KYOTO UNIVERSITY

統計的モデリング基礎⑥
～正則化と事後確率最大化～

鹿島久嗣
（情報学科 計算機科学コース）

DEPARTMENT OF INTELLIGENCE SCIENCE
AND TECHNOLOGY

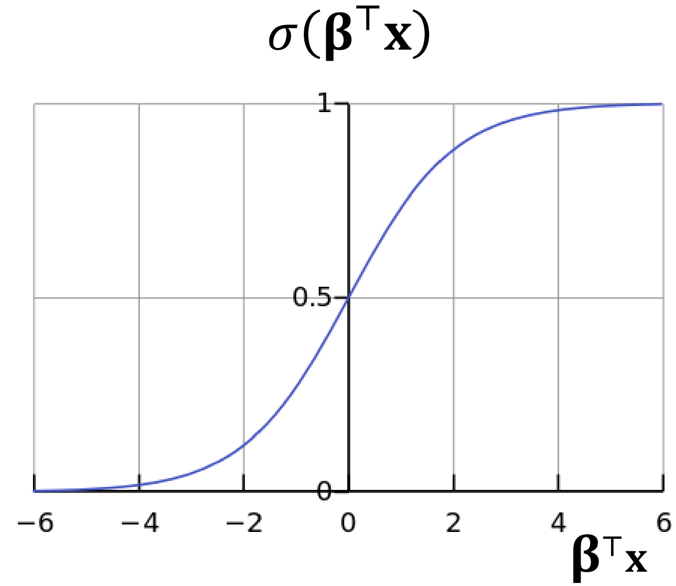
計算グラフと自動微分

ロジスティック回帰：ダミー変数を従属変数とするモデル

- 以前、重回帰モデルでダミー変数を従属変数とすると、厳密には少しおかしいという話だった → もっときちんと扱いたい
  - 重回帰モデル$y = \boldsymbol{\beta}^\top \mathbf{x}$の従属変数の値域は実数全体
- 従属変数の値域が$\{-1, +1\}$もしくは$(0,1)$（$Y = +1$となる確率）となるようにしたい
- ロジスティック回帰モデル：

$$P(Y = 1 | \mathbf{x}, \boldsymbol{\beta}) = \frac{1}{1+\exp(-\boldsymbol{\beta}^\top \mathbf{x})} = \sigma(\boldsymbol{\beta}^\top \mathbf{x})$$

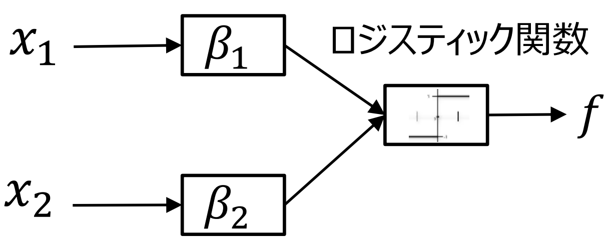
  - σ：ロジスティック関数（$\sigma: \mathbb{R} \to (0,1)$）



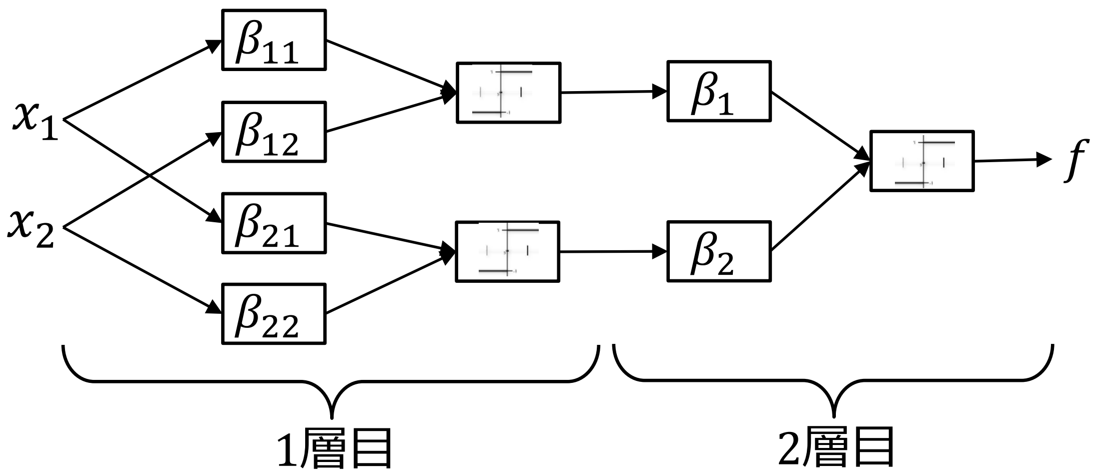
ニューラルネットワーク：（ざっくりいえば）ロジスティック回帰モデルを連結したもの

- ニューラルネットワークはロジスティック回帰モデルを連結したもの
  - 複数のロジスティック回帰モデルの出力が、別のロジスティック回帰モデルの入力になる
  - ロジスティック関数（非線形）によりモデルに非線形性を導入
  - 両者ともに、$y = +1$である確率$f(\mathbf{x}; \boldsymbol{\beta})$を出力するモデル

