数理工学実験 課題4 最小2乗法

2023年11月17日

課題1(最小2乗推定)

入力ベクトル $\boldsymbol{x}_i \in \mathbb{R}^2$ に対して出力 $y_i \in \mathbb{R}$ が

$$y_i = \boldsymbol{x}_i^T \boldsymbol{\theta} + w_i \tag{1}$$

となっているような系を考える。ここで、$\boldsymbol{\theta} \in \mathbb{R}^2$ はパラメーターであり、$w_i (i = 1, 2, \dots)$ は互いに独立に $\mathcal{N}(0, 1)$ に従う確率変数とする。

(1)

問題の概要

10000個の入出力データ (x_{i1}, x_{i2}, y_i) $(i = 1, 2, \dots, 10000)$ を格納したファイル".txt"について最小2乗法を用いることで系のパラメータ $\boldsymbol{\theta}$ を推定し、その推定誤差共分散行列を求める。

問題の解法

最小2乗法を用いると推定パラメータ θ は

$$\hat{\boldsymbol{\theta}} = \{ \sum_{i=1}^{10000} (\begin{bmatrix} x_{1i} \\ x_{2i} \end{bmatrix} \begin{bmatrix} x_{1i} & x_{2i} \end{bmatrix}) \}^{-1} \{ \sum_{i=1}^{10000} (\begin{bmatrix} x_{1i} & x_{2i} \end{bmatrix} y_i) \} \tag{2}$$

と求められる。また、誤差 w_i について

$$\boldsymbol{E}[w_i w_j] = \delta_{ij} \tag{3}$$

であることが分かっている (*) ので、推定誤差共分散行列は

$$\boldsymbol{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T] = \{ \sum_{i=1}^{10000} (\begin{bmatrix} x_{1i} \\ x_{2i} \end{bmatrix} \begin{bmatrix} x_{1i} & x_{2i} \end{bmatrix}) \}^{-1} \tag{4}$$

数値計算の結果

前項で述べた解法をJuliaで実装し、データ対".txt"について数値計算を行った。このとき、$\boldsymbol{\theta}$ の推定値 $\hat{\boldsymbol{\theta}}$ は

$$\hat{\boldsymbol{\theta}} = \begin{bmatrix} 1.5065508075931555 \\ 1.9976956653988034 \end{bmatrix} \tag{5}$$

となり、推定誤差共分散行列 $\boldsymbol{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T]$ は

$$\boldsymbol{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T] = \begin{bmatrix} 9.8794 \times 10^{-5} & -4.087 \times 10^{-7} \\ -4.087 \times 10^{-7} & 0.000100656 \end{bmatrix} \tag{6}$$

となった。推定誤差共分散行列の値が0に近いことから分かるように (誤差 w_i が互いに独立な標準正規分布に従うという仮定のもとで) 最小2乗法が良いパラメータ推定を与えることが分かる。

補足

本問の (1) において誤差 w_i について式 (3) の関係が成り立っていることを既知としたが、共分散の情報が分かっていない場合は何らかの形で観測誤差の共分散行列を求める必要がある。このような場合は最小 2 乗推定を行った上で観測誤差共分散行列の推定値 $\hat{V}$ を求める必要がある。具体的には最小 2 乗誤差の推定値 $\hat{\boldsymbol{\theta}}$ を用いて

$$\hat{V} = \frac{1}{N-2}\sum_{i=1}^{N}(y_i - \begin{bmatrix} x_{i1}, x_{i2} \end{bmatrix}\hat{\boldsymbol{\theta}})(y_i - \begin{bmatrix} x_{i1}, x_{i2} \end{bmatrix}\hat{\boldsymbol{\theta}})^T \tag{7}$$

と求める。このようにして観測誤差共分散行列 $\hat{V} \in R$ を求めると、

$$\hat{V} = 0.9986937409125233 \tag{8}$$

となり、この値を用いると、推定誤差共分散行列は

$$\boldsymbol{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T] = (\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})^{-1}(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\hat{V}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})^{-1} \tag{9}$$

$$= \hat{V}(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})^{-1}(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})^{-1} \tag{10}$$

$$= \hat{V}(\sum_{i=1}^{N}\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\begin{bmatrix} x_{i1}x_{i2} \end{bmatrix})^{-1} \tag{11}$$

と書ける。この値を数値計算すると

$$\boldsymbol{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T] = \begin{bmatrix} 9.86649 \times 10^{-5} & -4.08166 \times 10^{-7} \\ -4.08166 \times 10^{-7} & 0.000100525 \end{bmatrix} \tag{12}$$

となる。いずれにせよ、推定誤差共分散行列の値が 0 に近いことから (誤差 w_i が互いに独立な標準正規分布に従うという仮定のもとで) 最小 2 乗法が良いパラメータ推定を与えることが分かる。

(2)

問題の概要

10000 個の入出力データ (x_{i1}, x_{i2}, y_i) $(i = 1, 2, \ldots, 10000)$ を格納したファイル”.txt”について最小 2 乗法を用いることで求めた推定パラメータ $\hat{\boldsymbol{\theta}}$ について、最小 2 乗推定に用いるデータ数 N を様々に変え、N を大きくしたときの $\hat{\boldsymbol{\theta}}$ の挙動について調べる。

問題の解法

問題 (1) で用いたソースコードを流用することができるようにソースコードを構築している。具体的には関数を

- linreg(N)：データ $(x_{i1}, x_{i2}, y_i)_{i=1}^{N}$ について最小 2 乗推定量を求める関数
- Vw(N)：データ $(x_{i1}, x_{i2}, y_i)_{i=1}^{N}$ について推定誤差共分散行列を出力する関数

のように構築している。このような関数を用いて $N = 2^i (i = 1, \ldots, 13)$ について最小 2 乗推定量 $\hat{\boldsymbol{\theta}}$ を求め、$\hat{\boldsymbol{\theta}}$ の各成分について底を 2 として片対数プロットする。

数値計算の結果

結果は図 1 および 2 のようになった。

図 1 から、$\hat{\theta}_{1,N}$ が N の値を大きくするにつれておよそ 1.5 という値に漸近していくことが見て取れる。また、図 2 から、$\hat{\theta}_{2,N}$ が N の値を大きくするにつれておよそ 2 という値に漸近していくことが見て取れる。

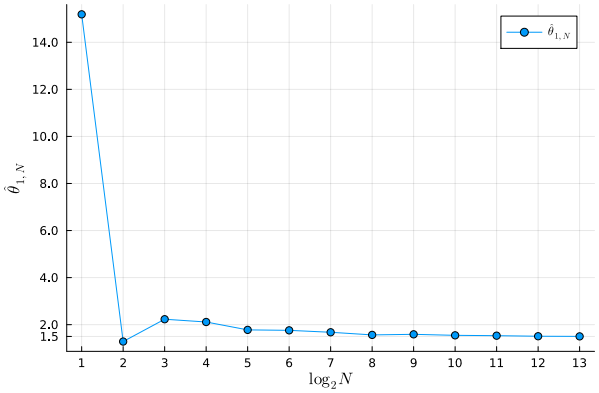

図 1: データ数 N と、N 個のデータを用いて求めた最小 2 乗推定量 $\hat{\boldsymbol{\theta}}_N$ の第 1 成分 $\hat{\theta}_{1,N}$ の関係

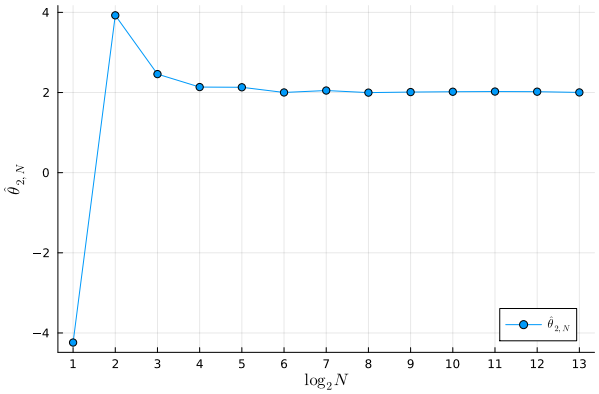

図 2: データ数 N と、N 個のデータを用いて求めた最小 2 乗推定量 $\hat{\boldsymbol{\theta}}_N$ の第 2 成分 $\hat{\theta}_{2,N}$ の関係

(3)

