金融工学
アプリケーションにおける
数理計画法の実際

田辺隆人
tanabe@msi.co.jp
nuopt-info@msi.co.jp
（株）数理システム

ポートフォリオ

$$R(x) \equiv \sum_{j \in Asset} r_j x_j \equiv r^t x$$

$$\sum_{j \in Asset} x_j = 1, \quad x_j \geq 0$$

収益率の期待値とリスク

$$E\left[R(x)\right] \equiv \sum_{j \in Asset} \overline{r}_j x_j \equiv \overline{r}^t x$$

$$V\left[R(x)\right] \equiv \sum_{i,\, j \in Asset} Q_{ij} x_i x_j \equiv x^t Q x$$

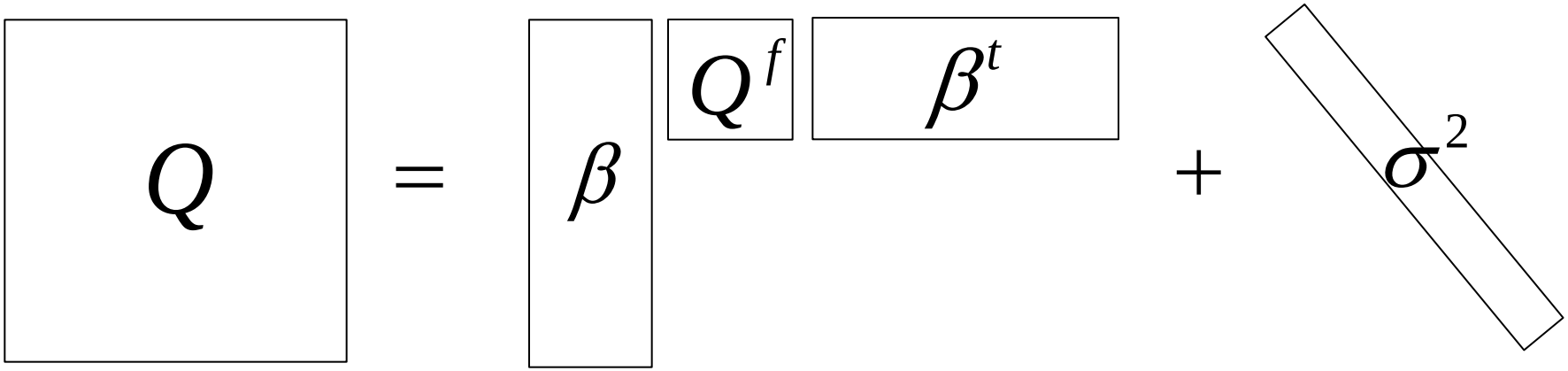
ファクターモデル（収益率）

$$r_j = \alpha_j + \sum_k \beta_{jk} f_k + \varepsilon_j$$

$$\begin{bmatrix} r \end{bmatrix} = \begin{bmatrix} \alpha \end{bmatrix} + \begin{bmatrix} \beta \end{bmatrix} \begin{bmatrix} f \end{bmatrix} + \begin{bmatrix} \varepsilon \end{bmatrix}$$

ファクターモデル
（分散共分散）

$$Q_{ij} = \sum_{k,l \in Factor} Q^{f}_{k,l} \beta_{j,k} \beta_{i,l} + \delta_{ij} \cdot \sigma_i^2$$



中間変数の導入

n × n

$$x^t Q x = x^t \beta Q^f \beta^t x + x^t diag(\sigma^2) x$$

$$= s^t Q^f s + x^t diag(\sigma^2) x,$$

$$s = \beta^t x$$

m × m

m × n

中間変数
なし.vs.あり

- 計算結果

	内点法		*有効制約法*	
	計算時間	メモリ所要	計算時間	メモリ所要
なし	201秒	550M	74秒	550M
あり	1秒	30M	0.75秒	30M

(使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes)

中間変数
なし.vs.あり

- サンプル点 k における収益率

$$R^k(x) \equiv \sum_{j \in Asset} r_j^k x_j$$

- 全サンプル点の収益率

$$\left[r_j^k \right] x \equiv R \cdot x$$

中間変数
なし.vs.あり

n × n

$$\sum_{i,j \in Asset} Q_{ij} x_i x_j \equiv x^t Q x$$

$$= \frac{1}{|K|} x^t (R - \bar{R})^t (R - \bar{R}) x = \frac{1}{|K|} s^t s,$$

$$s = (R - \bar{R}) x$$

|K| × n

中間変数
なし.vs.あり

- |K| = 60の場合

	内点法		*有効制約法*	
	n=1000	n=2000	n=1000	n=2000
なし	40秒	219秒	1秒	6秒
あり	1秒	4秒	0.3秒	0.5秒

（使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes）

中間変数
なし.vs.あり

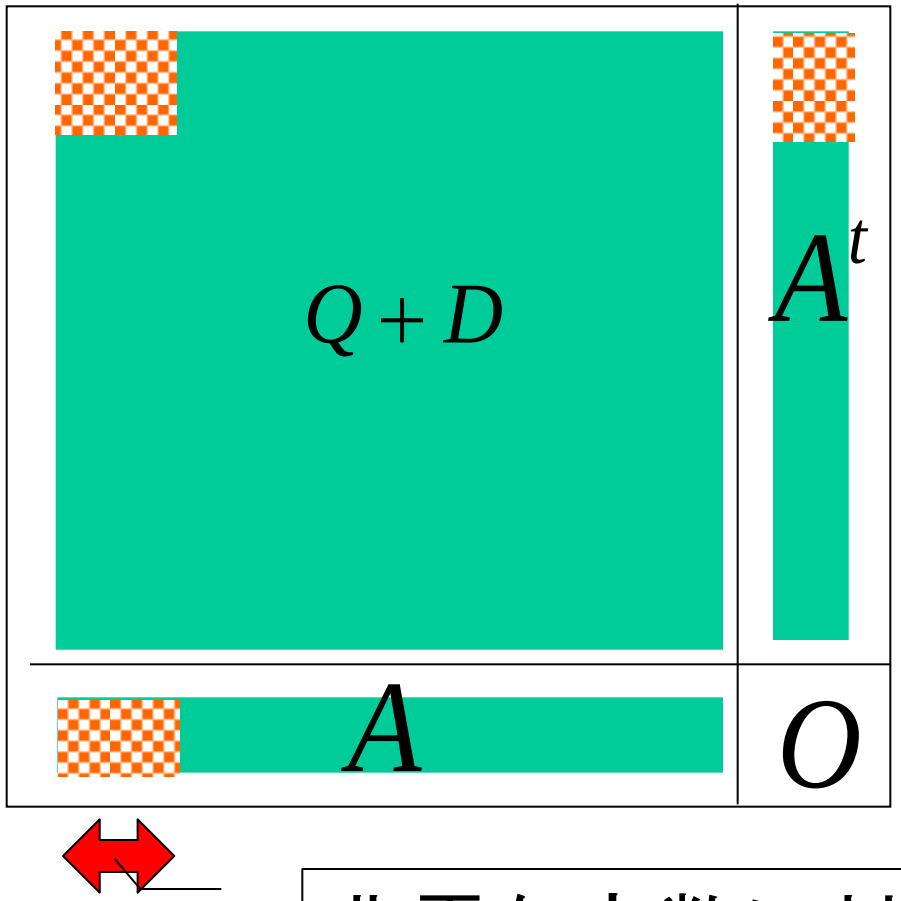
- その他にも

あえて変数を消去しない方が有利な
ケースあり

$$x \cdot z = \mu$$

$$x + z = 1$$

内点法.vs.有効制約法



非零な変数に対応する列

内点法.vs.有効制約法

- インデックス運用の場合の目的関数

$$V\left[R(x)-R(x^B)\right]=$$

$$\sum_{i,j\in Asset}\sum_{k,l\in Factor} Q^f_{k,l}\beta_{j,k}\beta_{j,l}(x_i-x^B_i)(x_j-x^B_j)$$

$$+\sum_{j\in Asset}\sigma_j^2(x_j-x^B_j)^2$$

x_Bの非零が多いと
有効制約法は不利

内点法.vs.有効制約法

X_B の組み入れ銘柄数1329

	組み入れ銘柄数	*内点法*	*有効制約法*
リスク最小化指向	782	1.0秒	2.6秒
中庸	404	1.0秒	1.2秒
収益率最大化指向	179	1.0秒	0.8秒

（使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes）

内点法.vs.有効制約法

- 有効制約法
  - ○基底が得られる
  - ○リスタートの効率が良い
  - ○組み入れ銘柄が少ない
- 内点法
  - ○縮退に関する情報が得られる

リスク尺度の一般化

- サンプル点 k における収益率

$$R^k(x) \equiv \sum_{j \in Asset} r_j^k x_j$$

リスク尺度の一般化

リスク尺度	意味
分散	$E\left[\left(R^k(x)-E\left[R^k(x)\right]\right)^2\right]$
絶対偏差	$E\left[\left\|R^k(x)-E\left[R^k(x)\right]\right\|\right]$
下方リスク	$R^k(x)$ で負のものの絶対値の和
一次の下方積分積率	$R^k(x)$ が所与の定数値を下回るものの和の合計
CVaR最小化	$-R^k(x)$ の最大から　p（所与）個の平均値
Maximum drawdown	$R^k(x)$ の部分列の最大と最小の差