ロジスティック回帰モデル

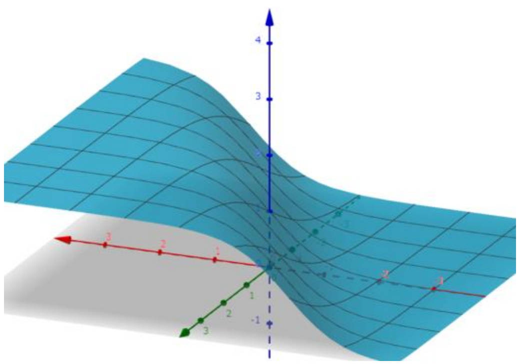


ニューラルネットワーク（2層）

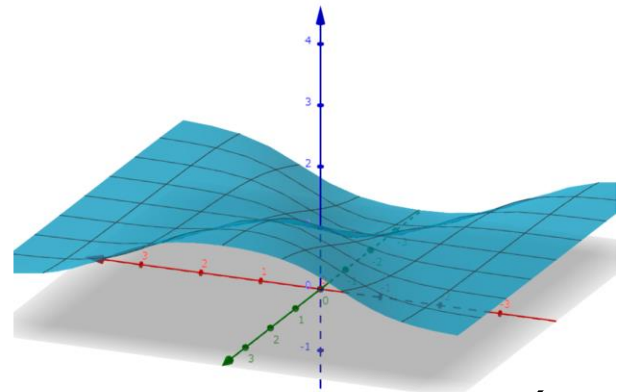


ニューラルネットワークの非線形性の例：ロジスティック回帰を2層積むと非線形分類が可能

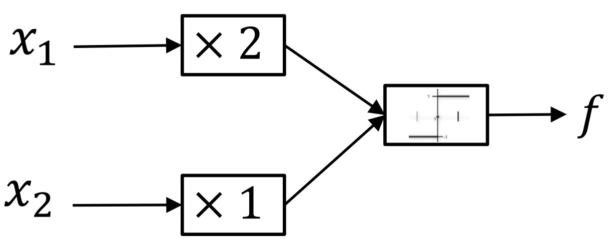
- ロジスティック回帰は1層では線形判別しかできない（AND/OR）
- 2層以上積むことで非線形の表現力を獲得（XOR）

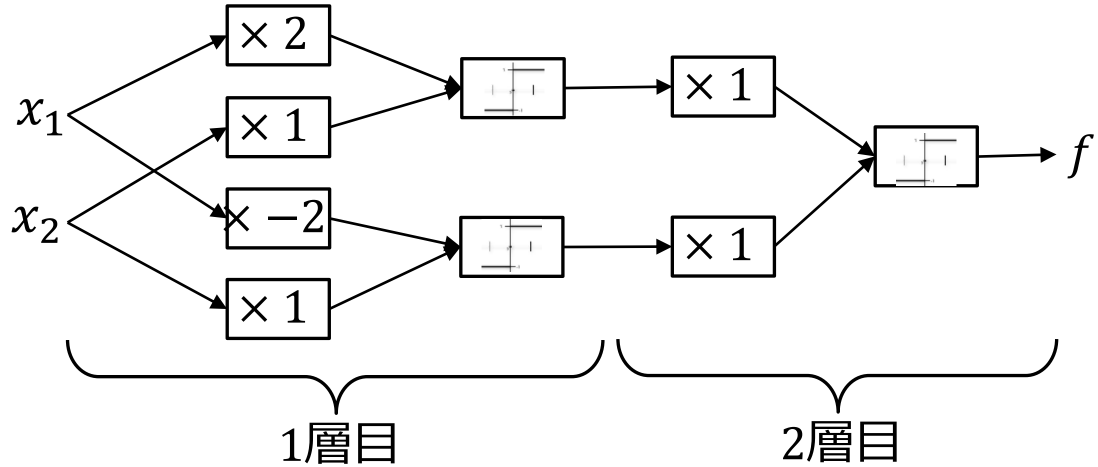


ロジスティック回帰モデル



ニューラルネットワーク（2層）





ニューラルネットワークのパラメータ推定：
最急降下法を適用するために勾配の計算が必要

- 対数尤度関数$L(\boldsymbol{\beta})$を最大化するパラメータ$\boldsymbol{\beta}$を求める：

$$L(\boldsymbol{\beta}) = -\sum_{i=1}^{n}\left(\delta\left(y^{(i)} = 1\right)\log f\left(x^{(i)}\right) + \delta\left(y^{(i)} = -1\right)\log\left(1 - f\left(x^{(i)}\right)\right)\right)$$