問題の概要

線形回帰モデル (1) がどの程度データに当てはまっているかを評価する指標として**決定変数**がある。系 (1) について、データを 10000 個全て使った場合の決定変数 C は

$$C = \frac{\sum_{i=1}^{10000}(\begin{bmatrix} x_{i1} & x_{i2} \end{bmatrix} \hat{\boldsymbol{\theta}} - \bar{y})^2}{\sum_{i=1}^{10000}(y_i - \bar{y})^2} \tag{13}$$

と求められる。この決定変数を求める。

問題の解法

データを 10000 個全て使った場合の最小 2 乗推定量 $\hat{\boldsymbol{\theta}}$ を表式 (13) をに代入することで決定変数を出力する。

数値計算の結果

数値計算の結果、決定変数 C の値は

$$C = 0.8629734364522683 \tag{14}$$

として得られた。

決定変数の値が 1 に近いほど線形回帰式のデータへの当てはまりが良いということが式 (13) から見て取れる。(14) の値は線形回帰の当てはまりが一定程度良いことを示している。

今回は決定変数の値が一定程度 1 に近い値だったので推定の当てはまりが良いと結論できる。しかし、決定変数の値の大小で推定の良し悪しを判断することは (特殊な仮定を置かない限り) できない。たとえば、式 (1) について誤差 ω_i の分散が大きい場合は決定変数の値が小さくなるが、決定変数の値が小さいからといって推定自体が悪いとは結論できない。決定変数は単に当てはまりの良さを示しているだけで、推定の良さを示しているわけではない点に注意が必要である。

課題 8(一般化最小 2 乗推定・推定量の合成)

問題の概要

入力 $x_i \in \mathbb{R}$ に対して出力 y_i が

$$y_i = \varphi(x_i)\boldsymbol{\theta} + w_i \tag{15}$$

$$\varphi(x_i) = \begin{bmatrix} 1 & \exp(-\frac{(x_i-1)^2}{2}) & \exp(-(x_i+1)^2) \end{bmatrix} \tag{16}$$

という規則で出力される系を考える。ただし、このとき $\boldsymbol{\theta}$ はパラメータであり、w_i は $1 \leq i \leq 6000$ と $6001 \leq i \leq 10000$ でそれぞれ異なる平均 0 の独立な分布を持つ観測誤差である。このような系において入力値と出力値の組 $\{(x_i, y_i)\}_{i=1}^{10000}$ が"input.txt"として与えられたとき、

- $1 \leq i \leq 6000$ までの観測誤差の共分散を推定する
- $6001 \leq i \leq 10000$ までの観測誤差の共分散を推定する
- $1 \leq i \leq 6000$ と $6001 \leq i \leq 10000$ について最小 2 乗推定量を重みをつけて合成する
- $1 \leq i \leq 10000$ まで重みをつけずに求めた最小 2 条推定量と比較する

の 4 つの数値計算を実行する。

問題の解法

共分散の推定

各データについて最小 2 乗推定量 $\hat{\boldsymbol{\theta}}$ を求め、最小 2 乗推定量を用いて観測誤差の共分散の推定値 $\hat{V}$ を

$$\hat{V} = \frac{1}{N-3}\sum_{i=1}^{N}(y_i - \varphi(x_i)\hat{\theta})(y_i - \varphi(x_i)\hat{\theta})^T \tag{17}$$

の表式によって求める。(ただし、ここで N はデータ数を表す。)

最小 2 乗推定量の合成

いま、N 個の入出力データに対して 2 乗誤差の最適化問題

$$\min_{\theta \in \mathbb{R}^3} \sum_{i=1}^{N}(y_i - \varphi(x_i)\boldsymbol{\theta})^T(y_i - \varphi(x_i)\boldsymbol{\theta}) \tag{18}$$

を考える。この最適化問題の最適解は

$$\boldsymbol{\theta} = (\sum_{i=1}^{N}\varphi(x_i)^T\varphi(x_i))^{-1}(\sum_{i=1}^{N}\varphi(x_i)^T y_i) \tag{19}$$

と与えられることが分かっている。式 (19) は誤差 w_i が共分散 $\sigma^2 I$(I は単位行列) をもつような場合に推定誤差の共分散を最小にすることが知られている。しかし、共分散行列が $V \neq \sigma^2 I$ である場合は、推定誤差の共分散を最小にするようなパラメータ $\boldsymbol{\theta}$ は

$$\boldsymbol{\theta} = (\sum_{i=1}^{N}\varphi(x_i)^T V^{-1}\varphi(x_i))^{-1}(\sum_{i=1}^{N}\varphi(x_i)^T V^{-1} y_i) \tag{20}$$

と与えられる。この $\boldsymbol{\theta}$ は、重みつき最適化問題

$$\min_{\theta \in \mathbb{R}^3} \sum_{i=1}^{N}(y_i - \varphi(x_i)\boldsymbol{\theta})^T Q_i(y_i - \varphi(x_i)\boldsymbol{\theta}) \tag{21}$$

$$Q_i = V^{-1} \ (i = 1, 2, \ldots, N) \tag{22}$$

の解に相当する。

いま、$i = 1, 2, \ldots, 6000$ について観測誤差の共分散の推定値を V_{6000} とし、式 (20) の式を用いて得られた $\boldsymbol{\theta}$ を $\hat{\boldsymbol{\theta}}_{6000}$ とおく。また、$i = 6001, 6002, \ldots, 10000$ について観測誤差の共分散の推定値を V_{4000} とし、式 (20) の式を用いて得られた $\boldsymbol{\theta}$ を $\hat{\boldsymbol{\theta}}_{4000}$ とおく。さらに

$$\Phi_{6000} = (\sum_{i=1}^{6000} \varphi(x_i)^T V_{6000} \varphi(x_i))^{-1} \tag{23}$$

$$\Phi_{4000} = (\sum_{i=6001}^{10000} \varphi(x_i)^T V_{4000} \varphi(x_i))^{-1} \tag{24}$$

とおく。このとき、重みつき最適化問題

$$\min_{\theta \in \mathbb{R}^3} \sum_{i=1}^{10000} ((y_i - \varphi(x_i)\boldsymbol{\theta})^T Q_i (y_i - \varphi(x_i)\boldsymbol{\theta})) \tag{25}$$

$$Q_i = V_{6000} \ (i = 1, 2, \ldots, 6000) \tag{26}$$

$$Q_i = V_{4000} \ (i = 6001, 6002, \ldots, 10000) \tag{27}$$

の解 $\boldsymbol{\theta}$ は

$$\boldsymbol{\theta} = (\Phi_{6000}^{-1} + \Phi_{4000}^{-1})^{-1} (\Phi_{6000}^{-1} \hat{\boldsymbol{\theta}}_{6000} + \Phi_{4000}^{-1} \hat{\boldsymbol{\theta}}_{4000}) \tag{28}$$

として与えられることが知られている。これが重みつき合成に相当する。

全てのデータによる最小 2 乗推定量

データ $\{(x_i, y_i)\}_{i=1}^{10000}$ を全て使って最小 2 乗推定量を求める方法は、課題 1 と全く同じなので省略する。

本問ではデータの分散が均一でないような入出力データを考えているので、決定変数による線形回帰の能力評価は使えない。従って、決定変数とは別の方法で線形回帰の能力評価を行う必要があるが、本問では与えられた正解

$$\boldsymbol{\theta} = \begin{bmatrix} 0.1 \\ 3.1 \\ 2.1 \end{bmatrix} \tag{29}$$

との比較によって線形回帰の能力評価を行う。すなわち、全てのデータによる最小 2 乗推定量・重みつき合成によって求めた推定量・正解のパラメータの値を比較することで線形回帰の能力比較を行う。

数値計算の結果

共分散の推定

”問題の解法”の”共分散の推定”の小節で示した方法を用いて、$i = 1, 2, \ldots, 6000$ の共分散の推定値 V_{6000} および i =6001, 6002, . . . , 10000 の共分散の推定値 V_{4000} を計算すると、

$$V_{6000} = 96.86329354733185 \tag{30}$$

$$V_{4000} = 0.010304240740875473 \tag{31}$$

となる。1:6000 のデータの方がデータの分散がはるかに大きく、最小 2 乗法を素朴に適用すると推定パラメータが外れ値の影響を受けてしまうので、重みつき合成によってパラメータを求めた方がよりよいパラメータ推定を行える、ということが結論できる。

