演習

線形回帰モデル再考

特徴選択 (subset selection)

- **線形回帰モデルをもう一度考えよう**

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$$

- **今度はモデルの解釈について考える**
  - p個の説明変数の中で**どれがどの程度**Yに効いているのか
  - 説明変数の個数がある程度多い状況を想定, **but n > p**
  - 一般に説明変数が多いと解釈が困難

- **数学的な定式化は簡単**
  - どの回帰係数が非ゼロか考えればOK

$$\texttt{Find} \;\; M = \{j = 1, 2, \ldots, p \,|\, \beta_j \neq 0\}$$

- **集合Mを知っていれば...**

$$Y = \beta_0 + \sum_{j \in M} \beta_j X_j + \varepsilon \quad \Rightarrow \quad \hat{\boldsymbol{\beta}}_M = (\boldsymbol{X}_M^T \boldsymbol{X}_M)^{-1} \boldsymbol{X}_M^T \boldsymbol{y}$$

#X_M : サンプルサイズ × (M+1)変数のデータ行列

回帰係数の推定値を見れば解釈可能！

- **集合Mをどうやって見つけるか？**
  - 特徴選択 (subset selection)
  - 正則化 (regularization)

特徵選択

特徴選択 (subset selection)

- **Best subset selection**
  - すべての組み合わせで最適なMを探す方法

1. NullモデルM_0 : 定数(切片)項のみのモデル
2. $k = 1,2,\ldots,p$ に対して
   (a) ${}_pC_k$個のモデル集合を用意する
   (b) モデル集合の中で**RSS**を最小にするモデルを見つける
   (c) それをM_kをする
3. $M_0, M_1, \ldots, M_p$から最適なモデルをCV, Cp (AIC), BICで探す

p=2で考えよう

Fullモデル $Y = \beta_0 + \beta_1 X_1 + \beta_p X_2 + \varepsilon$

1. Nullモデルの用意とRSSの計算

$$Y = \beta_0 + \varepsilon \quad M_0 = \phi \quad \hat{\beta}_0 = \bar{y}$$

$$\mathtt{RSS}_0 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \ \ \mathtt{where} \ \ \hat{y}_i = \bar{y}$$

2. k=1に対して...

${}_2C_1$個のモデル：$\{\{1\}, \{2\}\} = \{M_{11}, M_{12}\}$

$$\mathtt{RSS}_{11} = \|\boldsymbol{y} - \boldsymbol{X}_{M_{11}} \hat{\boldsymbol{\beta}}_{M_{11}}\|_2^2 \qquad \mathtt{RSS}_{12} = \|\boldsymbol{y} - \boldsymbol{X}_{M_{12}} \hat{\boldsymbol{\beta}}_{M_{12}}\|_2^2$$

小さい方のモデルをM_1と定める

2. k=2に対して...

$$M_2 = \{1, 2\} \qquad \mathrm{RSS}_2 = \|\boldsymbol{y} - \boldsymbol{X}_{M_2}\hat{\boldsymbol{\beta}}_{M_2}\|_2^2$$

#この場合は候補のモデルがひとつなのでRSSを比べないでOK

3. M_0, M_1, M_2から下記の規準が最小になるモデルを選択

クロスバリデーション (CV) #これは今まで通り

Mallow's Cp (or AIC) $\left(\mathrm{RSS}_M + 2\hat{\sigma}^2|M|\right)/n$

Bayesian Information criteria (BIC) $\left(\mathrm{RSS}_M + \log n\, \hat{\sigma}^2|M|\right)/n$

$$\hat{\sigma}^2 = \frac{1}{n-p-1}\|\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}\|_2^2$$

#Fullモデルで計算

参考：Cp (AIC)とBIC

- **Cp (AIC) 規準**
  - 将来のYへの**予測力を重視**した選択
    - 学習データにおけるYへの当てはめとは異なる
  - 一般的に少し過分に変数選択する傾向にある

- **BIC規準**
  - 完璧な**変数選択を重視**した選択
  - Cp (AIC) 規準よりも少ない変数を選択する傾向にある

Rでやってみよう(p=2)

- **Creditデータを読み込む**

```
> credit <- read.csv("Credit.csv",header=T,row.names=1)
> head(credit)
```

income : 収入, limit : 借入上限, rating : 会員レベル, cards : カード枚数

age : 年齢, education : 教育年数, gender : 性別, student : 学生かどうか

married : 結婚しているかどうか, ethnicity : 国籍, **balance** : 債務

- **説明変数を2つに制限 (まずは練習のため)**

```
> credit2 <- credit[,c("Income","Rating","Balance")]
> head(credit2)
   Income Rating Balance
1  14.891    283     333
```

- **M1を求める (Step2)**

$\texttt{Balance} \approx \beta_0 + \beta_1 \texttt{Income}$ **vs.** $\texttt{Balance} \approx \beta_0 + \beta_1 \texttt{Rating}$

```
> y <- credit2[,"Balance"]
> x1 <- cbind(rep(1,length(y)),credit2[,"Income"])
> beta1 <- solve(t(x1)%*%x1)%*%t(x1)%*%y
> rss11 <- sum((y-x1%*%beta1)^2)
```

#Incomeだけのモデルで RSSを計算

同様にratingだけのモデルでRSSを計算して比較

```
> which.min(c(rss11,rss12))
[1] 2
```

$\texttt{Balance} \approx \beta_0 + \beta_1 \texttt{Rating}$ **が選出** $M_1 = \{2\} = \{\texttt{Rating}\}$

- **最適なモデルを選出 (Step3)**

候補モデル : $\{\phi, \{2\}, \{1,2\}\} = \{\phi, \{\texttt{Rating}\}, \{\texttt{Income}, \texttt{Rating}\}\}$

$$M_0 = \phi, \quad M_1 = \{2\}, \quad M_2 = \{1,2\}$$

- **残りのRSSを計算**

```
> rss0 <- sum((y-mean(y))^2)  #RSS_M0を計算
> x <- cbind(rep(1,length(y)),credit2[,c("Income","Rating")])
> x <- as.matrix(x)
> beta <- solve(t(x)%*%x)%*%t(x)%*%y
> rssf <- sum((y-x%*%beta)^2)
> sighat <- sum((y - x%*%beta)^2)/(400-2-1)
```

#RSS$_{M_2}$を計算

#ついでに$\hat{\sigma}^2$も計算しておく

- **Mallow's Cpで選出**

$$\left(\mathrm{RSS}_M + 2\hat{\sigma}^2|M|\right)/n$$

候補モデル $M_0 = \phi, \quad M_1 = \{2\}, \quad M_2 = \{1, 2\}$

の中から上記の規準が最小になるものを選出！

```
> cp0 <- (rss0 + 2*sighat*0)/400
> cp1 <- (rss12 + 2*sighat*1)/400
> cp2 <- (rssf + 2*sighat*2)/400
>
> which.min(c(cp0,cp1,cp2))
[1] 3
```

$\mathtt{Balance} \approx \beta_0 + \beta_1 \mathtt{Income} + \beta_2 \mathtt{Rating}$ **がCpで選出された！**

Best subset selectionの問題点

- **説明変数が大きいと計算時間が爆発**
  - 可能な組み合わせは全部で2^p
  - p=2なら2^2=4個だがp=20だと2^{20}=1048576個
  - これら全てに対してRSSを計算するのはまず不可能
- **代替策**
  - Forward selection
    - Nullモデルに一つずつ足していく
  - Backward selection
    - Fullモデルから一つずつ引いていく
  - Forward-backward selection
    - 上記二つの混合

Forward selection

1. NullモデルM_0：定数(切片)項のみのモデル
2. $k = 0,1,\ldots,p-1$ に対して
   (a) M_kにp-k個の変数をそれぞれ加えたモデル集合を考える
   (b) モデル集合の中で**RSS**を最小にするモデルを見つける
   (c) それをM_{k+1}をする
3. $M_0, M_1, \ldots, M_p$から最適なモデルをCV, Cp (AIC), BICで探す

