Poisson DGLM with MCMC Inference

Meini Tang

2022-10-19

```
knitr::opts_chunk$set(echo = TRUE,
                      error = TRUE, # print error but do not stop on errors
                      message = FALSE,
                      out.width = "70%",
                      fig.align = "center")

rm(list=ls())
library(Rcpp)
library(ggplot2)
```

```
sourceCpp("/Users/meinitang/Repository/pois_dglm_inference/lbe_poisson.cpp")
sourceCpp("/Users/meinitang/Repository/pois_dglm_inference/mcmc_disturbance_poisson.cpp")
sourceCpp("/Users/meinitang/Repository/pois_dglm_inference/pl_poisson.cpp")
source("/Users/meinitang/Repository/pois_dglm_inference/lbe_pois_utils.R")
```

TODO

- Smoothing with LBE
- [Done] Is LBE correct? - Filtering is working
- [Done] LBE with transfer function
- Particle Filtering

1. No Transfer Function

Simulated Data

```
n = 200       number of observations
rho = 0.9    state transition / evolution coefficient, the "G",
Q = 0.1     Evolution error variance.
E0 = 0         Initial states
```

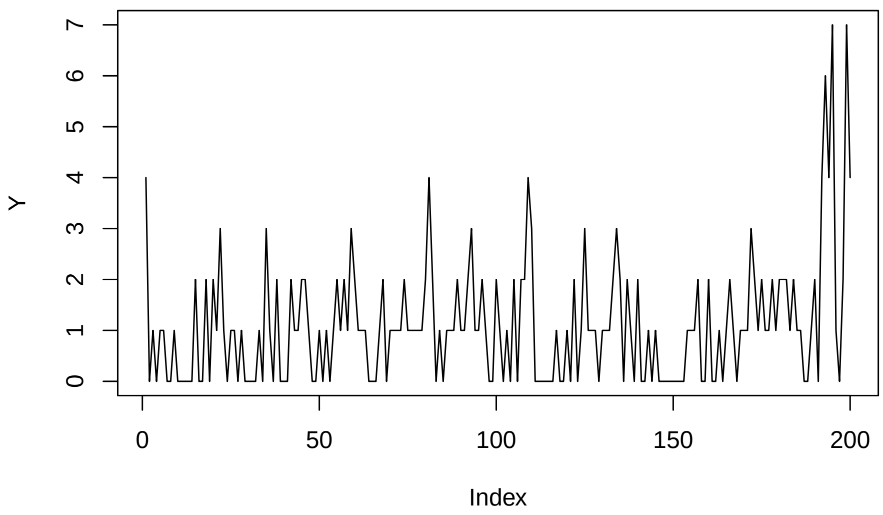
```
v = rnorm(n,mean=0,sd=sqrt(Q))    (w_1, ··· , w_n). realizations of evolution errors,
Fx = update_Fx0(n,rho)           Transition matrix between (θ_1,···,θ_n) and (w_1,···, w_n).
E = Fx%*% as.matrix(v,ncol=1)                                      "E"                "v".
Y = rpois(n,lambda=exp(E))
plot(Y,type="l")
```

Model.

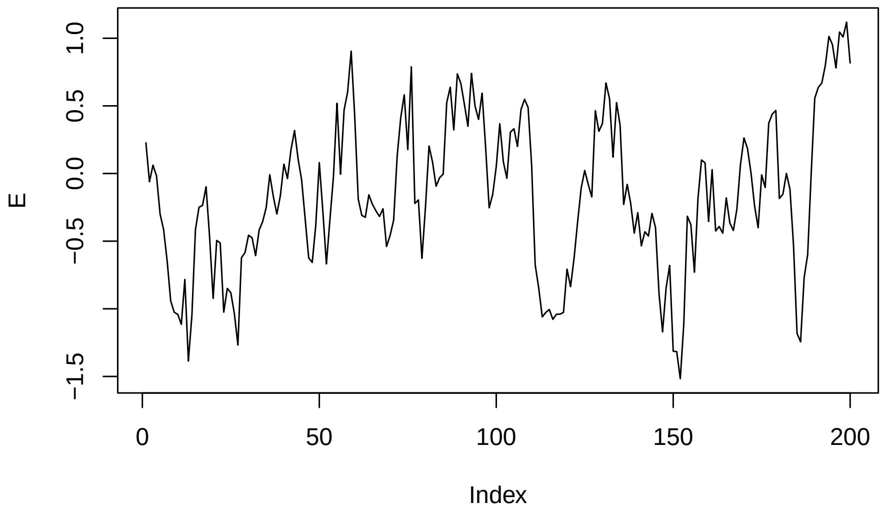
$$y_t = \text{Pois}(\exp(\theta_t)).$$

$$\theta_t = \rho\theta_{t-1} + w_t, \quad w_t \sim N(0, W).$$

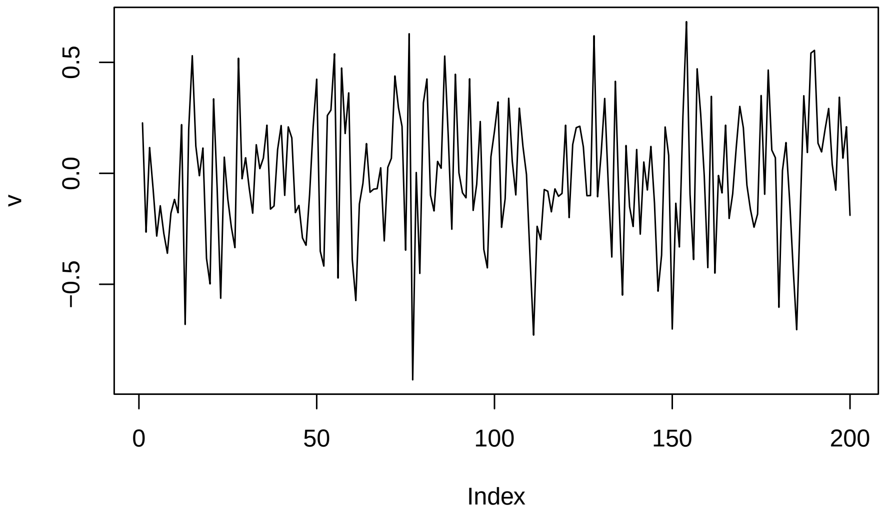
$$\begin{pmatrix} \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} = \begin{pmatrix} 1 & & & 0 \\ \rho & \ddots & & \\ \vdots & \ddots & \ddots & \\ \rho^{n-1} & \cdots & \rho & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}.$$



```
plot(E,type="l")
```

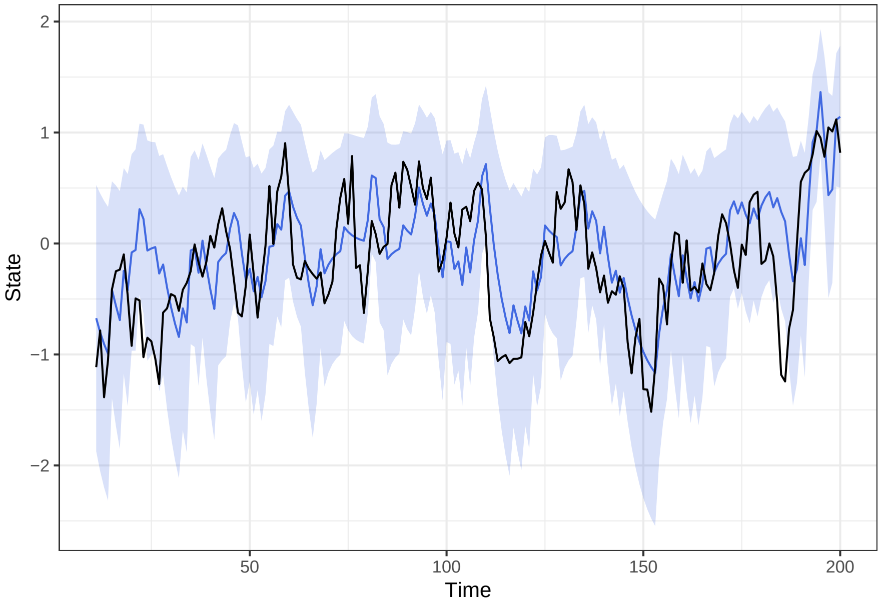


```
plot(v,type="l")
```

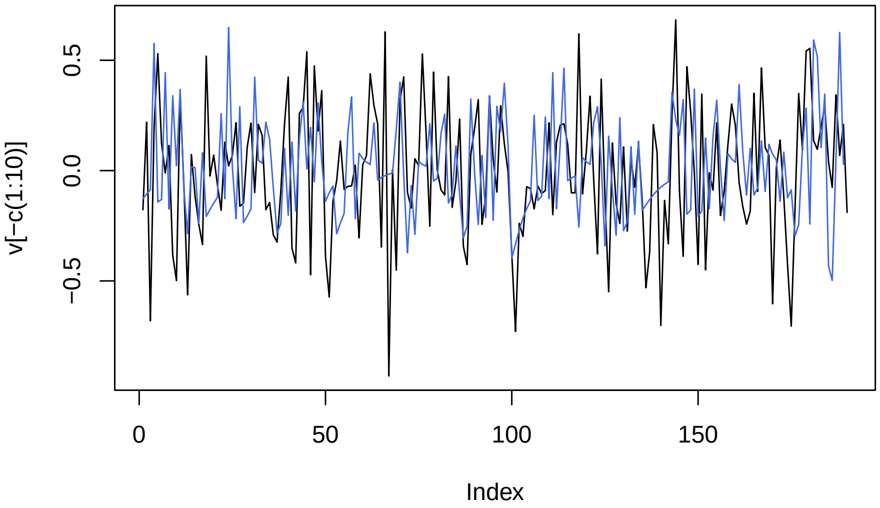


Linear Bayes Filtering

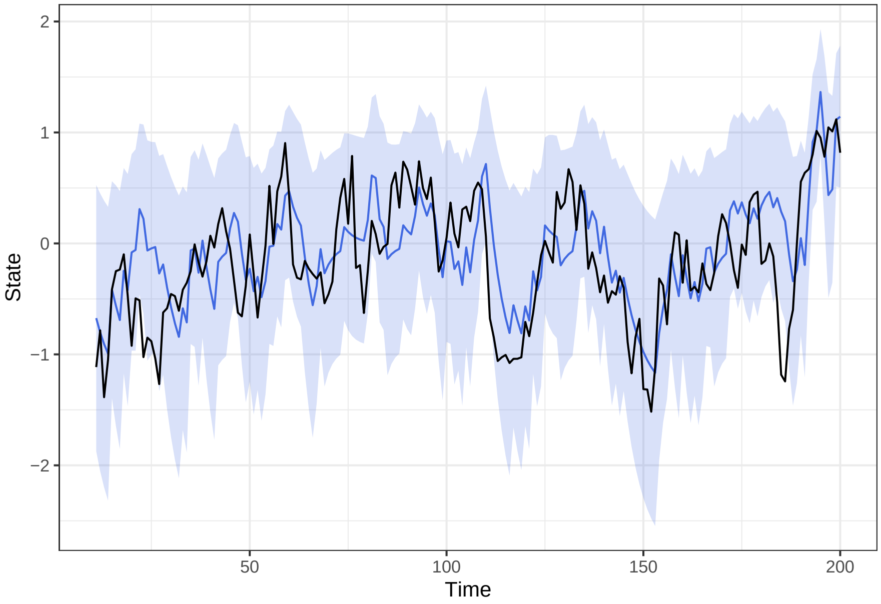
```
m0 = 0
C0 = 1e5
output = lbe_poisson0(Y,rho,Q,m0,C0)

tmp = data.frame(time=11:n, true=E[-c(1:10)],y=Y[-c(1:10)],
                 mt=c(output$mt[-c(1:11)]),
                 mt_lo=c(output$mt[-c(1:11)])-2*sqrt(c(output$Ct[-c(1:11)])),
                 mt_hi=c(output$mt[-c(1:11)])+2*sqrt(c(output$Ct[-c(1:11)])))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("State")
```



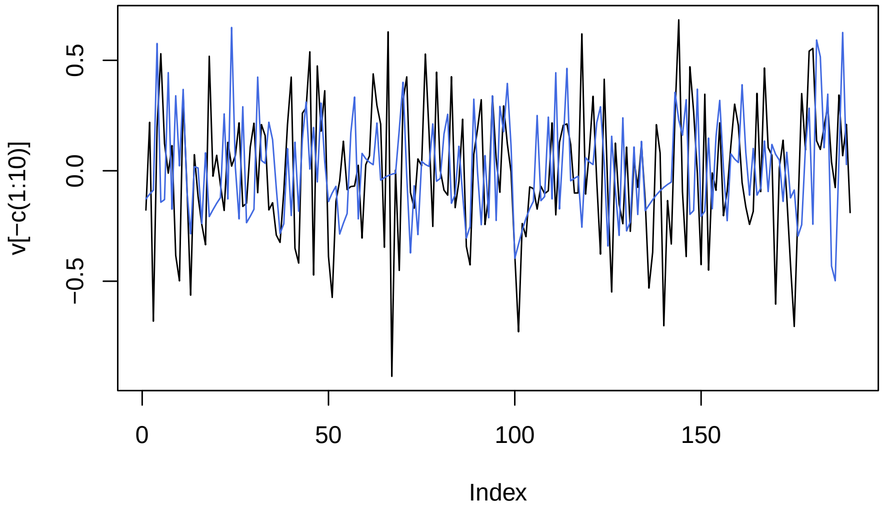
```
plot(v[-c(1:10)],type="l")
lines(diff(tmp$mt),col="royalblue")
```



```
Rw = 1e5
output = lbe_poisson1(Y,rho,Q,E0,Rw)

tmp = data.frame(time=11:n, true=E[-c(1:10)],y=Y[-c(1:10)],
                 mt=c(output$mt[-c(1:11)]),
                 mt_lo=c(output$mt[-c(1:11)])-2*sqrt(c(output$Ct[-c(1:11)])),
                 mt_hi=c(output$mt[-c(1:11)])+2*sqrt(c(output$Ct[-c(1:11)])))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("State")
```



```
plot(v[-c(1:10)],type="l")
lines(diff(tmp$mt),col="royalblue")
```

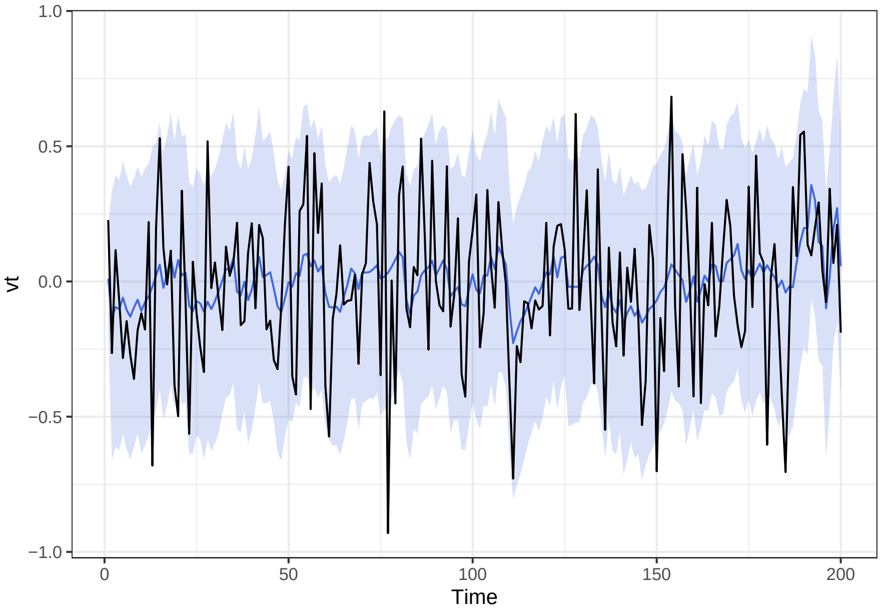


MCMC reparameterized with disturbance

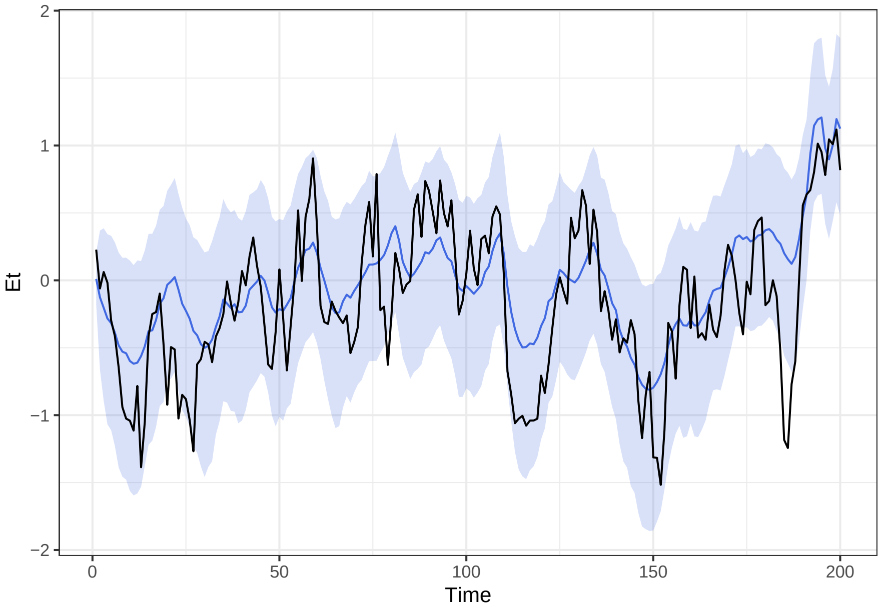
Infer v and Q: rho is known

```
av = 0
Rv = 1e-2
output = mcmc_disturbance_pois0(Y,c(av,Rv),E0,
                                rho_true=rho,
                                QPrior=c(1e-5,1),
                                Vxi = 0.3,
                                nburnin=10000,
                                nthin=2,
                                nsample=5000)
```

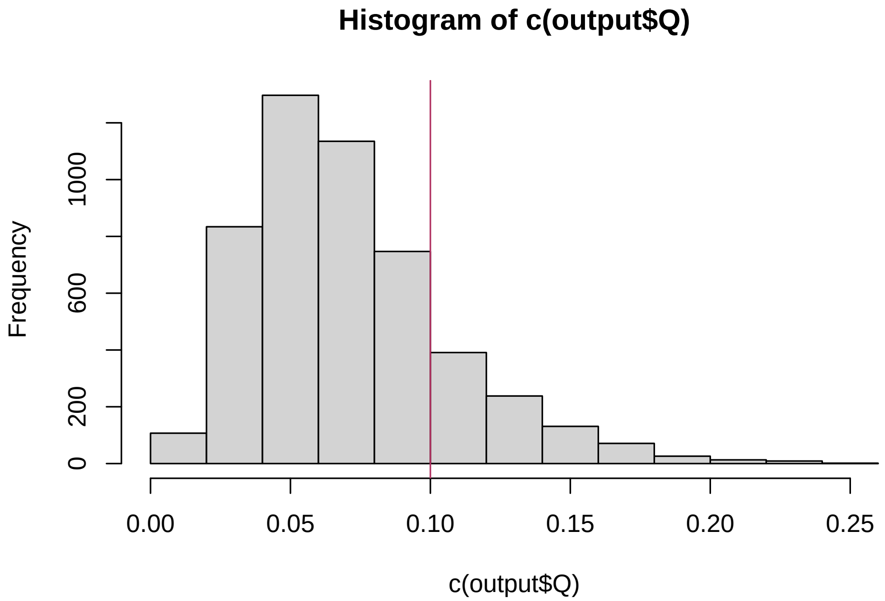
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



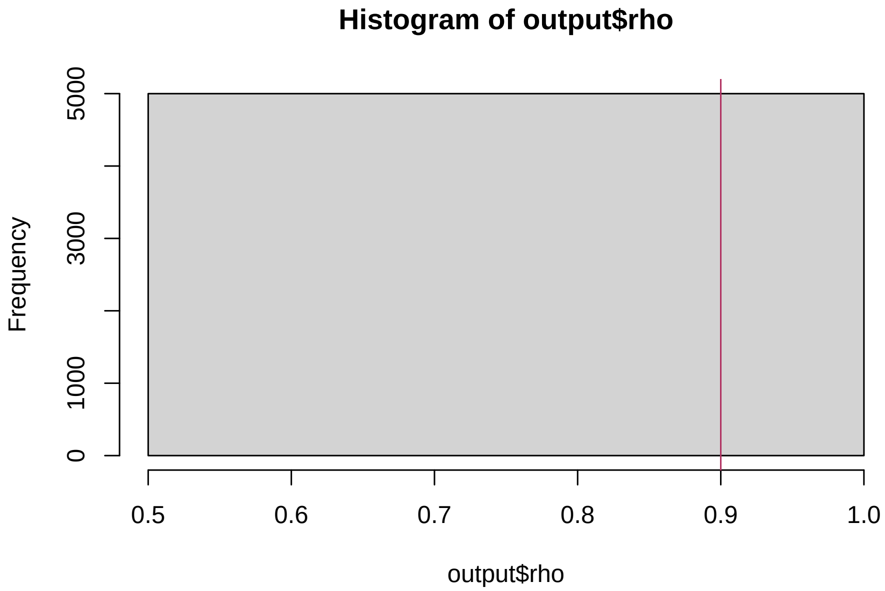
```
Et_est = matrix(0,nrow=n,ncol=5000)
Et_est[1,] = rho*rep(E0,5000) + output$v[1,]
for (t in 2:n) {
  Et_est[t,] = rho*Et_est[t-1,] + output$v[t,]
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



```
hist(c(output$Q))
abline(v=Q,col="maroon")
```



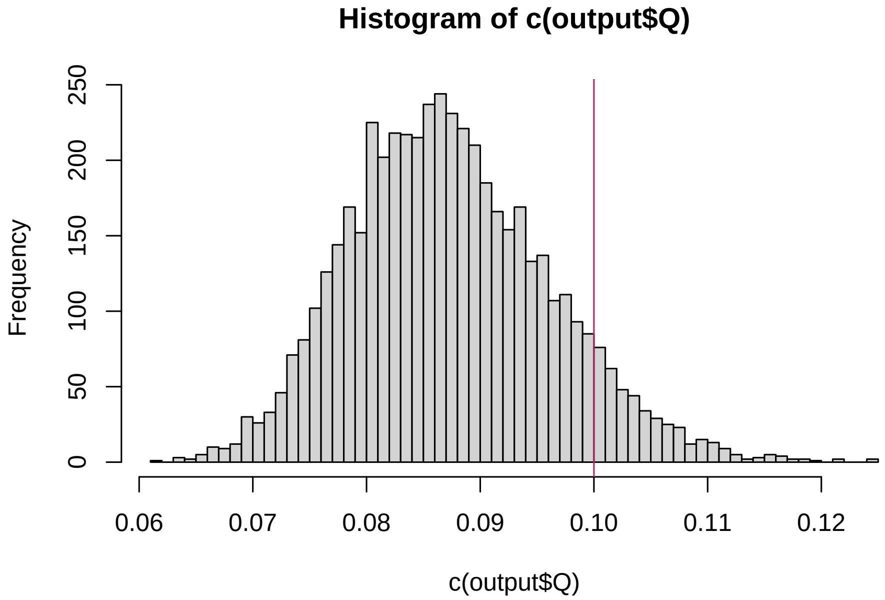
```
hist(output$rho)
abline(v=rho,col="maroon")
```



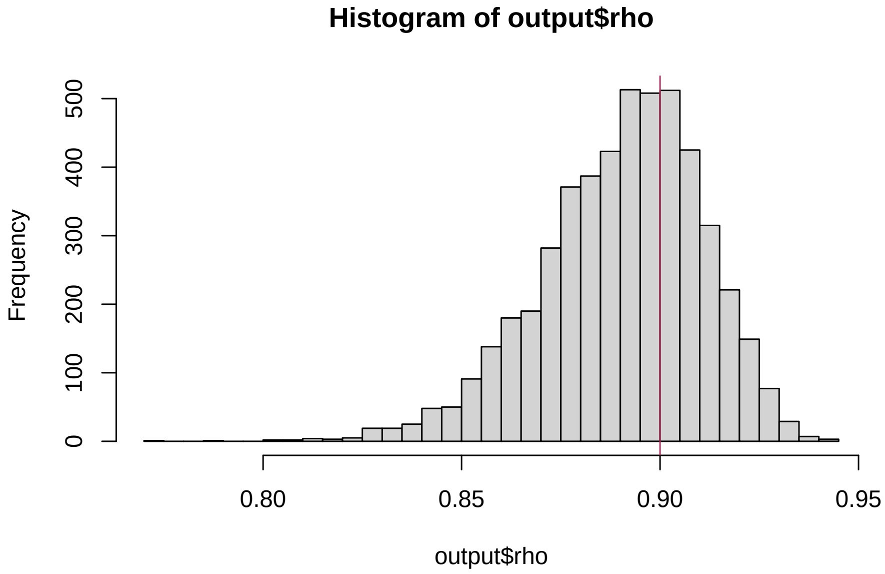
Infer rho and Q: v is known

```
av = 0
Rv = 1e-2
output = mcmc_disturbance_pois0(Y,c(av,Rv),E0,
                                QPrior=c(1e-5,1),
                                Vxi = 0.3,
                                vt_true=v,
                                nburnin=10000,
                                nthin=2,
                                nsample=5000)
```

```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



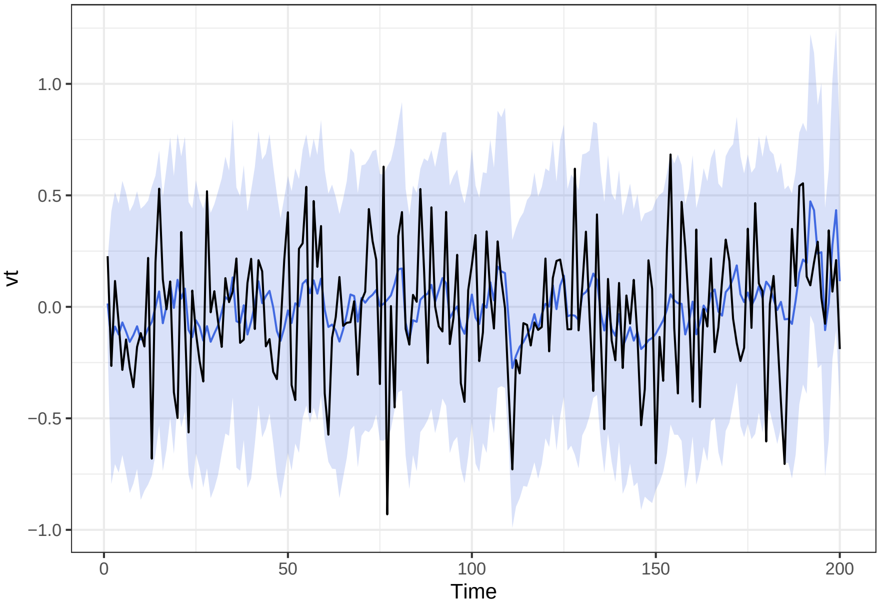
```
hist(output$rho,breaks=50)
abline(v=rho,col="maroon")
```