最小 2 乗推定量の合成

”問題の解法”の”最小 2 乗推定量の合成”の小節で示した方法を用いて、$i = 1, 2, \ldots, 6000$ のデータについての最小 2 乗誤差推定量 $\hat{\boldsymbol{\theta}}_{6000}$ (= BLUE) および V_{6000} および $i = 6001, 6002, \ldots, 10000$ のデータについての最小 2 乗誤差推定量 $\hat{\boldsymbol{\theta}}_{4000}$ (= BLUE) を合成すると、重みつき最小 2 乗問題 (25) の解 $\boldsymbol{\theta}$ は

$$\boldsymbol{\theta} = \begin{bmatrix} 0.09846858600774328 \\ 3.1004337093449115 \\ -2.091018211502674 \end{bmatrix} \tag{32}$$

となる。

全てのデータによる最小 2 乗推定量

全てのデータを用いて最小 2 乗推定量 $\hat{\boldsymbol{\theta}}$ を求めると

$$\hat{\boldsymbol{\theta}} = \begin{bmatrix} 0.04373209111048817 \\ 3.2086656240553584 \\ -2.1511785561287113 \end{bmatrix} \tag{33}$$

となる。重みつき合成によって求めたパラメータ $\hat{\boldsymbol{\theta}}_w$ は

$$\hat{\boldsymbol{\theta}}_w = \begin{bmatrix} 0.09846858600774328 \\ 3.1004337093449115 \\ -2.091018211502674 \end{bmatrix} \tag{34}$$

となる。いま、正解のパラメータは

$$\boldsymbol{\theta} = \begin{bmatrix} 0.1 \\ 3.1 \\ 2.1 \end{bmatrix} \tag{35}$$

と分かっているので、

$$\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\|_2 \approx 0.13264064245003684 \tag{36}$$

$$\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_w\|_2 \approx 0.009121724465376098 \tag{37}$$

となり、重みつき合成によって求めたパラメータ $\hat{\boldsymbol{\theta}}_w$ の方がより良い推定をしていることがわかる。

課題 11(カルマンフィルタ)

離散時間変化する状態変数 θ_i があって、それに伴って出力 y_i を発生する系があったとする。このとき、y_i と θ_i について

$$\theta_{i+1} = 0.9\theta_i + v_i \tag{38}$$

$$y_i = 2\theta_i + w_i \tag{39}$$

となっていることは既知とする。(ここで v_i, w_i は各離散時間で独立に $\mathcal{N}(0,1)$ に従う確率変数であるとする。) また、θ_0 は平均 2, 分散 3 の分布から発生する確率変数とする。

問題の概要

系 (38), (39) というモデルについて、出力 y_i の観測のみで θ_i の推定を行うような手法として**カルマンフィルタ**が知られている。カルマンフィルタでは、θ_0, V_0 として適当な値を仮定して、

$$\hat{\theta}_{i+1} = 0.9\hat{\theta}_i + F_{i+1}(y_i - 1.8\theta_i) \tag{40}$$

$$X_{i+1} = 0.81V_i + 1 \tag{41}$$

$$V_{i+1} = \frac{X_{i+1}}{4X_{i+1} + 1} \tag{42}$$

$$F_{i+1} = \frac{2X_{i+1}}{4X_{i+1} + 1} \tag{43}$$

という手続きで θ_i の推定を行う。

本問では疑似乱数 $\mathcal{N}(3,2)$ から θ_0 を発生し、さらに別の疑似乱数を用いることで式 (38) と式 (39) に従ってデータセット $\{(\theta_i, y_i)\}_{i=0}^{100}$ を生成した上で、データ $\{y_i\}_{i=0}^{100}$ に対して、$\boldsymbol{E}[\theta_0] = 3, \boldsymbol{E}[(\theta_0 - \bar{\theta}_0)^2] = 2$ と仮定したときのカルマンフィルタを適用して θ_i の推定値 $\{\hat{\theta}_i\}_{i=0}^{100}$ を求める。そして、数値計算によって求めた推定値 $\{\hat{\theta}_i\}_{i=0}^{100}$ を実際の $\{\theta_i\}_{i=0}^{100}$ と比較する。

数値計算の結果

カルマンフィルタの再帰式 (40), (41), (42), (43) を用いてパラメータ θ_i を推定した結果は図 3 のようになった。(上でも述べたが、θ_0 が平均 3 分散 2 の確率分布にしたがって発生することを既知として、$\hat{\theta_0} = 3, V_0 = 2$ とした。)

状態変数の漸化式 (38)、出力の決定式 (39)、θ_0 の平均・分散の情報、および混入誤差の分布の情報が分かっているだけで、出力 y_i の情報だけから状態変数 θ_i の推定を小さい誤差で行えることが図 3 から分かる。カルマンフィルタは、出力だけで状態変数を求めるような工学的問題 (人工衛星・ロケットなどの位置・速度の推定など) に応用できることが知られている。

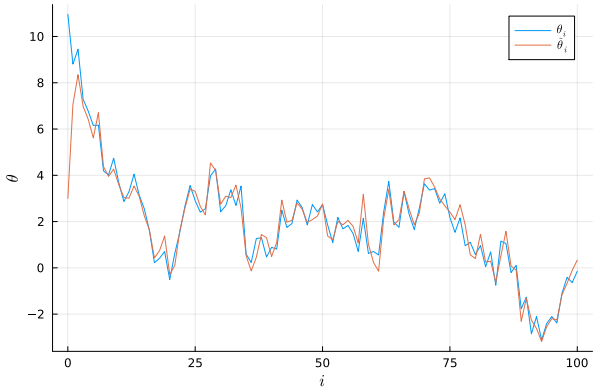

図 3: θ_0 が平均が 3, 分散が 2 の分布に従うことおよび混入誤差が $\mathcal{N}(0,1)$ という分布に従うことを既知とした (あるいは仮定した) 場合に、カルマンフィルタを用いて時系列の出力 y_i から状態変数 θ_i を各時間で推定した結果 $\hat{\theta}_i$ を示す。推定の正確度合いを示すために真の値 θ_i も重ねてプロットしている。

補足

カルマンフィルタの原理

カルマンフィルタの原理について解説する。今、系

$$\theta_{k+1} = a_{k+1}\theta_k + v_i \tag{44}$$

$$y_k = b_k\theta_k + w_i \tag{45}$$

について、

- 時刻 $k=0$ における θ_0 は平均 $\bar{\theta}$、分散 P の分布に従って生じる
- 時刻 $k=i$ において出力 y_i は観測できる
- 時刻 $k=i$ における状態変数 θ_i は分からない
- 全ての時刻 $k=i$ における a_i, b_i は既知である
- v_i, w_i は各時刻で独立にそれぞれ $\mathcal{N}(0,\sigma_v^2), \mathcal{N}(0,\sigma_w^2)$ に従う確率変数

とするとき、出力 y_i の情報だけから状態変数 θ_i の推定を行う。ただし、推定方法として、各時刻の推定パラメータを $\hat{\theta}_i$ として

$$\hat{\theta}_i = a_i\hat{\theta}_{i-1} + F_i(y_i - c_i a_i \hat{\theta}_{i-1}) \tag{46}$$

という関係式を用いて逐次的に行う。(この $F_i(y_i - c_i a_i\hat{\theta}_{i-1})$ は $\hat{\theta}_i = a_i\hat{\theta}_{i-1}$ という関係式によって推定を行なったときに、実際の値から誤差が蓄積していくのを防ぐために必要な修正項である。) このとき、推定の目標とする最適化問題は

$$\min_{F_i} \boldsymbol{E}[(\theta_i - \hat{\theta}_i)^2] \tag{47}$$

とする。この推定は不偏的である。実際に

$$\begin{aligned}
\boldsymbol{E}[\theta_i - \bar{\theta}_i] &= \boldsymbol{E}[a_i\theta_{i-1} + v_i - a_i\hat{\theta}_{i-1} - F_i(y_i - c_i a_i\hat{\theta}_{i-1})] \\
&= a_i\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] + \boldsymbol{E}[v_i] - F_i\boldsymbol{E}[y_i - c_i a_i\hat{\theta}_{i-1}] \\
&= a_i\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] + \boldsymbol{E}[v_i] - F_i\boldsymbol{E}[c_i(a_i\theta_{i-1} + v_i) + w_i - c_i a_i\hat{\theta}_{i-1}] \\
&= a_i\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] + \boldsymbol{E}[v_i] - F_i\boldsymbol{E}[c_i a_i(\theta_{i-1} - \hat{\theta}_{i-1}) + c_i v_i + w_i] \\
&= a_i\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] + \boldsymbol{E}[v_i] - F_i c_i a_i\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] + F_i c_i\boldsymbol{E}[v_i] + F_i\boldsymbol{E}[w_i] \\
&= a_i(1 - F_i c_i)\boldsymbol{E}[\theta_{i-1} - \hat{\theta}_{i-1}] (\because \boldsymbol{E}[v_i] = 0, \boldsymbol{E}[w_i] = 0)
\end{aligned} \tag{48}$$

