慶應義塾大学理工学部

機械学習基礎

第 4 回　最適化

情報工学科　教授　杉浦孔明
komei.sugiura@keio.jp

本講義の到達目標と今回の授業の狙い

本講義の到達目標

- DNNの基礎理論と実装の関係を理解する
- 種々のDNNをコーディングできる

今回の授業の狙い

- 種々の最適化アルゴリズムを習得する
- 過適合とそれを軽減する方法を習得する

- 出席確認： K-LMS上の機械学習基礎のMainページへアクセス

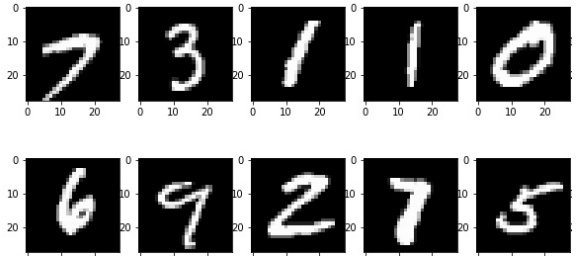
最適化

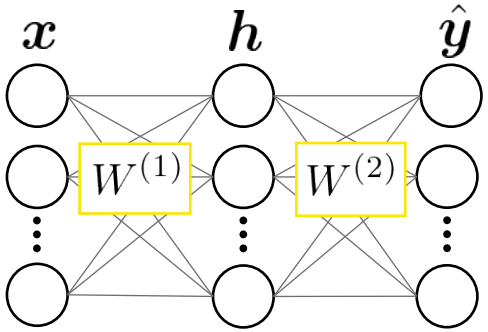
機械学習の主要要素：
データ・モデル・目的関数を定めたうえでの最適化問題

学習に使用される**データ**

ニューラルネット等の**モデル**

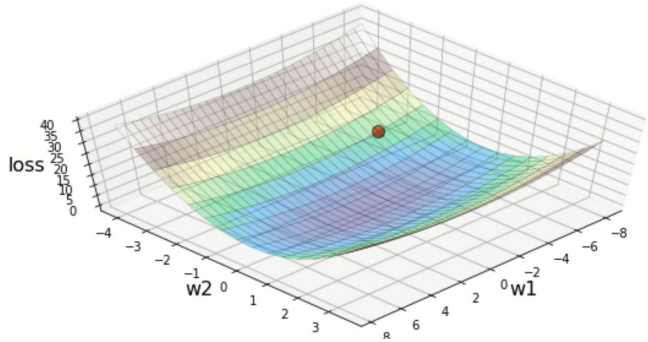
モデルの良さを定量化する**目的関数**





$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \log P(\hat{y}_{nk})$$

目的関数を最大化/最小化するために、モデルのパラメータを調整する
最適化



最適化とは

- **最適化**（optimization）

  与えられた制約条件のもとで関数の値を最大または最小にする変数の値を求める問題

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, E(\boldsymbol{w})$$

最適解

E(w)を最小化する
wを探すという意味

最適化とは

- **最適化**（optimization）

  与えられた制約条件のもとで関数の値を最大または最小にする変数の値を求める問題

$$\underbrace{\boldsymbol{w}^*}_{\text{最適解}} = \underbrace{\underset{\boldsymbol{w}}{\operatorname{argmin}}\, E(\boldsymbol{w})}_{\text{E(w)を最小化するwを探すという意味}}$$

- 例：大気汚染物質の濃度予測（前回資料）の損失関数

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2 \rightarrow \min$$

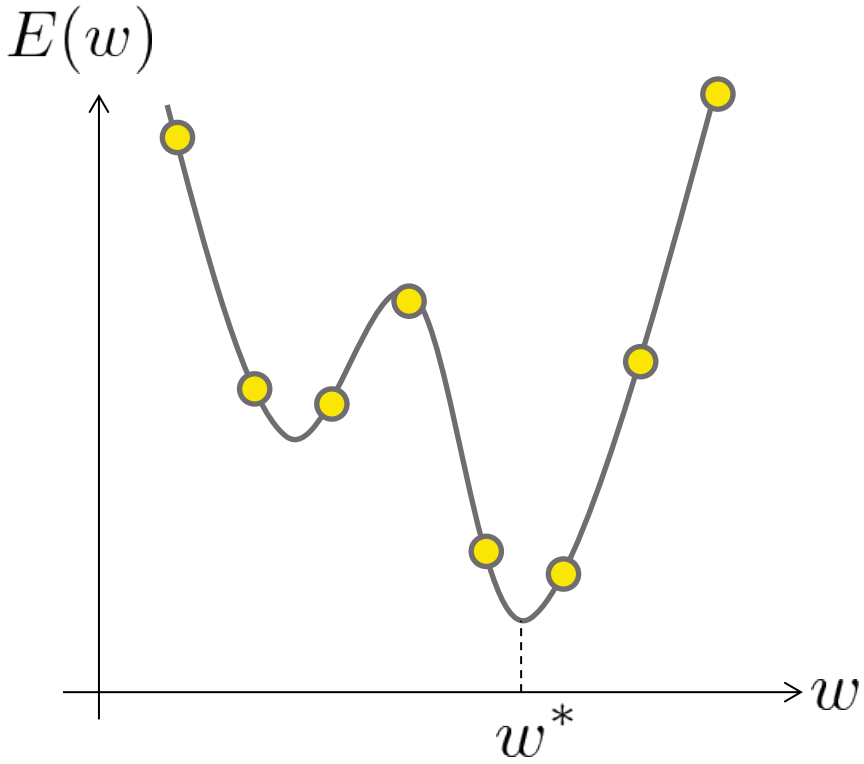
- 解析的な解は得られるか？
  →実用問題ではほぼ不可能

Q. 損失関数の概形を見れば最適解が見つかるのでは？
A. No！パラメータ数が多いので、概形を見ることすら大変

- 例：パラメータ数をKとする。
  パラメータ毎に8点ずつプロットするとき、プロット点は全部でいくつか？

- K=2　　→64
- K=3　　→512
- K=1億　→8の1億乗

卒論でも1億パラメータは普通



プロット点が不足している可能性もある

数値的に近似解を求めるための代表的な方法：勾配降下法

- **勾配降下法**（gradient descent method） or **最急降下法**（steepest descent method）

1. 初期値 $\boldsymbol{w}^{(0)}$ を用意

2. 更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E(\boldsymbol{w}^{(t)})$$

学習率（learning rate; lr）

更新回数

数値的に近似解を求めるための代表的な方法：勾配降下法

■ **勾配降下法**（gradient descent method） or **最急降下法**（steepest descent method）

1. 初期値 $\boldsymbol{w}^{(0)}$ を用意

2. 更新則

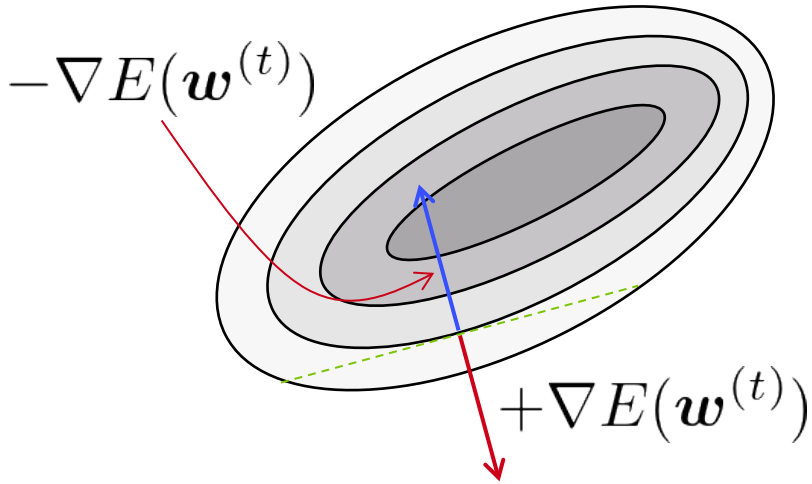
$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E(\boldsymbol{w}^{(t)})$$

学習率（learning rate; lr）

更新回数

■ **勾配**（gradient）

$$\nabla E(\boldsymbol{w}) \triangleq \left(\frac{\partial E(\boldsymbol{w})}{\partial w_1}, \frac{\partial E(\boldsymbol{w})}{\partial w_2}, \cdots, \frac{\partial E(\boldsymbol{w})}{\partial w_D} \right)$$



数値的に近似解を求めるための代表的な方法：勾配降下法

- **勾配降下法**（gradient descent method） or **最急降下法**（steepest descent method）

1. 初期値 $\boldsymbol{w}^{(0)}$ を用意
2. 更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E(\boldsymbol{w}^{(t)})$$

更新回数

学習率（learning rate; lr）

- **勾配**（gradient）

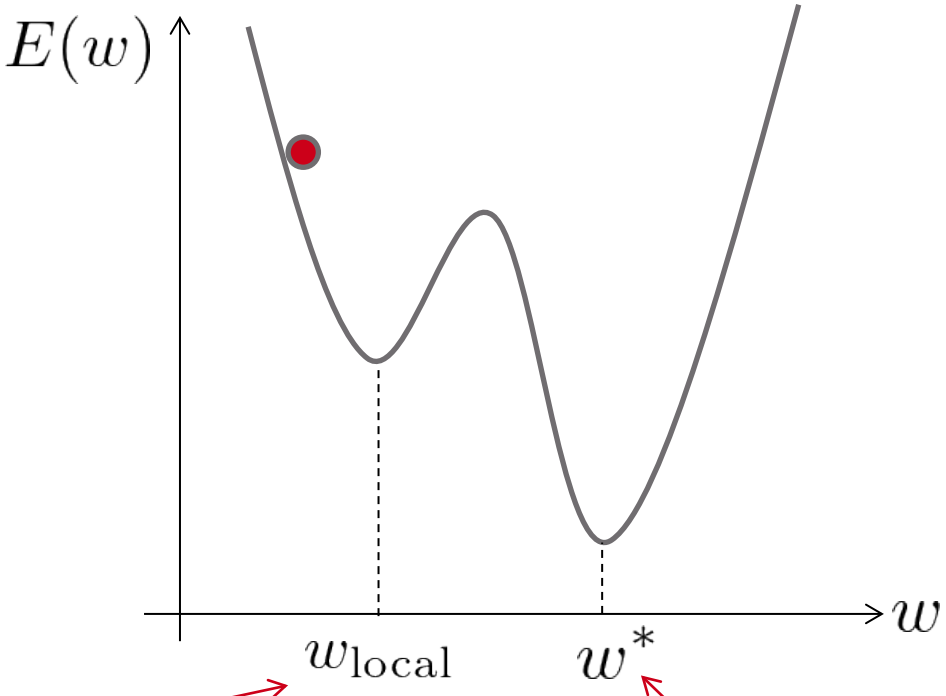
$$\nabla E(\boldsymbol{w}) \triangleq \left(\frac{\partial E(\boldsymbol{w})}{\partial w_1}, \frac{\partial E(\boldsymbol{w})}{\partial w_2}, \ldots, \frac{\partial E(\boldsymbol{w})}{\partial w_D} \right)$$

