

# **知能工学特別講義**

**担当：和田山 正**

**名古屋工業大学**

# 本講義の学習目標

- 深層学習とスパースモデリングに関する基礎と実践を学ぶ
- 全8コマ
- オンライン講義 + Google Colab(演習課題)
- Pytorch のコードを動かして理解を深める
- キーワード: ニューラルネットワーク, 勾配法, 誤差逆伝播法, クラス分類問題, 回帰問題, スパースモデリング, 圧縮センシング

# 講義の進め方(1)

9/21 3コマ

1 深層学習入門

9/21 4コマ

2 ニューラルネットワーク学習

9/21 5コマ

3 パターン認識問題

9/22 3コマ

4 回帰問題

9/22 4コマ

5 圧縮センシング入門

9/22 5コマ

6 ニューラルネットで信号処理 (1)

9/24 3コマ

7 ニューラルネットで信号処理 (2)

9/24 4コマ

8 予備時間

# 講義の進め方(2)

60 ~ 70分

スライドによる内容の説明

20 ~ 30分

Google colabによる演習

- 演習課題で時間中に出来なかつたものは、可能であれば23日、または集中講義後に自分でやってもらえると理解が進みます。

# 成績評価

- ・レポート問題を最後に出します。
- ・そのレポートに基づき成績評価を行います。

# 本日の内容

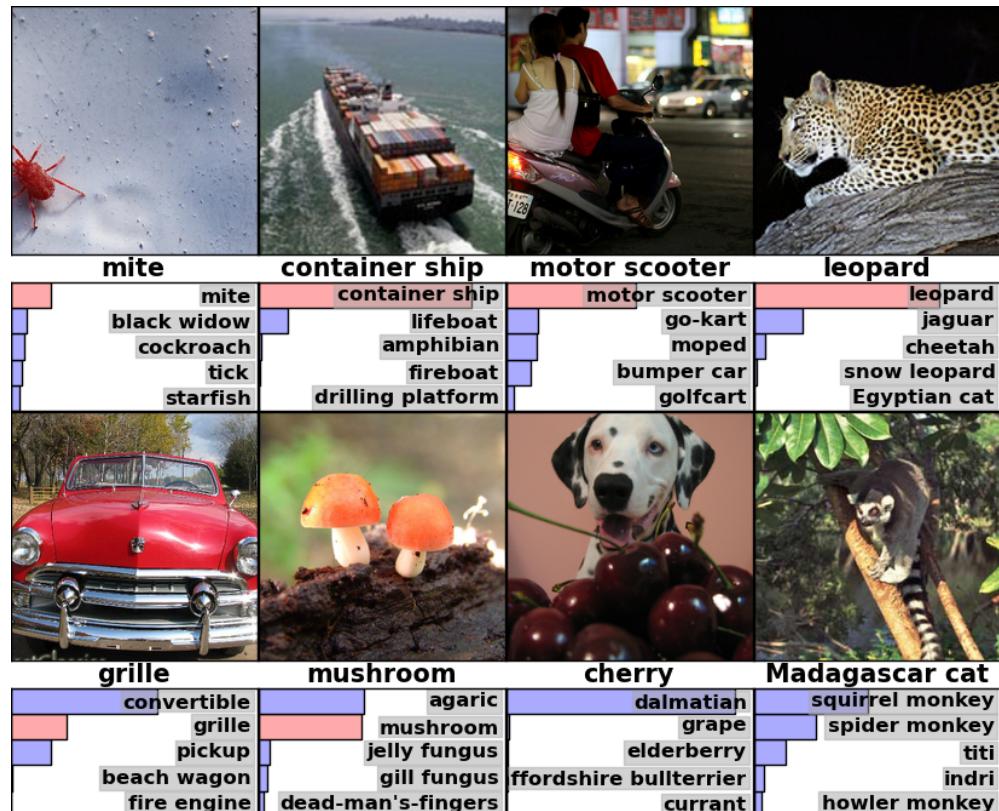
- 深層学習の概要
- 勾配法による最小化
- Google Colabを使ってみる

# 深層学習技術の進展

- 画像認識
- 音声認識
- 自然言語処理
- 機械翻訳

深層学習技術は、これらの分野において特に圧倒的な強みを見せている

ImageNet Classification



cited from: ``ImageNet Classification with Deep Convolutional Neural Networks'', Alex Krizhevsky et al.

<https://www.nvidia.cn/content/tesla/pdf/machine-learning/imagenet-classification-with-deep-convolutional-nn.pdf>

