【ベイズ推定（スカラー） Bayesian Estimation -scalar- 】

モデル $z = As + w$ とする．
信号 $s : \mathcal{N}(\mu_s, \sigma_s^2)$，ノイズ $w : \mathcal{N}(0, \sigma_w^2)$ は, 正規分布に従うとする．
事前分布 $f(s)$，尤度 $f(z|s)$，事後分布 $f(s|z)$ は，次のように書ける．

$$f(s) = \frac{1}{\sqrt{2\pi\sigma_s^2}}\exp\left\{-\frac{(s-\mu_s)^2}{2\sigma_s^2}\right\}$$
$$f(z|s) = \frac{1}{\sqrt{2\pi\sigma_w^2}}\exp\left\{-\frac{(z-As)^2}{2\sigma_w^2}\right\}$$
$$f(s|z) = \frac{1}{\sqrt{2\pi\omega^2}}\exp\left\{-\frac{(s-\bar{s})^2}{2\omega^2}\right\}$$

まず，事後分布の指数部は，

$$-\frac{1}{2\omega^2}s^2 + \frac{1}{\omega^2}\bar{s}\cdot s + \mathrm{Const} - (*)$$

である．ここで，ベイズの定理，$f(s|z) \propto f(z|s)\cdot f(s)$ より，尤度，事前分布の指数部の和を求める．

$$-\frac{(z-As)^2}{2\sigma_w^2} - \frac{(s-\mu_s)^2}{2\sigma_s^2} = -\frac{1}{2}\left(\frac{A^2}{\sigma_w^2}+\frac{1}{\sigma_s^2}\right)s^2 + \left(\frac{Az}{\sigma_w^2}+\frac{\mu_s}{\sigma_s^2}\right)s + \mathrm{Const} - (*2)$$

$(*)$, $(*2)$ の s についての次数恒等条件より，次の推定値を得る．

$$\bar{s} = \frac{Az\sigma_s^2 + \mu_s\sigma_w^2}{A^2\sigma_s^2 + \sigma_w^2}$$

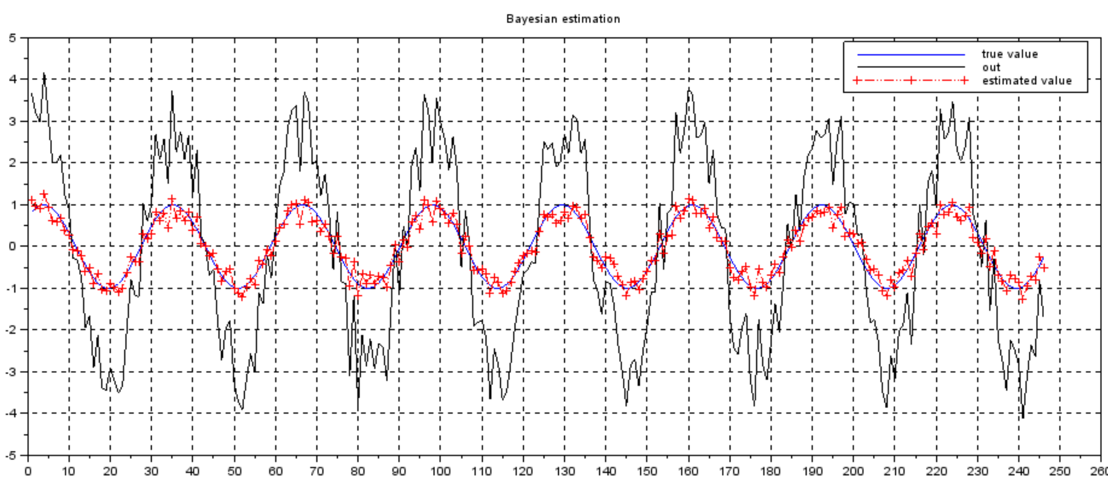

Figure 1: Scilab 実行結果

Source Code 1: Scilab

```
//////////////////////////////////////
//      ベイズ推定  -スカラー-
//      Bayesian estimation  scalar
//
//                        M.Tsutsui
///////////////////////////////////////

clear all;

Data=50; //データ数
s=sin(1:0.2:Data); //信号
```

```
Data_renew=size(s,2);//列数カウント
w=1.5*rand(1,Data_renew)+(-1.5)*rand(1,Data_renew);//ノイズ
A=3;//係数
z=A*s+w;//出力

E_s=mean(s);//信号S平均
E_w=0;//ノイズw平均

var_s=(s-E_s*ones(1,Data_renew))*(s-E_s*ones(1,Data_renew))';//信号_分散
var_w=(w-E_w*ones(1,Data_renew))*(w-E_w*ones(1,Data_renew))';//ノイズ_分散

s_hat=(A*z*var_s+E_s*var_w)/(A^2*var_s+var_w);//推定値

plot(s);//真値
plot(z,'k');//出力値
plot(s_hat,'r+:');//推定値
xgrid();
hl=legend(['true value';'out';'estimated value';]);
title('Bayesian estimation');
```

Source Code 2: Python

```
#-----------------------------------------
# Module Name:Bayesian Estimation
# Author: m_tsutsui
#-----------------------------------------

#Library_Import##############
from numpy import*
from scipy import*

import math, numpy as np
import matplotlib.pyplot as plt
#end########################

Data=50 #Data Number
sita=linspace(0,Data,Data*5)
s=matrix(sin(sita)) #Signal

[Data_renew_p,Data_renew]=s.shape #size_check

A=5 #coefficient

w=matrix(np.random.rand(1,Data_renew))+(-1)*matrix(np.random.rand(1,Data_renew))     #Noise

z=A*s+w

#_Expectation_########
E_s=np.mean(s)
E_w=0

#_Variance_##########
ones_block=matrix(np.ones((1,Data_renew)))
var_s=(s-E_s*ones_block)*(s-E_s*ones_block).T
var_w=(w-E_w*ones_block)*(w-E_w*ones_block).T

#_Estimated_value_######
s_hat=(A*z.T*var_s+E_s*var_w)/((A**2)*var_s+var_w)

#_Plot_command_###########################
plt.figure(facecolor='w')#Backgroundcolor_white
plt.plot(s.T)
plt.plot(array(z.T))
plt.plot(array(s_hat),"r--")
```

```
plt.grid()#grid on
plt.legend(('True value','Output value','Estimated value'))
plt.title('Bayesian Estimation')
plt.show()
#_end_#####################################
```