  - 各次元は傾きを表す
  - 各次元を比較すれば、関数値が最も急激に変化する次元（方向）がわかる

- B2「応用数学」では 2 階微分を用いる**ニュートン法**を習った
  - 計算量が多すぎるため、DNN では非主流

学習において問題となるケース：局所的極小値や鞍点からの脱出

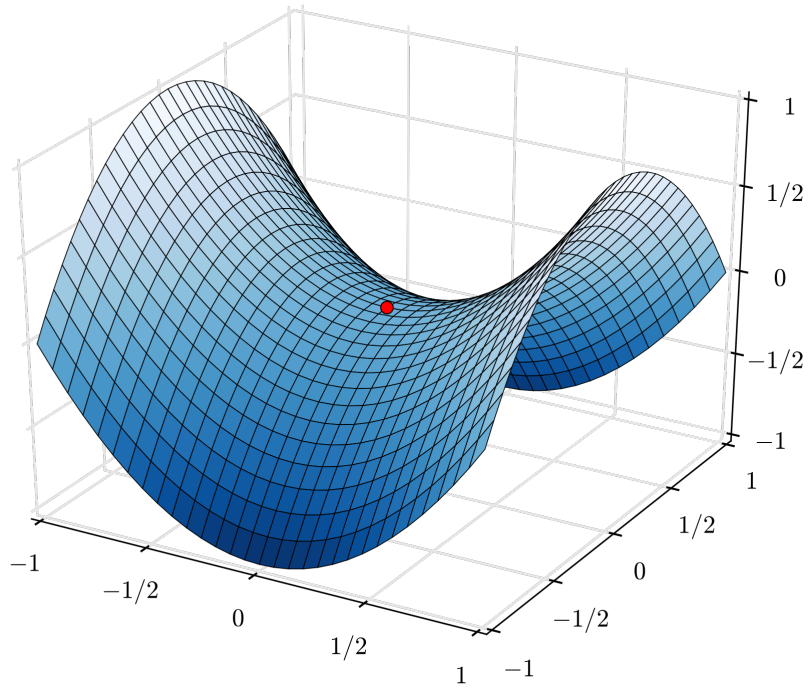
局所的極小値（local minimum）



局所的極小値 or
局所解（local solution）

大域的極小値
（global minimum）

鞍点（saddle point）：
勾配は 0 であるが局所的極小値ではない点



https://en.wikipedia.org/wiki/Saddle_point

ミニバッチSGD

- ミニバッチSGDの更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla \underline{E^{(t)}}(\boldsymbol{w}^{(t)})$$

勾配降下法との違いはこれがミニバッチを表すようになっただけ

- ランダムにサンプルが選ばれる→勾配が揺らぐ→局所解や鞍点からの脱出を助ける
- E_n を独立に計算できるので、GPU計算に向く

ミニバッチSGD

- ミニバッチSGDの更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E^{(t)}(\boldsymbol{w}^{(t)})$$

勾配降下法との違いはこれがミニバッチを表すようになっただけ

- ランダムにサンプルが選ばれる→勾配が揺らぐ→局所解や鞍点からの脱出を助ける
- E_n を独立に計算できるので、GPU計算に向く

- **ミニバッチ学習**（mini-batch learning）
  - １回の更新に訓練集合の一部（ミニバッチ $\mathcal{B}^{(t)}$）を使用
  - 更新ごとに損失関数の形状が変わる

$$E^{(t)}(\boldsymbol{w}) = \frac{1}{|\mathcal{B}^{(t)}|} \sum_{n \in \mathcal{B}^{(t)}} E_n(\boldsymbol{w})$$

ミニバッチに含まれるインデックスの集合

１サンプル毎の損失

ミニバッチの作成方法の例

濃度（今）	風速（今）	濃度（未来）
$\boldsymbol{x}_n$		y_n
5	2.0	4
7	1.2	5
10	1.6	11
…	…	…
10	1.8	10
9	2.6	10
8	1.8	6

※これ以外の作成方法でも良い

ミニバッチの作成方法の例

濃度（今）	風速（今）	濃度（未来）
$\boldsymbol{x}_n$		y_n
8	1.8	6
10	1.8	10
7	1.2	5
…	…	…
5	2.0	4
10	1.6	11
9	2.6	10

シャッフル

※これ以外の作成方法でも良い

ミニバッチの作成方法の例

濃度（今）	風速（今）	濃度（未来）
$\boldsymbol{x}_n$		y_n
8	1.8	6
10	1.8	10
7	1.2	5
…	…	…
5	2.0	4
10	1.6	11
9	2.6	10

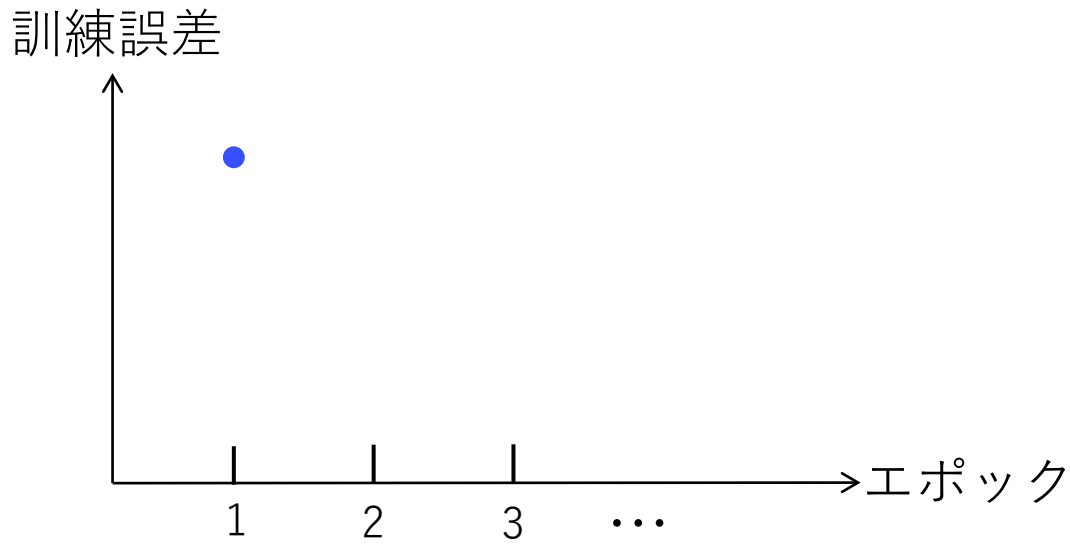
$\mathcal{B}^{(1)}$ → 1回目の更新

$\mathcal{B}^{(2)}$ → 2回目の更新

⋮

$\mathcal{B}^{(K)}$ → K回目の更新

エポック
（epoch）



エポック毎に再シャッフル

※これ以外の作成方法でも良い

ミニバッチの作成方法の例

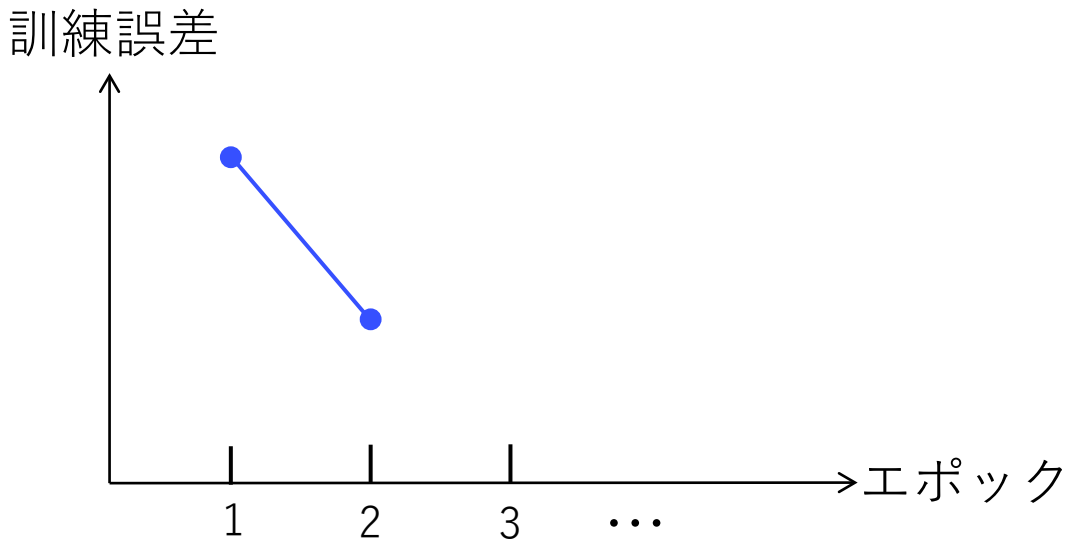
濃度（今）	風速（今）	濃度（未来）
$\boldsymbol{x}_n$		y_n
9	2.6	10
5	2.0	4
10	1.8	10
…	…	…
8	1.8	6
7	1.2	5
10	1.6	11

$\mathcal{B}^{(1)}$ → 1回目の更新

$\mathcal{B}^{(2)}$ → 2回目の更新

⋮

$\mathcal{B}^{(K)}$ → K回目の更新



エポック
（epoch）

エポック毎に再シャッフル

※これ以外の作成方法でも良い

ミニバッチの作成方法の例

濃度（今）	風速（今）	濃度（未来）
$\boldsymbol{x}_n$		y_n
10	1.6	11
5	2.0	4
8	1.8	6
…	…	…
7	1.2	5
10	1.8	10
9	2.6	10