CVaR最小化

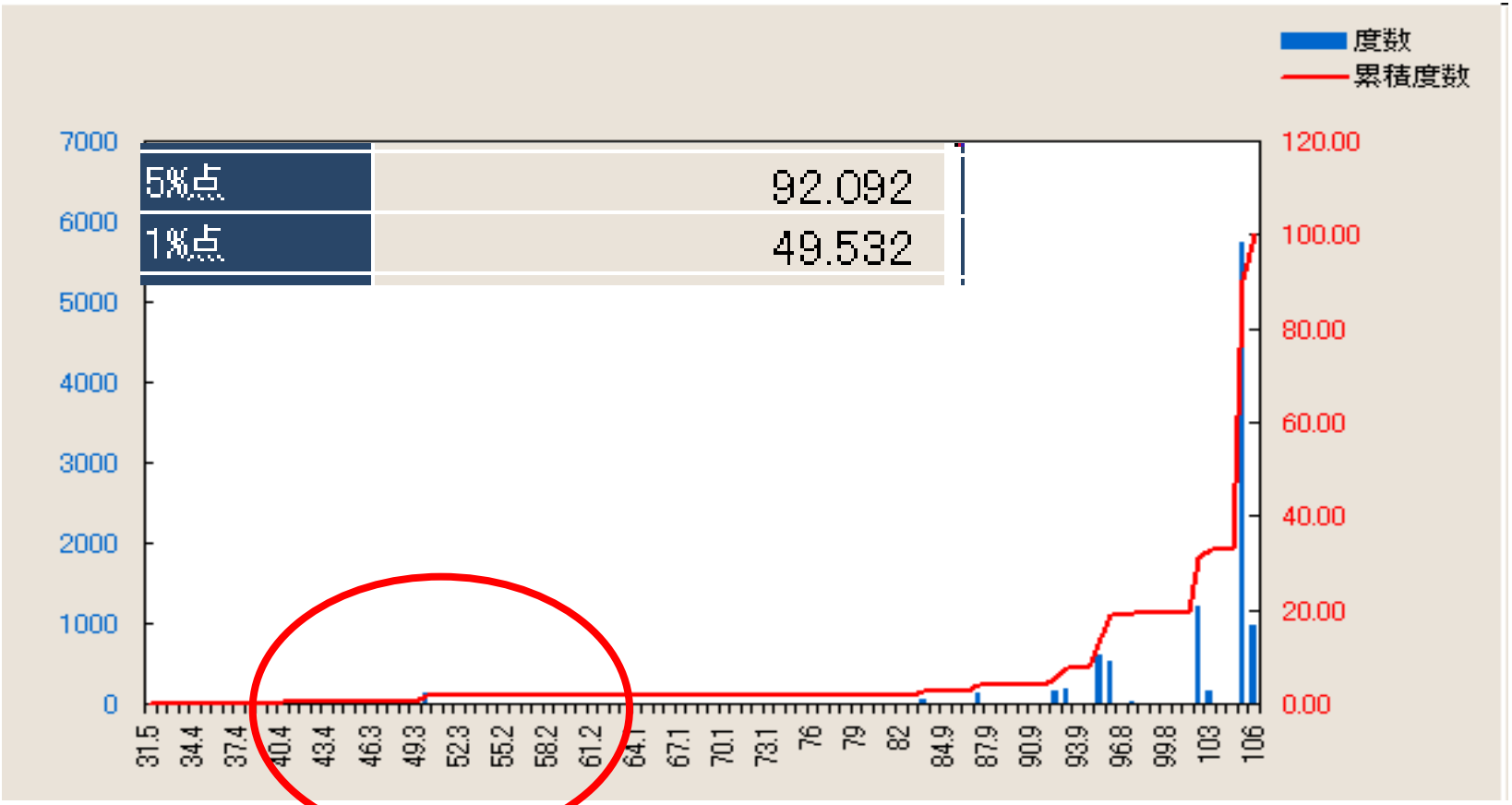
最小化 $\frac{1}{p}\sum_{k\in Sample} s_k + \alpha$

制約 $s_k + \alpha \geq -R^k(x)$

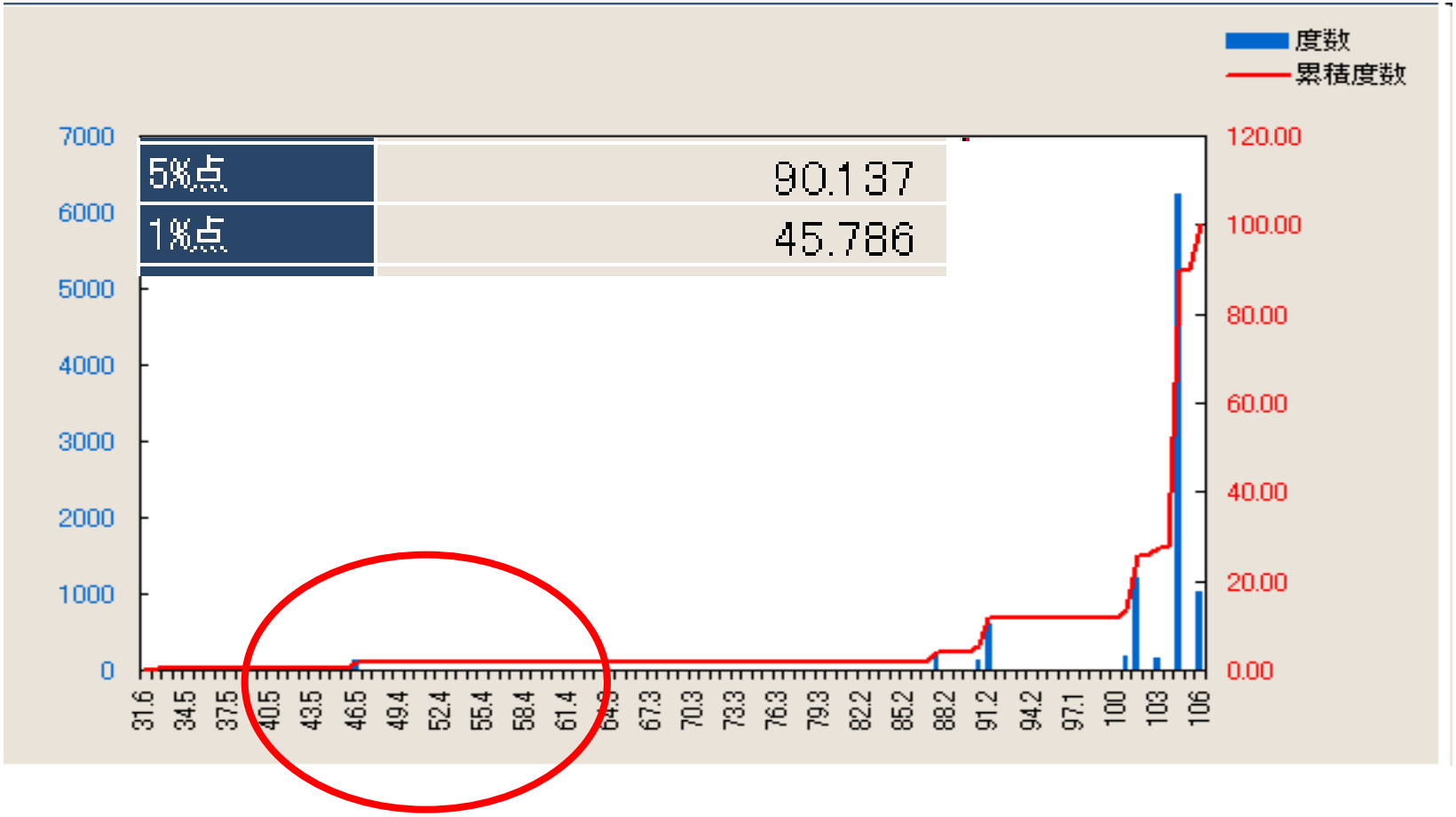
$s_k \geq 0, \quad x \in \Omega$

投資可能集合

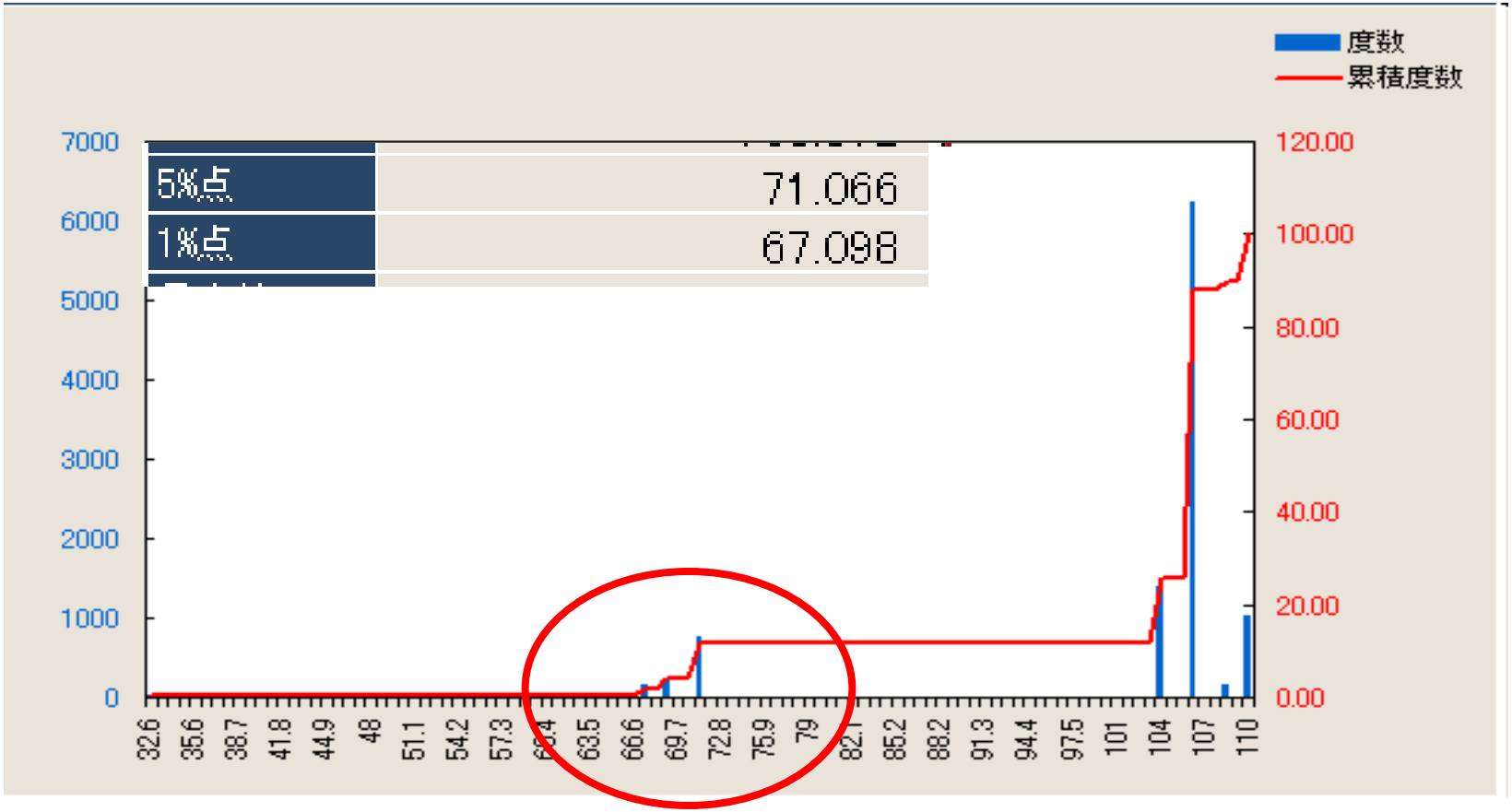
分散.vs.CVaR
分散最小化による社債ポートフォリオ
期待収益率＝1.281（最小）



