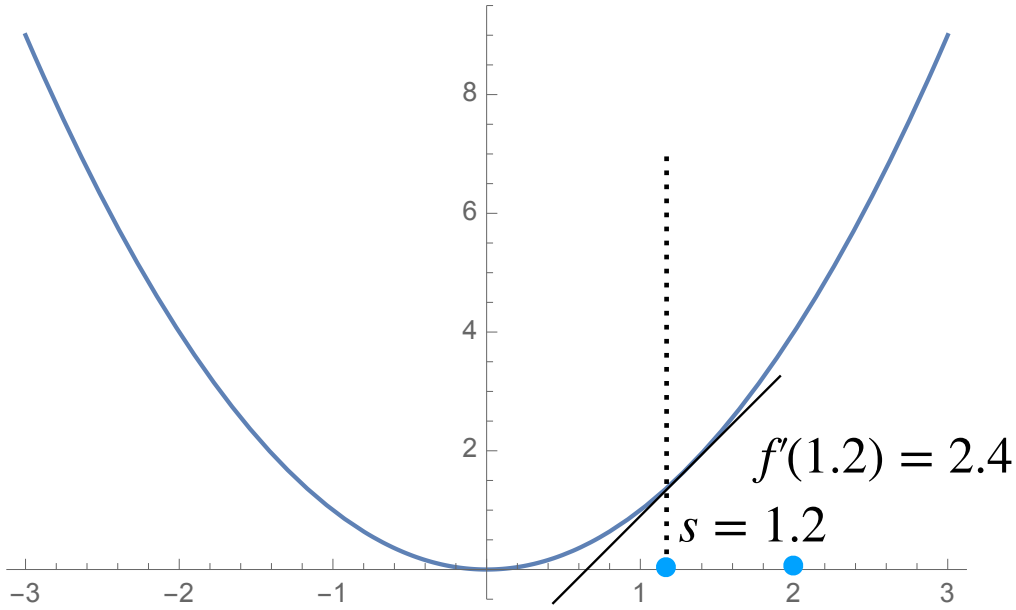


初期値をセット
勾配（微分）を計算

勾配法(数値的最小化法)

ステップ１：s を初期値にセットする
ステップ２：g := f'(s) とする
ステップ３：s := s - a g とする（a は学習係数と呼ばれる）
ステップ４：ステップ２に戻る

[練習問題] 初期値s = 2, a = 0.2 として上の勾配法のステップを実行せよ。

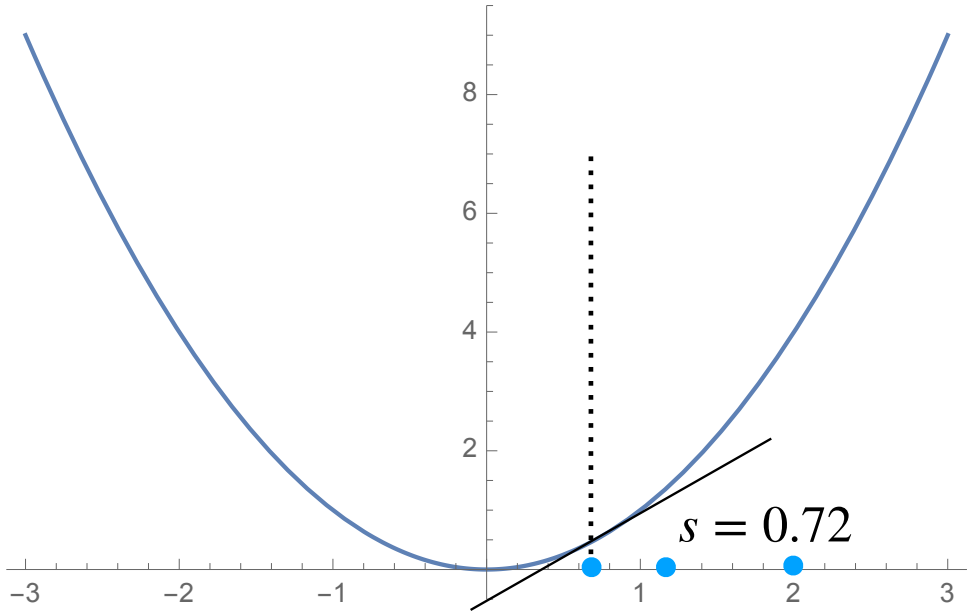


sを更新
勾配（微分）を計算

勾配法(数値的最小化法)