  - $f\left(x^{(i)}\right)$は$x^{(i)}$に対するニューラルネットの出力（$y^{(i)} = 1$である確率）

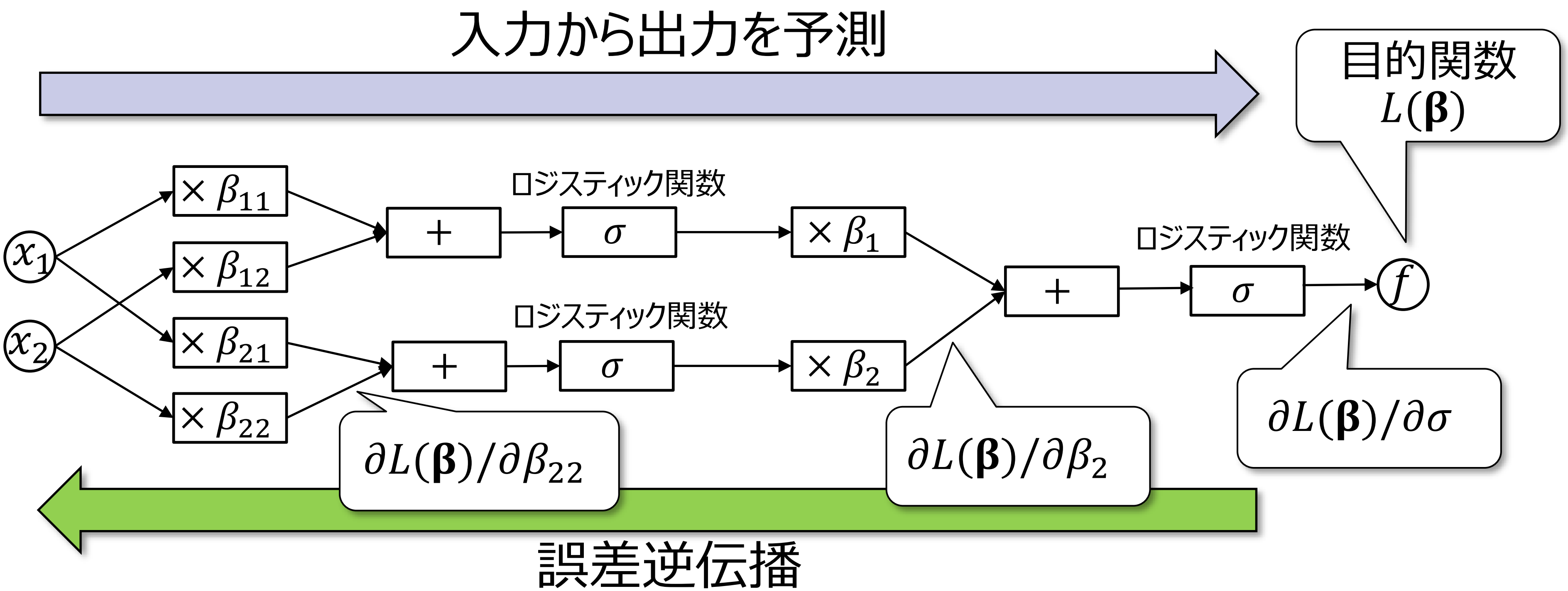
- 勾配$\nabla L(\boldsymbol{\beta}) = \dfrac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$が計算できれば最急降下法を適用できる：

$$\boldsymbol{\beta}^{\mathrm{NEW}} \leftarrow \boldsymbol{\beta} + \eta \nabla L(\boldsymbol{\beta})$$

  - 実際は確率的最適化やミニバッチを用いることも多い

ニューラルネットワークのパラメータ推定法：
誤差逆伝播法による勾配法で効率的にパラメータ推定

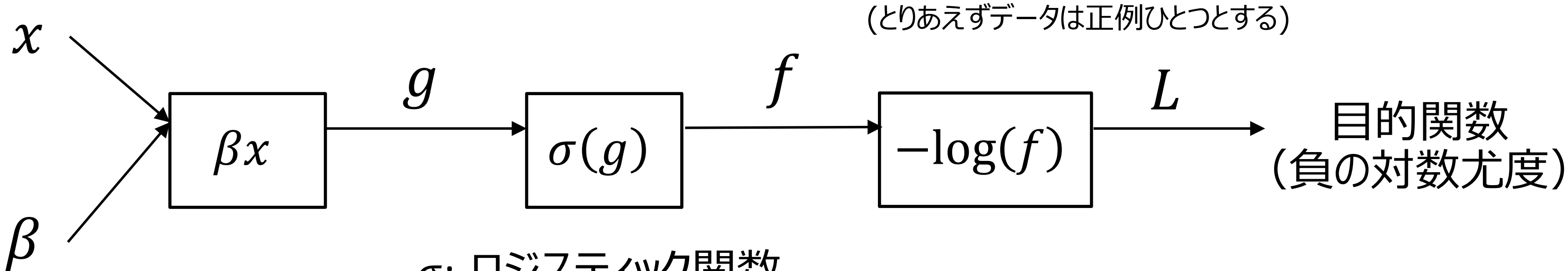
- 対数尤度$L(\boldsymbol{\beta})$をパラメータで微分したい
- 誤差逆伝播法（自動微分）：
  層を遡って再帰的に微分計算する効率的な計算法



計算グラフ：
ニューラルネットの入力から出力までの計算を図示

- 計算グラフ：関数の入出力の間の関係を、単純な計算ユニットをつないで表したもの
- 計算グラフをたどりながら、入力に順番に単純な操作（重み付き和やロジスティック関数の適用など）を適用していくと、出力が得られる
- ロジスティック回帰の計算グラフ：

ロジスティック回帰の出力： $f = \sigma(\beta x)$
最適化問題の目的関数：$L = -\log f$
(とりあえずデータは正例ひとつとする)