実際のデータでやってみよう

```
> library(makedummies) #パッケージのインストールと読み込み
> credit <- read.csv("Credit.csv",header=T,row.names=1)
> dcredit <- makedummies(credit)
                                #カテゴリ変数をダミー変数に変換
```

Gender	Student	Married
Male	No	Yes
Female	Yes	Yes
Male	No	No
Female	No	No

Gender	Student	Married
1	0	1
0	1	1
1	0	0
0	0	0

- **Step1：Nullモデルにおける RSSの計算**

```
> y <- dcredit[,"Balance"]
> rss0 <- sum((y-mean(y))^2)
```

- **Step2-1：モデルM1を決定**

$\{1\}, \{2\}, \{3\}, \ldots, \{11\}$ **#この中からRSS最小のモデルを選ぶ**

```
> rss <- rep(0,11)
> for(j in 1:11){
+   x1 <- cbind(rep(1,length(y)),dcredit[,j])
+   x1 <- as.matrix(x1)
+   beta1 <- solve(t(x1)%*%x1)%*%t(x1)%*%y
+   rss[j] <- sum((y-x1%*%beta1)^2)
+ }
> which.min(rss)
[1] 3
```

#Ratingが選ばれた

- **Step2-2 : モデルM2の決定**

$$\{3,1\},\{3,2\},\{3,4\},\ldots,\{3,11\}$$

#Ratingにひとつ変数を
加えたモデル集合

```
> m1 <- names(dcredit)[which.min(rss)]  #変数の名前を保存
```

#Nullモデルに変数Ratingを追加

```
> x1 <- cbind(rep(1,length(y)),dcredit[,m1])
> dcredit1 <- dcredit[,-which(names(dcredit) %in% m1)]
```

#変数Ratingを元データから削除

```
> rss <- rep(0,10)
> for(j in 1:10){
+   x2 <- cbind(x1,dcredit1[,j])
+   beta2 <- solve(t(x2)%*%x2)%*%t(x2)%*%y
+   rss[j] <- sum((y-x2%*%beta2)^2)
+ }
> (m2 <- names(dcredit1)[which.min(rss)])
[1] "Income"
```

・この流れを順次続けていく

```
models <- "Rating"    #最初に選ばれた説明変数M1
x1 <- as.matrix(cbind(rep(1,length(y)),dcredit[,m1])) #XM1
for(k in 1:10){
  rss <- rep(0,11-k)                          #すでに選ばれた説明変数を除く
  dcredit0 <- dcredit[,-which(names(dcredit) %in% models)]
  for(j in 1:(11-k)){
    x2 <- as.matrix(cbind(x1,dcredit0[,j]))
    beta2 <- solve(t(x2)%*%x2)%*%t(x2)%*%y
    rss[j] <- sum((y-x2%*%beta2)^2)
  }
  models <- c(models,names(dcredit0)[which.min(rss)])
  x1 <- as.matrix(cbind(rep(1,length(y)),dcredit[,models]))
}
```

```
> models
 [1] "Rating"                    "Income"
 [3] "Student"                   "Limit"
 [5] "Cards"                     "Age"
 [7] "Gender"                    "Ethnicity_Asian"
 [9] "Married"                   "Ethnicity_Caucasian"
[11] "Education"
```

#選ばれた変数の順番に並べている

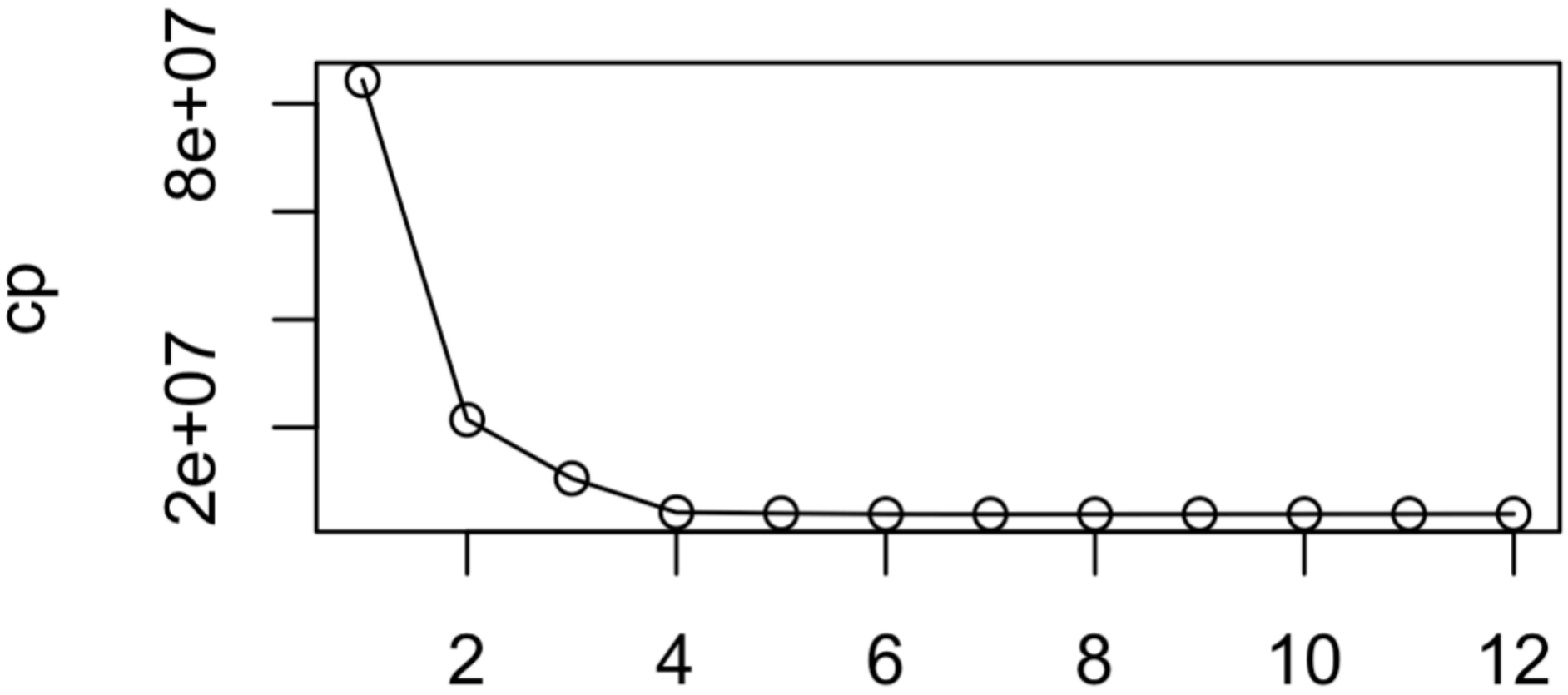
- **Step3 : Cpを最小にするモデルを選択**

```
> xf <- as.matrix(cbind(rep(1,length(y)),dcredit[,1:11]))
> betaf <- solve(t(xf)%*%xf)%*%t(xf)%*%y
> rssf <- sum((y-xf%*%betaf)^2)      #FullモデルのRSS
> sighat <- sum((y - xf%*%betaf)^2)/(400-11-1)
                                     #まずは σ̂ の計算
> cp <- rep(0,12)  #箱を用意
> cp[1] <- rss0                      #M0とM12に対するCpを計算
> cp[12] <- rssf + 2*sighat*11
```

#箱の残り箇所を埋めていく

```
> for(j in 2:11){
+   x <- cbind(rep(1,length(y)),dcredit[,models[1:(j-1)]])
+   x <- as.matrix(x); beta <- solve(t(x)%*%x)%*%t(x)%*%y
+   rss <- sum((y-x%*%beta)^2)
+   cp[j] <- rss + 2*sighat*(j-1)
+ }
```

- **Forward selection + Cpで選択されたモデル**




```
> models[1:which.min(cp)]
[1] "Rating"  "Income"  "Student" "Limit"   "Cards"   "Age"
[7] "Gender"
```

Backward selection

1. FullモデルM_p : 定数(切片)項 + 全ての説明変数を持つモデル
2. $k = p, p-1, \ldots, 1$ に対して
   (a) M_kから(k-1)個の変数をそれぞれ除いたモデル集合を考える
   (b) モデル集合の中で**RSS**を最小にするモデルを見つける
   (c) それをM_{k-1}をする
3. $M_0, M_1, \ldots, M_p$から最適なモデルをCV, Cp (AIC), BICで探す