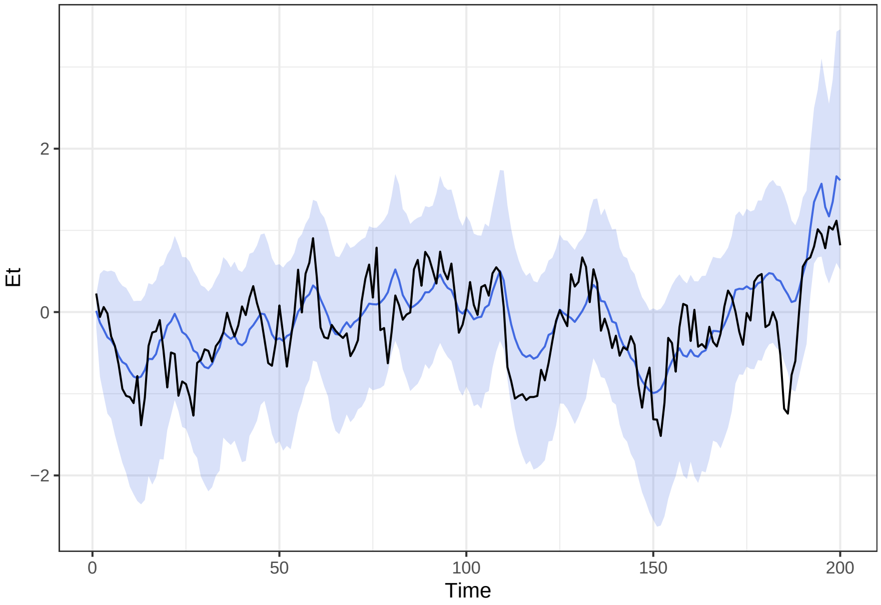
Infer all (that is vt,rho,Q)

```
av = 0
Rv = 1e-2
output = mcmc_disturbance_pois0(Y,c(av,Rv),E0,
                                QPrior=c(1e-5,1),
                                Vxi = 0.3,
                                nburnin=10000,
                                nthin=2,
                                nsample=5000)
```

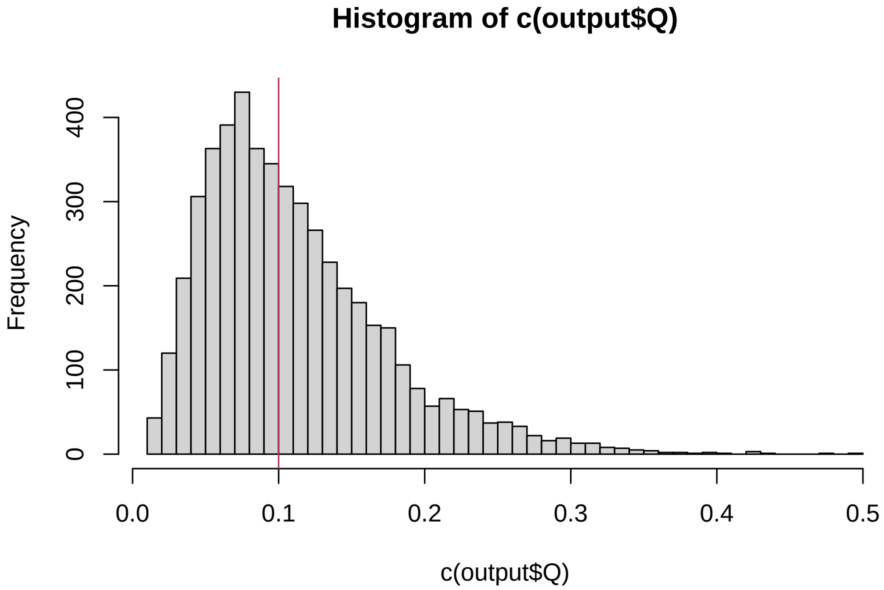
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



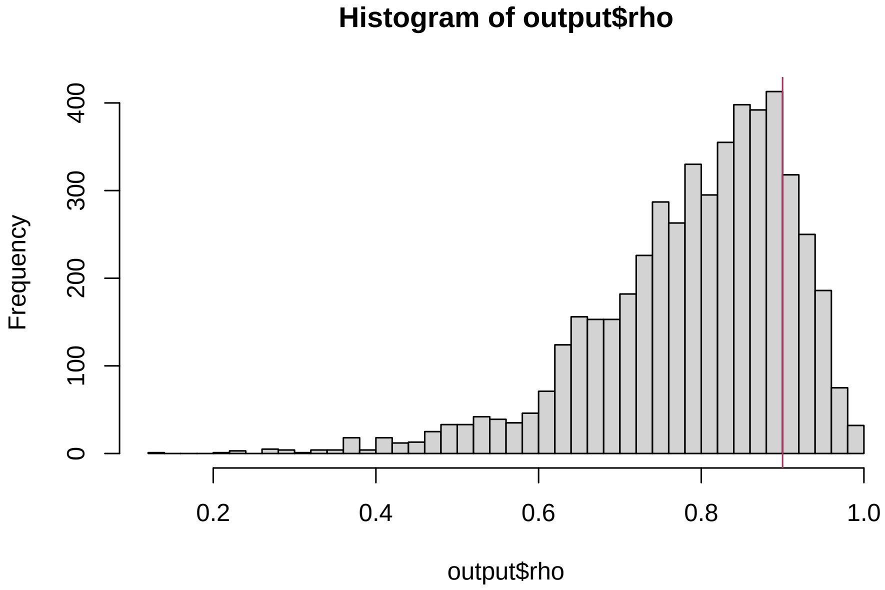
```
Et_est = matrix(0,nrow=n,ncol=5000)
Et_est[1,] = rho*rep(E0,5000) + output$v[1,]
for (t in 2:n) {
  Et_est[t,] = rho*Et_est[t-1,] + output$v[t,]
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



```
hist(output$rho,breaks=50)
abline(v=rho,col="maroon")
```



2. Koyck's Transfer Function

Simulated Data

```
n = 200
rho = 0.9     State to state transition, "G", the memory.
Q = 0.01      Evolution variance.
E0 = 0        Initial state of logarithm link
```

```
# X = sim.rw(n, 0, 0, 0.15)
X = rep(0.5,n)    step response.
v = rnorm(n,mean=0,sd=sqrt(Q))

Fx = update_Fx1(n,rho,X)
E = update_Et(n,Fx,v,rho,E0)

lambda = exp(E)
Y = rpois(n,lambda)

plot(Y,type="l", main="Observations")
```

Model.

$$y_t \sim Pois(\exp(\theta_t)).$$

$$\begin{cases} \theta_t = \rho\theta_{t-1} + \beta_t x_t \\ \beta_t = \beta_{t-1} + w_t, \quad w_t \sim N(0, W), \Rightarrow \beta_t = \sum_{j=1}^{t} w_j. \end{cases}$$

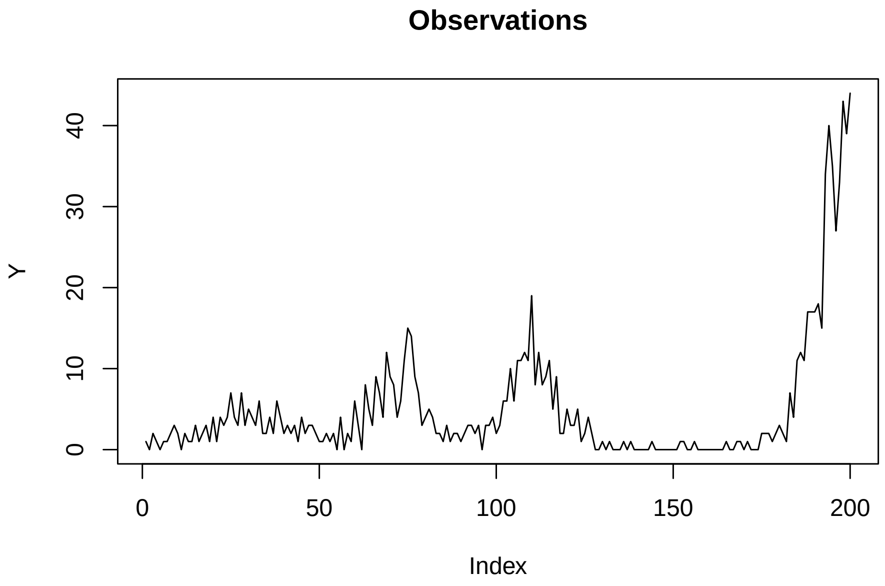
$\beta_0 \equiv 0.$

$w_1 \sim N(a_0, R_0),$

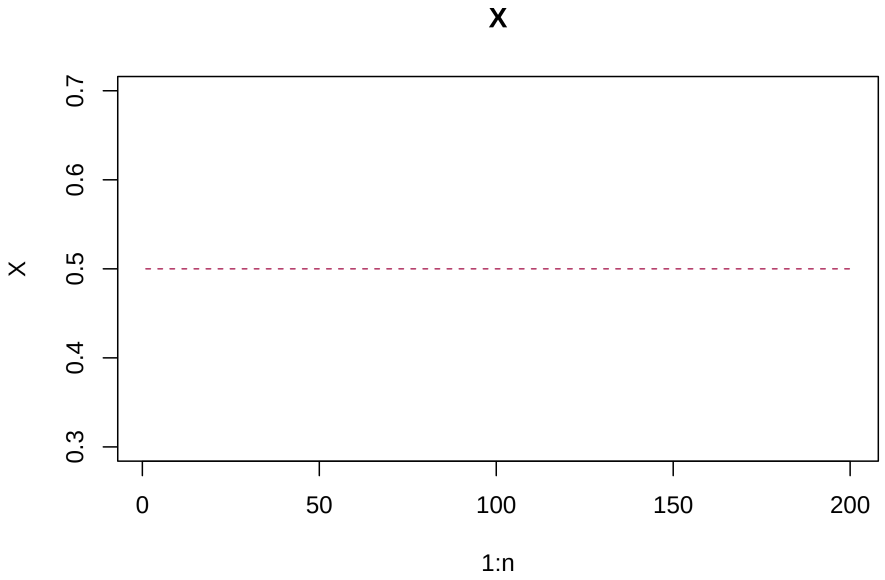
$$\Rightarrow \theta_t = \rho\theta_{t-1} + \beta_t x_t,$$
$$= \rho(\rho\theta_{t-2} + \beta_{t-1}x_{t-1}) + \beta_t x_t$$
$$= \rho^2\theta_{t-2} + \rho\beta_{t-1}x_{t-1} + \beta_t x_t.$$
$$= \rho^2(\rho\theta_{t-3} + \beta_{t-2}x_{t-2}) + \beta_{t-1}\rho x_{t-1} + \beta_t x_t.$$
$$= \rho^t\theta_0 + \sum_{k=0}^{t-1}\rho^k\beta_{t-k}x_{t-k}.$$
$$= \rho^t\theta_0 + \sum_{k=0}^{t-1}\rho^k x_{t-k}\sum_{j=1}^{t-k}w_j.$$
$$= \rho^t\theta_0 + x_t(w_1 + \cdots + w_t) + \rho x_{t-1}(w_1 + \cdots + w_{t-1}) + \cdots + \rho^{t-2}x_2(w_1 + w_2) + \rho^{t-1}x_1 w_1,$$
$$= \rho^t\theta_0 + w_1(x_t + \rho x_{t-1} + \cdots + \rho^{t-1}x_1) + w_2(x_t + \rho x_{t-1} + \cdots + \rho^{t-2}x_2) + \cdots + w_t \cdot x_t,$$

$$\theta_t = \rho^t\theta_0 + \sum_{j=1}^{t}\rho^{t-j}x_j \cdot w_1 + \sum_{j=2}^{t}\rho^{t-j}x_j \cdot w_2 + \cdots + w_t \cdot x_t.$$

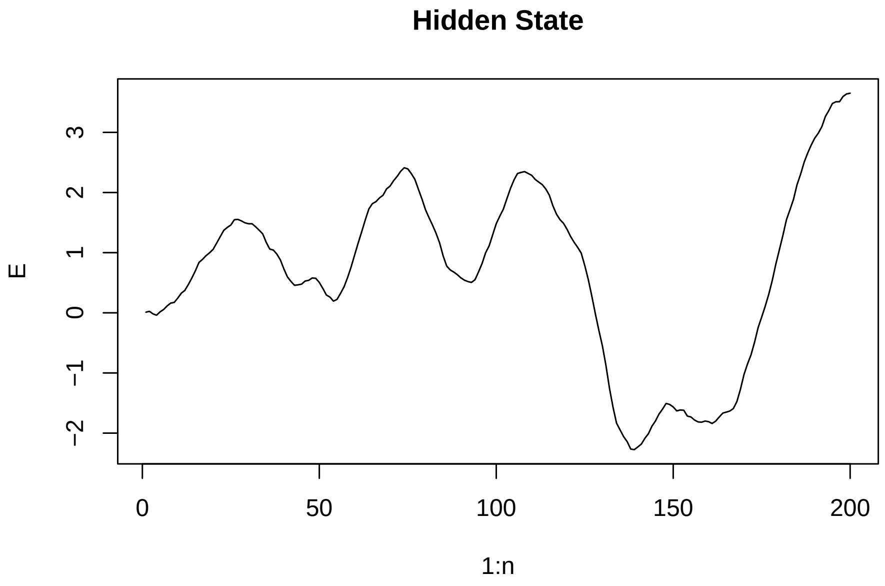
$$\begin{pmatrix}\theta_1\\ \vdots\\ \theta_n\end{pmatrix} = \begin{pmatrix}\rho_1\\ \vdots\\ \rho^n\end{pmatrix}\theta_0 + \begin{pmatrix} x_1 & & & 0\\ \rho x_1+x_2 & x_2 & & \\ \vdots & \vdots & \ddots & \\ \sum_{j=1}^{n}\rho^{n-j}x_j & \sum_{j=2}^{n}\rho^{n-j}x_j & \cdots & x_n\end{pmatrix}\begin{pmatrix}w_1\\ w_2\\ \vdots\\ w_n\end{pmatrix} = E_0 + F_x\cdot w.$$



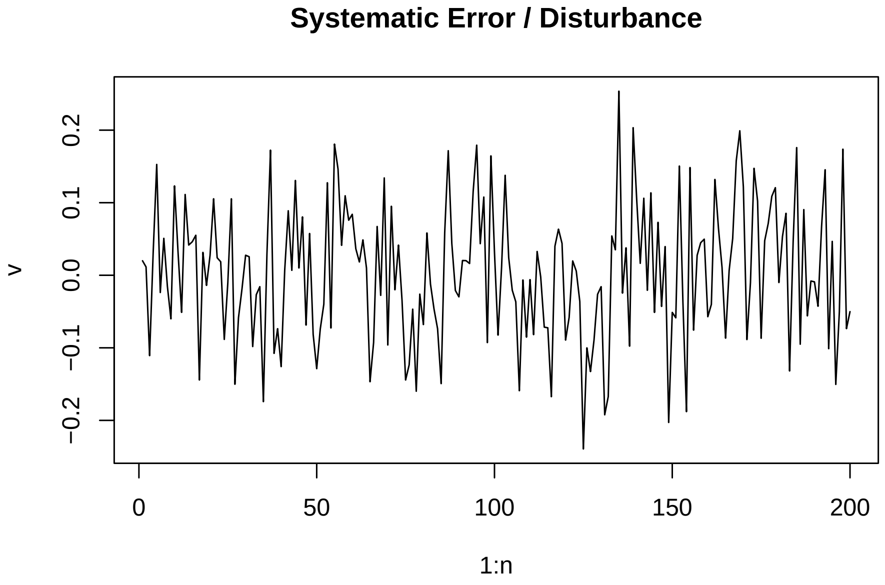
```r
plot(1:n,X,type="l",col="maroon",lty=2, main="X")
```



```r
plot(1:n,E,type="l", main="Hidden State")
```



```
plot(1:n,v,type="l", main="Systematic Error / Disturbance")
```



Linear Bayes Filtering

```
delta_grid = seq(0.1,0.9,by=0.1)
rho_grid = seq(0.1,0.9,by=0.05)
ndelta = length(delta_grid)
delta_prob = rep(0,ndelta)
rho_sel = rep(0,ndelta)
m0 = c(0,0)
C0 = diag(c(0.1,0.1))

for (i in 1:ndelta) {
  delta = delta_grid[i]
  logprob = get_rho_prob(Y,X,delta,rho_grid,m0,C0)
  rho_idx = which.max(logprob)
  rho_sel[i] = rho_grid[rho_idx]
  delta_prob[i] = logprob[rho_idx]
}
plot(delta_grid,delta_prob,type="l",
     xlab="delta",ylab="logprob")
```

```
# delta = 0.5
# logprob = get_rho_prob(Y,X,delta,rho_grid,m0,C0)
# plot(rho_grid,logprob,type="l")
#
delta = 0.8
logprob = get_rho_prob(Y,X,delta,rho_grid,m0,C0)
```

```
## Error in get_rho_prob(Y, X, delta, rho_grid, m0, C0): object 'rho_grid' not found
```

```
plot(rho_grid,logprob,type="l")
```

```
## Error in plot(rho_grid, logprob, type = "l"): object 'rho_grid' not found
```

```
rho_idx = which.max(logprob)
```

```
## Error in which.max(logprob): object 'logprob' not found
```

```
rho_hat = rho_grid[rho_idx]
```

```
## Error in eval(expr, envir, enclos): object 'rho_grid' not found
```

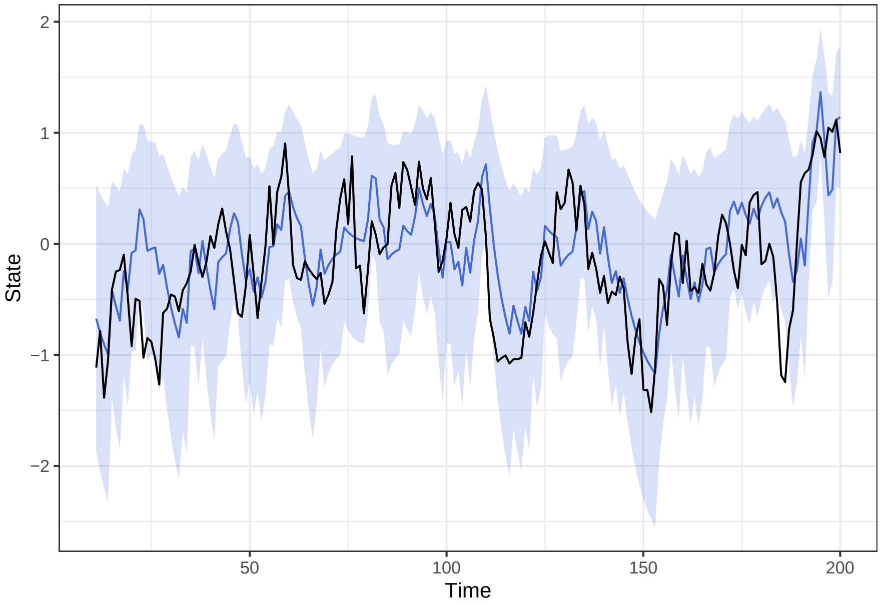
```
output = lbe_poissonX(Y,X,rho_hat,delta,m0,C0)
```

```
## Error in lbe_poissonX(Y, X, rho_hat, delta, m0, C0): object 'rho_hat' not found
```

```
ts = 2
tmp = data.frame(time=(ts+1):n, true=E[-c(1:ts)],y=Y[-c(1:ts)],
                 mt=c(output$mt[1,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[1,-c(1:(ts+1))])-2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[1,-c(1:(ts+1))])+2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])))
```

```
## Error in sqrt(c(output$Ct[1, 1, -c(1:(ts + 1))])): non-numeric argument to mathematical function
```

```
ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("State")
```



```
vest = diff(output$mt[2,-c(1:(ts+1))])
plot(vest,type="l",col="royalblue",
     ylim=c(min(c(vest,v)), max(c(vest,v))) )
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf

## Warning in max(x): no non-missing arguments to max; returning -Inf

## Error in plot.window(...): need finite 'xlim' values
```

```
lines(v[-c(1:ts)])
```

MCMC reparameterisation

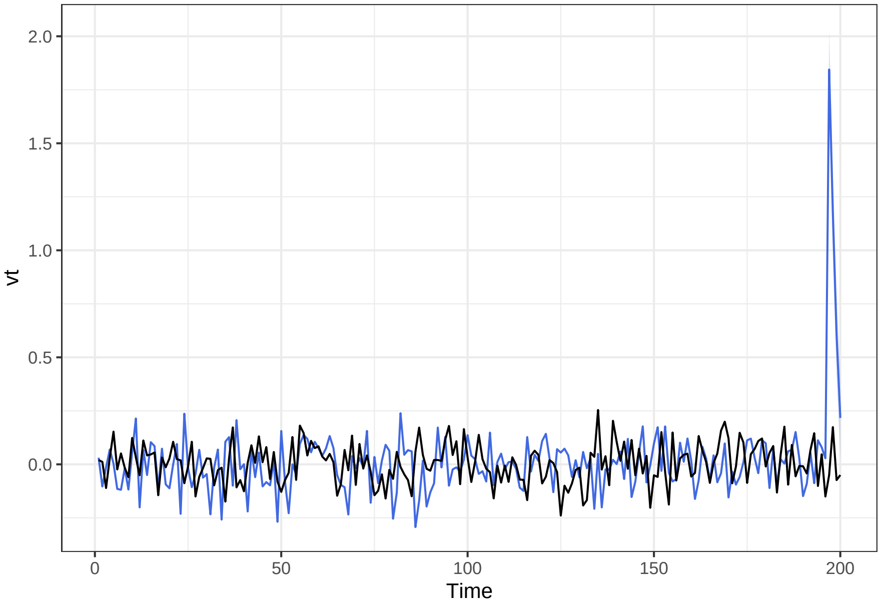
Settings

```
av = 0
Rv = 0.01
v1Prior = c(av,Rv)
QPrior=c(1e-2,1)
wt_hat = diff(output$mt[2,-1])
What = estimate_state_var(wt_hat,QPrior)
Vxi = 0.005
E0Prior = c(0,0.1)
```

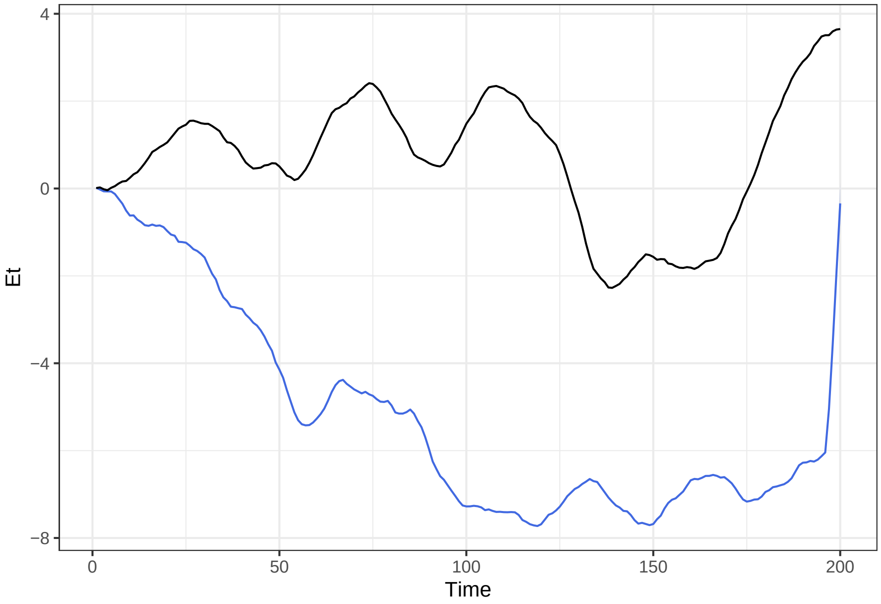
Infer w: W and rho is known

```
output = mcmc_disturbance_pois(Y,X,
                               Vxi = Vxi,
                               QPrior = QPrior,
                               v1Prior = v1Prior,
                               E0_true=E0,
                               rho_true = rho,
                               Q_true = Q,
                               nburnin = 10000,
                               nthin = 2,
                               nsample = 5000)
```

```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```

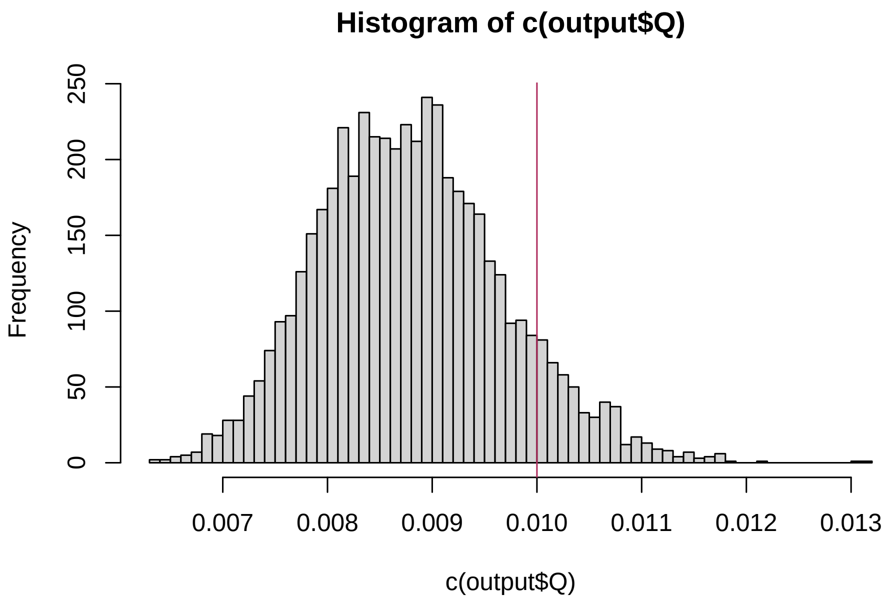


⇐ slow moving closer to the truth.