となり、$\boldsymbol{E}[\theta_0 - \hat{\theta}_0] = 0$ となるように $\hat{\theta}_0$ を選ぶと常に $\boldsymbol{E}[\theta_i - \hat{\theta}_i] = 0$ となるからである。いま、時点 $k = i-1$ における分散 $\boldsymbol{E}[(\hat{\theta}_{i-1} - \theta_{i-1})^2] := V_{i-1}$ が既知であるとすると

$$\begin{aligned}
\boldsymbol{E}[(\theta_i - \hat{\theta}_i)^2] &= \boldsymbol{E}[\{(a_i\theta_{i-1} + v_i) - (a_i\hat{\theta}_{i-1} + F_i(y_i - c_i a_i \hat{\theta}_{i-1}))\}^2] \\
&= \boldsymbol{E}[\{a_i(\theta_{i-1} - \hat{\theta}_{i-1}) + v_i - F_i(c_i(a_i\theta_{i-1} + v_i) + w_i - c_i a_i\hat{\theta}_{i-1}))\}^2] \\
&= \boldsymbol{E}[\{a_i(\theta_{i-1} - \hat{\theta}_{i-1}) + v_i - F_i c_i a_i(\theta_{i-1} - \hat{\theta}_{i-1}) - F_i(c_i v_i + w_i)\}^2] \\
&= a_i^2\boldsymbol{E}[(\theta_{i-1} - \hat{\theta}_{i-1})^2] + \boldsymbol{E}[v_i^2] + F_i^2 c_i^2 a_i^2 \boldsymbol{E}[(\theta_{i-1} - \hat{\theta}_{i-1})^2] + F_i^2\boldsymbol{E}[(c_i v_i + w_i)^2] \\
&\quad + 2a_i(1 - F_i c_i)\boldsymbol{E}[v_i(\theta_{i-1} - \hat{\theta}_{i-1})] - 2F_i\boldsymbol{E}[c_i v_i^2 + v_i w_i] - 2a_i^2 c_i F_i \boldsymbol{E}[(\theta_{i-1} - \hat{\theta}_{i-1})^2] \\
&\quad + 2F_i^2 a_i c_i \boldsymbol{E}[(\theta_{i-1} - \hat{\theta}_{i-1})(c_i v_i + w_i)] - 2a_i F_i \boldsymbol{E}[(\theta_{i-1} - \hat{\theta}_{i-1})(c_i v_i + w_i)] \\
&= a_i^2 V_{i-1} + \sigma_v^2 + F_i^2 c_i^2 a_i^2 V_{i-1} + F_i^2(c_i^2\sigma_v^2 + \sigma_w^2) - 2F_i c_i \sigma_v^2 - 2a_i^2 c_i F_i V_{i-1} \\
&= (c_i^2 a_i^2 V_{i-1} + c_i^2\sigma_v^2 + \sigma_w^2)F_i^2 - 2(c_i\sigma_v^2 + a_i^2 c_i V_{i-1})F_i + (a_i^2 V_{i-1} + \sigma_v^2)
\end{aligned} \tag{49}$$

と変形できる。ただし、ここで v_i, w_i の性質や

$$\begin{aligned}
\boldsymbol{E}[v_i(\theta_{i-1} - \hat{\theta}_{i-1})] = 0 \\
\boldsymbol{E}[w_i(\theta_{i-1} - \hat{\theta}_{i-1})] = 0
\end{aligned} \tag{50}$$

という関係を用いた。($\hat{\theta}_{i-1}$ や θ_{i-1} が v_i や w_i によらず決まる、すなわち独立であることから従う。) 式 (49) より、最適化問題 (47) の解は

$$F_i = \frac{a_i^2 c_i V_{i-1} + c_i\sigma_v^2}{a_i^2 c_i^2 V_{i-1} + c_i^2\sigma_v^2 + \sigma_w^2} \tag{51}$$

となる。また、式 (49) にこの値を代入することで V_i も求まるので、推定を逐次的に行うことができる。以上がカルマンフィルタの動作原理である。

課題 14(K-平均法)

問題の概要

与えられた 1000 個の 2 次元データ $\{(x_i, y_i)\}_{i=1}^{1000}$ を K-平均法で 3 個のクラスターに分類する。その際、K-平均法の初期点の選び方を変化させ、最も良いクラスタリングを求める。

問題の解法

K-平均法の原理を記す。問題はデータ点の集合 $\{p_i\}_{i=1}^{1000}$ を非交差な部分集合 $\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3$ で分割することに相当する。各クラスター $\mathcal{V}_i (i = 1, 2, 3)$ についてその重心 (=代表点)μ_i は

$$\mu_i = \frac{1}{|\mathcal{V}_i|}\sum_{p \in \mathcal{V}_i} p \tag{52}$$

と書ける。各クラスターについて重心からのデータ点のばらつきが少なければ、それは良いクラスタリングと言えそうである。(確率変数の分散に相当し、その分散を最小化する問題に相当している。) このことから、最適なクラスタリングを求める際の最適化問題は

$$\min_{\{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}} \sum_{i=1}^{3}\sum_{p \in \mathcal{V}_i} \|p - \mu_i\|^2 \tag{53}$$

と書ける。この最適化問題 (53) の解法として **K-平均法**が知られている。K-平均法では最初に代表点 (仮の重心) を適切に設定し、設定された重心についてクラスタを関係式

$$r_{i,l} = \begin{cases} 1 & (l = \arg\min_{i=1,2,3} \|p_l - \mu_i\|^2) \\ 0 & (otherwise) \end{cases} \tag{54}$$

によって定める。ただし、$r_{i,l} = 1$ はデータ点 p_l がクラスター $\mathcal{V}_i$ に属すことを示し、$r_{i,l} = 0$ はデータ点 p_l がクラスター $\mathcal{V}_i$ に属さないことを示している。任意に与えられた重心については、このクラスターの定め方が目的関数の最小値を与える。($\arg\min$ で添字を選んでいるため。この選び方は貪欲法的である。) いま、任意に与えられたクラスターについては、クラスターの重心 (52) が目的関数の最小値を与えることが知られている。従って、代表点 μ_i を重心に変更する。この操作を繰り返すと目的関数

の値が狭義単調に減少するので、この手続きを繰り返すことで最適解への漸近が期待される。3 個の代表点の座標の変化の最大値が ε よりも小さくなったところで反復を終了する。以上の手続きが K-平均法である。

K-平均法を用いることで局所的最適解への漸近は期待されるが、大域的最適解を求められるとは限らない。そこで、データ点の存在範囲の中で一様乱数を用いて初期点を 3 個ランダムに生成し、その初期点に対して K-平均法を用いることでクラスタリングのを行う。この操作を 30 回繰り返すことで大域的最適解を推測する。また、終了判定の条件 ε を $\varepsilon = 0.01$ とする。

数値計算の結果

数値計算の結果は図 4, 図 5, 図 6 のようになった。ただし、各クラスタリングの上部にデータ点とそのデータ点の代表点の間の距離の 2 乗の平均値

$$\sum_{i=1}^{3} \sum_{p \in \mathcal{V}_i} \|p - \mu_i\|^2 \tag{55}$$

を表示してプロットした。

結果を踏まえて最も良さそうなクラスタリングについて議論する。まず、定量的な議論を行う。目的関数は距離の 2 乗の総和であったので、クラスタの上部に表示されている値が最も小さいようなクラスタリングは良いクラスタリングであると、短絡的ではあるが結論できる。図 4, 5, 6 を見ると目的関数の最小値の候補として (1 の位までの相違を無視して)