# AlphaGOの登場(2016)

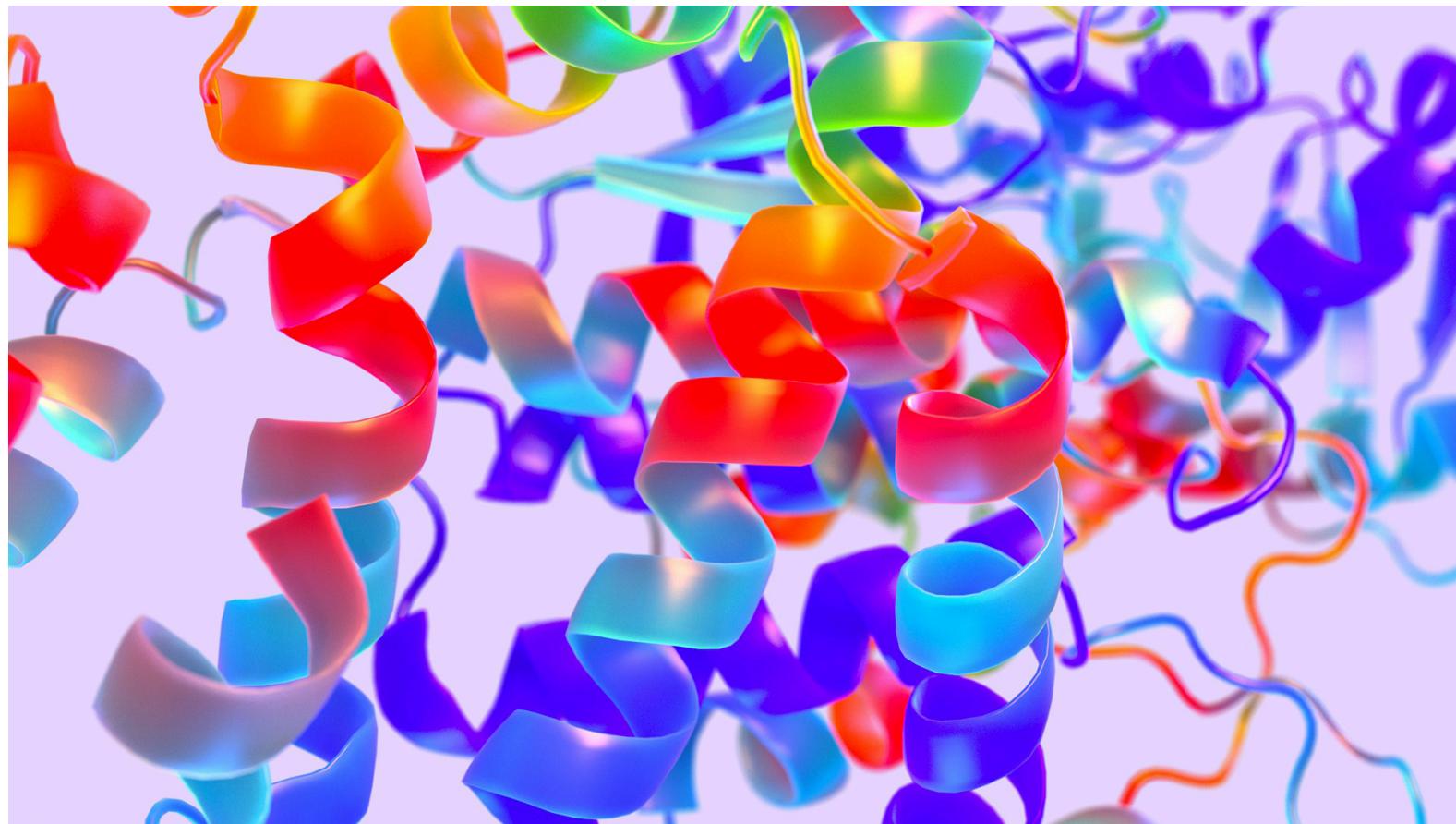


(左)2016年 AlphaGo と対戦したイ・セドル

(右)DeepMind のデミス・ハサビス

- ・モンテカルロ木探索 + 深層畳み込みネットワーク(ポリシーネットワーク)
- ・3週間50GPUを利用し、3億4000万回のトレーニング

# AlphaFold2(2021)

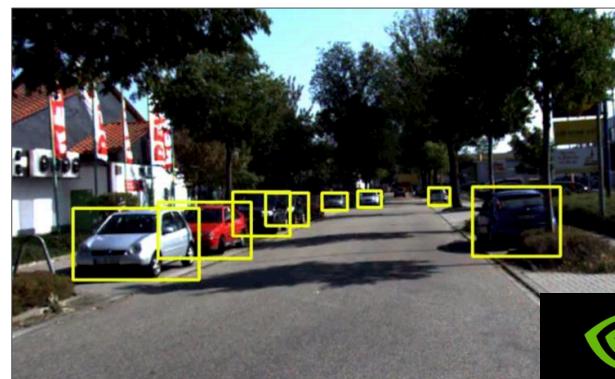
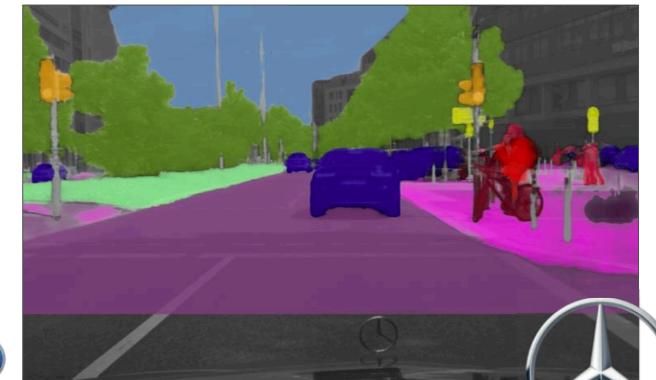


<https://deepmind.com/blog/article/AlphaFold-Using-AI-for-scientific-discovery>

# 自動運転技術の基盤技術としての深層学習



Audi



Audi

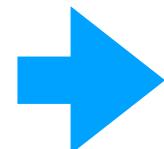
# 深層学習の概要



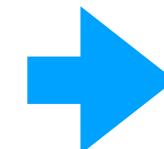
# イヌ・ネコ分類問題

こんな関数を作りたい！

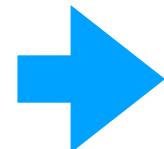
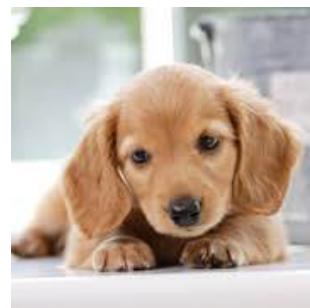
- ・入力された画像について、写っているのがイヌなのかネコなのか  
判別する関数を構成したい（クラス分類問題）



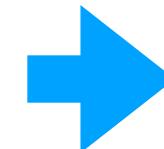
$$f: \mathbb{R}^n \rightarrow \mathbb{R}$$



0: 猫



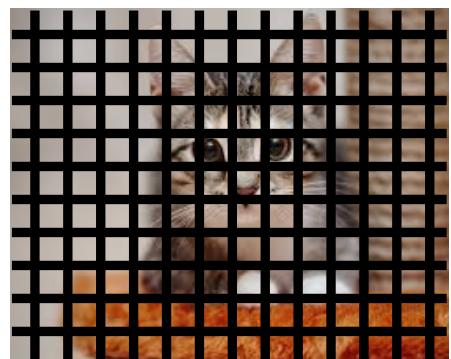
$$f: \mathbb{R}^n \rightarrow \mathbb{R}$$



