モデルパラメータの推定

上原 悠槙

統計数理研究所

2018/12/1
YUIMA **チュートリアル**
「確率微分方程式のデータサイエンス入門」

概要

尤度関数

定義

確率変数 $X_1, X_2, \ldots, X_n$ がパラメータ $\theta \in \Theta$ により特徴付けられた同時密度関数 $p(X_1, X_2, \ldots, X_n, \theta)$ を持つとする. 実現値
$X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n$ が得られた時, パラメータ θ の関数

$$L_n(\theta) = p(x_1, x_2, \ldots, x_n, \theta)$$

を尤度関数と呼ぶ.

- パラメータ θ ごとに現在起きている事象の確率を与える関数である (すなわち密度関数ではない).
- $X_1, X_2, \ldots, X_n$ が i.i.d の時,

$$L_n(\theta) = p(x_1, \theta)p(x_2, \theta)\ldots p(x_n, \theta).$$

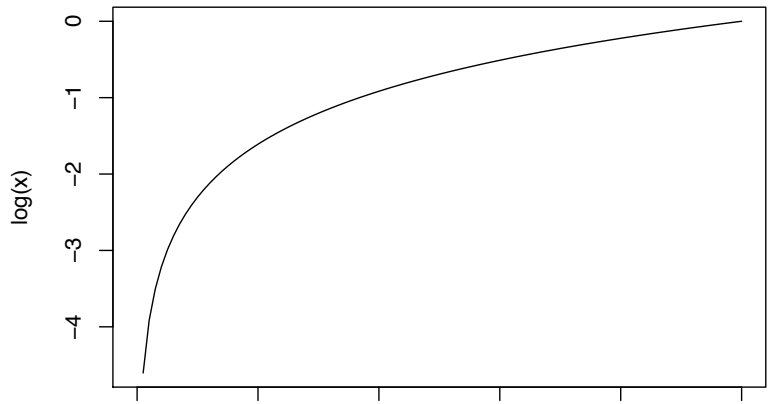
- 簡単のため, 以下では $X_1, X_2, \ldots, X_n$ が i.i.d としてしばらく解説を行う.

最尤推定法

- $L_n(\theta_1) > L_n(\theta_2) \Rightarrow \theta_1$ の方が θ_2 より実現値を得られやすいと解釈できる (θ_1 の方が θ_2 より尤もらしい).
- **最尤推定法**とは尤度関数 $L_n(\theta)$ を最大化する $\hat{\theta}_n := \hat{\theta}(x_1, \ldots, x_n) \in \Theta$ をパラメータ θ の推定値とする手法である. この推定量 $\hat{\theta}_n$ を特に**最尤推定量**と呼ぶ.
- 対数関数の単調性 (下図) により, 最尤推定量 $\hat{\theta}_n$ は**対数尤度関数**

$$\log L_n(\theta) = \sum_{j=1}^{n} \log p(x_j, \theta)$$

を最大化する $\tilde{\theta}_n \in \Theta$ と等しい. 積が和に変わるため, 計算が容易になることが多い.



対数尤度関数の収束について

- 大数の法則により, 確率 1 で収束

$$\frac{1}{n}\left(\log L_n(\theta_0) - \log L_n(\theta)\right) \to \int \log\left[\frac{p(x,\theta_0)}{p(x,\theta)}\right] p(x,\theta_0)dx$$

  が成り立つ.

- 右辺は, Kullback-Leibler **情報量**と呼ばれる分布間の近さを測る尺度であり, これを $\mathrm{KL}(p(x,\theta_0); p(x,\theta))$ と書く. この時, 不等式

$$\mathrm{KL}(p(x,\theta_0); p(x,\theta)) \geq 0$$

  が成り立つことが知られており, その等号成立条件はほとんどいたるところ $p(x,\theta_0) = p(x,\theta)$ である.