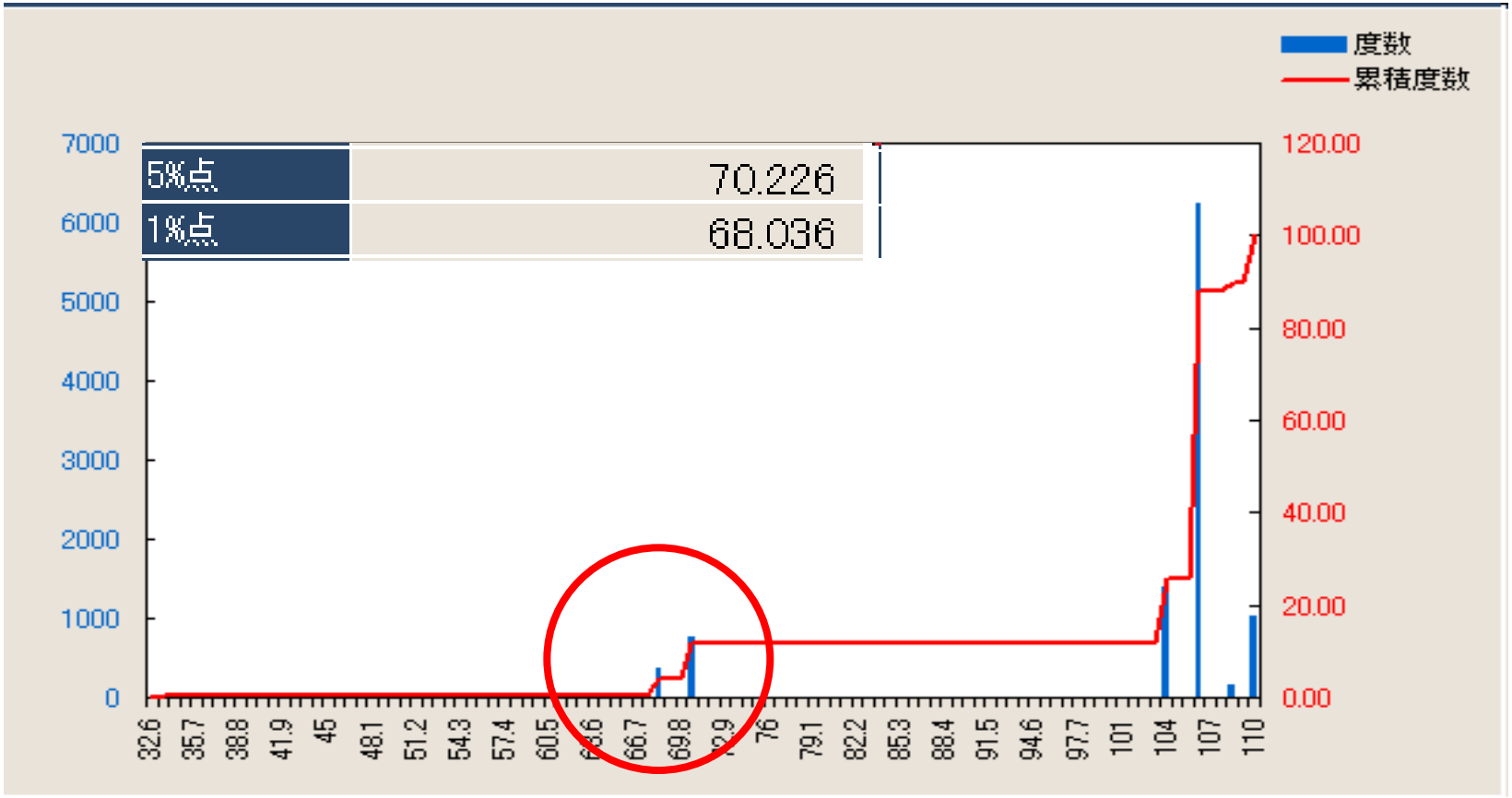
分散.vs.CVaR
分散最小化による社債ポートフォリオ
期待収益率＝1.4



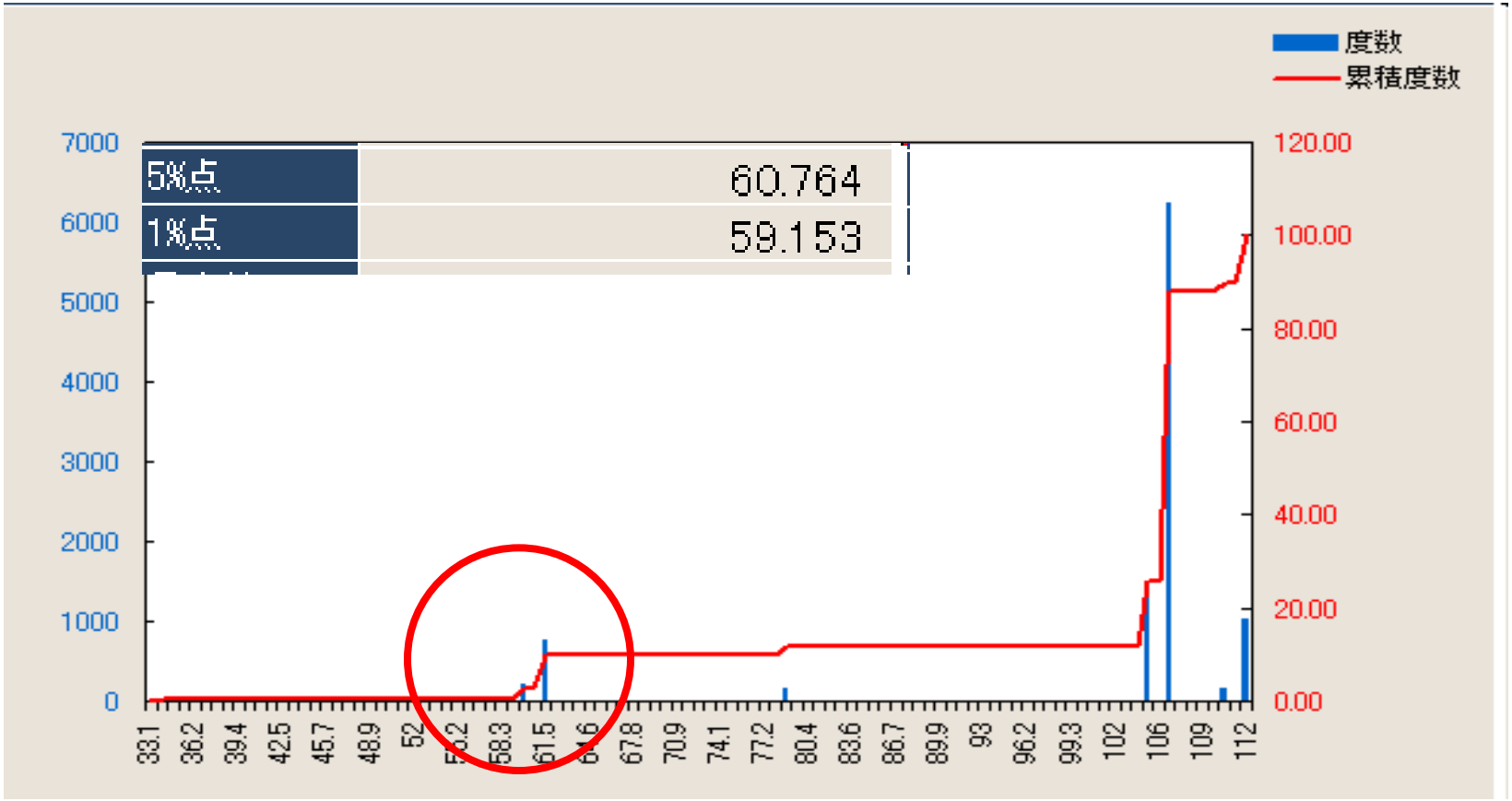
分散.vs.CVaR
分散最小化による社債ポートフォリオ
期待収益率＝1.5



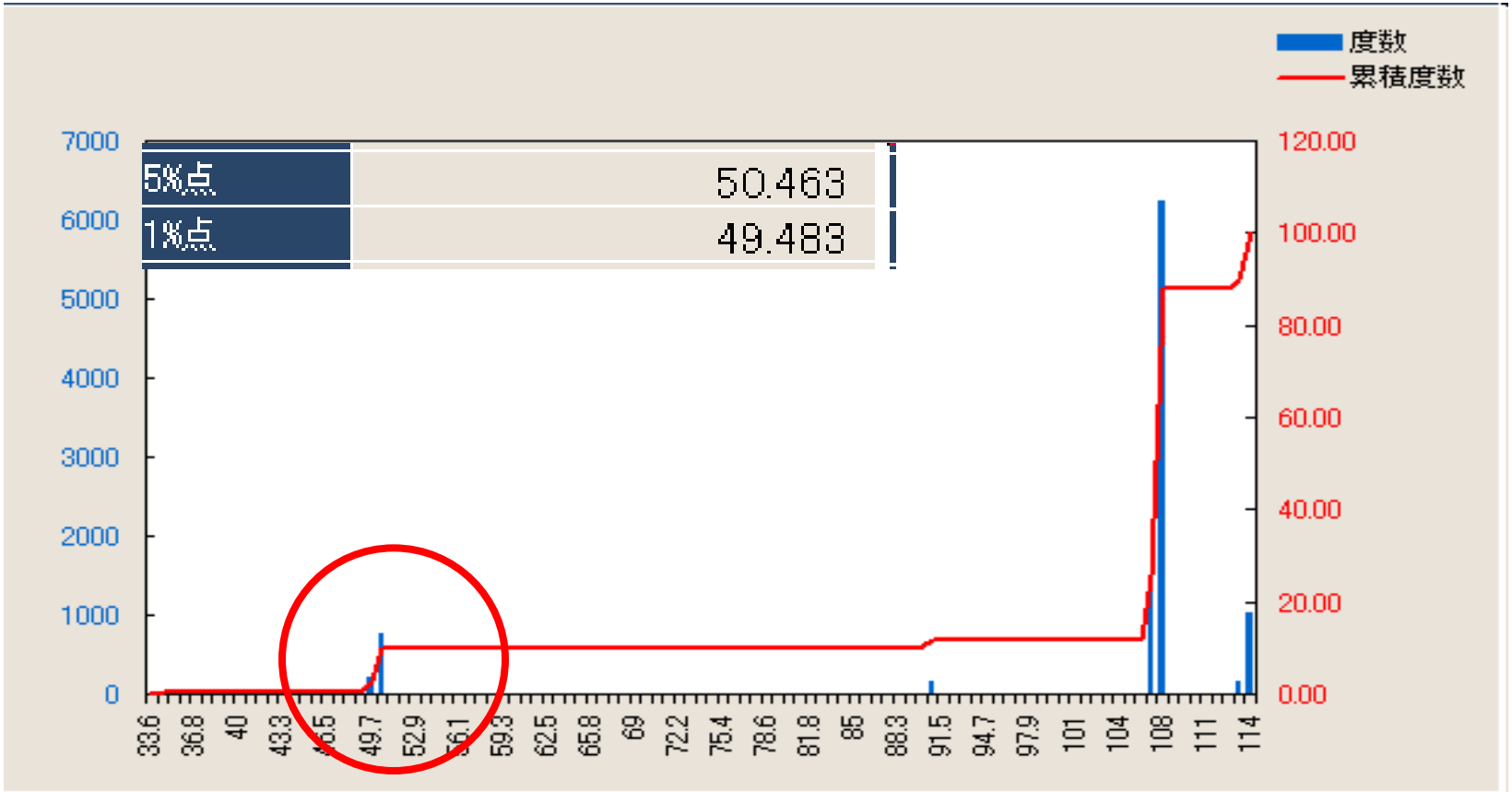
分散.vs.CVaR
CVaR最小化による社債ポートフォリオ
期待収益率＝1.504（最小）



一般の分布
分散最小化による社債ポートフォリオ
期待収益率＝1.55



分散.vs.CVaR
分散最小化による社債ポートフォリオ
期待収益率＝1.6



分散.vs.CVaR

- 分散
  〇簡便
  ×fat tail なケースへの対処
- CVaR
  ×大量データ必要
  〇LPとなる
  〇fat tail なケースのコントロール

内点法.vs.単体法
CVaR最小化

サンプル数	*内点法*	*単体法*
5000	0.6秒	3.3秒
10000	2.1秒	24.3秒
20000	4.7秒	107.5秒
50000	12.9秒	―
100000	42.3秒	―

(使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes)

内点法.vs.単体法

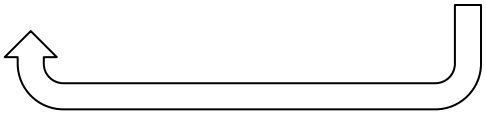
CVaR最小化の双対問題

サンプル数	*内点法*	*単体法*
5000	0.4秒	0.2秒
10000	1.5秒	0.7秒
20000	4.0秒	2.6秒
50000	18.6秒	15.0秒
100000	41.7秒	67.9秒

(使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes)

内点法.vs.単体法
CVaR最小化でフロンティア曲線

- 内点法＋クロスオーバー
  ⇒基底解⇒リスタート



16点の描画，サンプル数10000

クロスオーバー	0.1秒
リスタート時間平均	23.7秒
リスタート時間最小	12.8秒
リスタート時間最大	24.9秒
総合計	397.8秒

(使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes)

ロング・ショートモデル

- 組み入れ比率の定義

$$x_j \equiv x_j^{long} - x_j^{short} = x_j^0 + d_j^{buy} - d_j^{sell}$$