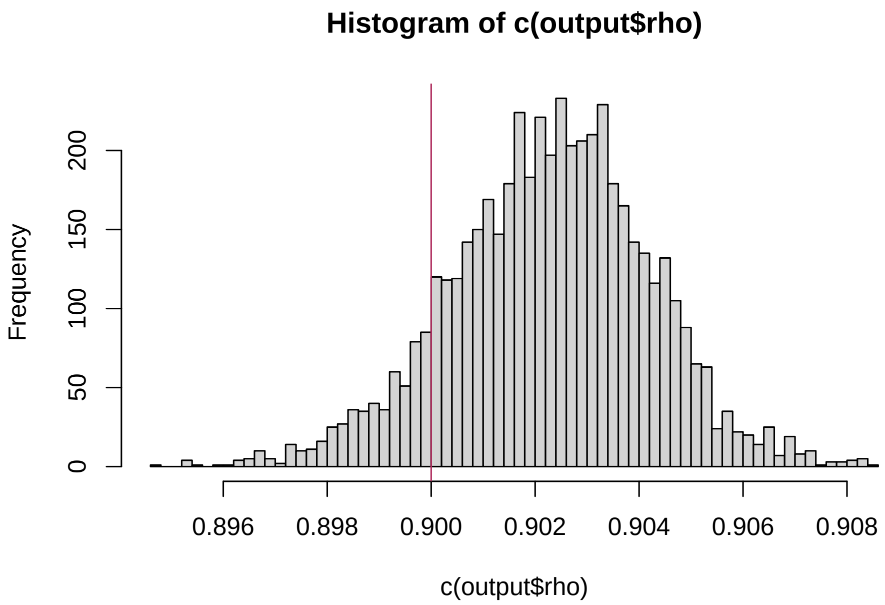
Infer W and rho: w is known

```
av = 0
Rv = 0.01
output = mcmc_disturbance_pois(Y,X,
                               Vxi = Vxi,
                               QPrior = QPrior,
                               v1Prior = v1Prior,
                               vt_true = v,
                               E0_true = E0,
                               nburnin = 10000,
                               nthin = 2,
                               nsample = 5000)
print(output$rho_accept)
```

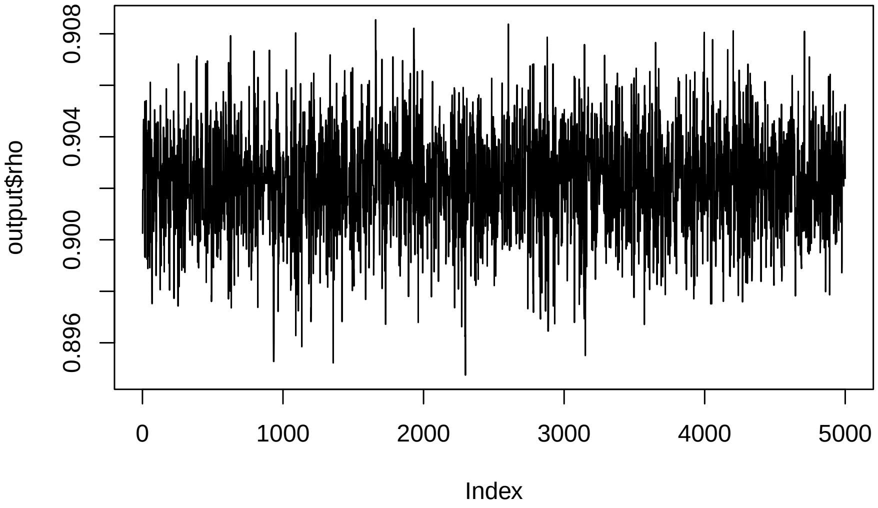
```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



```
hist(c(output$rho),breaks=50)
abline(v=rho,col="maroon")
```



```
plot(output$rho,type="l")
```



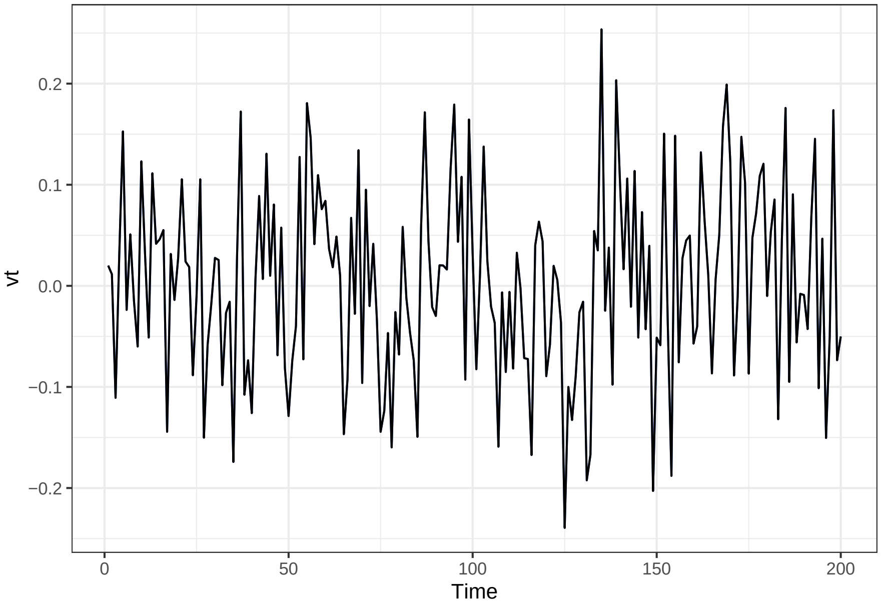
Infer w and rho: W and theta[0] is known

```
output = mcmc_disturbance_pois(Y,X,
                               Vxi = Vxi,
                               QPrior = QPrior,
                               v1Prior = v1Prior,
                               rho_init = rho_hat,
                               vt_init = c(0,wt_hat),
                               Q_true = Q,
                               E0_true = E0,
                               nburnin = 100000,
                               nthin = 20,
                               nsample = 10000)
```

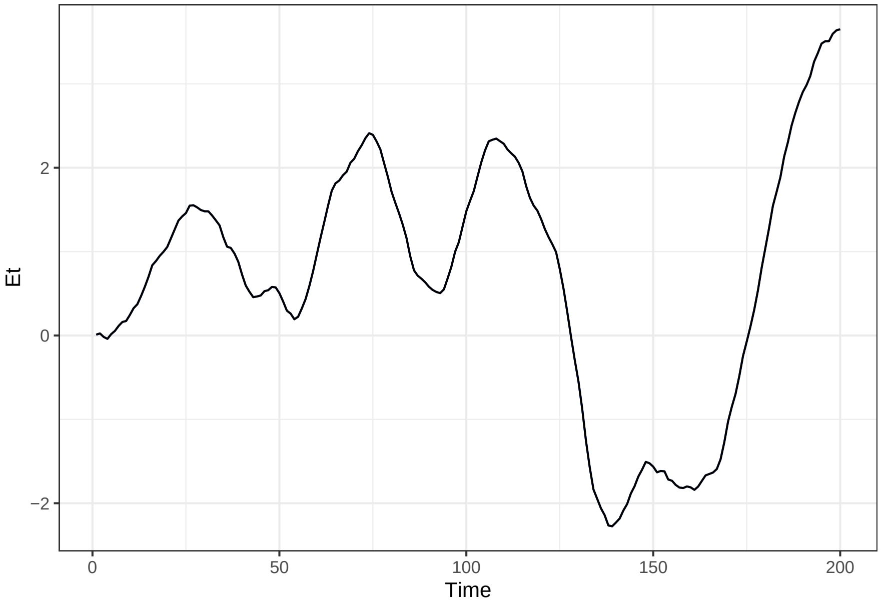
```
## Error in mcmc_disturbance_pois(Y, X, Vxi = Vxi, QPrior = QPrior, v1Prior = v1Prior, : object 'rho_ha
```

```
print(output$rho_accept)
```

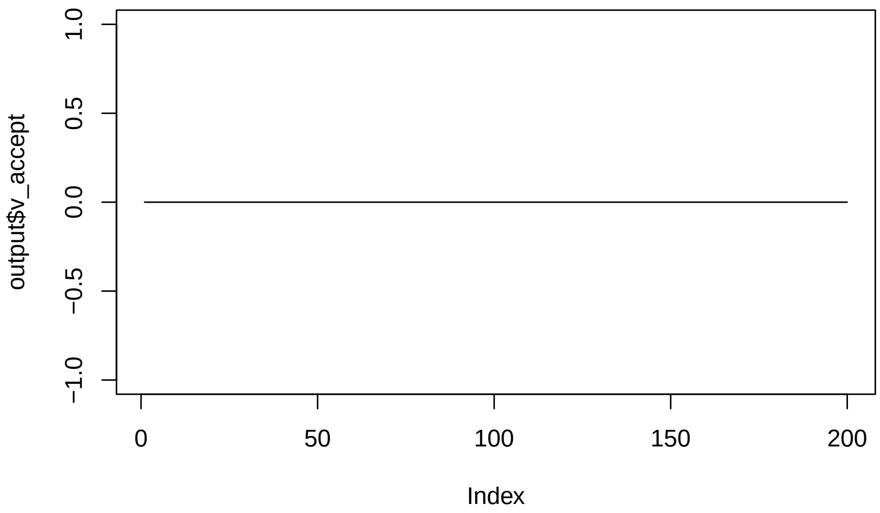
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



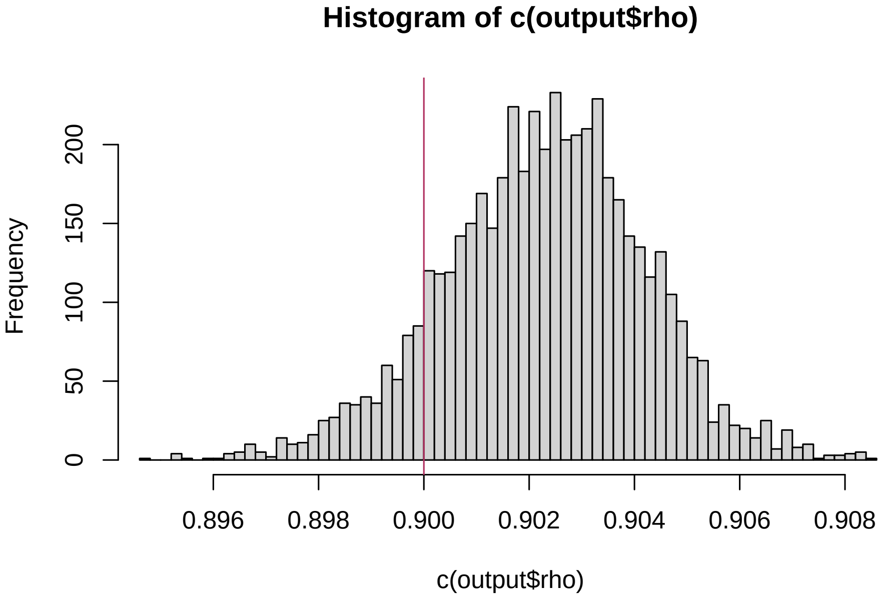
```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



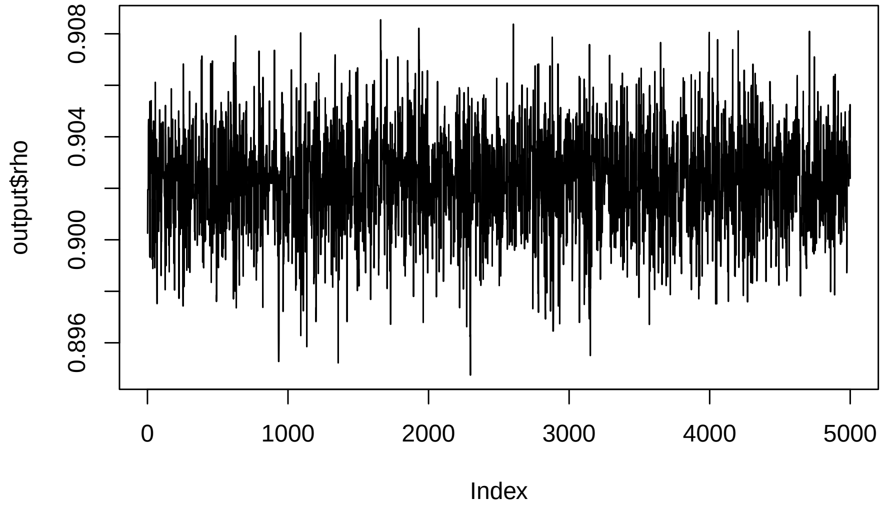
```
plot(output$v_accept,type="l")
```



```
hist(c(output$rho),breaks=50,
     xlim=c(min(c(rho,output$rho)),
              max(c(rho,output$rho))))
abline(v=rho,col="maroon")
```

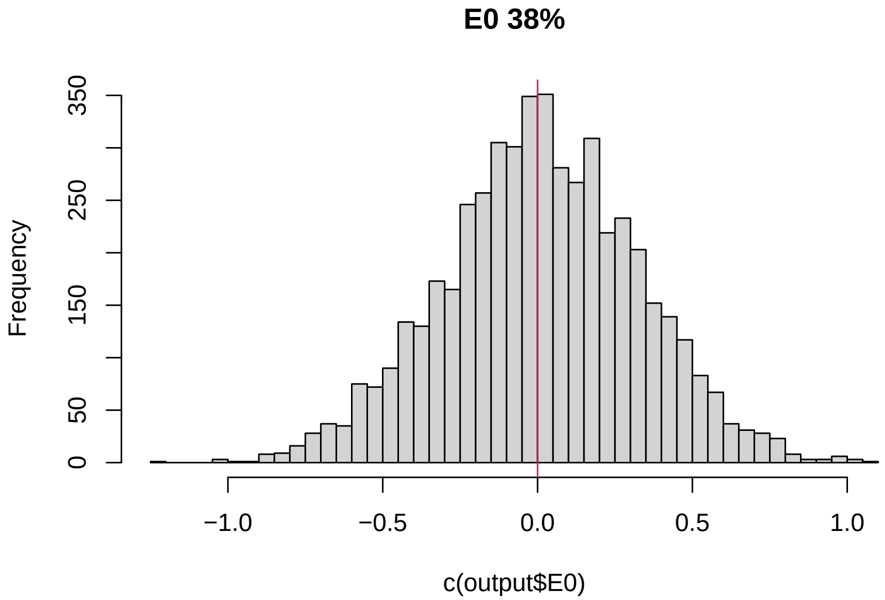


```
plot(output$rho,type="l")
```

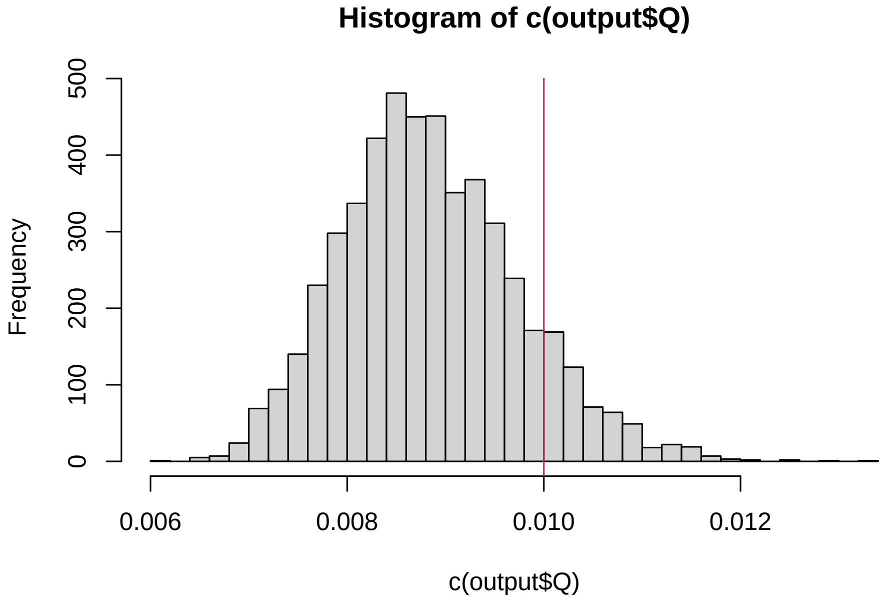


Infer theta[0] and W

```
output = mcmc_disturbance_pois(Y,X,
                               E0Prior = E0Prior,
                               Vxi = Vxi,
                               QPrior = QPrior,
                               rho_true = rho,
                               vt_true = v,
                               nburnin = 10000,
                               nthin = 2,
                               nsample = 5000)
print(output$E0_accept)
hist(c(output$E0),breaks=50,
     main=paste0("E0 ",round(output$E0_accept*100),"%"))
abline(v=E0,col="maroon")
```



```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



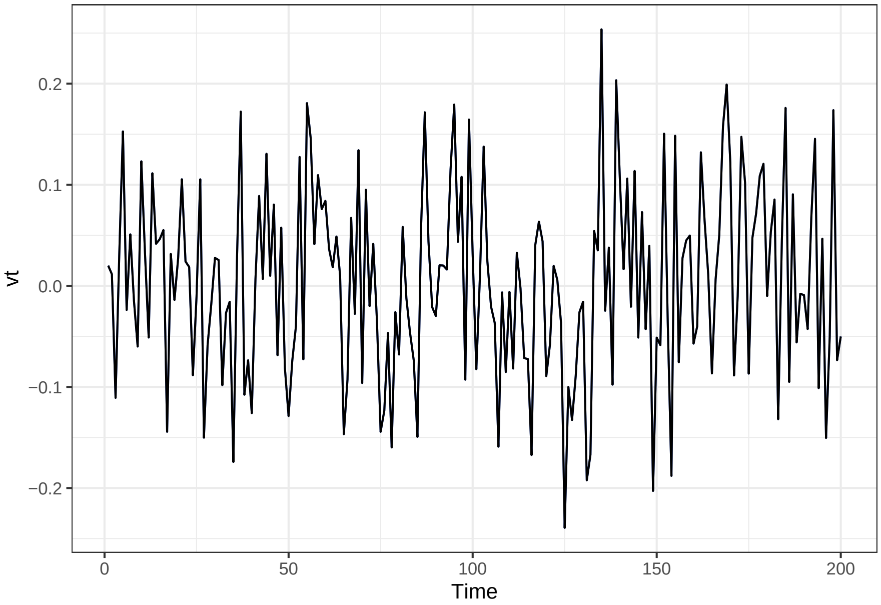
Infer w and rho and E0

```
output = mcmc_disturbance_pois(Y,X,
                               E0Prior = E0Prior,
                               Vxi = Vxi,
                               QPrior = QPrior,
                               v1Prior = v1Prior,
                               E0_init = E0,
                               rho_init = rho_hat,
                               vt_init = c(0,wt_hat),
                               Q_true = Q,
                               nburnin = 100000,
                               nthin = 20,
                               nsample = 10000)
```

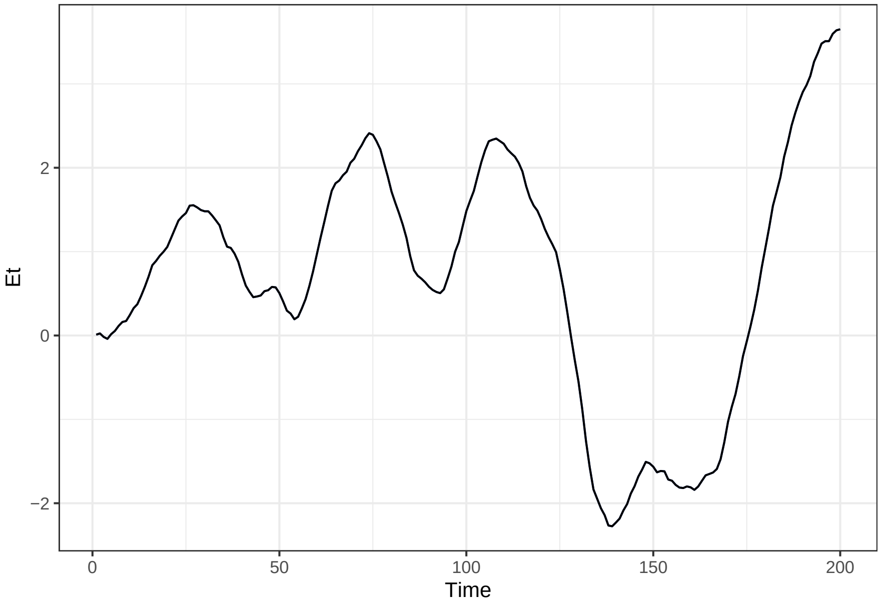
```
## Error in mcmc_disturbance_pois(Y, X, E0Prior = E0Prior, Vxi = Vxi, QPrior = QPrior, : object 'rho_ha
```

```
print(output$rho_accept)
print(output$E0_accept)
```

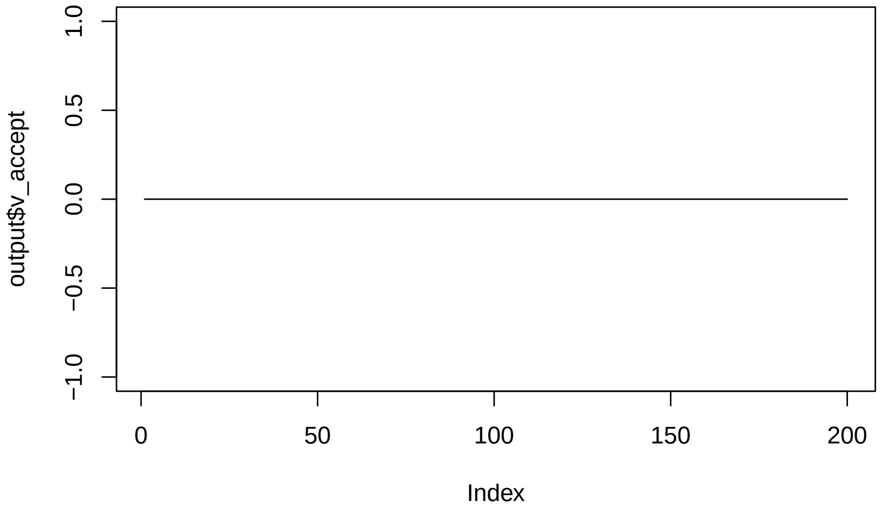
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



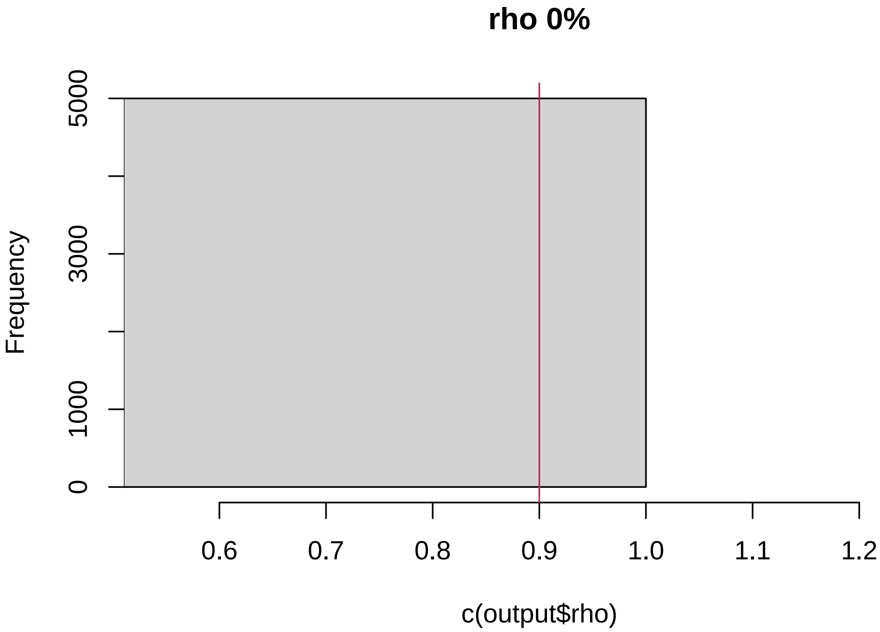
```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



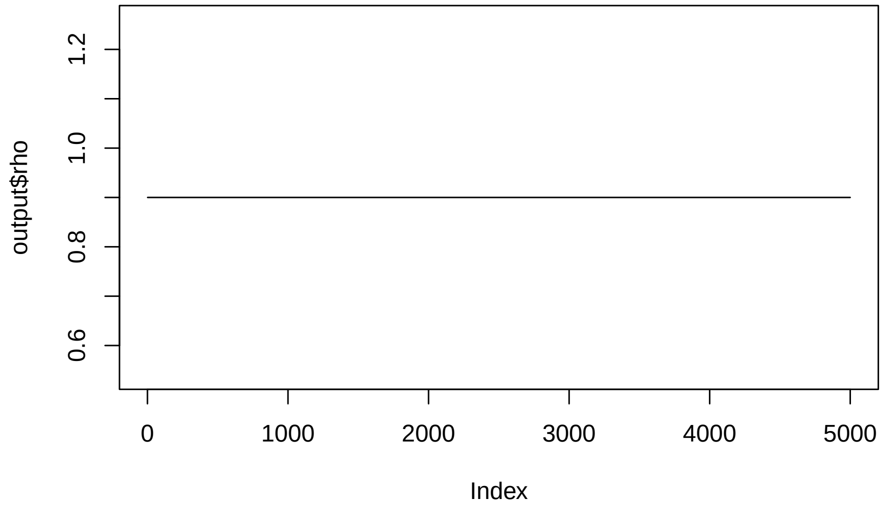
```
plot(output$v_accept,type="l")
```



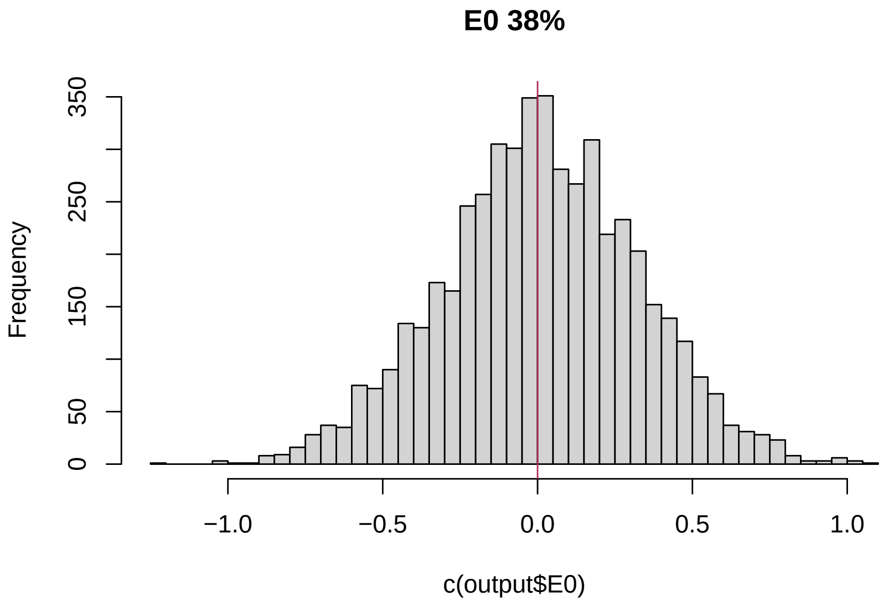
```
hist(c(output$rho),breaks=50,
     xlim=c(min(c(rho,output$rho)),
            max(c(rho,output$rho))),
     main=paste0("rho ",round(output$rho_accept*100),"%"))
abline(v=rho,col="maroon")
```



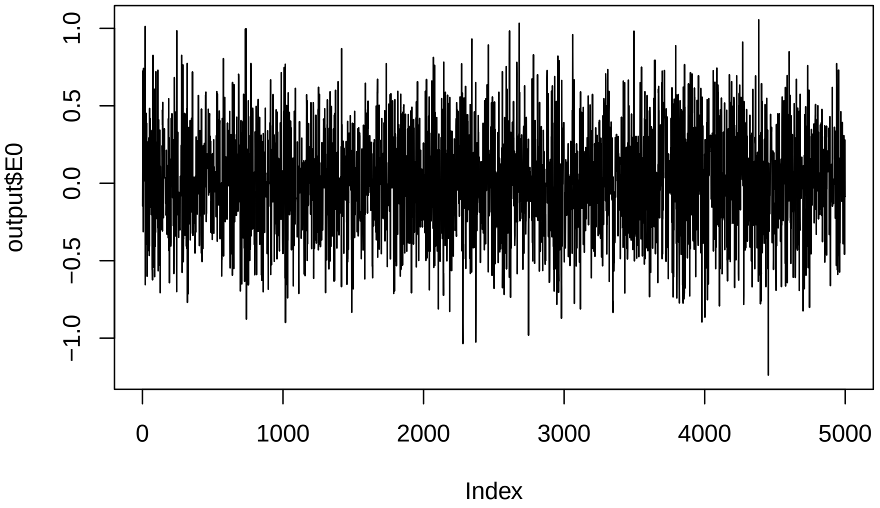
```
plot(output$rho,type="l")
```



```
hist(c(output$E0),breaks=50,
     main=paste0("E0 ",round(output$E0_accept*100),"%"))
abline(v=E0,col="maroon")
```



```
plot(output$E0,type="l")
```



3. Pascal-Distributed Lags

Model.

$y_t \sim Pois(\lambda_t = exp(\theta_t))$.