- 3510
- 4010

が存在することが分かる。これらの中で 3510 という値を実現しているクラスターが分散を最も小さくしているという意味で最も優れていると結論できる。以上の定量的な議論によって最も良いと結論されたクラスターについて定性的な議論を行う。凡例として目的関数値 3510 を達成しているクラスタリングを図 7 に、目的関数値 4010 を実現しているクラスタリングを図 8 に表示する。図 7 においてはクラスタの代表点 μ_1, μ_2, μ_3 の周辺にデータ点が密集している。代表点が点の密集している点にあるということは、「クラスタが代表点からランダムな誤差によって生じている」というような仮説が立てられるという面で有用であるといえる。一方で図 8 においてはクラスタの代表点 μ_2 がデータ点の密集している部分の周縁部に位置している。このようなクラスタリングでは「クラスタが代表点からランダムな誤差によって生じている」という仮説を補強できず、クラスタリングとしては図 7 のようなクラスタリングよりも劣っているといえる。

以上の議論から、最も良いクラスタリングは 7 のようなクラスタリングであると結論できる。(図 4, 5, 6 を見ると図 7 のようなクラスタリングに限らず、目的関数値 3510 を達成しているクラスタリングは定量的にも定性的にも良いクラスタリングであると考えられる。)

ソースコード

レポートの最後に、各問題の数値計算に用いたソースコードを示す。ソースコードは次のように分かれている。

- 1.jl : 課題 1 に用いた julia ソースコード
- 8.jl : 課題 8 に用いた julia ソースコード
- 11.jl : 課題 11 に用いた julia ソースコード
- 14.jl : 課題 14 に用いた julia ソースコード

表示の都合上、ページをまたいでソースコードを示している部分がある。その際にはタイトルを

```
1.jl(1), 1.jl(2) ...
```

のように記した。また、表示がはみ出てしまった部分についてソースコードの行数を ”82.5 : ... ” のようにしている場所があるが、小数点以下を切り捨てて同じ行を表している。

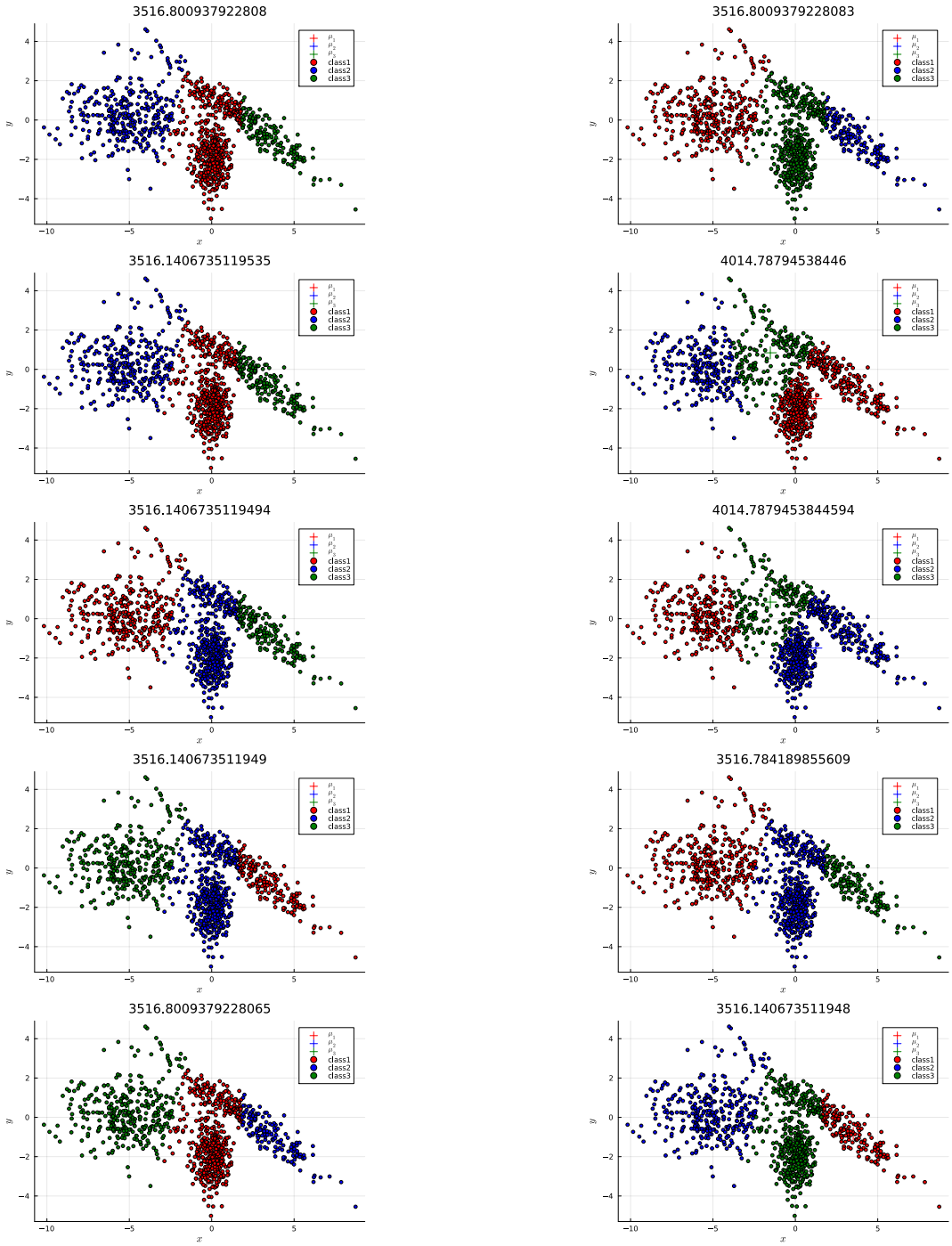

図 4: 1 枚目から 10 枚目までのクラスタリング結果 (各クラスタリングの上部に最適化問題の目的関数の値を記してある)(μ_i は class i の代表点を表す)