- 識別性条件 「$\theta_0 \neq \theta \Rightarrow p(x,\theta_0) \neq p(x,\theta)$」を仮定すると上の条件から, 真値 θ_0 が $\mathrm{KL}(p(x,\theta_0); p(x,\theta))$ を最小化する唯一の点であることがわかる.

- 最尤推定量 $\hat{\theta}_n$ が, $\frac{1}{n}\left(\log L_n(\theta_0) - \log L_n(\theta)\right)$ を最小にする点であるため, $\hat{\theta}_n$ は θ_0 に収束すると予想できる.

最尤推定量の収束について

- 適切な条件の下で $\hat{\theta}_n$ は θ_0 へ確率収束する (最尤推定量の一致性と呼ばれる).
- 加えて, 誤差ベクトルをスケーリングした $\sqrt{n}(\hat{\theta}_n - \theta_0)$ の漸近正規性, すなわち,

$$\sqrt{n}(\hat{\theta}_n - \theta_0) \xrightarrow{\mathcal{L}} N(0, I(\theta_0)^{-1}),$$

も知られている. ここで $I(\theta_0) \in \mathbb{R}^p \otimes \mathbb{R}^p$ はフィッシャー情報行列

$$I(\theta_0) = \int (\partial_\theta \log p)^\top (\partial_\theta \log p)(x, \theta_0) p(x, \theta_0) dx$$

である.

- $\xrightarrow{\mathcal{L}}$ は分布収束を意味する. d-次元確率変数列 (X_n) と X について, (X_n) が X に分布収束するとは, $P(X_n \leq x) \to F(x) := P(X \leq x)$ が $F(x)$ の任意の連続点 $x \in \mathbb{R}^d$ について成り立つことを意味する.

$X_1, X_2, \ldots, X_n$ が i.i.d でない場合

- ここまで, $X_1, X_2, \ldots, X_n$ が i.i.d として議論を行ったが, 確率微分方程式からの離散観測を想定するとき, その仮定は不適合である.
- しかし, $X_1, X_2, \ldots, X_n$ が Markov 性, すなわち任意の $j \in \{1, \ldots, n-1\}$ に対して, 条件付き確率に関する関係式

$$P(X_{j+1} = x_{j+1} | X_1 = x_1, X_2 = x_2, \ldots, X_j = x_j) = P(X_{j+1} = x_{j+1} | X_j = x_j)$$

を満たす時, 対数尤度関数 $\log L_n(\theta)$ は推移確率 $p(x_j | x_{j-1}, \theta)$ を用いて

$$L_n(\theta) = p(x_1, \theta) p(x_2 | x_1, \theta) p(x_3 | x_2, \theta) \ldots p(x_n | x_{n-1}, \theta)$$

と書くことができる.

- 上式により, 大数の法則などが成り立つ (エルゴード性の) 下で, 先と同様の議論が展開できる.

ベイズ推定法

- 確率変数 $X_1, X_2, \ldots, X_n$ からの 実現値 $X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n$ が得られているとする.
- 最尤推定法では, パラメータ θ は非確率的な量として捉えていたが, ベイズ推定法ではパラメータをある分布に従う確率変数とみなす.
- これにより, 解析に先立ってパラメータの事前知識を推定に取り入れることができ, それに基づくパラメータの事前分布を π と書く (パラメータ空間 Θ 上の分布).
- また, 実現値 $X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n$ に対する尤度関数をここでは, $p(x_1, x_2, \ldots, x_n|\theta)$ と書く.

事後分布

- ベイズ推定においては, θ の事後分布 $p(\theta|x_1, x_2, \ldots, x_n)$ が重要な役割を果たす. その定義から, 事後分布は観測が与えられた際の θ の分布であることがわかる.
- ベイズの定理を用いることで事後分布は,

$$p(\theta|x_1, x_2, \ldots, x_n)$$
$$= \left[\int_\Theta p(x_1, x_2, \ldots, x_n|\theta)\pi(\theta)d\theta\right]^{-1} p(x_1, x_2, \ldots, x_n|\theta)\pi(\theta), \quad (1)$$

と書き表すことができ, 右辺は観測と事前分布のみで表現されている.