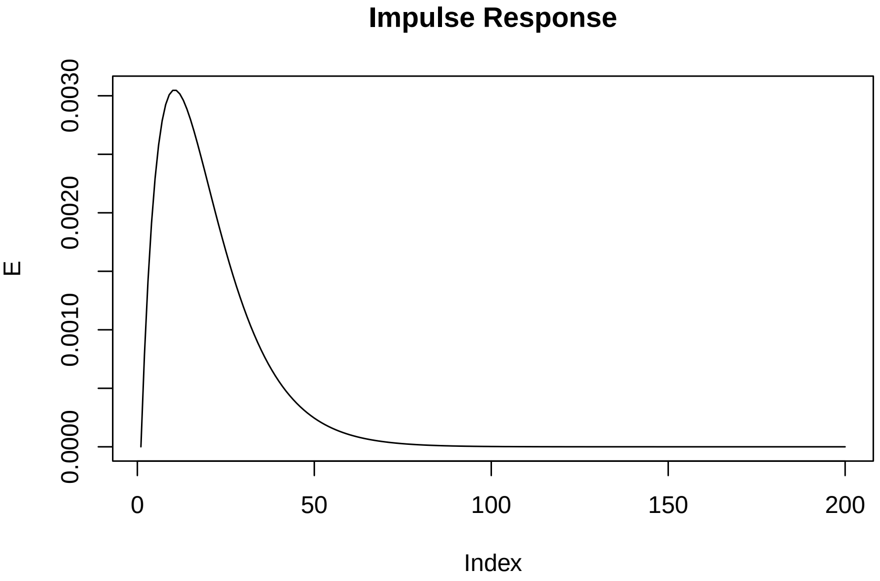
$\theta_t = 2\rho\theta_{t-1} - \rho^2\theta_{t-2} + \beta_t(1-\rho)^2 x_t$.

$\beta_t = \beta_{t-1} + w_t$, $w_t \sim N(0, W)$,

Simulated Data

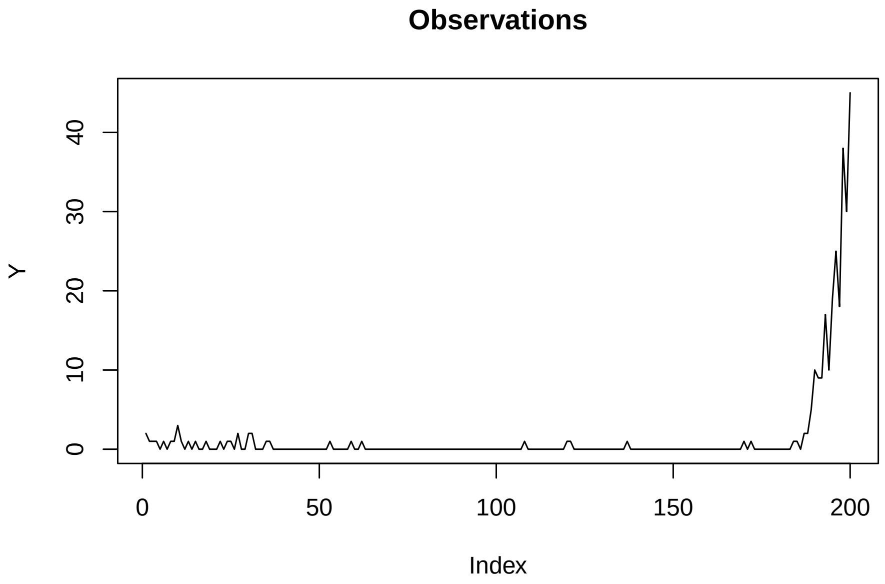
```
n = 200
rho = 0.9
Q = 0.5
r = 2
```

```
# X = sim.rw(n, 0, 0, 0.15)
v = rnorm(n,mean=0,sd=sqrt(Q))
X = c(1,rep(0,n-1))
Fx = update_Fx_Solow(n,rho,X)
E = update_Et(n,Fx,v)
plot(E,type="l",main="Impulse Response")
```

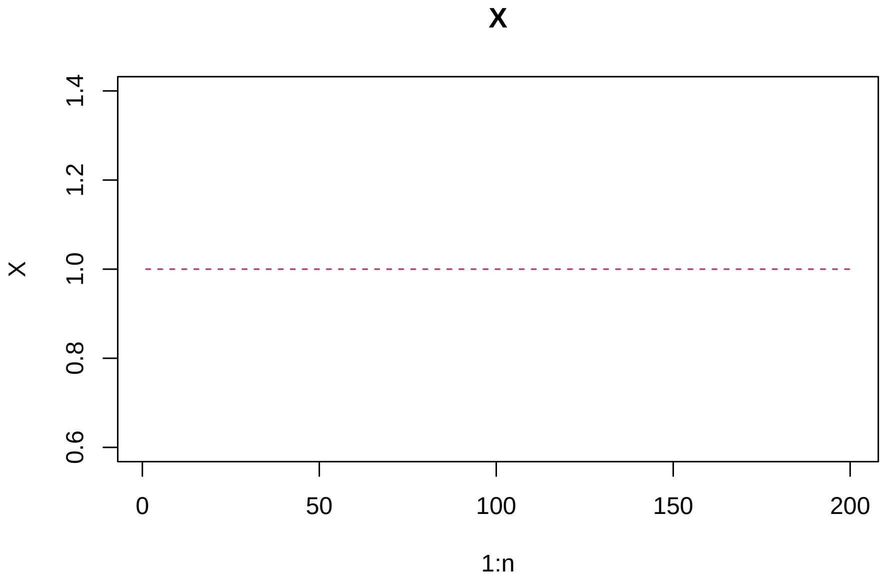


```
X = rep(1,n)
Fx = update_Fx_Solow(n,rho,X)
E = update_Et(n,Fx,v,rho,0)
```

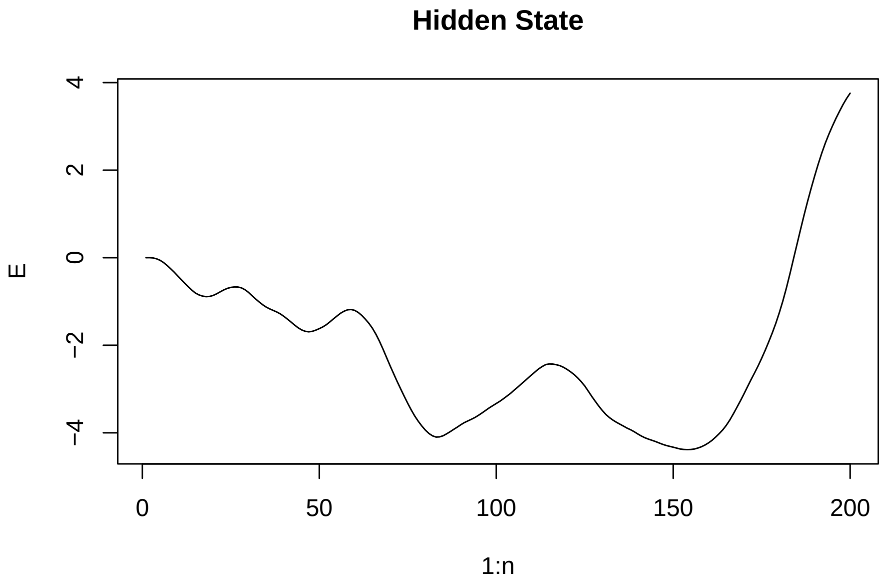
```
lambda = exp(E)
Y = rpois(n,lambda)

plot(Y,type="l", main="Observations")
```



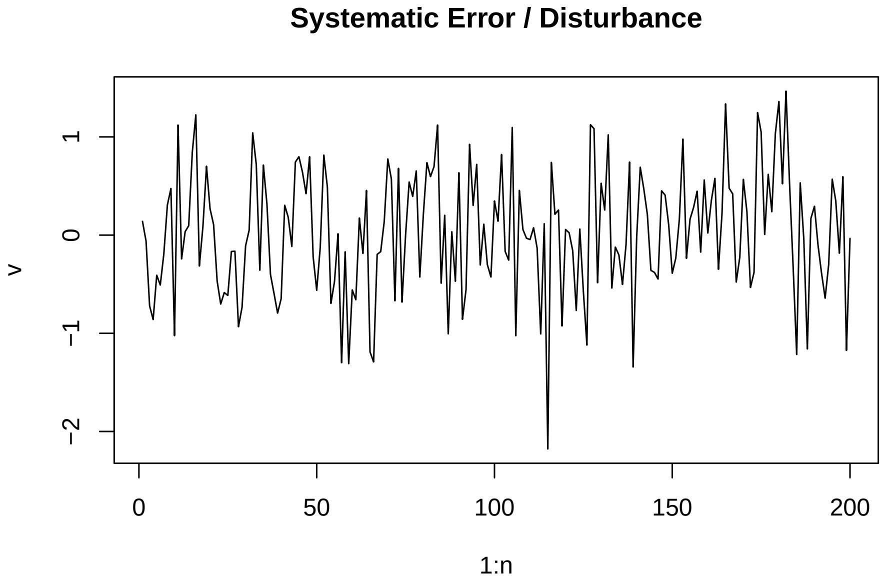
```
plot(1:n,X,type="l",col="maroon",lty=2, main="X")
```



```
plot(1:n,E,type="l", main="Hidden State")
```



```
plot(1:n,v,type="l", main="Systematic Error / Disturbance")
```



Linear Bayes Filetering

```
delta_grid = seq(0.5,0.9,by=0.1)
rho_grid = seq(0.1,0.9,by=0.05)
ndelta = length(delta_grid)
delta_prob = rep(NA,ndelta)
rho_sel = rep(0,ndelta)
m0 = c(0,0,0)
C0 = diag(c(0.01,0.01,0.01))

for (i in 1:ndelta) {
  delta = delta_grid[i]
  logprob = get_rho_prob(Y,X,delta,rho_grid,m0,C0,"Solow")
  names(logprob) = rho_grid
  logprob2 = logprob[is.finite(logprob)]
  if (length(logprob2)>0) {
    rho_sel = as.numeric(names(which.max(logprob2)))
    rho_idx = which(rho_sel == rho_grid)
    rho_sel[i] = rho_grid[rho_idx]
    delta_prob[i] = logprob[rho_idx]
  }
}
plot(delta_grid,delta_prob,type="l",
     xlab="delta",ylab="logprob")
```

```
delta = 0.85
logprob = get_rho_prob(Y,X,delta,rho_grid,m0,C0,"Solow")
```

```
## Error in get_rho_prob(Y, X, delta, rho_grid, m0, C0, "Solow"): object 'rho_grid' not found
```

```
rho_idx = which.max(logprob)
```

```
## Error in which.max(logprob): object 'logprob' not found
```

```
rho_hat = rho_grid[rho_idx]
```

```
## Error in eval(expr, envir, enclos): object 'rho_grid' not found
```

```
plot(rho_grid,logprob,type="l",
     main=paste0("rho_hat=",rho_hat))
```

```
## Error in plot(rho_grid, logprob, type = "l", main = paste0("rho_hat=", : object 'rho_grid' not found
```

```
abline(v=rho,col="maroon",lty=2)
```

```
## Error in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...): plot.new has not been called yet
```

```
abline(v=rho_hat,col="royalblue")
```

```
## Error in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...): object 'rho_hat' not found
```

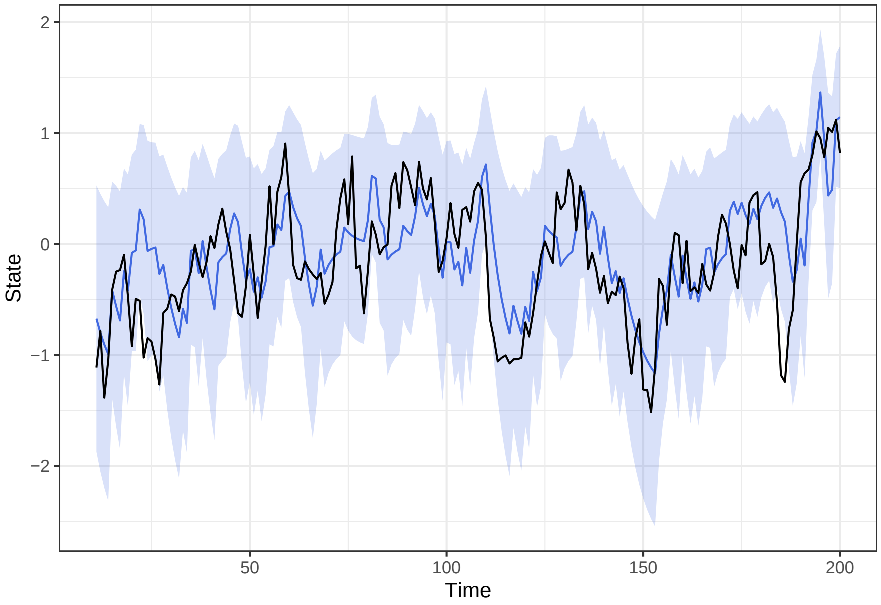
```
output = lbe_poissonSolow(Y,X,rho_hat,delta,m0,C0)
```

```
## Error in lbe_poissonSolow(Y, X, rho_hat, delta, m0, C0): object 'rho_hat' not found
```

```
ts = 2
tmp = data.frame(time=(ts+1):n, true=E[-c(1:ts)],y=Y[-c(1:ts)],
                 mt=c(output$mt[1,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[1,-c(1:(ts+1))])-2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[1,-c(1:(ts+1))])+2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])))
```

```
## Error in sqrt(c(output$Ct[1, 1, -c(1:(ts + 1))])): non-numeric argument to mathematical function
```

```
ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("State")
```



```
vest = diff(output$mt[3,-c(1:(ts+1))])
plot(vest,type="l",col="royalblue",
     ylim=c(min(c(vest,v)), max(c(vest,v))) )
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf
## Warning in max(x): no non-missing arguments to max; returning -Inf
## Error in plot.window(...): need finite 'xlim' values
```

```
lines(v[-c(1:ts)])
```

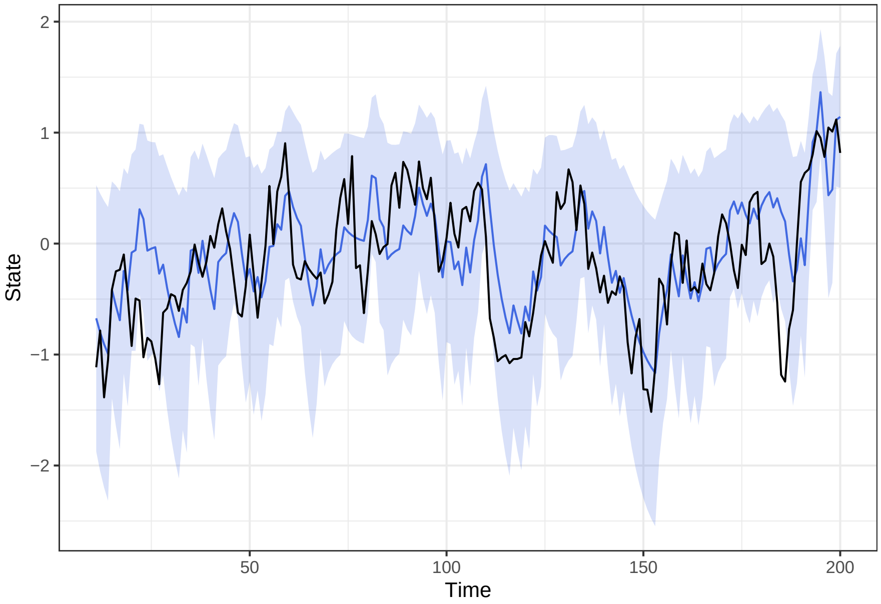
```
output = lbe_poissonSolow0(Y,X,rho,Q,m0,C0)
```

```
## Error in lbe_poissonSolow0(Y, X, rho, Q, m0, C0): Not a matrix.
```

```
ts = 2
tmp = data.frame(time=(ts+1):n, true=E[-c(1:ts)],y=Y[-c(1:ts)],
                 mt=c(output$mt[1,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[1,-c(1:(ts+1))])-2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[1,-c(1:(ts+1))])+2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])))
```

```
## Error in sqrt(c(output$Ct[1, 1, -c(1:(ts + 1))])): non-numeric argument to mathematical function
```

```
ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("State")
```



```
vest = diff(output$mt[3,-c(1:(ts+1))])
plot(vest,type="l",col="royalblue",
     ylim=c(min(c(vest,v)), max(c(vest,v))) )
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf
```

```
## Warning in max(x): no non-missing arguments to max; returning -Inf
```

```
## Error in plot.window(...): need finite 'xlim' values
```

```
lines(v[-c(1:ts)])
```

MCMC reparameterisation

Settings

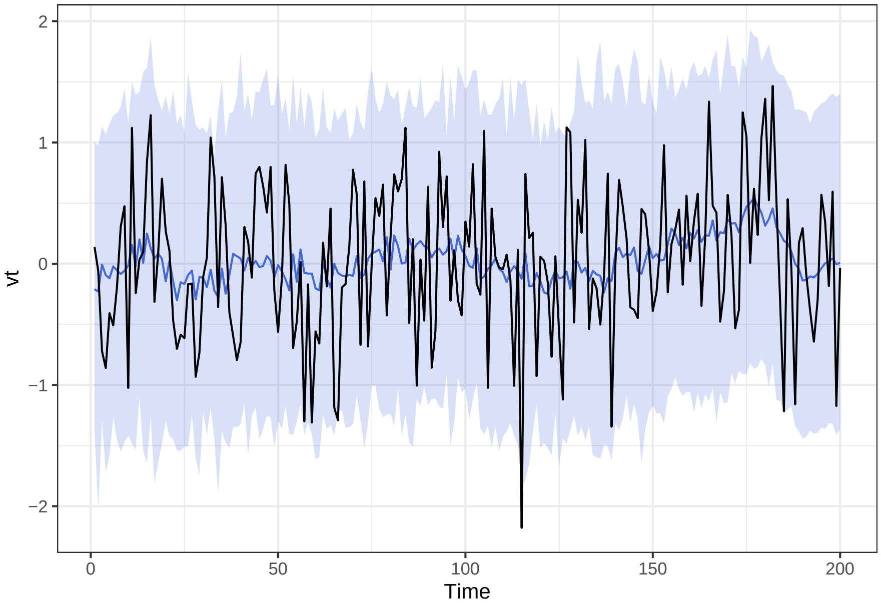
```
v1Prior = c(0,0.5)
QPrior=c(1e-2,1)
Vxi = 0.02
```

```
wt_hat = diff(output$mt[2,-1])
What = estimate_state_var(wt_hat,QPrior)
```

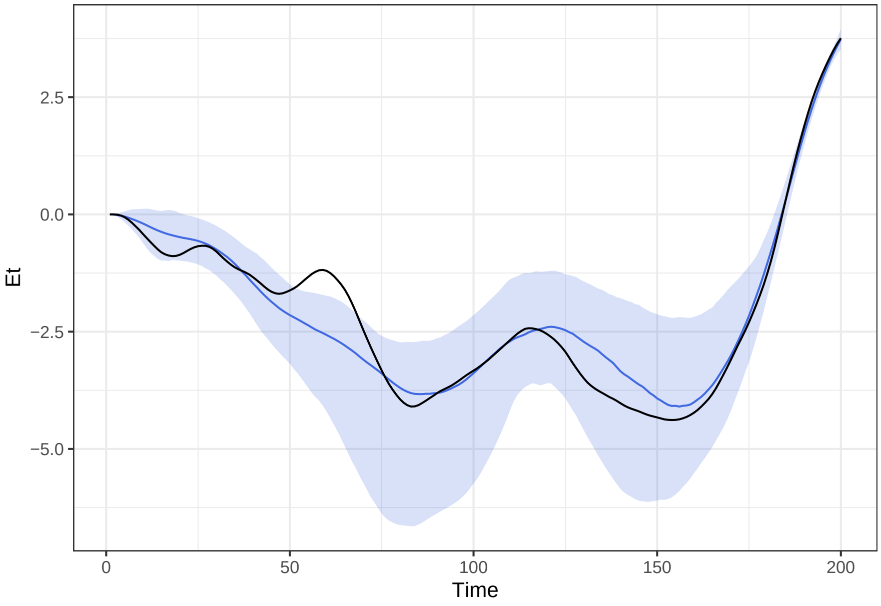
Infer w: W and rho is known

```
output = mcmc_disturbance_pois_solow(Y,X,
                                     v1Prior = v1Prior,
                                     vt_init = v,
                                     rho_true = rho,
                                     Q_true = Q,
                                     nburnin = 10000,
                                     nthin = 2,
                                     nsample = 5000)
```

```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```

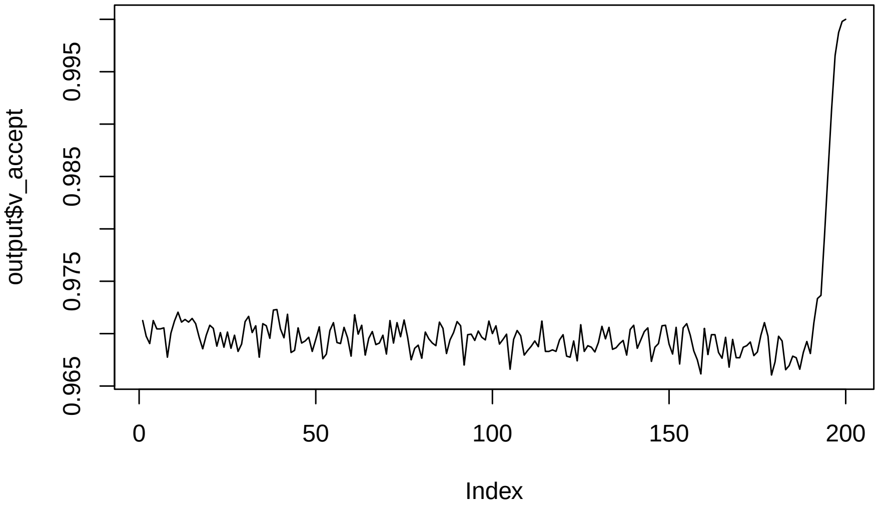


```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



⇐ Look's good ,

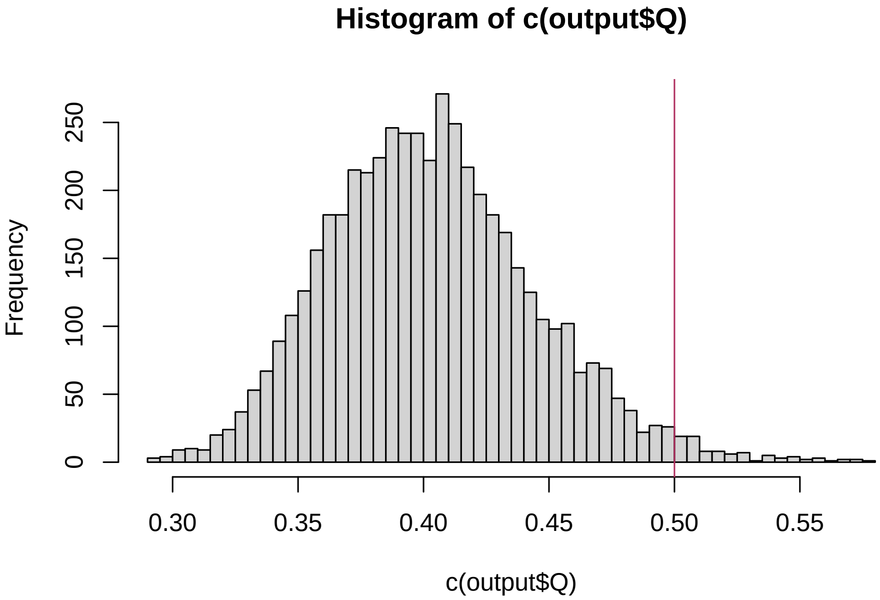
```
plot(output$v_accept,type="l")
```



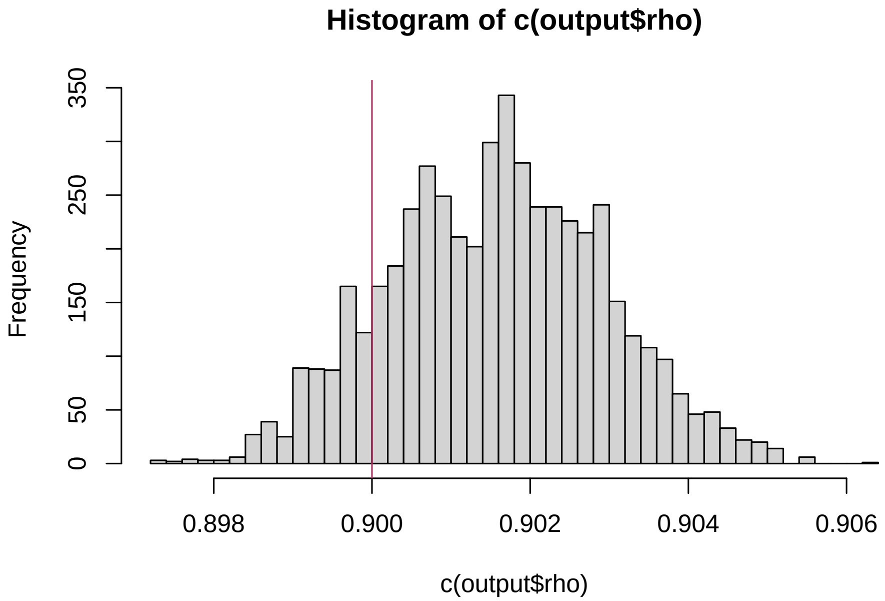
Infer W and rho: w is known

```
output = mcmc_disturbance_pois_solow(Y,X,
                                     Vxi = Vxi,
                                     QPrior = QPrior,
                                     rho_init = rho,
                                     Q_init = Q,
                                     vt_true = v,
                                     nburnin = 10000,
                                     nthin = 2,
                                     nsample = 5000)
print(output$rho_accept)
```

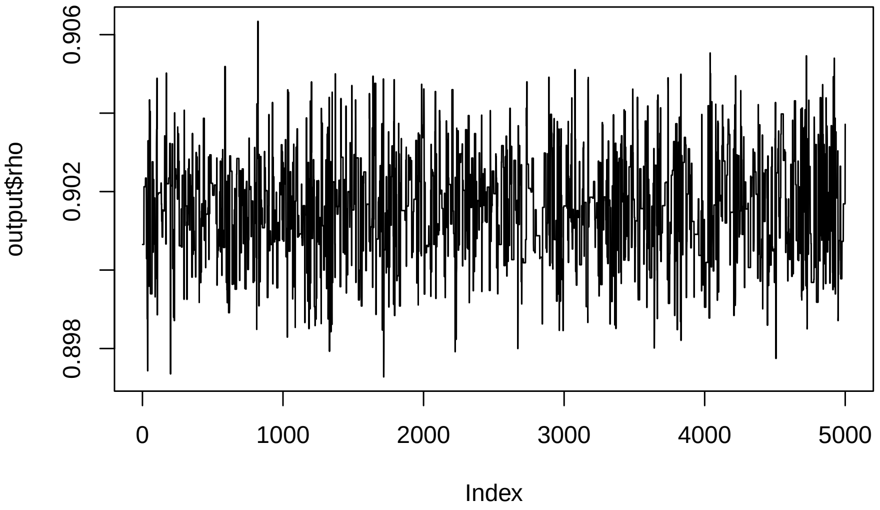
```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