1: 犬

# イヌ・ネコ分類問題

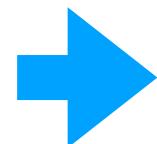
$$f: \mathbb{R}^n \rightarrow \mathbb{R}$$



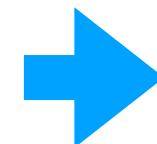
- ・一次元に並べ替える
  - ・ $n$ 個の実数の組として、画像が表現される
  - ・関数 $f$ を適用する
  - ・0 または 1 の値が返る
- ・ピクセルに分割
- ・例ではカラーだけど白黒  
各ピクセルは実数だと思う

# イヌ・ネコ分類問題

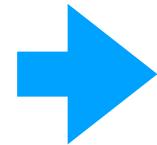
こんな関数を作りたい！



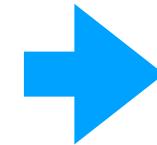
$$f: \mathbb{R}^n \rightarrow \mathbb{R}$$



0: 猫



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



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

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

$\Theta$ : 関数の形状をコントロールするパラメータ群

---

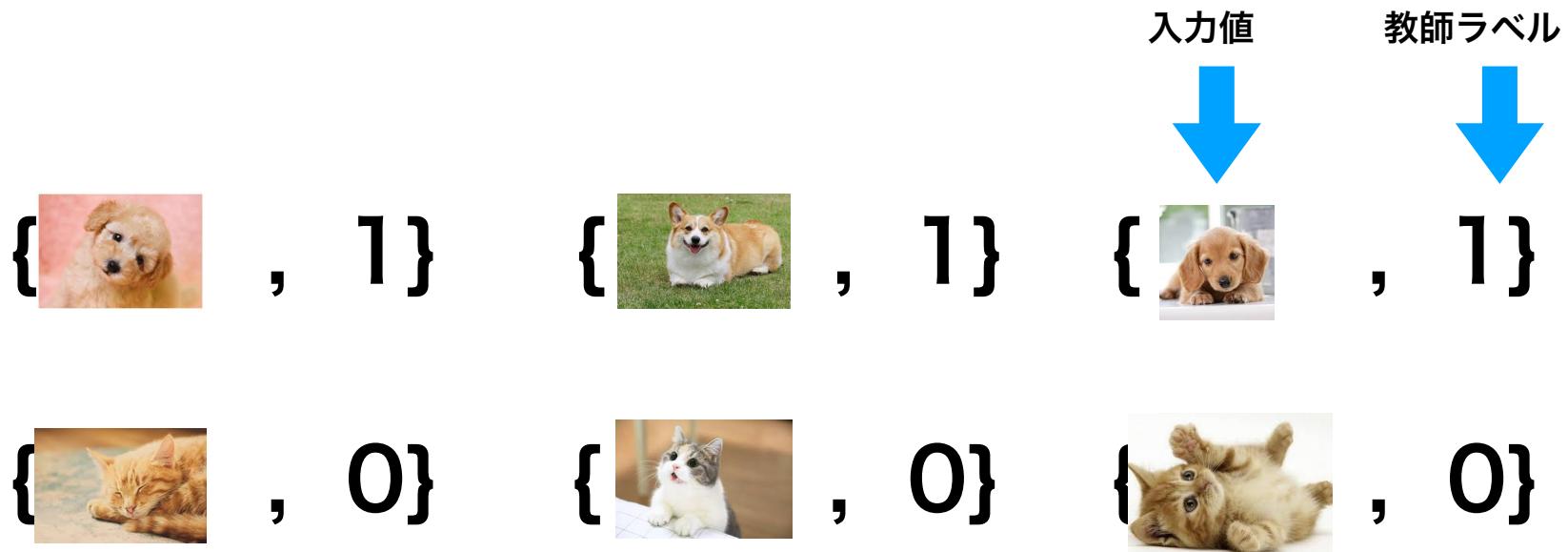
例：2次関数に基づくパラメトリックモデル

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

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

# 訓練データを準備する

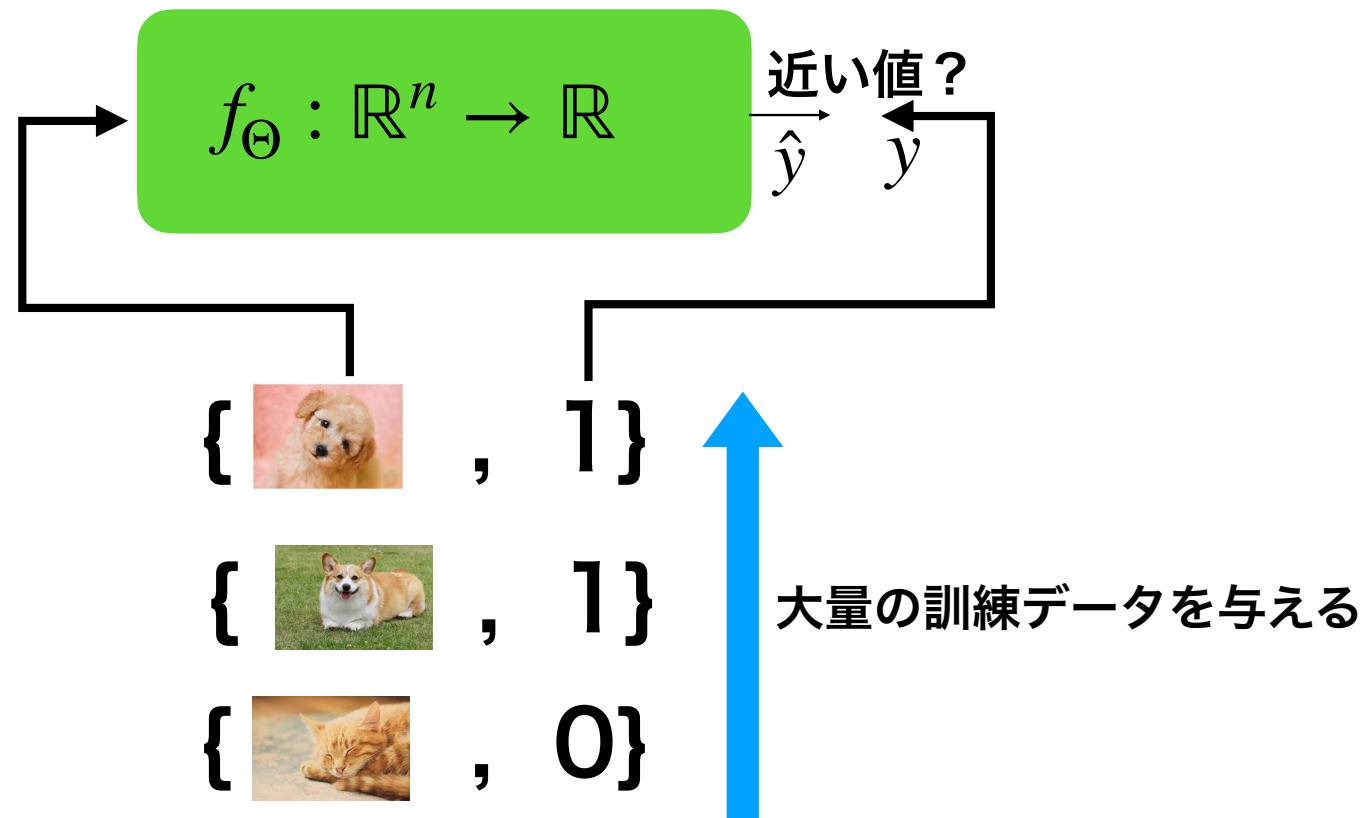
学習パラメータをチューニング（調整）するために訓練データを準備する



- ・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

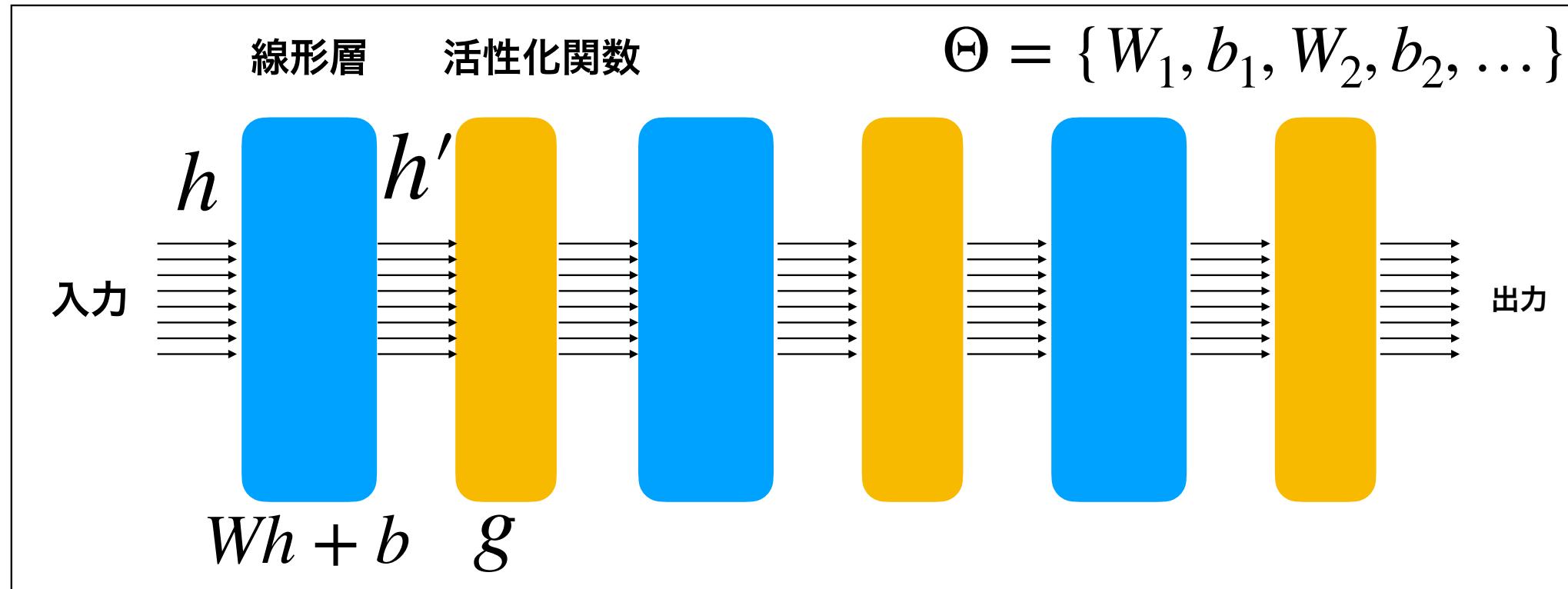
# パラメータを更新する(訓練・学習)