$$x_j^{long}, x_j^{short}, d_j^{buy}, d_j^{sell} \geq 0$$

整数変数不要な要件

- ターンオーバー上限制約

$$\sum_j d_j^{buy} + d_j^{sell} \leq T$$

- 取引コスト最小化

$$\text{最小化} \sum_j \alpha \cdot d_j^{buy} + \beta \cdot d_j^{sell}$$

整数変数が必要となる要件

- ポジションの等式制約

$$\sum_j x_j^{long} = LT$$

- 一般の投資制約

$$\sum_j A_j^i x_j^{long} \leq b^i \qquad \sum_j C_j^i d_j^{buy} \leq d^i$$

整数変数の導入

- ロング・ショート

$$x_j^{long} \le U_j^{long} \delta_j^{long},\ x_j^{short} \le U_j^{short} (1-\delta_j^{long})$$

- 購入・売却

$$d_j^{buy} \le U_j^{buy} \delta_j^{buy},\ d_j^{sell} \le U_j^{sell} (1-\delta_j^{buy})$$

分枝限定法の性能

- 収益率最大化＞リスク最小化
  ⇒　組み入れ銘柄数
- 相補性制約＞＞銘柄数制約
  ⇒　効率的な補強の有無

ロングショートモデルの補強

- 制約追加

$$d_j^{buy} \le x_j^{long} \le x_j^0 + d_j^{buy}$$

$$x_j^{short} \le d_j^{sell} \qquad , j \in Long$$

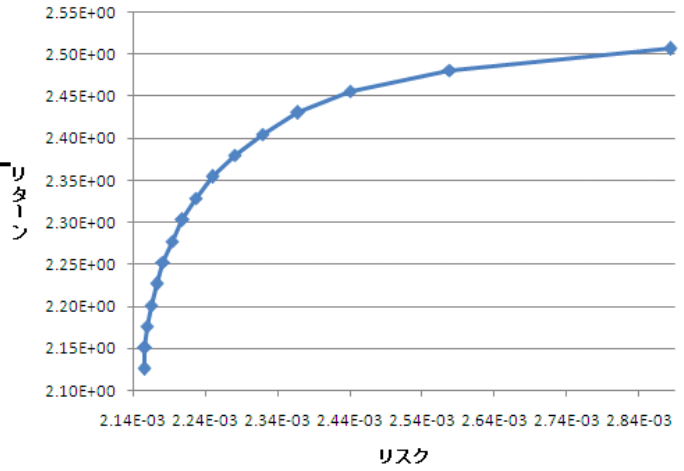
- 表現の変更

$$(x_j)^2 \Rightarrow (x_j^{long})^2 + (x_j^{short})^2$$

ロングショートモデルの性能

- １６点のフロンティア曲線描画

計算時間平均	23.7秒
計算時間最小	11.2秒
計算時間最大	34.0秒
総時間合計	392.3秒



（使用環境：NUOPT9.1　Core2 1.6GHz メモリ2GBytes）

非線形計画法の用途

- フィッティング
  - 誤差最小化
- 判別分析
  - 誤判別最小化

一階微係数のみ.vs.二階微係数
主双対内点法の理論的成果

- メリット関数(バリアペナルティ関数)

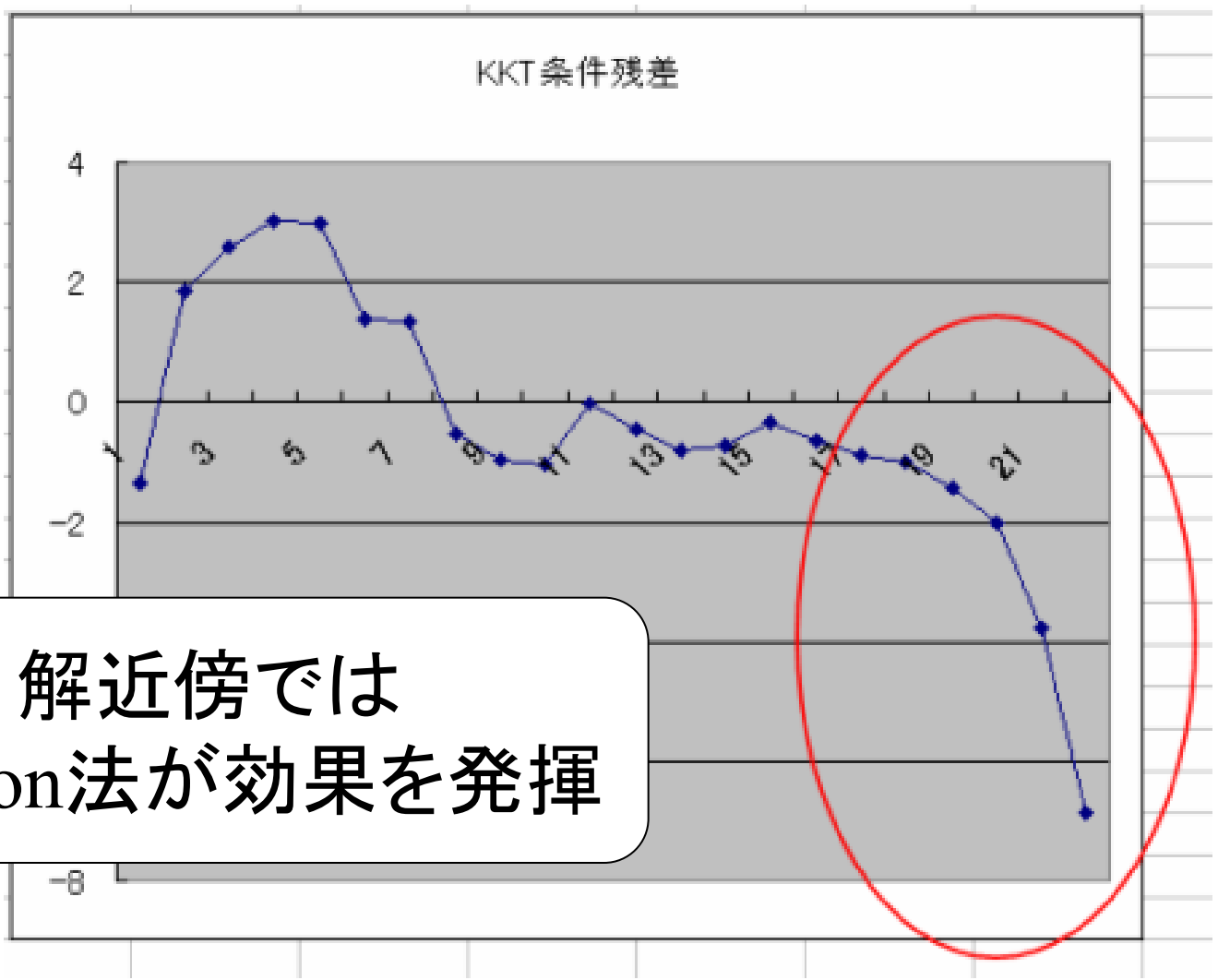
$$F(x,\mu) = f(x) - \mu\sum \log(x_i) + \rho\sum |g_i(x)|$$