事後平均

- この事後分布による平均 (事後平均), すなわち

$$\int_{\Theta} \theta p(\theta|x_1, x_2, \ldots, x_n) d\theta$$

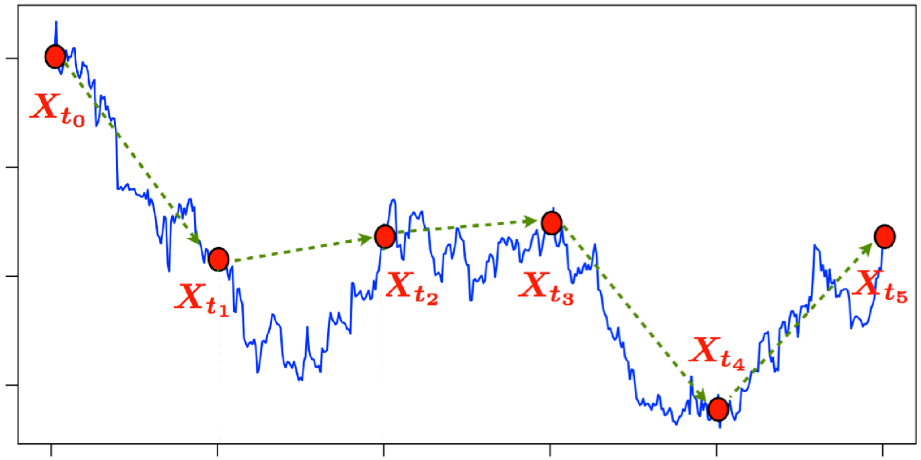
$$= \left[\int_{\Theta} p(x_1, x_2, \ldots, x_n|\theta)\pi(\theta) d\theta \right]^{-1} \int_{\Theta} \theta p(x_1, x_2, \ldots, x_n|\theta)\pi(\theta) d\theta,$$

  がしばしばパラメータ θ の推定値として用いられる.

- 事後平均も最尤推定量と同様に, 適切な条件の下で一致性や漸近正規性を満たすことが知られている

問題設定

$$dX(t) = a(X(t), \alpha)dt + b(X(t), \beta)dB(t)$$



- $(B_t)_{t\geq 0}$: ブラウン運動
- 推定対象:
  $\theta := (\alpha, \beta) \in \Theta_\alpha \times \Theta_\beta =: \Theta$.
- 観測データ:
  $\mathbb{X}_n := (X(t_1), \ldots, X(t_n))$.
- $t_j := j\Delta_n, n\Delta_n \to \infty, n\Delta_n^2 \to 0$.

- ドリフト係数, 拡散係数が適当な正則条件を満たしていれば確率微分方程式の解が一意に存在し, Markov 性やエルゴード性を持つことが知られている.

目的

未知パラメータ θ を高頻度離散観測データ $\mathbb{X}_n$ を用いて推定する.

エルゴード性について

- 解過程 $X(t)$ がエルゴード性を持つとは, どんな初期値から出発しても時間の経過とともに $X(t)$ の分布が不変分布 ν に近づくことであった.
- このとき, 以下が適当な関数 f について成り立つ

$$\frac{1}{n}\sum_{j=1}^{n} f(X(t_{j-1})) \to \int f(x)\nu(dx).$$

拡散過程の尤度関数

$$dX(t) = a(X(t), \alpha)dt + b(X(t), \beta)dB(t)$$

- Markov 性から, 観測 $\mathbb{X}_n$ に対する対数尤度関数
  $\log L_n(\theta) = \log p(X(t_0), X(t_1), X(t_2), \ldots, X(t_n), \theta)$ は

$$\log L_n(\theta) = \log p(X(t_0), \theta) + \sum_{j=1}^{n} \log p(X(t_j)|X(t_{j-1}), \theta)$$

  と書き直すことができる.