ForwardとBackward

- **Best subset selectionとの比較**
  - 全組み合わせを探している訳ではない
  - CVやAICをさらに小さくするモデルがあるかも

- **Forward selectionの利点と欠点**
  - 利点：計算が速い, n<pでも実行可能
  - 欠点：説明変数を一度選ぶとそのまま

- **Backward selectionの利点と欠点**
  - 利点：計算が速い
  - 欠点：n<pで実行不可能, 説明変数を一度除外するとそのまま

正則化

中心化

- **中心化によって定数項を除外しておく**
  - 回帰係数にのみ興味がある

#定数項が消えている

$$Y - \mathbb{E}(Y) = \beta_1\{X_1 - \mathbb{E}(X_1)\} + \cdots + \beta_p\{X_p - \mathbb{E}(X_p)\} + \varepsilon$$

- **データではどうやる？**
  - 結果変数と説明変数のそれぞれから平均を取り除く (**中心化**)

```
> credit <- read.csv("Credit.csv",header=T,row.names=1)
> dcredit <- makedummies(credit)
> y <- dcredit[,"Balance"]
> x <- dcredit[,1:11]
> y <- y-mean(y); x <- scale(x,center=T,scale=F)
```

・一応チェックしておく

```
> mean(y)
[1] 1.621314e-14

> apply(x,2,mean)
                  Income                    Limit                   Rating
           -1.532628e-16            -3.636416e-13             1.982676e-15
                   Cards                      Age                Education
           -1.115644e-17             3.128721e-15             7.104083e-16
                  Gender                  Student                  Married
            4.218875e-17            -2.752959e-18            -4.445324e-17
       Ethnicity_Asian Ethnicity_Caucasian
           -4.510281e-18            -2.544697e-17
```

Ridge回帰

- **回帰モデルは次のようになる(中心化ver.)**

$$Y = \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$$

Ridge回帰による回帰係数の推定

$$\underbrace{\sum_{i=1}^{n} \left\{ y_i - (\beta_1 x_{i1} + \cdots \beta_p x_{ip}) \right\}^2}_{\text{\#最小二乗基準}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{\#正則化項}} \longrightarrow \min$$

#$\lambda > 0$: チューニングパラメータ (CVなどで決定)

#$\lambda \approx 0$ ならほぼ最小二乗解; $\lambda \to \infty$ のとき0に縮小推定

Ridge回帰

- **ベクトルで書いてみる**

$$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2 \longrightarrow \min$$

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix} \quad \boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Ridge回帰の解の導出 (*advanced)

$$L := \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2$$

$$\frac{\partial L}{\partial \boldsymbol{\beta}} := \begin{pmatrix} \frac{\partial L}{\partial \beta_0} \\ \frac{\partial L}{\partial \beta_1} \\ \vdots \\ \frac{\partial L}{\partial \beta_p} \end{pmatrix} = -2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + 2\lambda\boldsymbol{\beta}$$

これを0 (ベクトル) とおいてβに関して解くと...

$$\hat{\boldsymbol{\beta}}_{\texttt{ridge}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Ridge回帰の性質

$$\hat{\boldsymbol{\beta}}_{\mathtt{ridge}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- **0への縮小推定**
  - λが大きければ**逆行列の中身が大きい = 分母が大きい**
  - 絶対値の小さな回帰係数は効いていない可能性がある
- **多重共線性の回避**
  - 説明変数の相関が高い(共線性が高い)とXᵀXは非正則行列
    - 最小二乗法だと解が存在しないケースがある
  - それに対してRidge回帰は常に存在
    - λ>0ならばXᵀX+λIは正則行列!

RでRidge回帰

- **Creditデータでやってみる**

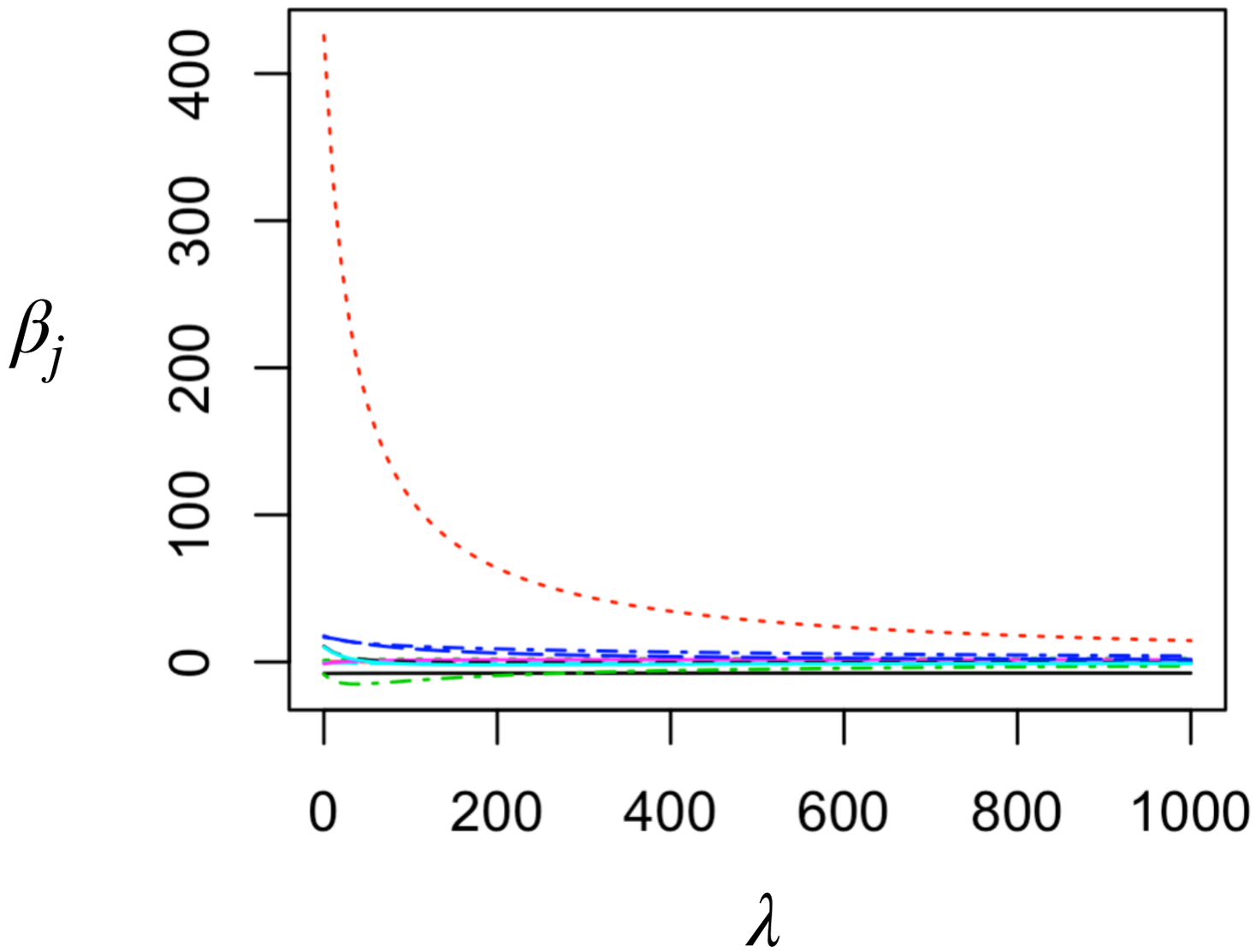
```
> credit <- read.csv("Credit.csv",header=T,row.names=1)
> dcredit <- makedummies(credit)
> y <- dcredit[,"Balance"]
> x <- dcredit[,1:11]
> y <- y-mean(y); x <- scale(x,center=T,scale=F)

> lambda <- 0.5
> b.ridge <- solve(t(x)%*%x + lambda*diag(11))%*%t(x)%*%y
```