ステップ１：s を初期値にセットする
ステップ２：g := f'(s) とする
ステップ３：s := s - a g とする（a は学習係数と呼ばれる）
ステップ４：ステップ２に戻る

[練習問題] 初期値s = 2, a = 0.2 として上の勾配法のステップを実行せよ。

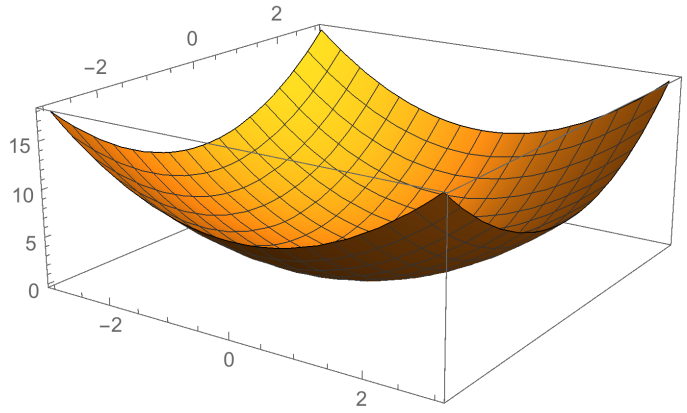


繰り返すと
極小値（この場合s=0）
を探すことができる

多次元の関数の場合には？

深層学習の場合、パラメータは複数ある
（場合によっては数万パラメータにも及ぶ）

多次元の場合もそれぞれの変数に関する
微分を使用すればよい.



$$f(x_1, x_2) = x_1^2 + x_2^2$$

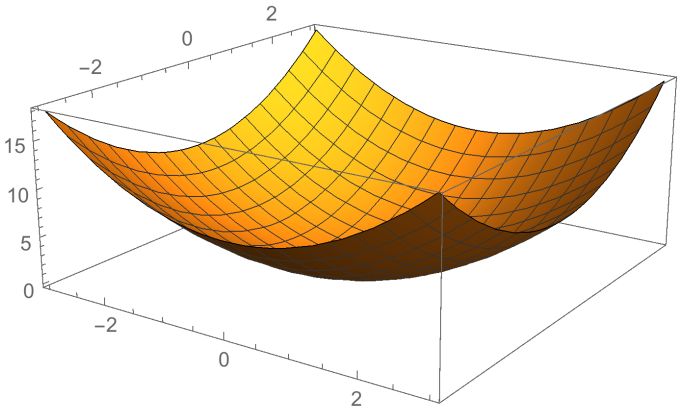
勾配ベクトル

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1} = 2x_1, \frac{\partial f}{\partial x_2} = 2x_2 \right)^T$$