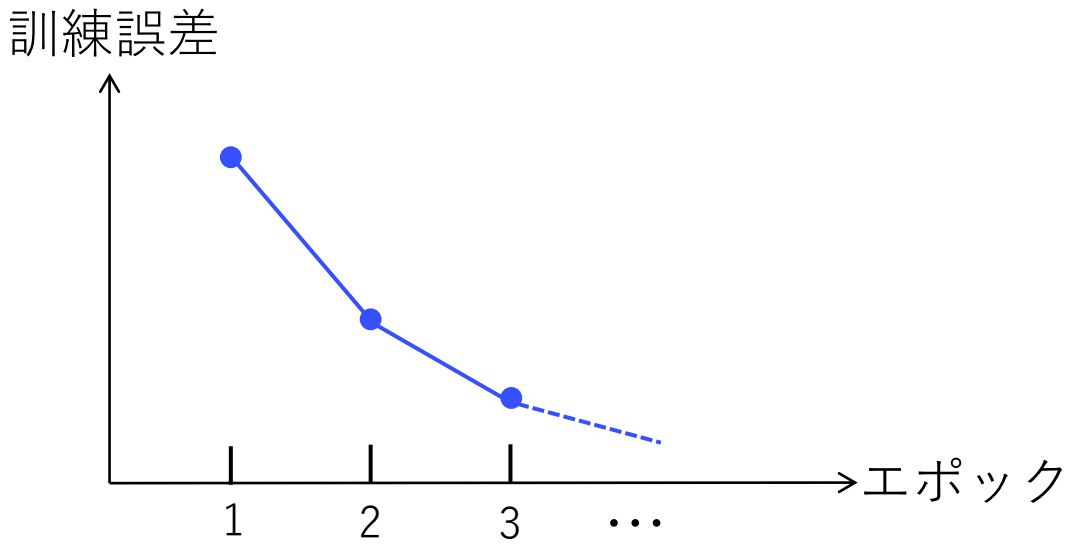
$\mathcal{B}^{(1)}$ → 1回目の更新

$\mathcal{B}^{(2)}$ → 2回目の更新

⋮

$\mathcal{B}^{(K)}$ → K回目の更新

エポック
（epoch）



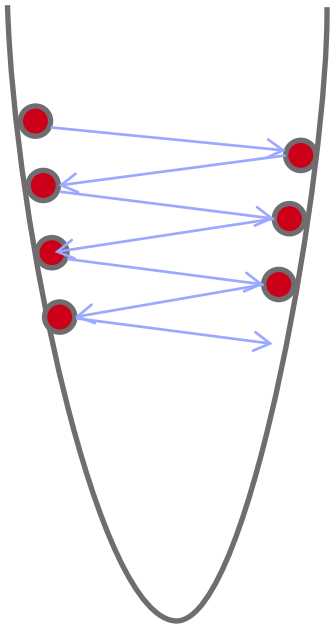
エポック毎に再シャッフル

※これ以外の作成方法でも良い

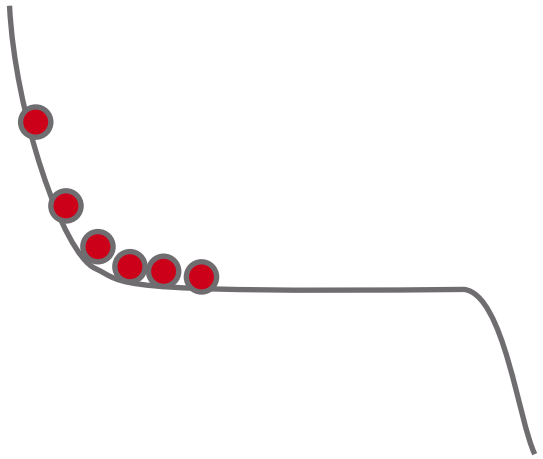
勾配降下法の改良

勾配降下法の課題：
更新時に振動したり停止したりするケース

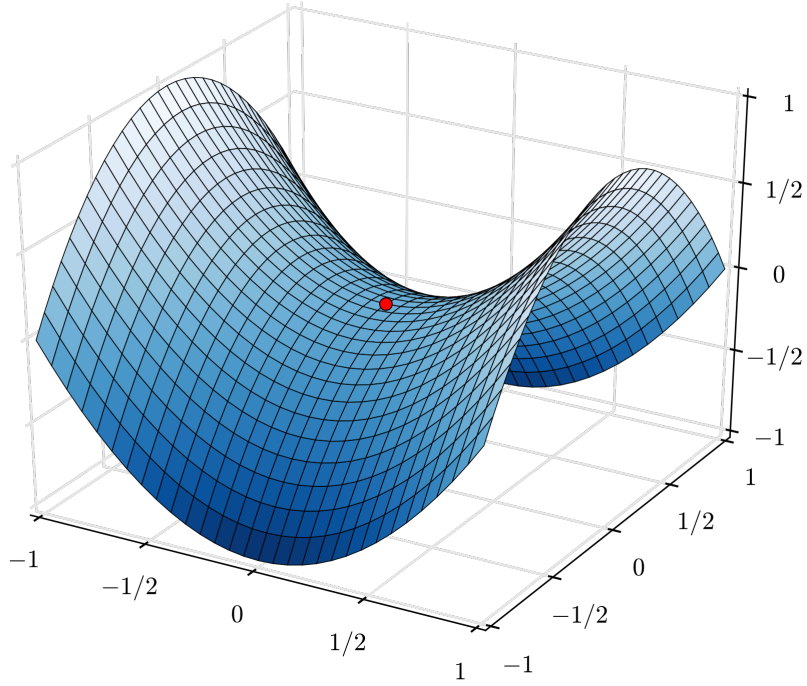
正負交互の急な勾配
（**振動**を起こす）



プラトー（勾配が０）



鞍点（saddle point）：
勾配は０であるが局所的極小値ではない点



https://en.wikipedia.org/wiki/Saddle_point

モーメンタム法
（the momentum method）

- 背景
  - 正負交互の急な勾配（振動）を抑制したい
- ステップtの勾配だけ利用するのではなく、t-1の勾配も利用
  （振動を打ち消し合うはず）

モーメンタム法
（the momentum method）

- 背景
  - 正負交互の急な勾配（振動）を抑制したい
- ステップtの勾配だけ利用するのではなく、t-1の勾配も利用
  （振動を打ち消し合うはず）

- **モーメンタム法**の更新則
- 初期値$\boldsymbol{w}^{(0)}$と$\Delta\boldsymbol{w}^{(0)} = \mathbf{0}$ を用意

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \Delta\boldsymbol{w}^{(t)}$$

$$\Delta\boldsymbol{w}^{(t)} = \rho\Delta\boldsymbol{w}^{(t-1)} - \eta\nabla E\left(\boldsymbol{w}^{(t)}\right)$$

↑前回の ρ 割を使う

- $\rho = 0$ とすれば勾配降下法と同じ

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta\nabla E\left(\boldsymbol{w}^{(t)}\right)$$

AdaGrad [Duchi+ 2011]
背景

- 背景
  - w_1 方向：急な勾配
  - w_2 方向：緩やか

    ↓

    同じ学習率だとw_2方向の更新が遅くなる

- 各パラメータ方向に応じて、学習率の影響を変えたい

  →過去に勾配が大きかった方向の更新量を抑制しよう

AdaGrad [Duchi+ 2011]
更新則

- 背景
  - w_1 方向：急な勾配
  - w_2 方向：緩やか

  ↓

  同じ学習率だとw_2方向の更新が遅くなる

- 各パラメータ方向に応じて、学習率の影響を変えたい

  →過去に勾配が大きかった方向の更新量を抑制しよう

- **AdaGrad**の更新則

  初期値 $\boldsymbol{w}^{(0)}$と $\Delta \boldsymbol{w}^{(0)} = \boldsymbol{0}$ を用意

$$\boldsymbol{w}_i^{(t+1)} = \boldsymbol{w}_i^{(t)} + \Delta \boldsymbol{w}_i^{(t)}$$

第i成分

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{\sum_{\tau=1}^{t} (\nabla E(\boldsymbol{w}^{(\tau)})_i)^2 + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

ゼロ除算を避けるため
0.000001などを入れる

AdaGrad [Duchi+ 2011]
欠点

- 学習の初期に勾配が大きいと、急激に $\Delta \boldsymbol{w}_i^{(t)}$ が小さくなる
（この部分は常に増大するため）

- **AdaGrad**の更新則
初期値 $\boldsymbol{w}^{(0)}$ と $\Delta \boldsymbol{w}^{(0)} = \boldsymbol{0}$ を用意

$$\boldsymbol{w}_i^{(t+1)} = \boldsymbol{w}_i^{(t)} + \Delta \boldsymbol{w}_i^{(t)}$$

今後この式は書かない

第i成分

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{\sum_{\tau=1}^{t} (\nabla E(\boldsymbol{w}^{(\tau)})_i)^2} + \varepsilon} \nabla E(\boldsymbol{w}^{(t)})_i$$

ゼロ除算を避けるため
0.000001などを入れる

RMSprop
更新則

■ **RMSprop**の更新則
状態変数を１つ導入

$$v_{i,0} = 0$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{v_{i,t} + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

■ **AdaGrad**の更新則
初期値 $\boldsymbol{w}^{(0)}$ と $\Delta \boldsymbol{w}^{(0)} = \boldsymbol{0}$ を用意

$$\boldsymbol{w}_i^{(t+1)} = \boldsymbol{w}_i^{(t)} + \Delta \boldsymbol{w}_i^{(t)}$$

今後この式は書かない

第i成分

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{\sum_{\tau=1}^{t} (\nabla E(\boldsymbol{w}^{(\tau)})_i)^2 + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

分母が違う

ゼロ除算を避けるため
0.000001などを入れる

RMSprop
指数移動平均の利用

■ **RMSprop**の更新則
状態変数を１つ導入

$$v_{i,0} = 0$$

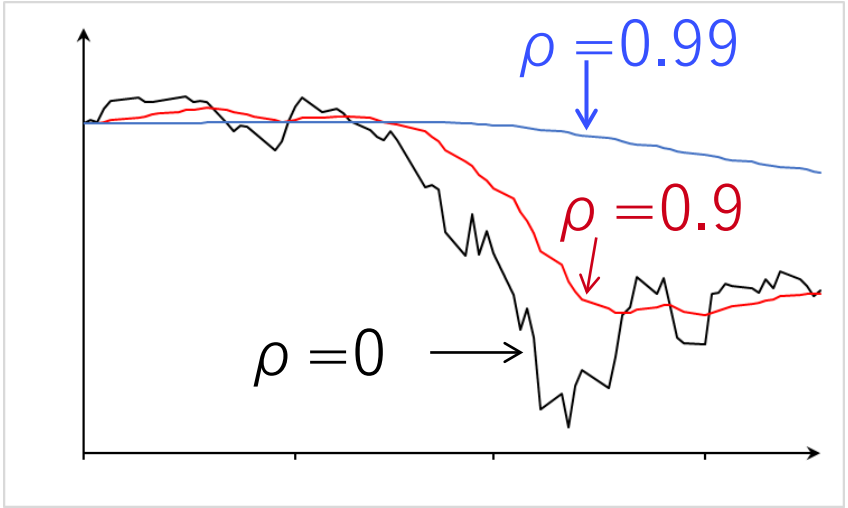
$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{v_{i,t} + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

■ **指数移動平均**（exponential moving average; EMA）

$$S_t = \rho S_{t-1} + (1-\rho) a_t$$

↑ ρ 割をキープし、(1-ρ)割を新しい値に



RMSprop
特徴と問題

- **RMSprop**の更新則
  状態変数を１つ導入

$$v_{i,0} = 0$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{v_{i,t} + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

2乗平均の平方根なのでroot mean square (RMS)

- RMSpropの特徴
  - AdaGradに似ている
  - 勾配の二乗の指数移動平均を学習率のスケーリングに利用
- RMSpropの問題
  - 学習率に鋭敏

学習率を変えると結果がガラッと変わってしまう

AdaDelta [Zeiler 2012]
更新則

■ **RMSprop**の更新則
状態変数を１つ導入

$$v_{i,0} = 0$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\eta}{\sqrt{v_{i,t} + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

2乗平均の平方根なのでroot mean square (RMS)