  のもとでの大域的収束

- 適当な μ の更新方法のもとでの
  超一次収束

H. Yamashita, H. Yabe, and T. Tanabe,
A globally and superlinearly convergent primal-dual interior point trust region method for large scale constrained optimization, Mathematical Programming Ser.B,January 2005, Vol.102, Number 1, pp.111-151 (2005)

超一次収束（n=20000）



解近傍では
Newton法が効果を発揮

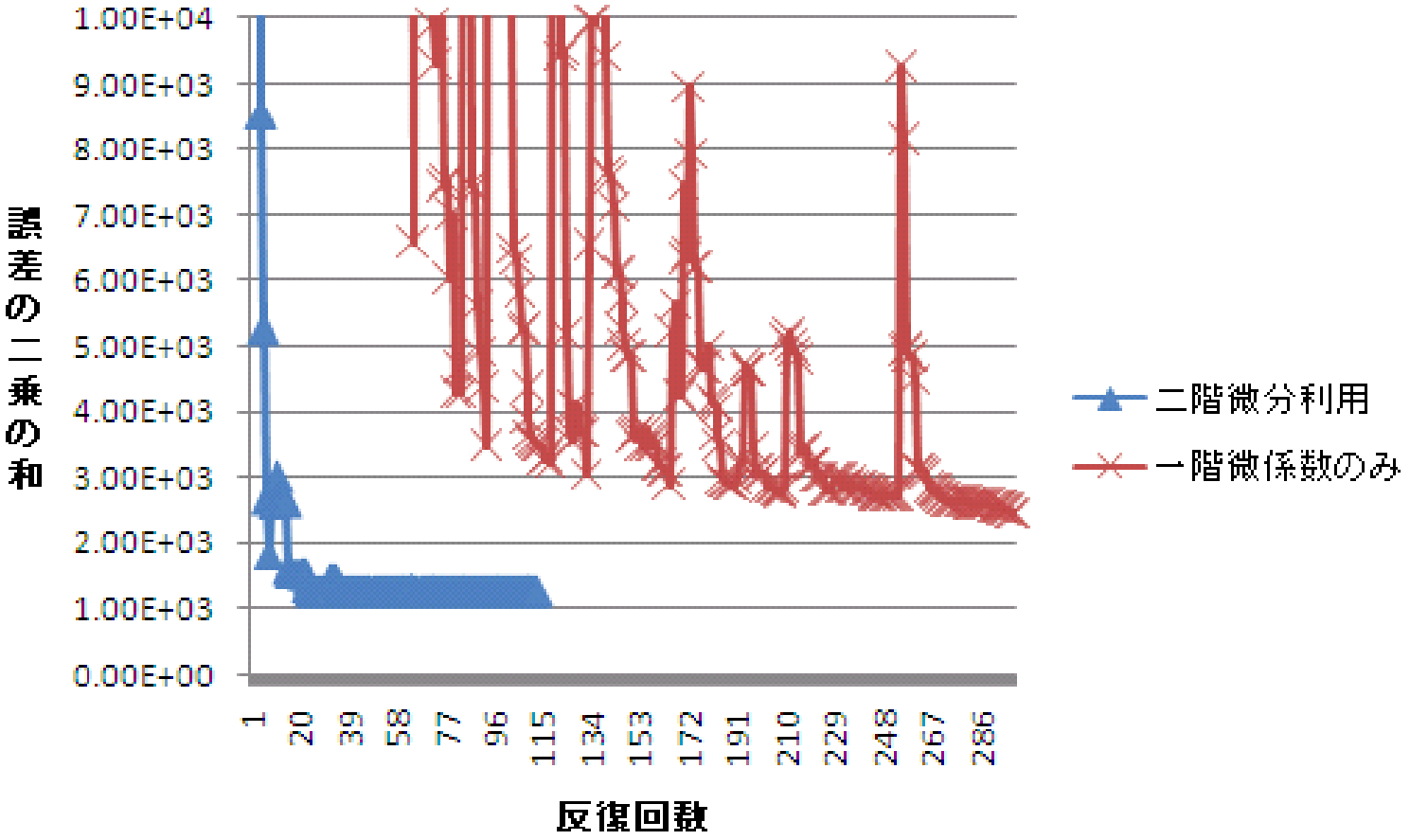
一階微係数のみ.vs.二階微係数
実務的なフィッティング問題

方法	*反復回数*	*停留条件残差*	*誤差二乗和*	*計算時間*
二階微係数	112回	5.0e-7	1257.6	67.8秒
一階微係数のみ	300回	1.8e-2	2452.8	105.5秒

（使用環境：NUOPT9.1　Core2 1.6GHz　メモリ2GBytes）

一階微係数のみ.vs.二階微係数

実務的なフィッティング問題



一階微係数のみ.vs.二階微係数

格付け推移確率行列推定問題

変数: X $\quad X \in \mathbf{R}^{n \times n}$

最小化: $\left\| X^{12} - A_{year} \right\|$ $\quad A_{year} \in \mathbf{R}^{n \times n}$

制約: $\sum_{j \in S} X_{ij} = 1$ $\quad i \in S$

$X_{ij} \geq 0$ $\quad i, j \in S$

二階微係数
計算コストに
見合う効果

方法	*反復回数*	*微係数計算時間*	*誤差二乗和*
二階微係数	9回	925秒	1秒
一階微係数のみ	128回	3676秒	3秒

(使用計算機：PentiumM1.4GHz,メモリ1.5GBytes)