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



# 深層ニューラルネットワークモデル

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



# 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)

$$\text{出力 } h' = \text{重み行列 } W \cdot \text{入力 } h + \text{バイアス } b$$

活性化関数

シグモイド関数

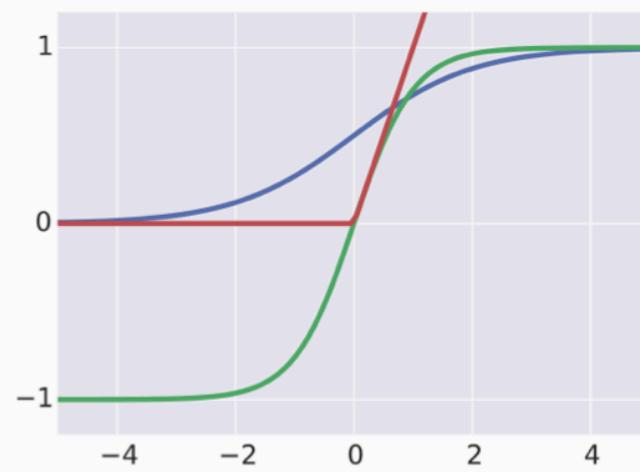
$$\sigma(u) = \frac{1}{1 + e^{-u}}$$

tanh関数

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

ReLU関数

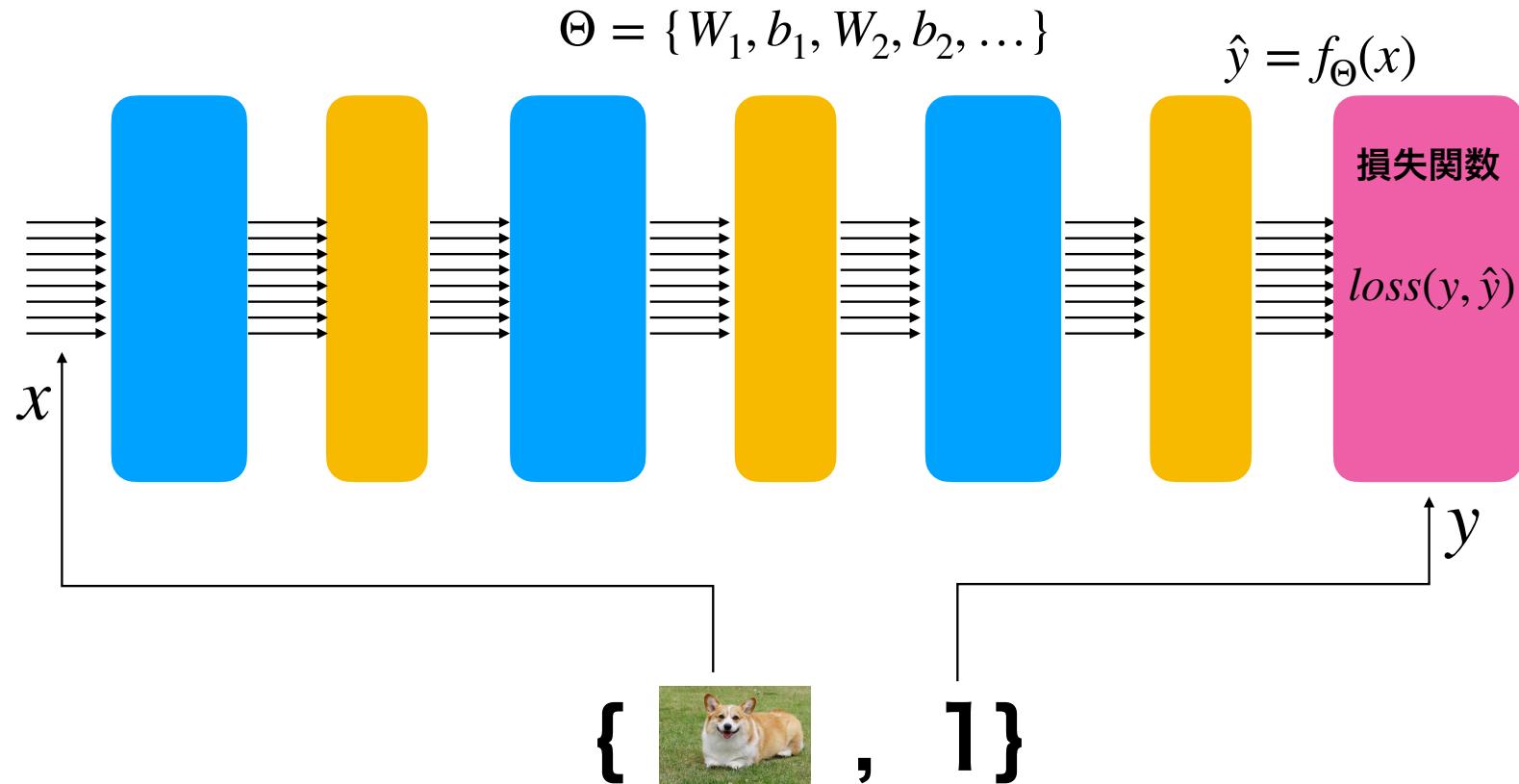
$$\text{ReLU}(u) = \max(0, u)$$