同じ

■ **AdaDelta**の更新則
状態変数を２つ導入＝２種類の指数移動平均を使う

$$(u_{i,0}, v_{i,0}) = (0, 0)$$

$$u_{i,t} = \rho u_{i,t-1} + (1-\rho)(\Delta \boldsymbol{w}_i^{(t)})^2$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\sqrt{u_{i,t-1} + \varepsilon}}{\sqrt{v_{i,t} + \varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

↑学習率を陽に持たない

AdaDelta [Zeiler 2012]
RMSpropとの相違点

- 分子
  - 学習率ではなく更新量のRMS
  - $u_{i,t}$ とすると循環的定義になってしまうので $u_{i,t-1}$ で代用
- 分母はRMSpropと同じ

■ **AdaDelta**の更新則

状態変数を２つ導入＝２種類の指数移動平均を使う

$$(u_{i,0}, v_{i,0}) = (0, 0)$$

$$u_{i,t} = \rho u_{i,t-1} + (1-\rho)(\Delta \boldsymbol{w}_i^{(t)})^2$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\sqrt{u_{i,t-1}+\varepsilon}}{\sqrt{v_{i,t}+\varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

↑学習率を陽に持たない

Adam [Kingma+ 2014]
特徴

- 現時点で、深層学習において最も広く使われている
  「とりあえずAdamで試す」

- これまで紹介した手法を取り入れている

- **AdaDelta**の更新則

  状態変数を２つ導入＝２種類の指数移動平均を使う

$$(u_{i,0}, v_{i,0}) = (0, 0)$$

$$u_{i,t} = \rho u_{i,t-1} + (1-\rho)(\Delta \boldsymbol{w}_i^{(t)})^2$$

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\sqrt{u_{i,t-1}+\varepsilon}}{\sqrt{v_{i,t}+\varepsilon}} \nabla E(\boldsymbol{w}^{(t)})_i$$

↑学習率を陽に持たない

Adam [Kingma+ 2014]
相違点

■ Adam

$$(m_{i,0}, v_{i,0}) = (0, 0)$$

$$m_{i,t} = \rho_1 m_{i,t-1} + (1-\rho_1)\nabla E(\boldsymbol{w}^{(t)})_i$$

勾配の指数移動平均

$$v_{i,t} = \rho_2 v_{i,t-1} + (1-\rho_2)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

■ AdaDeltaの更新則

状態変数を２つ導入＝２種類の指数移動平均を使う

$$(u_{i,0}, v_{i,0}) = (0, 0)$$

違う

$$u_{i,t} = \rho u_{i,t-1} + (1-\rho)(\Delta \boldsymbol{w}_i^{(t)})^2$$

同じ

$$v_{i,t} = \rho v_{i,t-1} + (1-\rho)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

$$\Delta \boldsymbol{w}_i^{(t)} = -\frac{\sqrt{u_{i,t-1}+\varepsilon}}{\sqrt{v_{i,t}+\varepsilon}}\nabla E(\boldsymbol{w}^{(t)})_i$$

↑学習率を陽に持たない

Adam [Kingma+ 2014]
更新則

■ **Adam**

$$(m_{i,0}, v_{i,0}) = (0, 0)$$

$$m_{i,t} = \rho_1 m_{i,t-1} + (1-\rho_1)\nabla E(\boldsymbol{w}^{(t)})_i$$

勾配の指数移動平均

$$v_{i,t} = \rho_2 v_{i,t-1} + (1-\rho_2)(\nabla E(\boldsymbol{w}^{(t)})_i)^2$$

■ Adamの更新則

真の値より0側に偏るので、
以下のように補正

$$\hat{m}_{i,t} = \frac{m_{i,t}}{1-(\rho_1)^t} \qquad \hat{v}_{i,t} = \frac{v_{i,t}}{1-(\rho_2)^t}$$

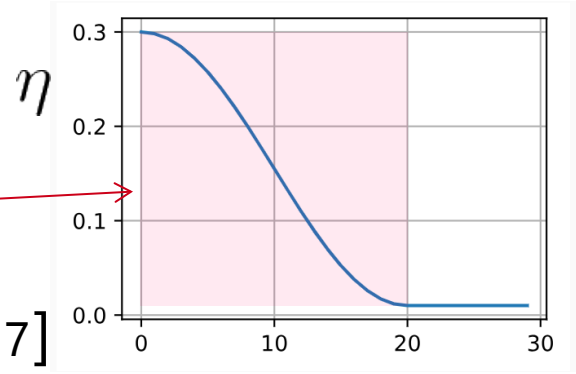
↑ρ_1のt乗

$$\Delta \boldsymbol{w}_i^{(t)} = -\eta \frac{\hat{m}_{i,t}}{\sqrt{\hat{v}_{i,t} + \varepsilon}}$$

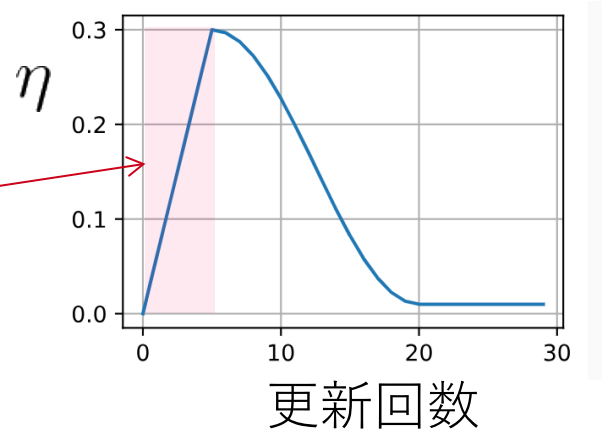
学習率のスケジューリング

- 固定された学習率では収束したい点に近づけないことがある☹
  →学習率を変化させる

- コサインアニーリング
  - 学習率を減衰させる
  - 増加と減衰を繰り返す方法もある [Loshchilov+ ICLR17]



- ウォームアップ（warmup）
  - 大きな学習率から開始すると発散することがあるので、初期のみ学習率を増加させる



理解度確認

理解度確認
以下について周りと相談して１分以内に答えよ

1. ３層ニューラルネットを構成する層は何と何と何と呼ばれるか？
2. 尤度とは何か？
3. ベルヌーイ分布の確率質量関数をμとxを使って表せ
4. Kクラス分類問題に対する交差エントロピー誤差関数をy_{nk}を使って表せ
5. 1-of-K表現とは何か？

理解度確認
以下について周りと相談して１分以内に答えよ

1. 訓練集合とテスト集合の違いは何か？
2. 訓練集合と訓練サンプルの違いは何か？
3. ミニバッチ確率的勾配降下法の英語名は何か？
4. 損失関数の例を挙げよ。

汎化と過学習

訓練誤差・テスト誤差・汎化誤差の違い

- **訓練誤差**（training error）
  - 訓練集合に対する誤差
- **テスト誤差**（test error）
  - テスト集合に対する誤差

ならば、仮想で考えるしかない

- 機械学習の目標
  - 新規未知データに対して誤りを小さくしたい

訓練誤差・テスト誤差・汎化誤差の違い

- **訓練誤差**（training error）
  - 訓練集合に対する誤差
- **テスト誤差**（test error）
  - テスト集合に対する誤差

ならば、仮想で考えるしかない

- 機械学習の目標
  - 新規未知データに対して誤りを小さくしたい

- 当該データを生成する仮想的な分布を考える（現実には計算できない）

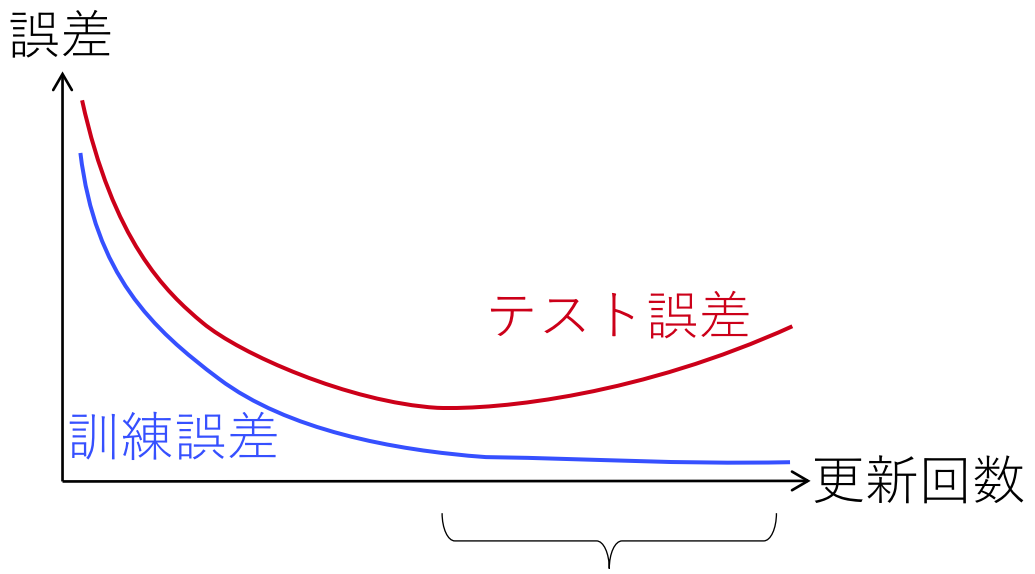
↓

- データの生成分布に対するモデルの誤差の期待値
  ＝**汎化誤差**（generalization error）

汎化誤差の手軽な代用物として
テスト誤差を使用する

学習曲線（learning curve）

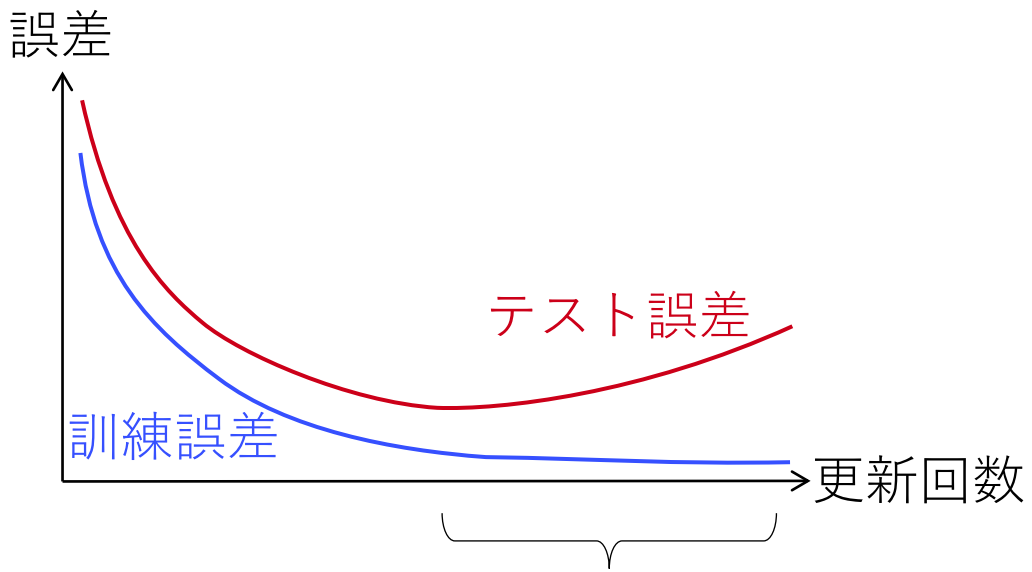
- 途中まで
  ☺訓練誤差とテスト誤差が共に下がる
- 途中から
  ☺訓練誤差が下がる
  ☹テスト誤差が上がる
- **過学習**（過適合、overfitting）