σ: ロジスティック関数

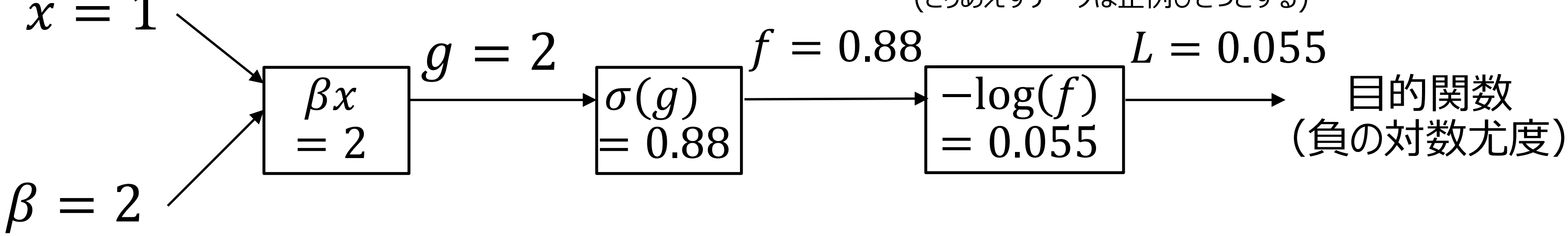
計算グラフ：
ニューラルネットの入力から出力までの計算を図示

- 計算グラフ：関数の入出力の間の関係を、単純な計算ユニットをつないで表したもの

- 計算グラフをたどりながら、入力に順番に単純な操作（重み付き和やロジスティック関数の適用など）を適用していくと、出力が得られる

- ロジスティック回帰の計算グラフ：

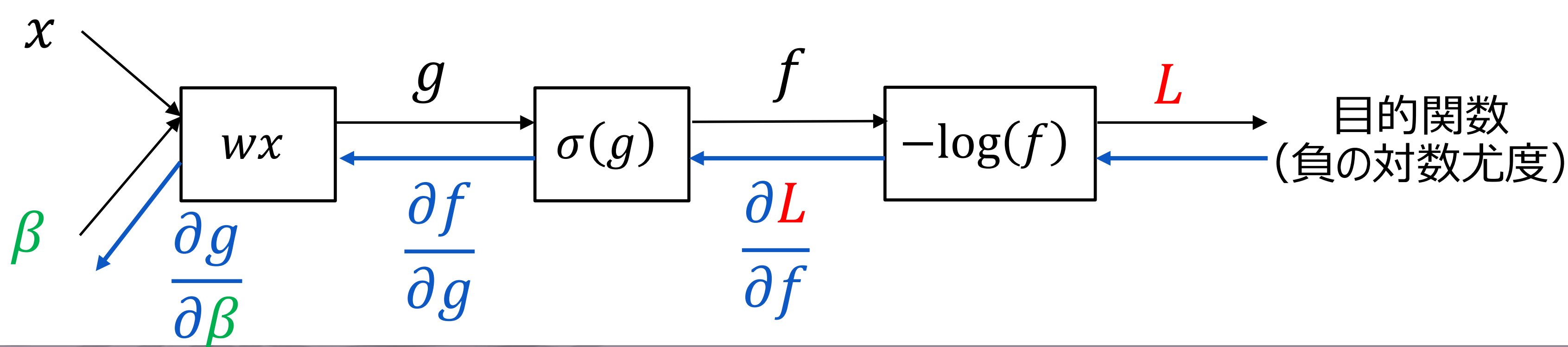
ロジスティック回帰の出力： $f = \sigma(\beta x)$
最適化問題の目的関数：$L = -\log f$
(とりあえずデータは正例ひとつとする)



σ: シグモイド関数

計算グラフ上での自動微分：
計算グラフを出力から逆向きにたどることで勾配計算

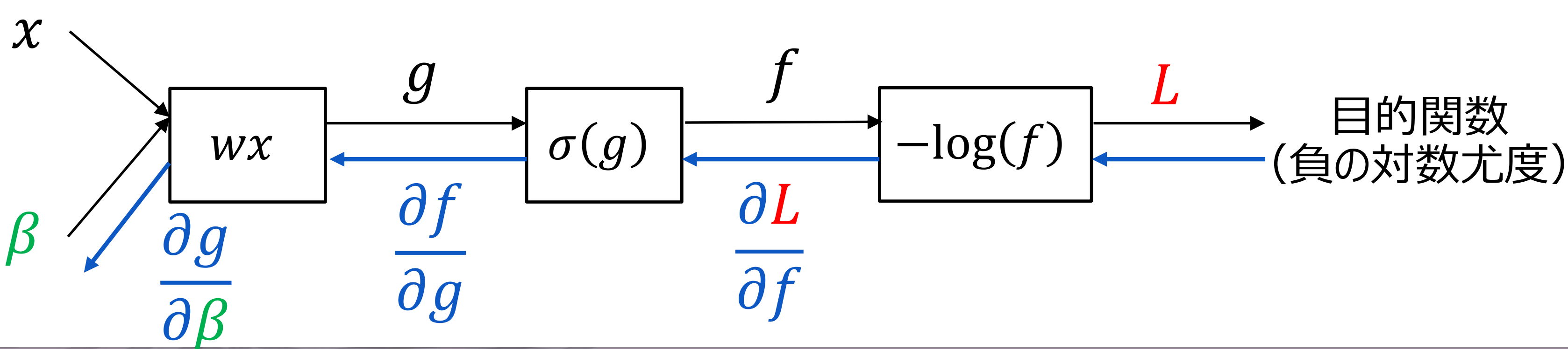
- 勾配計算：$\partial L / \partial \beta$を求めたい
- 計算グラフ上でLとβは遠い
- 計算グラフを逆向きにたどりながら、微分を計算する
  - ロジスティック回帰の場合： $\dfrac{\partial L}{\partial \beta} = \dfrac{\partial g}{\partial \beta} \cdot \dfrac{\partial f}{\partial g} \cdot \dfrac{\partial L}{\partial f}$



