時系列解析（１３）

－粒子フィルタ－

東京大学 数理・情報教育研究センター

北川　源四郎

非線形状態空間モデル

$$x_n = F(x_{n-1}, v_n)$$
$$y_n = H(x_n, w_n)$$

x_n： 状態
y_n： 時系列

- 構造変化（非定常性）
- 異常値
- 非線形システム
- 離散過程

モンテカルロ法の導入

拡張カルマンフィルタの問題点

- 非線形変換の結果，予測分布が多峰になる場合に破綻する可能性がある

非線形・非ガウス型フィルタの問題点

- 計算量が大（高次元モデルへの適用ができない）
- 非線形変換など，モデルごとの実装に手間がかかる（多様なモデリングが困難）

⇨ モンテカルロ法の導入

非線形・非ガウス型フィルタの計算量

m	状態の次元
ℓ	システムノイズの次元
k_i	分点数
k_{q_j}	システムノイズの分点数

$$\text{計算量} \sim (k_1 \times \cdots \times k_m)(k_{q_1} + \cdots + k_{q_\ell})n \sim \ell k^{m+1} n$$

k	1	2	3	4	5	6	7
10	100	$1K$	$10K$	$100K$	$1M$	$10M$	$100M$
30	$1K$	$30K$	$1M$	$30M$	$1G$	$30G$	$1T$
100	$10K$	$1M$	$100M$	$10G$	$1T$	$100T$	$10P$
300	$100K$	$30M$	$10G$	$3T$	$1P$	$300P$	$100E$
1000	$1M$	$1G$	$1T$	$1P$	$1E$	$1Z$	$1Y$

逐次モンテカルロ法

Bootstrap Filter

Gordon, Salmond and Smith (1993)

Monte Carlo Filter/Smoother

Kitagawa (1993,1996)

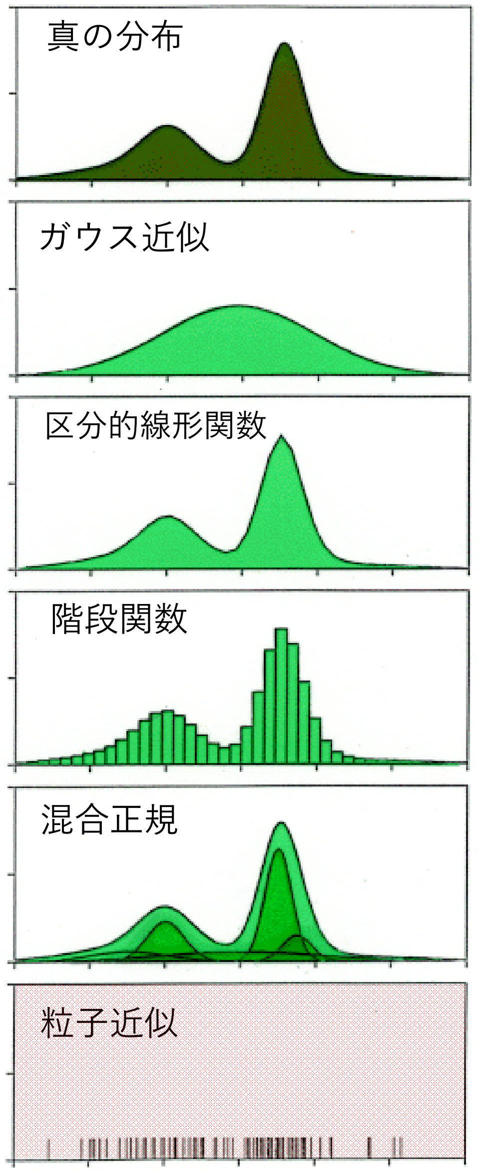
Sequential Monte Carlo Methods

Doucet, de Freitas and Gordon (2001)

Particle Filter（粒子フィルタ）

分布の近似

1. ガウス近似
   （拡張）カルマンフィルタ・平滑化
2. 区分的線形（階段）関数近似
   非ガウス型フィルタ・平滑化
3. 混合ガウス近似
   ガウス和フィルタ・平滑化
4. 粒子近似
   粒子フィルタ・平滑化



粒子による分布の近似

$$\left\{p_n^{(1)},\ldots,p_n^{(m)}\right\} \sim p(x_n \mid Y_{n-1})$$ 予測分布

$$\left\{f_n^{(1)},\ldots,f_n^{(m)}\right\} \sim p(x_n \mid Y_n)$$ フィルタ分布

$$\left\{s_{n|N}^{(1)},\ldots,s_{n|N}^{(m)}\right\} \sim p(x_n \mid Y_N)$$ 平滑化分布

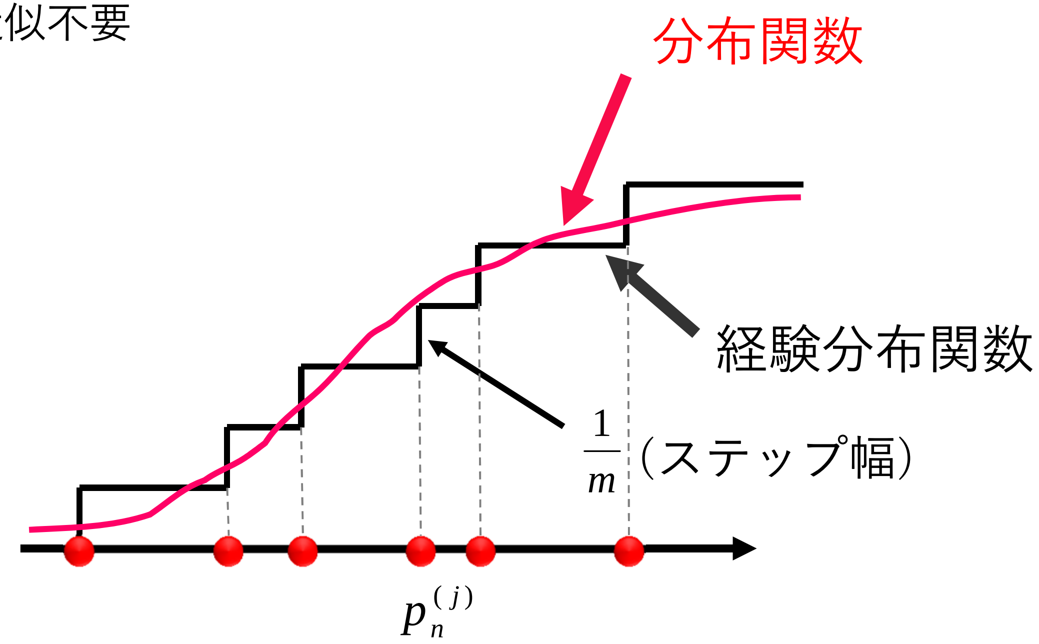
$$\left\{v_n^{(1)},\ldots,v_n^{(m)}\right\} \sim p(v_n)$$ システムノイズ分布

※ 観測ノイズは粒子近似不要

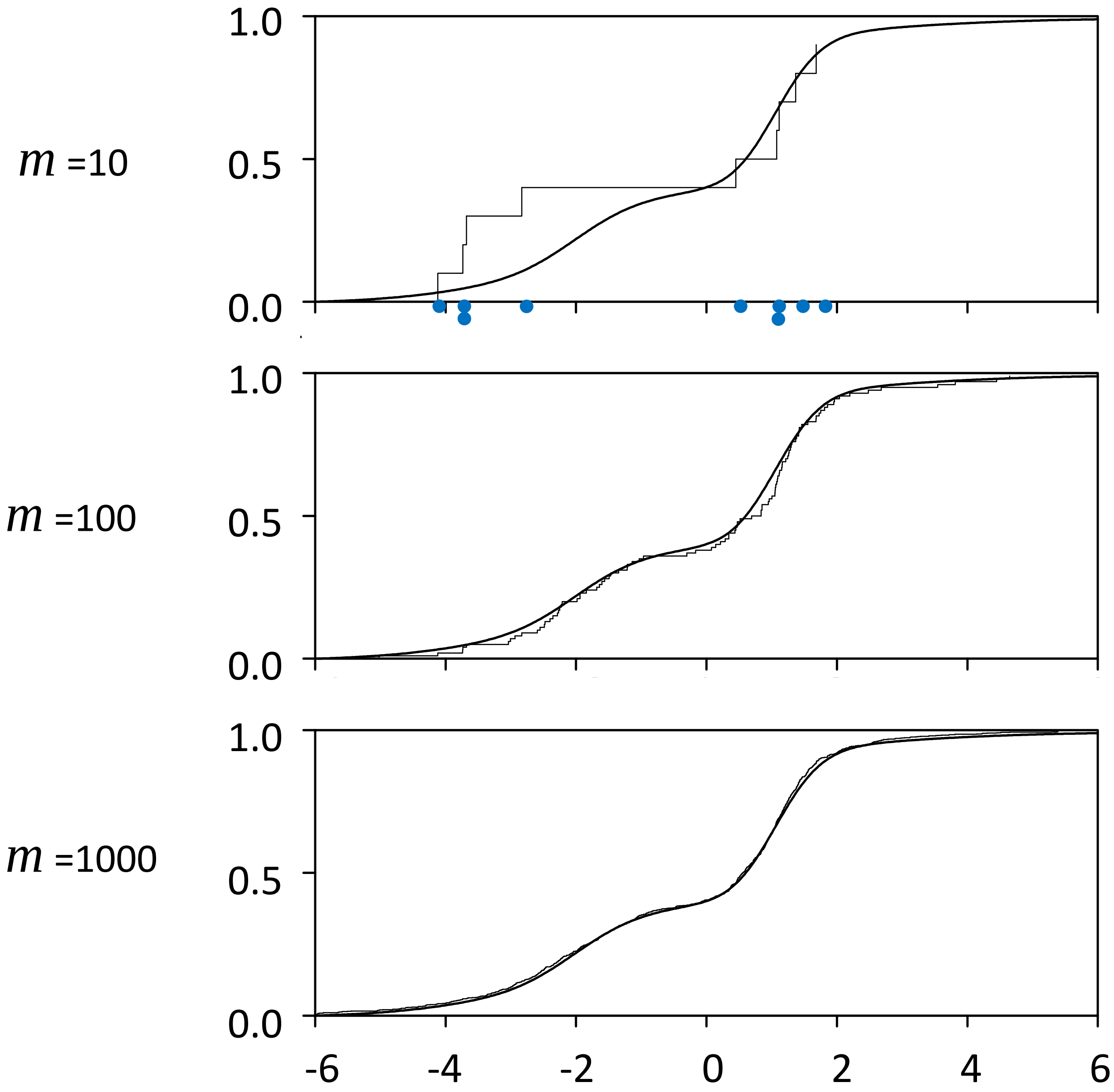
$$\left\{p_n^{(1)},\cdots,p_n^{(m)}\right\} \sim p(x_n \mid Y_{n-1})$$

$$\Leftrightarrow \quad \Pr(X_n = p_n^{(j)} | Y_{n-1}) = \frac{1}{m}$$

$$\Leftrightarrow \quad F_n(x) = \frac{1}{m}\sum_{j=1}^{m} I(x; p_n^{(j)})$$

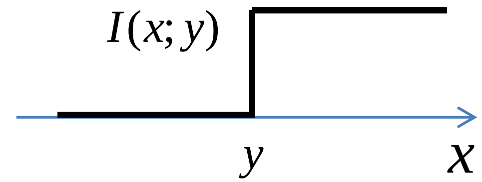


経験分布関数による分布の近似



$$F(x) = \int_{-\infty}^{x} f(x_n \mid Y_{n-1}) dx_n$$

$$F_n(x) = \frac{1}{m} \sum_{j=1}^{m} I(x; p_n^{(j)})$$



一期先予測の導出

$$p(x_n | Y_{n-1}) = \iint p(x_n, x_{n-1}, v_n | Y_{n-1}) dx_{n-1} dv_{n-1}$$

$$= \iint p(x_n | x_{n-1}, v_n, Y_{n-1}) p(v_n | x_{n-1}, Y_{n-1}) p(x_{n-1} | Y_{n-1}) dx_{n-1} dv_{n-1}$$

$$= \iint \delta(x_n - F(x_{n-1}, v_n)) p(v_n) p(x_{n-1} | Y_{n-1}) dx_{n-1} dv_{n-1}$$

$$f_{n-1}^{(j)} \sim p(x_{n-1} | Y_{n-1})$$

$$v_n^{(j)} \sim p(v)$$

$$p_n^{(j)} = F(f_{n-1}^{(j)}, v_n^{(j)}) \sim p(x_n | Y_{n-1})$$

一期先予測

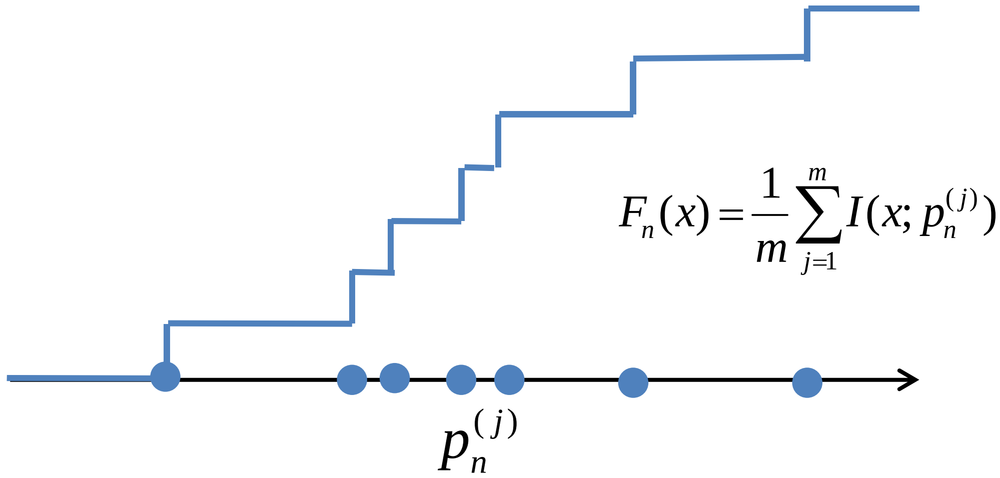
$$x_n = F(x_{n-1}, v_n)$$

システムモデル

$$v_n^{(j)} \sim p(v)$$

$$f_{n-1}^{(j)} \sim p(x_{n-1}|Y_{n-1})$$

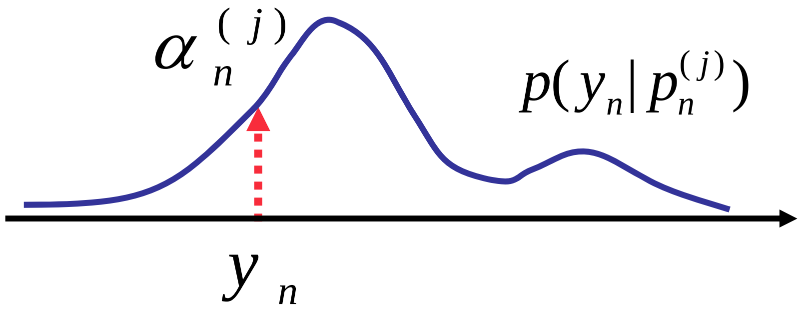
$$p_n^{(j)} = F(f_{n-1}^{(j)}, v_n^{(j)})$$



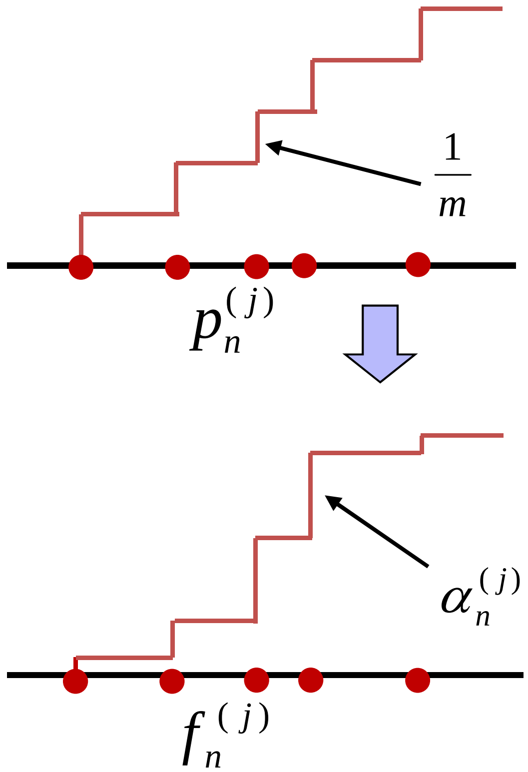
フィルタ

$\alpha_n^{(j)}$：粒子　$p_n^{(j)}$の重要度

$$\boxed{\alpha_n^{(j)} = p(y_n | X_n = p_n^{(j)})}$$



$$\begin{aligned}
\Pr(X_n = p_n^{(j)} | Y_n) &= \Pr(X_n = p_n^{(j)} | Y_{n-1}, y_n) \\
&= \frac{\Pr(X_n = p_n^{(j)}, y_n | Y_{n-1})}{\Pr(y_n | Y_{n-1})} \\
&= \frac{\Pr(y_n | X_n = p_n^{(j)}) \Pr(X_n = p_n^{(j)} | Y_{n-1})}{\sum_{i=1}^{m} \Pr(y_n | X_n = p_n^{(i)}) \Pr(X_n = p_n^{(i)} | Y_{n-1})} \\
&= \frac{\alpha_n^{(j)} \frac{1}{m}}{\sum_{i=1}^{m} \alpha_n^{(j)} \frac{1}{m}} = \frac{\alpha_n^{(j)}}{\sum_{i=1}^{m} \alpha_n^{(j)}}
\end{aligned}$$



リサンプリング

m 個の重み付き粒子で近似されたフィルタ分布を等確率の m 個の粒子で近似し直す

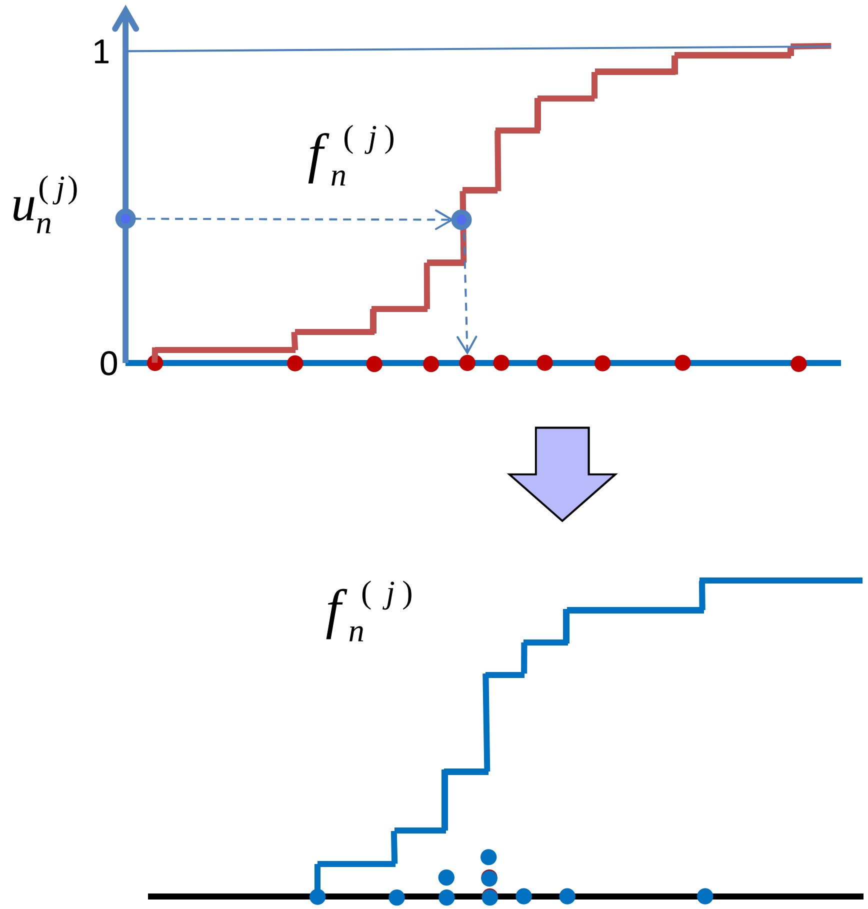
(a) 一様乱数 $u_n^{(j)} \in U[0,1)$ を生成する。

(b) $\frac{1}{C}\sum_{k=1}^{i-1}\alpha_n^{(k)} < u_n^{(j)} \le \frac{1}{C}\sum_{k=1}^{i}\alpha_n^{(k)}$ を満たす i を探す。