一階微係数のみ.vs.二階微係数
格付け推移確率行列推定問題

- 繰り返し構造を利用した
  自動微分法の工夫

微係数計算時間
925秒 ⇒ 100秒

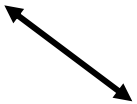
（使用計算機：PentiumM1.4GHz,メモリ1.5GBytes）

田辺隆人,
数理計画法に適した自動微分算法の開発と実装
中央大学博士論文

微係数が連続.vs.不連続
絶対値関数の扱い

- イディオム

最小化 $|f|$

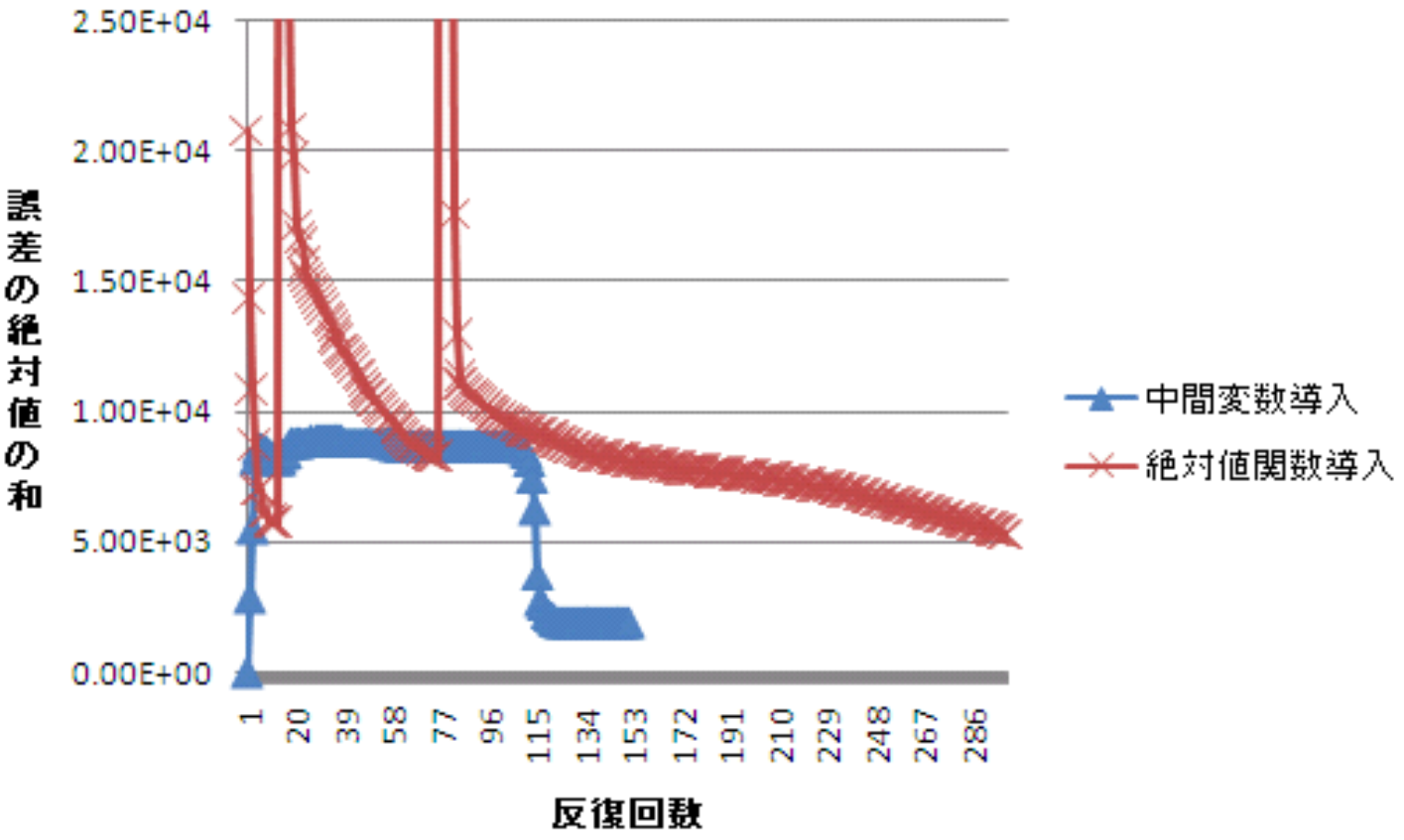


最小化　s

制約　$-s \leq f \leq s$

$s \geq 0$

微係数が連続.vs.不連続
絶対値関数の扱い



判別分析

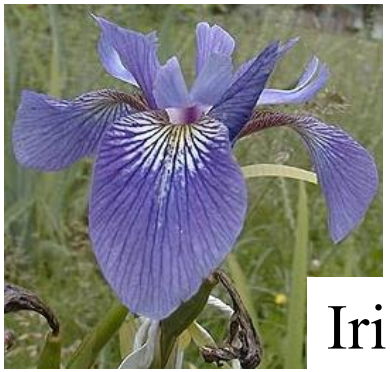
- 計測結果から性質を予想する




判別分析

- 計測結果から性質を予想する

Iris virsinica



Iris versicolor

Iris virsinica

Iris versicolor

数理計画モデル

- 変数

$$a_0, a_1, a_2, a_3, a_4 \quad (\text{パラメータ})$$

- 目的関数

*versicolor*なのに$\dfrac{e^{\eta}}{1+e^{\eta}}$が1/2以上になる

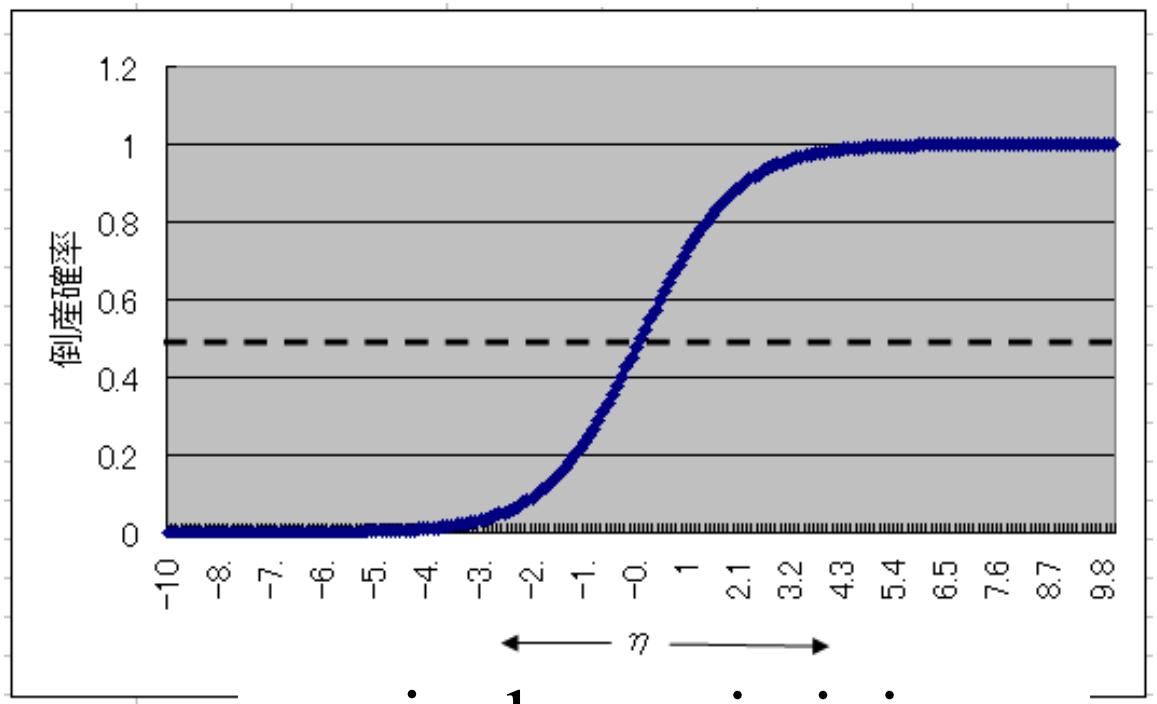
（誤判定）確率

$$\eta = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3 + a_4 X_4$$

$$X_1, X_2, X_3, X_4 \quad (\text{計測データ})$$

判別分析

- ロジスティクス関数 $\Phi(\eta)$



判別の実際（パラメータ推定）

	A	B	C	D	E	F
1			教師データ(80個)			
2	data	1	2	3	4	正解
3	1	5.4	3.7	1.5	0.2	1
4	2	4.8	3.4	1.6	0.2	1
5	3	4.8	3	1.4	0.1	1
6	4	4.3	3	1.1	0.1	1
7	5	5.8	4	1.2	0.2	1
8	6	5.7	4.4	1.5	0.4	1
9	7	5.4				
10	8	5.1				
11	9	5.7				
12	10	4.9				
13	11	5				
14	12	5.5				
15	13	7				
16	14	6.4				

数理計画
モデル

入力 → **NUOPT** → 出力

判別モデルパラメータ

a0	a1	a2	a3	a4
6.545	2.060	5.038	−10.420	−6.323

非線形最適化

アルゴリズム

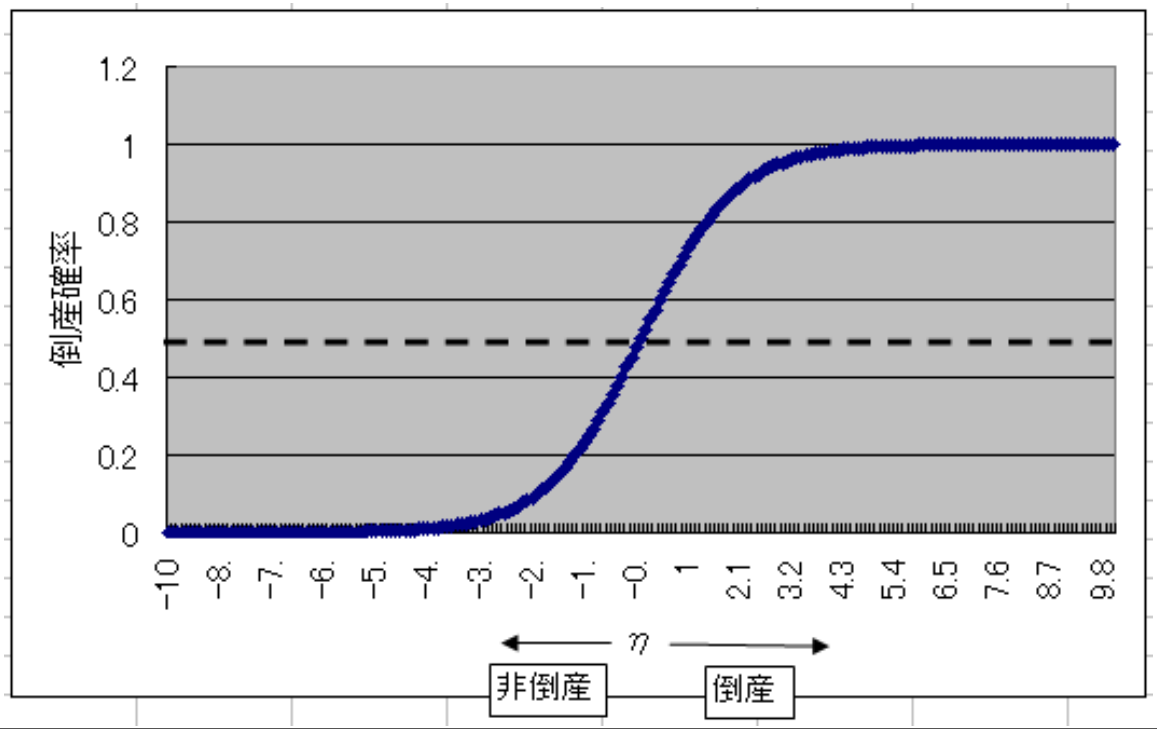
判別の実際（予測力検証）

	予測力検証（20個）						
ケースNo.					正解	予測	η
1	5.1	3.5	1.4	0.2	1	1	18.831
2	5	3.6	1.4	0.2	1	1	19.129
3	5.7	2.8	4.1	1.3	0	0	−18.549
4	4.9	3	1.4	0.2	1	1	15.900
5	4.7	3.2	1.3	0.2	1	1	17.537
6	4.6	3.1	1.5	0.2	1	1	14.744
7	6.2	2.9	4.3	1.3	0	0	−19.099
8	5.1	2.5	3	1.1	0	0	−8.570
9	5.7	3	4.2	1.2	0	0	−17.951
10	5.7	2.9	4.2	1.3	0	0	−19.087
11	5.4	3.9	1.7	0.4	1	1	17.073
12	5	2.3	3.3	1	0	0	−12.277
13	5.6	2.7	4.2	1.3	0	0	−20.301
14	4.6	3.4	1.4	0.3	1	1	16.665
15	4.4	2.9	1.4	0.2	1	1	14.366
16	4.9	3.1	1.5	0.1	1	1	15.994
17	5.5	2.6	4.4	1.2	0	0	−22.462
18	6.1	3	4.6	1.4	0	0	−22.559
19	5.8	2.6	4	1.2	0	0	−17.676
20	5	3.4	1.5	0.2	1	1	17.079

正解率100％

判別分析

- ロジスティクス関数 $\Phi(\eta)$



ηの値によって倒産確率が決定

η:財務諸表の関数形

- 線形モデル

$$\eta = a_0 + a_1 X_1 + a_2 X_2 + a_3 X_3 + a_4 X_4$$

$$\equiv a_0 + \mathbf{a}^t \mathbf{X}$$

$X_1, X_2, X_3, X_4 \ (\equiv \mathbf{X})$　財務諸表の値

$a_0, a_1, a_2, a_3, a_4 (\equiv \mathbf{a})$　パラメータ

η:財務諸表の値の関数

- 二次モデル

$$\eta = a_0 + \mathbf{a}^t\mathbf{X} + \mathbf{X}^t\mathbf{Q}\mathbf{X}$$

表現力
増大

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{bmatrix} \equiv \mathbf{Q}$$

パラメータ
(二次項)

