Simple linear Gaussian model

Abolfazl Joukar

R Markdown

```
rm(list=ls())
suppressWarnings({library(MASS)})
suppressWarnings({library(ggplot2)})
suppressWarnings({library(gridExtra)})
suppressWarnings({library(FKF)})
suppressWarnings({library(pmhtutorial)})
#-----------------------------------------------------------------------
#                           Q1
#-----------------------------------------------------------------------
set.seed(123456)
simu_func<-function(T,phi,sigma2_V,sigma2_W){
X<-V<-Y<-W<-numeric(T)
X[1]<-rnorm(1,0,1)
for(t in 2:T){
V[t]<-rnorm(1,0,1)
X[t]<-(phi*X[t-1])+(sqrt(sigma2_V)*V[t])
W[t]<-rnorm(1,0,1)
Y[t]<-X[t]+(sqrt(sigma2_W)*W[t])
}
return(list(X=X,Y=Y))
}
observ_data<-simu_func(100,0.95,1,1)
lable<-c(rep("X",100),rep("Y",100))
observation<-c(observ_data$X,observ_data$Y)
result_data<-data.frame(observation,lable)
Dataset<-factor(lable)
plot_data<-ggplot(result_data,aes(x=rep(1:100,2),
           y=observation,group=factor(lable),colour=Dataset))+
           geom_line(size=0.8)+xlab("Time")+ylab("Observation")+
           scale_color_manual(name="",values=c("X"="blue","Y"="red"),labels=c("X","Y"))+
           theme(legend.position="bottom")
```

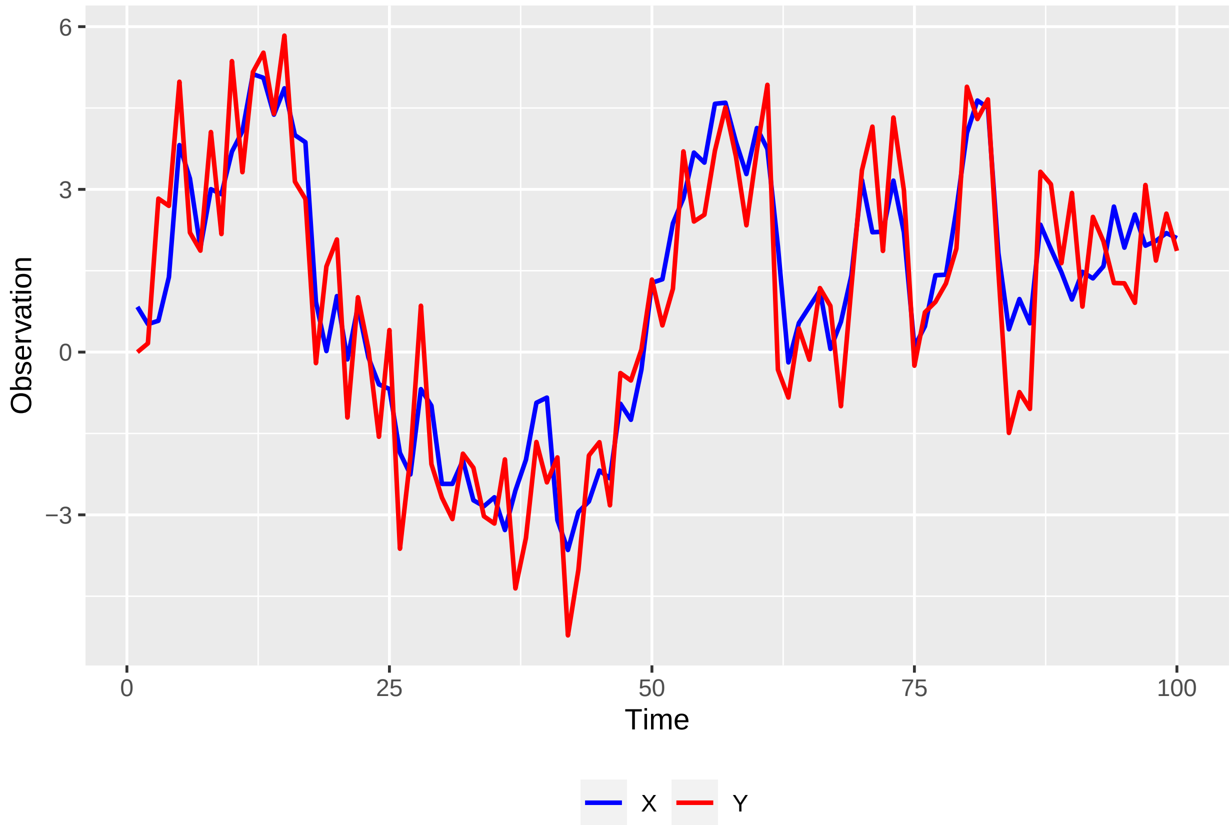
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
observ_data
```