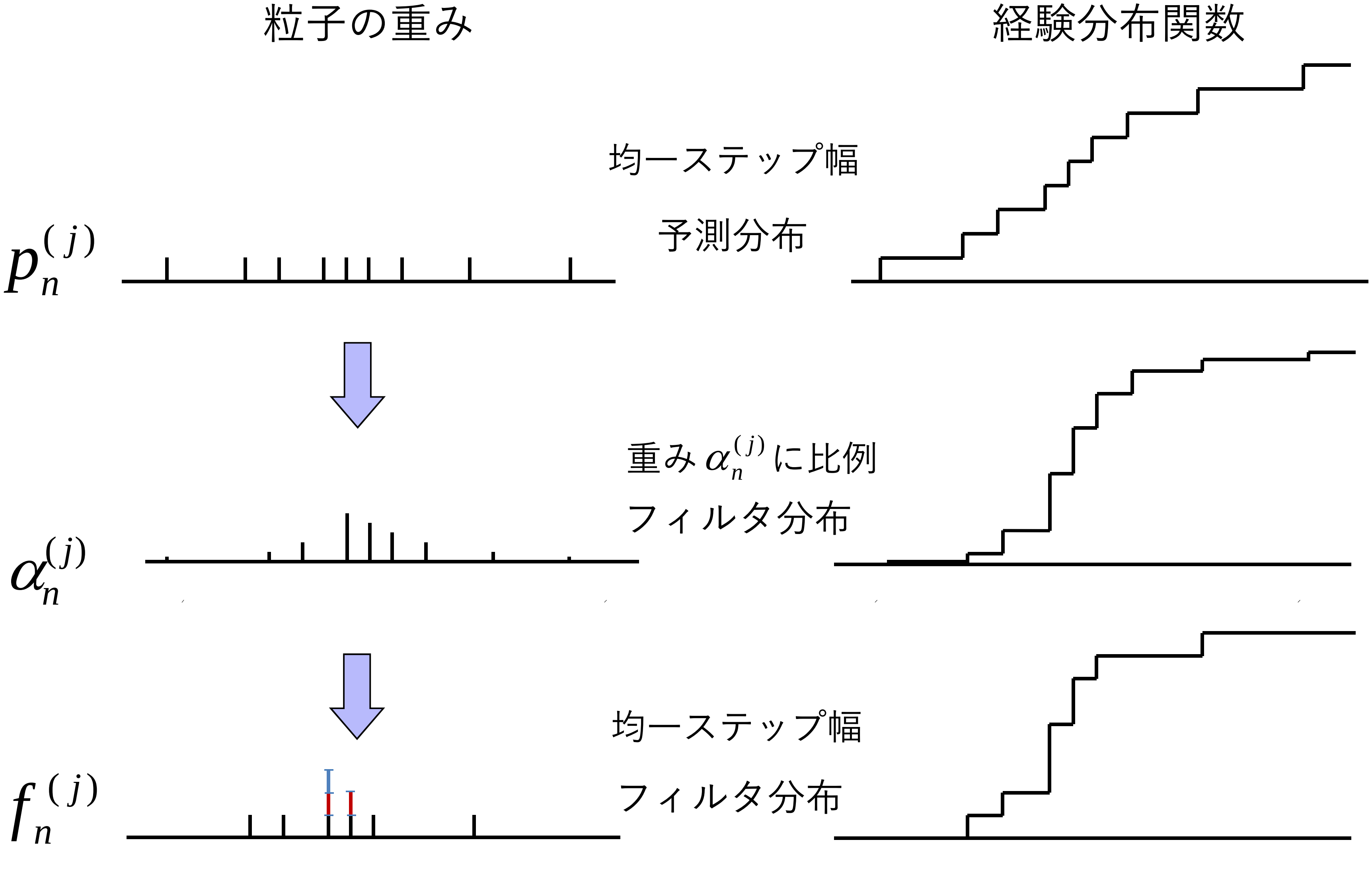
ただし，$C = \sum_{k=1}^{m}\alpha_n^{(k)}$

(c) フィルタの粒子を $f_n^{(j)} = p_n^{(i)}$ とする。

- 粒子の場所は不変
- 復元抽出（重複あり）



フィルタリング


粒子の重み
経験分布関数
$p_n^{(j)}$
均一ステップ幅
予測分布
$\alpha_n^{(j)}$
重み $\alpha_n^{(j)}$ に比例
フィルタ分布
$f_n^{(j)}$
均一ステップ幅
フィルタ分布

粒子フィルタ（MCF）

・システムノイズ

$$v_n^{(j)} \sim p(v) \qquad j = 1, \ldots, m$$

・予測分布

$$p_n^{(j)} = F(f_{n-1}^{(j)}, v_n^{(j)})$$

・重要度（ベイズ係数）

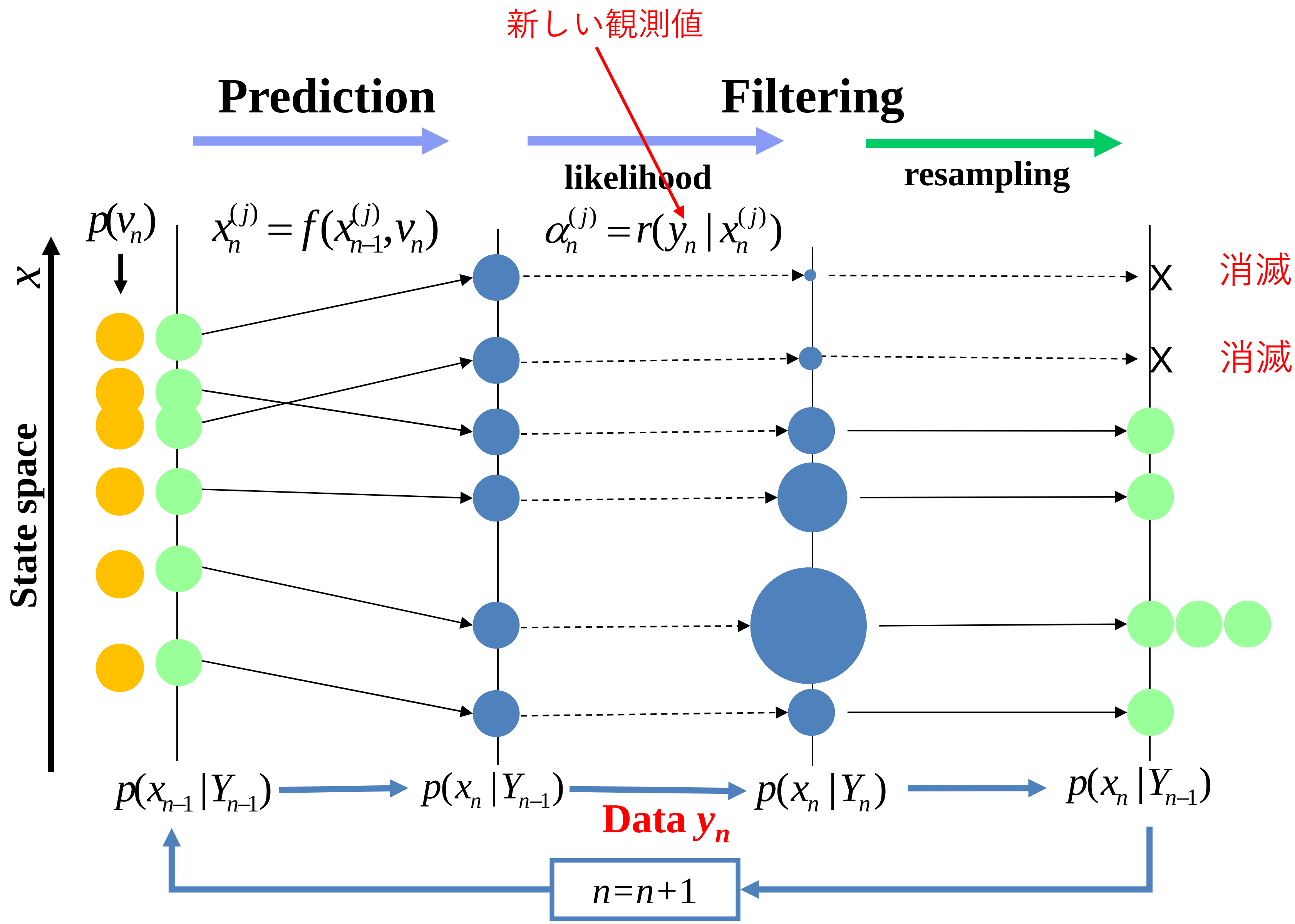
$$\alpha_n^{(j)} = p(y_n | p_n^{(j)})$$

・フィルタ分布のリサンプリング

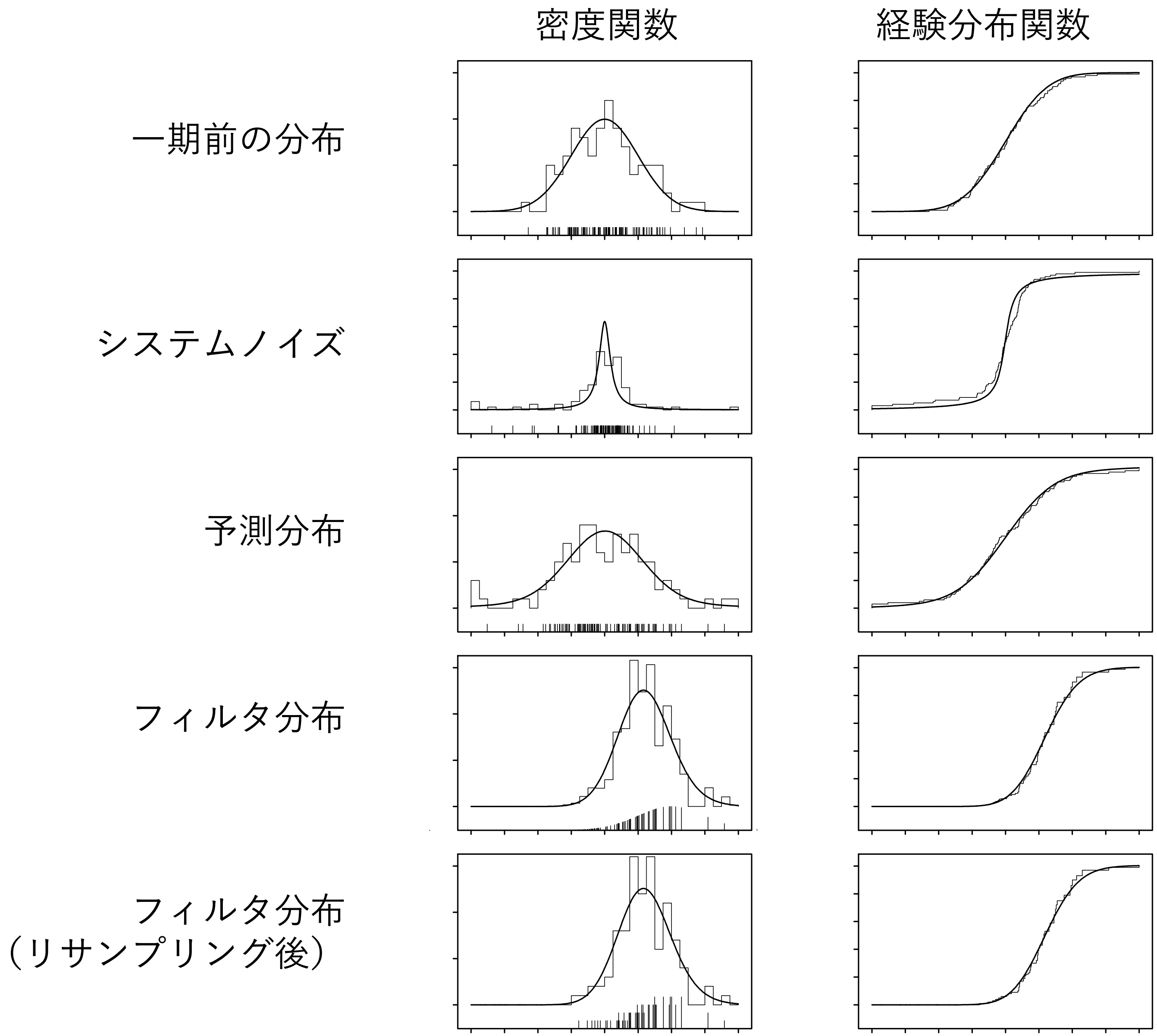
$$\left\{p_n^{(j)}\right\} \Rightarrow \left\{f_n^{(j)}\right\}$$

Gordon et al. (1993), Kitagawa (1996)
Doucet, de Freitas and Gordon (2001) “Sequential Monte Carlo Methods in Practice”

One Cycle of Monte Carlo Filtering

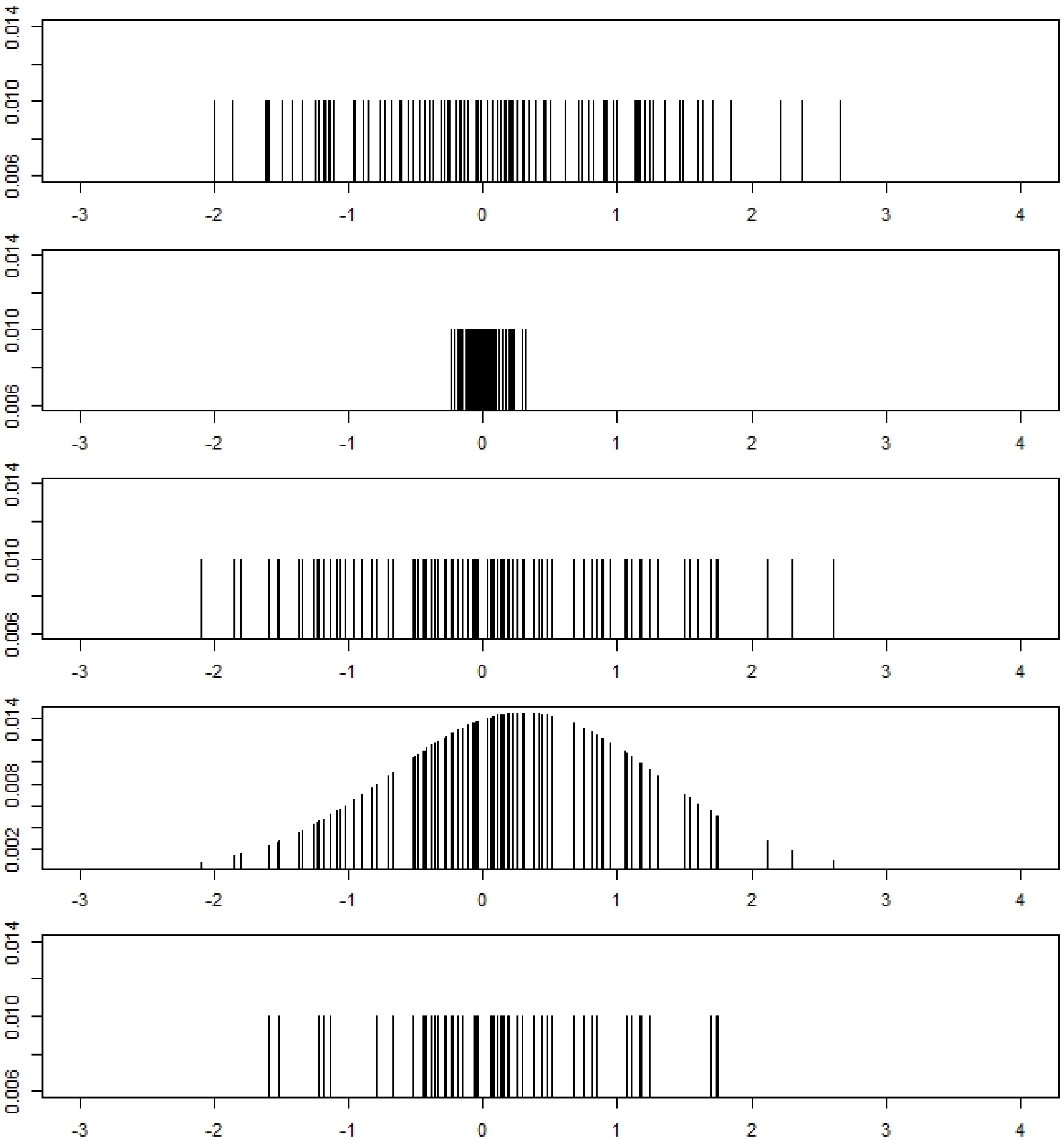


粒子フィルタの１サイクル



粒子フィルタの1サイクル（Rによる）

```
########################
#   粒子フィルタの1サイクル   #
########################
m <- 100
tau <- 0.1
sig <- 1.0
par(mar=c(2,2,1,1)+0.1)
par(mfrow=c(5,1))
alpha0 <- rep(1/m,length=m)
#
# Initial distribution
xf <- rnorm(m,mean=0,sd=1)
plot(xf,alpha0,type="h",xlim=c(-3,4))

# Prediction step
v  <- rnorm(m,mean=0,sd=tau)
xp <- xf + v
plot(v,alpha0,type="h",xlim=c(-3,4))
plot(xp,alpha0,type="h",xlim=c(-3,4))

# Bayes factor (weights of particles)
y <- 0.3
alpha <- dnorm(y-xp,mean=0,sd=sig,log=FALSE)
alpha <- alpha/sum(alpha)
# predictive, filter distributions
plot(xp,alpha,type="h",xlim=c(-3,4))

# re-sampling
xf1 <- sample(xp,prob=alpha,replace=T)
plot(xf1,alpha0,type="h",xlim=c(-3,4))

alpha2 <- rep(0,length=m)
```



リサンプリングは必要か？

- ✓ 必須ではない.
- ✓ かえって精度を落としている?

- 繰り返すと多くの重要度が 0 になる．毎回実施することは必須ではないが，時々実施することは不可欠.