#lambdaの値を変えて推定値の変化をみてみよう

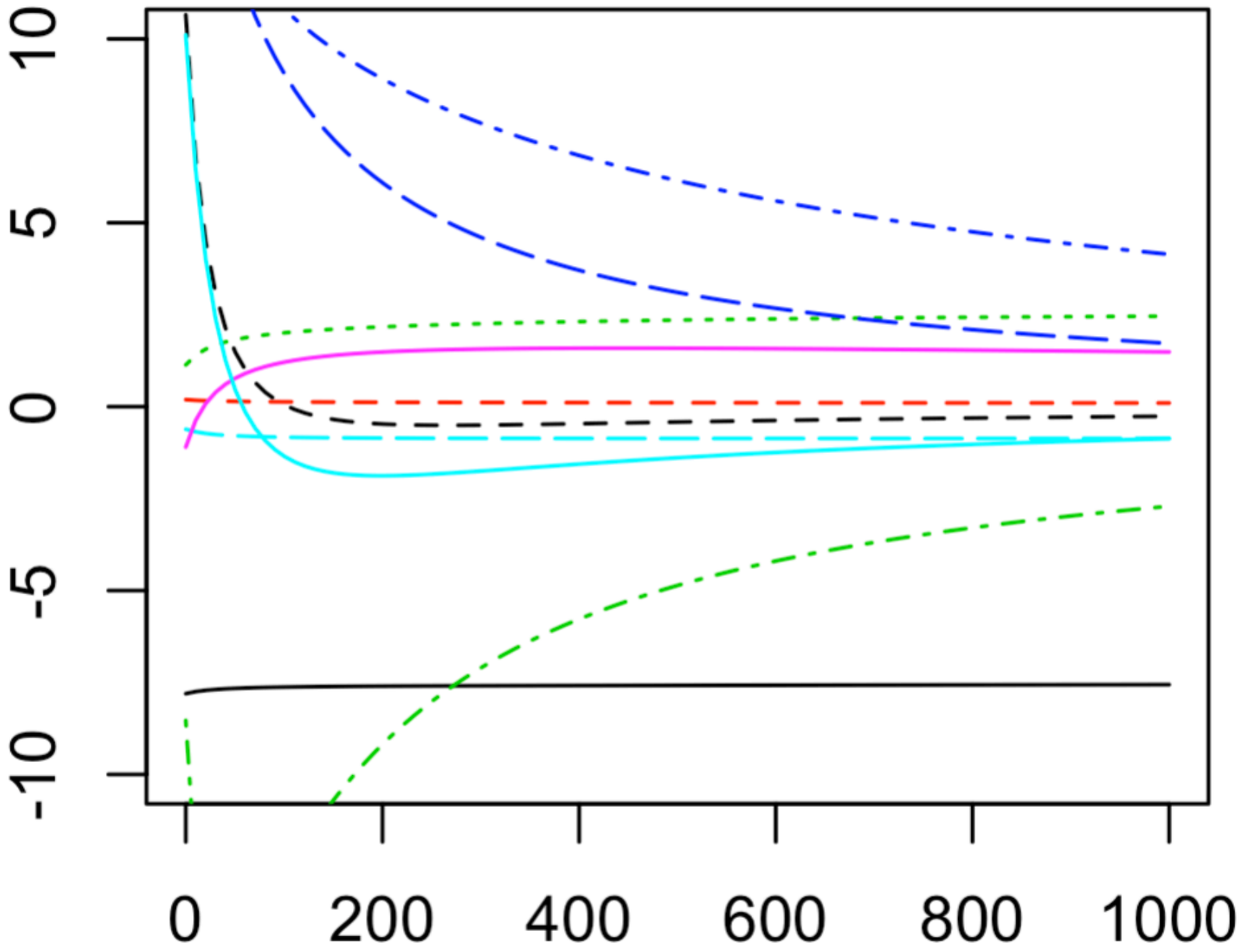
Solution path

- **推定値の挙動の図示**
  - x軸にλ, y軸に各回帰係数の推定値をプロットする



拡大してみる

0に縮小している様子が分かる



おまけ

- **Rのコード**

```
> ridge <- function(lambda){
+   solve(t(x)%*%x + lambda*diag(11))%*%t(x)%*%y
+ }
```

```
> cand <- seq(0,1000,length=100)  #λの候補を用意
> sols <- sapply(cand,function(x)ridge(x))
                                  #λの候補に対する推定値を一気に計算

> matplot(cand,t(sols),type="l")
> matplot(cand,t(sols),type="l",ylim=c(-10,10))
      #行列形式のデータを一気にプロットする関数”matplot”
```

BiasとVariance

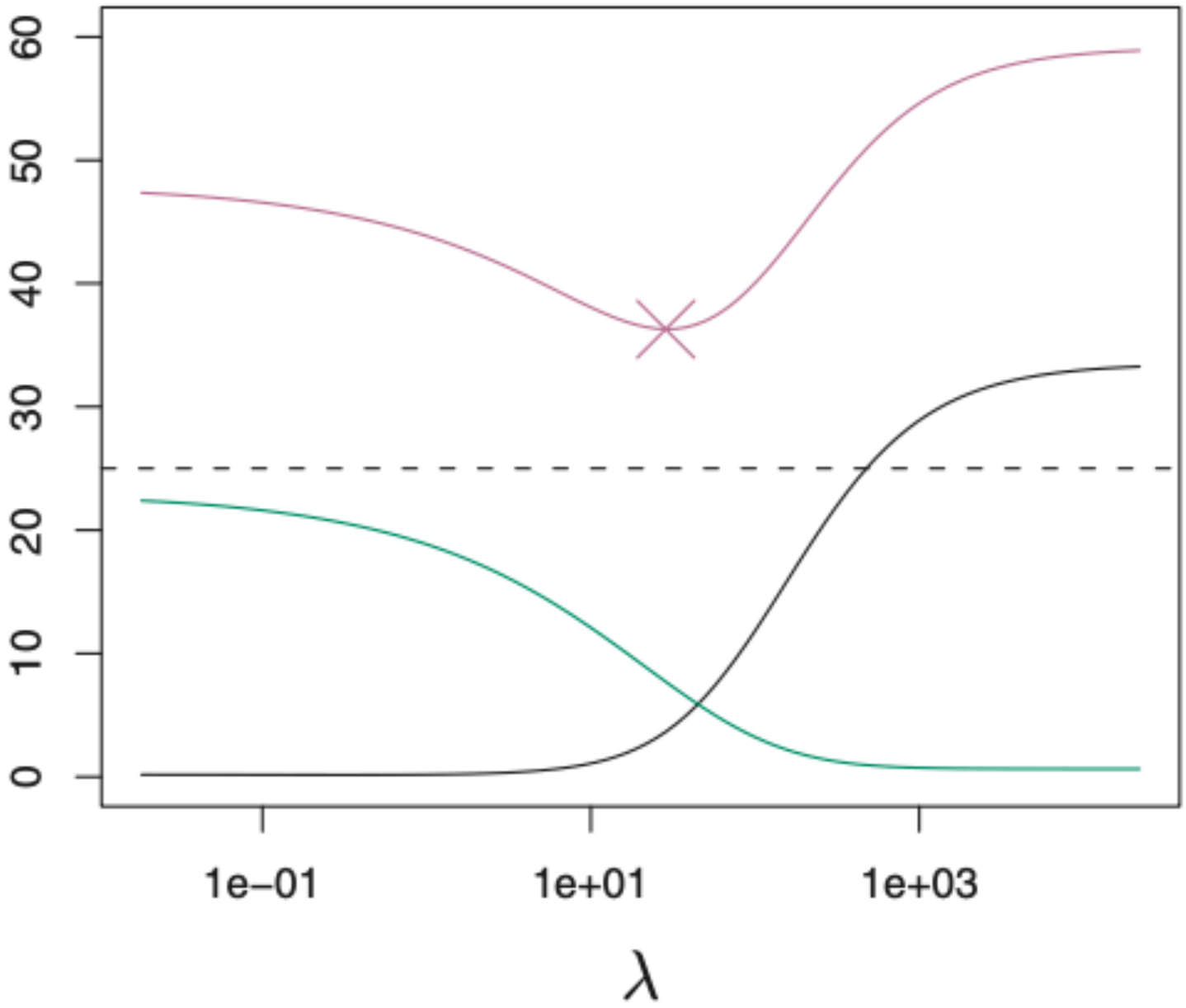
- **Ridge回帰のチューニングパラメータλ**
  - Ridge回帰はバイアスのある推定量
    - λが小さいとバイアスは小さくなる
    - しかし一方で分散が大きくなってしまう
  - λが大きな場合
    - 分散が小さくなる
    - しかし一方でバイアスが大きくなる