```
hist(c(output$rho),breaks=50)
abline(v=rho,col="maroon")
```



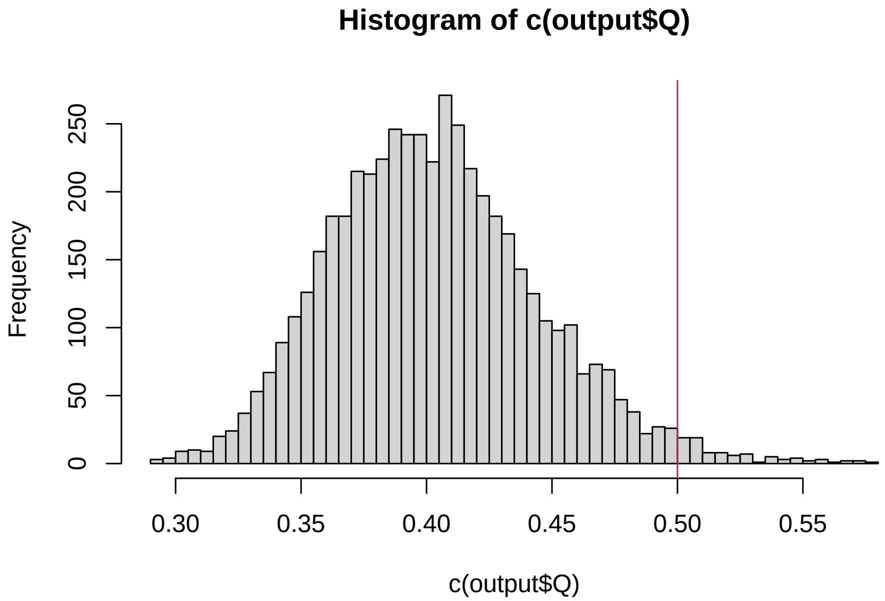
```
plot(output$rho,type="l")
```



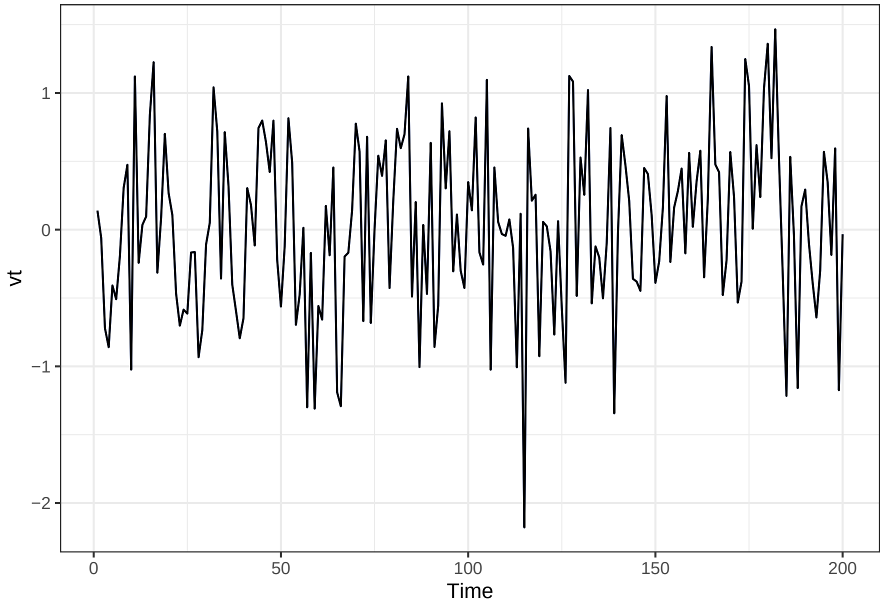
Infer w and W: rho is known

```
output = mcmc_disturbance_pois_solow(Y,X,
                                     QPrior = QPrior,
                                     v1Prior = v1Prior,
                                     Q_init = What,
                                     vt_init = c(0,wt_hat),
                                     rho_true = rho,
                                     nburnin = 10000,
                                     nthin = 2,
                                     nsample = 5000)
```

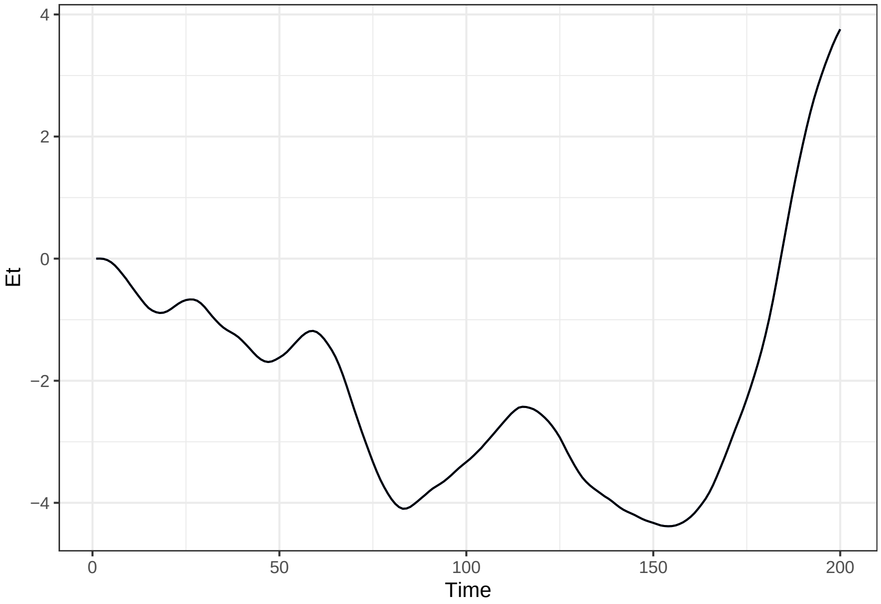
```
## Error in mcmc_disturbance_pois_solow(Y, X, QPrior = QPrior, v1Prior = v1Prior, : matrix multiplicati
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



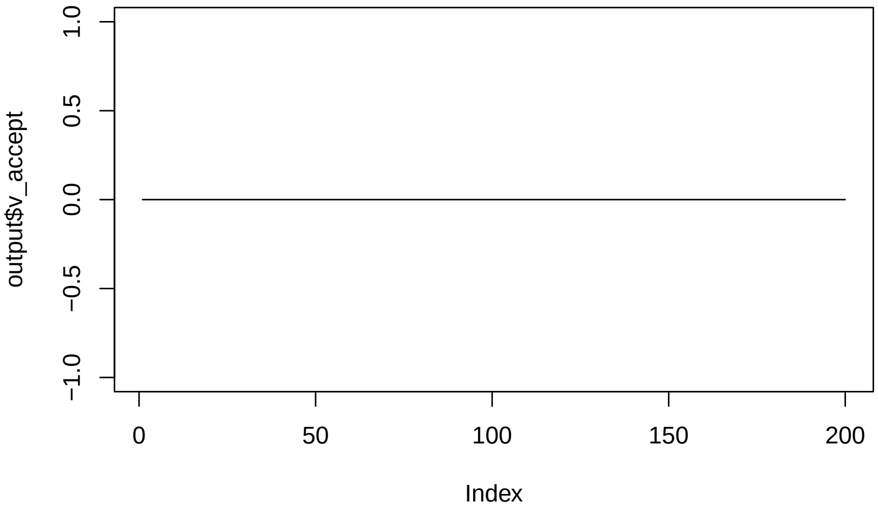
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



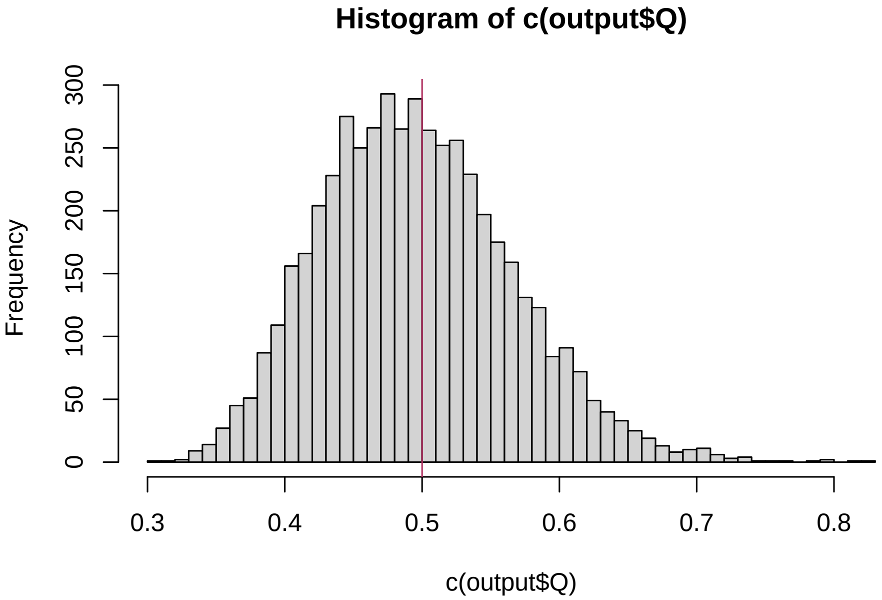
```
plot(output$v_accept,type="l")
```



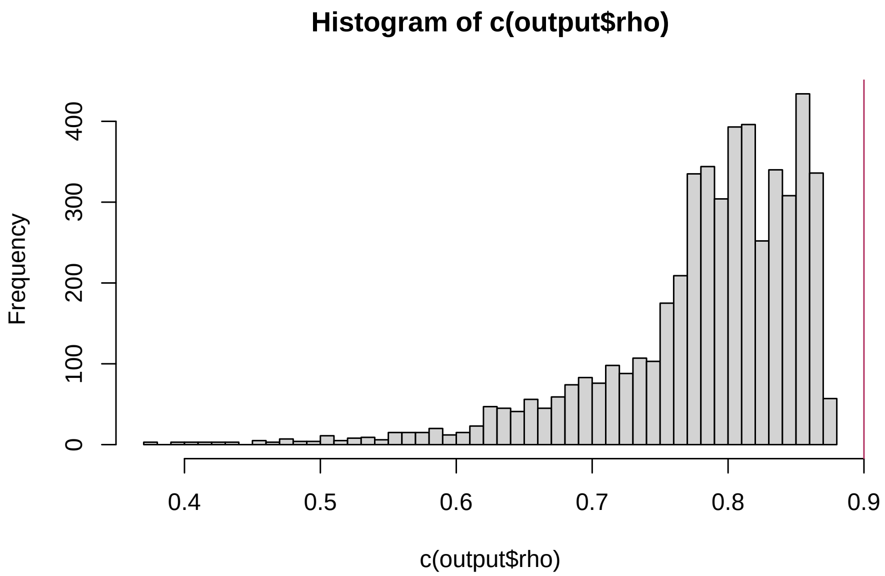
Infer all

```
output = mcmc_disturbance_pois_solow(Y,X,
                                     Vxi = Vxi,
                                     QPrior = QPrior,
                                     v1Prior = v1Prior,
                                     rho_init = rho,
                                     Q_init = Q,
                                     vt_init = v,
                                     nburnin = 10000,
                                     nthin = 2,
                                     nsample = 5000)
print(output$rho_accept)
```

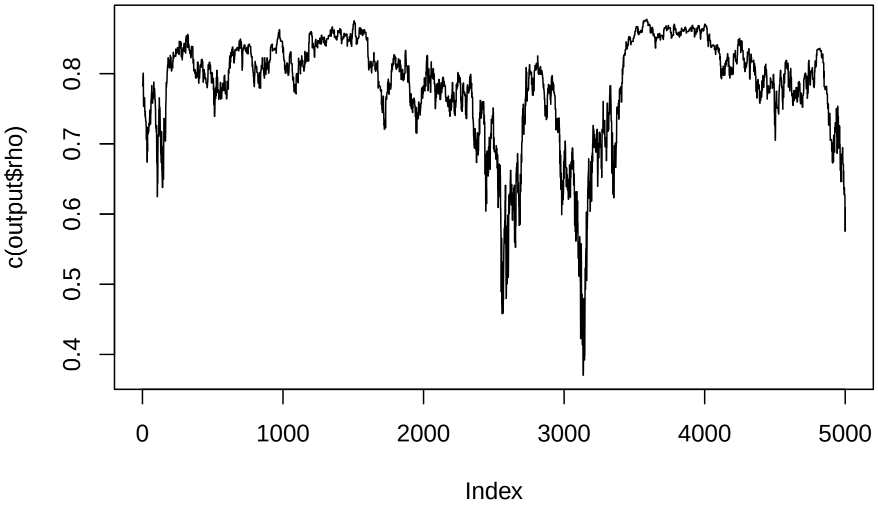
```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



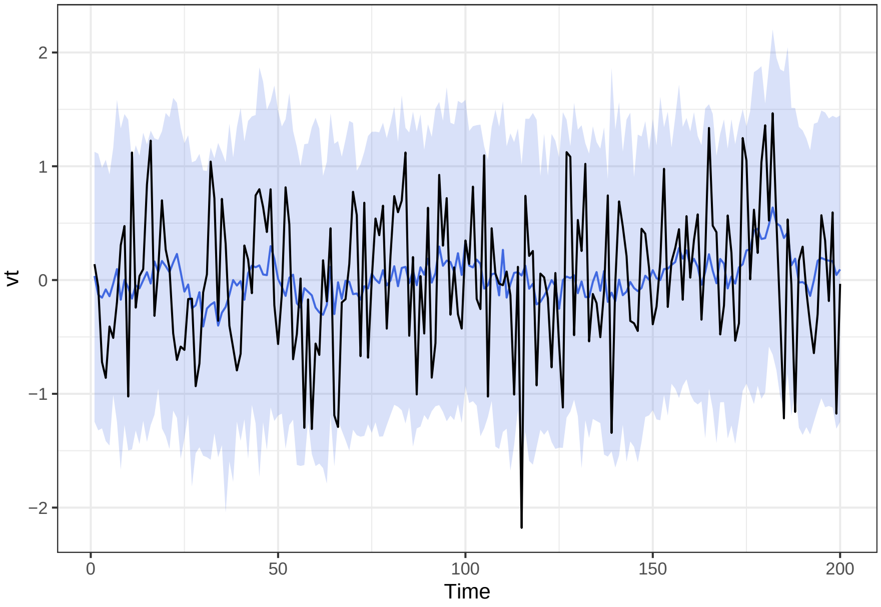
```
hist(c(output$rho),breaks=50)
abline(v=rho,col="maroon")
```



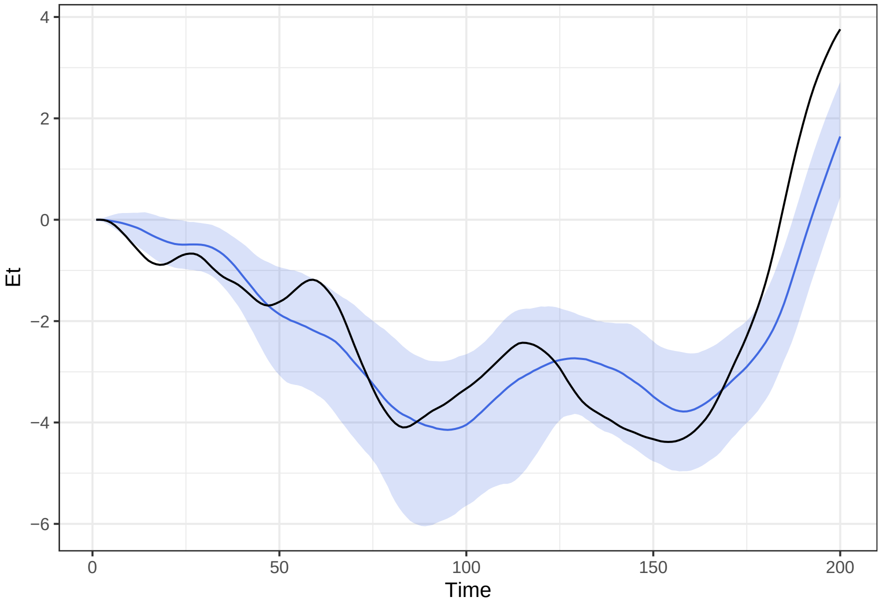
```
plot(c(output$rho),type="l")
```



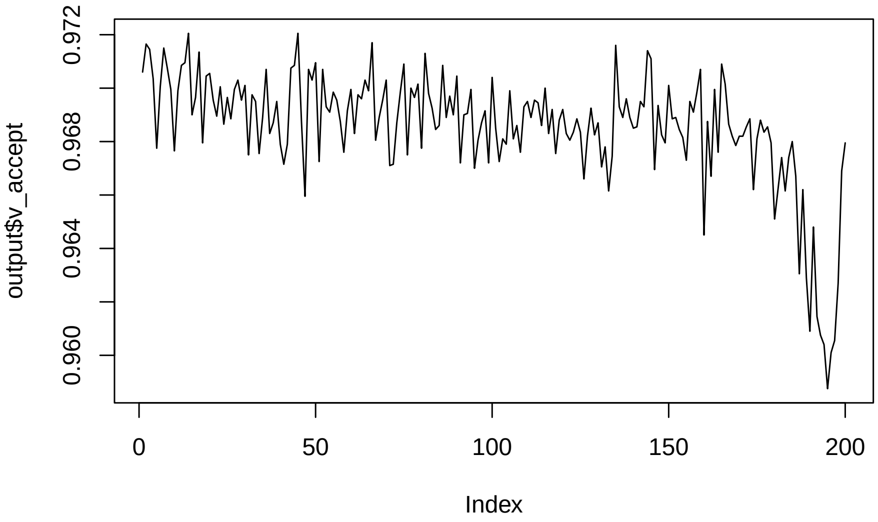
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v,Time=1:n))
```



```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



```
plot(output$v_accept,type="l")
```

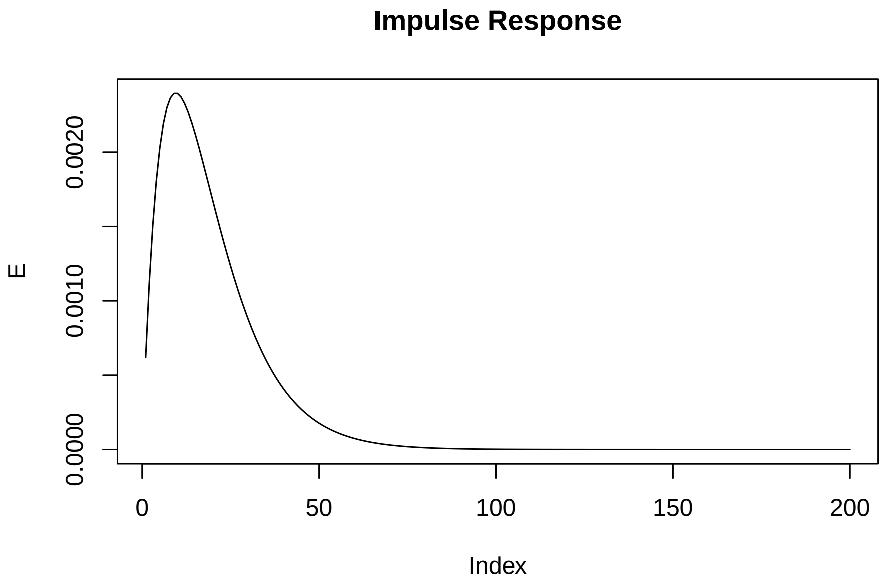


4. Identity Link + Pascal

Simulated Data

```
n = 200
rho = 0.9
Q =0.01
r = 2
```

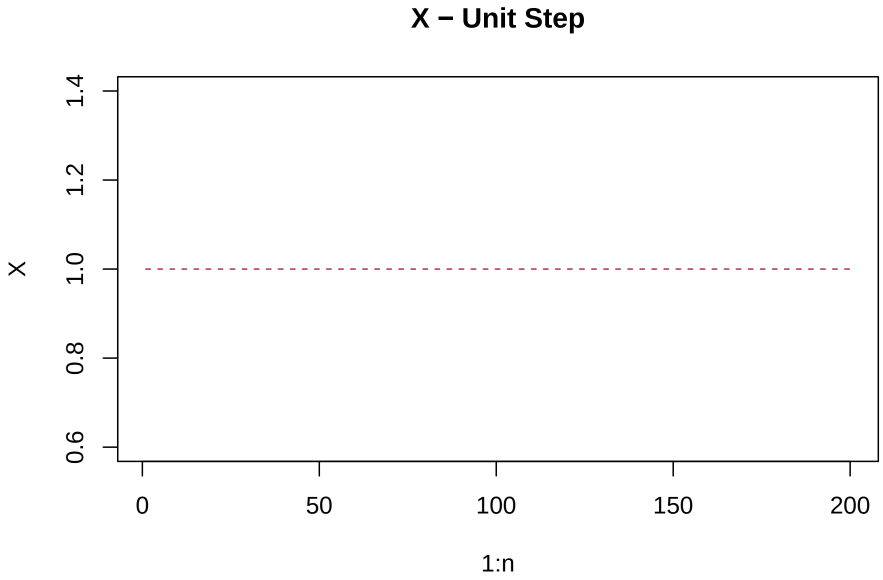
```
# X = sim.rw(n, 0, 0, 0.15)
v = rnorm(n,mean=0,sd=sqrt(Q))
beta = cumsum(v)
beta[beta<0] = 0

X = c(1,rep(0,n-1))
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*beta[t]*X[t]
}
E = E[-c(1:2)]
plot(E,type="l",main="Impulse Response")
```



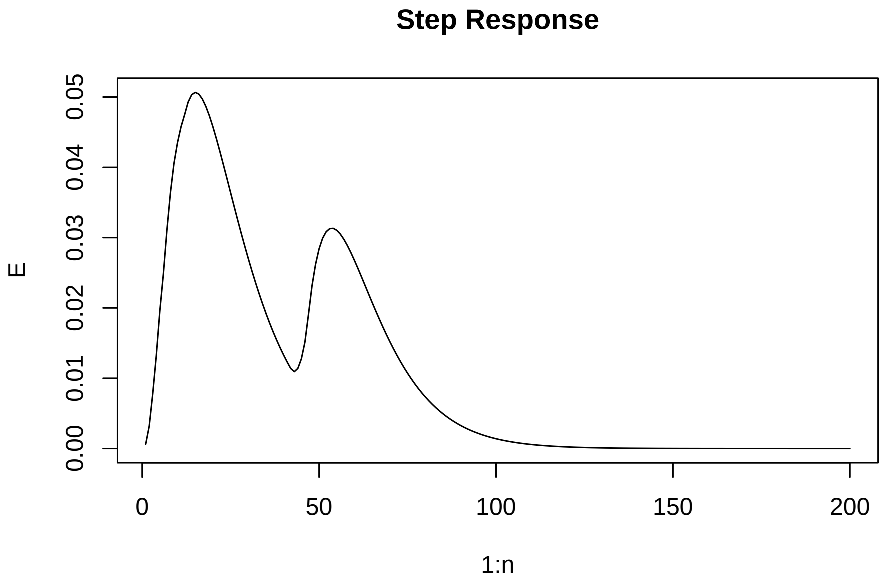
```
X = rep(1,n)
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*beta[t]*X[t]
}
E = E[-c(1:2)]

print(all(E>=0))
```

```
## [1] TRUE
```

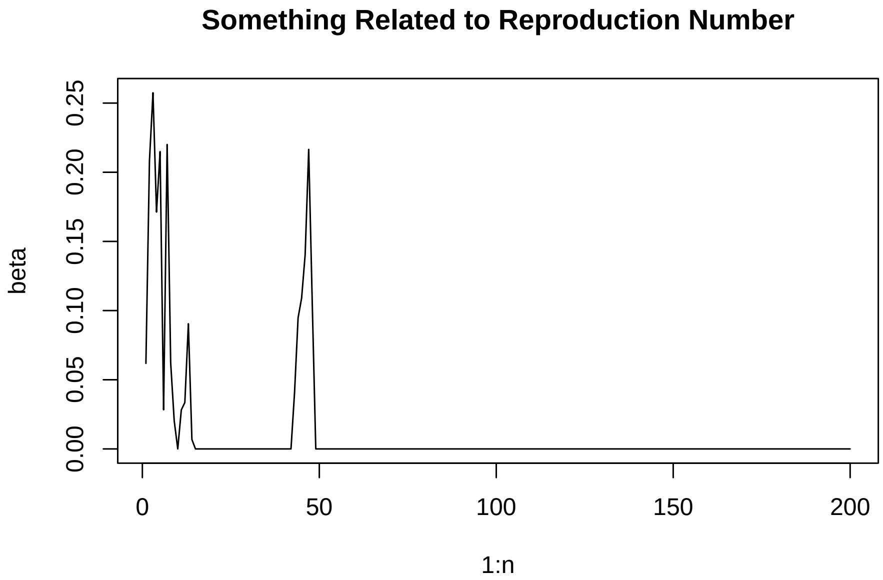
```
# lambda = exp(E)
Y = rpois(n,E)

plot(1:n,X,type="l",col="maroon",lty=2, main="X - Unit Step")
```



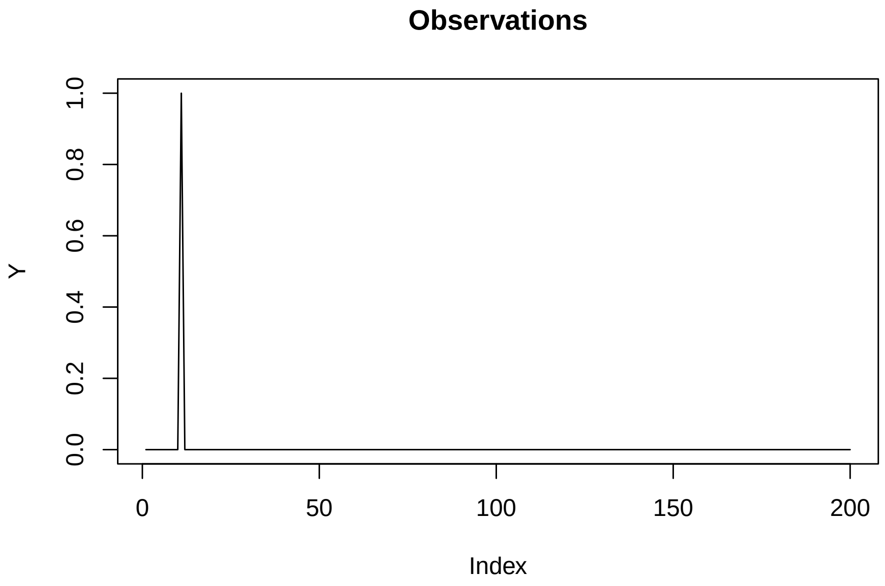
```
plot(1:n,E,type="l", main="Step Response")
```