自動微分のポイント：
微分可能な計算ユニット

- 自動微分：計算グラフを逆向きにたどりながら（微分の連鎖律によって）微分を計算する

- 各ユニットは、入力について微分可能である必要がある
  - $\frac{\partial f}{\partial g} = \frac{\partial \sigma(g)}{\partial g} = \sigma(g)\big(1 - \sigma(g)\big)$ （ロジスティック関数の微分）
  - $\frac{\partial g}{\partial \beta} = \frac{\partial \beta x}{\partial \beta} = x$



ニューラルネットワーク推定のポイント：
微分可能なユニットを組みあわせて自動微分にまかせる

- ニューラルネット推定法の汎用性

1. ネットワークを「計算グラフ」で記述する
   - 各ユニットはパラメータや入力について微分可能とする
2. 誤差逆伝播で自動的に勾配が計算できる
   （自動微分）

正則化

重回帰モデルの復習：最小二乗法による定式化

- 重回帰モデル：$y = \boldsymbol{\beta}^\top \mathbf{x}$
  - パラメータ：$\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_m, \alpha)^\top$
  - 独立変数：$\mathbf{x} = (x_1, x_2, \dots, x_m, 1)^\top$

  最後の次元は切片部分に相当
- データ：
  - 計画行列：$\boldsymbol{X} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(n)}\right)^\top$
  - 従属変数：$\mathbf{y} = \left(y^{(1)}, y^{(2)}, \dots, y^{(n)}\right)^\top$
- 目的関数：$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^\top \mathbf{x}^{(i)}\right)^2 = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$

重回帰モデルの解：解析解が得られる

- 目的関数：$L(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 = (\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})^\top(\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta})$
- 解: $\boldsymbol{\beta}^* = \mathrm{argmin}_{\boldsymbol{\beta}}\, L(\boldsymbol{\beta}) = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\mathbf{y}$
- ただし、解が存在するためには$\boldsymbol{X}^\top\boldsymbol{X}$が正則である必要
  - モデルの次元数mよりもデータ数nが大きい場合はおおむね成立
- 正則化：正則でない場合には$\boldsymbol{X}^\top\boldsymbol{X}$の対角成分に正の定数$\lambda > 0$を加えて正則にする
  - 新たな解: $\boldsymbol{\beta}^* = \mathrm{argmin}_{\boldsymbol{\beta}}\, L(\boldsymbol{\beta}) = (\boldsymbol{X}^\top\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^\top\mathbf{y}$
  - 目的関数に戻ると：$L(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2$

パラメータのノルムに関するペナルティ項

データへの過適合：
データへの過剰な適合は将来のデータへの予測力を損なう

- 先ほどは、正則化を計算を安定させるために導入した
- 例えば、データ数nが次元数mより小さいとき、重回帰の解は一意に定まらない
  - 任意の数の解が存在し、どれが良いのかわからない
- データへの過適合：
  - 汎化：予測を目的とする場合、我々の真の目的は将来のデータへの予測力が高いモデルを得ること
  - 手持ちのデータへのモデルの過剰な適合は、将来の予測力を損なう可能性がある

オッカムの剃刀：
できるだけ"単純な"なモデルを採用せよ

- データに同程度適合している無数のモデルのうち、どれが最も"良い" モデルだろうか
- オッカムの剃刀：単純なモデルを採用せよ
  - 「ある事柄を説明するためには、必要以上に多くを仮定するべきでない」
- 単純さを何で測るか？
  - 例えば、モデルに含まれる独立変数の数
  - 自由度調整済決定係数、AIC、BICなどの情報量基準

0-ノルムを用いた正則化：パラメータ中の非零成分の数を減らす

- モデルに含まれる独立変数の数＝ $\boldsymbol{\beta}$ 中の非零成分の数 ＝ $\boldsymbol{\beta}$ の0-ノルム

モデルに含まれる
独立変数の数

- 0-ノルム制約を入れた回帰問題：

$$\boldsymbol{\beta}^* = \mathrm{argmin}_{\boldsymbol{\beta}} \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 \ \ \mathrm{s.t.}\ \|\boldsymbol{\beta}\|_0 \leq \eta$$