倒産判別のための数理計画モデル

- 変数

$$a_0, \mathbf{a}, \mathbf{Q}\ （パラメータ）$$

- 目的関数

倒産データについては $\dfrac{e^{\eta}}{1+e^{\eta}}$ が１に近い

存続データについては $\dfrac{e^{\eta}}{1+e^{\eta}}$ が０に近い

$$\eta = a_0 + \mathbf{a}^t\mathbf{X} + \mathbf{X}^t\mathbf{Q}\mathbf{X}$$

倒産判別のための数理計画モデル

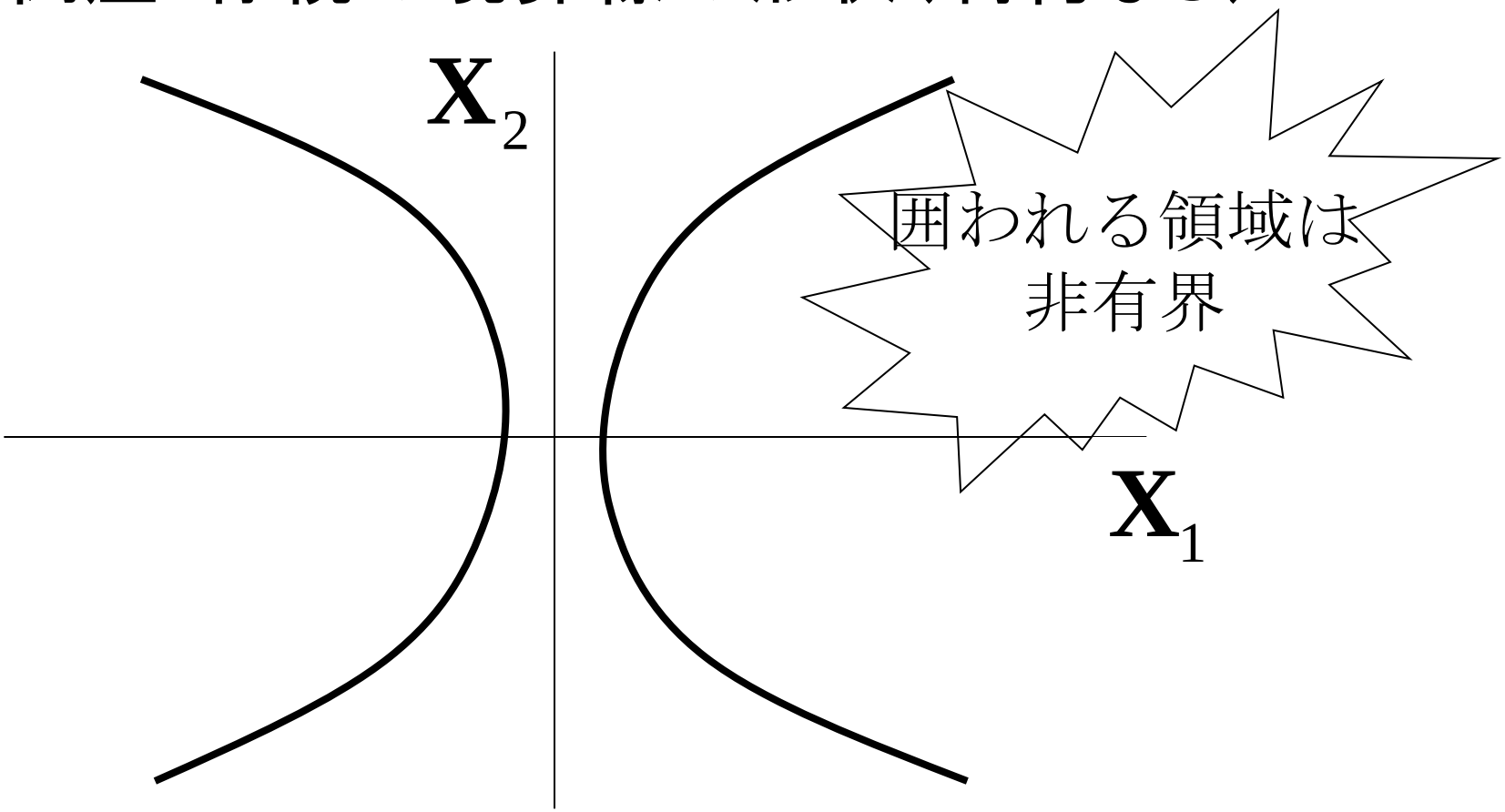
- 制約

Qが半正定値

予測力向上

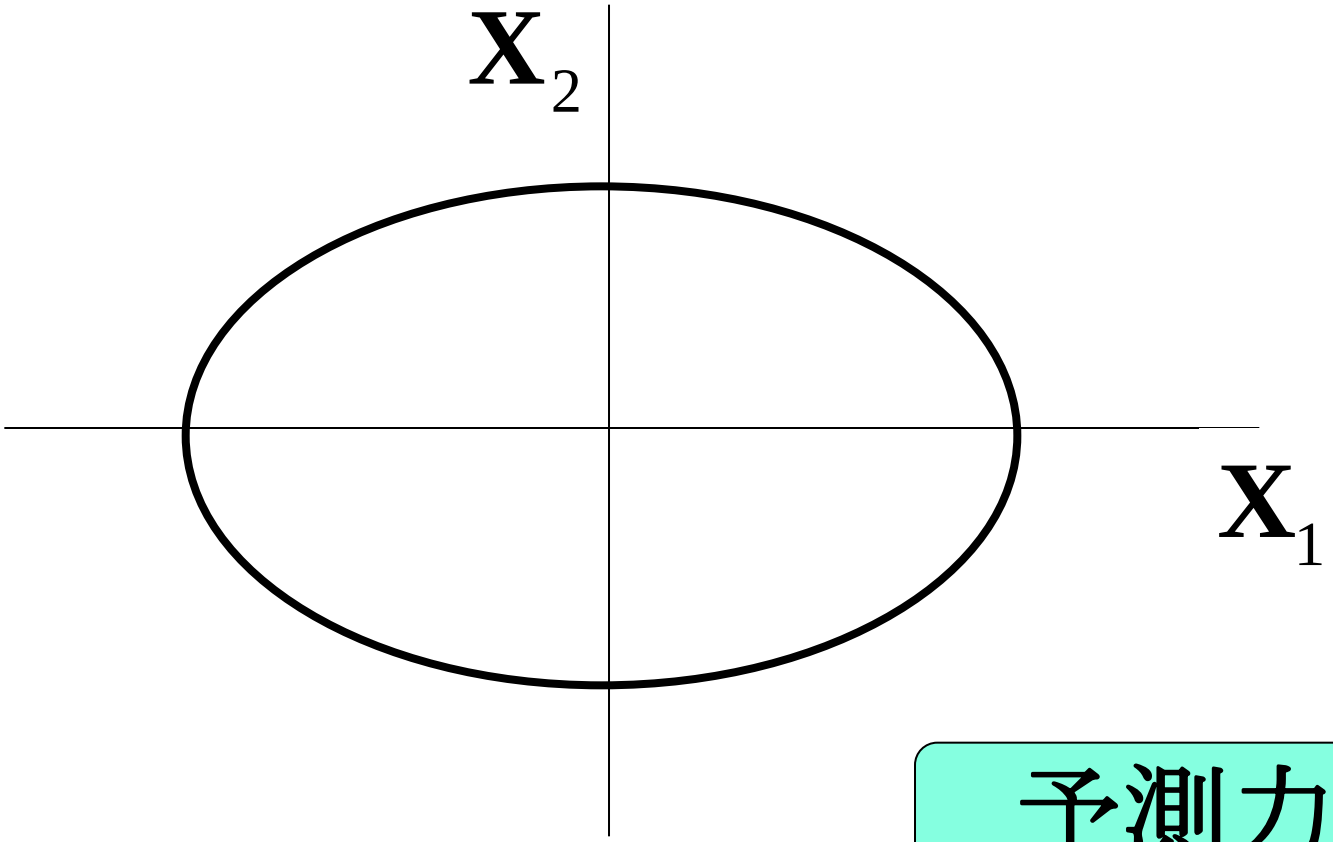
半正定値制約の意義

- 倒産・存続の境界線の形状（制約なし）



半正定値制約の意義

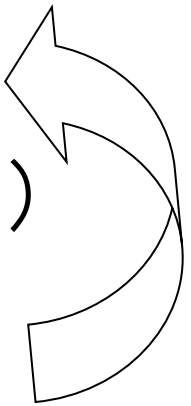
- 倒産・存続の境界線の形状（制約あり）



予測力向上

切除平面法による解法

1. 内点法起動
2. 半正定値性のチェック（OKなら終了）
3. カットの追加，1へ



ループ（～３０回）

切除平面法.vs.非線形SDP用内点法

方法	*尤度*	*計算時間*
切除平面法	143.745	30.3秒
非線形SDP	143.747	1.5秒

（使用計算機：Pentium4 3.2GHz,メモリ1GBytes）

ノルム最小化問題

Given

最小化　$\|G - X\|_F$

制約　$X \succ 0$

非線形ＳＤＰの応用例
⇒相関行列の生成

相関を持った乱数を発生したい

- 相関行列Gを恣意的に決定したい
  - アセット1とアセット2は逆相関だから
    G[1,2] = -0.6くらいかな
  - アセット2とアセット3は強い相関があるので
    G[2,3] = 0.8 にしたい
  - . . .