# 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值の最小化

# 深層ネットワークの訓練



訓練・学習過程では、損失関数值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

$$h = W h + b$$

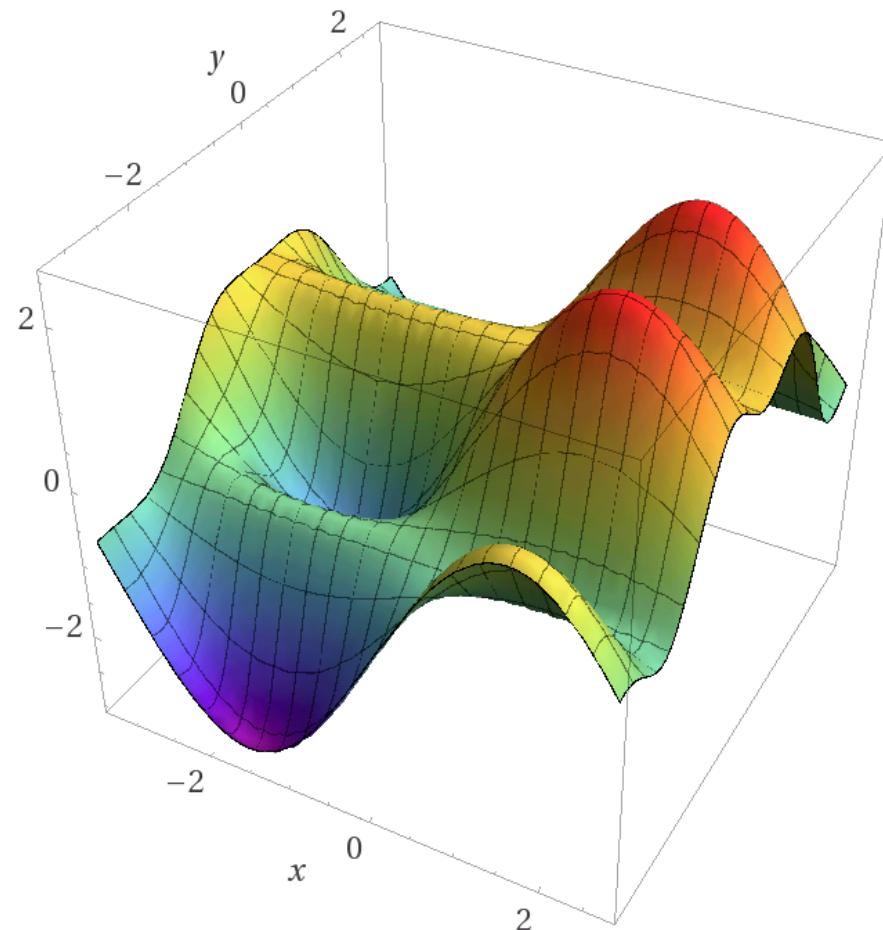
中身は動かしてよい・うまく  
チューンアップしたい！

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

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

# 勾配法による数値最小化

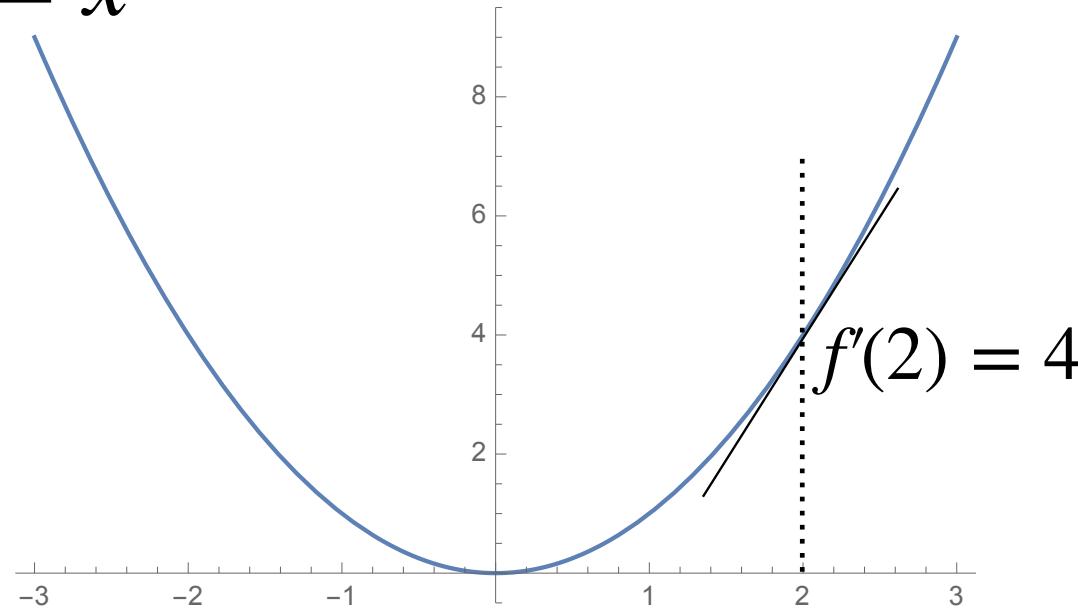
# 学習プロセス=学習パラメータの調整



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

# 勾配法(数値的最小化法)

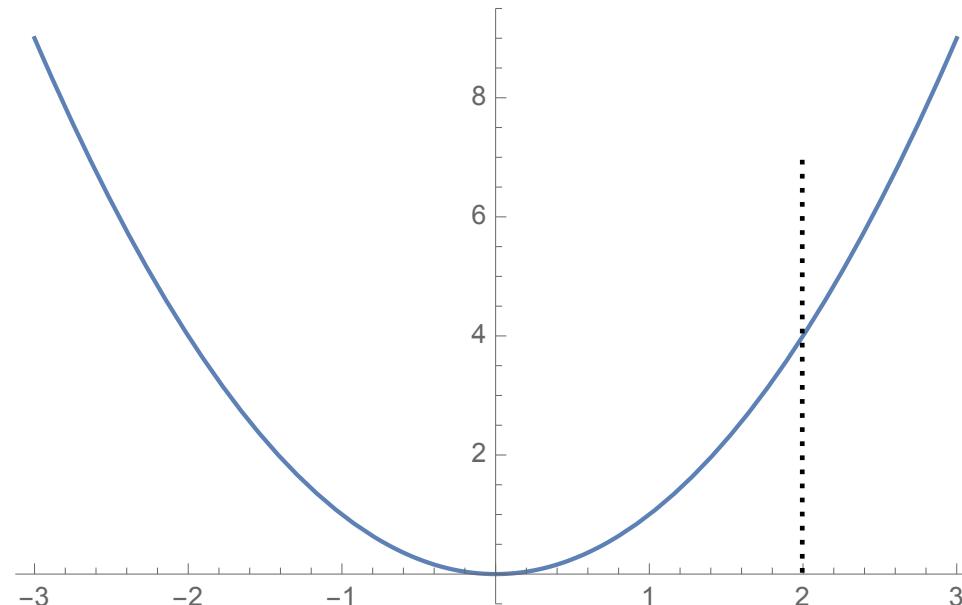
ステップ1 :  $s$  を初期値にセットする

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)