- 判断基準
  - $\alpha_n^{(1)},\cdots,\alpha_n^{(m)}$ の最大値が $\frac{1}{m}$ の何倍か
  - $\sum_{j=1}^{m}\left(\alpha_n^{(j)}\right)^2$ の値　　　$(\frac{1}{m} < x < 1)$

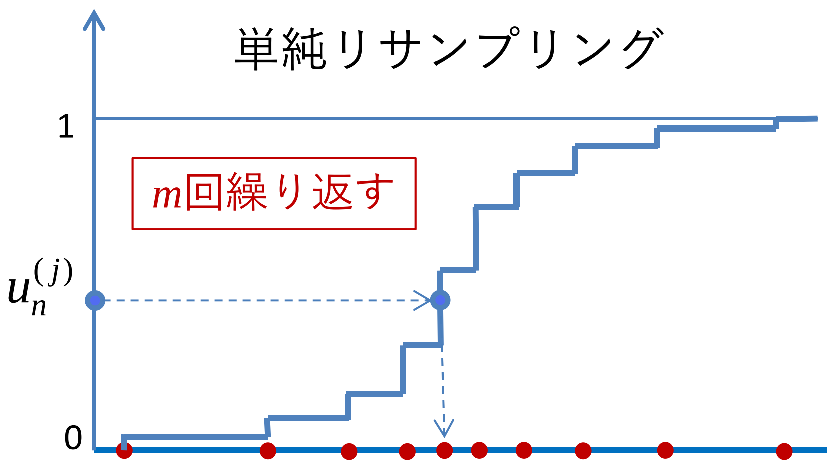
- 毎回実施するのも合理的判断

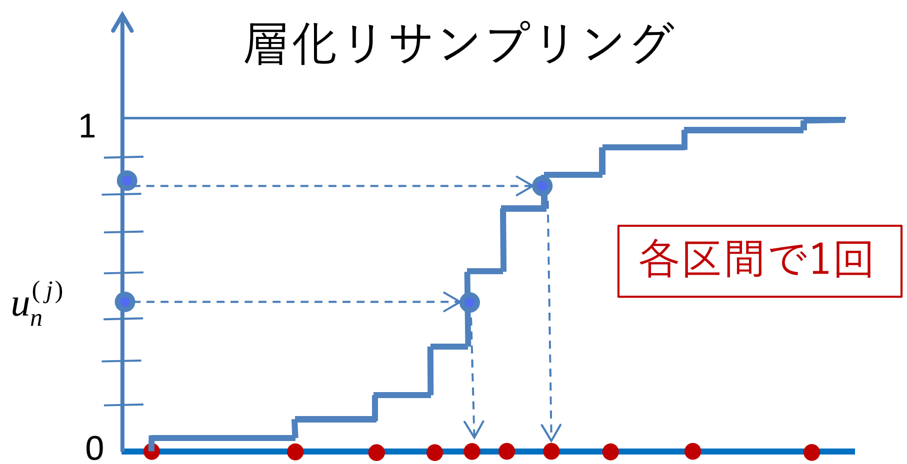
層化リサンプリング

リサンプリングの目的は重み $\{\alpha_n^{(1)},\cdots,\alpha_n^{(m)}\}$ を持つ粒子 $\{p_n^{(1)},\cdots,p_n^{(m)}\}$ で定義される分布関数を，重みが等しい経験分布で表現しなおすことであり，厳密なランダムサンプリングは必ずしも必要ではない。

(a') 一様乱数 $u_n^{(j)} \in U\left[\frac{j-1}{m},\frac{j}{m}\right)$ を生成する。

(a'') 固定した $\alpha \in [0,1)$ について，$u_n^{(j)} = \frac{j-1}{m}$ とする。





リサンプリング法の比較

D_f^*：厳密なフィルタ分布

m		$J(D_f, D_p)$			$J(D_p, D_f^*)$
		ランダム	層化ランダム	層化確定的	
10	Sort	0.0326	0.00379	0.00176	0.1021
	No-sort	0.0321	0.00859	0.00513	0.1021
100	Sort	0.00381	0.636×10^{-3}	0.311×10^{-4}	0.860×10^{-2}
	No-sort	0.00405	0.939×10^{-3}	0.612×10^{-3}	0.860×10^{-2}
1,000	Sort	0.398×10^{-3}	0.838×10^{-6}	0.407×10^{-6}	0.102×10^{-2}
	No-sort	0.379×10^{-3}	0.982×10^{-4}	0.611×10^{-4}	0.102×10^{-2}
10,000	Sort	0.387×10^{-4}	0.101×10^{-7}	0.498×10^{-8}	0.248×10^{-3}
	No-sort	0.406×10^{-4}	0.936×10^{-5}	0.636×10^{-5}	0.348×10^{-3}

$J(D_f, D_p)$ の違いは無視できる

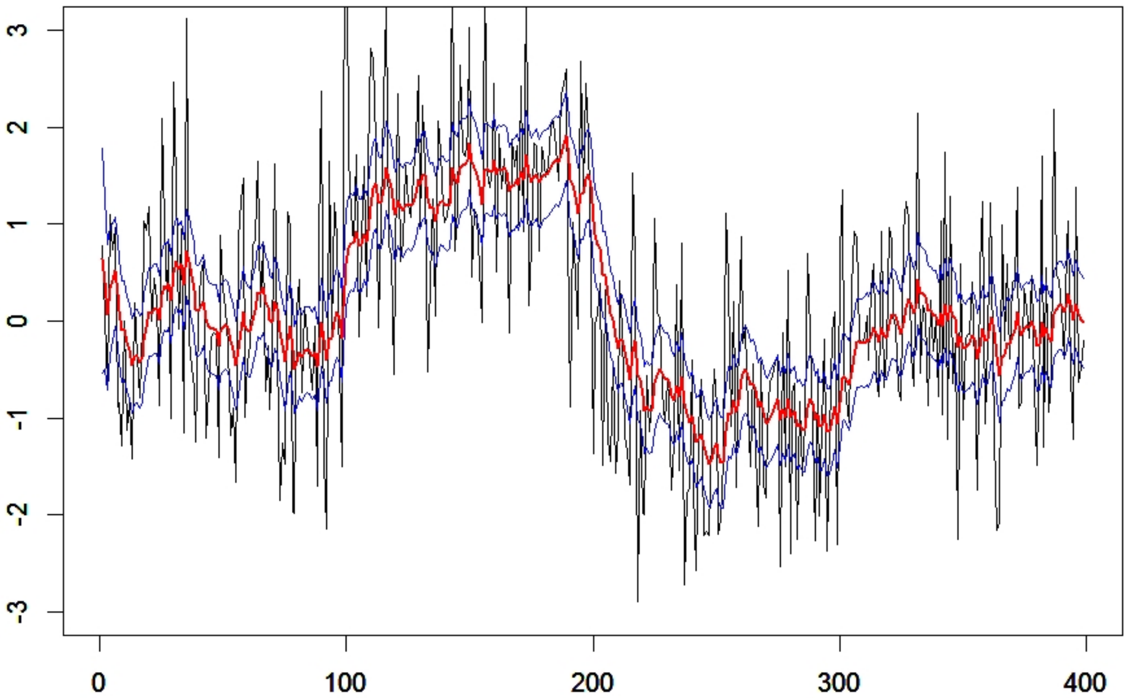
$$J(D_p, D_f) = \int_{-\infty}^{\infty} \left(D_p(x) - D_f(x)\right)^2 dx$$

$$= \int_{-\infty}^{\infty} \Big(\frac{1}{c}\sum_{i=1}^{m} \alpha_n^{(i)} I(x, p_n^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} I(x, f_n^{(i)})\Big) dx$$

$$c = \sum_{i=1}^{m} \alpha_n^{(i)}$$

Particle Filter

```
# Particle Filter
model <- 1
m <- 1000

# Gauss model
if (model == 1){
tau2 <- 0.018
sig2 <- 1.045
tau <- sqrt(tau2)
sig <- sqrt(sig2)
}

# Initial distribution
xf <- rnorm(m,mean=0,sd=2)
n <- length(y)
# trend <- rep(0,length=n)
trend <- matrix(0,nrow=n,ncol=3)
xs <- matrix( ncol=m, nrow=lag+1 )

for (ii in 1:n) {
# Prediction step
if ( model == 1 ) {v  <- rnorm(m,mean=0,sd=tau) }
if ( model == 2 ) {v  <- rcauchy(m,location=0,scale=tau) }
xp <- xf + v

# Bayes factor (weights of particles)
alpha <- dnorm(y[ii]-xp,mean=0,sd=sig,log=FALSE)
alpha <- alpha/sum(alpha)

# re-sampling
xf <- sample(xp,prob=alpha,replace=T)

trend[ii,1] <- quantile(xf,probs=0.1)
trend[ii,2] <- quantile(xf,probs=0.5)
trend[ii,3] <- quantile(xf,probs=0.9)
}

plot(y,type="l",ylim=c(-3,3))
par(new=T)
plot(trend[,1],type="l",col="blue",lwd=1,ylim=c(-3,3))
par(new=T)
plot(trend[,2],type="l",col="red",lwd=2,ylim=c(-3,3))
par(new=T)
plot(trend[,3],type="l",col="blue",lwd=1,ylim=c(-3,3))
```



平滑化

- 粒子近似の場合，固定区間平滑化のアルゴリズムは利用できない
- 他の方法が必要

フィルタ

$$\left\{p_n^{(j)}\right\} \xRightarrow{\left\{\alpha_n^{(j)}\right\}} \left\{f_n^{(j)}\right\}$$

平滑化

$$\left\{s_{1|n-1}^{(j)}, \cdots, s_{n-1|n-1}^{(j)}, p_n^{(j)}\right\} \xRightarrow{\left\{\alpha_n^{(j)}\right\}} \left\{s_{1|n}^{(j)}, \cdots, s_{n|n}^{(j)}\right\}$$

平滑化（粒子を保存する方法）

$$(s_{1|i}^{(j)},\cdots,s_{n|i}^{(j)})^T \sim p(x_1,\cdots,x_n \mid Y_i)$$

$$\Pr\left(x_1 = s_{1|n-1}^{(j)},\cdots,x_{n-1} = s_{n-1|n-1}^{(j)} \mid Y_{n-1}\right) = \frac{1}{m}$$

$$v_n^{(j)} \sim q(v)$$

とするとき、n 個の粒子を

$$p_{i|n-1}^{(j)} = \begin{cases} s_{i|n-1}^{(j)} & \text{for } i = 1,\ldots,n-1 \\ F(s_{n-1|n-1}^{(j)}, v_n^{(j)}) & \text{for } i = n \end{cases}$$

と定義すると

$$(p_{1|n-1}^{(j)},\cdots,p_{n|n-1}^{(j)})^T \sim p(x_1,\cdots,x_n \mid Y_{n-1})$$

平滑化（粒子を保存する方法）(2)

y_n　　新しいデータ

$$\Pr(x_i = p_{1|n-1}^{(j)}, \cdots, x_n = p_{n|n-1}^{(j)} \mid Y_n)$$

$$= \Pr(x_i = p_{1|n-1}^{(j)}, \cdots, x_n = p_{n|n-1}^{(j)} \mid Y_{n-1}, y_n)$$

$$= \frac{p(y_n \mid x_i = p_{1|n-1}^{(j)}, \cdots, x_n = p_{n|n-1}^{(j)}, Y_{n-1}) \Pr(x_i = p_{1|n-1}^{(j)}, \cdots, x_n = p_{n|n-1}^{(j)} \mid Y_{n-1})}{\Pr(y_n \mid Y_{n-1})}$$

$$= \frac{\Pr(y_n \mid p_{n|n-1}^{(j)}) \Pr(x_i = p_{1|n-1}^{(j)}, \cdots, x_n = p_{n|n-1}^{(j)} \mid Y_{n-1})}{\Pr(y_n \mid Y_{n-1})}$$

ただし、$p_{n|n-1}^{(j)} \equiv p_n^{(j)}$

$p(x_1, \cdots, x_n \mid Y_{n-1})$ に従うn次元粒子 $\left\{ \left(s_{1|n\text{-}1}^{(j)}, \ldots, s_{n-1|n-1}^{(j)}, p_n^{(j)} \right)^T, j = 1, \ldots, m \right\}$ のリサンプリングによって$p(x_1, \cdots, x_n \mid Y_n)$ に従う等確率のn 次元粒子 $\left\{ \left(s_{1|n}^{(j)}, \ldots, s_{n-1|n}^{(j)}, s_{n|n}^{(j)} \right)^T, j = 1, \ldots, m \right\}$を生成できる