```
plot(1:n,beta,type="l", main="Something Related to Reproduction Number")
```



```
# plot(1:n,v,type="l", main="Systematic Error / Disturbance")
plot(Y,type="l", main="Observations")
```



5. Identity Link + Shifted Pascal

Simulated Data

$y_t \sim \text{Pois}(\lambda_t = \theta_t).$

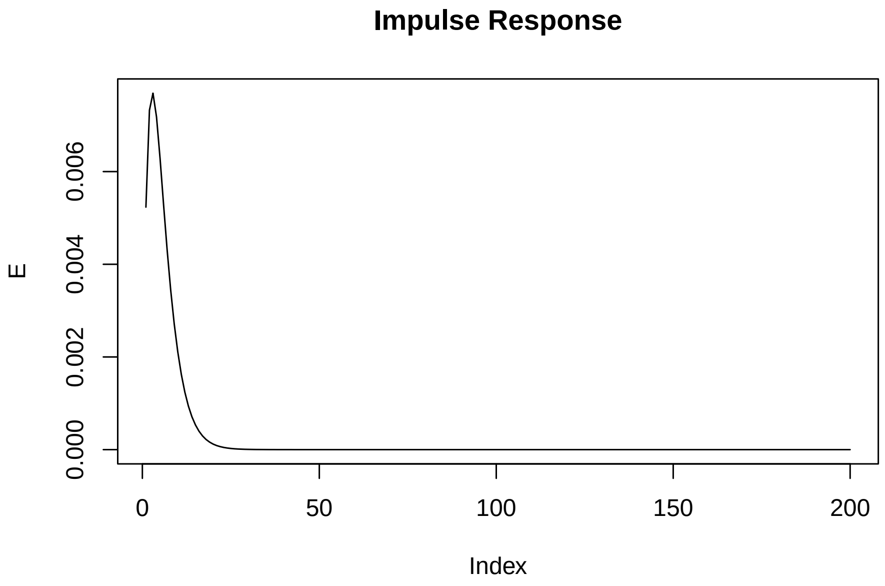
$\theta_t = 2\rho\theta_{t-1} - \rho^2\theta_{t-2} + (1-\rho)^2 \max(\beta_t, 0) X_{t-1}.$

$\beta_t = \beta_{t-1} + w_t.$

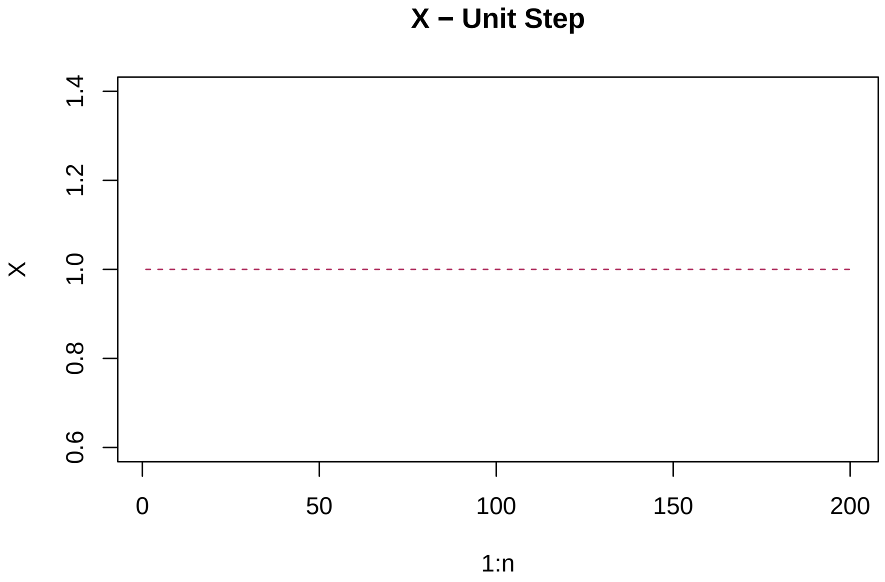
$w_t \sim N(0, W).$

```
n = 200
rho = 0.7
Q =0.01
r = 2
```

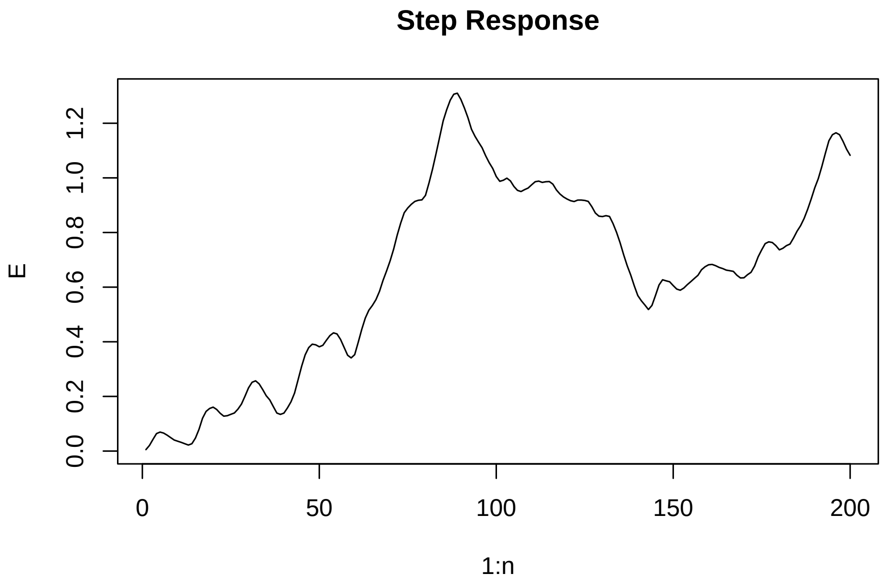
```
# X = sim.rw(n, 0, 0, 0.15)
v = rnorm(n+1,mean=0,sd=sqrt(Q))
beta = cumsum(v)
beta[beta<.Machine$double.eps] = .Machine$double.eps
# beta = c(0,beta)

X = c(1,rep(0,n-1))
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*beta[t]*X[t]
}
E = E[-c(1:2)]
plot(E,type="l",main="Impulse Response")
```



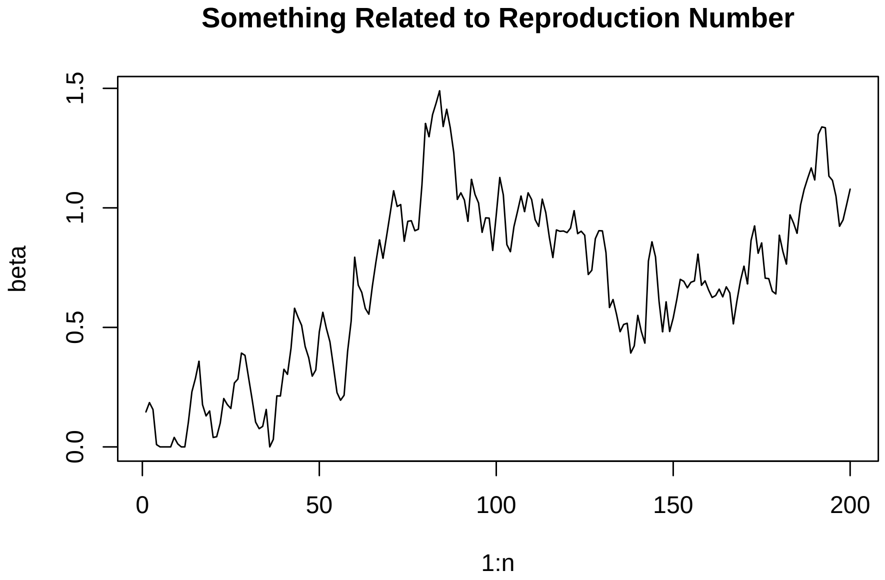
```
X = rep(1,n)
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*beta[t]*X[t]
}
E = E[-c(1:2)]
beta = beta[-1]

# lambda = exp(E)
Y = rpois(n,E)

plot(1:n,X,type="l",col="maroon",lty=2, main="X - Unit Step")
```



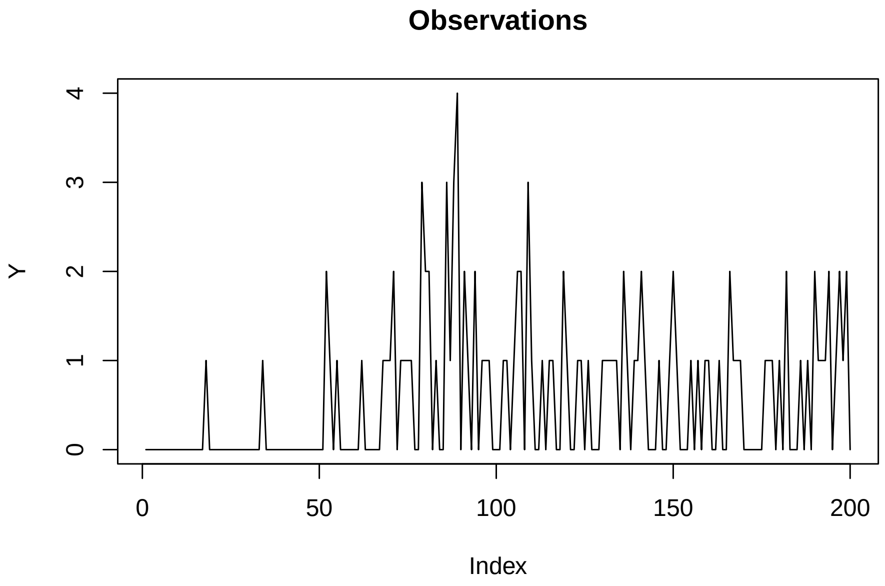
```
plot(1:n,E,type="l", main="Step Response")
```



```
plot(1:n,beta,type="l", main="Something Related to Reproduction Number")
```

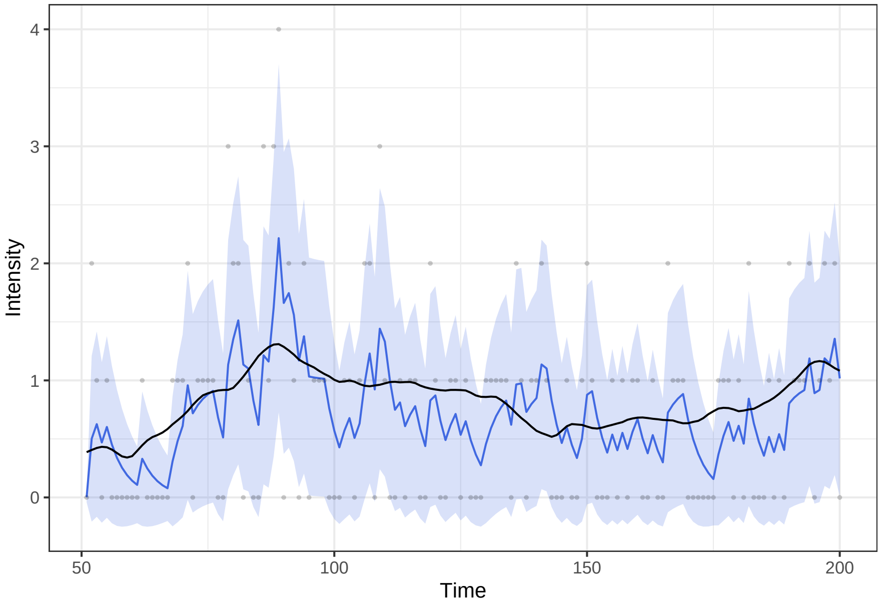


```
# plot(1:n,v,type="l", main="Systematic Error / Disturbance")
plot(Y,type="l", main="Observations")
```

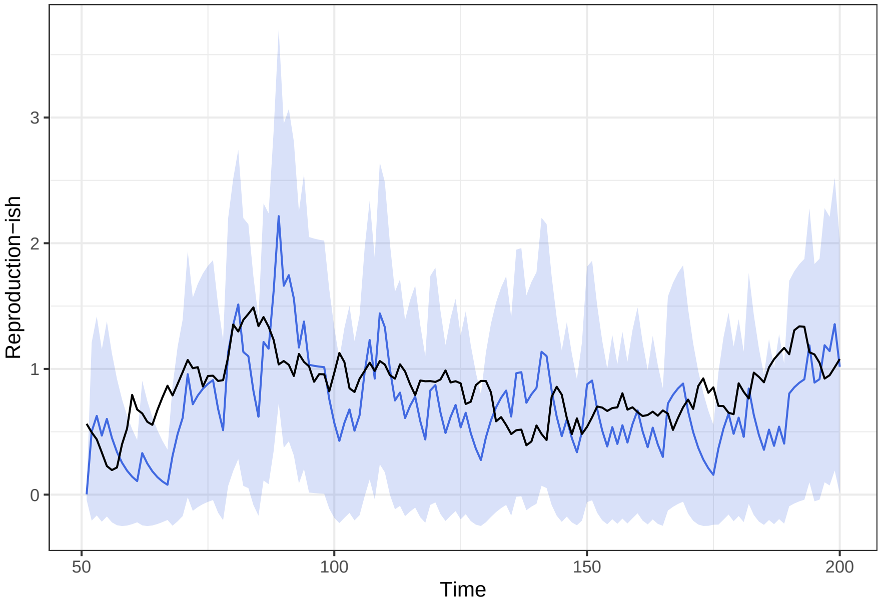


Linear Bayes Filtering

```
m0 = c(0,0,0)
C0 = diag(c(0.01,0.01,0.01))

rho_hat = 0.7
delta = 0.75

output = lbe_poissonSolowIdentity(Y,X_,rho_hat,delta,m0,C0)

ts = 50
tmp = data.frame(time=(ts+1):n, true=E[-c(1:ts)],y=Y[-c(1:ts)],
                 mt=c(output$mt[1,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[1,-c(1:(ts+1))])-2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[1,-c(1:(ts+1))])+2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.2,size=0.5) +
  xlab("Time") + ylab("Intensity")
```



```r
tmp = data.frame(time=(ts+1):n, true=beta[-c(1:ts)],
                 mt=c(output$mt[3,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[3,-c(1:(ts+1))])-2*sqrt(c(output$Ct[3,3,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[3,-c(1:(ts+1))])+2*sqrt(c(output$Ct[3,3,-c(1:(ts+1))])))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction-ish")
```



MCMC reparameterisation

Settings

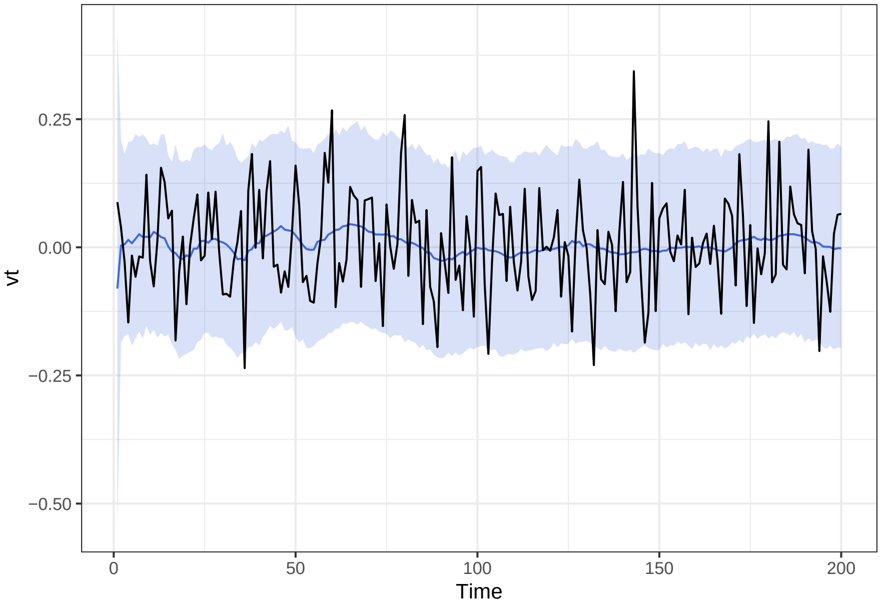
```
v1Prior = c(0,0.5)
QPrior=c(1e-2,1)
Vxi = 1
Fx = update_Fx_Solow(n,rho,X)
```

```
wt_hat = diff(output$mt[3,-1])
wt_hat[abs(wt_hat)>1] = 1
What = estimate_state_var(wt_hat,QPrior)
```

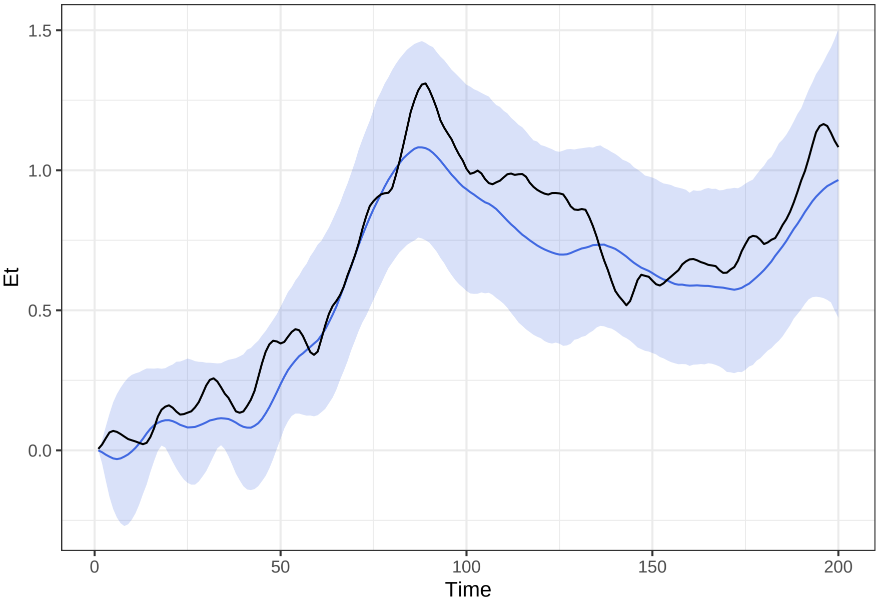
Infer w: W and rho is known

```
output = mcmc_disturbance_pois_solow_eye(Y,X,
                                         v1Prior = v1Prior,
                                         vt_init = v[-1],
                                         rho_true = rho,
                                         Q_true = Q,
                                         nburnin = 10000,
                                         nthin = 2,
                                         nsample = 5000)
```

```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v[-1],Time=1:n))
```

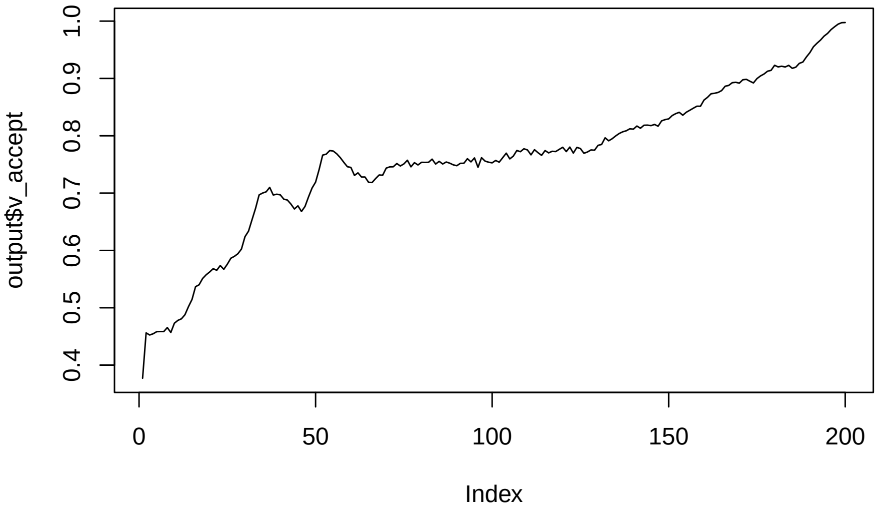


```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
            alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
           data=data.frame(true=E,time=1:n))
```



Still looks good,

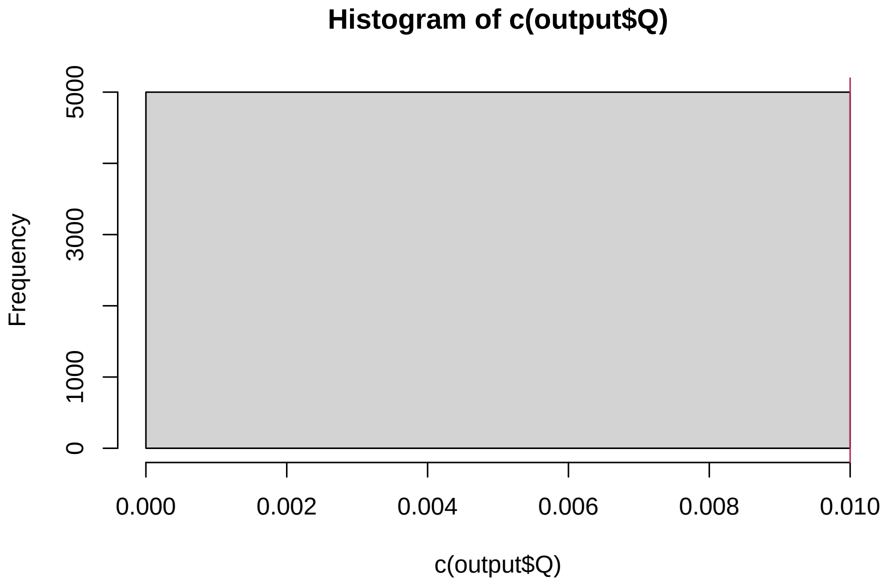
```
plot(output$v_accept,type="l")
```



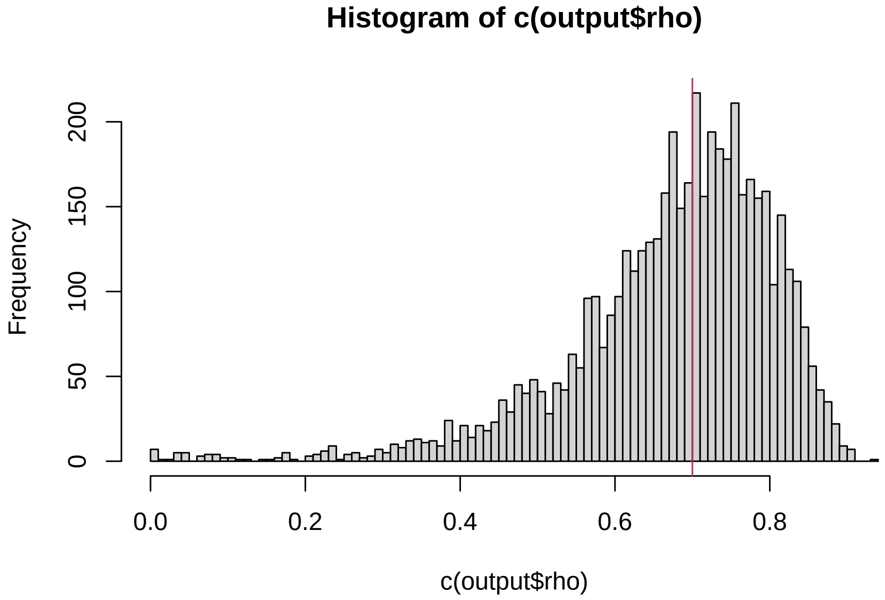
Infer rho: others are known

```
output = mcmc_disturbance_pois_solow_eye(Y,X,
                                         Vxi = Vxi,
                                         rho_init = rho,
                                         Q_true = Q,
                                         vt_true = v[-n],
                                         nburnin = 10000,
                                         nthin = 2,
                                         nsample = 5000)
print(output$rho_accept)
```

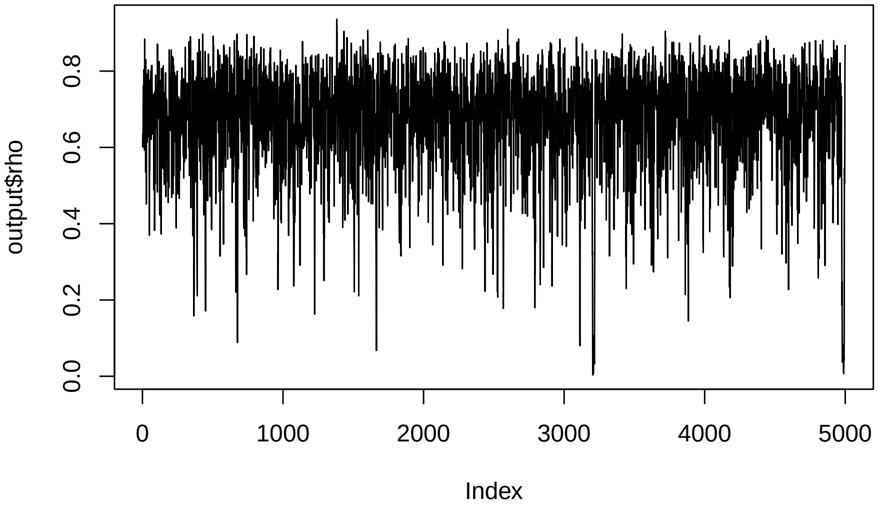
```
hist(c(output$Q),breaks=100)
abline(v=Q,col="maroon")
```



```
hist(c(output$rho),breaks=100)
abline(v=rho,col="maroon")
```



```
plot(output$rho,type="l")
```

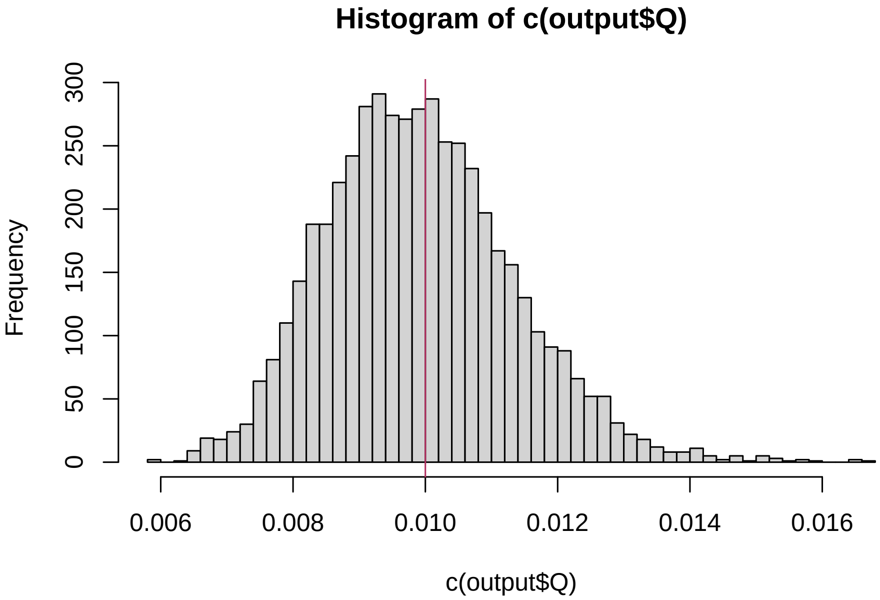


Infer w and W: rho is known

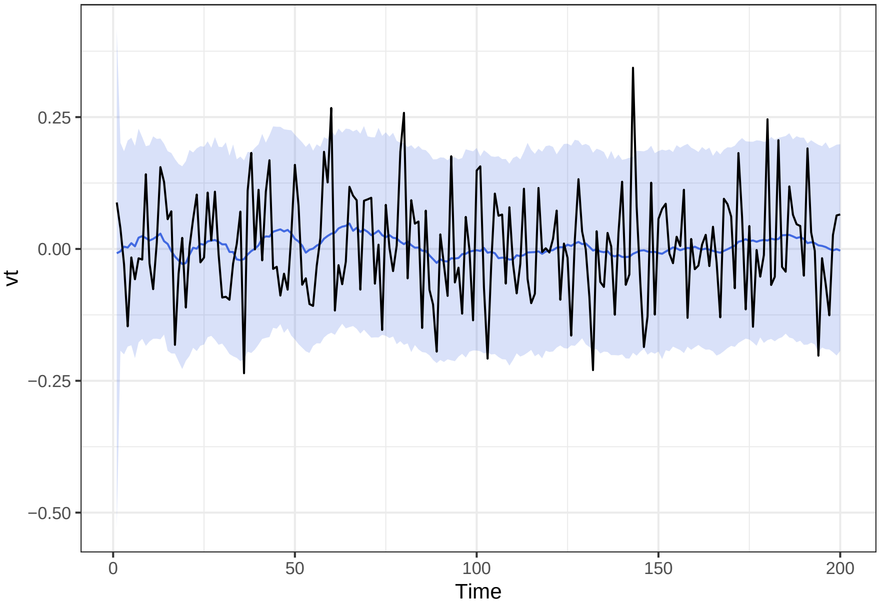
The MCMC sampler is sensitive to initial values.