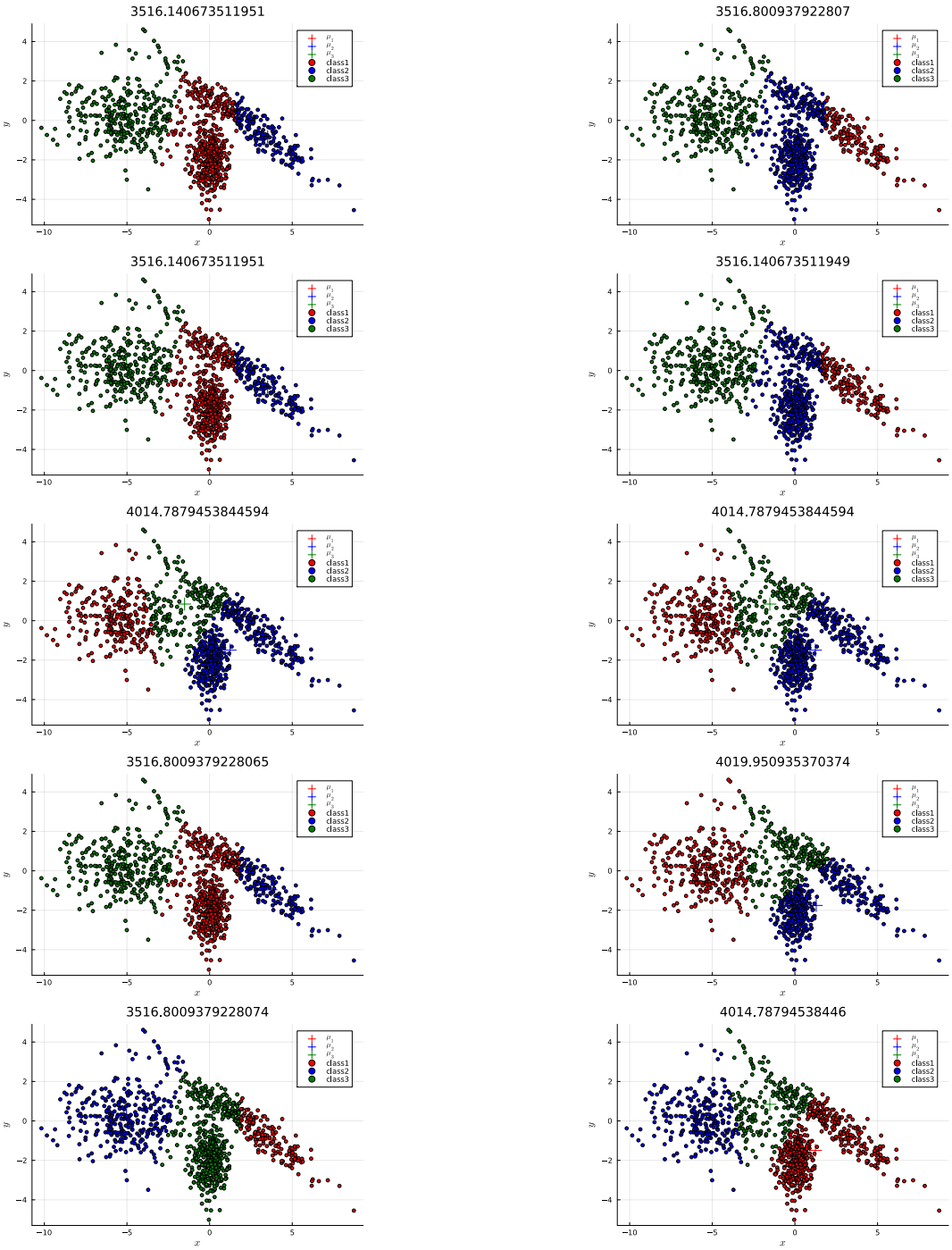

図 5: 11 枚目から 20 枚目までのクラスタリング結果 (各クラスタリングの上部に最適化問題の目的関数の値を記してある)(μ_i は class i の代表点を表す)

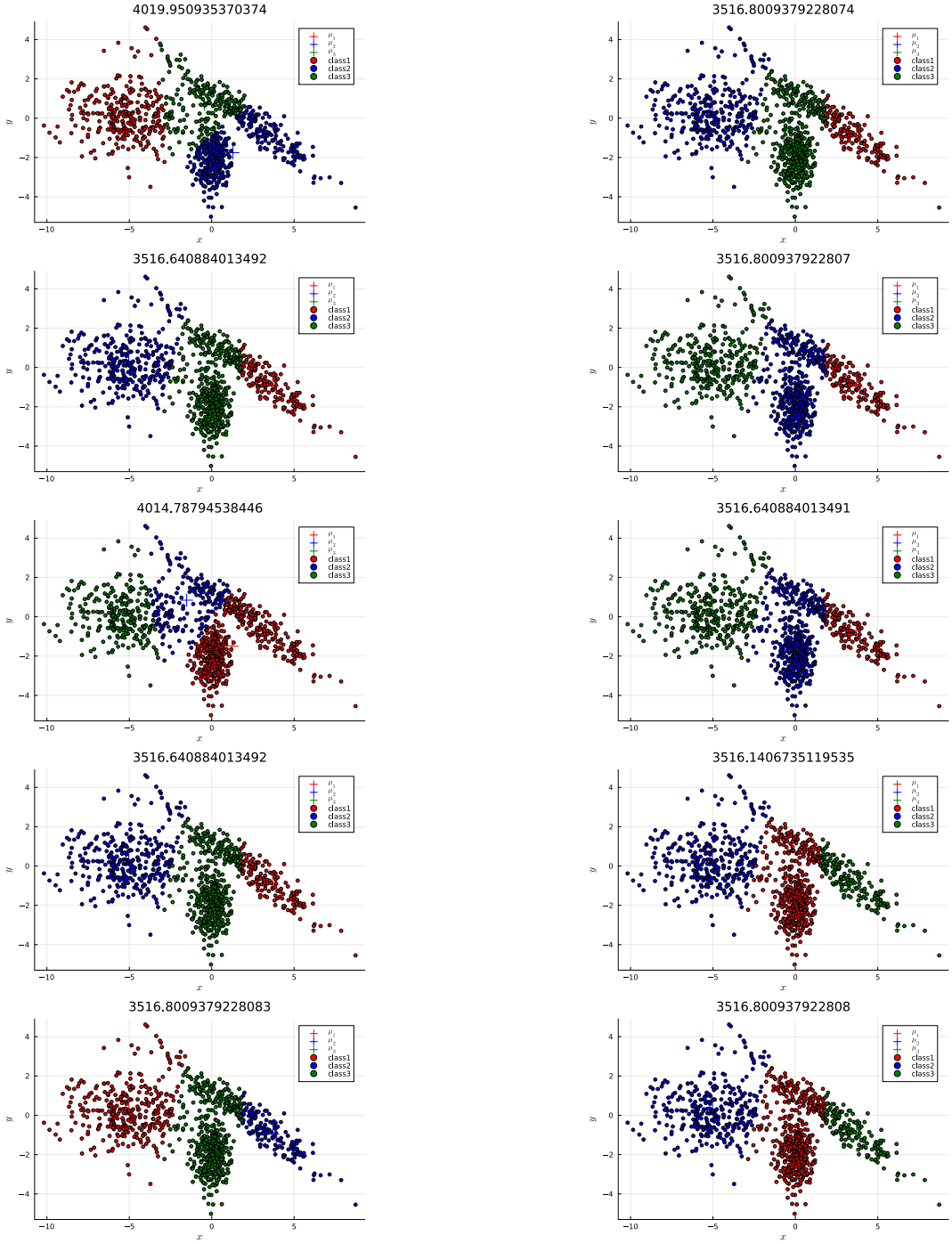

図 6: 21 枚目から 30 枚目までのクラスタリング結果 (各クラスタリングの上部に最適化問題の目的関数の値を記してある)(μ_i は class i の代表点を表す)

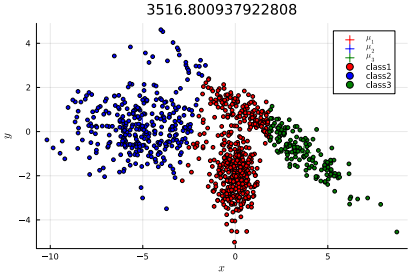

図 7: 目的関数値 3510 を実現しているクラスタの例 (1 枚目の図と同一)

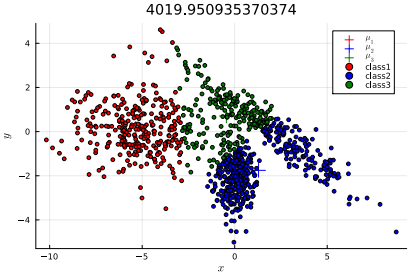

図 8: 目的関数値 4010 を実現しているクラスタの例 (18 枚目の図と同一)

```
using Plots, StatsPlots
using LaTeXStrings
using LinearAlgebra

# N 個、',' 区切り形式のファイルから点列を多次元配列で取り出す
function loader(file_name, N)
    arr = zeros(Float64, N, 3)

    f = open(file_name, "r")
    list = readlines(f)
    close(f)

    for i in 1:N
        s = list[i]
        point = split(s, ',')
        nums = [parse(Float64, n) for n in point]
        arr[i, 1] = nums[1]
        arr[i, 2] = nums[2]
        arr[i, 3] = nums[3]
    end

    return arr
end

# 逆行列を計算
function my_inv(A)
    return A \ Matrix(I, size(A))
end

# 最小二乗誤差推定量を出力する関数
function linreg(N)
    data = loader("mmse_kadai1.txt", N)

    A = zeros(Float64, 2, 2)
    b = zeros(Float64, 2, 1)

    for i in 1:N
        A += data[i, 1:2] * transpose(data[i, 1:2])
        b += data[i, 1:2] * data[i, 3]
    end

    return A \ b
end

# 観測誤差の共分散行列の推定値を出力する関数
function Vw(N)
    data = loader("mmse_kadai1.txt", N)

    # 線形回帰
    θ = linreg(N)

    # 観測誤差の共分散行列
    V = zeros(Float64, 1, 1)

    for i in 1:N
        V[1, 1] += (data[i, 3] - ([data[i, 1] data[i, 2];] * θ)[1, 1])^2
    end

    return V ./ (N - 2)
end

# 推定誤差共分散行列を出力する関数
function Vt(N)
    data = loader("mmse_kadai1.txt", N)

    # 線形回帰
    θ = linreg(N)

    # 推定誤差共分散行列
    Φ = zeros(Float64, 2, 2)
    buf = zeros(Float64, 2, 2)

    # Φの計算
    for i in 1 : N
        buf += transpose([data[i, 1] data[i, 2];]) *[data[i, 1] data[i, 2];]
    end

    Φ = my_inv(buf)

    # 今回は誤差が N(0, 1) なのでΦが求める行列になる
    return Φ
end