※代表的な学習曲線は上記であるが、テスト誤差が再度低下する現象（二重降下）についても近年研究されている

過学習：
単なる訓練誤差の最小化だけでは不十分

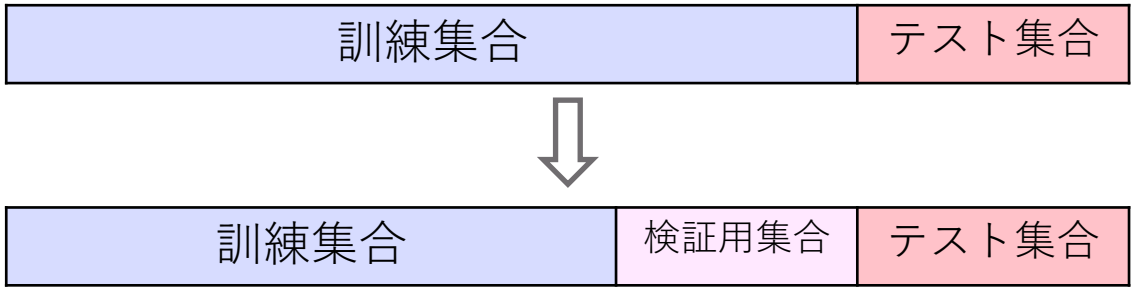
- 過学習の原因
  - 本来学習させたい特徴とは無関係な特徴にまで適合してしまうため
    ↑訓練集合は有限なのでどうしても統計的ばらつきが発生してしまう
- 過学習は機械学習における普遍的問題
  - 最適化対象と汎化誤差のミスマッチ



次スライドから過学習を避ける手法を紹介する

早期終了（early stopping）
過学習に陥る前にパラメータ更新を停止

- 訓練集合をさらに分割

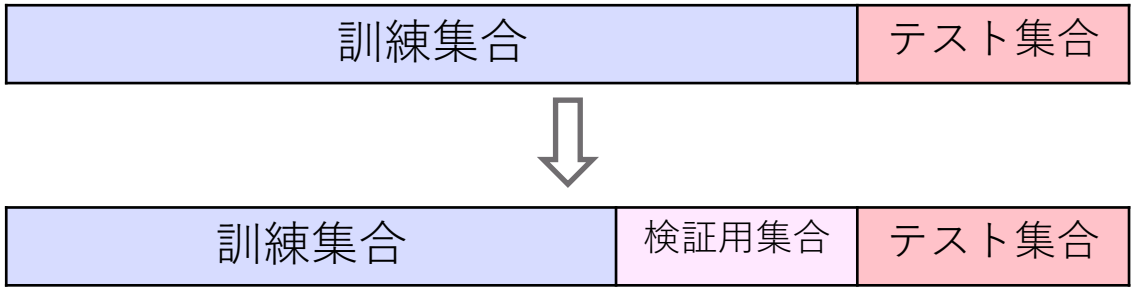


  - **検証用集合**（validation set, development set）
    →パラメータ推定に使用しない

- 分割比の例
  - 訓練：検証：テスト＝8:1:1

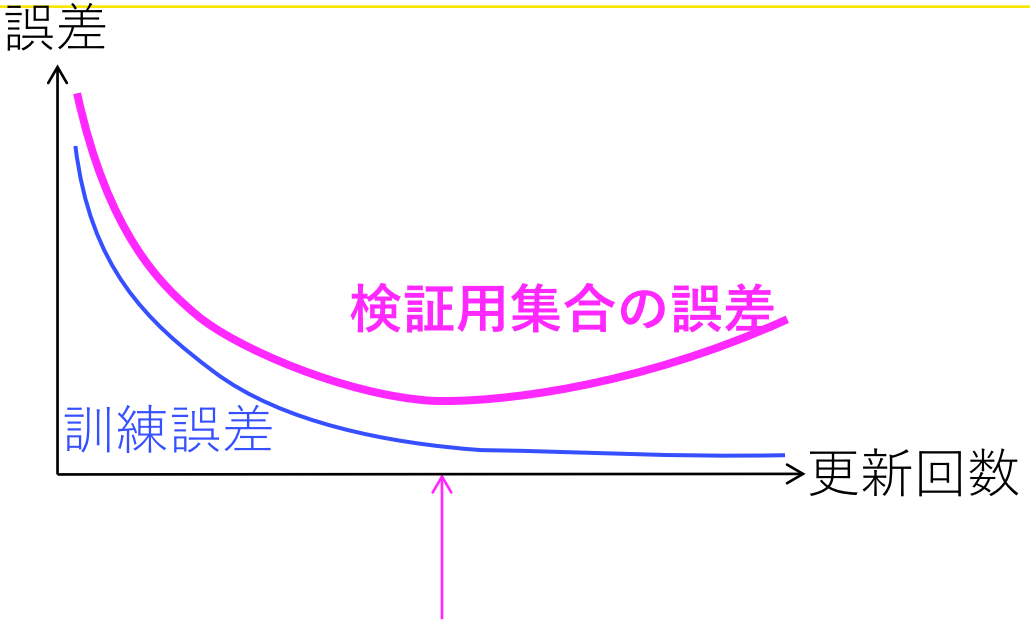
早期終了（early stopping）
過学習に陥る前にパラメータ更新を停止

- 訓練集合をさらに分割



  - **検証用集合**（validation set, development set）
    →パラメータ推定に使用しない

- 分割比の例
  - 訓練：検証：テスト＝8:1:1

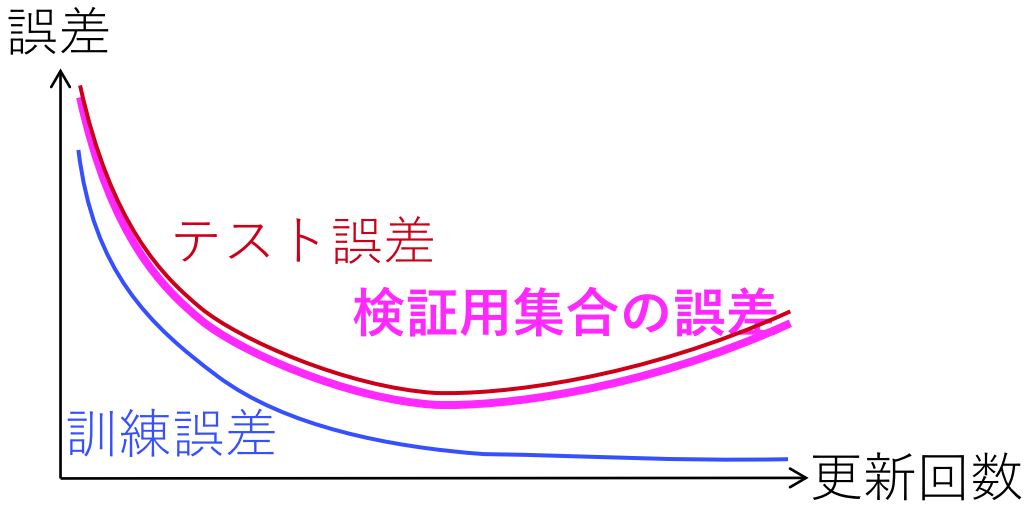


検証用集合の誤差を最小とするモデルで**テスト誤差を評価**することで過学習を避ける

※**テスト集合**の誤差を最小とするモデルを選択するのはチート

本講義における検証用集合とテスト集合の定義

- 検証用集合の主な用途は**モデル選択**
  （or ハイパーパラメータ探索）
- 検証用集合とテスト集合の定義
  - 一度も評価に使用されていない集合を「真のテスト集合」とする流儀もある
  - 本講義では、データセットを訓練集合とテスト集合に分割し、テスト集合を評価に使うものとする
    ↑多くの教科書の慣習に従う



エポック毎に検証用集合とテスト集合の誤差をプロットするのであれば、両者の違いは何？ということになる

正則化（regularization）

- 損失関数に**正則化項**（penalty term）$R(\boldsymbol{w})$を追加することで、過学習を避ける
  - 正則化項：モデルの複雑さに対するペナルティ

$$\widetilde{E}(\boldsymbol{w}) = E(\boldsymbol{w}) + \lambda R(\boldsymbol{w})$$

通常の誤差

正則化パラメータ： $E(\boldsymbol{w})$ と $R(\boldsymbol{w})$ のバランスを指定

- 例：lasso＝正則化項がパラメータの絶対値の和

$$\widetilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{n}(y_n - \hat{y}_n)^2 + \frac{\lambda}{2}||\boldsymbol{w}||$$

L1ノルムと呼ぶ

L2ノルム（パラメータの２乗和）を用いる場合はリッジ回帰と呼ばれる

★ドロップアウト（dropout） [Srivastava+ 2014]

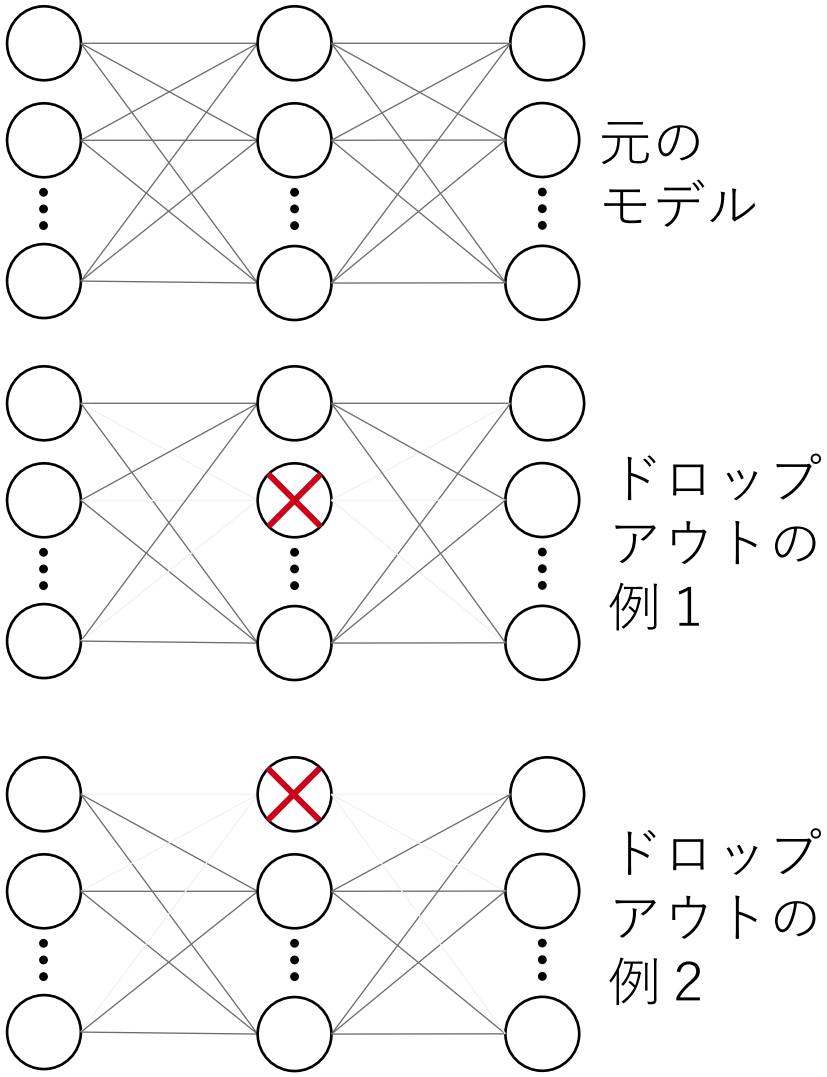
- ユニット出力を確率pで0にする
  - p=0.2-0.5が多い

$$h_j = \begin{cases} 0 & (\text{確率} p) \\ \frac{h_j}{1-p} & (\text{確率} 1-p) \end{cases}$$