- しかしながら, 遷移確率 $p(X(t_j)|X(t_{j-1}), \theta)$ は一般に陽な形が得られないことが知られているため, 尤度関数を直接推定などに用いることはできない.

拡散過程の離散近似

- 尤度関数の代替となる疑似尤度関数 を構成する.
- 解過程 $X(t)$ のシミュレーションで用いたオイラー・丸山法に基づき, 観測系列 $\mathbb{X}_n$ の離散近似

$$X(t_j) \approx X(t_{j-1}) + a(X(t_{j-1}), \alpha)\Delta_n + b(X(t_{j-1}), \beta)(B(t_j) - B(t_{j-1})),$$

を考える.

- ブラウン運動の定義から $B(t_j) - B(t_{j-1})$ は平均 0, 分散 Δ_n の正規分布に従うため, 平均 μ, 分散 Σ の正規分布の密度関数 $\phi(x; \mu, \Sigma)$ を用いて, 推移確率の近似

$$p(X(t_j)|X(t_{j-1}), \theta) \approx \phi(X(t_j); X(t_{j-1}) + a(X(t_{j-1}), \alpha)\Delta_n, S(X(t_{j-1}), \beta)\Delta_n)$$

を得ることができる.

疑似尤度関数, 疑似最尤推定量

- $\Delta_j X = X(t_j) - X(t_{j-1})$, $S(x,\beta) = b(x,\beta)b^\top(x,\beta)$ とする. 行列 A, B について, $A^{\otimes 2} = AA^\top$, $B[A] = \mathrm{trace}(BA^\top)$ と書く.
- 先ほどの遷移確率の近似を用いて, 観測 $\mathbb{X}_n$ の**対数疑似尤度関数** $\mathbb{H}(\theta)$ を以下で定義する.

対数疑似尤度関数

$$\begin{aligned}\mathbb{H}_n(\theta) &= \sum_{j=1}^n \log \phi(X(t_j); X(t_{j-1}) + a(X(t_{j-1}),\alpha)\Delta_n, S(X(t_{j-1}),\beta)\Delta_n) \\ &= \sum_{j=1}^n \left\{ -\frac{d}{2}\log(2\pi\Delta_n) - \frac{1}{2}\log|S(X(t_{j-1}),\beta)| \right. \\ &\qquad \left. -\frac{1}{2\Delta_n} S(X(t_{j-1}),\beta)^{-1}[(\Delta_j X - a(X(t_{j-1}),\alpha)\Delta_n)^{\otimes 2}] \right\}\end{aligned}$$

- 最尤推定法と同様に, $\mathbb{H}_n$ を最大化する $\hat{\theta}_n := \hat{\theta}(x_1, \ldots, x_n) \in \Theta$ をパラメータ θ の推定値とし, これを特に**疑似最尤推定量**と呼ぶ.

適応型疑似ベイズ推定量

先ほど構成した対数疑似尤度関数を用いて，以下の要領で適応型疑似ベイズ推定量を構成する．

1. まず，任意の初期値 $\alpha^\star \in \Theta_\alpha$ を固定し，β の疑似ベイズ推定量 $\bar{\beta}_n$ を

$$\bar{\beta}_n = \left[\int_{\Theta_\beta} \exp\left\{\mathbb{H}(\alpha^\star, \beta)\right\} \pi_1(\beta) d\beta\right]^{-1} \int_{\Theta_\beta} \beta \exp\left\{\mathbb{H}(\alpha^\star, \beta)\right\} \pi_1(\beta) d\beta,$$

   により定義する．ここで，π_1 は Θ_β 上の事前分布を表す．

2. 次に $\bar{\beta}_n$ を用いて，α の疑似ベイズ推定量 $\bar{\alpha}_n$ を

$$\bar{\alpha}_n = \left[\int_{\Theta_\alpha} \exp\left\{\mathbb{H}(\alpha, \bar{\beta}_n)\right\} \pi_2(\alpha) d\alpha\right]^{-1} \int_{\Theta_\alpha} \alpha \exp\left\{\mathbb{H}(\alpha, \bar{\beta}_n)\right\} \pi_2(\alpha) d\alpha,$$