```

1.jl(2)

```
 84 : # 推定誤差共分散行列を出力する関数
 85 : function Vt(N, Vw)
 86 :     data = loader("mmse_kadai1.txt", N)
 87 :
 88 :     # 推定誤差共分散行列
 89 :     Φ = zeros(Float64, 2, 2)
 90 :     buf = zeros(Float64, 2, 2)
 91 :
 92 :     # Φの計算
 93 :     for i in 1 : N
 94 :         buf += transpose([data[i, 1] data[i, 2];]) *[data[i, 1] data[i, 2];]
 95 :     end
 96 :
 97 :     Φ = my_inv(buf)
 98 :
 99 :     # 今回は共分散行列がスカラーなので V Φが求める行列になる
100 :     return Vw .* Φ
101 : end
102 :
103 : # 決定変数を出力する関数
104 : function dec(N)
105 :     data = loader("mmse_kadai1.txt", N)
106 :
107 :     a = 0
108 :     b = 0
109 :
110 :     θ = linreg(N)
111 :     y ¯ = sum(data[1:N, 3]) / N
112 :
113 :     for i in 1:N
114 :         a += (([data[i, 1] data[i, 2];] * θ )[1] - y ¯)^2
115 :         b += (data[i, 3] - y ¯)^2
116 :     end
117 :
118 :     return a / b
119 : end
120 :
121 : # プロット関数
122 : function plotter(d, s)
123 :     θ = zeros(Float64, 13, 2)
124 :
125 :     n = 2
126 :     θ [1, 1:2] = transpose(linreg(n))
127 :
128 :     for i in 1:12
129 :         n *= 2
130 :         θ [i + 1, 1:2] = transpose(linreg(n))
131 :     end
132 :
133 :     plt = plot([i * 1.0 for i in 1:13], θ [1:13, d], markershape = :circle, label = s)
134 :
135 :     return plt
136 : end
137 :
138 : # (2) 仕様のプロットを得る関数
139 : function plot_2()
140 :     plt1 = plotter(1, L"\hat{\theta}_{1, N}")
141 :     plt2 = plotter(2, L"\hat{\theta}_{2, N}")
142 :
143 :     # 軸の調整
144 :     y1 = [2.0 * i for i in 1:7]
145 :     push!(y1, 1.5)
146 :     s1 = [string(i) for i in y1]
147 :
148 :     x = [i for i in 1:13]
149 :
150 :     # plt1 の調整
151 :     xticks!(plt1, x, [string(i) for i in x])
152 :     yticks!(plt1, y1, s1)
153 :     xlabel!(plt1, L"\log_{2} N")
154 :     ylabel!(plt1, L"\hat{\theta}_{1, N}")
155 :
156 :     # plt2 の調整
157 :     xticks!(plt2, x, [string(i) for i in x])
158 :     xlabel!(plt2, L"\log_{2} N")
159 :     ylabel!(plt2, L"\hat{\theta}_{2, N}")
160 :
161 :     savefig(plt1, "1-2-1.pdf")
162 :     savefig(plt2, "1-2-2.pdf")
163 : end
```

8.jl(1)

```
 1 : using Plots, StatsPlots
 2 : using LaTeXStrings
 3 : using LinearAlgebra
 4 :
 5 : # N 個、',' 区切り形式のファイルから指定された範囲の点列を多次元配列で取り出す
 6 : function loader(file_name, range, N)
 7 :     arr = zeros(Float64, N, 2)
 8 :
 9 :     f = open(file_name, "r")
10 :     list = readlines(f)
11 :     close(f)
12 :
13 :     for i in 1:N
14 :         s = list[i]
15 :         point = split(s, ',')
16 :         nums = [parse(Float64, n) for n in point]
17 :         arr[i, 1] = nums[1]
18 :         arr[i, 2] = nums[2]
19 :     end
20 :
21 :     return arr[range, 1:2]
22 : end
23 :
24 : # 逆行列を計算
25 : function my_inv(A)
26 :     return A \ Matrix(I, size(A))
27 : end
28 :
29 : # 計算結果を構造体として保持
30 : struct reg
31 :     θ
32 :     Φ
33 :     V
34 :     N
35 : end
36 :
37 : # 関数φ
38 : function phi(x)
39 :     buf = zeros(Float64, 1, 3)
40 :     buf[1, 1] = 1.0
41 :     buf[1, 2] = exp(- (x - 1)^2 / 2.0)
42 :     buf[1, 3] = exp(-(x + 1)^2)
43 :     return buf
44 : end
45 :
46 : # 最小二乗誤差推定量を出力する関数
47 : function linreg(N, range)
48 :     data = loader("mmse_kadai8.txt", range, N)
49 :
50 :     A = zeros(Float64, 3, 3)
51 :     b = zeros(Float64, 3, 1)
52 :
53 :     # パラメーターの推定値
54 :     for i in 1:length(range)
55 :         A = A + transpose(phi(data[i, 1])) * phi(data[i, 1])
56 :         b = b + transpose(phi(data[i, 1])) * data[i, 2]
57 :     end
58 :
59 :     # 共分散の推定値
60 :     buf = zeros(Float64, 1)
61 :
62 :     for i in 1:length(range)
63 :         buf += ([data[i, 2]] - (phi(data[i, 1]) * (A \ b)))^2
64 :     end
65 :
66 :     # 返り値は構造体
67 :     return reg(A \ b, my_inv(A), buf / (length(range) - 3) ,length(range))
68 : end
69 :
70 : # 重みつき合成を行う関数
71 : function integrate(range1, range2)
72 :     reg1 = linreg(10000, range1)
73 :     reg2 = linreg(10000, range2)
74 :
75 :     X = zeros(Float64, 3, 3)
76 :     Y = zeros(Float64, 3, 1)
77 :
78 :     X = my_inv((reg1.V)^(-1) .* my_inv(reg1. Φ) + (reg2.V)^(-1) .* my_inv(reg2. Φ))
79 :     # θを素朴な最小2乗法で求めたものと同一としているが、重み一様かつ出力誤差の共分散が1次元なので問題ない
80 :     Y = ((reg1.V)^(-1) .* my_inv(reg1. Φ)) * reg1. θ + ((reg2.V)^(-1) .* my_inv(reg2. Φ)) * reg2. θ
81 :
82 :     return X * Y
83 : end
84 :
```

8.jl(2)

```
 85 : # 決定変数を出力する関数
 86 : function dec(θ)
 87 :     data = loader("mmse_kadai8.txt", 1:10000, 10000)
 88 :
 89 :     a = 0
 90 :     b = 0
 91 :
 92 :     y ¯ = sum(data[1:10000, 2]) / 10000
 93 :
 94 :     for i in 1:10000
 95 :         a += ((phi(data[i, 1]) * θ )[1] - y ¯)^2
 96 :         b += (data[i, 2] - y ¯)^2
 97 :     end
 98 :
 99 :     return a / b
100 : end
101 :
102 : # 真のパラメーターはθ = [0.1, 3.1, -2.1]
103 : # 比較
104 : function compare()
105 :     θ = [0.1;3.1;-2.1]
106 :     println("1:10000: ", norm(linreg(10000, 1:10000). θ - θ, 2))
107 :     println("1:6000 + 6001:10000: ", norm(integrate(1:6000, 6001:10000) - θ, 2))
108 : end
```