トレードオフの関係性

$$\mathbb{E}(\hat{\boldsymbol{\beta}}_{\texttt{ridge}}) = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} \neq \boldsymbol{\beta}$$

$$\mathbb{V}(\hat{\boldsymbol{\beta}}_{\texttt{ridge}}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}$$

紫：テスト誤差　緑：バイアス　黒：分散



Ridge正則化からLasso正則化へ

- **Ridge回帰の問題点**
  - 回帰係数が正確に0と推定される訳ではない
    - 小さい値が必ず残っていしまう…
  - その意味で集合Mを特定できていない
  - これを回避して正確に0と推定(スパース推定)できるのがLasso

Lasso

Lasso回帰による回帰係数の推定

$$\sum_{i=1}^{n} \left\{ y_i - (\beta_1 x_{i1} + \cdots \beta_p x_{ip}) \right\}^2 + \lambda \sum_{j=1}^{p} |\beta_j| \longrightarrow \min$$

#最小二乗基準 **#正則化項**

- **Ridge回帰との違い**
  - 正則化項の部分が絶対値和になっている
  - 正則化項が原点で微分不可能
  - **この微分不可能性がスパース性を促している**

なぜスパースに？

- **LassoとRidgeの別表現**

Lasso回帰

$$\min \sum_{i=1}^{n} \left\{ y_i - (\beta_1 x_{i1} + \cdots \beta_p x_{ip}) \right\}^2 \quad \text{s.t.} \quad \sum_{j=1}^{p} |\beta_j| \leq s_1$$

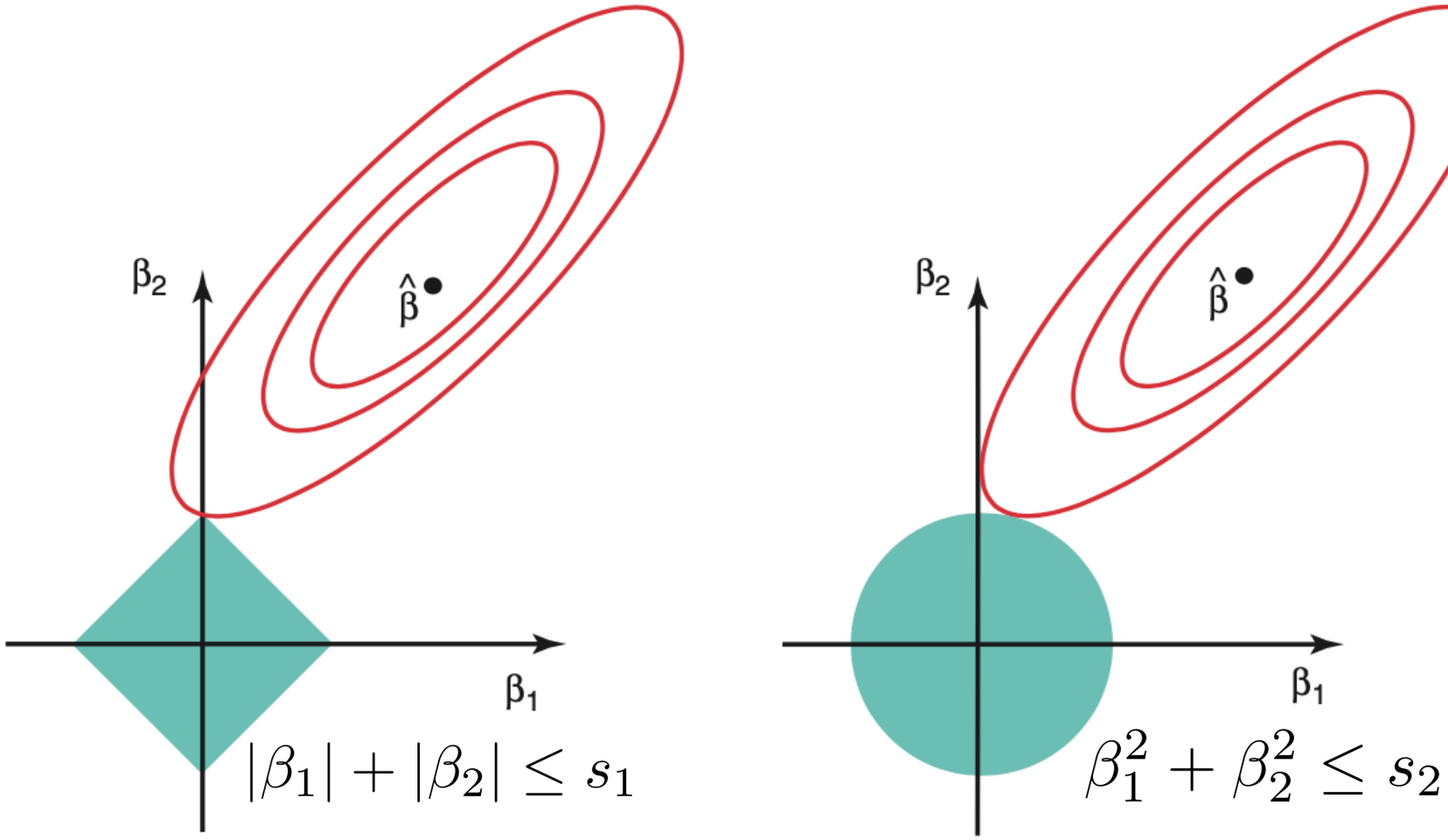
Ridge回帰

$$\min \sum_{i=1}^{n} \left\{ y_i - (\beta_1 x_{i1} + \cdots \beta_p x_{ip}) \right\}^2 \quad \text{s.t.} \quad \sum_{j=1}^{p} \beta_j^2 \leq s_2$$

Advanced : KKT条件から，解がそれぞれ同じになる定数s_1, s_2が存在する

p=2のときのイメージ

赤線：最小二乗基準の等高線 $\sum_{i=1}^{n}\left\{y_i-(\beta_1 x_{i1}+\beta_2 x_{i2})\right\}^2=t$



緑がそれぞれ制約空間

なぜスパースに？解析解ver.

- **簡単な場合で実際に解いてみる**
  - 基本的に下記のケースしか陽に解けない(解析解が得られない)