   により定義する．ここで，π_2 は Θ_α 上の事前分布を表す．

推定量の漸近挙動

疑似最尤推定量 $\hat{\theta}_n$ と適応型疑似ベイズ推定量 $\bar{\theta}_n$ は, 解過程 X がエルゴード性を持つとき, 通常の最尤推定量やベイズ推定量と同様, 一致性および漸近正規性を満たすことが知られている. すなわち, $\hat{\theta}_n$ および $\bar{\theta}_n$ は真の値 $\theta_0 := (\alpha_0, \beta_0)$ へ確率収束し, さらに,

$$\begin{pmatrix} \sqrt{T_n} & O \\ O & \sqrt{n} \end{pmatrix} \begin{pmatrix} \hat{\alpha}_n - \alpha_0 \\ \hat{\beta}_n - \beta_0 \end{pmatrix} \xrightarrow{\mathcal{L}} N(0, I(\theta_0)^{-1}),$$

$$\begin{pmatrix} \sqrt{T_n} & O \\ O & \sqrt{n} \end{pmatrix} \begin{pmatrix} \bar{\alpha}_n - \alpha_0 \\ \bar{\beta}_n - \beta_0 \end{pmatrix} \xrightarrow{\mathcal{L}} N(0, I(\theta_0)^{-1}),$$

は成り立つ. ここで, 解過程 X の不変分布を ν とすると, 漸近分散 $I(\theta_0) = \mathrm{diag}[\Gamma_1(\theta_0), \Gamma_2(\theta_0)]$ は,

$$\Gamma_1(\theta_0)[u_1^{\otimes 2}] = \int_{\mathbb{R}^d} S(x, \beta_0)^{-1}[\partial_\alpha a(x, \alpha_0)[u_1], \partial_\alpha a(x, \alpha_0)[u_1]]\nu(dx)$$

$$\Gamma_2(\theta_0)[u_2^{\otimes 2}] = \frac{1}{2}\int_{\mathbb{R}^d} \mathrm{trace}\left\{S^{-1}(\partial_\beta S)S^{-1}(\partial_\beta S)(x, \beta_0)[u_1^{\otimes}]\right\}\nu(dx)$$

により定義される.

関数 “qmle"

YUIMA パッケージ上では, 拡散過程の疑似最尤推定法は, 関数 qmle を用いて実行できる. 関数 qmle は以下の書式で定義される.

```
qmle(yuima, start, lower, upper, joint = FALSE, rcpp =FALSE, ...)
```

ここで, 各引数はそれぞれ,

- yuima: 係数の情報やデータを包含した yuima オブジェクト.
- start: 推定量計算のための初期値. 構造は名前付きリスト.
- lower: それぞれのパラメータ空間の下限. 構造は名前付きリスト.
- upper: それぞれのパラメータ空間の上限. 構造は名前付きリスト.
- joint: 推定を段階的に行うか否か. TRUE または FALSE で入力し, FALSE の方が計算速度が速い (デフォルトは FALSE).
- rcpp: C++ のコードを使用するか否か. TRUE または FALSE で入力し, TRUE の方が計算速度が速い (デフォルトは FALSE).

関数 adaBayes

YUIMA パッケージ上では, 拡散過程の適応型疑似ベイズ推定は関数 adaBayes を用いて行うことができる. 関数 adaBayes は以下の書式で定義される.

```
adaBayes(yuima, start, prior, lower, upper, rcpp = TRUE,
         method = "mcmc", mcmc = 1000,...)
```

ここで, 各引数はそれぞれ,

- yuima: 係数の情報やデータを包含した yuima オブジェクト.
- start: 推定量計算のための初期値. 構造は名前付きリスト.
- prior: パラメータの事前分布. 構造は名前付きリスト.
- lower: それぞれのパラメータ空間の下限. 構造は名前付きリスト.
- upper: それぞれのパラメータ空間の上限. 構造は名前付きリスト.