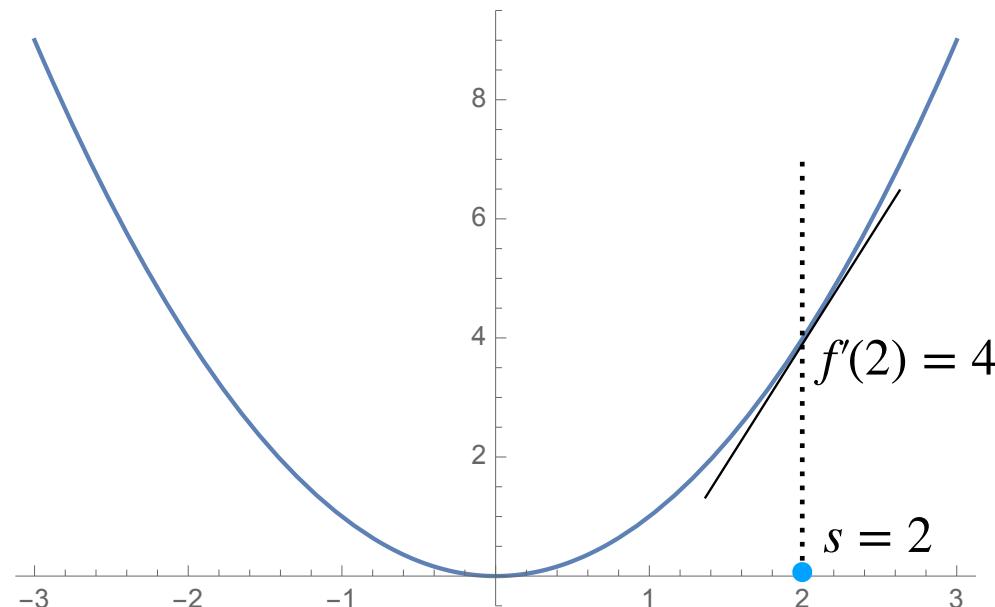
ステップ1 :  $s$  を初期値にセットする

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。



# 勾配法(数値的最小化法)

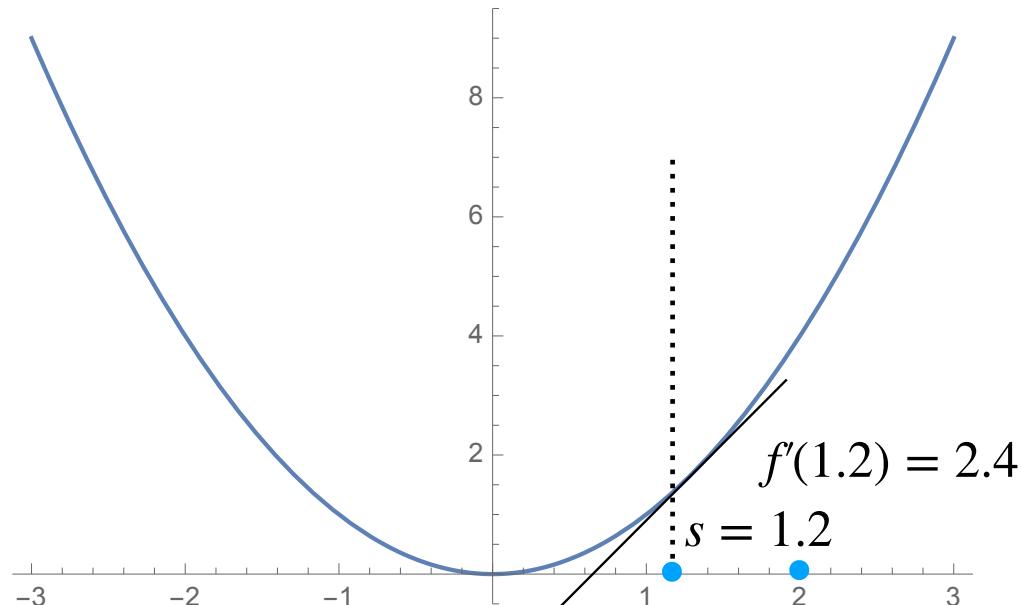
ステップ1 :  $s$  を初期値にセットする

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)

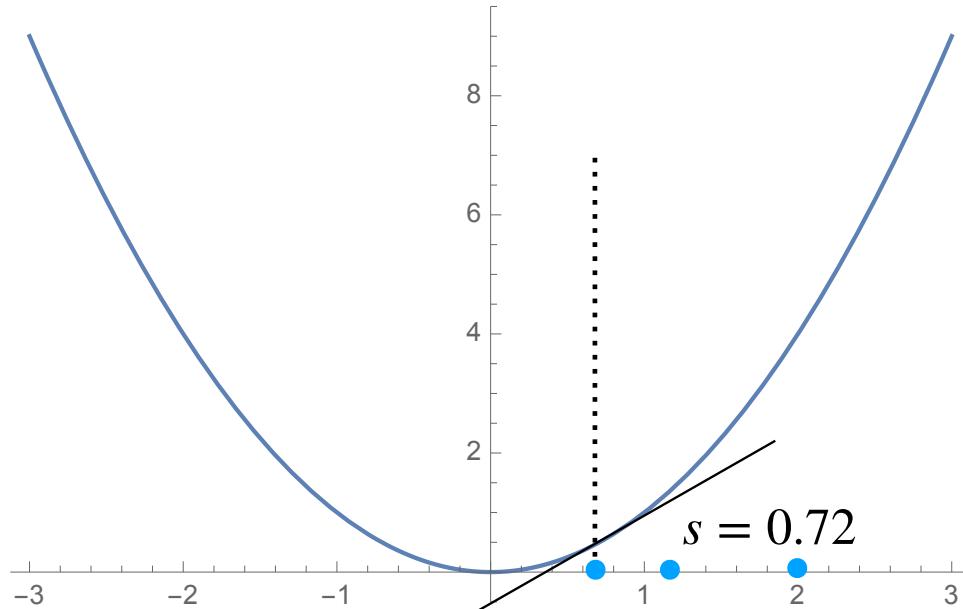
ステップ1 :  $s$  を初期値にセットする

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

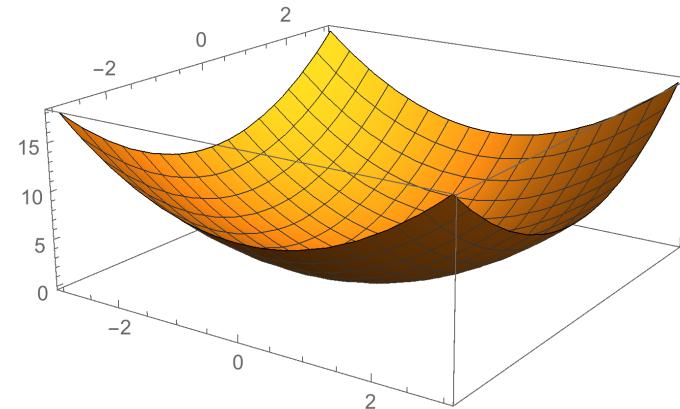
[練習問題] 初期値  $s = 2$ ,  $a = 0.2$  として上の勾配法のステップを実行せよ。