$$L_\lambda(\beta) = \sum_{i=1}^{n}(y_i - \beta)^2 + \lambda|\beta|$$

$\lambda = 0$の場合

$L_\lambda(\beta)$の場合を最小にする $\hat{\beta}$ は平均値 $\bar{y}$

正則化項がRidgeの場合

最小にする $\hat{\beta}$ は $\sum_{i=1}^{n} y_i/(n+\lambda)$

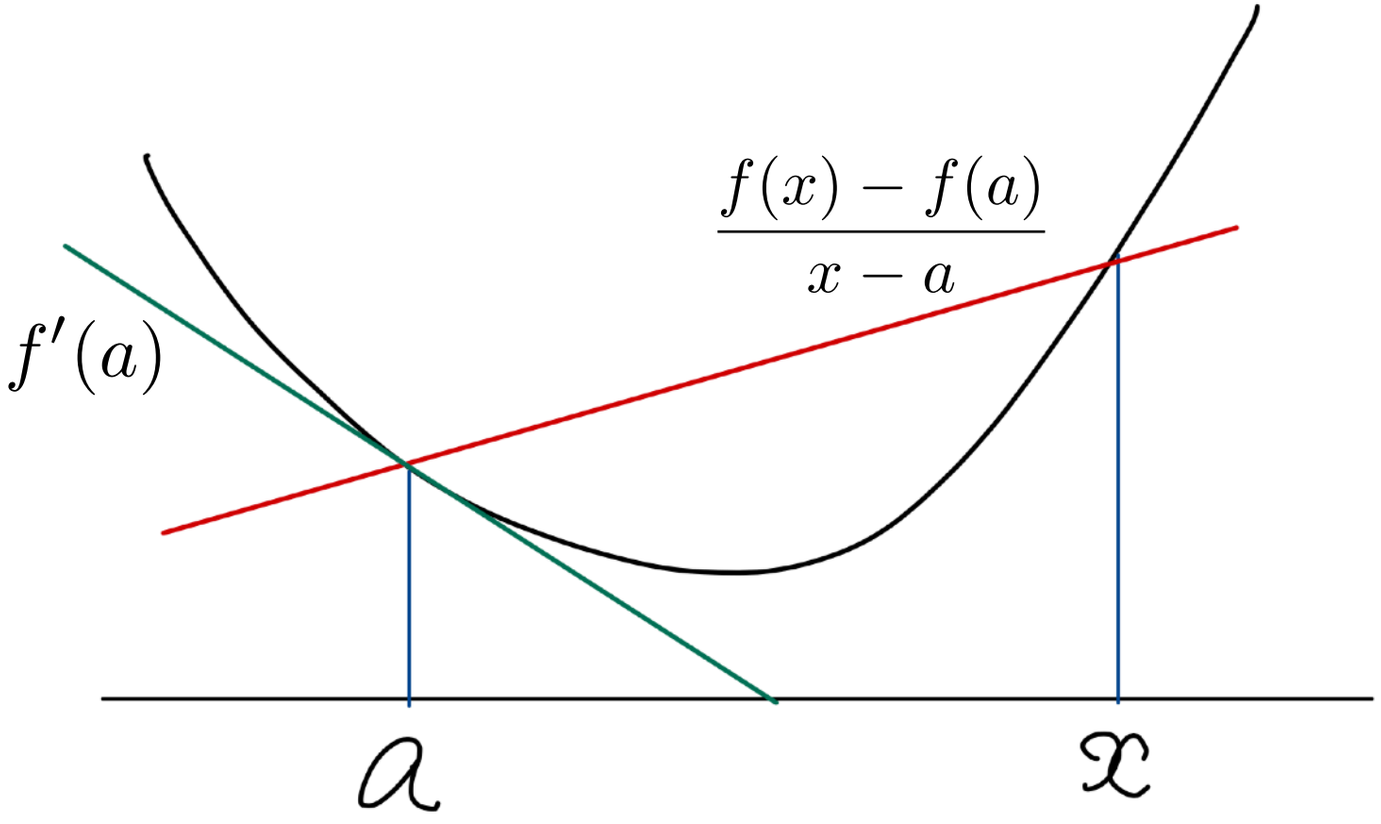
#平均値よりもλの分だけ0に縮小されている

凸関数の性質 (定義)

- **関数fが凸関数かつ微分可能の場合**

$$f(x) \geq f(a) + f'(a)(x - a), \quad \text{for any } x \in \mathbb{R}$$

この性質は関数fが点aで
微分可能な場合のみ成立



- **微分不可能な場合はどうするか？**
  - 劣勾配(sub-gradient)を導入する

- **定義**
  - 関数fは凸関数
  - 次を満たす実数cを点aにおけるfの**劣勾配**という

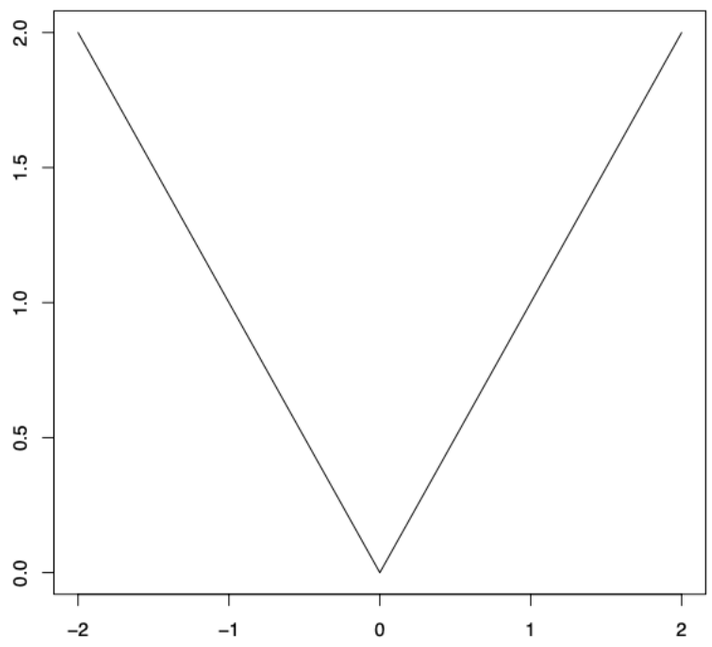
$$f(x) \geq f(a) + c(x - a), \quad \text{for any } x \in \mathbb{R}$$

  - 点aにおける劣勾配全体を$\partial f(a)$と書く
  - **性質**：$\partial f(a)$が1点集合 $\Leftrightarrow f$ は点aで微分可能

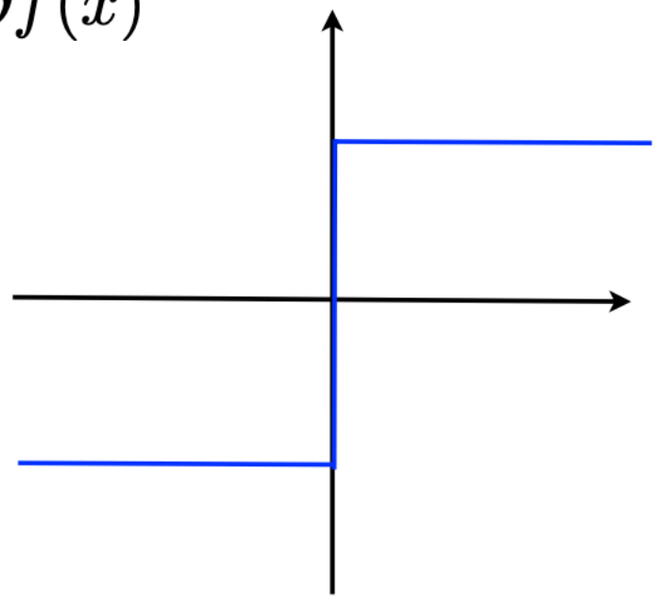
劣勾配の例

- **Example 1**
  - $f(x) = |x|$
  - $a > 0 \Rightarrow \partial f(a) = \{1\}, a < 0 \Rightarrow \partial f(a) = \{-1\}$
  - $a = 0 \Rightarrow \partial f(a) = [-1,1]$