関数 adaBayes (続き)

- `rcpp`: C++ のコードを使用するか否か. TRUE または FALSE で入力し, TRUE の方が計算速度が速い (デフォルトは FALSE).
- `method`: 積分計算で何を用いるか. `mcmc`: または `nomcmc` で入力し, デフォルトは `mcmc`. また, `nomcmc` を用いる際は `cubature` が必要 (`library(cubature)` でインストールできる).
- `mcmc`: `method` で `mcmc` を選んだ場合に指定する必要があり, MCMC の繰り返し回数を表す. デフォルトは 1000.

シミュレーション

一例として, 以下のオルンシュタイン・ウーレンベックモデルのパラメータ推定を考えてみよう.

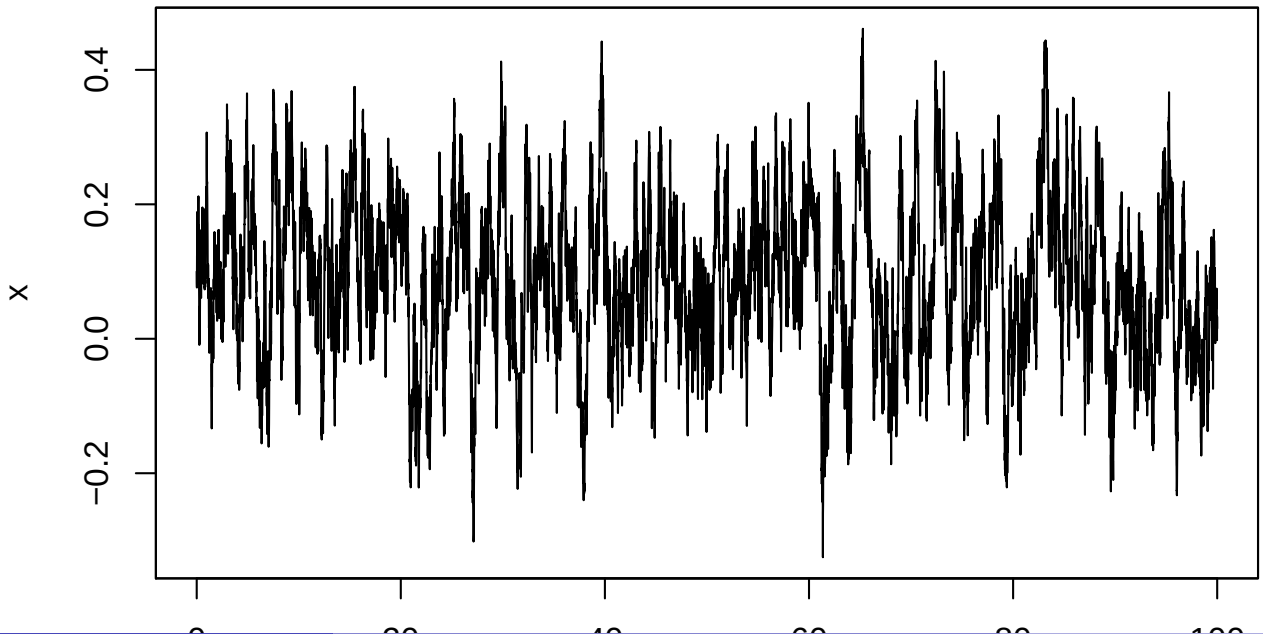
$$dX(t) = -\alpha_1(X(t) - \alpha_2)dt + \beta_1 dB(t),$$

ここで,

- パラメータの真値を $(\alpha_{1,0}, \alpha_{2,0}, \beta_{1,0}) = (3, 1, 0.3)$ する.
- サンプルサイズ, および観測時間幅を $n = 10000, \Delta_n = 5n^{-2/3}$ とする.
- パラメータ空間をそれぞれ, $\Theta_\alpha = [0.1, 5] \times [0.5, 2]$, $\Theta_\beta = [0.1, 2.5]$ とする.
- YUIMA パッケージ上では, モデルの設定やデータの生成は, 関数 `setModel` や `simulate` を用いて実行できる.