0にするのみだと期待値が低くなってしまうため

- 効果：　性能を安定化させる※
- 中間層に対するノイズ付加に相当

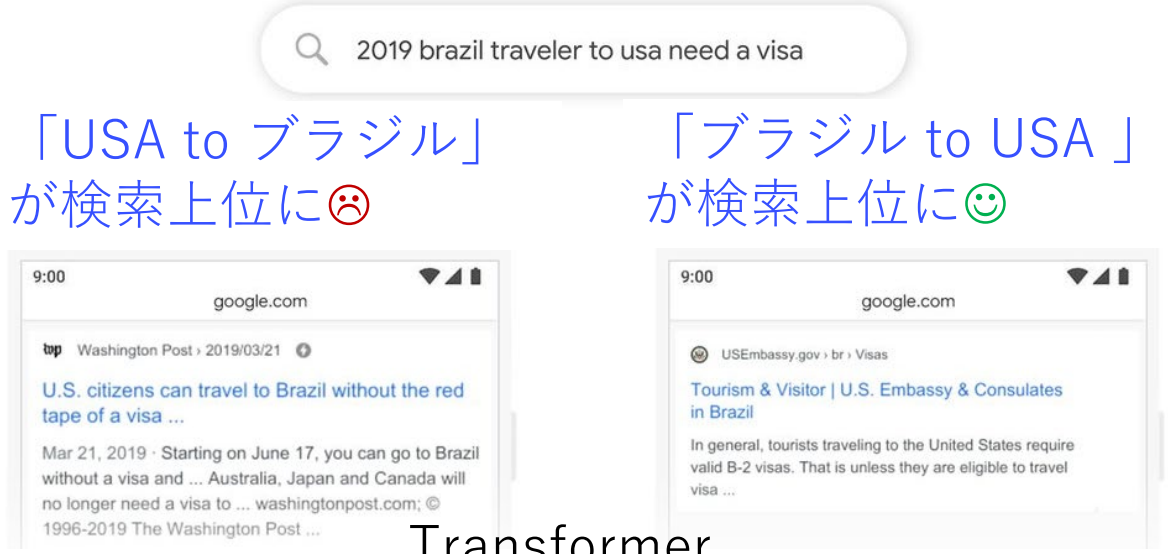
※アンサンブル学習の一種と考えられる理論的背景がある



★バッチ正規化（batch normalization） [Ioffe+ 2015]

- 効果：　学習を安定化させる
  ↑バッチ正規化発明以前：　☹鋭い極小値の影響が強いため学習率を小さくしなければならなかった
- 現代的なDNNではバッチ正規化（とその後継）を多用
- ドロップアウトを一部代替

GAN
https://www.whichfaceisreal.com/



Transformer
https://blog.google/products/search/search-language-understanding-bert/