$f(x)$

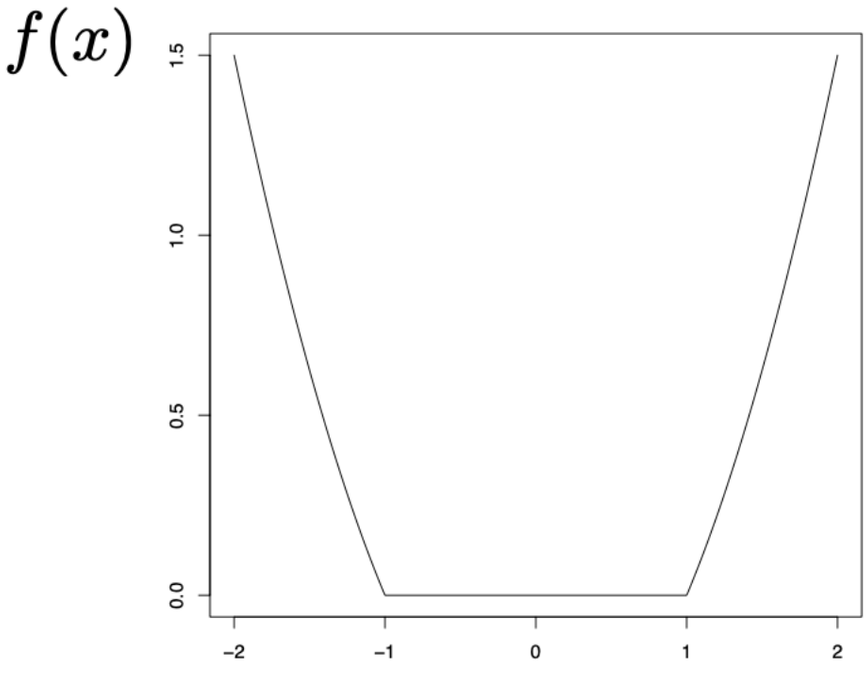


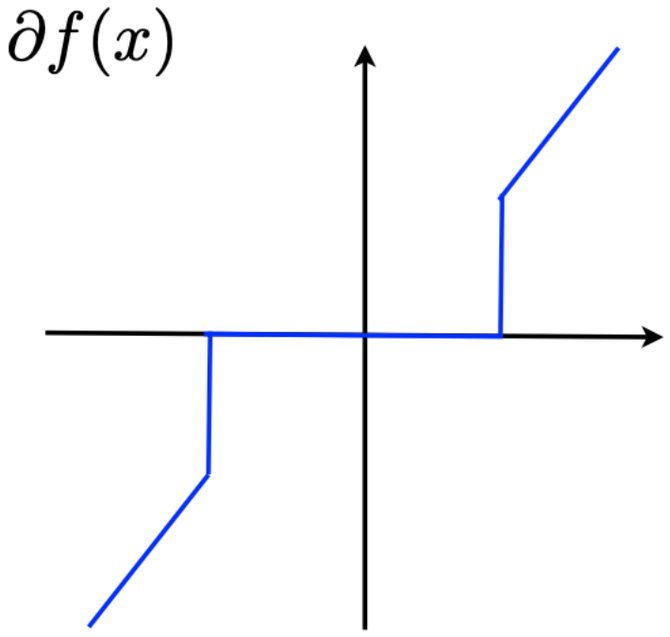
$\partial f(x)$



劣勾配の例

- **Example 2**
  - $f(x) = \max(0,(x^2 - 1)/2)$
  - $\partial f(-1) = [-1,0], \partial f(1) = [0,1]$





最小値と劣微分

- **性質**
  - $\hat{x}$ が凸関数 f の最小値 $\Leftrightarrow 0 \in \partial f(\hat{x})$
  - 凸関数の最小値を見つけるためには劣微分をみればOK
- **前述の例に適用してみる**
  - Example 1 : 最小値 $\hat{x} = 0$
  - Example 2 : 最小値 $\hat{x} \in [-1,1]$

元々の問題に戻る

$$L_\lambda(\beta) = \sum_{i=1}^{n}(y_i - \beta)^2 + \lambda|\beta|$$

の解 $\hat{\beta}$ を求めよう

$$\partial L_\lambda(\beta) = -2\sum_{i=1}^{n}(y_i - \beta) + \lambda\partial|\beta|$$

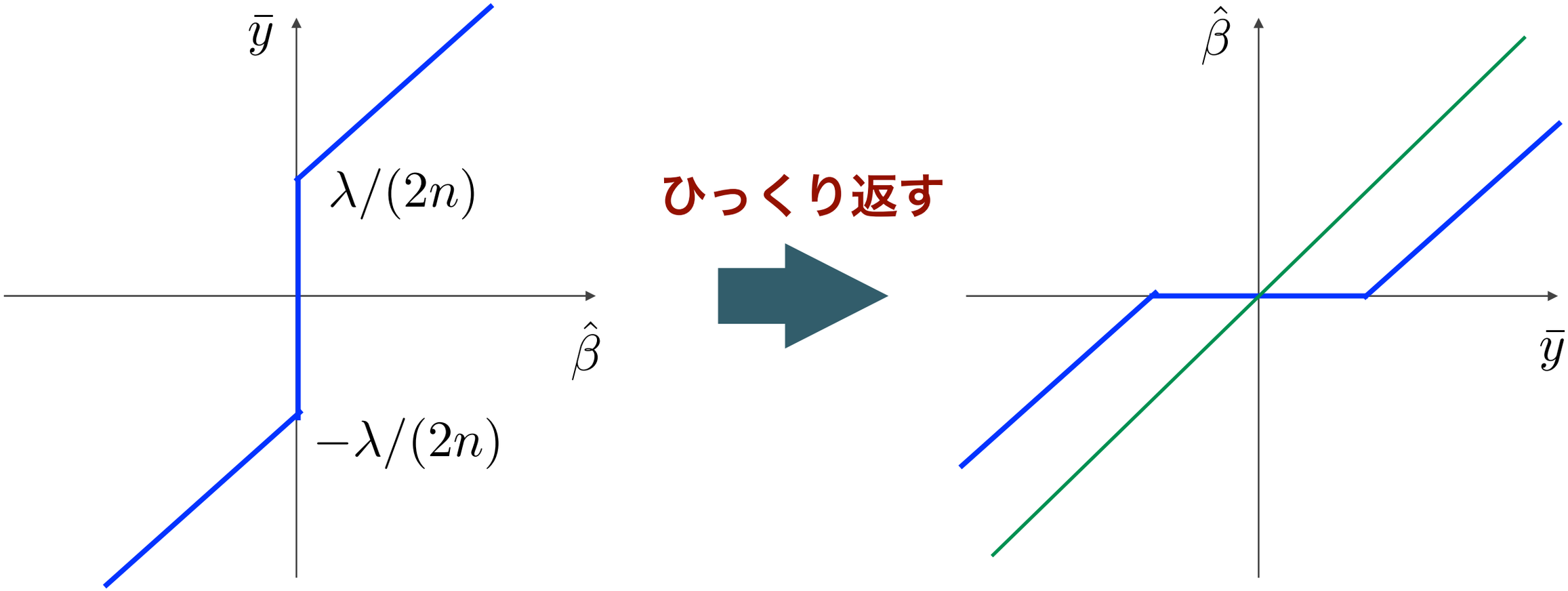
$$\partial|\beta| = \{z \in \mathbb{R} : z = \text{sgn}(\beta) \text{ if } \beta \neq 0,\ z \in [-1, 1] \text{ if } \beta = 0\}$$

$0 \in \partial L_\lambda(\hat{\beta})$ $\Leftrightarrow$ there exists $z \in \partial|\hat{\beta}|$ such that

$$-2\sum_{i=1}^{n}(y_i - \hat{\beta}) + \lambda z = 0$$

- 少し変形する

$$\bar{y} = \hat{\beta} + \frac{\lambda}{2n} z$$



$$\hat{\beta} = \mathrm{sgn}(\bar{y}) \max(0, |\bar{y}| - \lambda/(2n))$$

How to solve Lasso

$$\min_{\beta_1,\ldots,\beta_p} \frac{1}{2}\sum_{i=1}^{n}\left\{y_i-(\beta_1 x_{i1}+\cdots\beta_p x_{ip})\right\}^2+\lambda\sum_{j=1}^{p}|\beta_j|$$

- **座標降下法 (Coordinate descent method)**
  - 変数βをひとつずつ更新していくアルゴリズム
  - 変数ひとつだけの最適化 → 解析的に解ける！