11.jl

```
using Plots
using StatsPlots
using LaTeXStrings

ENV["GKS_ENCODING"] = "utf8"; # ラベルで日本語を使うときに必要

gr() # GR バックエンドを使うことを宣言

function f1(x)
    return 0.9 * x + randn()
end

function f2(θ)
    return 2.0 * θ + randn()
end

struct data
    θ s
    ys
end

#  データ生成
function generand()
    # 平均 3 分散 2 の乱数で初期値を y0 を生成
    y0 = 3 + sqrt(2.0) * randn()
    θ 0 = 2 * y0 + randn()

    ys = zeros(Float64, 1, 101)
    θ s = zeros(Float64, 1, 101)

    ys[1, 1] = y0
    θ s[1, 1] = θ 0

    for i in 1:100
        θ 0 = f1(θ 0)
        θ s[i + 1] = θ 0
        y0 = f2(θ 0)
        ys[i + 1] = y0
    end

    return data(θ s, ys)
end
# Kalman-Fielter
function kalman_fielter(ys, N)
    θ 0 = 3
    V0 = 2

    Xi = zeros(Float64, 1, N)
    θ i = zeros(Float64, 1, N)
    Vi = zeros(Float64, 1, N)
    Fi = zeros(Float64, 1, N)

    θ i[1, 1] = θ 0
    Vi[1, 1] = V0

    for i in 2:N
        Xi[1, i] = 0.81 * Vi[1, i - 1] + 1

        Vi[1, i] = (Xi[1, i]) / (4.0 * Xi[1, i] + 1.0)

        Fi[1, i] = (2.0 * Xi[1, i]) / (4.0 * Xi[1, i] + 1.0)

        θ i[1, i] = 0.9 * θ i[1, i - 1] + Fi[1, i] * (ys[i] - 1.8 * θ i[1, i - 1])
    end

    return θ i
end

# plot 関数
function plot_traction()
    data = generand()

    θ s = kalman_fielter(data.ys, 101)

    r = [i for i in 1:101]

    p = plot(r .- 1, data.θ s[1:101], label = L"θ_{i}")
    plot!(p, r .- 1, θ s[1:101], label = L"\hat{\theta}_{i}")

    # 軸ラベルの調整
    xlabel!(p, L"i")
    ylabel!(p, L"\theta")

    savefig(p, "11.pdf")
end
```

14.jl(1)

```
 1 : using Plots, StatsPlots
 2 : using LaTeXStrings
 3 : using LinearAlgebra
 4 :
 5 : # 1000 個のデータ点の読み込み
 6 : function loader(file_name, range)
 7 :     N = length(range)
 8 :     arr = zeros(Float64, N, 2)
 9 :
10 :     f = open(file_name, "r")
11 :     list = readlines(f)
12 :     close(f)
13 :
14 :     for i in 1:N
15 :         s = list[i]
16 :         point = split(s, ',')
17 :         nums = [parse(Float64, n) for n in point]
18 :         arr[i, 1] = nums[1]
19 :         arr[i, 2] = nums[2]
20 :     end
21 :
22 :     return arr[range, 1:2]
23 : end
24 :
25 : # 初期点を与える関数
26 : # 一様乱数で生成
27 : function init_point(arr, N)
28 :     res = zeros(Float64, 3, 2)
29 :
30 :     # 初期点はデータ点の存在範囲から生成
31 :     # データ点の存在範囲を計算
32 :     xl = arr[argmin(arr[:, 1]), 1]
33 :     xh = arr[argmax(arr[:, 1]), 1]
34 :     yl = arr[argmin(arr[:, 2]), 2]
35 :     yh = arr[argmax(arr[:, 2]), 2]
36 :
37 :     # データ点の存在範囲から一様乱数で生成
38 :     for i in 1:3
39 :         res[i, 1] = (xh - xl) * rand() + xl
40 :         res[i, 2] = (yh - yl) * rand() + yl
41 :     end
42 :
43 :     return res
44 : end
45 :
46 : # クラスター関数
47 : function classter(init_point, ε)
48 :     # 1000 個のデータ点の読み込み
49 :     data = loader("mmse_kadai14.txt", 1:1000)
50 :
51 :     # 終了条件
52 :     dec = [10000.0 10000.0 10000.0;]
53 :     decn = 10000
54 :
```

14.jl(2)

```
 55 :     # クラス分け行列
 56 :     r = zeros(Int, 1000, 3)
 57 :
 58 :     while (decn >= ε)
 59 :         # クラス分け行列の初期化
 60 :         r = zeros(Int, 1000, 3)
 61 :
 62 :         # クラス分けを実行
 63 :         for i in 1:1000
 64 :             l = zeros(1, 3)
 65 :             for j in 1:3
 66 :                 l[1, j] = norm(init_point[j,1:2] - data[i, 1:2], 2)
 67 :             end
 68 :             r[i, argmin(l[1, :])] = 1
 69 :         end
 70 :
 71 :         # 初期点の更新
 72 :         new_init_point = zeros(Float64, 3, 2)
 73 :
 74 :         for l in 1:3
 75 :             tmp = 0
 76 :             x = zeros(Float64, 1, 2)
 77 :
 78 :             for i in 1:1000
 79 :                 tmp += r[i, l]
 80 :                 x += [r[i, l] * data[i, 1] r[i, l] * data[i, 2];]
 81 :             end
 82 :
 83 :             if (tmp != 0)
 84 :                 new_init_point[l, 1:2] = x ./ tmp
 85 :             end
 86 :
 87 :             dec[1, l] = norm(init_point[l, 1:2] - new_init_point[l, 1:2], 2)
 88 :         end
 89 :
 90 :         decn = dec[1, argmax(dec[1, :])]
 91 :
 92 :         init_point = new_init_point
 93 :     end
 94 :
 95 :
 96 :     # 散布図の作成
 97 :     plt = scatter()
 98 :
 99 :     # クラス 1, 2, 3 の個体数
100 :     n1 = 0
101 :     n2 = 0
102 :     n3 = 0
103 :
104 :     # クラスの個体数の数え上げ
105 :     for i in 1:1000
106 :         if (argmax(r[i, :]) == 1)
107 :             n1 += 1;
108 :         end
109 :         if (argmax(r[i, :]) == 2)
110 :             n2 += 1;
111 :         end
112 :         if (argmax(r[i, :]) == 3)
113 :             n3 += 1;
114 :         end
115 :     end
116 :
117 :     # 集団ごとに配列を作る
118 :     c1 = zeros(2, n1)
119 :     c2 = zeros(2, n2)
120 :     c3 = zeros(2, n3)
121 :
122 :     n1 = 1
123 :     n2 = 1
124 :     n3 = 1
125 :
126 :     # クラス分け
127 :     for i in 1:1000
128 :         if (argmax(r[i, :]) == 1)
129 :             c1[1, n1] = data[i, 1]
130 :             c1[2, n1] = data[i, 2]
131 :             n1 += 1;
132 :         end
133 :         if (argmax(r[i, :]) == 2)
134 :             c2[1, n2] = data[i, 1]
135 :             c2[2, n2] = data[i, 2]
136 :             n2 += 1;
137 :         end
138 :         if (argmax(r[i, :]) == 3)
139 :             c3[1, n3] = data[i, 1]
140 :             c3[2, n3] = data[i, 2]
141 :             n3 += 1;
142 :         end
143 :     end
144 :
```

14.jl(3)

```
145 :     # 目的関数値の計算
146 :     dis = 0.0
147 :
148 :     for i in 1:length(c1[1,:])
149 :         dis += (norm(c1[1:2, i] - init_point[1, 1:2], 2))^2
150 :     end
151 :
152 :     for i in 1:length(c2[1,:])
153 :         dis += (norm(c2[1:2, i] - init_point[2, 1:2], 2))^2
154 :     end
155 :
156 :     for i in 1:length(c3[1,:])
157 :         dis += (norm(c3[1:2, i] - init_point[3, 1:2], 2))^2
158 :     end
159 :
160 :     # (1, 2, 3) = (red, blue, green)
161 :     scatter!(plt, (init_point[1, 1], init_point[1, 2]), markercolor = :red, marker = 10,
161.5 :  markershape = :+, label = L"\mu_{1}")
162 :     scatter!(plt, (init_point[2, 1], init_point[2, 2]), markercolor = :blue, marker = 10,
162.5 :  markershape = :+, label = L"\mu_{2}")
163 :     scatter!(plt, (init_point[3, 1], init_point[3, 2]), markercolor = :green, marker = 10,
163.5 : markershape = :+, label = L"\mu_{3}")
164 :     title!(plt, string(dis))
165 :
166 :     # 代表点の位置もプロット
167 :     scatter!(plt, c1[1, :], c1[2, :], markercolor = :red, markersize = 3, label = "class1")
168 :     scatter!(plt, c2[1, :], c2[2, :], markercolor = :blue, markersize = 3, label = "class2")
169 :     scatter!(plt, c3[1, :], c3[2, :], markercolor = :green, markersize = 3, label = "class3")
170 :
171 :     xlabel!(plt, L"$x$")
172 :     ylabel!(plt, L"$y$")
173 :
174 :     return plt
175 : end
176 :
177 : # プロット関数
178 : function plottings(range)
179 :     for i in range
180 :         filename = string(i) * ".pdf"
181 :
182 :         savefig(classter(init_point(loader("mmse_kadai14.txt", 1:1000), 1000), 0.01), filename)
183 :     end
184 : end
```