バッチ正規化：ユニットが１つの場合

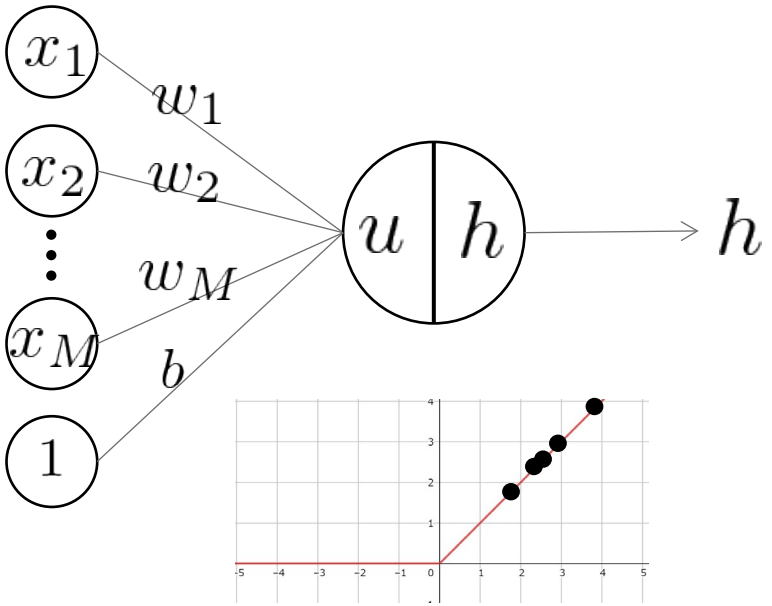
①活性値uを**標準化**（＝平均０、分散１になるように変換）

ミニバッチ内のサンプルに対するuの平均

$$\hat{u} = \frac{u - \mu}{\sqrt{\sigma^2 + \varepsilon}}$$

ミニバッチ内のサンプルに対するuの分散

ゼロ除算を避けるための微小な正数



☹uが（偶然）正に偏った場合、非線形性が生かせない

⇔☺標準化すれば正負にまたがるので非線形

バッチ正規化：ユニットが１つの場合

①活性値uを**標準化**（＝平均０、分散１になるように変換）

ミニバッチ内のサンプルに対するuの平均

$$\hat{u} = \frac{u - \mu}{\sqrt{\sigma^2 + \varepsilon}}$$

ミニバッチ内のサンプルに対するuの分散

ゼロ除算を避けるための微小な正数

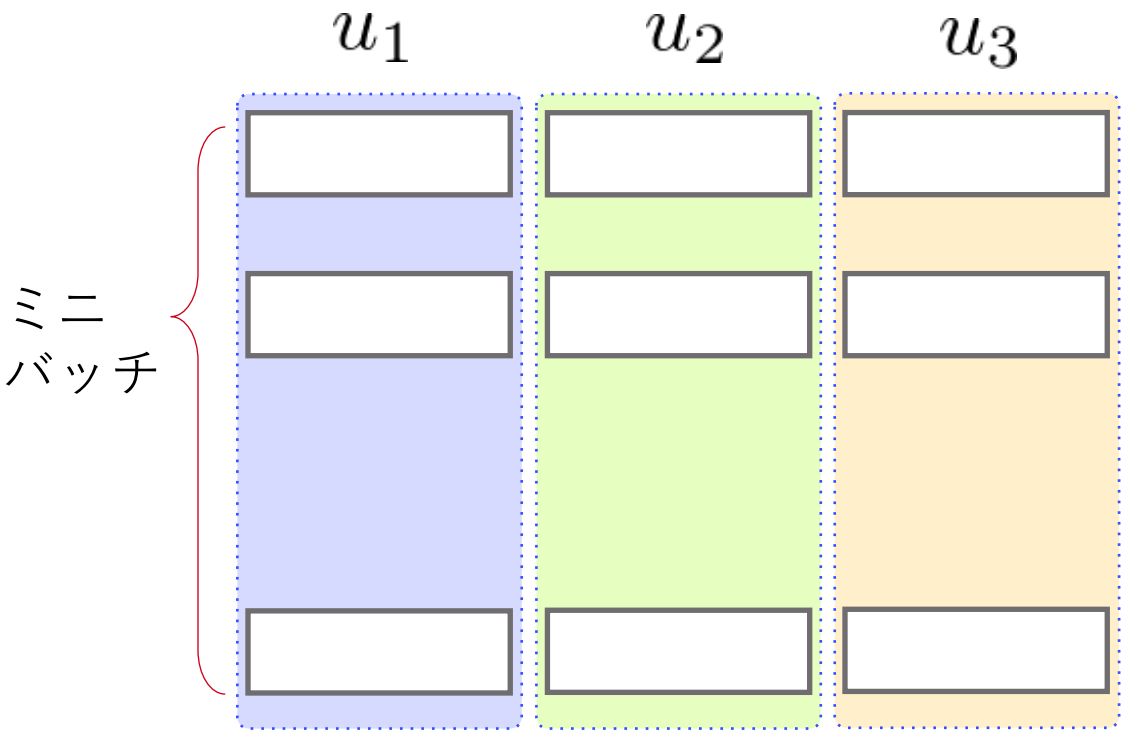
②活性値uに対するバッチ正規化の定義

$$f_{\mathrm{BN}}(u) = \gamma\hat{u} + \beta$$

学習パラメータ

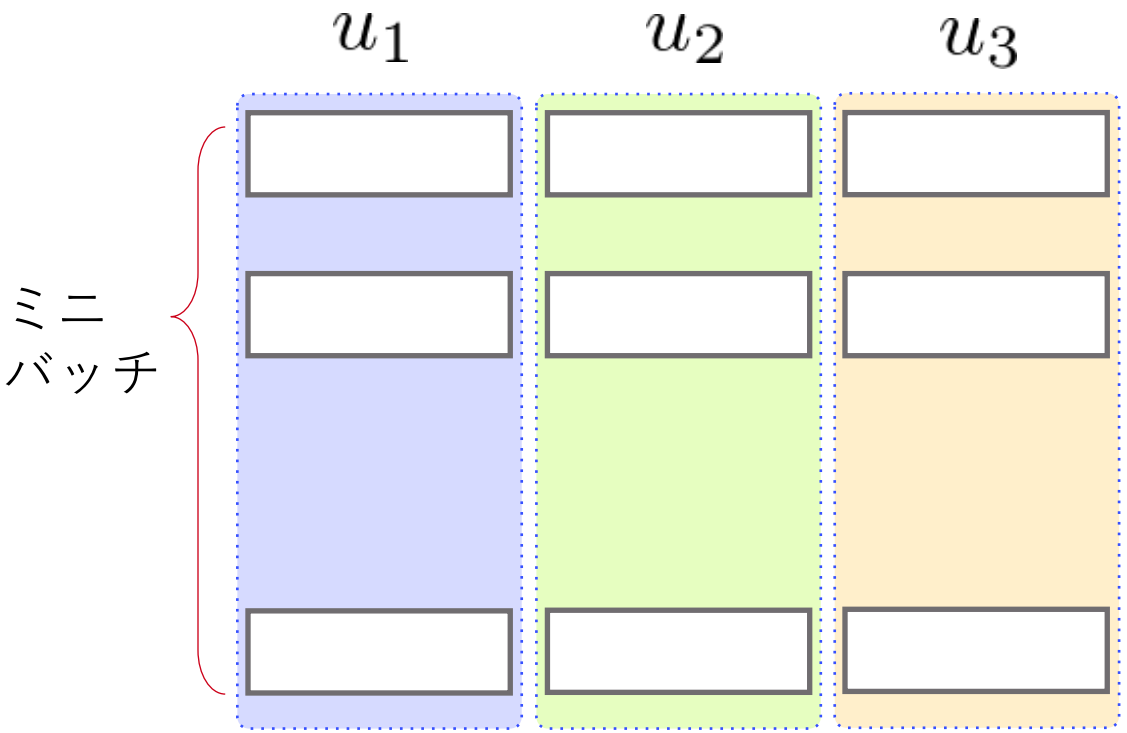
バッチ正規化：ユニットが複数の場合

- バッチ正規化では u_i ごとに標準化



バッチ正規化の注意点

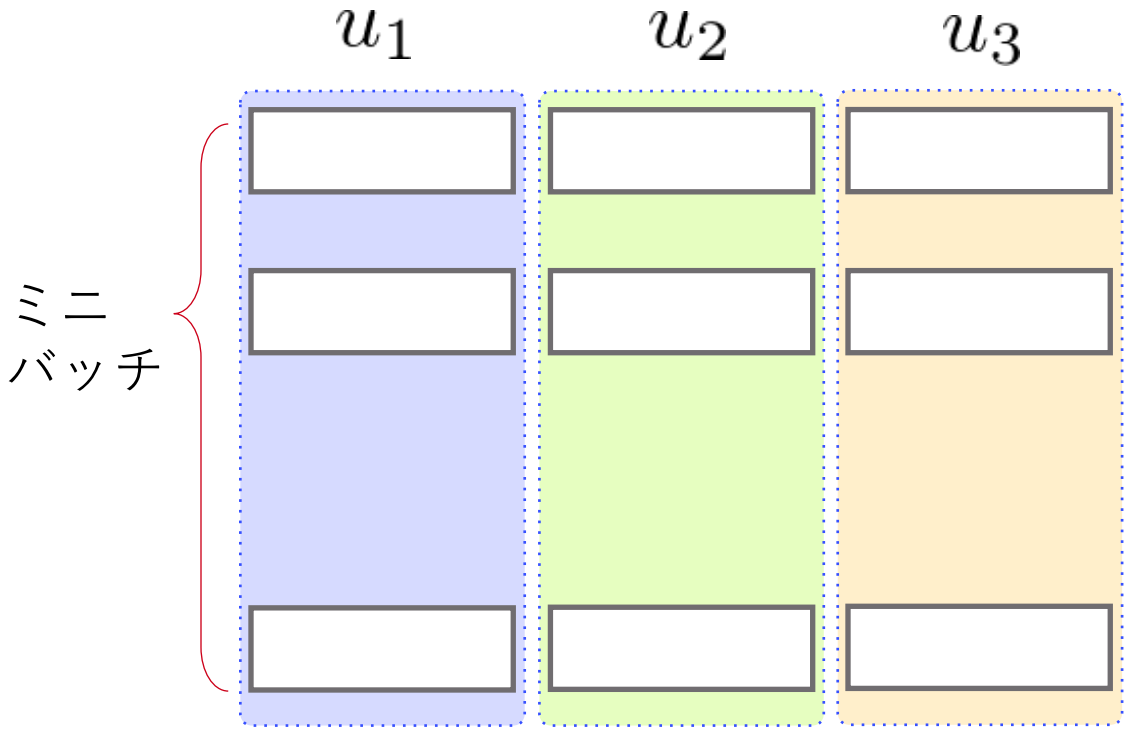
- バッチ正規化では u_i ごとに標準化



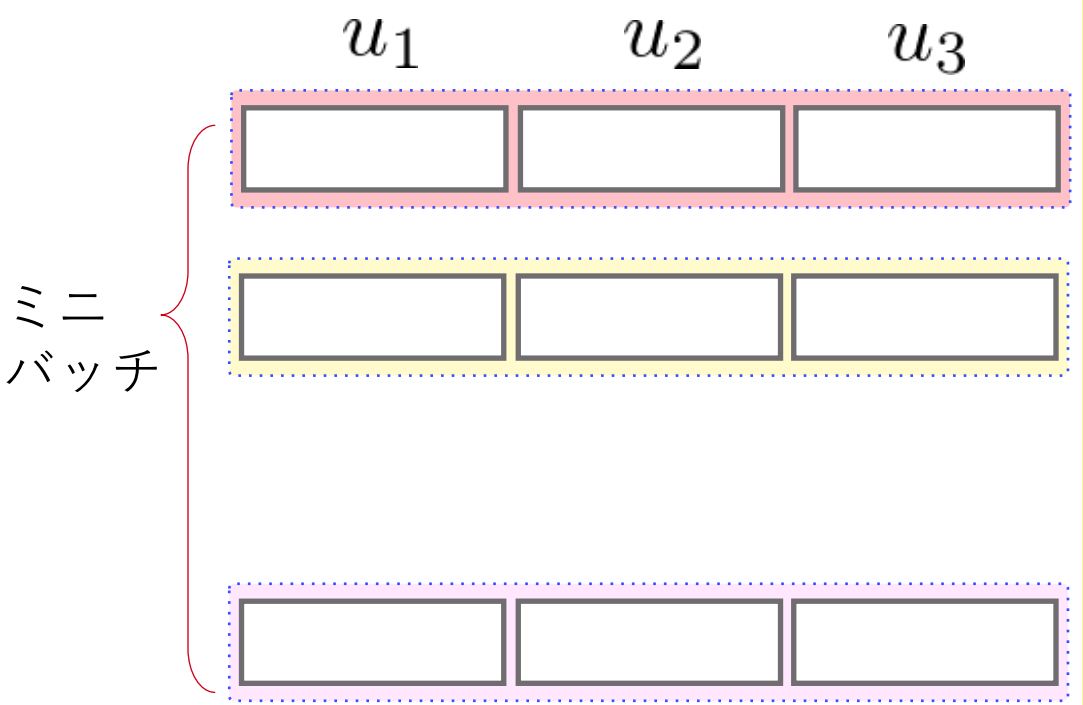
- バッチサイズが小さい場合には平均・分散が信頼できない
- スケール情報を保持したいときは使用を避ける
  - 例：回帰問題の最終層、Softmax関数の前
- 「推論時の(μ, σ^2)」＝「訓練時の(μ, σ^2)の平均」と仮定することが多い
  → ☹仮定が正しいとは限らない

レイヤー正規化（Layer normalization）

- バッチ正規化では u_i ごとに標準化



- レイヤー正規化では各サンプルに関して標準化
  →バッチサイズに依存しない



参考文献

1. 金谷 健一, これなら分かる最適化数学―基礎原理から計算手法まで, 共立出版, 2005.
2. https://qiita.com/omiita/items/1735c1d048fe5f611f80
3. Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. JMLR, 12(7).
4. Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.
5. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
6. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Proc. AISTATS (pp. 249-256).
7. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proc. IEEE ICCV (pp. 1026-1034).

参考文献

1. A., Burda, Y., Edwards, H., Babuschkin, I., & Misra, V. (2022). Grokking: Generalization beyond overfitting on small algorithmic datasets. arXiv preprint arXiv:2201.02177.

実習

実習

実習の目的

- コーディングと基礎理論の関係を学ぶ

実習課題の場所

- K-LMSから辿る

実習に関する質問

- ChatGPTに説明させる
- 教科書で調べる・検索・周囲と相談（私語禁止ではありません）
- 上記で解消しなければ挙手

付録

数値的に近似解を求めるための代表的な方法：勾配降下法

- **勾配降下法**（gradient descent method） or **最急降下法**（steepest descent method）

1. 初期値 $\boldsymbol{w}^{(0)}$ を用意

2. 更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E(\boldsymbol{w}^{(t)})$$

$\boldsymbol{w}^{(t+1)}$：更新回数

η：**学習率**（learning rate; lr）

- **勾配**（gradient）

$$\nabla E(\boldsymbol{w}) \triangleq \left(\frac{\partial E(\boldsymbol{w})}{\partial w_1}, \frac{\partial E(\boldsymbol{w})}{\partial w_2}, \ldots, \frac{\partial E(\boldsymbol{w})}{\partial w_D} \right)$$

- $E(\boldsymbol{w}) = 0.1w_1^2 + 2w_2^2$ とする。
以下の条件のとき $\boldsymbol{w}^{(1)}$ を求めよ

$$\boldsymbol{w}^{(0)} = (-5, -2),\ \eta = 0.1$$

数値的に近似解を求めるための代表的な方法：勾配降下法

■ **勾配降下法**（gradient descent method） or **最急降下法**（steepest descent method）

1. 初期値 $\boldsymbol{w}^{(0)}$ を用意

2. 更新則

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla E(\boldsymbol{w}^{(t)})$$

更新回数

学習率（learning rate; lr）

■ **勾配**（gradient）

$$\nabla E(\boldsymbol{w}) \triangleq \left(\frac{\partial E(\boldsymbol{w})}{\partial w_1}, \frac{\partial E(\boldsymbol{w})}{\partial w_2}, \cdots, \frac{\partial E(\boldsymbol{w})}{\partial w_D} \right)$$