```
output = mcmc_disturbance_pois_solow_eye(Y,X,
                                         QPrior = QPrior,
                                         v1Prior = v1Prior,
                                         Q_init = Q,
                                         vt_init = v[-n],
                                         rho_true = rho,
                                         nburnin = 10000,
                                         nthin = 2,
                                         nsample = 5000)
```

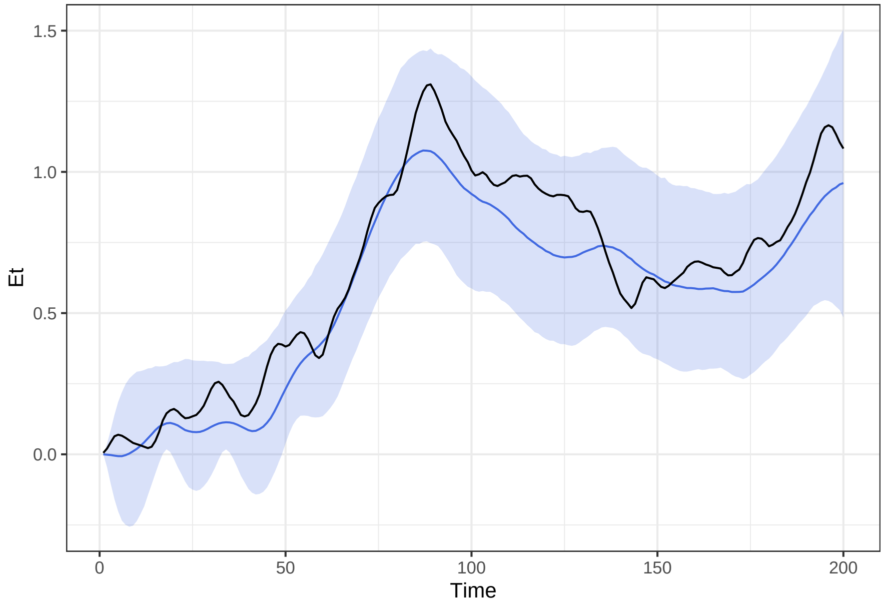
```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



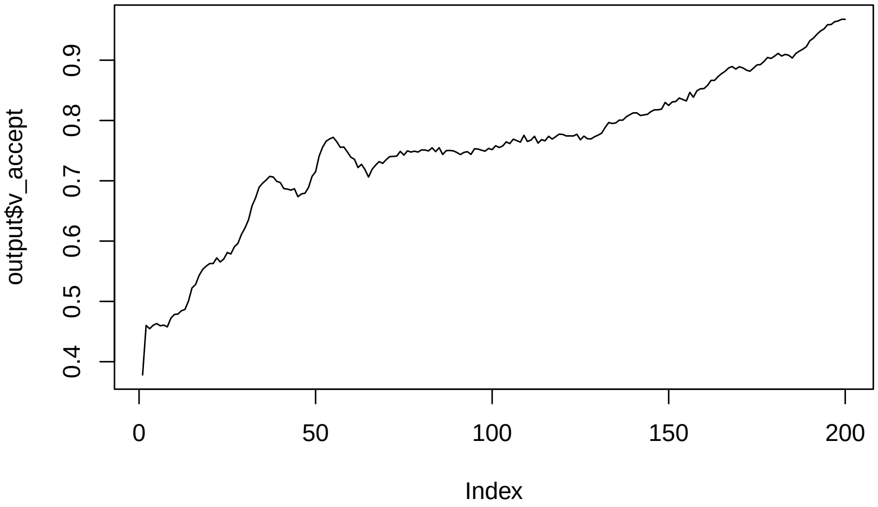
```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v[-1],Time=1:n))
```



```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



```
plot(output$v_accept,type="l")
```

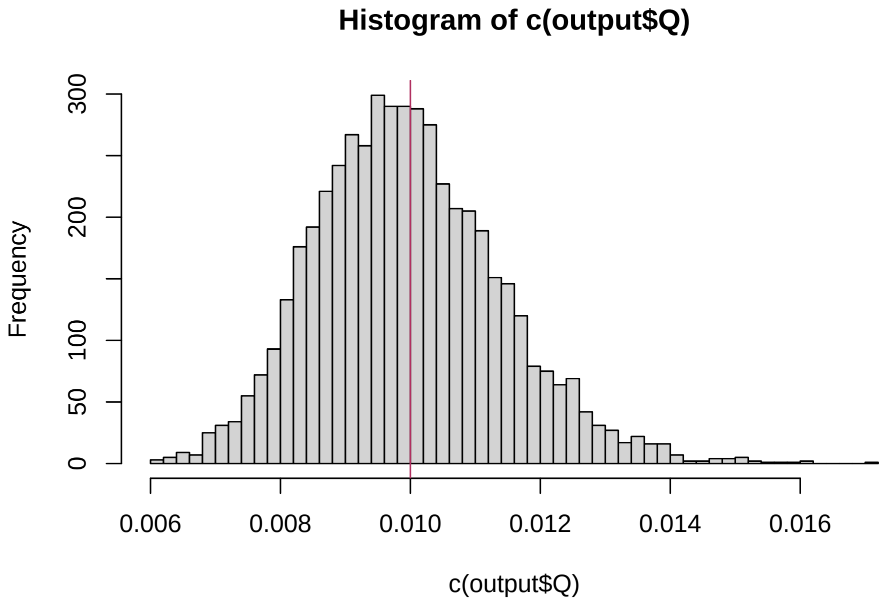


Infer all: initialize to true values

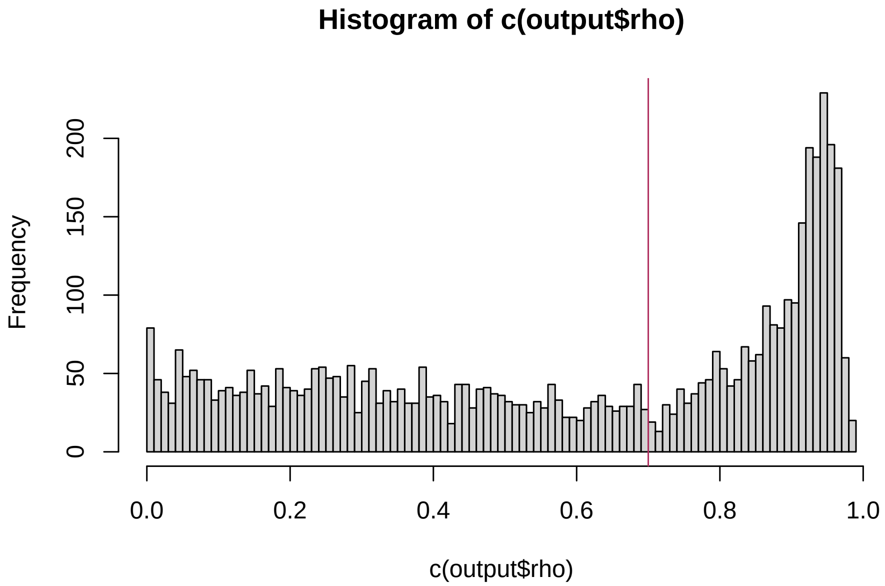
The MCMC sampler is sensitive to initial values.

```
output = mcmc_disturbance_pois_solow_eye(Y,X,
                                          Vxi = Vxi,
                                          QPrior = QPrior,
                                          v1Prior = v1Prior,
                                          Q_init = Q,
                                          vt_init = v[-n],
                                          rho_init = rho,
                                          nburnin = 10000,
                                          nthin = 2,
                                          nsample = 5000)
print(output$rho_accept)
```

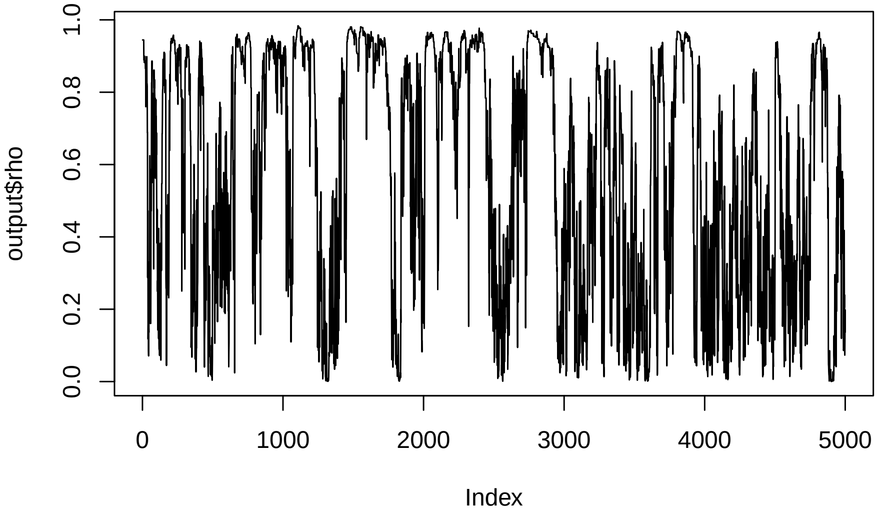
```
hist(c(output$Q),breaks=50)
abline(v=Q,col="maroon")
```



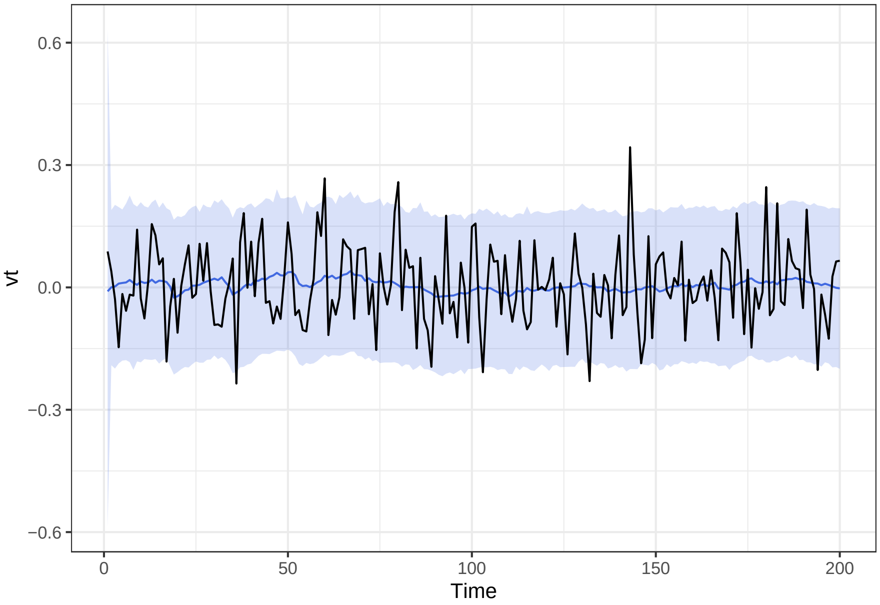
```
hist(c(output$rho),breaks=100)
abline(v=rho,col="maroon")
```



```
plot(output$rho,type="l")
```

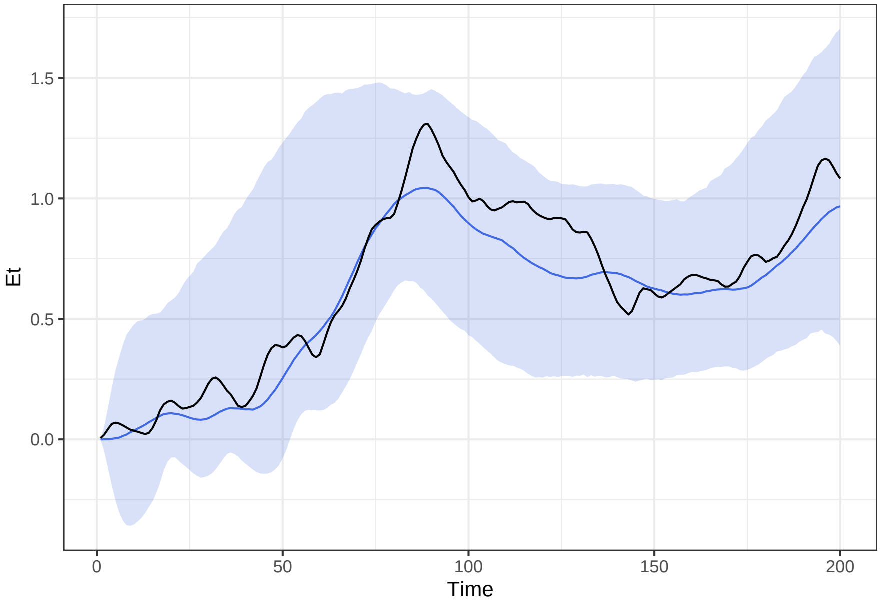


```
vt_est = t(apply(output$v,1,quantile,c(0.025,0.5,0.975)))
vt_est = as.data.frame(vt_est)
colnames(vt_est) = c("lobnd", "vt", "hibnd")
vt_est$Time = 1:n
ggplot(vt_est,aes(x=Time)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=vt),color="royalblue") +
  geom_line(aes(y=True),data=data.frame(True=v[-1],Time=1:n))
```

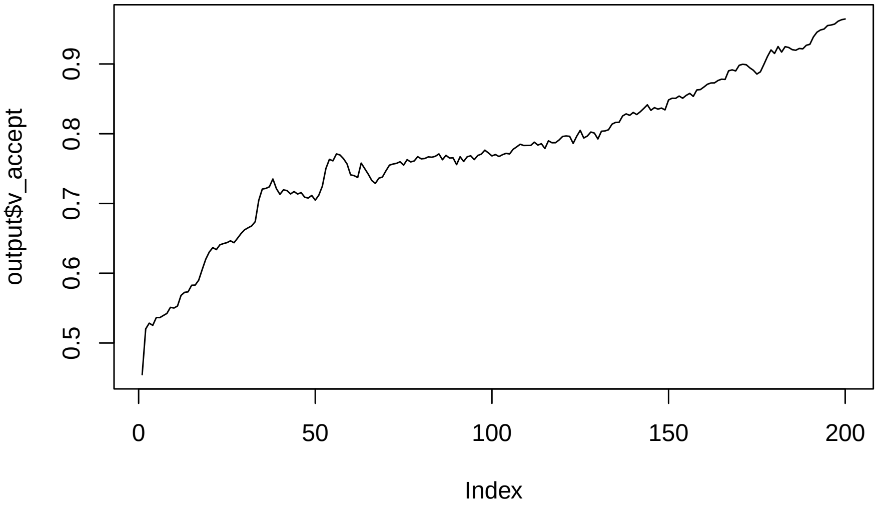


```
Et_est = matrix(0,nrow=n,ncol=5000)
for (i in 1:5000) {
  Et_est[,i] = c(Fx %*% as.matrix(output$v[,i],ncol=1))
}
Et_est = apply(Et_est,1,quantile,c(0.025,0.5,0.975))
Et_est = as.data.frame(t(Et_est))
colnames(Et_est) = c("lobnd","Et","hibnd")
Et_est$Time = 1:n
```

```
ggplot(Et_est,aes(x=Time,y=Et)) + theme_bw() +
  geom_ribbon(aes(ymin=lobnd,ymax=hibnd),
              alpha=0.2,fill="royalblue") +
  geom_line(color="royalblue") +
  geom_line(aes(x=time,y=true),
            data=data.frame(true=E,time=1:n))
```



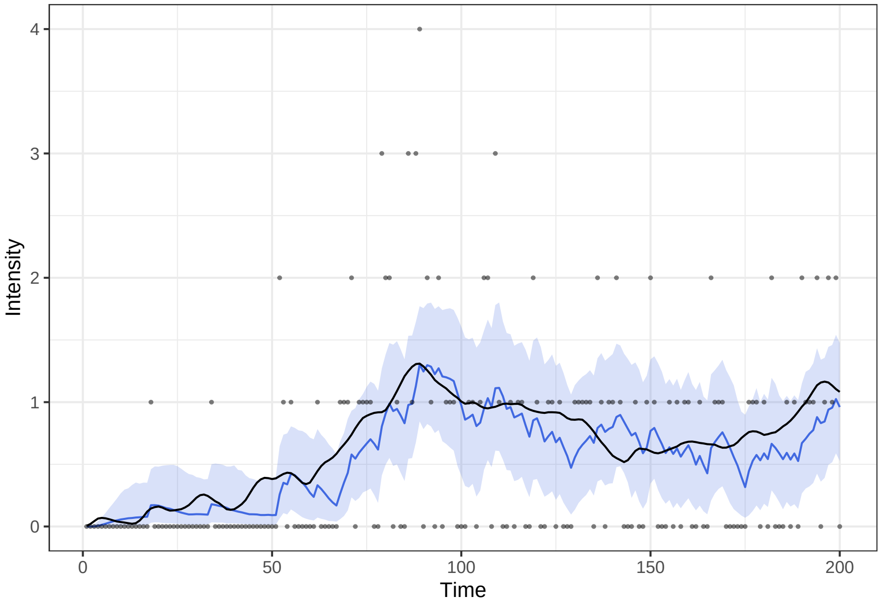
```
plot(output$v_accept,type="l")
```



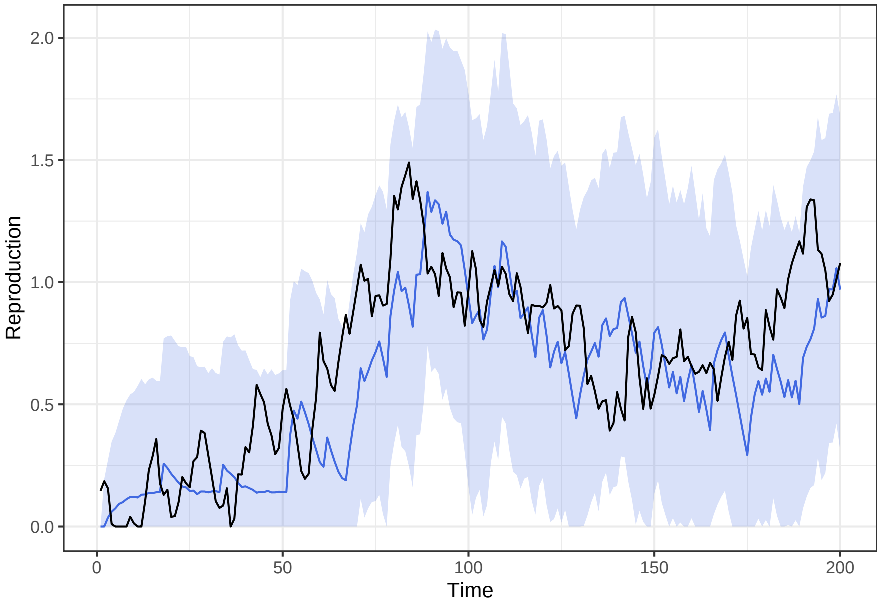
Particle Filtering

```
output = bf_pois_solow_eye_max(Y,X,rho,Q,N=5000)

tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))
```

```
ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```

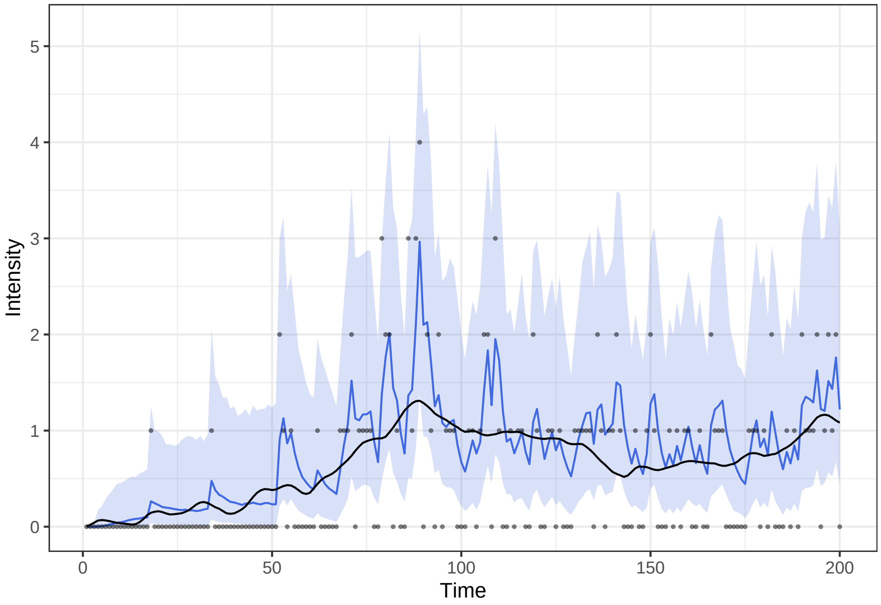


```
tmp = data.frame(time=1:n, true=beta,
                 mt = apply(output$beta,2,quantile,0.5),
                 mt_hi = apply(output$beta,2,quantile,0.025),
                 mt_lo = apply(output$beta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```

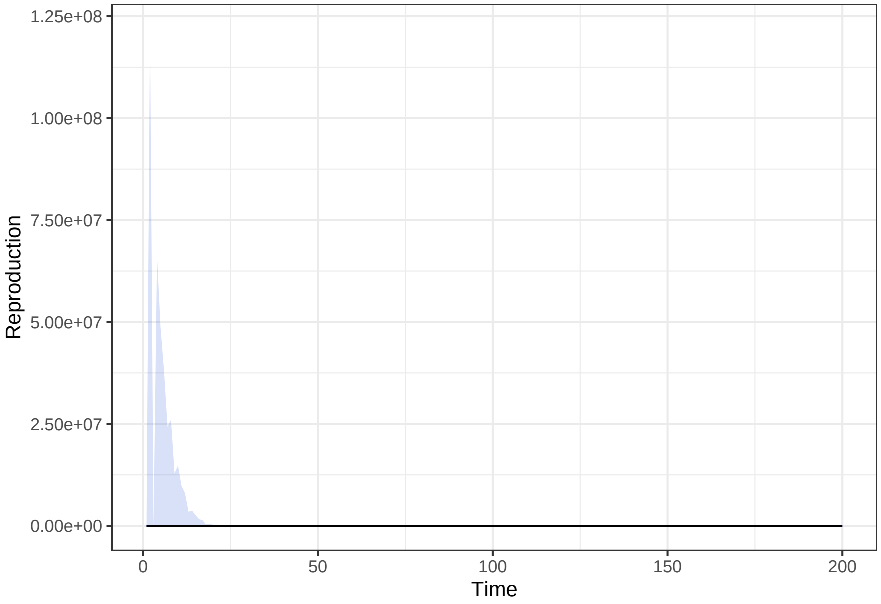


Particle Learning

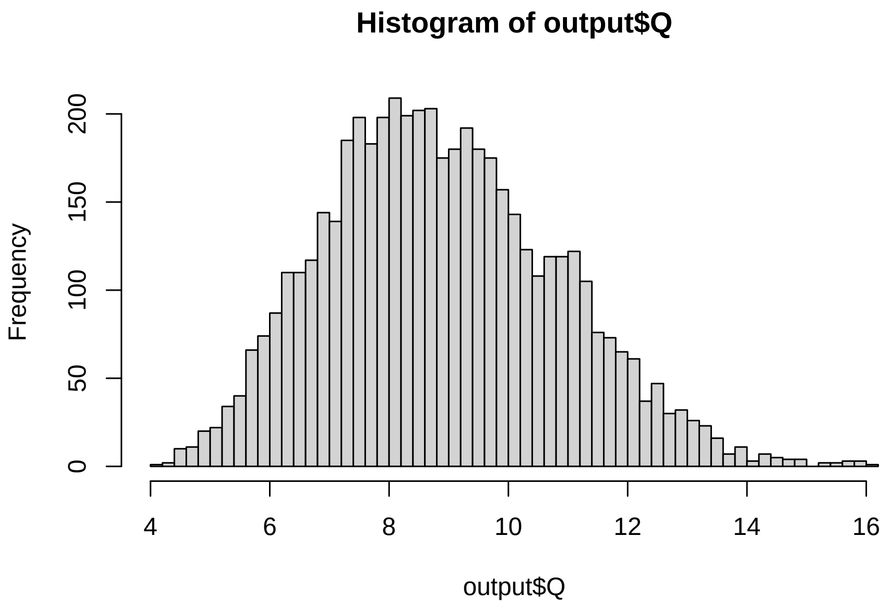
```
m0 = rep(0,3)
C0 = diag(rep(10,3))
output = sf_pois_solow_eye_max(Y,X,rho,
                               QPrior=c(1e-2,1e-2),
                               Q_init=Q,
                               N=5000)

tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```



```
tmp = data.frame(time=1:n, true=beta,
                 mt = apply(output$beta,2,quantile,0.5),
                 mt_hi = apply(output$beta,2,quantile,0.025),
                 mt_lo = apply(output$beta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```



```
hist(output$Q,breaks=50)
abline(v=Q,col="maroon")
```



6. Identity Link + Nonlinear SS + Shifted Pascal

Simulated Data

Model

$y_t \sim Pois(\lambda_t = exp(\theta_t))$.

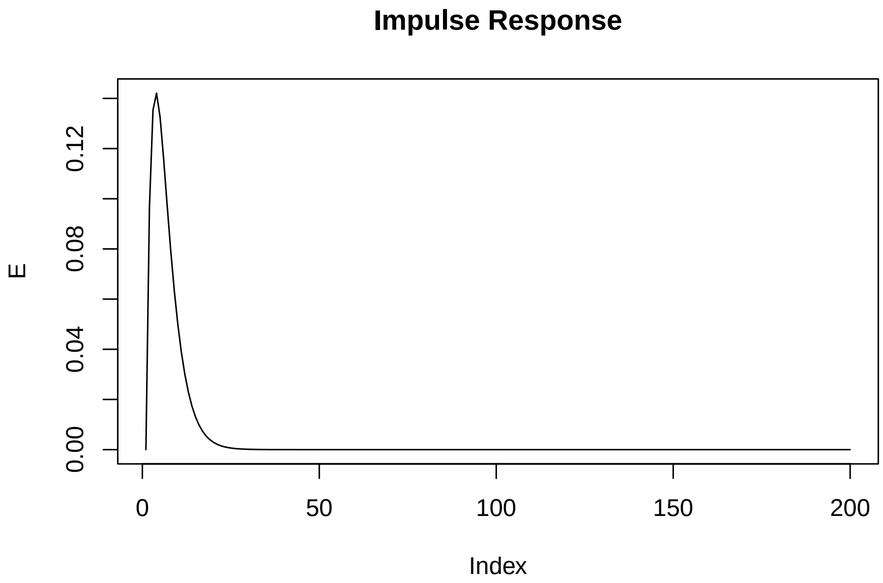
$\theta_t = 2\rho\theta_{t-1} - \rho^2\theta_{t-2} + (1-\rho)^2 \exp(\beta_t) X_{t-1}$,

$\beta_t = \beta_{t-1} + w_t$,

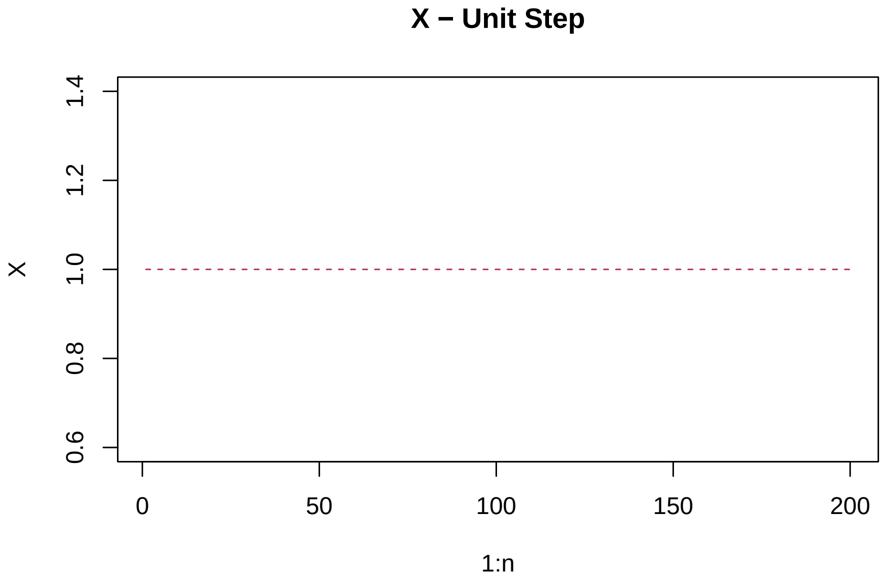
$w_t \sim N(0, W)$.