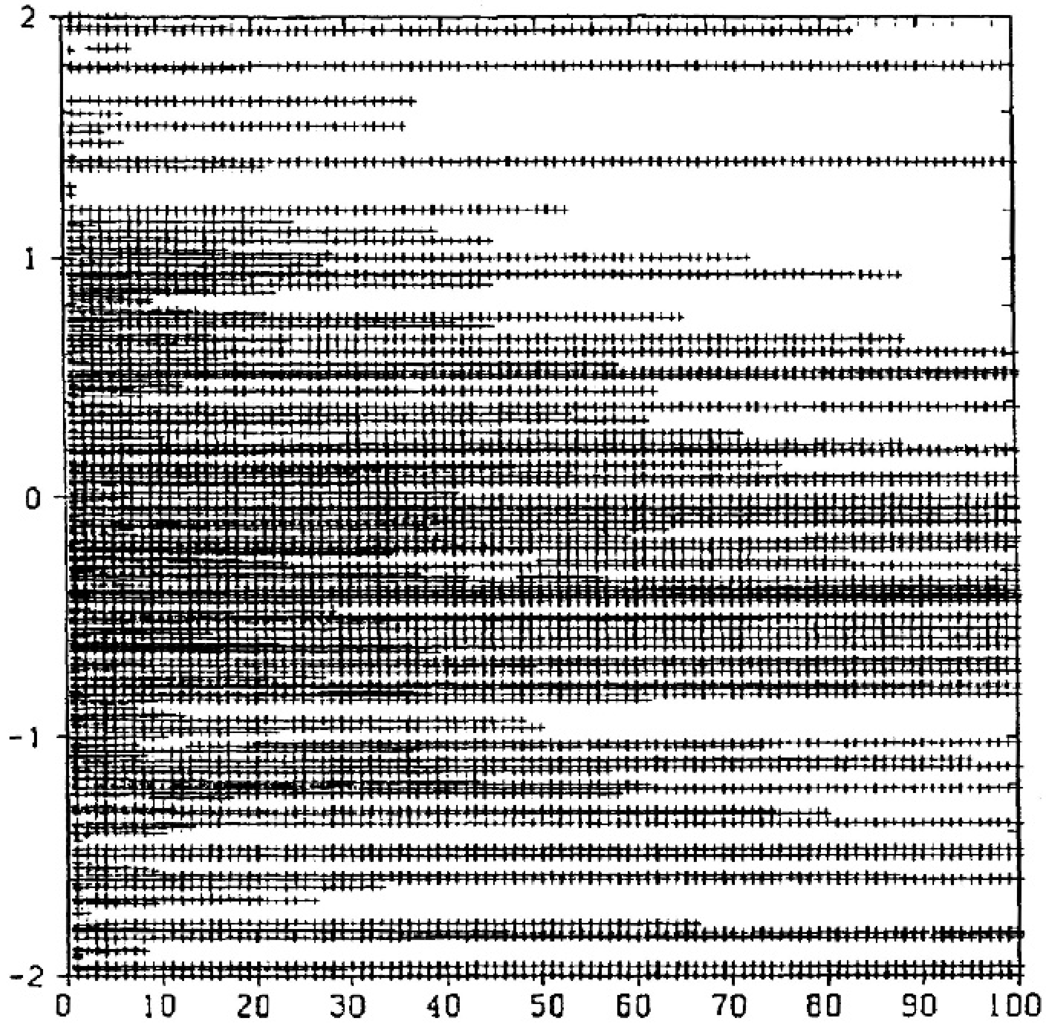
平滑化の困難

問題点

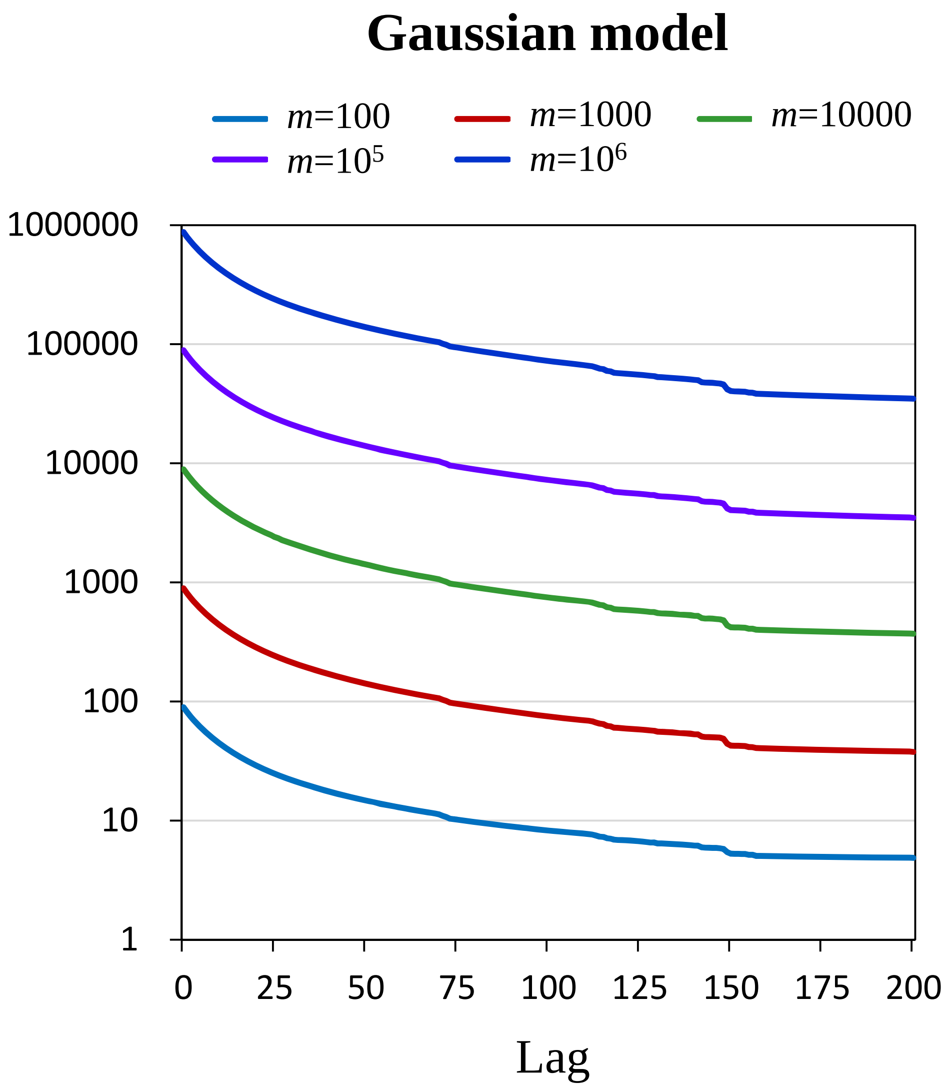
- リサンプリングによって（実質的）粒子数が単調減少する

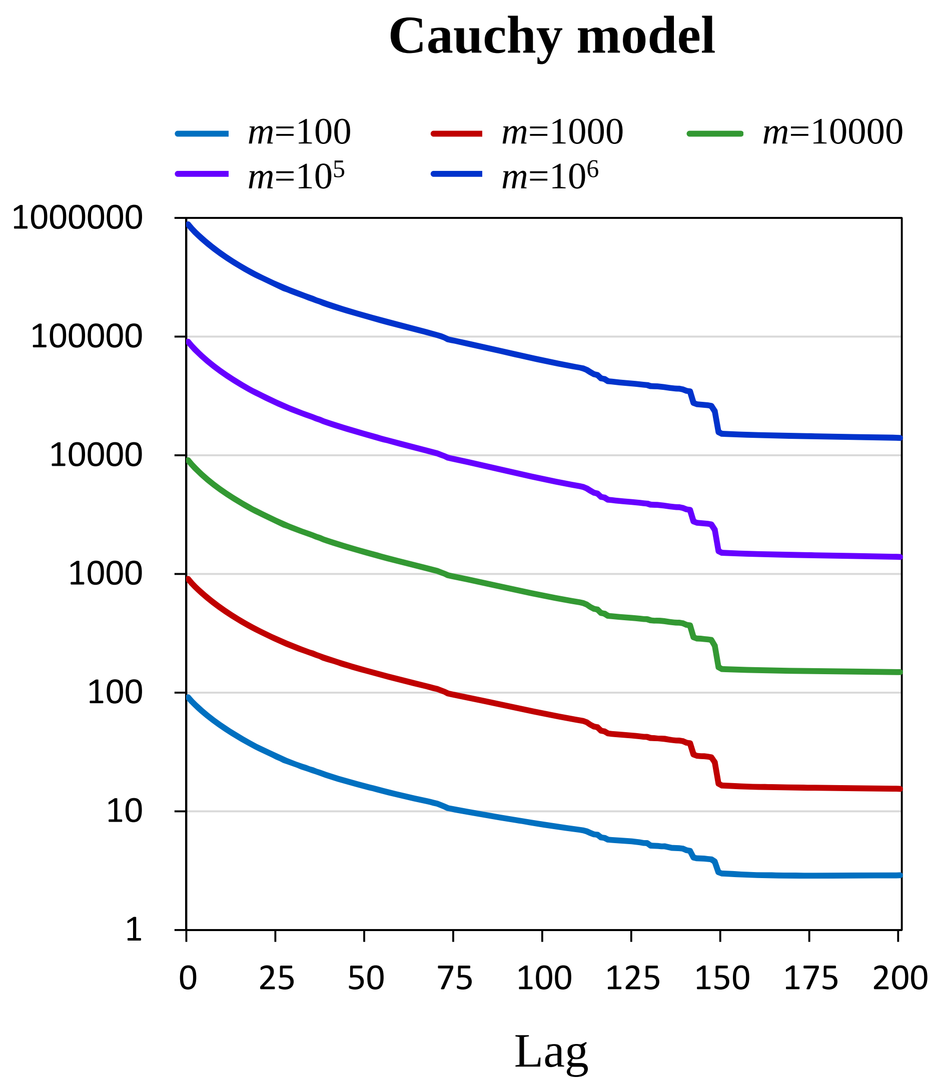
解決策

1. 固定ラグ平滑化を用い、ラグを大きくしない
2. 2方向フィルタを用いる

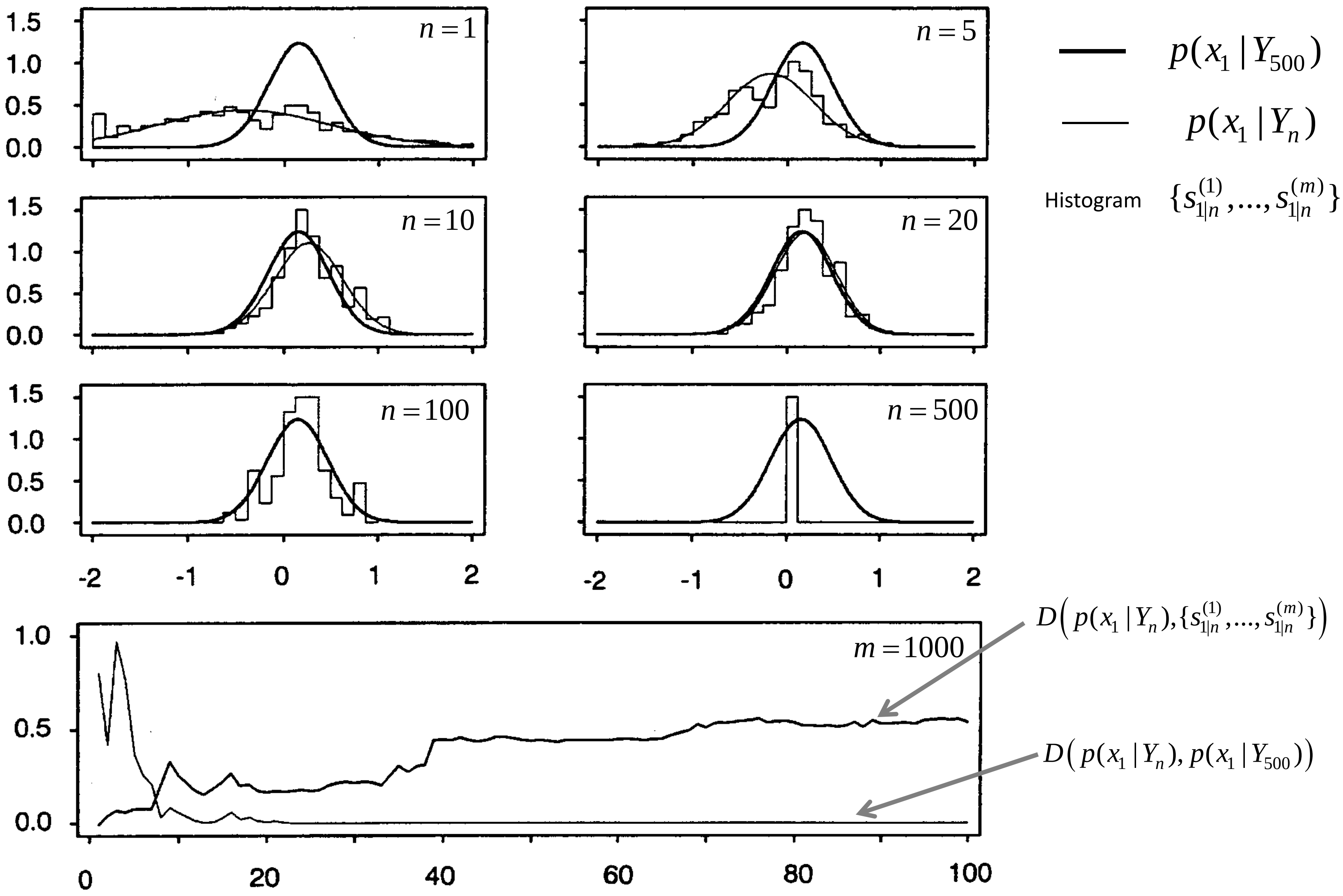

2
1
0
-1
-2
0
10
20
30
40
50
60
70
80
90
100

固定ラグ平滑化における異なる粒子の数





ラグと近似精度



$n=1$
$n=5$
$n=10$
$n=20$
$n=100$
$n=500$
$m=1000$
$p(x_1 \mid Y_{500})$
$p(x_1 \mid Y_n)$
Histogram $\{s_{1|n}^{(1)},\ldots,s_{1|n}^{(m)}\}$
$D\left(p(x_1 \mid Y_n),\{s_{1|n}^{(1)},\ldots,s_{1|n}^{(m)}\}\right)$
$D\left(p(x_1 \mid Y_n),\, p(x_1 \mid Y_{500})\right)$

平滑化分布の改善

（固定区間）平滑化

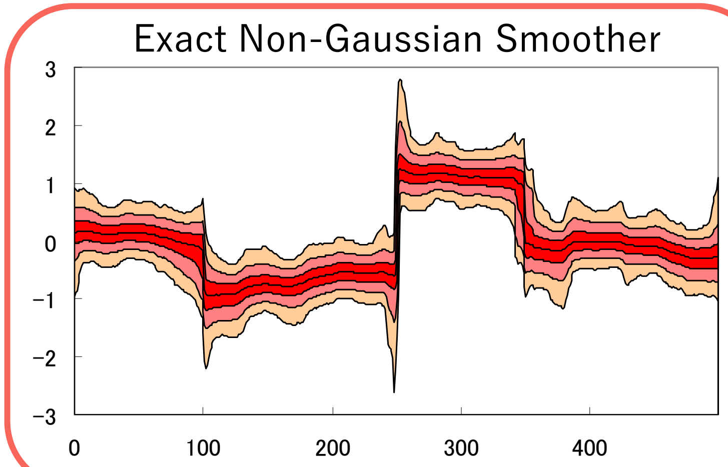
$$\left\{ s_{1|n-1}^{(j)}, \cdots, s_{n-1|n-1}^{(j)}, p_n^{(j)} \right\} \Longrightarrow \left\{ s_{1|n}^{(j)}, \cdots, s_{n|n}^{(j)} \right\}$$

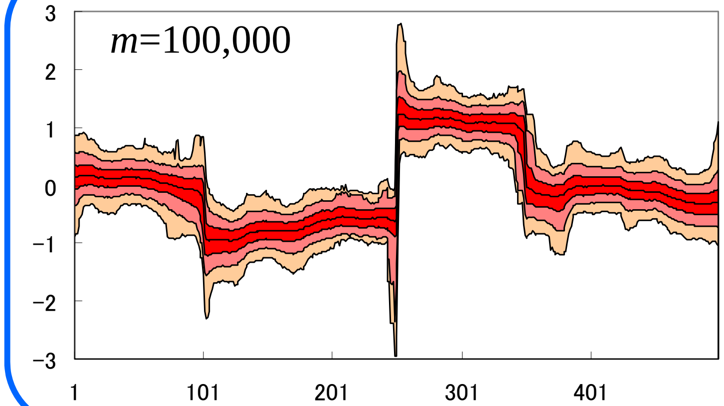
固定ラグ平滑化

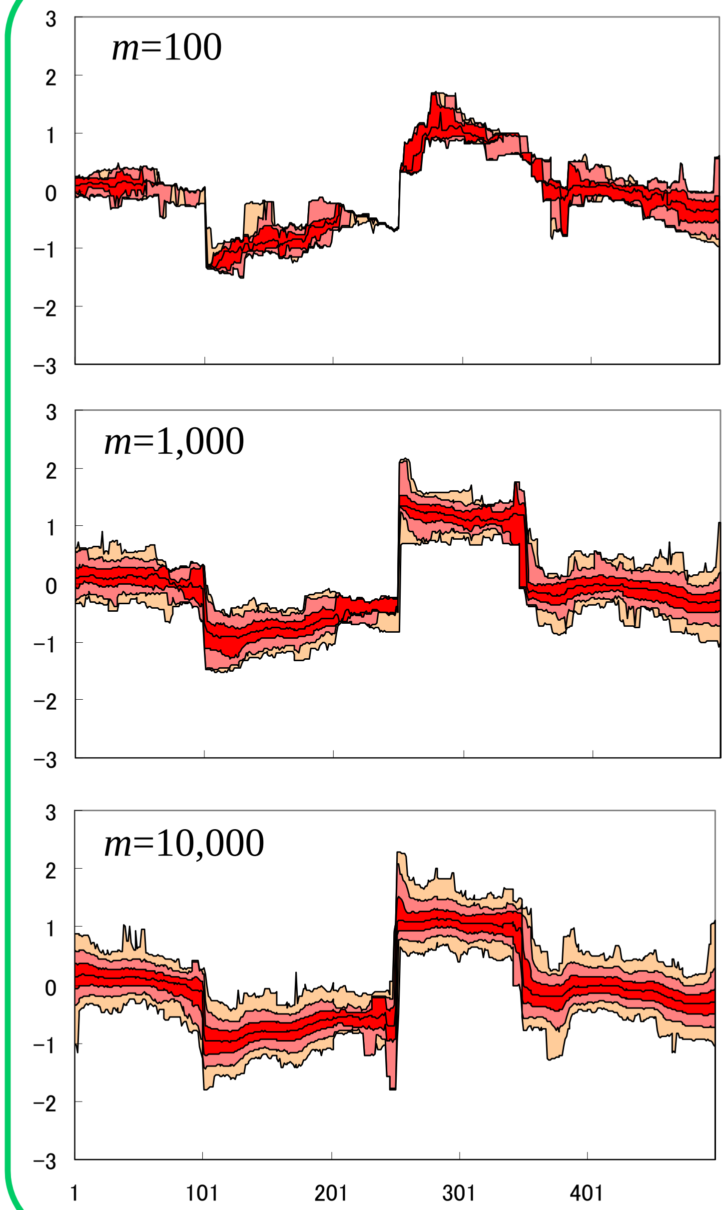
$$\left\{ s_{n-L|n-1}^{(j)}, \cdots, s_{n-1|n-1}^{(j)}, p_n^{(j)} \right\} \Longrightarrow \left\{ s_{n-L|n}^{(j)}, \cdots, s_{n|n}^{(j)} \right\}$$

すべての履歴でなく、最近の L 個だけをリサンプルすると固定ラグ（Lラグ）平滑化が実現できる。

粒子数による変化


Exact Non-Gaussian Smoother
3
2
1
0
−1
−2
−3
0
100
200
300
400


m=100,000
3
2
1
0
−1
−2
−3
1
101
201
301
401


m=100
m=1,000
m=10,000
3
2
1
0
−1
−2
−3
1
101
201
301
401

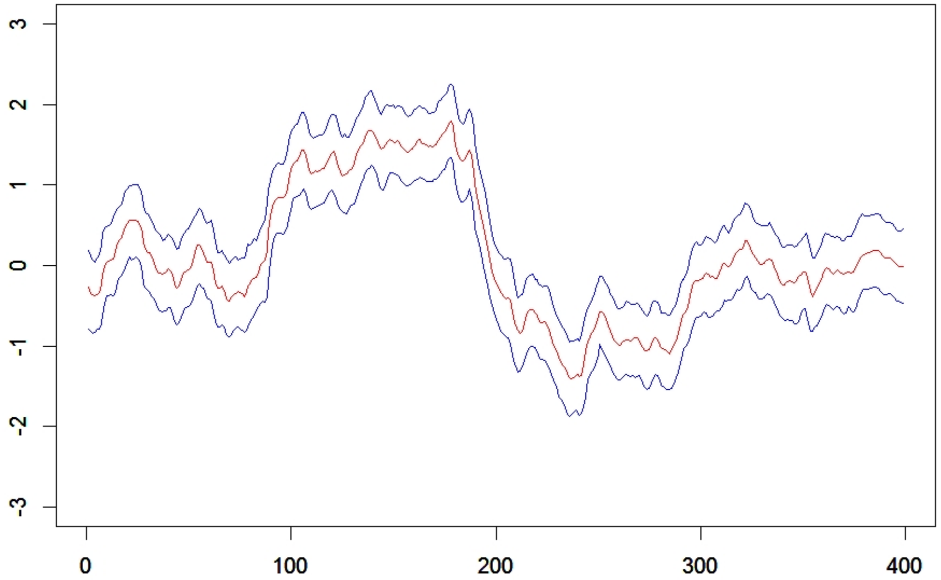
Particle Smoother

```
#  TEST data
y <- as.ts(read.csv("Chap83B_new2.csv"))
par(mar=c(2,2,1,1)+0.1)
plot(y,ylim=c(-4,4))

#  Particle smoother
#
model <- 2
m <- 10000
lag <- 20

# Gauss model
if (model == 1){
tau2 <- 0.018
sig2 <- 1.045
tau <- sqrt(tau2)
sig <- sqrt(sig2)
}
# Cauchy model
if (model == "2"){
tau22 <- 0.0000355
sig22 <- 1.006
scal2 <- sqrt(tau22)
sig <- sqrt(sig22)
}

# Initial distribution
xf <- rnorm(m,mean=0,sd=1)
n <- length(y)
trend <- matrix( nrow=n, ncol=7 )
xs <- matrix( nrow=m, ncol=lag+1 )
#strend <- rep(0,length=n)
strend <- matrix( nrow=n, ncol=7 )

for (ii in 1:n) {
#
# Prediction step
if ( model == 1 ) {v  <- rnorm(m,mean=0,sd=tau)}
if ( model == 2 ) {v  <- rcauchy(m,location=0,scale=scal2)}
xp <- xf + v

# Bayes factor (weights of particles)
alpha <- dnorm(y[ii]-xp,mean=0,sd=sig,log=FALSE)
alpha <- alpha/sum(alpha)

# re-sampling
# xf <- sample(xp,prob=alpha,replace=T)
ind  <- 1:m
jnd <- sample(ind,prob=alpha,replace=T)
for (i in 1:m){
 xf[i] <- xp[jnd[i]]
 for (j in 1:lag) {
   xs[i,lag+2-j] <- xs[jnd[i],lag+1-j]
 }
xs[i,1] <- xf[i]
}

# trend[ii,4] <- mean(xs[,lag+1])
 trend[ii,3] <- quantile(xs[,lag+1], prob=0.1, na.rm=TRUE)
 trend[ii,4] <- quantile(xs[,lag+1], prob=0.5, na.rm=TRUE)
 trend[ii,5] <- quantile(xs[,lag+1], prob=0.9, na.rm=TRUE)
}

for (ii in lag+1:n) {
  strend[ii-lag,4] <- trend[ii,4]
  strend[ii-lag,3] <- trend[ii,3]
  strend[ii-lag,5] <- trend[ii,5]
}
for (ii in 1:lag) {
#strend[n+1-ii,4] <- mean(xs[,ii])
 strend[n+1-ii,3] <- quantile(xs[,ii], prob=0.1, na.rm=TRUE)
 strend[n+1-ii,4] <- quantile(xs[,ii], prob=0.5, na.rm=TRUE)
 strend[n+1-ii,5] <- quantile(xs[,ii], prob=0.9, na.rm=TRUE)
}
#plot(y,type="l",ylim=c(-3,3))
#par(new=T)
plot(strend[,3],type="l",col="blue",lwd=1,ylim=c(-3,3))
par(new=T)
plot(strend[,4],type="l",col="red",lwd=1,ylim=c(-3,3))
par(new=T)
plot(strend[,5],type="l",col="blue",lwd=1,ylim=c(-3,3))
```

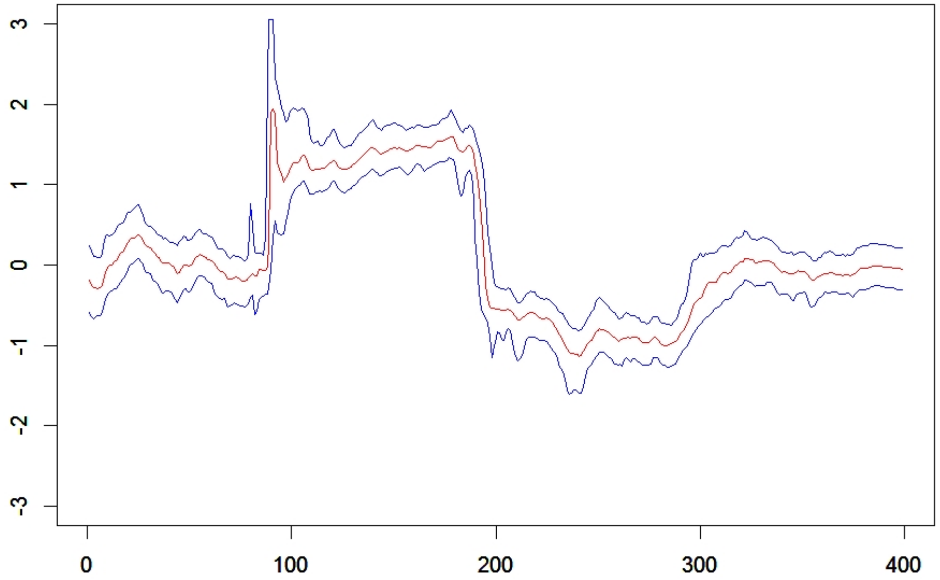
Particle Smoother

m=10000
lag = 20

Gauss



Cauchy

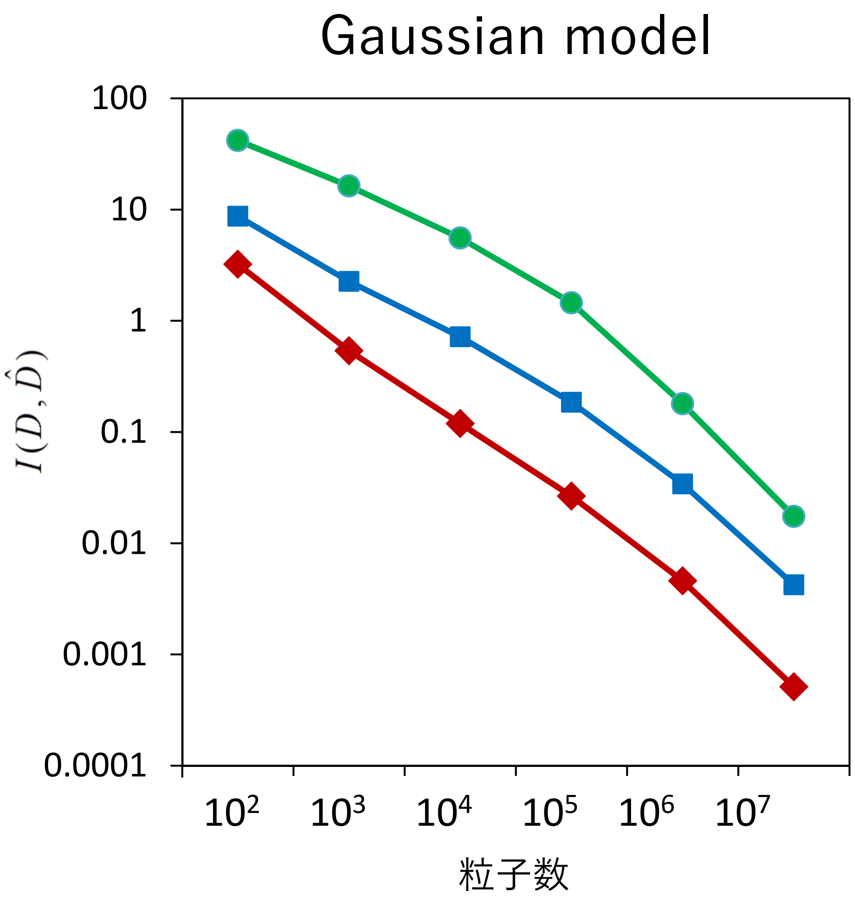


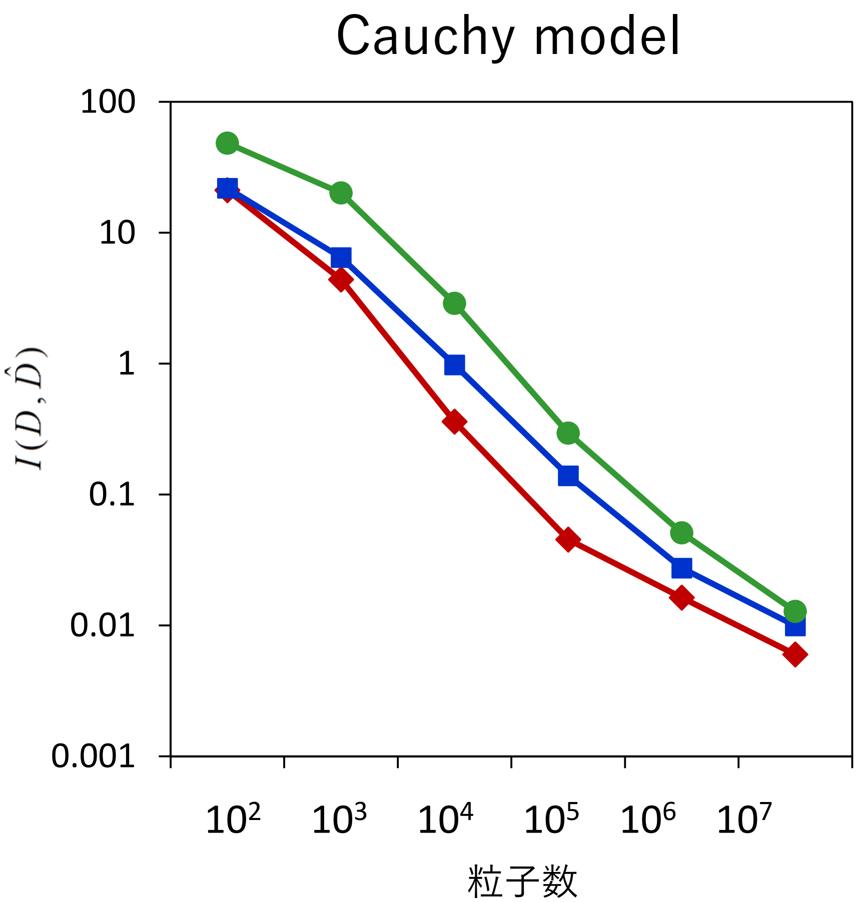
$m \to \infty$

粒子数を巨大にするとどうなるか？

- 現在の計算機では（特に工夫はしないで）1億粒子（フィルタリングだけなら10億）の計算はできる.
- 並列計算機を利用すれば100億（フィルタリングなら1兆）粒子程度までは実行できそう.