実行例

```
mod.ou <- setModel(drift="-alpha1*(x-alpha2)",diffusion="beta")
n <- 10000
Terminal <- 5*n^{1/3}
samp <- setSampling(Initial = 0, Terminal = Terminal, n = n)
ou <- setYuima(mod=mod.ou,sampling = samp)
set.seed(123)
x0 <- 1
param <- list(alpha1 = 3, alpha2 = 1, beta = 0.3)
obj <- simulate(ou,xinit=x0,true.parameter=param)
plot(obj)
```



関数 qmle および adaBayes の適用

```
### パラメータの設定
start <- list(alpha1 = 2, alpha2 = 1.3, beta = 1.6)
lower <- list(alpha1 = 0.1, alpha2 = 0.5, beta = 0.1)
upper <- list(alpha1 = 5, alpha2 = 2, beta = 2.5)


### 事前分布の設定
prior <- list(alpha1 = list(measure.type = "code",
                            df = "dunif(alpha1, 0.1, 5)"),
              alpha2 = list(measure.type = "code",
                            df = "dunif(alpha1, 0.5, 2)"),
              beta = list(measure.type = "code",
                          df = "dunif(alpha1, 0.1, 2.5)"))

### 疑似最尤推定量の計算
res <- qmle(obj,start = start, lower = lower,
            upper = upper, joint = TRUE, rcpp = TRUE)

### 適応型疑似ベイズ推定量の計算
bres <- adaBayes(obj,start = start, lower = lower,
                 upper = upper, rcpp = TRUE, method =
                   "nomcmc")
```

実行結果 (qmle)

```
### 出力
summary(res)
```

```
## Quasi-Maximum likelihood estimation
##
## Call:
## qmle(yuima = obj, start = start, lower = lower, upper = upper,
##     joint = TRUE, rcpp = TRUE)
##
## Coefficients:
##         Estimate  Std. Error
## beta   0.3000204 0.002121366
## alpha1 3.0451871 0.236139708
## alpha2 0.9924896 0.009492667
##
## -2 log L: -41007.26
```

実行結果 (adaBayes)

```
summary(bres)
```

```
## Maximum likelihood estimation
##
## Call:
## adaBayes(yuima = obj, start = start, lower = lower, upper = upper,
##     method = "nomcmc", rcpp = TRUE)
##
## Coefficients:
##         Estimate  Std. Error
## beta   0.3000572 0.002120743
## alpha1 3.0279170 0.236168742
## alpha2 0.9924783 0.009547987
##
## -2 log L:
```