- あるいは0-ノルムをペナルティ項として導入：

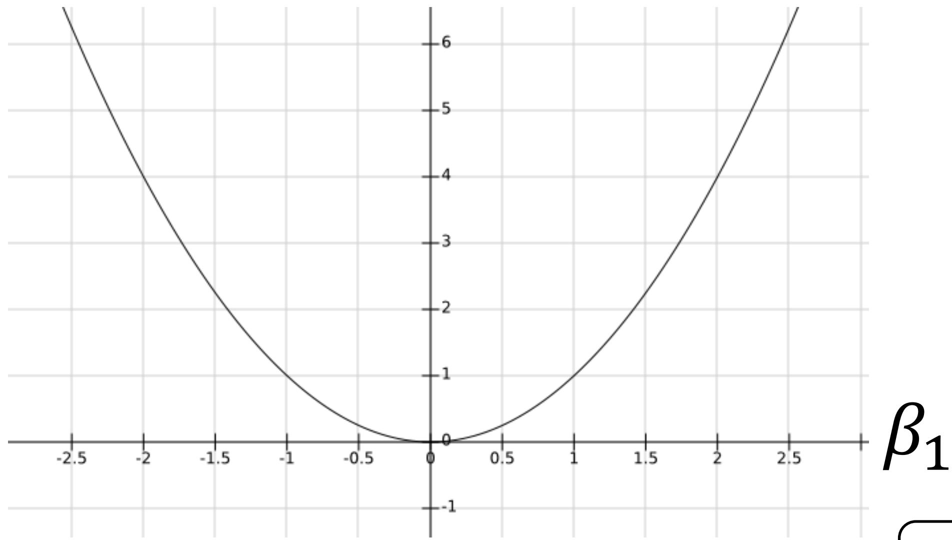
$$\boldsymbol{\beta}^* = \mathrm{argmin}_{\boldsymbol{\beta}} \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0$$

  - η と λ は一対一対応している

- ただし、この問題は凸最適化問題でない

0-ノルムの代替：
2-ノルム正則化は凸最適化になる

- 0-ノルム $\|\boldsymbol{\beta}\|_0$ の代わりに 2-ノルム $\|\boldsymbol{\beta}\|_2^2$ を用いる



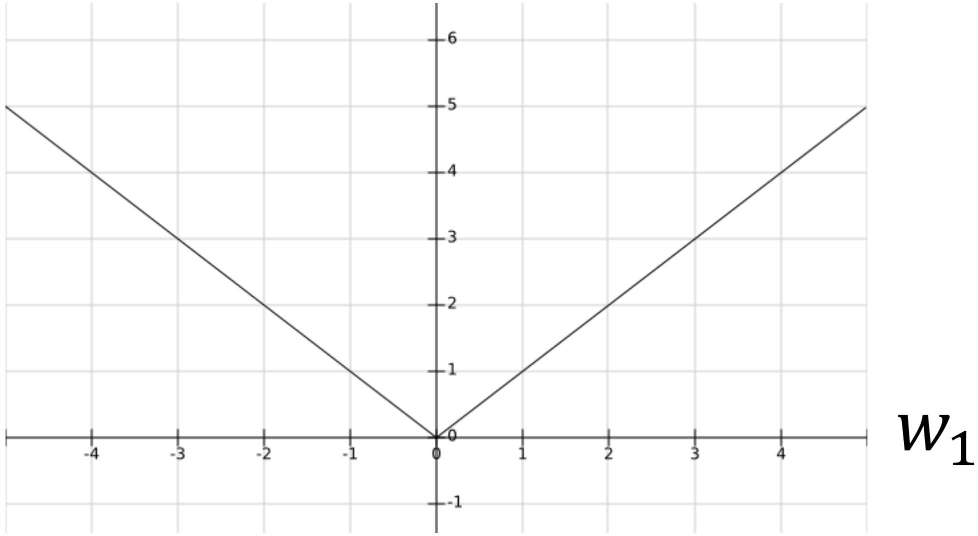
- リッジ回帰：$L(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2$ 　凸 ☺
  - 0-ノルム正則化の緩和版として捉えることができる：

$$L(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0$$

非凸 ☹

0-ノルムの代替：
1-ノルム正則化も凸最適化になる

- さらに、1-ノルム $\|\boldsymbol{\beta}\|_1 = |\beta_1| + |\beta_2| + \cdots + |\beta_D|$ も利用可能