粒子数と計算精度





Smoothing (max lag)　Smoothing (best lag)　Filter

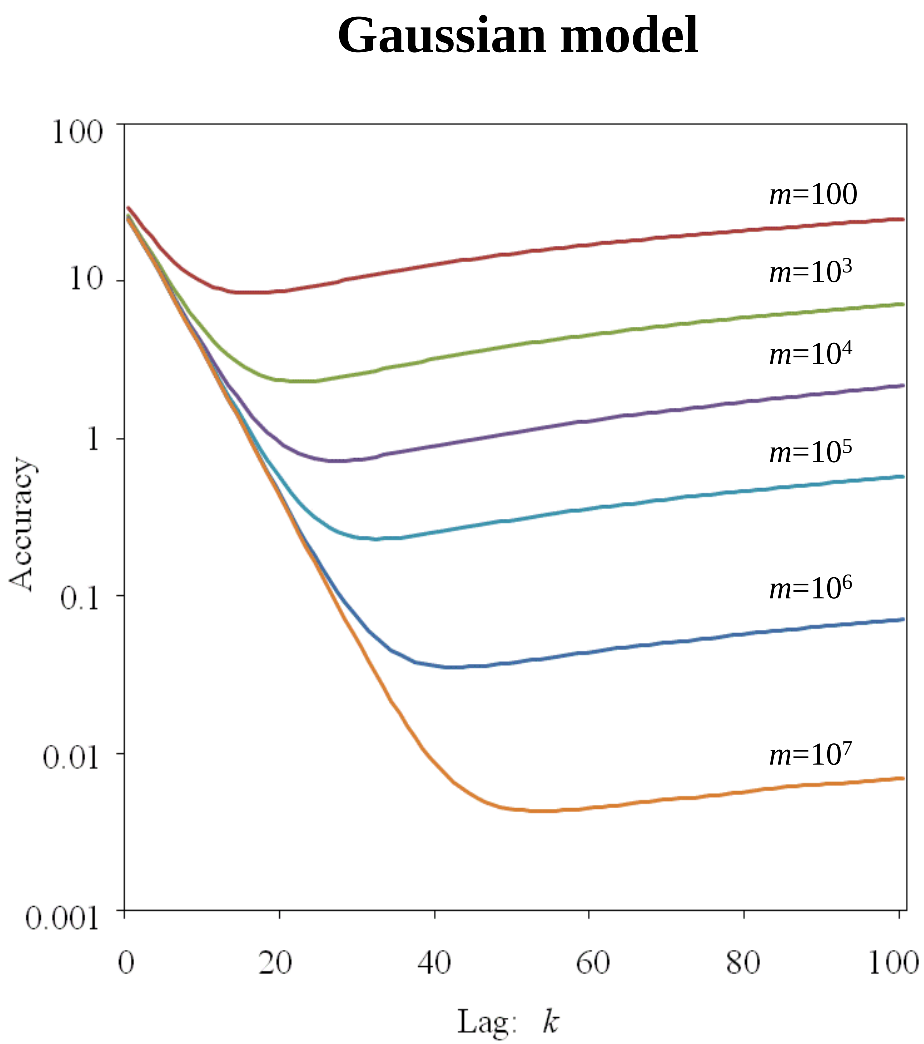
$$I(D,\hat{D}) = \sum_{n=1}^{N} \sum_{j=1}^{L} \left\{ D(x_j, n) - \hat{D}(x_j, n) \right\}^2 \Delta x$$

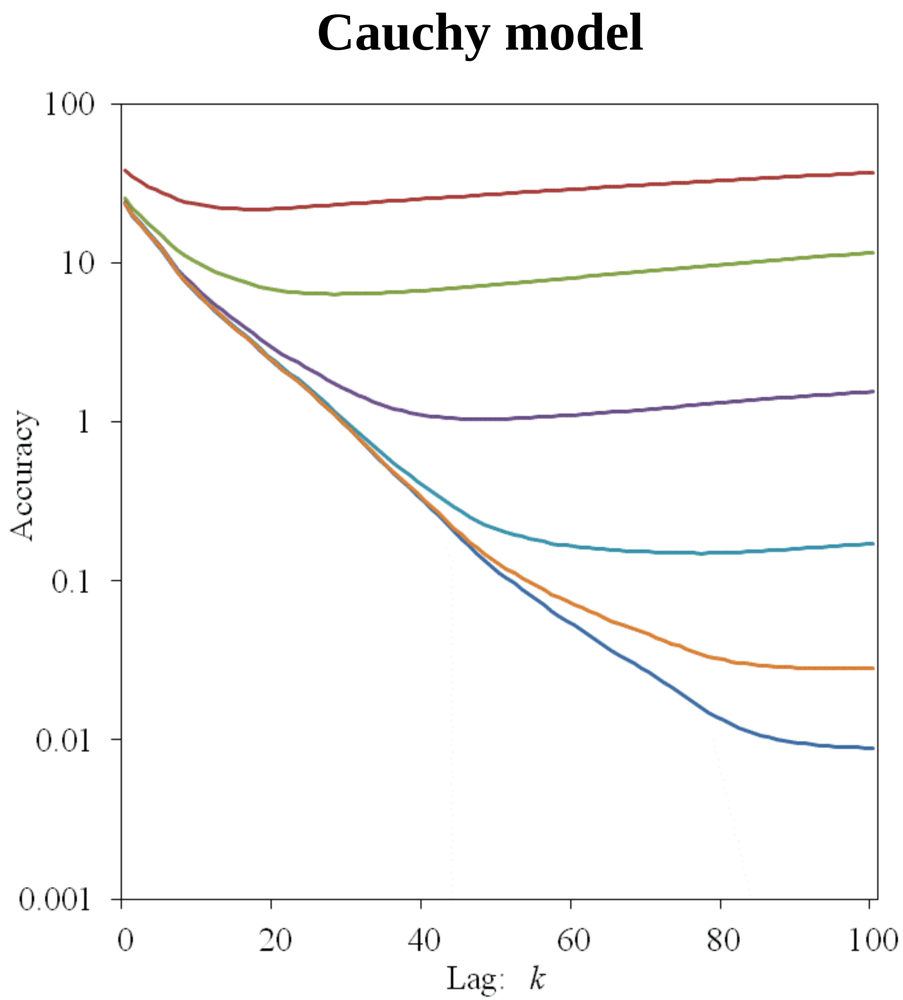
対数尤度の計算精度と計算時間

	Gaussian model			Cauchy model		
m	Log-L	S.D.	CPU-time	Log-L	S.D.	CPU-time
10^2	-750.859	2.287	0.02	-752.207	6.247	0.02
10^3	-748.529	1.115	0.06	-743.244	2.055	0.06
10^4	-748.127	0.577	0.58	-742.086	0.429	0.63
10^5	-747.960	0.232	5.84	-742.024	0.124	6.27
10^6	-747.931	0.059	59.41	-742.029	0.038	62.73
10^7	-747.926	0.023	591.04	-742.026	0.013	680.33
10^8	-747.930	0.008	5906.62	-742.026	0.003	6801.55
10^9	-747.928	0.002	59077.35	-742.026	0.001	69255.03

CPU-time：秒

固定ラグ平滑化の精度





Evaluated by $I(D, \hat{D})$

２方向フィルタによる平滑化

フィルタ

$$p(x_n|Y_n) = p(x_n|Y_{n-1}, y_n)$$
$$\propto p(x_n, y_n|Y_{n-1})$$
$$= \underline{p(y_n|x_n)}\,p(x_n|Y_{n-1})$$

$$Y_n = \{y_1, \ldots, y_n\}$$
$$Y_N^n = \{y_n, \ldots, y_N\}$$

平滑化

$$p(x_n|Y_N) = p(x_n|Y_{n-1}, Y_N^n)$$
$$\propto p(x_n, Y_N^n|Y_{n-1})$$
$$= \underline{p(Y_N^n|x_n)}\,p(x_n|Y_{n-1})$$

$$p(y_n|x_n)$$
$$\Downarrow$$
$$p(Y_N^n|x_n)$$

後向きフィルタ

初期化

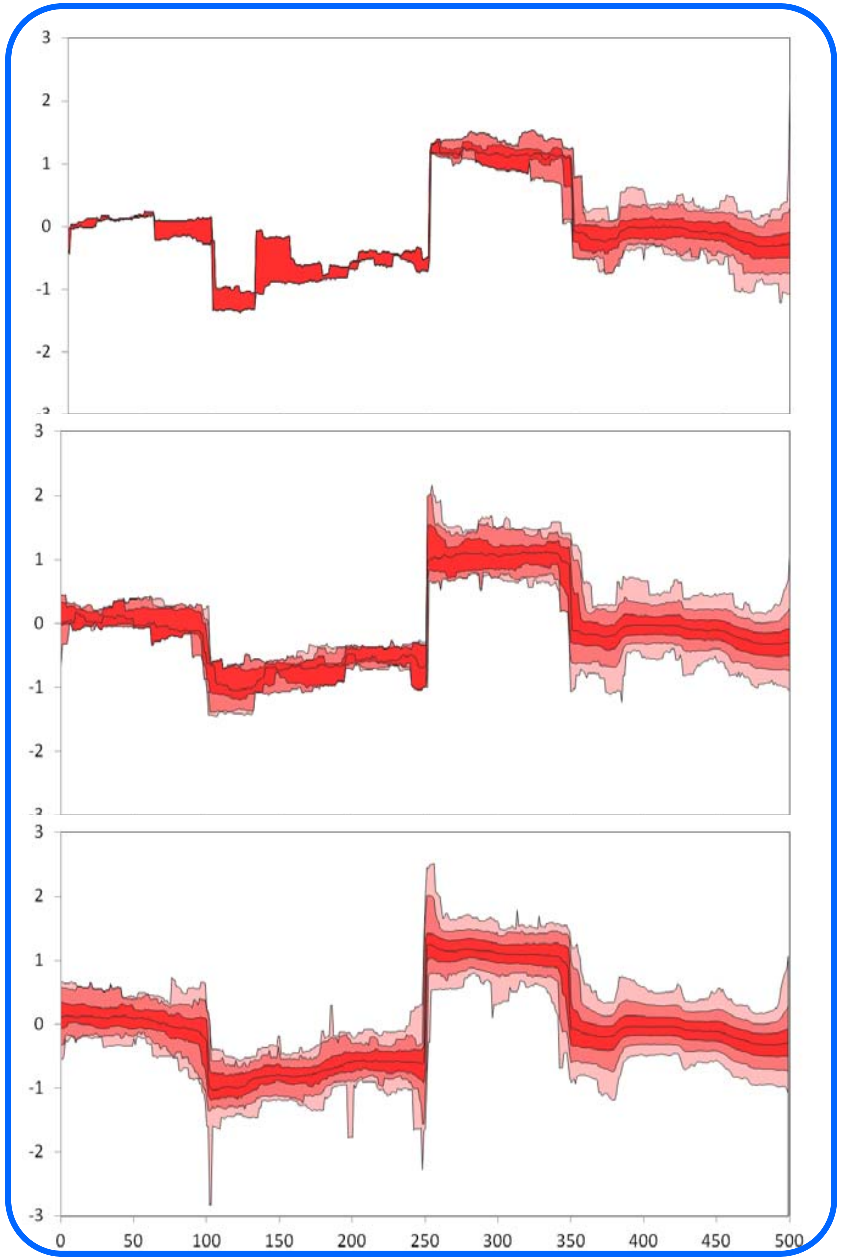
$$p(Y_N^N \mid x_N) = p(y_N \mid x_N)$$

後向きフィルタ

$$p(Y_N^{n+1} \mid x_n) = \int p(Y_N^{n+1} \mid x_{n+1}) p(x_{n+1} \mid x_n) dx_{n+1}$$

$$p(Y_N^n \mid x_n) = p(y_n \mid x_n) p(Y_N^{n+1} \mid x_n)$$

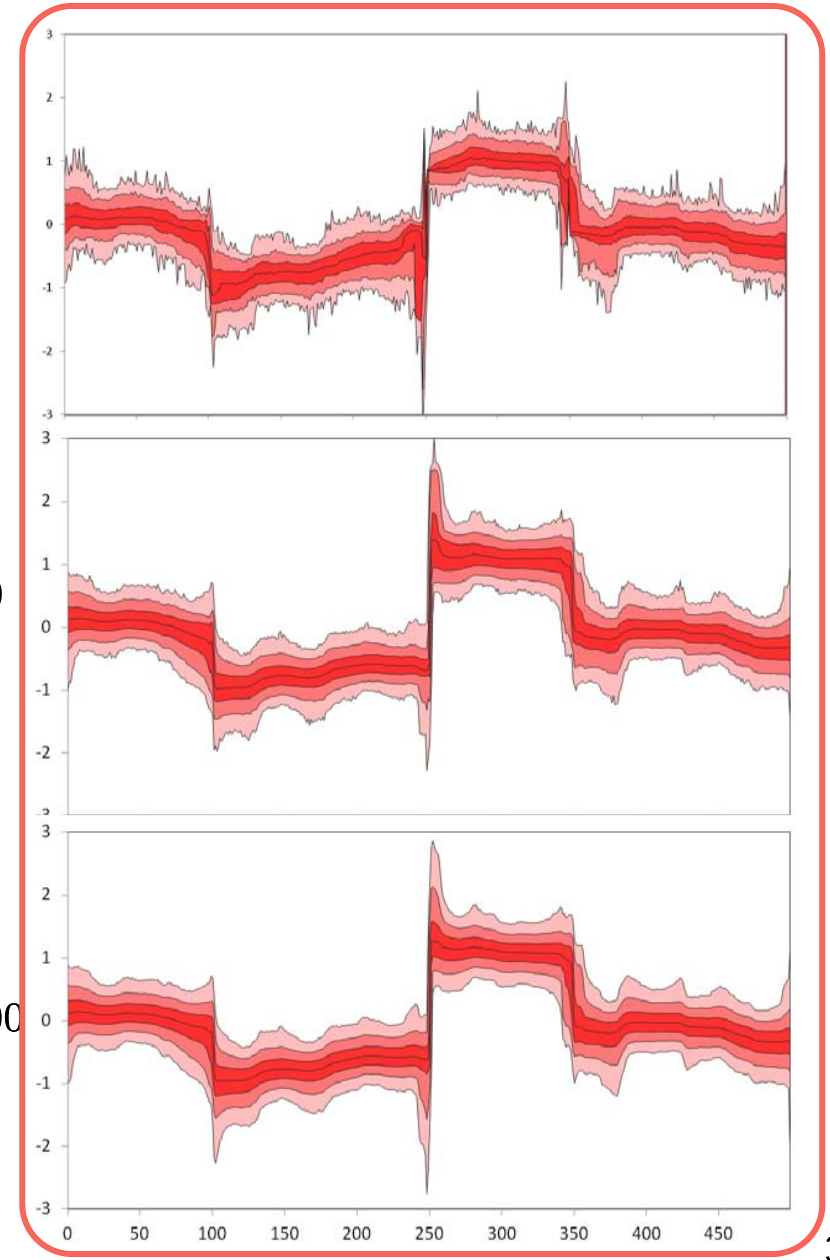
固定区間平滑化（LAG=500）　　　　２方向フィルタ公式



m=1,000

m=10,000

m=100,000

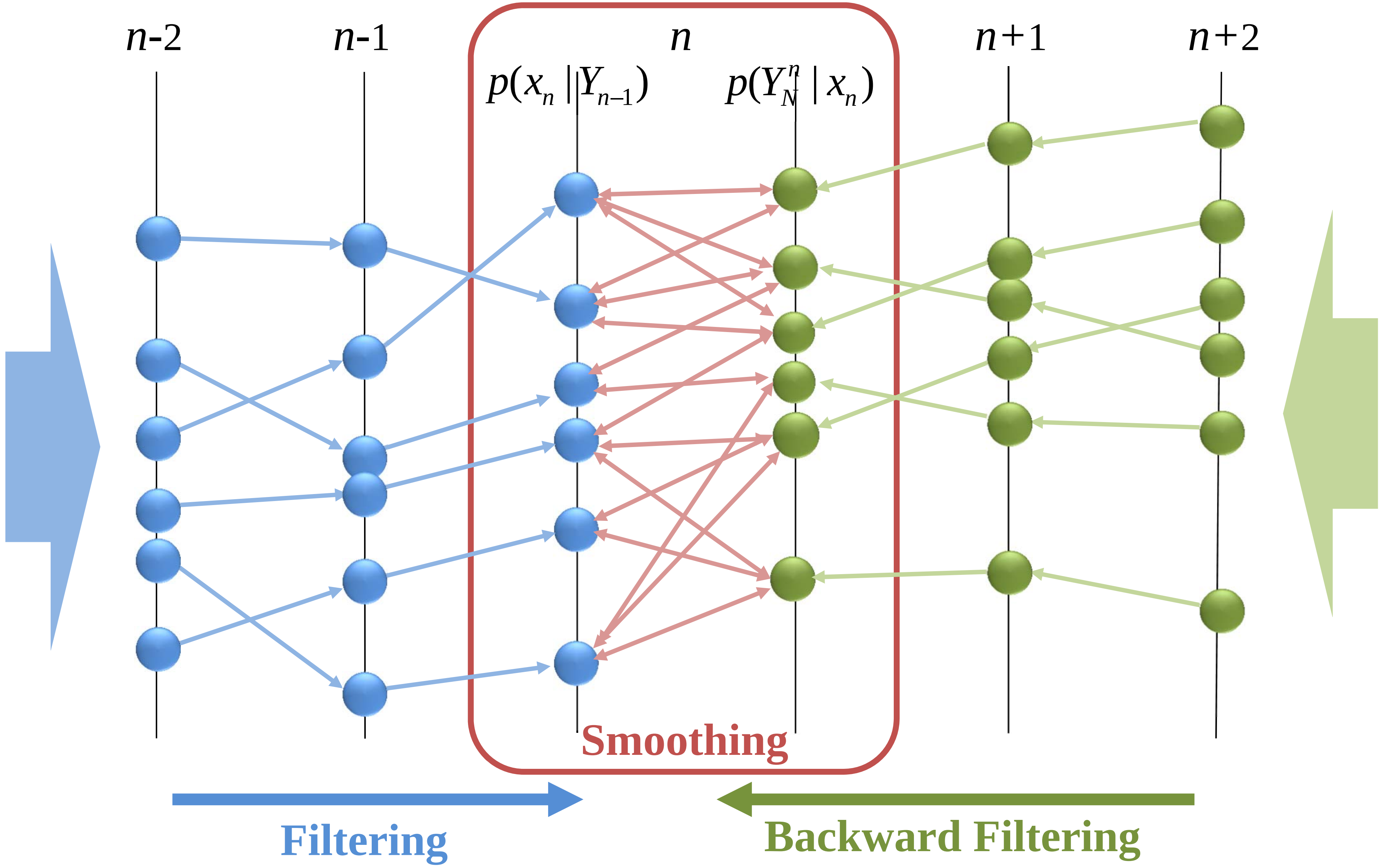


平滑化の精度

m	Gaussian Model			Cauchy Model		
	Fixed-lag	Fixed-interval	Two-filter	Fixed-lag	Fixed-interval	Two-filter
10^2	8.693	41.723	6.913	21.248	47.881	26.440
10^3	2.259	16.275	1.399	6.042	23.654	4.870
10^4	0.717	5.547	0.333	1.001	3.679	0.378
10^5	0.185	1.448	0.118	0.140	0.380	0.072