相関行列は $G = LL^T$ と分解できねばならない
⇒ 正定値でなければならない

正定値？

- 固有値がすべて正
- あらゆる v について

$$v^t G v > 0$$

無限個の制約が必要

半正定値：最小固有値で０を許す

半正定値行列
ノルム最小化問題

最小化　　$\|G - X\|_F$

制約　　$X - \varepsilon \cdot I$: 半正定値

最も近い相関行列の生成

SIMPLEによるモデル記述

```
Set N; // 行列の各要素
Element i(set=N),j(set=N);
Variable  x(index=(i,j));
Parameter a(index=(i,j)); // 与えられた行列の要素
SymmetricMatrix m((i,j));
Parameter minEig; // 出力される相関行列の最小固有値
m[i,i] = 1 - minEig;      // 対角は 1 とする
m[i,j] = x[i,j], i > j;   // 下三角部分の定義
m >= 0; // 半正定値制約
Objective diffnrm(type=minimize); // 差の行列のノルム
diffnrm = sum(pow(x[i,j]-a[i,j],2),(i,j,i < j));
```

実行

A	1	2	3	4	5	6	7	8	9	10
1	1.00									
2	0.17	1.00								
3	0.17	0.10	1.00							
4	0.90	0.90	0.20	1.00						
5	0.80	0.20	0.00	0.20	1.00					
6	0.00	0.40	0.00	0.20	0.40	1.00				
7	0.20	0.00	0.00	0.20	0.00	0.00	1.00			
8	0.00	0.00	0.00	0.20	0.00	0.00	0.00	1.00		
9	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	
10	0.70	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

NUOPT
最適化

X	1	2	3	4	5	6	7	8	9	10
1	1.00									
2	0.27	1.00								
3	0.15	0.10	1.00							
4	0.62	0.84	0.21	1.00						
5	0.59	0.16	0.01	0.32	1.00					
6	0.05	0.41	0.00	0.17	0.38	1.00				
7	0.18	0.00	0.00	0.21	0.01	0.00	1.00			
8	0.03	0.01	0.00	0.18	−0.01	0.00	0.00	1.00		
9	0.59	−0.04	0.01	0.12	0.09	−0.02	0.01	−0.01	1.00	
10	0.51	−0.04	0.01	0.11	0.08	−0.02	0.01	−0.01	0.08	1.00

ロバストポートフォリオ

- 収益率のぶれ（線形）

$$U_r = \{r \,||\, r - \bar{r} \,| \leq \delta\}$$

- 収益率のぶれ（二次）

$$U_r = \{r \,|\, (r - \bar{r})^t \sigma_r^{-1} (r - \bar{r}) \leq \delta^2\}$$

- 共分散のぶれ

$$U_Q = \{Q \,|\, Q^U \geq Q \geq Q^L\}$$

non-ロバスト

- マルコビッツモデル

リスク回避係数

$$\max_{x} \; r^T x - \lambda x^T Q x$$

$$s.t. \quad x^T e = 1$$

Qの変動を考慮しない

ロバスト

- 分散のぶれを考慮

$$\max_{w} \left\{ r^T x - \lambda \max_{Q_L \le Q \le Q_U} \left\{ x^T Q x \right\} \right\}$$

$$s.t. \quad x^T e = 1$$

ロバストマルコビッツモデル

- 等価な問題への変形

$$\max_{x} \quad r^T x - \lambda\left(Q_U \bullet U - Q_L \bullet L\right)$$

$$s.t. \quad x^T e = 1$$

$$\begin{pmatrix} U-L & x \\ x^T & 1 \end{pmatrix} \succeq 0$$

$$U \geq 0, L \geq 0$$

Fabozzi Kolm, Pachamanova Focardi,
Robust Portfolio Optimization and Management,2007

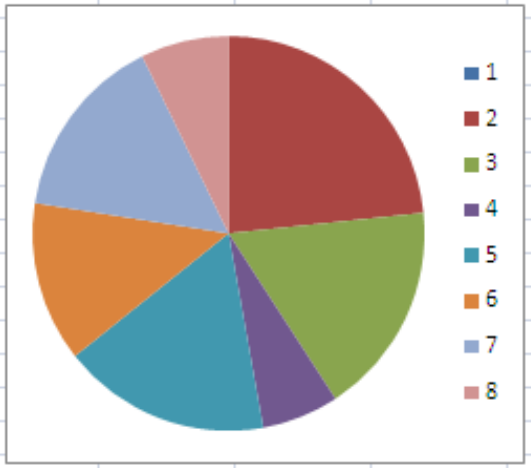
実行

Robustポートフォリオ

下限	1	2	3	4	5	6	7	8
1	1.9	-1.4	-1	-1	1	-0.5	-1	0.1
2	-1.4	3.5	-1.5	0.5	-1.2	-1.2	0.4	0.6
3	-1	-1.5	5	-1.125	1.1	0.9	0.3	-0.2
4	-1	0.5	-1.125	6	1.5	0.4	1.2	-0.9
5	1	-1.2	1.1	1.5	4.5	-1.4	0.15	-2
6	-0.5	-1.2	0.9	0.4	-1.4	9	-1	0.5
7	-1	0.4	0.3	1.2	0.15	-1	5.5	-1.25
8	0.1	0.6	-0.2	-0.9	-2	0.5	-1.25	11.5

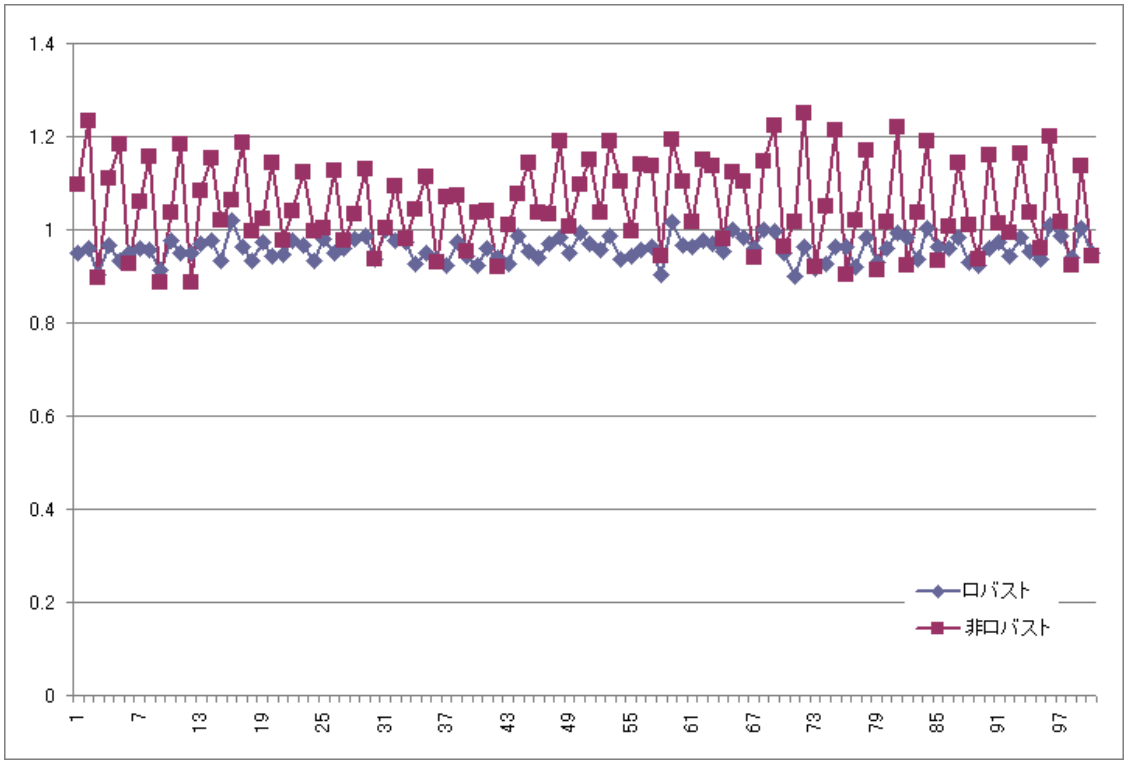
上限	1	2	3	4	5	6	7	8
1	3	1	2.5	-0.9	1	-0.4	2.5	2
2	1	4.5	-1.4	1.2	0.5	-1.1	0.5	2.6
3	2.5	-1.4	6	-0.5	1.2	1	0.4	-0.1
4	-1	1.2	-0.5	6.5	1.6	0.5	1.3	-0.8
5	1	0.5	1.2	1.6	5.5	-1.3	0.16	-1.9
6	-0.4	-1.1	1	0.5	-1.3	11	-0.9	0.6
7	2.5	0.5	0.4	1.3	0.16	-0.9	6.5	-1.2
8	2	2.6	-0.1	-0.8	-1.9	0.6	-1.2	13.5

ポートフォリオ	0.0000	0.2332	0.1753	0.0634	0.1710	0.1319	0.1529	0.0723



ロバスト性の検証

- 正定値行列100ケースについて
  リスクを比較



端株処理

- 組み入れ比率 ⇒ 購入量を得たい

$$rd_j^{long} = \left\lfloor \frac{\tilde{y}_j^{long}}{mb_j} \right\rfloor \cdot mb_j$$

理想購入量

最小取引金額
（価格情報含む）

$$y_j^{long} = rd_j^{long} + mb_j \cdot d_j^{long}$$

実現可能購入量

丸めは上か下か
（０－１変数）

端株処理

- 目的関数（例）
  - 指定した購入，売却量との差

$$\left| \left(\sum_{1 \le j \le |T|} y_j^{long} \right) - M^{long} \right|^2 + \left| \left(\sum_{1 \le j \le |T|} y_j^{short} \right) - M^{short} \right|^2$$

  - 理想購入量との差

$$\sum_{1 \le j \le |T|} \left| y_j^{long} - \tilde{y}_j^{long} \right|^2 + \sum_{1 \le j \le |T|} \left| y_j^{short} - \tilde{y}_j^{short} \right|^2$$

  - その他リスク尺度

銘柄分割

- ポートフォリオを全体属性
（金額，ファクター構成）の制約下で
グループ分け

$$\sum_{k} u_{j,k} = 1$$

銘柄jはグループk
に属するか（０－１変数）

銘柄分割

- 制約

銘柄jのmという属性の値

グループkの属性値

$$F_k^m \equiv \sum_{j \in J} f_j^m u_{j,k}$$

$$F_{HIGH}^m \geq F_k^m \geq F_{LOW}^m$$

属性値に関する制約

メタヒューリスティクスアルゴリズムの利用

- 離散計画について
  妥当な時間で良質な実行可能解

⇒ グルーピング
端株処理
には好適

NUOPTに導入されている
数理計画法アルゴリズム

- 線形計画法(単体法、内点法)
- 二次計画法(有効制約法、内点法)
- 非線形計画法(内点法，逐次二次計画法)
- 非線形半正定値計画(内点法)
- 線形混合整数計画法(分枝限定法)
- 混合整数二次計画法(分枝限定法)
- 非線形整数計画法(分枝限定法)
- 制約充足問題(タブ・サーチ)
- 資源制約付プロジェクトスケジューリング(タブ・サーチ)