```
## $X
##   [1]  0.83373317  0.51599874  0.57768623  1.38326204  3.81674435  3.19974155
```

```
##   [7]  1.92580457  3.00394674  2.91135538  3.69631603  4.07118811  5.12418290
##  [13]  5.05751359  4.37727485  4.86329021  3.99963410  3.86921072  0.92688180
##  [19]  0.01885294  1.03299867 -0.13455645  0.83999231 -0.09825759 -0.59529319
##  [25] -0.67926267 -1.85694779 -2.25579248 -0.68234204 -0.98779347 -2.42916899
##  [31] -2.42963304 -1.99507564 -2.73216227 -2.84148882 -2.67625761 -3.28035107
##  [37] -2.55183408 -1.98590347 -0.93377914 -0.83832988 -3.09877976 -3.64857621
##  [43] -2.94779585 -2.75335168 -2.18300634 -2.32774211 -0.95125385 -1.24832887
##  [49] -0.32182028  1.27786696  1.34268454  2.37410148  2.83725868  3.67844775
##  [55]  3.49346260  4.57799878  4.59814303  3.87535154  3.28234373  4.13141487
##  [61]  3.74375734  1.94807498 -0.18993669  0.53728239  0.83743563  1.13700683
##  [67]  0.05857812  0.55727368  1.41290783  3.17363645  2.20911398  2.21985534
##  [73]  3.16172882  2.20480956  0.09956173  0.47544839  1.41618623  1.42701567
##  [79]  2.62848196  4.03810942  4.63860344  4.50707331  1.82590709  0.41963012
##  [85]  0.97817041  0.53088265  2.34914177  1.90148403  1.47289070  0.96900246
##  [91]  1.47992096  1.35895032  1.58023409  2.68309095  1.92533006  2.53688732
##  [97]  1.96192621  2.04982657  2.19433724  2.10155338
## 
## $Y
##   [1]  0.00000000  0.16099690  2.82994196  2.69567755  4.98497609  2.20361180
##   [7]  1.87007303  4.05716535  2.17631248  5.36452700  3.31721334  5.16267545
##  [13]  5.52010854  4.39386086  5.83513953  3.14376710  2.82301245 -0.20297781
##  [19]  1.57892679  2.07699310 -1.20586095  1.01102486  0.06003166 -1.56121580
##  [25]  0.40668714 -3.62466985 -1.93432660  0.85490058 -2.06524221 -2.68191568
##  [31] -3.07861642 -1.87226507 -2.13174054 -3.02615347 -3.16090919 -1.97784355
##  [37] -4.35687843 -3.42699243 -1.65676153 -2.40186206 -1.94081619 -5.22239834
##  [43] -4.01272643 -1.90530425 -1.66020100 -2.82459404 -0.38627365 -0.52351906
##  [49]  0.04742635  1.33823737  0.49189527  1.16774645  3.70094857  2.40718750
##  [55]  2.53255245  3.71139189  4.51197481  3.60635262  2.33804622  3.72271080
##  [61]  4.92751319 -0.32436488 -0.83794304  0.43388237 -0.14016170  1.18082922
##  [67]  0.85200584 -0.99719668  1.20514003  3.35090083  4.15704645  1.86490621
##  [73]  4.32484278  2.98026485 -0.25100046  0.73366217  0.92490808  1.26524495
##  [79]  1.90821269  4.89177097  4.29598129  4.65906155  1.48387811 -1.49042189
##  [85] -0.73700733 -1.04786566  3.32490132  3.09488498  1.63925853  2.93496106
##  [91]  0.83841563  2.49309423  2.04263025  1.27336337  1.26824006  0.90896075
##  [97]  3.08094059  1.68776687  2.55290552  1.86627622
```