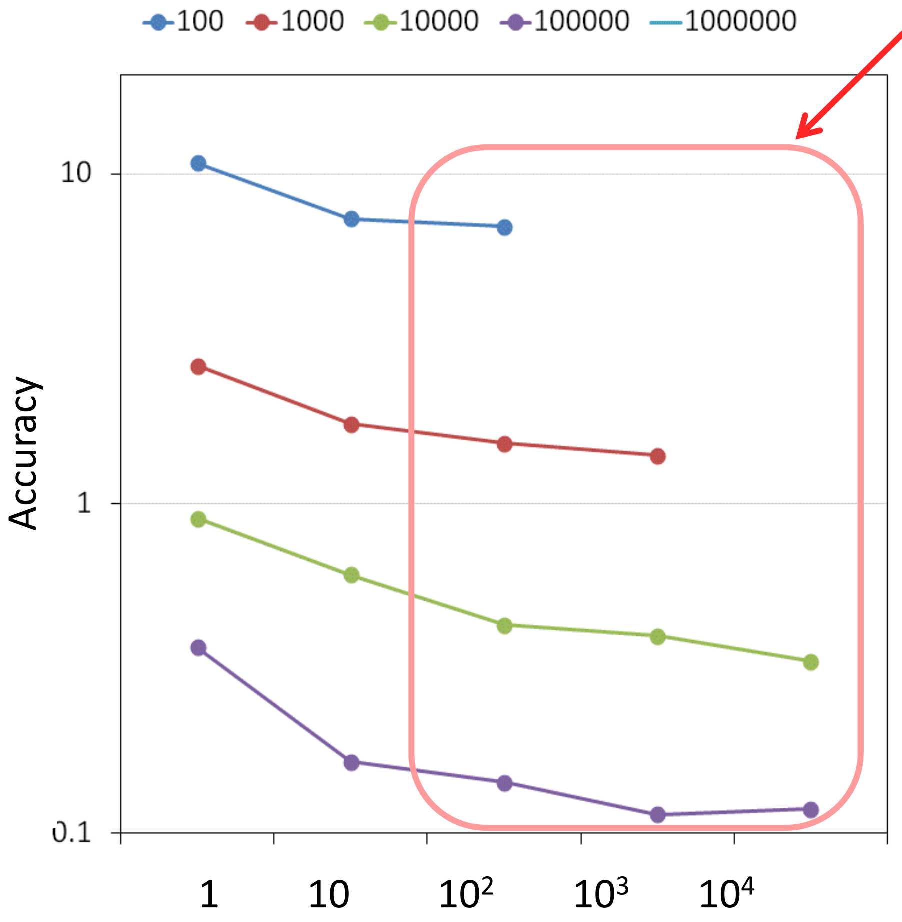
Evaluated by $I(D, \hat{D})$

２方向フィルタによる平滑化



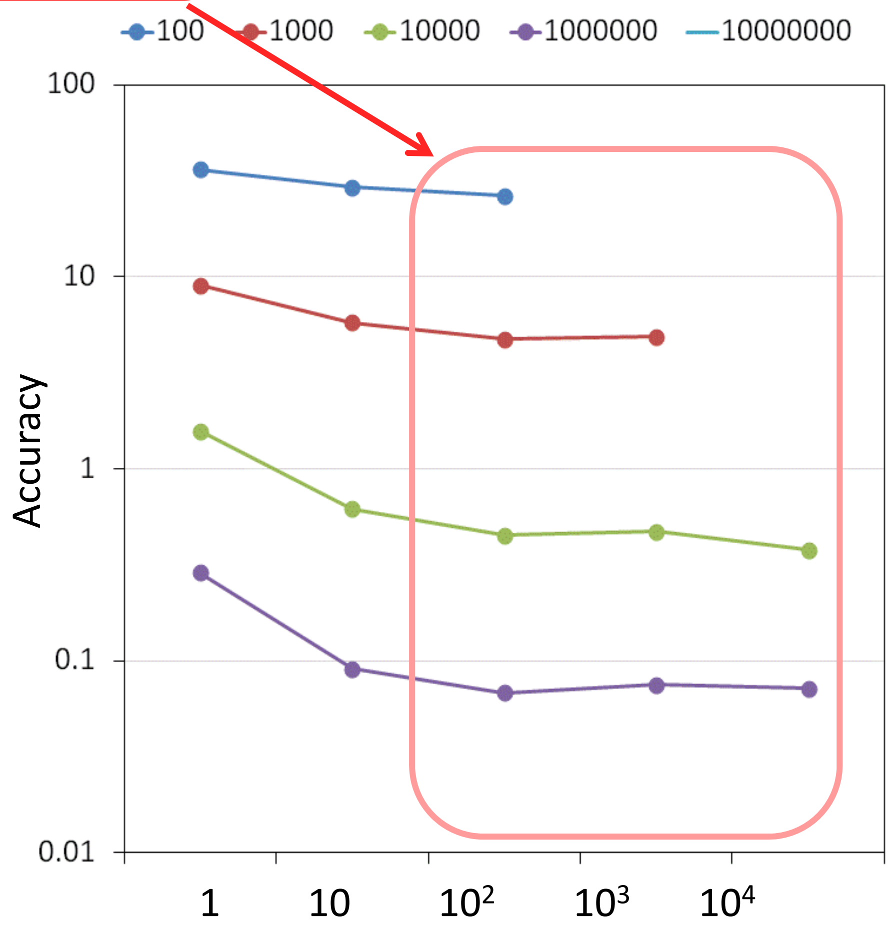
2方向フィルタ公式の精度

Gaussian model



m_s=100 で十分
m_s=10 でも妥当

Cauchy model



m_s: 各粒子に重み評価に使う粒子数

2 (方向) フィルタ公式に関するまとめ

- リサンプリングによる平滑化分布の崩壊は，２方向フィルタ公式によって，ほぼ回避できる．
- ただし，２方向フィルタ公式を厳密に適用すると，粒子数が大きな場合，大きな計算時間を要する．
- Thinning (粒子の評価に用いる粒子をサンプリングにより10-100個に限定する）により，精度をある程度維持しつつ計算時間を大幅に減少できる．

モンテカルロ計算の長所，欠点

長所
- 複雑なモデルでも実装が簡単（粒子ごとの代入計算は簡単で早い．分布の数値計算（畳み込み積分，非線形変換など）は面倒で時間がかかる）
- 状態の次元が増えても計算量はそれほど増えない

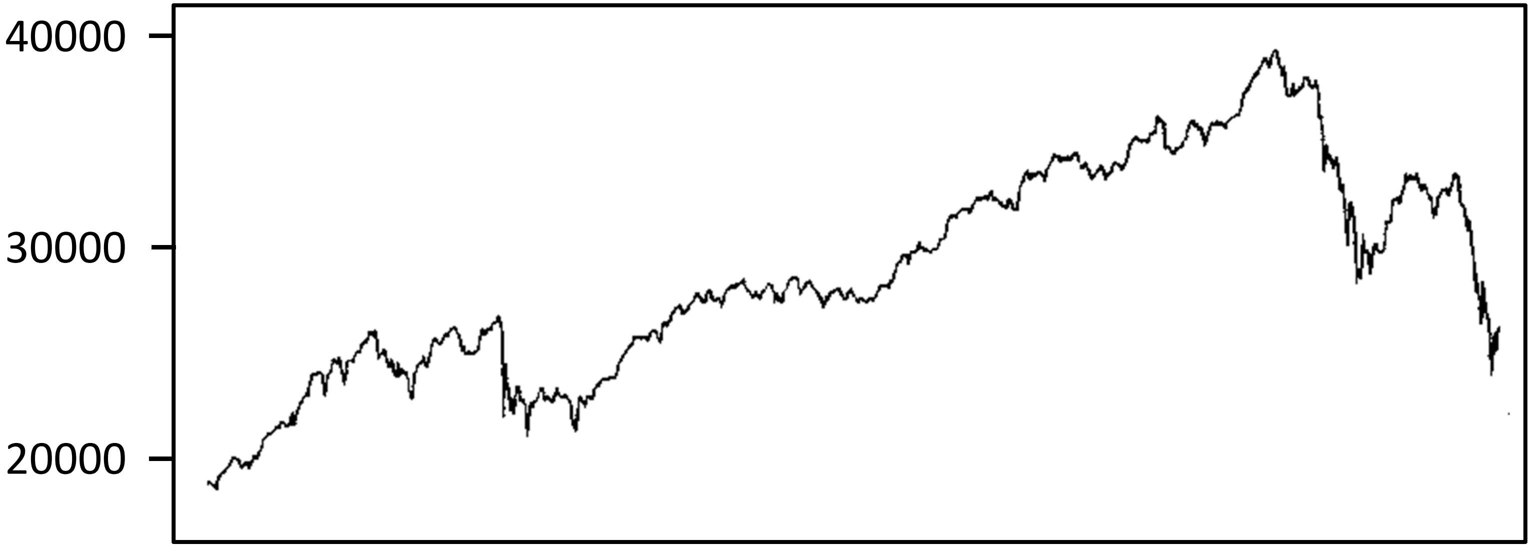
問題点
- 常にサンプリング誤差を伴う（パラメータ最適化は要注意）
- 高精度を求めると時間がかかる（粒子数 mを増やしても分散減少は $1/m$）

粒子フィルタの応用

1. 非ガウス型平滑化
    - ボラティリティモデル
    - 非ガウス型季節調整
2. 非線形平滑化
3. 自己組織型状態空間モデル
4. 計数データのモデリング
5. 高次元フィルタ（データ同化）

Gordon et al. (1993), Kitagawa (1996)
Doucet, de Freitas and Gordon (2001) “Sequential Monte Carlo Methods in Practice”

日経２２５平均株価



・平均値が変動
・分散が変動
・分散変動と平均値の変化に関連の可能性がある

トレンドとボラティリティの同時モデル化

観測モデル

$$y_n = t_n + \sigma_n w_n, \quad w_n \sim N(0,1)$$

トレンド成分モデル

$$t_n = 2t_{n-1} - t_{n-2} + v_n$$

分散変動成分モデル

$$\log \sigma_n^2 = 2\log \sigma_{n-1}^2 - \log \sigma_{n-2}^2 + u_n$$

トレンドと分散は独立

状態空間モデル

状態ベクトル

$$x_n = \begin{bmatrix} t_n \\ t_{n-1} \\ \log\sigma_n^2 \\ \log\sigma_{n-1}^2 \end{bmatrix}$$

状態空間モデル

$$x_n = Fx_{n-1} + Gv_n$$

$$y_n = H(x_n, w_n)$$

$$F = \begin{bmatrix} 2 & -1 & & \\ 1 & 0 & & \\ & & 2 & -1 \\ & & 1 & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$v_n = \begin{bmatrix} \varepsilon_n \\ \delta_n \end{bmatrix}$$

$$H(x_n, w_n) = t_n + \sigma_n w_n$$

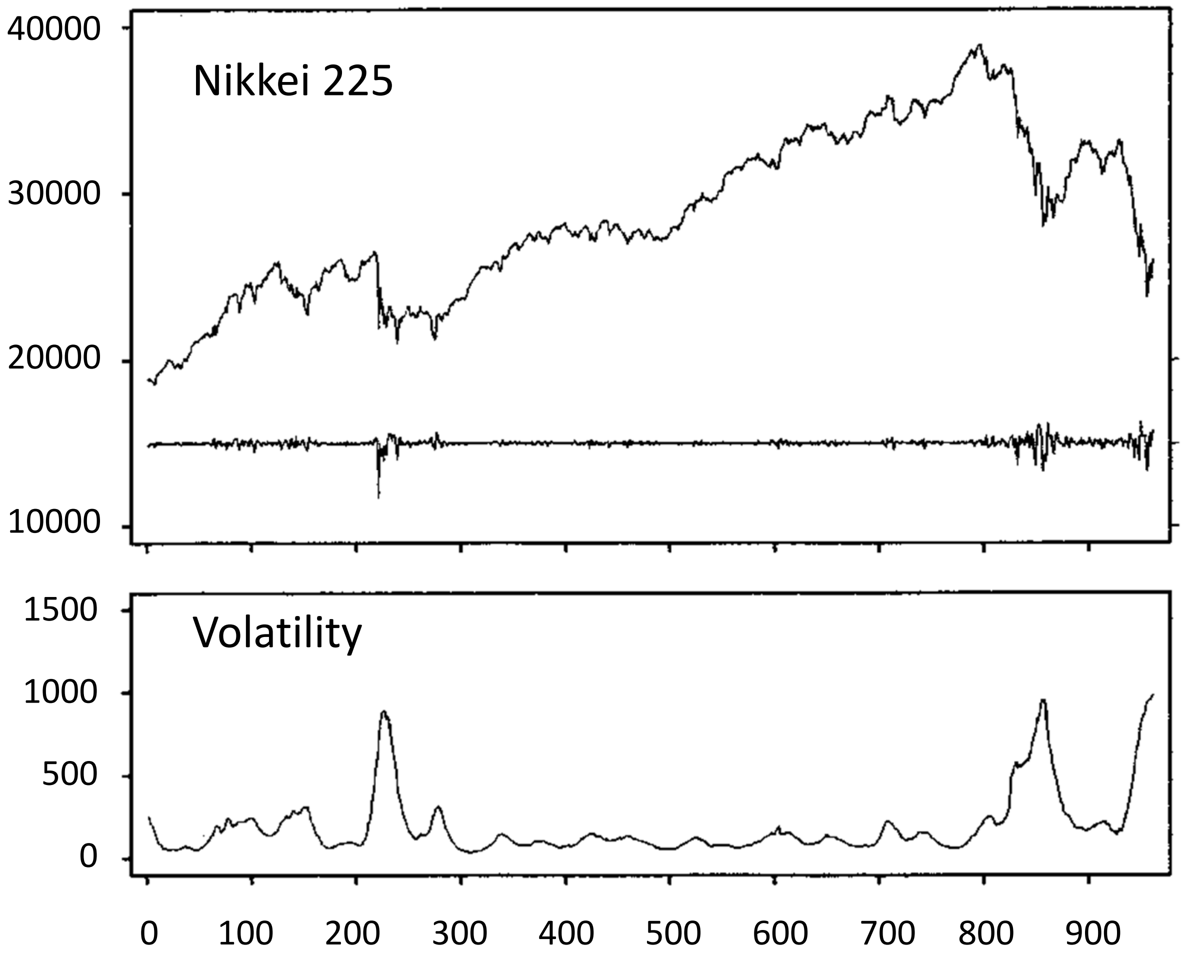
$$w_n = \frac{y_n - t_n}{\sigma_n}$$

トレンドとボラティリティの同時推定

Model(G,G) $\varepsilon_n \sim N(0,\tau_1^2)$ $\delta_n \sim N(0,\tau_2^2)$

Model(C,G) $\varepsilon_n \sim C(0,\tau_1^2)$ $\delta_n \sim N(0,\tau_2^2)$

Model	(G,G)	(C,G)
τ_1^2	9000	700
τ_2^2	0.0026	0.0005
AIC	13726.02	13765.84



確率的ボラティリティ（自己組織化）

オリジナルの状態空間モデル

$$x_n = \beta_n x_{n-1} + v_n \qquad v_n \sim N(0, \tau^2)$$

$$r_n = e^{\alpha_n + x_n} w_n$$

未知パラメータ ⇨ α, β, τ^2

同時推定のための状態空間モデル

$$\begin{bmatrix} x_n \\ \alpha_n \\ \beta_n \\ \log \tau_n^2 \end{bmatrix} = \begin{bmatrix} \beta x_{n-1} \\ \alpha_{n-1} \\ \beta_{n-1} \\ \log \tau_{n-1}^2 \end{bmatrix} + \begin{bmatrix} e^{\lambda_n/2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_n \\ \varepsilon_{n1} \\ \varepsilon_{n2} \\ \varepsilon_{n3} \end{bmatrix}$$

$$r_n \sim e^{\alpha_n + x_n} w_n$$

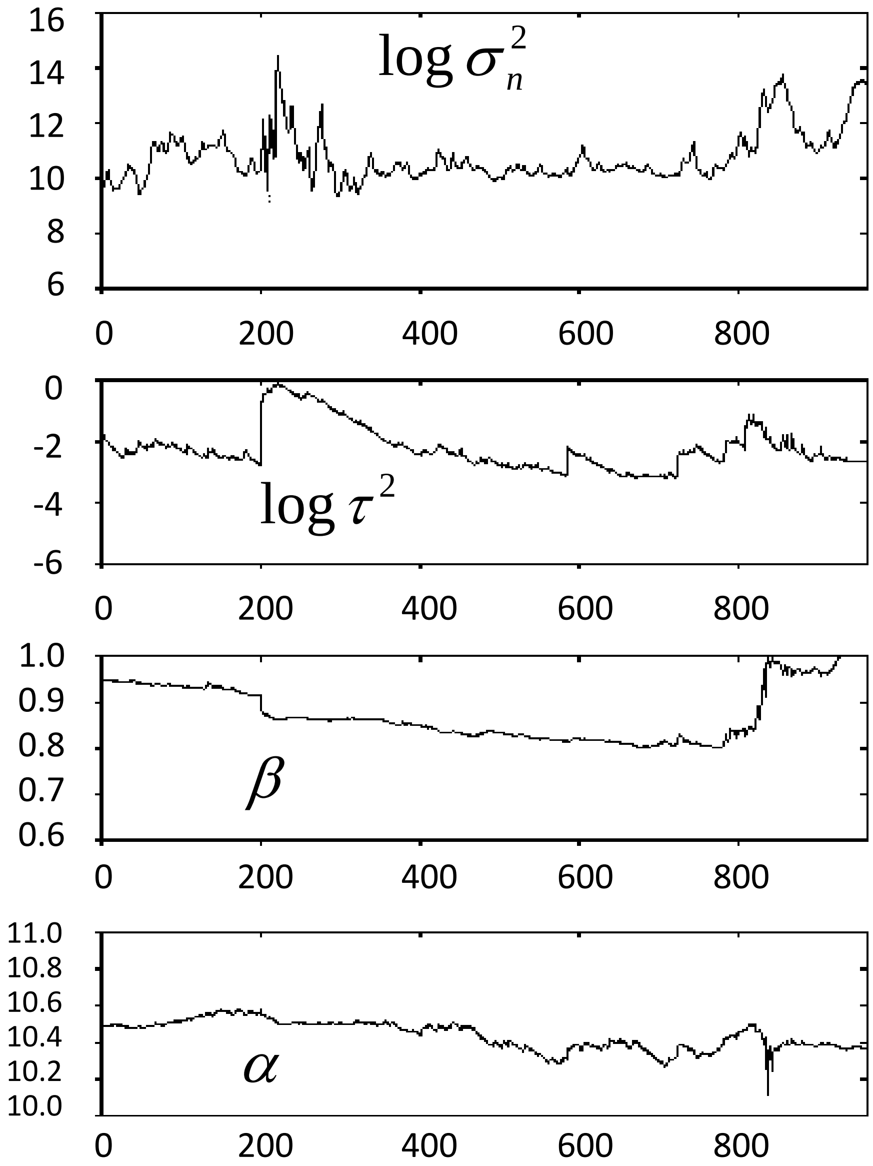
$$\lambda_n = \log \tau_n^2$$

粒子フィルタによる自己組織化

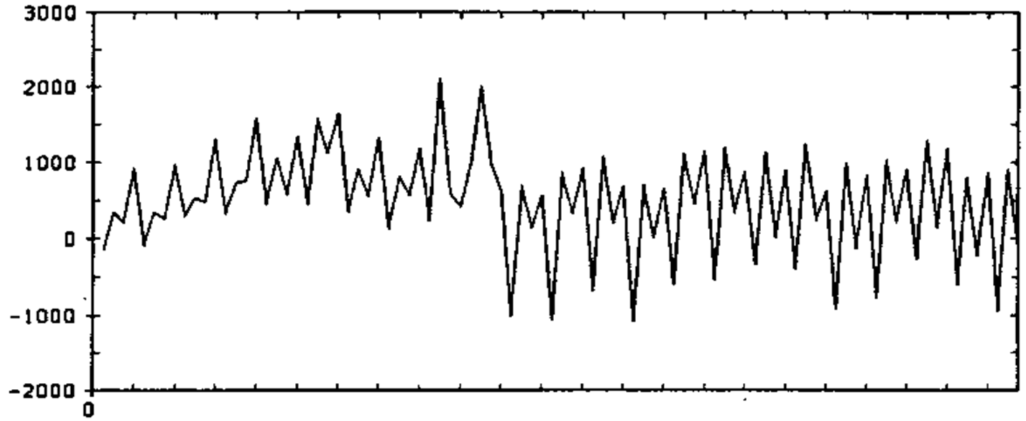
$$\theta_n = (\alpha_n, \beta_n, \log \tau_n^2)^T$$