# 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)

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

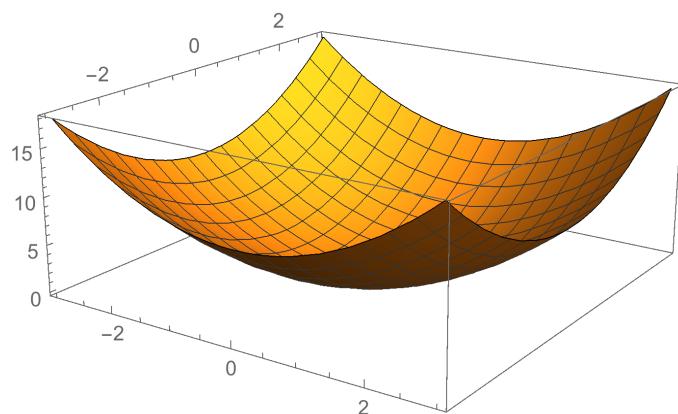


勾配ベクトル

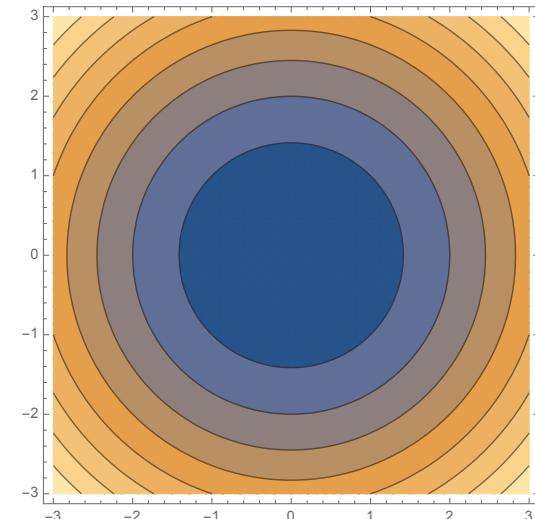
$$\nabla f(x) = \left( \frac{\partial f}{\partial x_1} = 2x_1, \frac{\partial f}{\partial x_2} = 2x_2 \right)^T$$

# 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

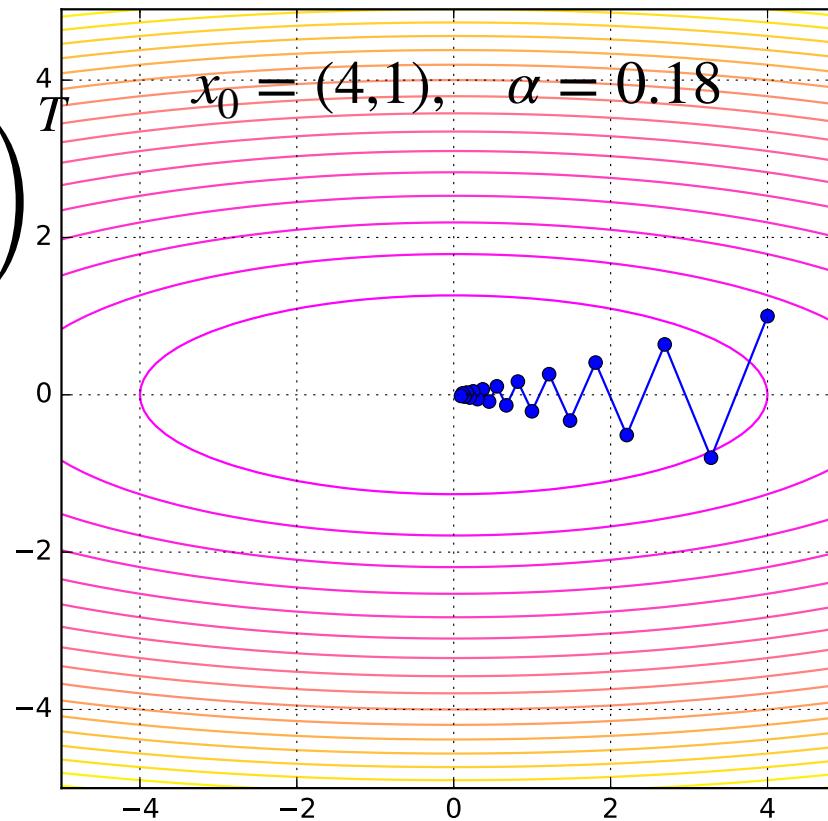
$$\nabla f(x) = \left( \frac{\partial f}{\partial x_1} = 2x_1, \frac{\partial f}{\partial x_2} = 2x_2 \right)^T$$

# 2次元関数における勾配法

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

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

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



# ふたたび：学習プロセス

$$h = W h + b$$

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！

# 中間まとめ

- ✓ 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

# Google Colaboratoryを 使ってみる



# Google Colabratoty

- ✓ 機械学習の教育と研究を促進するために  
Google が開発したツール
- ✓ 無償でクラウド上のマシンを利用可能
- ✓ ブラウザベース
- ✓ 環境構築不要でNNフレームワーク(TensorFlow,  
PyTorchが使える)
- ✓ GPUが利用できる

# Google Colabを使ってみよう

ブラウザで下記のURLにアクセスする

<https://colab.research.google.com/>

windowsユーザはchromeを使ってください。  
macユーザはsafariでOK

詳細な情報は次のQiitaの記事が詳しいです：

【秒速で無料GPUを使う】深層学習実践Tips on Colaboratory

[https://qiita.com/tomo\\_makes/items/b3c60b10f7b25a0a5935](https://qiita.com/tomo_makes/items/b3c60b10f7b25a0a5935)

# 演習ノートブックと演習課題(1)

本講義の演習課題はipython notebook形式(拡張子 .ipynb )で配布します。下記のレポジトリが入手してください。

- <https://github.com/wadayama/hogehoge>

この講義で利用するのは、Colab\_intro.ipynb

## 演習ノートブックと演習課題(2)

本講義の演習課題はipython notebook形式(拡張子 .ipynb )で配布します。Google Colabにアップロードして使用してください。

- Githubから演習ノートブック○○.ipynbをダウンロードする
- Google Colabに○○.ipynbをアップロード
- 演習の実施(レポート作成のために実行結果や編集結果を残すこと)
- 演習課題については指示に従い、Word, LaTeXなどを利用して作成する(手書き不可)。集中講義後にPDFを提出すること。

# 本日のまとめ

- 深層学習の概要
- 勾配法による最小化
- Google Colabの基本的な利用法