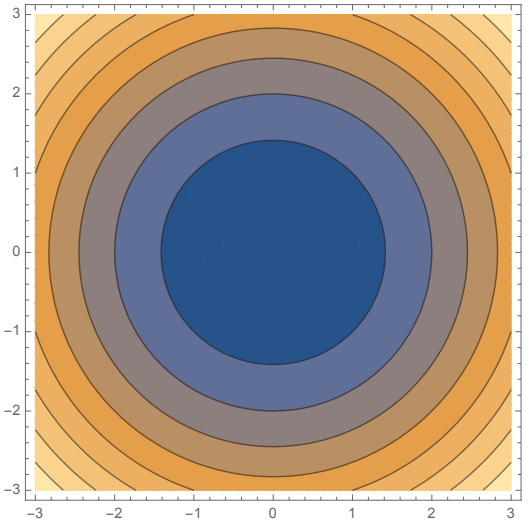
偏微分：多変数のうちの一変数に関する微分

勾配ベクトルと等高線

$$f(x_1, x_2) = x_1^2 + x_2^2$$



3D表示



等高線表示

勾配ベクトル

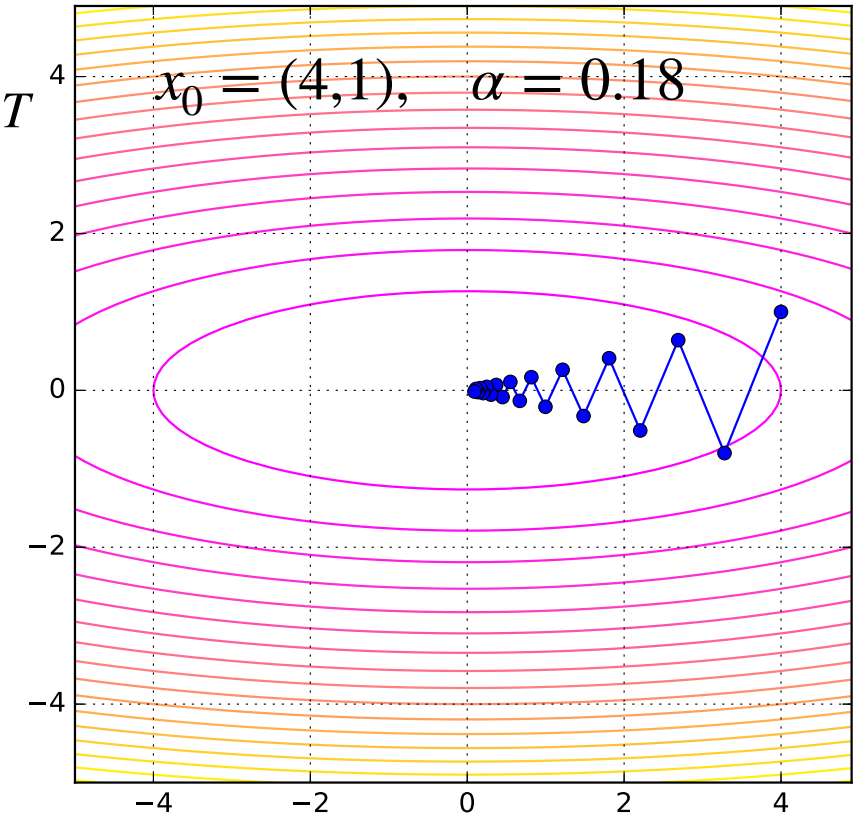
$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1} = 2x_1, \frac{\partial f}{\partial x_2} = 2x_2 \right)^T$$

２次元関数における勾配法

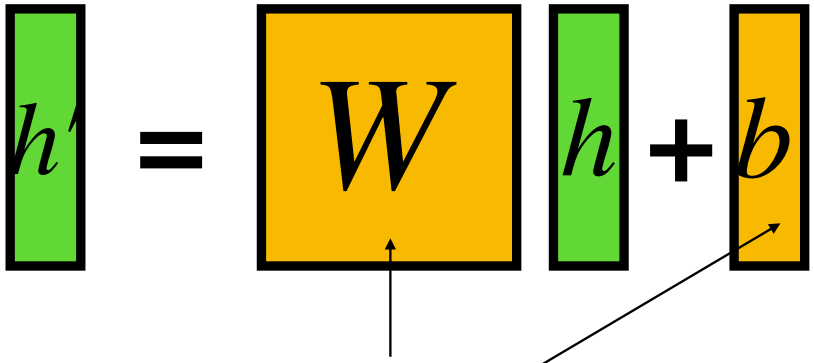
$$f(x_1, x_2) = \frac{1}{2}x_1^2 + 5x_2^2$$

$$\nabla f(x_1, x_2) = \left(\frac{\partial f}{\partial x_1} = x_1, \frac{\partial f}{\partial x_2} = 10x_2 \right)^T$$

勾配法： $x_t = x_{t-1} - \alpha \nabla f(x_{t-1})$



ふたたび：学習プロセス



中身は動かしてよい・うまく
チューンアップしたい！

二乗誤差関数を最小化するように学習パラメータを動かせばよい

→W, bの各要素の偏微分(勾配ベクトル)を求めて、勾配法
を使えばよい！

まとめ

- 深層学習では、層構造を持つパラメトリック関数モデルが利用されている
- 各層は、行列ベクトル積計算（アフィン変換）と非線形関数の要素ごとの適用からなる
- 所望の性能を出すためには、学習（訓練）プロセスが必要
- 学習プロセスでは、大量の訓練データが必要
- 学習プロセスでは勾配法(本当は確率的勾配法)を利用して学習パラメータを調節する