$$\begin{bmatrix} x_n \\ \theta_n \end{bmatrix} = \begin{bmatrix} \beta_n x_{n-1} \\ \theta_{n-1} \end{bmatrix} + \begin{bmatrix} e^{\log \tau_n^2/2} & \\ & I_3 \end{bmatrix} \begin{bmatrix} v_n \\ \varepsilon_n \end{bmatrix}$$

$$r_n = e^{\alpha_n + x_n} w_n$$



季節調整：　構造変化が存在する場合



ある商品の在庫データ

$$y_n = T_n + S_n + w_n \qquad w_n \sim N(0, \sigma^2)$$

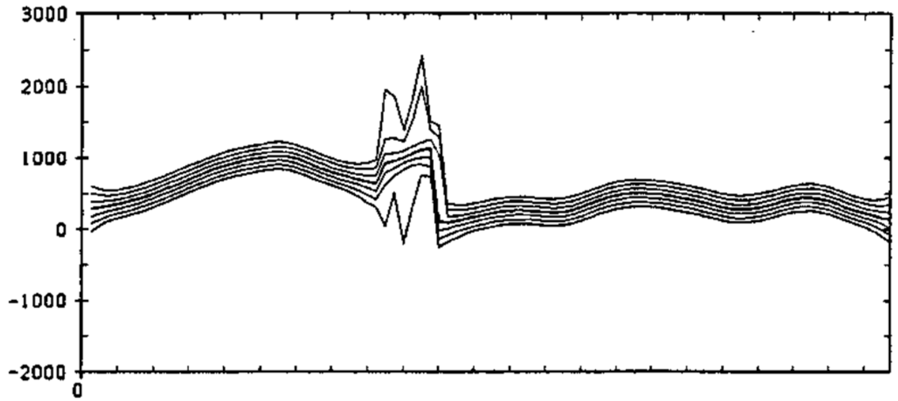
$$T_n = 2T_{n-1} - T_{n-2} + v_n \qquad v_n \sim \alpha N(0, \tau_1^2) + (1-\alpha) N(0, \zeta_1^2)$$

$$S_n = -(S_{n-1} + \cdots + S_{n-p+1}) + u_n \qquad u_n \sim \beta N(0, \tau_2^2) + (1-\beta) N(0, \zeta_2^2)$$

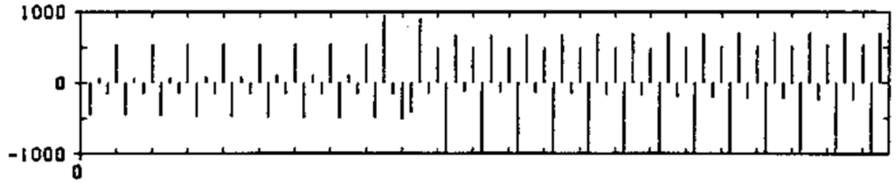
	$\hat{\sigma}^2$	$\hat{\tau}_1^2$	$\hat{\tau}_2^2$	$\hat{\alpha} = \hat{\beta}$	Log-LK	AIC
ガウス	0.874×10^5	0.146×10^6	0.365×10^4	1.00	-670.39	1346.8
混合ガウス	0.288×10^5	0.528×10^3	0.362×10^3	0.98	-667.01	1342.0

構造変化を伴う在庫データの季節調整結果

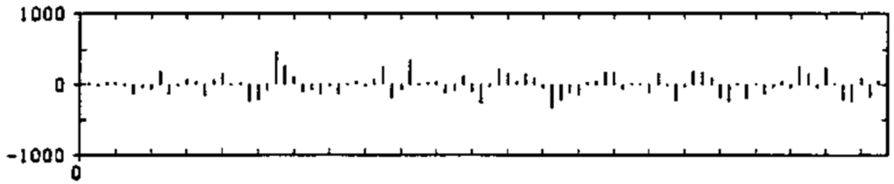
トレンド



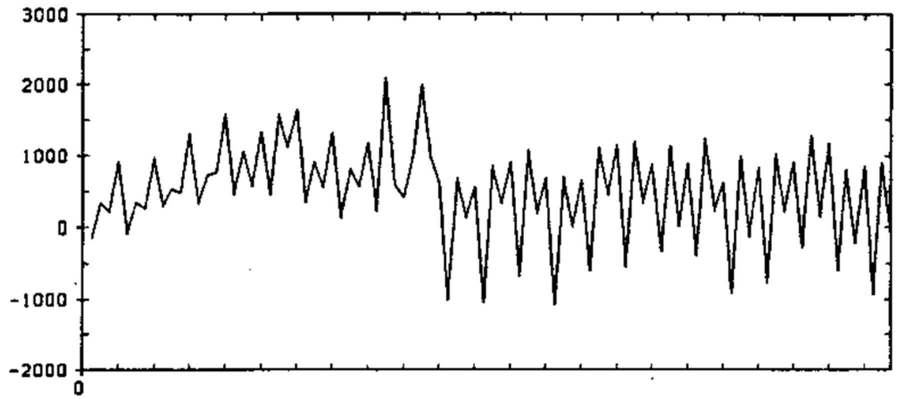
季節成分



ノイズ

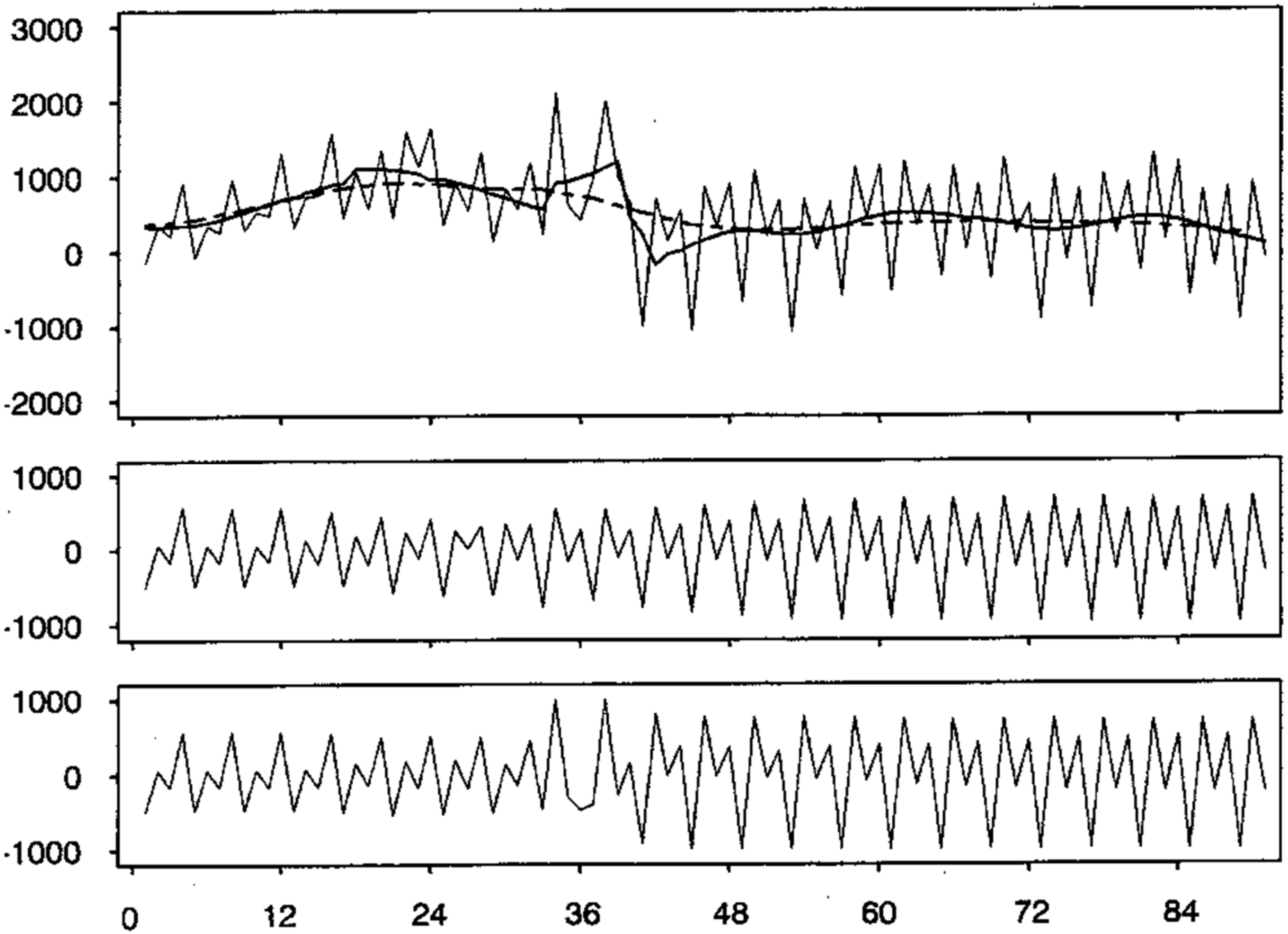


データ



季節調整：ガウスモデルと非ガウスモデルの比較

Inventory data



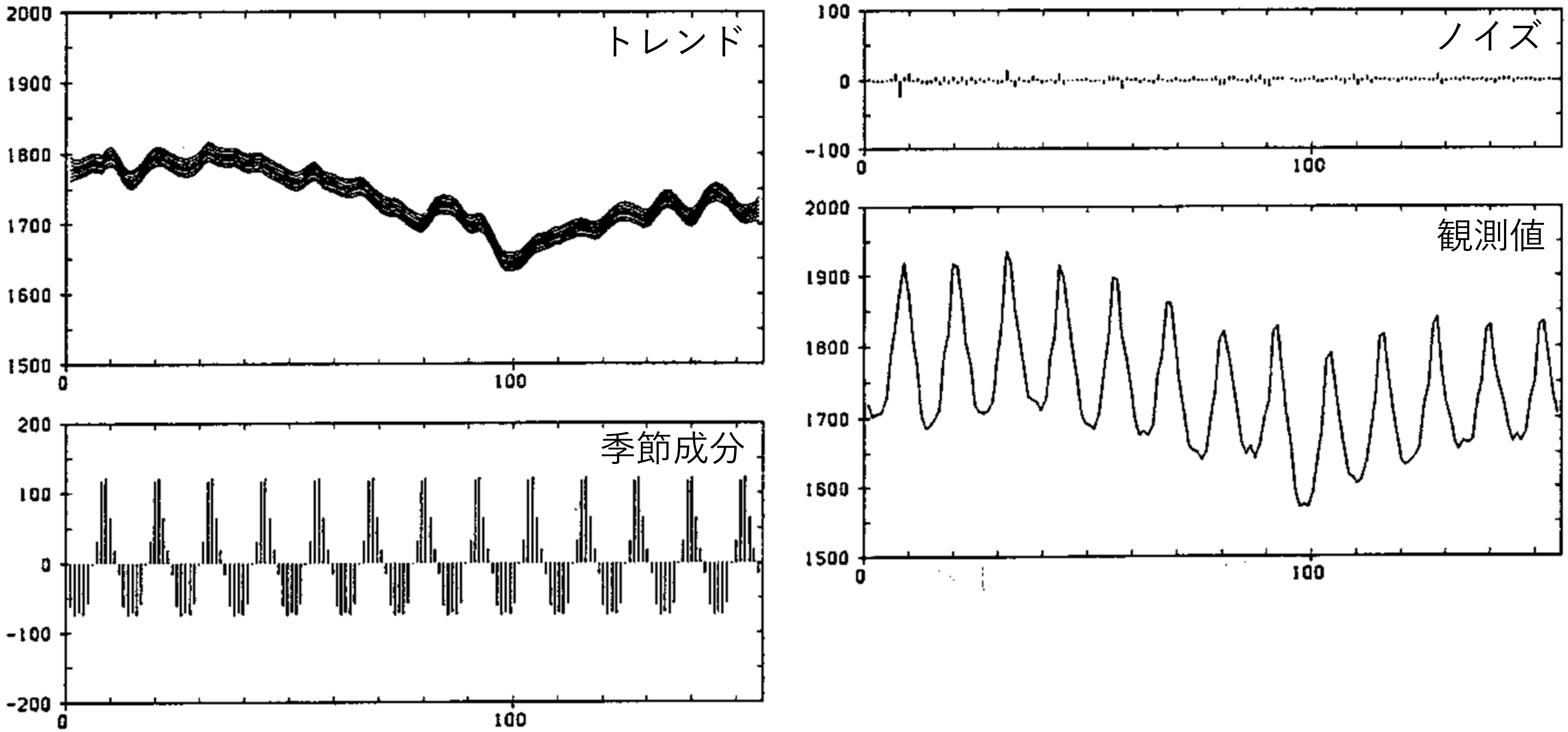
ガウスモデル

非ガウス型

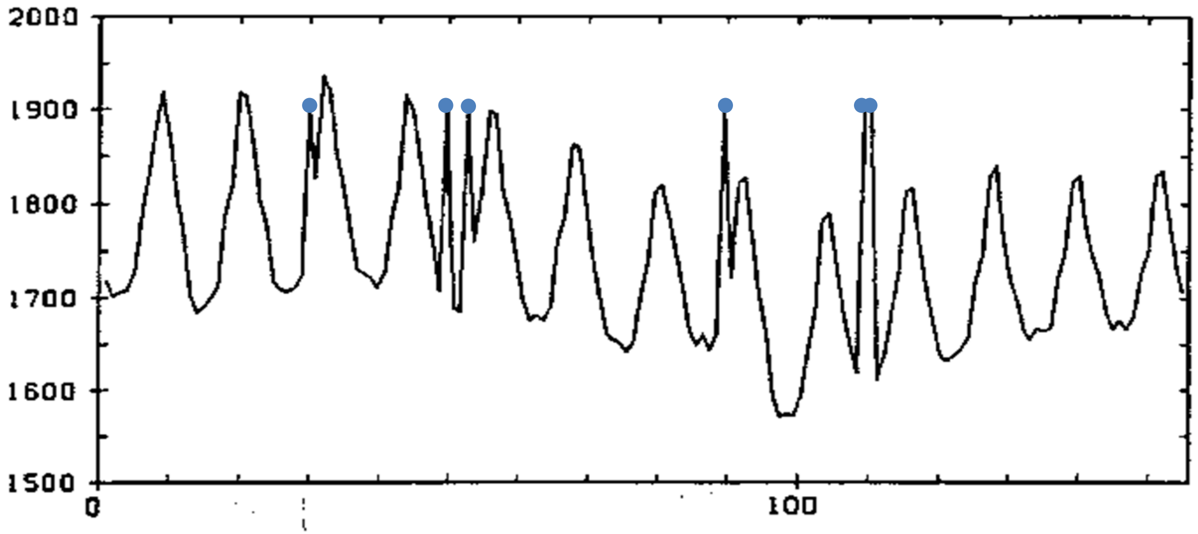
季節成分
ガウスモデル

季節成分
非ガウス型

BLSALLFOOD data



BLSALLFOOD dataに人工的に異常値を設定

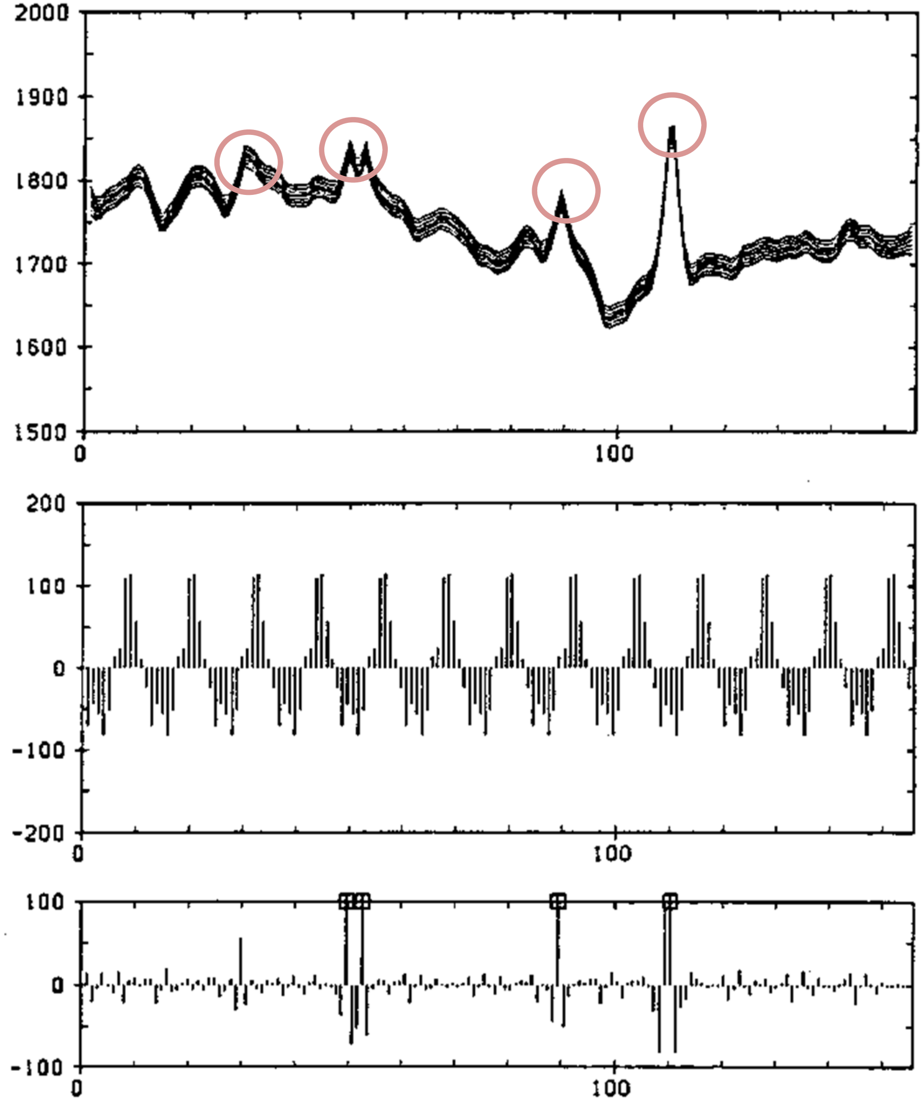


$$q(w) \sim \alpha N(0, \sigma^2) + (1 - \alpha) N(\mu, \tau^2)$$

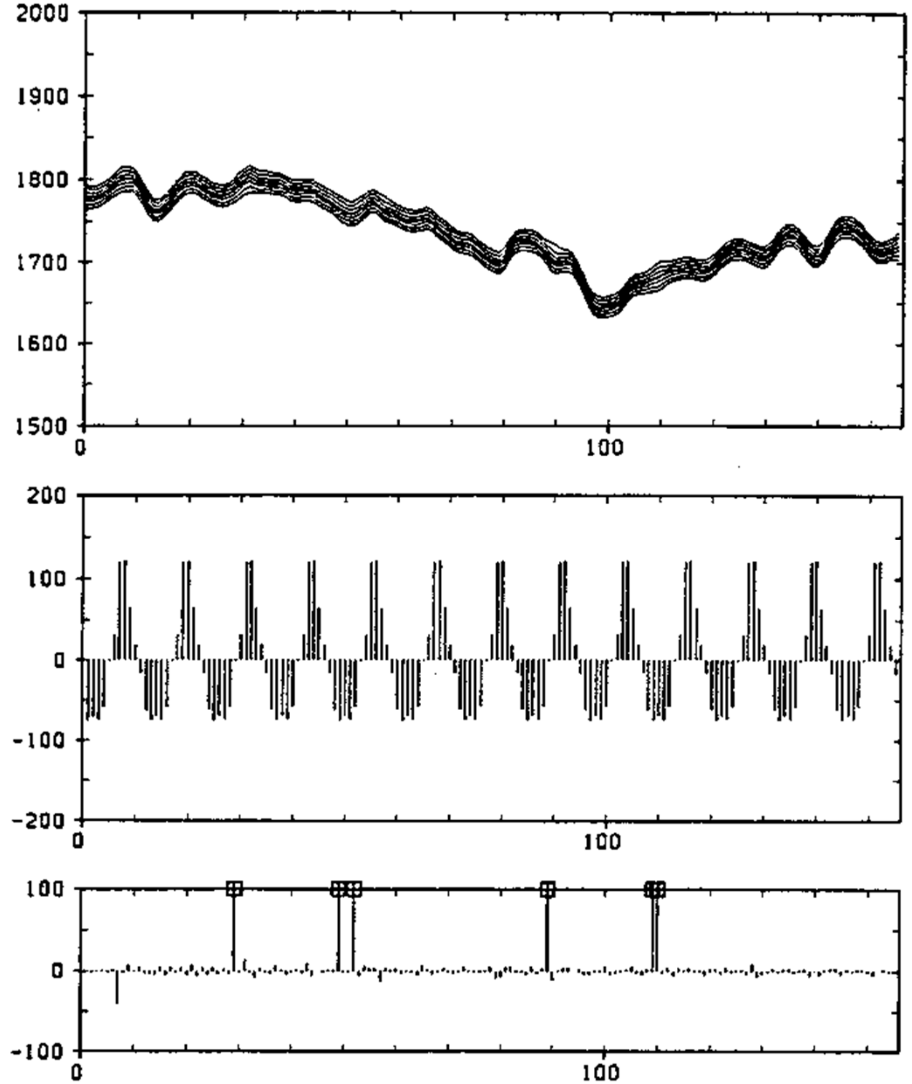
α	σ^2	τ^2	AIC
1.00	32.9	–	5596.4
0.96	30.3	4×10^4	–690.2