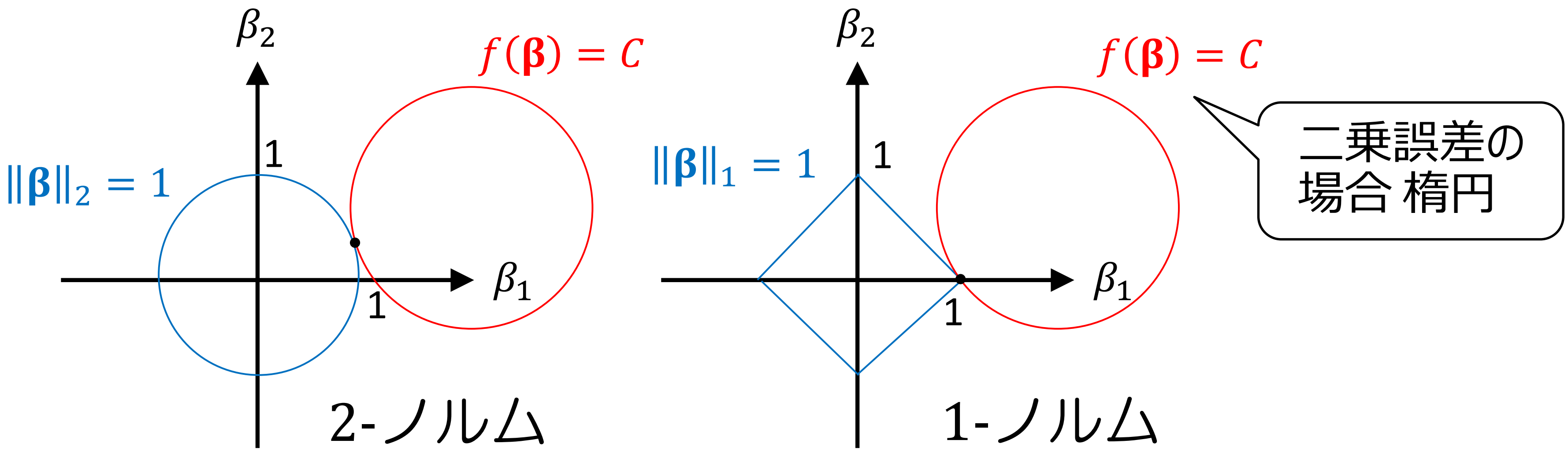
- ラッソ：$L(\boldsymbol{\beta}) = \|\mathbf{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1$
  - 凸最適化だが、解析解はもたない
- 1-ノルムを用いると疎な解になる（$\boldsymbol{\beta}^*$の多くの要素が0になる）

1-ノルム正則化の利点：
疎な解をもつ

- 1-ノルム正則化は1-ノルム制約で書き直せる：

$$\mathrm{argmin}_{\boldsymbol{\beta}}\, f(\boldsymbol{\beta}) + \gamma \|\boldsymbol{\beta}\|_1 \Leftrightarrow \mathrm{argmin}_{\boldsymbol{\beta}}\, f(\boldsymbol{\beta}) \text{ s.t. } \|\boldsymbol{\beta}\|_1 \le \lambda$$

- 1-ノルム正則化は疎な解をもつ
  - 2-ノルム制約の等高線（円形）と1-ノルム制約の等高線（菱形）の比較



事後確率最大化推定

回帰のベイズ統計的解釈：
事後確率最大化推定

- 線形回帰モデルの推定は最尤推定として解釈できた
- 正則化のもとでの回帰モデルの推定はベイズ統計の枠組みで解釈できる
  - 事前分布・事後分布の導入
  - 事後確率最大化（MAP）推定
  - リッジ回帰＝MAP推定

線形回帰モデルの最尤推定：
線形回帰の確率モデルの最尤推定 ＝ 最小二乗法

- 線形回帰モデルに対応する確率モデル
  - 確率密度関数：$f\left(y^{(i)} \mid \mathbf{x}^{(i)}\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y^{(i)} - \boldsymbol{\beta}^\top \mathbf{x}^{(i)}\right)^2}{2\sigma^2}\right)$
- 対数尤度：$L(\boldsymbol{\beta}) = \sum_{i=1}^{n} \log f\left(y^{(i)} \mid \mathbf{x}^{(i)}\right)$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^\top \mathbf{x}^{(i)}\right)^2 + \text{const.}$$

- 対数尤度を$\boldsymbol{\beta}$について最大化すること（最尤推定）
＝二乗誤差を$\boldsymbol{\beta}$について最小化すること（最小二乗法）

ベイズ的統計モデリングの考え方：尤度の代わりに事後分布を考える

- 最尤推定（MLE）では尤度を最大化するパラメータ**β**を求めた：

$$P(\mathbf{y} \mid \boldsymbol{X}, \boldsymbol{\beta}) = \prod_{i=1}^{n} f(y^{(i)} | \mathbf{x}^{(i)}, \boldsymbol{\beta})$$

  - これは、パラメータが与えられたもとでデータが再現される条件付確率: $P\left(\text{データ} \mid \text{パラメータ}\right) = P(\mathbf{y} \mid \boldsymbol{X}, \boldsymbol{\beta})$

    ※ 回帰の場合は$\boldsymbol{X}$が定数として与えられ、$\mathbf{y}$のみが確率変数と思っている

- ベイズ的なモデリングの考え方では、事後分布を考える：

$$P\left(\text{パラメータ} \mid \text{データ}\right) = P(\boldsymbol{\beta} \mid \boldsymbol{X}, \mathbf{y})$$

  - 事後分布はパラメータを確率変数と考える

事後分布：
事後分布 ∝ 尤度 × 事前分布

- 事後分布：

$$P(\text{パラメータ} \mid \text{データ}) = \frac{P(\text{データ} \mid \text{パラメータ})P(\text{パラメータ})}{P(\text{データ})}$$

ベイズの定理

パラメータに依存しない
（定数扱い）

  - $P(\text{データ}) = \sum_{\text{パラメータ}} P(\text{データ} \mid \text{パラメータ})P(\text{パラメータ})$

- 対数事後分布：

$$\log P(\text{パラメータ} \mid \text{データ})$$
$$= \underbrace{\log P(\text{データ} \mid \text{パラメータ})}_{\text{尤度}} + \underbrace{\log P(\text{パラメータ})}_{\text{事前分布}} + \text{const.}$$