```
n = 200
rho = 0.7
Q =0.01
r = 2
```

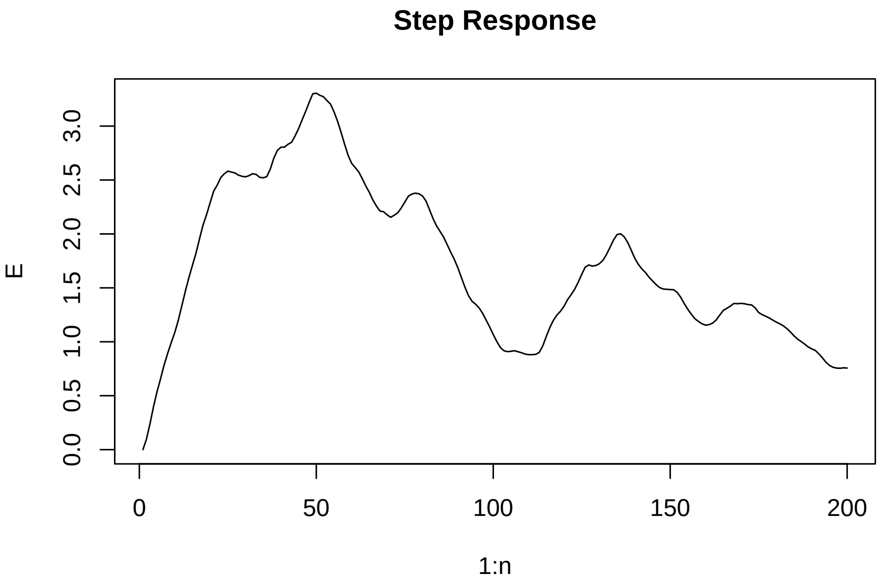
```
# X = sim.rw(n, 0, 0, 0.15)
v = rnorm(n+1,mean=0,sd=sqrt(Q))
beta = cumsum(v)
# beta = c(0,beta)

X = c(1,rep(0,n-1))
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*exp(beta[t])*X_[t]
}
E = E[-c(1:2)]
plot(E,type="l",main="Impulse Response")
```



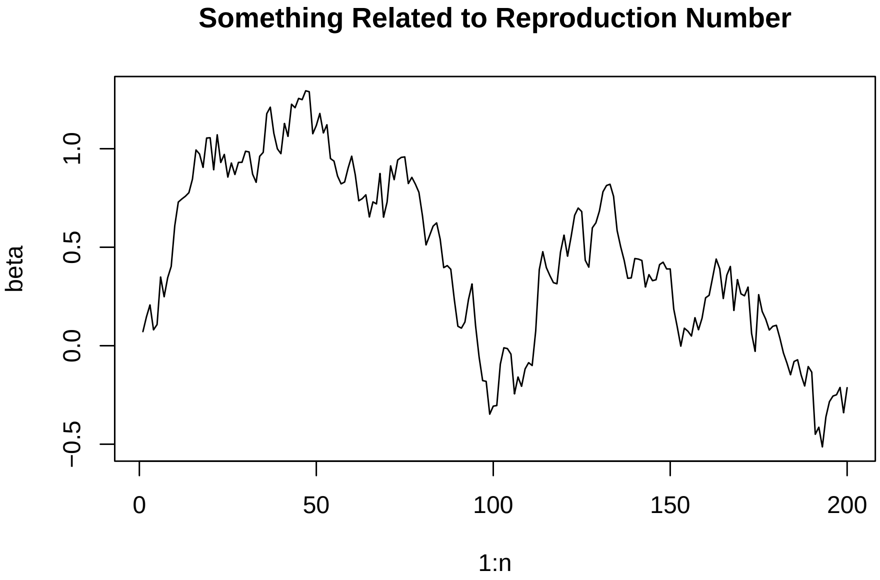
```
X = rep(1,n)
X_ = c(0,X)
E = rep(0,n+2)
for (t in 1:n) {
  E[t+2] = 2*rho*E[t+1] - rho^2*E[t] + (1-rho)^2*exp(beta[t])*X_[t]
}
E = E[-c(1:2)]
beta = beta[-1]

# lambda = exp(E)
Y = rpois(n,E)

plot(1:n,X,type="l",col="maroon",lty=2, main="X - Unit Step")
```



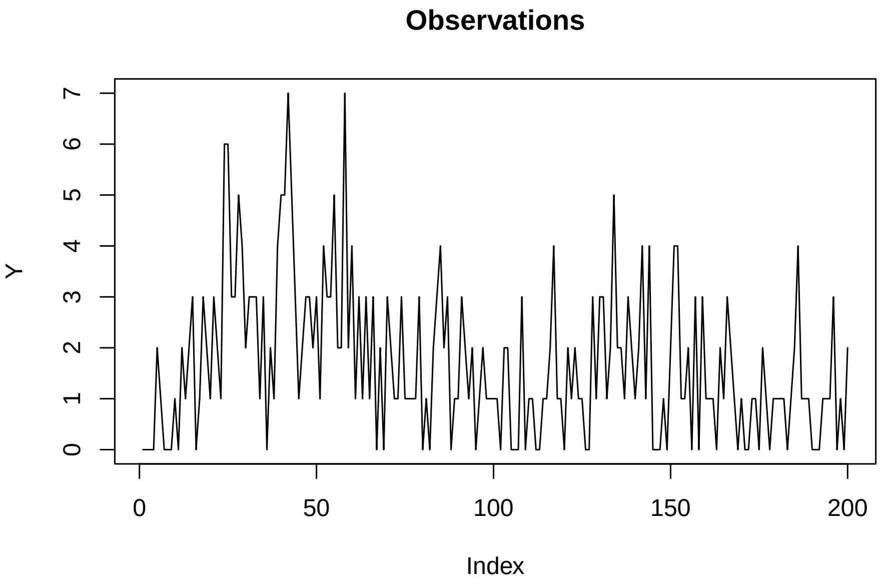
```
plot(1:n,E,type="l", main="Step Response")
```



```
plot(1:n,beta,type="l", main="Something Related to Reproduction Number")
```

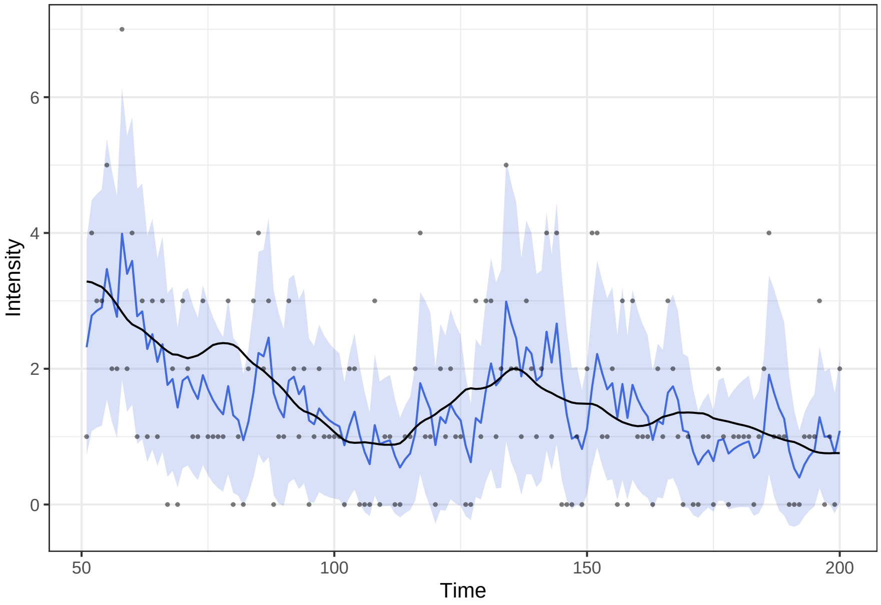


```
# plot(1:n,v,type="l", main="Systematic Error / Disturbance")
plot(Y,type="l", main="Observations")
```

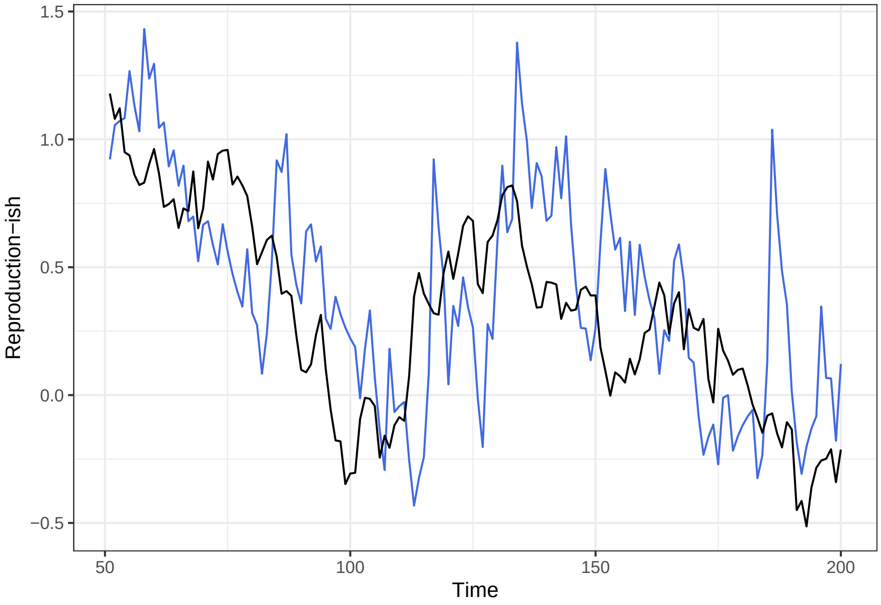


Linear Bayes Filtering

```
m0 = c(0,0,0)
C0 = diag(c(0.01,0.01,0.01))

rho_hat = 0.7
delta = 0.7

output = lbe_poissonSolowIdentity2(Y,X_,rho_hat,delta,m0,C0)

ts = 50
tmp = data.frame(time=(ts+1):n, true=E[-c(1:ts)],y=Y[-c(1:ts)],
                 mt=c(output$mt[1,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[1,-c(1:(ts+1))])-2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[1,-c(1:(ts+1))])+2*sqrt(c(output$Ct[1,1,-c(1:(ts+1))])))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```



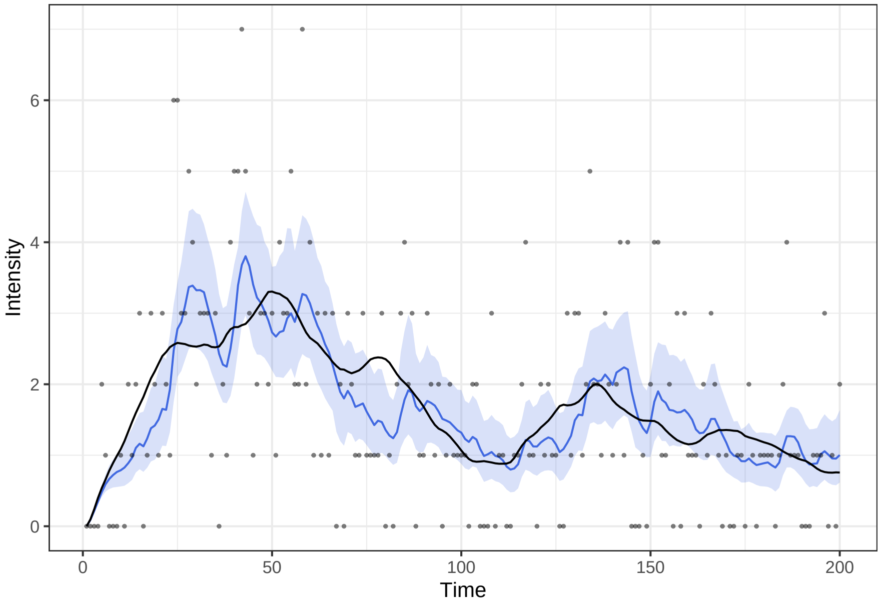
```
tmp = data.frame(time=(ts+1):n, true=beta[-c(1:ts)],
                 mt=c(output$mt[3,-c(1:(ts+1))]),
                 mt_lo=c(output$mt[3,-c(1:(ts+1))])-2*sqrt(c(output$Ct[3,3,-c(1:(ts+1))])),
                 mt_hi=c(output$mt[3,-c(1:(ts+1))])+2*sqrt(c(output$Ct[3,3,-c(1:(ts+1))])))

ggplot(tmp,aes(x=time)) +
  # geom_ribbon(aes(ymin=exp(mt_lo),ymax=exp(mt_hi)),
  #             fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction-ish")
```



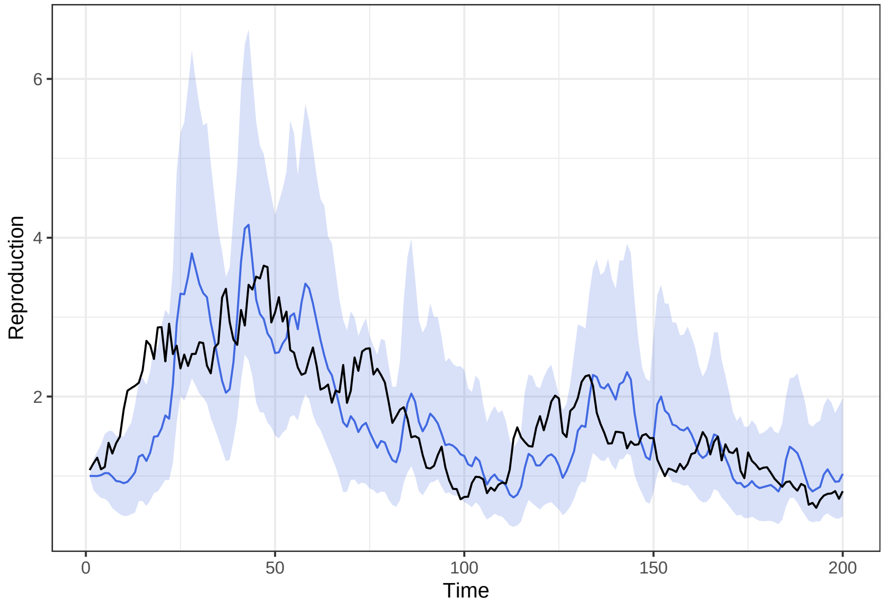
Particle Filtering

```
output = apf_pois_solow_eye_exp(Y,X,rho,Q,N=5000)
```

```
tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```

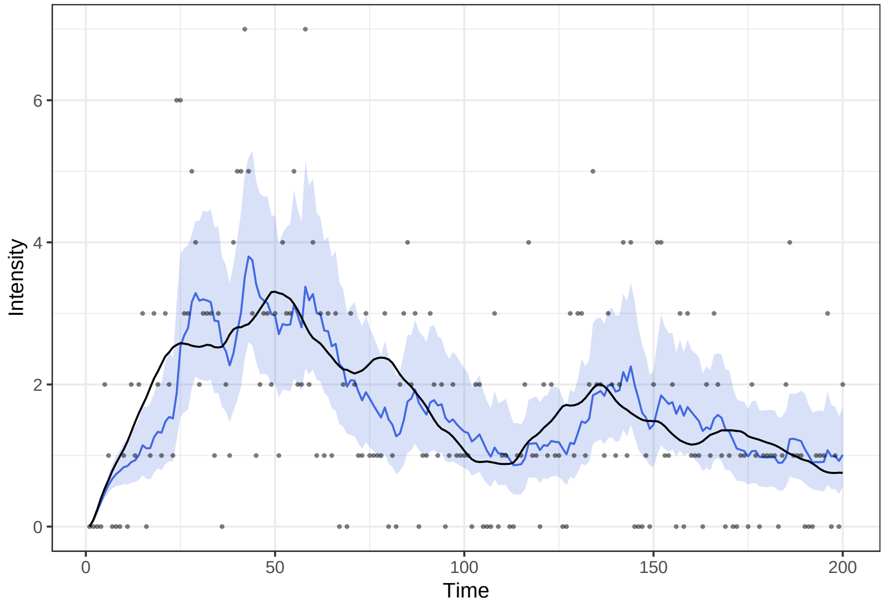


```
tmp = data.frame(time=1:n, true=exp(beta),
                 mt = apply(exp(output$beta),2,quantile,0.5),
                 mt_hi = apply(exp(output$beta),2,quantile,0.025),
                 mt_lo = apply(exp(output$beta),2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```

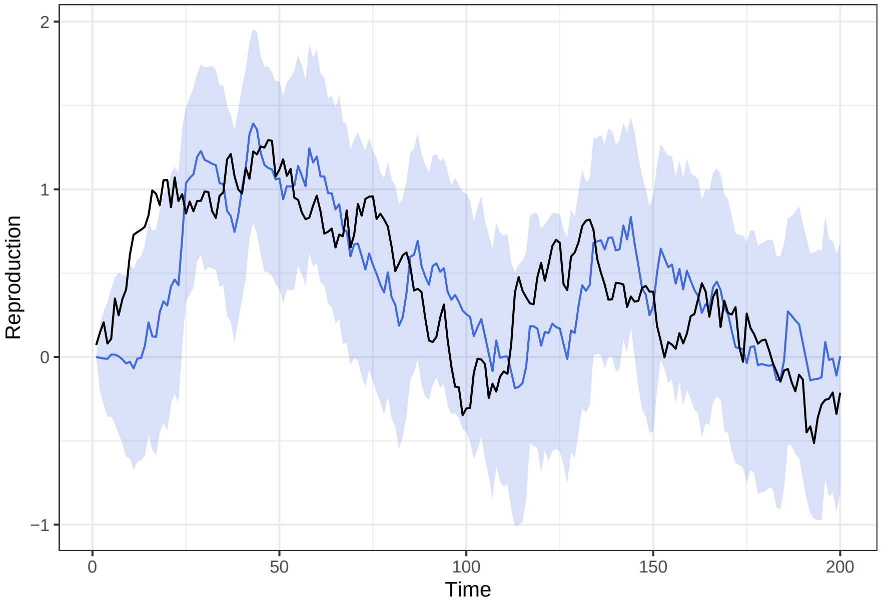


```
output = bf_pois_solow_eye_exp(Y,X,rho,Q,N=5000)
```

```
tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```



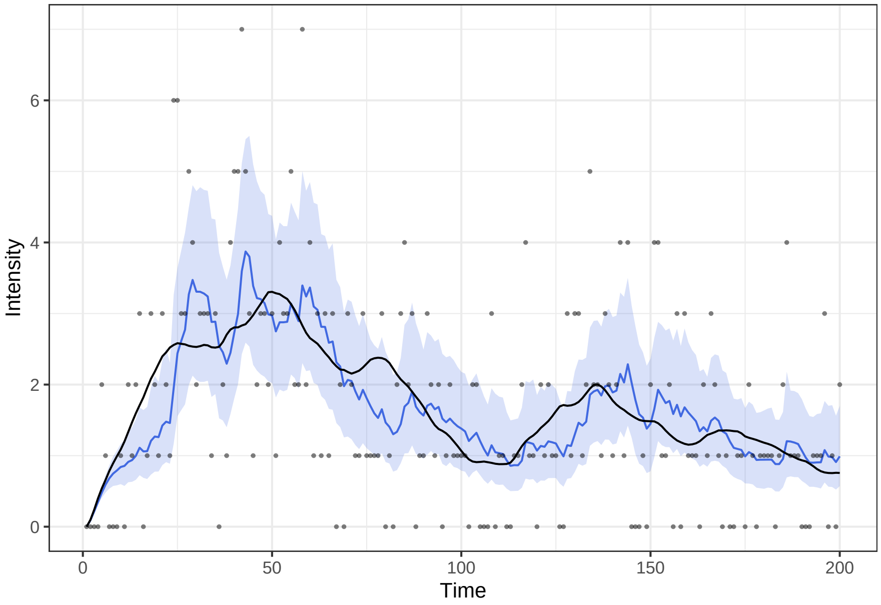
```
tmp = data.frame(time=1:n, true=beta,
                 mt = apply(output$beta,2,quantile,0.5),
                 mt_hi = apply(output$beta,2,quantile,0.025),
                 mt_lo = apply(output$beta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```



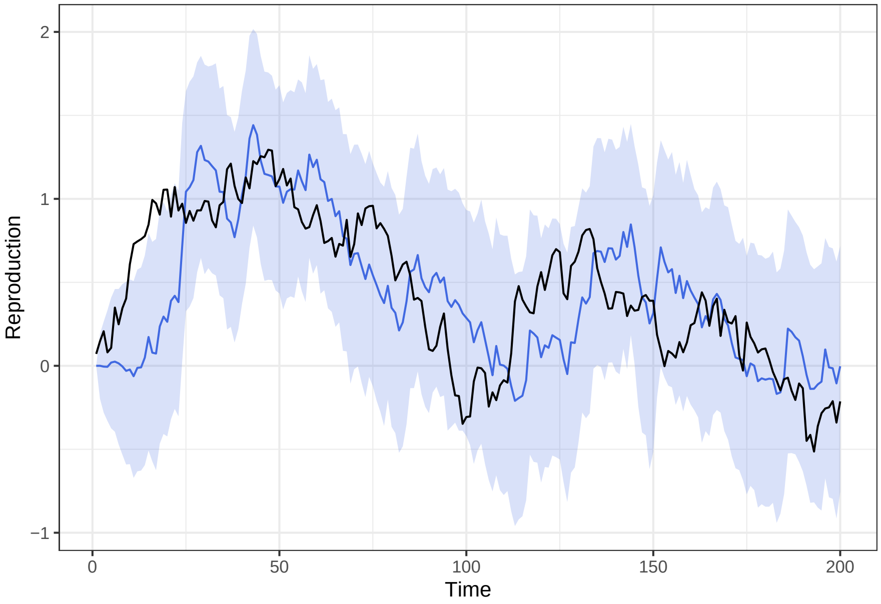
Particle Learning

```
m0 = rep(0,3)
C0 = diag(rep(10,3))
output = sf_pois_solow_eye_exp(Y,X,rho,
                               QPrior=c(1e-2,1e-2),
                               Q_true=Q,
                               N=5000)
```

```
tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```

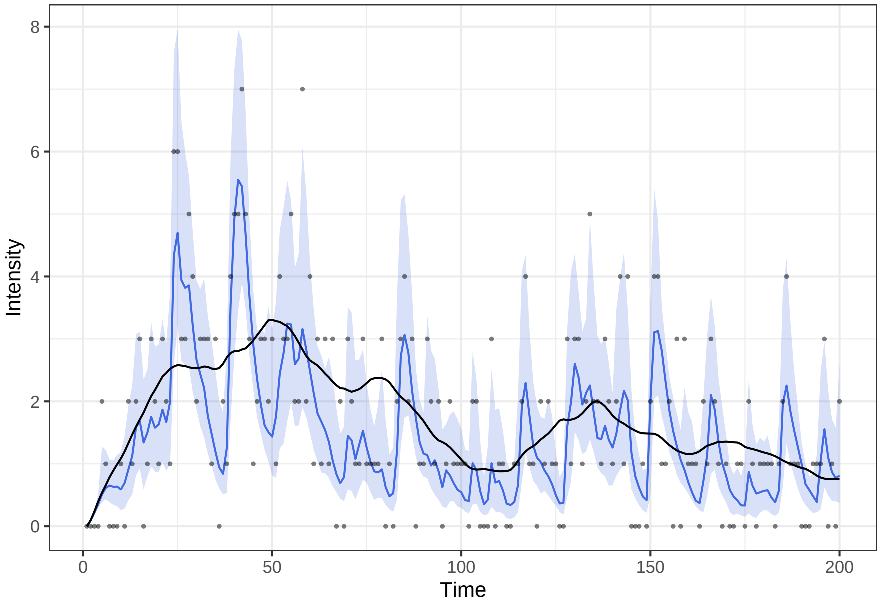


```
tmp = data.frame(time=1:n, true=beta,
                 mt = apply(output$beta,2,quantile,0.5),
                 mt_hi = apply(output$beta,2,quantile,0.025),
                 mt_lo = apply(output$beta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```

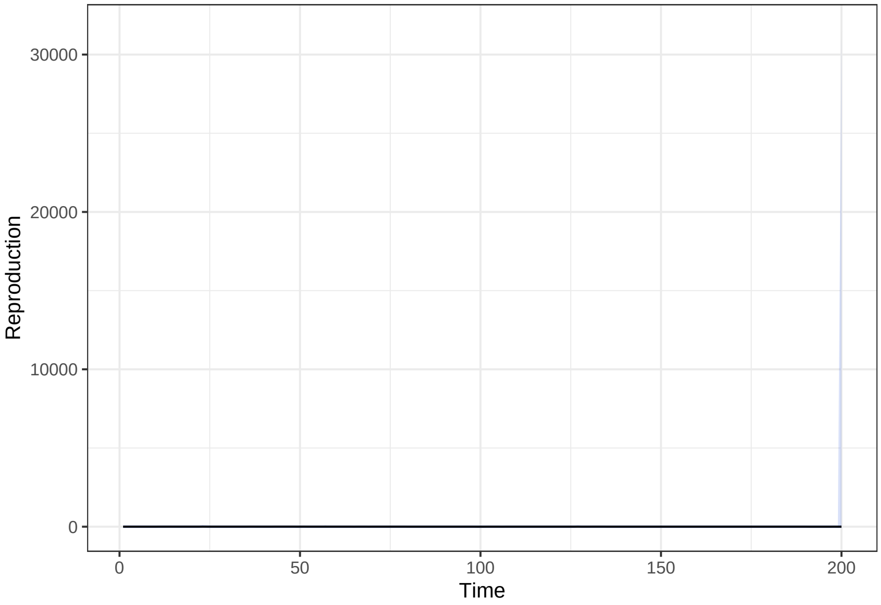


```
output = pl_pois_solow_eye_exp2(Y,X,rho,
                                QPrior=c(1e-2,1e-2),
                                Q_init=Q,
                                N=5000,
                                resample=TRUE)
```

```
tmp = data.frame(time=1:n, true=E, y=Y,
                 mt = apply(output$theta,2,quantile,0.5),
                 mt_hi = apply(output$theta,2,quantile,0.025),
                 mt_lo = apply(output$theta,2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  geom_point(aes(y=y),alpha=0.5,size=0.5) +
  xlab("Time") + ylab("Intensity")
```



```
tmp = data.frame(time=1:n, true=exp(beta),
                 mt = apply(exp(output$beta),2,quantile,0.5),
                 mt_hi = apply(exp(output$beta),2,quantile,0.025),
                 mt_lo = apply(exp(output$beta),2,quantile,0.975))

ggplot(tmp,aes(x=time)) +
  geom_ribbon(aes(ymin=mt_lo,ymax=mt_hi),
              fill="royalblue",alpha=0.2) +
  geom_line(aes(y=mt),color="royalblue") +
  geom_line(aes(y=true)) + theme_bw() +
  xlab("Time") + ylab("Reproduction")
```


30000
20000
10000
0
Reproduction
0
50
100
150
200
Time