- **近接勾配法 (Proximal gradient descent method)**
  - 二乗誤差の部分を二次近似する方法

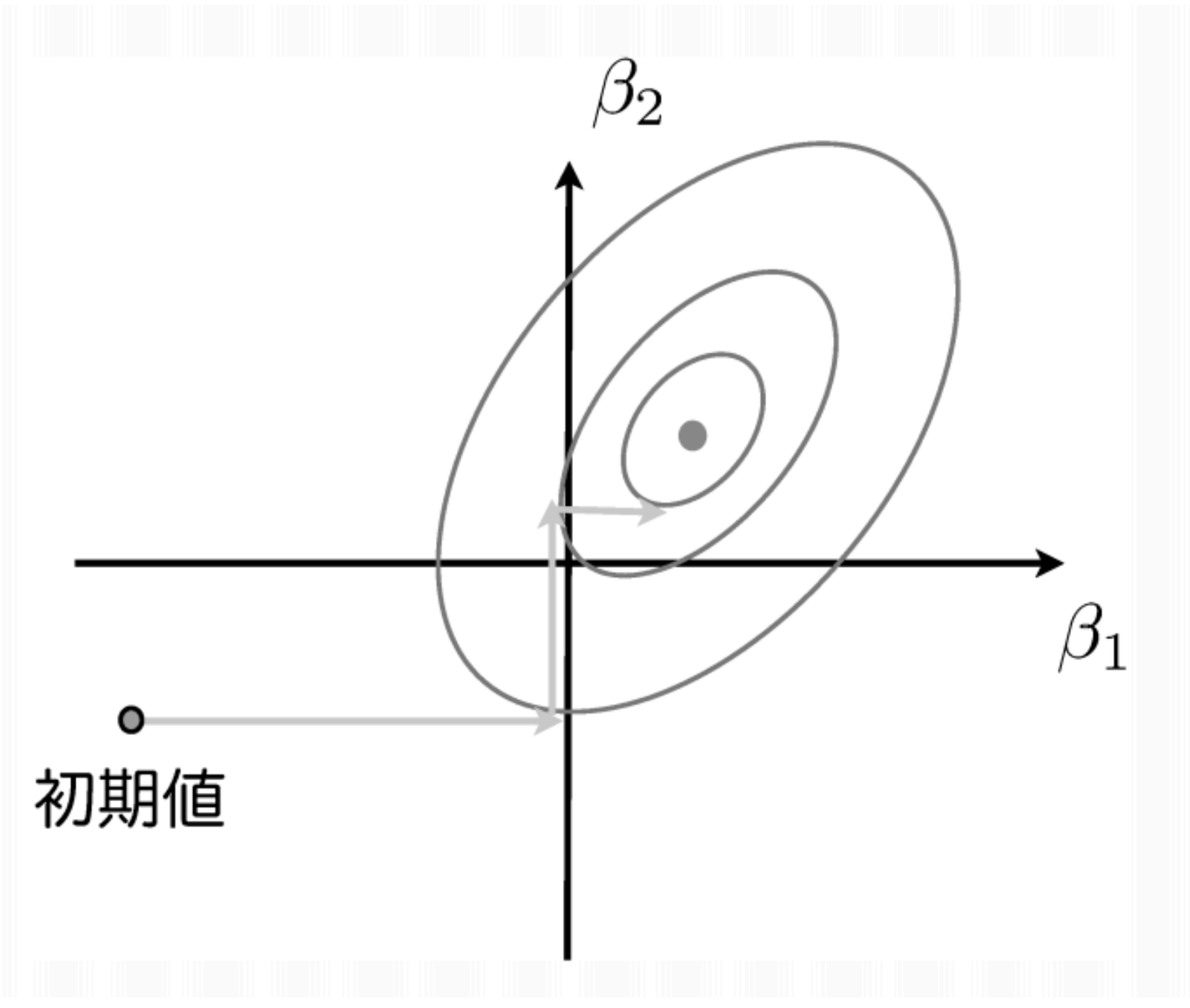
座標降下法

$$L_\lambda(\beta_1, \beta_2, \ldots, \beta_p) := \frac{1}{2}\sum_{i=1}^{n}\left\{y_i - (\beta_1 x_{i1} + \cdots \beta_p x_{ip})\right\}^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

- **第kステップでの解：** $\beta_1^k, \beta_2^k, \ldots, \beta_p^k$
  - UPDATE as

$$\begin{aligned}
\beta_1^{k+1} &\leftarrow \underset{\beta_1 \in \mathbb{R}}{\operatorname{argmin}}\, L_\lambda(\beta_1, \beta_2^k, \ldots, \beta_p^k) + \lambda|\beta_1| \\
\beta_2^{k+1} &\leftarrow \underset{\beta_2 \in \mathbb{R}}{\operatorname{argmin}}\, L_\lambda(\beta_1^{k+1}, \beta_2, \beta_3^k \ldots, \beta_p^k) + \lambda|\beta_2| \\
&\vdots \\
\beta_p^{k+1} &\leftarrow \underset{\beta_p \in \mathbb{R}}{\operatorname{argmin}}\, L_\lambda(\beta_1^{k+1}, \ldots, \beta_{p-1}^{k+1}, \beta_p) + \lambda|\beta_p|
\end{aligned}$$

イメージ



廣瀬 (2014). 数理解析研究所講究録, 第1908巻

・各ステップにおける解

$$\beta_1^{k+1} = \frac{\mathrm{sgn}(Z_1)\max(0, |Z_1| - \lambda)}{\sum_{i=1}^n x_{i1}^2}, \quad Z_1 = \sum_{i=1}^n x_{i1}\left(y_i - \sum_{j=2}^p x_{ij}\beta_j^k\right)$$

$$\beta_2^{k+1} = \frac{\mathrm{sgn}(Z_2)\max(0, |Z_2| - \lambda)}{\sum_{i=1}^n x_{i2}^2}, \quad Z_2 = \sum_{i=1}^n x_{i2}\left(y_i - x_{i1}\beta_1^{k+1} - \sum_{j=3}^p x_{ij}\beta_j^k\right)$$

$$\vdots$$

$$\beta_p^{k+1} = \frac{\mathrm{sgn}(Z_p)\max(0, |Z_p| - \lambda)}{\sum_{i=1}^n x_{ip}^2}, \quad Z_p = \sum_{i=1}^n x_{ip}\left(y_i - \sum_{j=1}^{p-1} x_{ij}\beta_j^{k+1}\right)$$

計算のテクニック

- **Zの計算を少し変形**

$$\sum_{i=1}^{n} x_{i1}\left(y_i - \sum_{j=2}^{p} x_{ij}\beta_j^k\right) = \sum_{i=1}^{n} x_{i1}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j^k + x_{i1}\beta_1^k\right)$$

$$= \sum_{i=1}^{n} x_{i1}y_i - \sum_{j=1}^{p}\left(\sum_{i=1}^{n} x_{i1}x_{ij}\right)\beta_j^k + \sum_{i=1}^{n} x_{i1}^2\beta_1^k$$

#次の計算を事前にしておくと便利：$\boldsymbol{X}^T\boldsymbol{X}, \boldsymbol{X}^T\boldsymbol{y}$

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{2p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}, \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{pmatrix}$$

書き換え

$$Z_1 = (\boldsymbol{X}^T\boldsymbol{y})_1 - (\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}^k)_1 + (\boldsymbol{X}^T\boldsymbol{X})_{11}\beta_1^k$$

#$()_1$はベクトルの第1要素を表し，$()_{11}$は行列の(i,i)成分を表す

β_1^k を更新してそれを同じ記号で書くと...

$$Z_2 = (\boldsymbol{X}^T\boldsymbol{y})_2 - (\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}^k)_2 + (\boldsymbol{X}^T\boldsymbol{X})_{22}\beta_2^k$$

実装してみよう

```
cda <- function(y,x,lambda){
  n <- dim(x)[1]; p <- dim(x)[2]
  covx <- t(x)%*%x; covxy <- t(x)%*%y ##t(X)X and t(X)y
  beta <- rep(0,p) ##initial value of beta
  judge <- 100 ##for convergence
  z <- rep(0,p); old <- rep(0,p)
  repeat{
    if(judge >= 0.001){
      for(j in 1:p){


      }
      judge <- sum(abs(beta-old)) ##compare old and new
    }else break
  }
  return(beta)
}
```

実行結果

```
> ##toy data
> x <- matrix(rnorm(100*20),100,20)
> beta <- c(c(1,1,1),rep(0,17))
> y <- x %*% beta + rnorm(100)
> round(cda(y,x,15),4)
 [1] 0.7545 0.9981 0.8604 0.0000 0.0000 0.0000 0.0000 0.0000
 [9] 0.0000 0.0000 0.0000 0.1141 0.0000 0.0000 0.0000 0.0000
[17] 0.0000 0.0000 0.0000 0.0000
```

λを変えて実行してみよう

Solution path

- **横軸にλの値, 縦軸に回帰係数をプロットしたもの**
  - λの変化に伴う回帰係数の挙動がわかる