季節調整結果

ガウスモデル



混合ガウスモデル

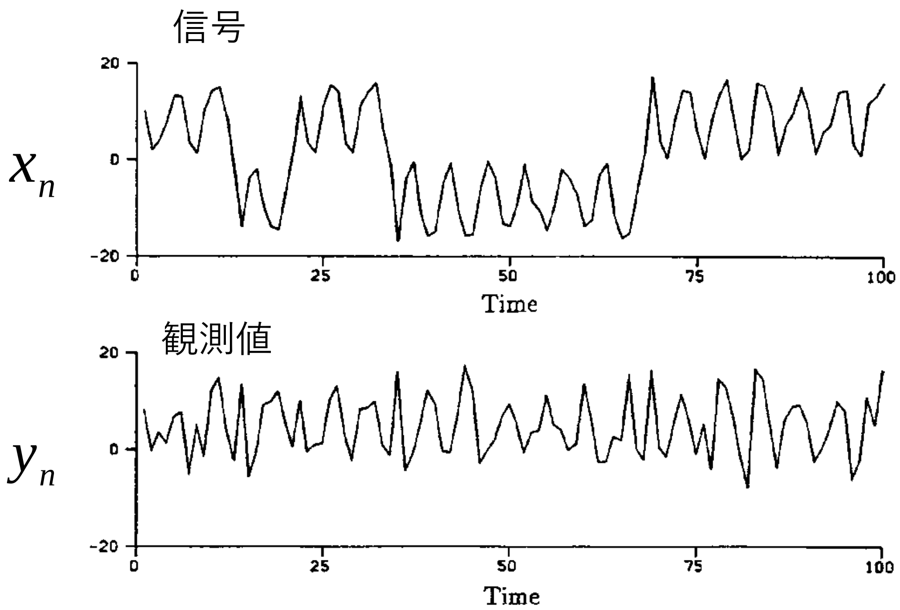


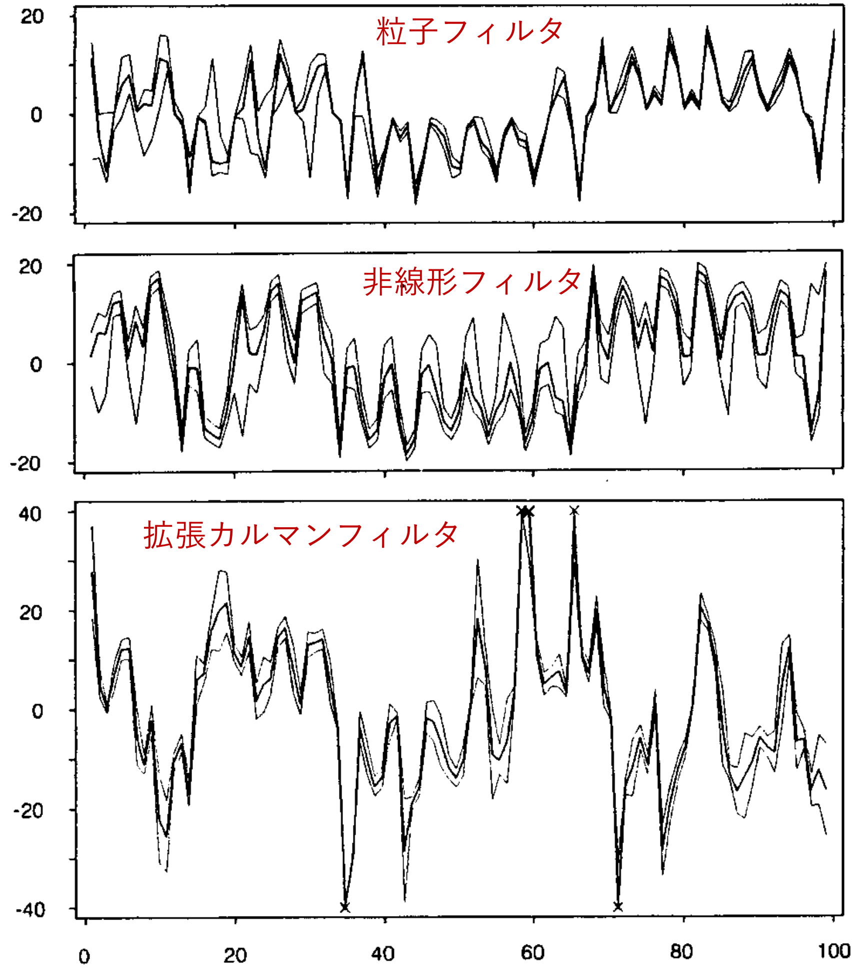
非線形平滑化

$$x_n = \frac{1}{2}x_{n-1} + \frac{25x_{n-1}}{1+x_{n-1}^2} + 8\cos(1.2n) + v_n$$

$$y_n = \frac{x_n^2}{20} + w_n$$

$$v_n \sim N(0,1), \quad w_n \sim N(0,10)$$





非線形モデル（自己組織化）

$$x_n = \theta_{n3} x_{n-1} + \frac{\theta_{n4} x_{n-1}}{1 + x_{n-1}^2} + \theta_{n5} \cos(1.2n) + v_n$$

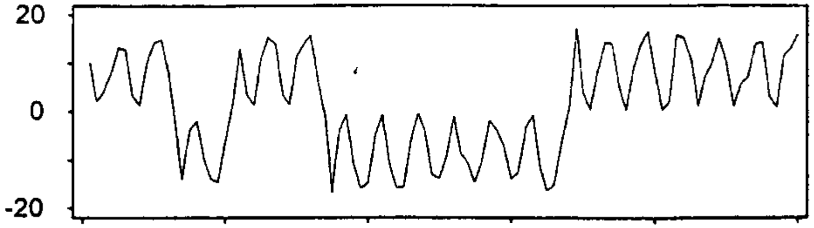
$$y_n = \theta_{n6} x_n^2 + w_n$$

$$v_n \sim N(0, \exp\{\theta_{n1}\}),\ \ w_n \sim N(0, \exp\{\theta_{n2}\})$$

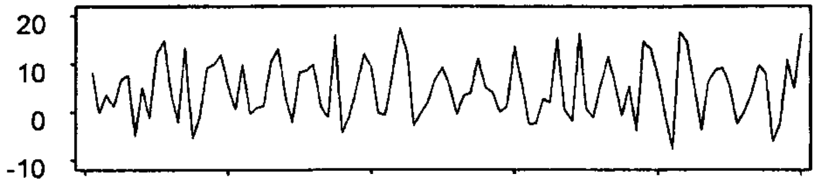
拡大した状態ベクトル

$$z_n = \begin{bmatrix} x_n \\ \theta_{n1} \\ \vdots \\ \theta_{n6} \end{bmatrix}$$

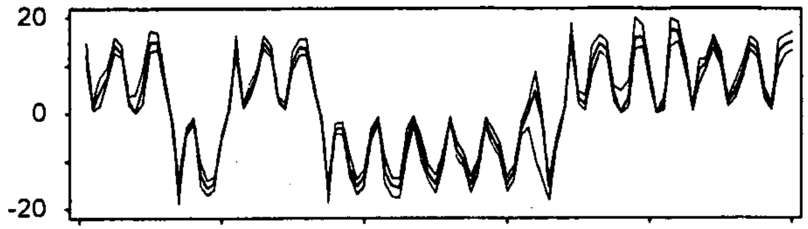
自己組織型平滑化



信号



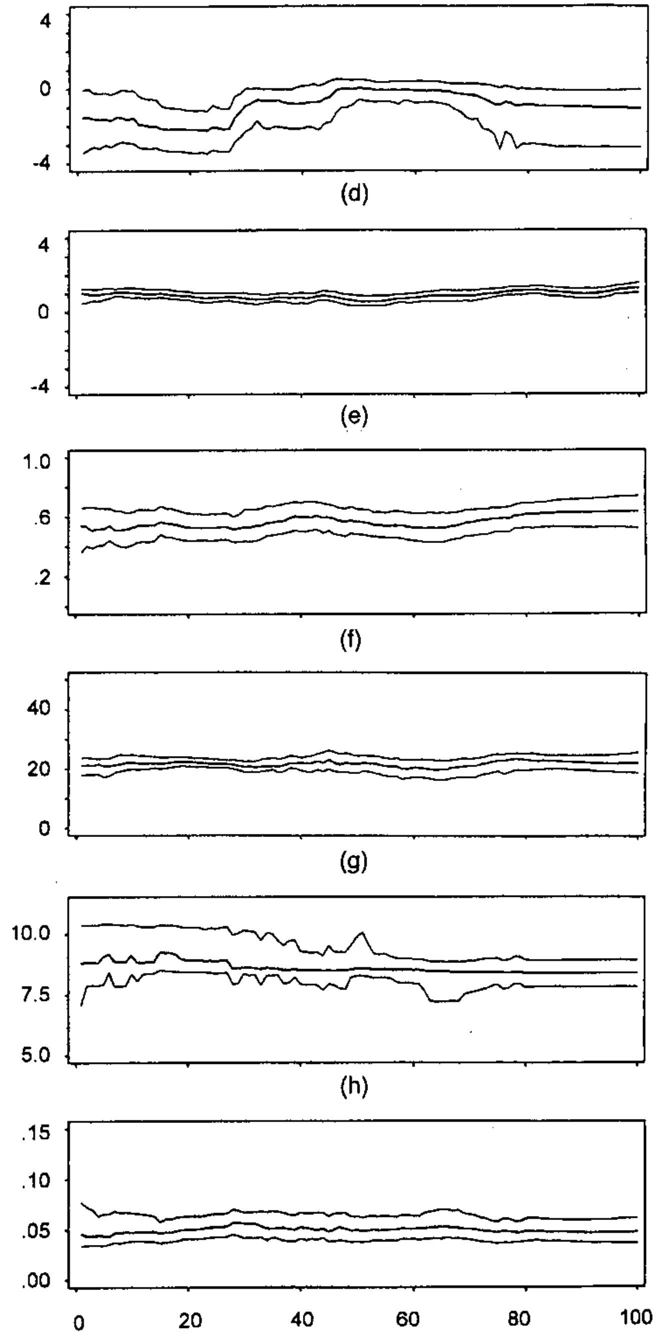
観測値



信号の推定

$$x_n = \theta_{n3} x_{n-1} + \frac{\theta_{n4} x_{n-1}}{1 + x_{n-1}^2} + \theta_{n5} \cos(1.2n) + v_n$$

$$y_n = \theta_{n6} x_n^2 + w_n \qquad v_n \sim N(0, \exp\{\theta_{n1}\}),\ w_n \sim N(0, \exp\{\theta_{n2}\})$$



θ_{n1}

θ_{n2}

θ_{n3}

θ_{n4}

θ_{n5}

θ_{n6}

計数データの季節調整

$$\log t_n = 2\log t_{n-1} - \log t_{n-2} + v_n$$

$$\log s_n = -(\log s_{n-1} + \cdots + \log s_{n-11}) + u_n$$

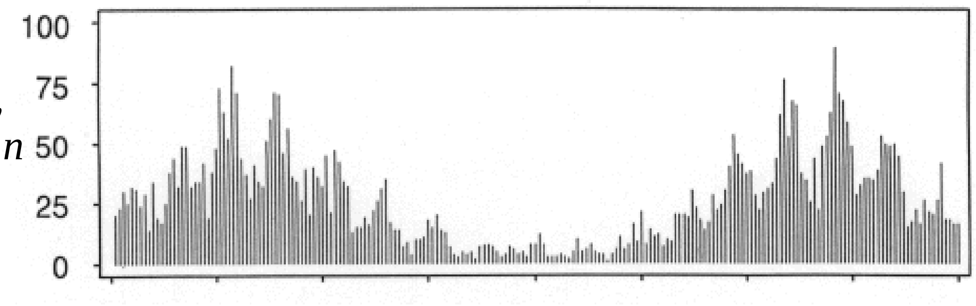
$$x_n = \begin{bmatrix} t_n \\ t_{n-1} \\ s_n \\ \vdots \\ s_{n-10} \end{bmatrix}$$

$$\lambda_n = t_n \times s_n$$

$$P(y_n \mid \lambda_n) = \frac{e^{-\lambda_n} \lambda_n^{y_n}}{y_n}$$

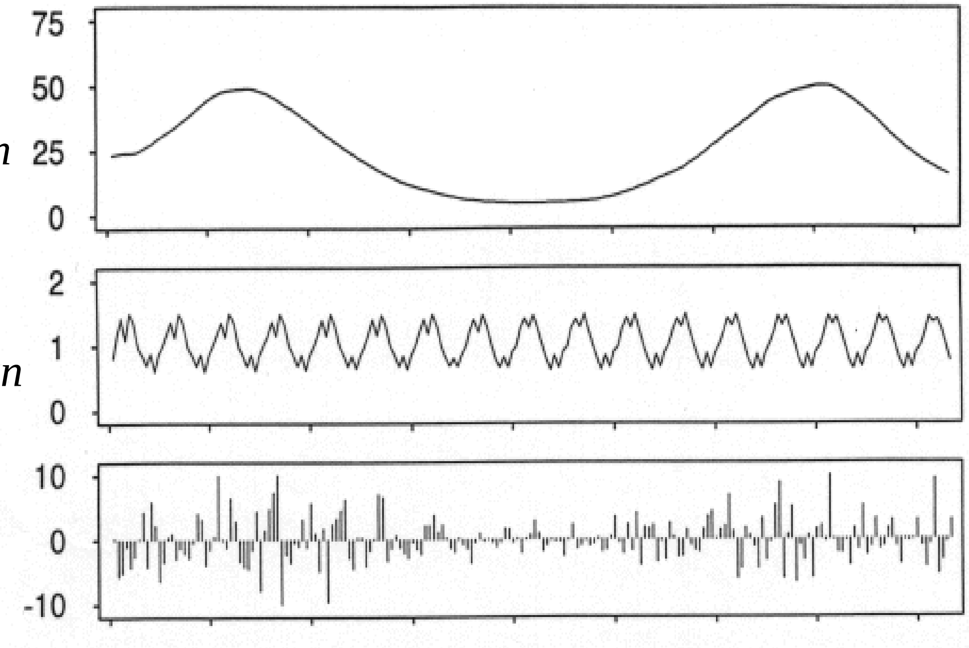
ポアソン分布

y_n



t_n

s_n



データ同化

シミュレーション
モデル

＋

観測データ

データ同化 → 高度な
シミュレーション

データ同化 ＝ 状態（x_t ）推定問題

$$x_t = F_t(x_{t-1}, v_t) \quad \text{（システムモデル）}$$

$$y_t = H_t x_t + w_t \quad \text{（観測モデル）}$$

x_t ：シミュレーションモデルの全変数
y_t ：全観測変数
v_t ：システムモデルの不確実性
w_t ：観測モデル

超高次元

x_t：$10^4 \sim 10^6$
y_t：$10^2 \sim 10^5$

オリジナルな研究分野
- 海洋学，気象学

適用領域
- 津波
- 大地震発生時の微気圧変動
- 細胞内流動
- 生体分子動態解析
- 流体制御

粒子フィルタ
アンサンブルカルマンフィルタ

アンサンブルカルマンフィルタ

- 状態ベクトル，時系列を多数の粒子で表現
- 状態ベクトル，時系列の共分散行列をアンサンブル平均で計算
- カルマンゲインを数値的に計算

アンサンブルカルマンフィルタ

- 初期化 $f_{0|0}^{(j)} \sim N(x_0, V_0),$
- 予測

$$v_n^{(j)} \sim N(0, Q_n),$$
$$p_{n|n-1}^{(j)} \sim F(x_{n-1|n-1}) + v_n^{(j)},$$

- フィルタ

$$E_{n|n-1} = \left[\varepsilon_{n|n-1}^{(1)}, \quad \cdots \quad , \varepsilon_{n|n-1}^{(m)} \right]$$

$$\varepsilon_{n|n-1}^{(j)} = x_{n|n-1}^{(j)} - \frac{1}{m} \sum_{j=1}^{m} x_{n|n-1}^{(j)},$$

$$Y_{n|n-1} = \left[y_{n|n-1}^{(1)}, \quad \cdots \quad , y_{n|n-1}^{(m)} \right]$$

$$w_n^{(j)} \sim N(0, R_n),$$
$$y_{n|n-1}^{(j)} = H(x_{n|n-1}^{(j)}) + w_n^{(j)},$$

$$V_{n|n-1} = \frac{1}{m-1} Y_{n|n-1} Y_{n|n-1}^T$$

$$U_{n|n-1} = \frac{1}{m-1} E_{n|n-1} Y_{n|n-1}^T$$

$$K_n = U_{n|n-1} V_{n|n-1}^{-1}$$

$$x_{n|n}^{(j)} = x_{n|n-1}^{(j)} + K_n \left(y_n - y_n^{(j)} \right)$$

$$x_{n|n} = \frac{1}{m} \sum_{j=1}^{m} x_{n|n}^{(j)}$$

フィルタのまとめ

	システムモデル	観測モデル	ノイズ	計算量	計算精度
カルマン	線形	線形	ガウス	◎	◎
拡張カルマン	非線形	非線形	ガウス	◎	△
非ガウス	非線形	非線形	非ガウス	×	◎
ガウス和	線形	線形	非ガウス	○	○
粒子	非線形	非線形	非ガウス	△	○
アンサンブルカルマン	非線形	線形	ガウス	◎	△

シミュレーション

時系列モデルのノイズ項に想定した分布の乱数を代入することにより，簡単にシミュレーションが実施できる.

- ARモデル　$y_n = \sum_{j=1}^{m} A_j y_{n-j} + v_n$

  初期値 $y_1, \ldots, y_m$と乱数$v_{m+1}, \ldots, v_N \sim N(0, \sigma^2)$

- 状態空間モデル

$$
\begin{aligned}
x_n &= F x_{n-1} + G v_n \\
y_n &= H x_n + w_n
\end{aligned}
$$

初期値 x_0と乱数$v_1, \ldots, v_N \sim N(0, Q), w_1, \ldots, w_N \sim N(0, R)$

繰り返し生成してアンサンブルとして見る