事後確率最大化（MAP）推定：事後確率を最大化するパラメータを採用

- 対数事後分布：
  $$\log P(\text{パラメータ} \mid \text{データ})$$
  $$= \log P(\text{データ} \mid \text{パラメータ}) + \log P(\text{パラメータ}) + \text{const.}$$
- 事後確率最大化（Maximum a posteriori; MAP）推定
  - 事後確率を最大化するパラメータを採用する：
    $$\text{パラメータ}^* = \operatorname{argmax}_{\text{パラメータ}} \log P(\text{パラメータ} \mid \text{データ})$$
  - 最尤推定では $\log P(\text{データ} \mid \text{パラメータ})$ の項のみ考える
  - 追加の項として事前分布の項：$\log P(\text{パラメータ})$

事後確率最大化としてのリッジ回帰：正規分布を事前分布とした事後確率最大化

- 対数事後分布：

$$\log P(\text{パラメータ} \mid \text{データ}) = \log P(\text{データ} \mid \text{パラメータ}) + \log P(\text{パラメータ}) + \text{const.}$$

対数尤度

対数事前分布

- リッジ回帰：

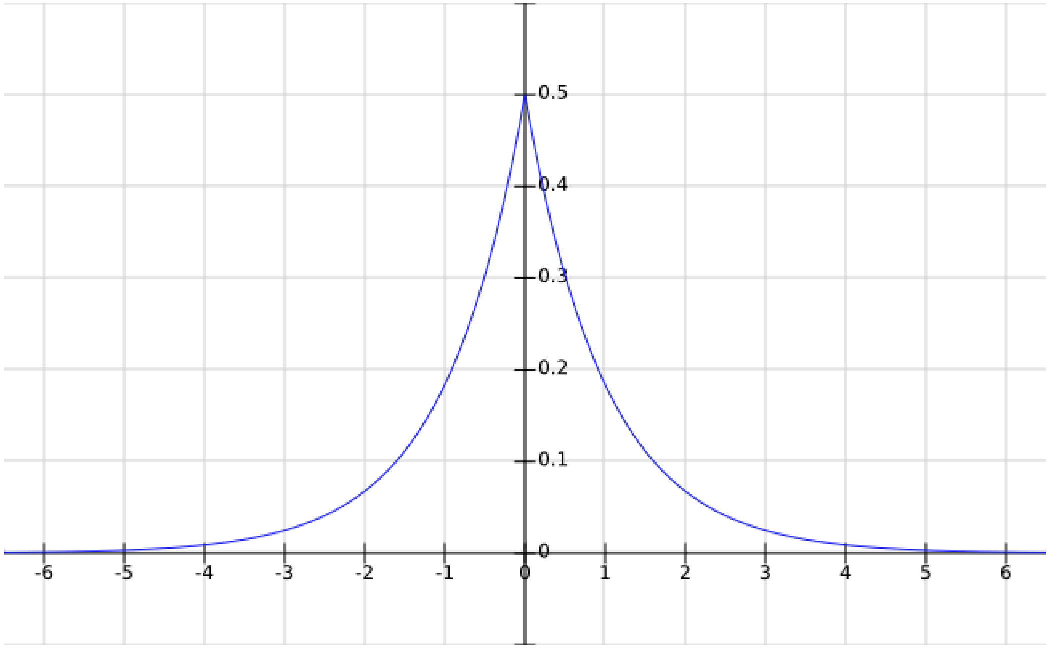
$$\boldsymbol{\beta}^* = \operatorname{argmin}_{\boldsymbol{\beta}} \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^\top \mathbf{x}^{(i)} \right)^2 + \frac{1}{2\sigma'^2} \|\boldsymbol{\beta}\|_2^2$$

- 対数尤度：$\sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi}\sigma'} \exp\left(-\frac{\left(y^{(i)} - \boldsymbol{\beta}^\top \mathbf{x}^{(i)} \right)^2}{2\sigma'^2} \right)$
- 事前分布：$P(\boldsymbol{\beta}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\boldsymbol{\beta}^\top \boldsymbol{\beta}}{2\sigma^2} \right)$

事後確率最大化としてのラッソ：ラプラス分布を事前分布として利用

- 事前分布を正規分布にすると2-ノルム正則化
- ラプラス分布：1-ノルム正則化に対応する事前分布

$$P(\boldsymbol{\beta}) = \frac{1}{2\phi}\exp\left(-\frac{|\boldsymbol{\beta}|}{\phi}\right)$$



ベイズ予測：
推定のばらつきを考慮した予測

- MAP推定では事後分布が最大となるパラメータを点推定する

$$\hat{\boldsymbol{\beta}} = \mathrm{argmax}_{\boldsymbol{\beta}} P(\boldsymbol{\beta} \mid \boldsymbol{X}, \mathbf{y})$$

- ベイズ予測では事後分布そのものを用いて予測する

$$P(y \mid \mathbf{x}) = \int_{\boldsymbol{\beta}} P(y \mid \mathbf{x}, \boldsymbol{\beta}) P(\boldsymbol{\beta} \mid \boldsymbol{X}, \mathbf{y}) \, \mathrm{d}\boldsymbol{\beta}$$

  - あらゆるパラメータにおけるモデルの予測を事後確率で重みづけて予測する
  - 最適化問題を解いてパラメータを点推定するのでなく「全部」使う