実データ解析

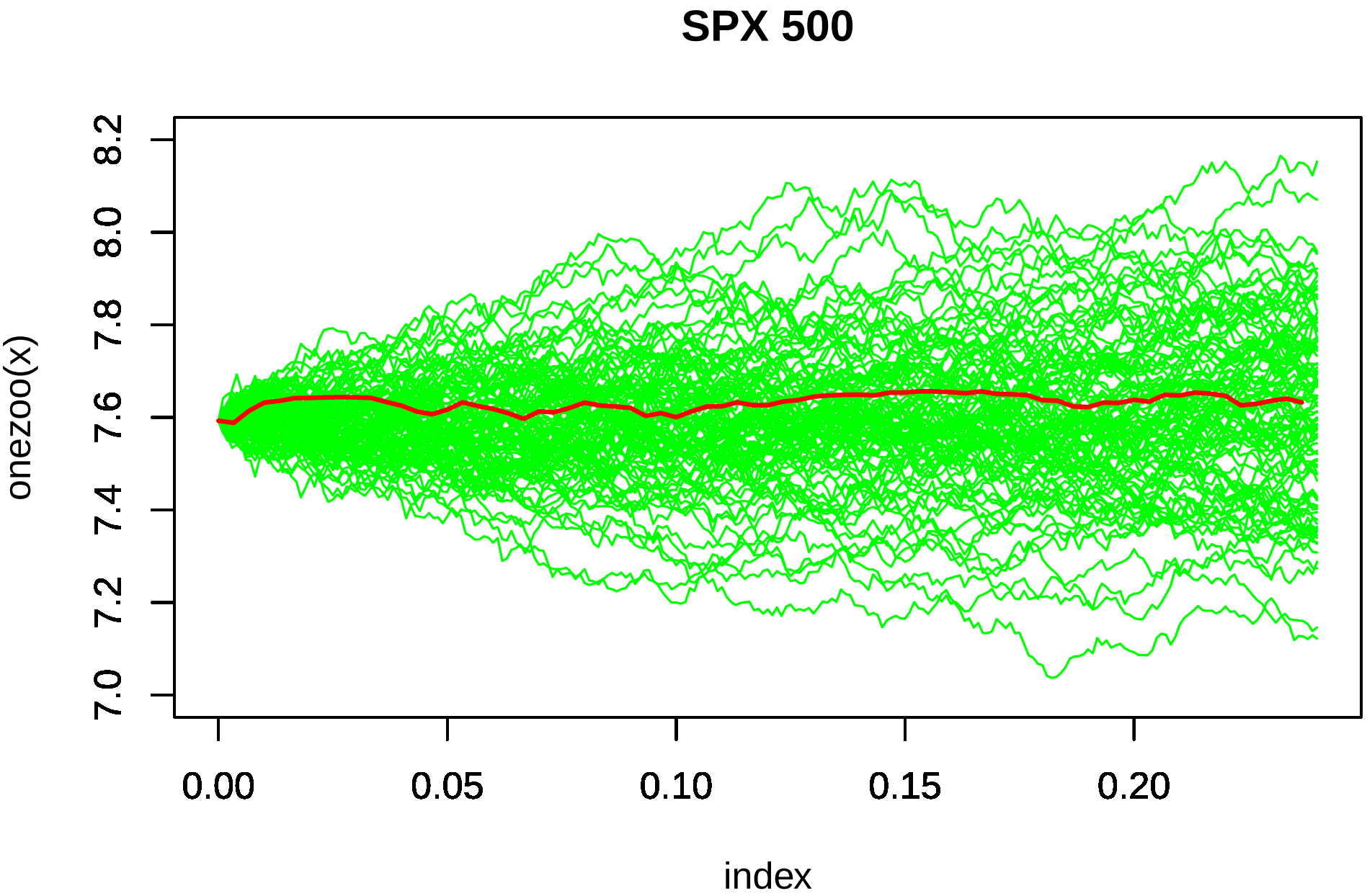
- YUIMA パッケージ上に内蔵されている SPX500 の 2012 年 7 月 9 日から 2015 年 4 月 1 日までの日次データ (logdayprice, 671 個) を用いる. 最初の 600 個を用いて推定量 を構成し, 残りの 71 個を用いてそのパフォーマンスを確認する.

```
data(LogSPX)
```

```
## List of 3
##  $ allObs     : num [1:53009] 0.00 -7.37e-05 -6.94e-04 -3.13e-03 -3.85e-03 ...
##  $ obsinday   : int 79
##  $ logdayprice: num [1:671] 7.21 7.21 7.2 7.2 7.2 ...
```

- このままだと, ベクトル値データなので, タイムスタンプを付け, YUIMA 上で解析可能にする (setYuima を用いることで実行できる).

実行結果



yuimaGUI

- YUIMA パッケージ内の (一部) 関数を実行可能なグラフィカルユーザインタフェース.
- R によるコーディングが不要で, web 上でも利用出来る (スマートフォンでも利用可).
- R から実行する場合は, `yuimaGUI` をインストール後 `yuimaGUI()` とすれば, 起動する.
- web 上では, https://yuima.shinyapps.io/yuimaGUI/ にアクセスすれば利用可能 (YUIMA プロジェクトのホームページにリンクがあります).