■ $E(\boldsymbol{w}) = 0.1w_1^2 + 2w_2^2$ とする。
以下の条件のとき $\boldsymbol{w}^{(1)}$ を求めよ

$$\boldsymbol{w}^{(0)} = (-5, -2),\ \eta = 0.1$$

$$\nabla E(\boldsymbol{w}) = (0.2w_1, 4w_2)$$

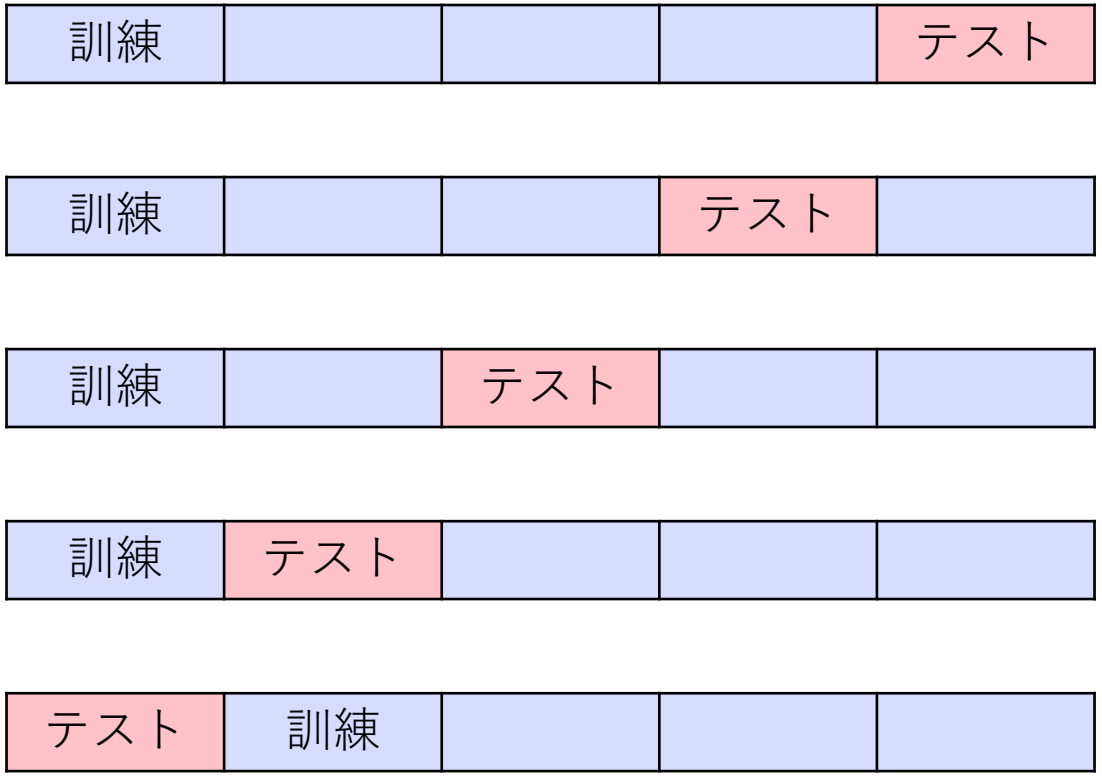
$$\boldsymbol{w}^{(1)} = (-5, -2) - 0.1(-1, -8)$$

$$= (-4.9, -1.2)$$

交差検証（cross-validation）
訓練集合とテスト集合に分ける場合

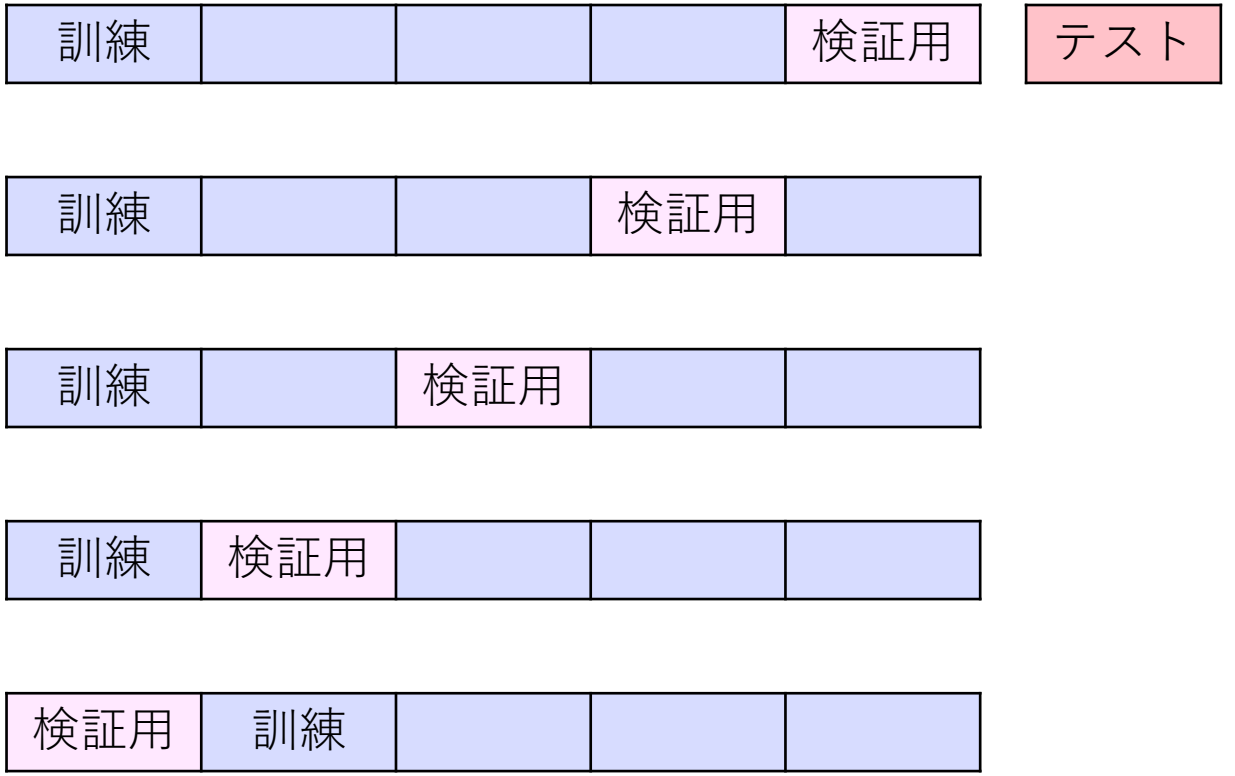
- 背景
  - テスト集合が一定だと結果が偏る可能性がある
- N-fold cross-validation
  - 右図のような集合に対して学習を行い、性能はN回の平均とする
- N=5または4が多い

- 5-fold cross-validationの例



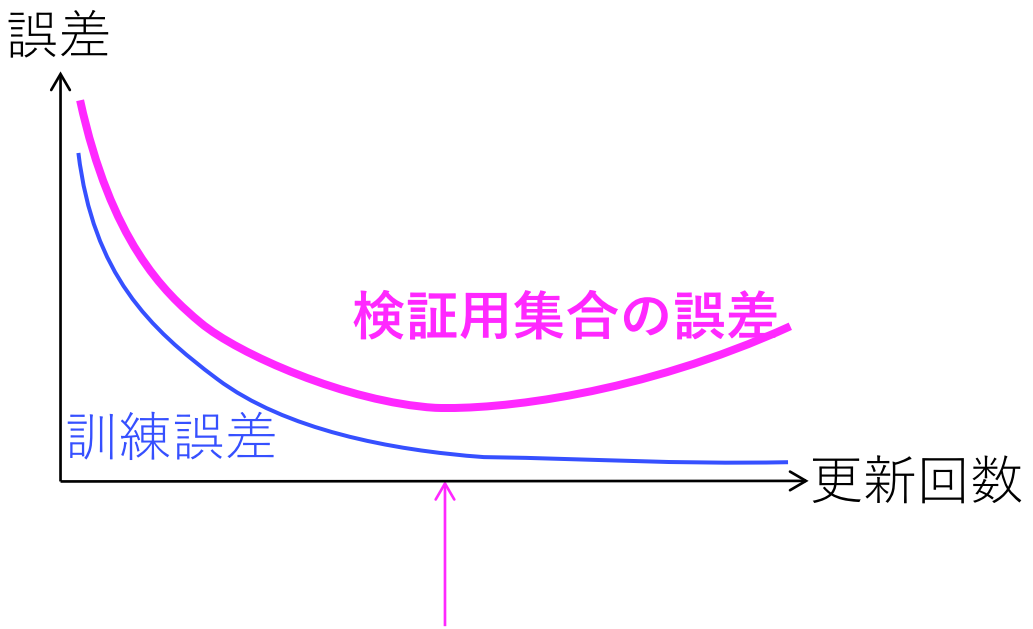
交差検証（cross-validation）
訓練・検証用・テスト集合を用いる場合

- モデル選択
  - 層数、層のユニット数等
- その他のハイパーパラメータ
  - 学習率等
- 手順
  1. N-fold cross-validationに基づき、モデルやハイパーパラメータ等を決定
  2. 訓練集合と検証用集合を合わせて１のモデルを再度訓練
  3. 未使用のテスト集合で２のモデルを性能評価



早期終了の基準例

- 例
  - Nエポック以内で検証集合に関する誤差が最小
  - Nエポック以内で検証集合に関する精度が最大
  - Nエポック間、検証集合に関する誤差関数が増加



検証用集合の誤差を最小とするモデルで**テスト誤差を評価**することで過学習を避ける

※テスト集合の誤差を最小とするモデルを選択するのはチート

ノイズ付加（noise injection）

- 背景
  - 入力が微小なノイズで乱された場合にも正しく予測できることが望ましい
- 学習時に入力 $\boldsymbol{x}$ に対してガウス分布に基づくノイズ $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \epsilon^2 I)$ を付加

各次元が（平均 0 , 標準偏差 ε ）であるガウス分布から独立に得られた

- 正則化と理論的にほぼ同等であることが知られている[Sietsma+ 1991]

パラメータの初期値

パラメータの初期値

- 初期値をすべて0に設定すると
  学習がうまく進まないことが多い

↓

一様分布やガウス分布からサンプルする＝「ランダムに初期化」

↓

分布の分散の選び方が学習の成否につながる

← 勾配が同じになるので、全ノードが「コピー状態」になる

パラメータの初期値

- 初期値をすべて0に設定すると
  学習がうまく進まないことが多い

↓

一様分布やガウス分布からサンプルする＝「ランダムに初期化」

↓

分布の分散の選び方が学習の成否につながる

- 一様分布の例

$$x \sim \mathcal{U}(-1, 1)$$

本講義では「分布からサンプルする」ことを意味する※

口語で言うと「-1以上1以下のランダムな数字を取ってきた」

※本来はX～P(x)のように記述し、「確率変数Xは分布P(x)に従う」を意味するが、上記の用法で使われることが多い

パラメータの初期値
Xavierの初期化 [Glorot+ 2010]

■ Xavierの初期化
（Xavier initialization）

l番目の隠れ層への重み

$$w_{ji}^{(l)} \sim \mathcal{U}\left(-\sqrt{\frac{6}{d_{l-1}+d_l}}, \sqrt{\frac{6}{d_{l-1}+d_l}}\right)$$

l番目の隠れ層のノード数

■ 一様分布の例

$$x \sim \mathcal{U}(-1, 1)$$

本講義では「分布からサンプルする」ことを意味する※

口語で言うと「-1以上1以下のランダムな数字を取ってきた」

※本来はX～P(x)のように記述し、「確率変数Xは分布P(x)に従う」を意味するが、上記の用法で使われることが多い

パラメータの初期値
Heの初期化 [He+ 2015]

■ Xavierの初期化
（Xavier initialization）

l番目の隠れ層への重み

$$w_{ji}^{(l)} \sim \mathcal{U}\left(-\sqrt{\frac{6}{d_{l-1}+d_l}}, \sqrt{\frac{6}{d_{l-1}+d_l}}\right)$$

l番目の隠れ層のノード数

■ Heの初期化
（He initialization）

$$w_{ji}^{(l)} \sim \mathcal{U}\left(-\sqrt{\frac{6}{d_{l-1}}}, \sqrt{\frac{6}{d_{l-1}}}\right)$$

- Xavier（ゼイビア）はfirst nameだが、
  He（ヒー）はfamily nameである。
  一貫していないが慣習に従う
- 発音にも混乱がある（本講義では講演音声から聞き取った発音を使う）