```
plot_data
```

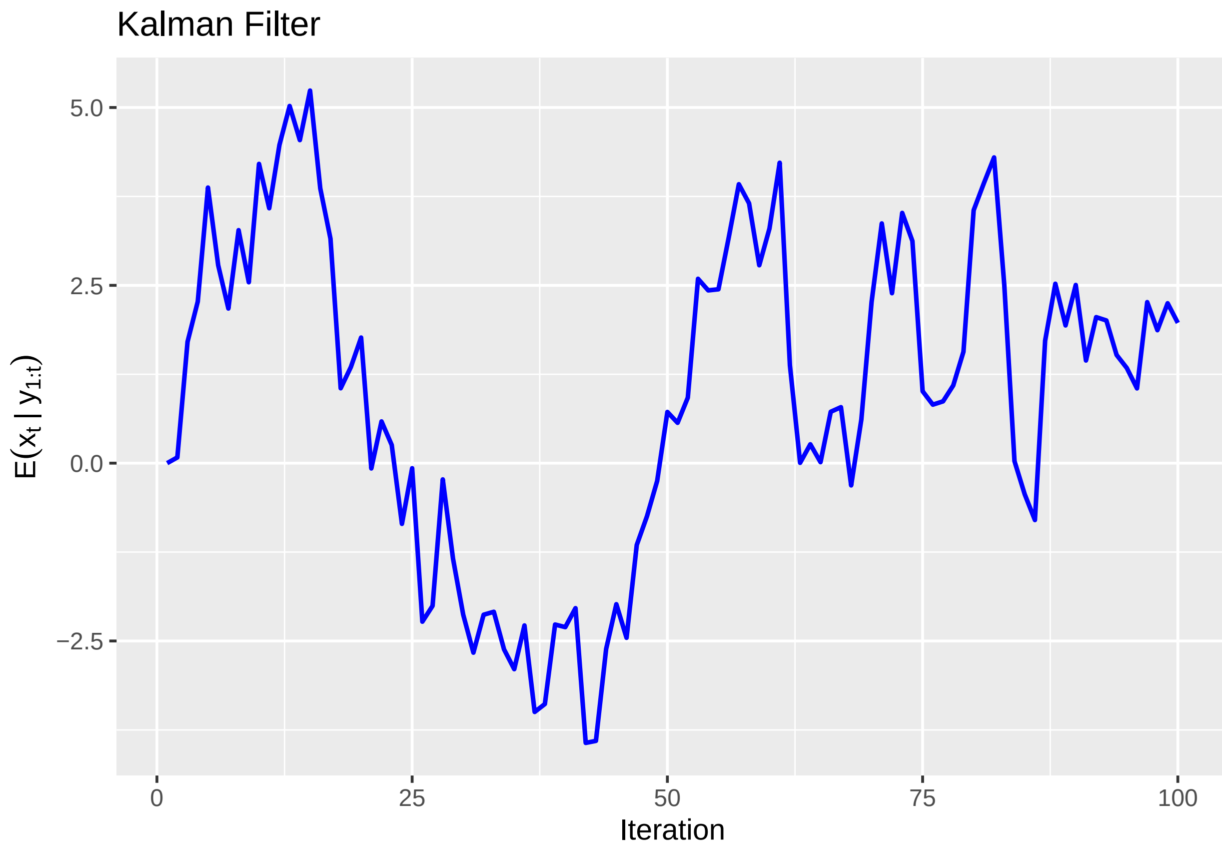



```
#-----------------------------------------------------------------------
#                              Q2
#-----------------------------------------------------------------------
Kalm_filt_func<-function(T,phi,sigma2_V,sigma2_W,y){
mu<-kalmanFilter(y,theta=c(phi,sqrt(sigma2_V),sqrt(sigma2_W)),initialState=0,
                 initialStateCovariance=1)$xHatFiltered
return(list(Kalman_Filter=c(mu)))
}
estimates<-Kalm_filt_func(100,0.95,1,1,observ_data$Y)
Kalman_Filter<-estimates$Kalman_Filter
plot_Kalman_Filter<-ggplot(data.frame(estimates$Kalman_Filter),
         aes(x=1:100,y=estimates$Kalman_Filter))+
         geom_line(size=0.8,colour="blue")+
         xlab("Iteration")+ylab(expression(E(x[t]~"|"~y[1:t])))+
         ggtitle("Kalman Filter")
Kalman_Filter
```

```
##   [1]  0.000000000  0.080498450  1.706650400  2.271753399  3.874850883
##   [6]  2.783459595  2.173887403  3.275492076  2.543375706  4.207574002
##  [11]  3.584045681  4.472882983  5.021405406  4.541593468  5.238429411
##  [16]  3.862954593  3.155303813  1.052939742  1.351864486  1.765920404
##  [21] -0.074349861  0.586570768  0.255142509 -0.853463028 -0.071064117
##  [26] -2.228801893 -2.006151609 -0.228445710 -1.339981004 -2.129034721
##  [31] -2.664217381 -2.130762397 -2.089549984 -2.617621940 -2.896358181
##  [36] -2.281450583 -3.497694604 -3.386110061 -2.268939424 -2.305183941
##  [41] -2.038569103 -3.933031192 -3.904284934 -2.613121872 -1.982866667
```

```
## [46] -2.455386113 -1.150039933 -0.746808275 -0.249587193  0.720054735
## [51]  0.567299666  0.920993856  2.591993624  2.428851109  2.444203527
## [56]  3.166176752  3.921746860  3.653169950  2.782437925  3.299144522
## [61]  4.223792554  1.377508581  0.004396867  0.265261308  0.013726136
## [66]  0.722575929  0.787038800 -0.312485349  0.615738414  2.265512150
## [71]  3.370337180  2.389525886  3.518519525  3.122446585  1.011512842
## [76]  0.822847368  0.868713627  1.092597100  1.566718885  3.556244393
## [81]  3.935925008  4.298069889  2.503867176  0.027853459 -0.437414113
## [86] -0.799735511  1.722040094  2.522378311  1.936313923  2.505089497
## [91]  1.443285411  2.052819751  2.006351276  1.521630113  1.337817867
## [96]  1.051000243  2.263748665  1.869372427  2.248001571  1.971962395
```

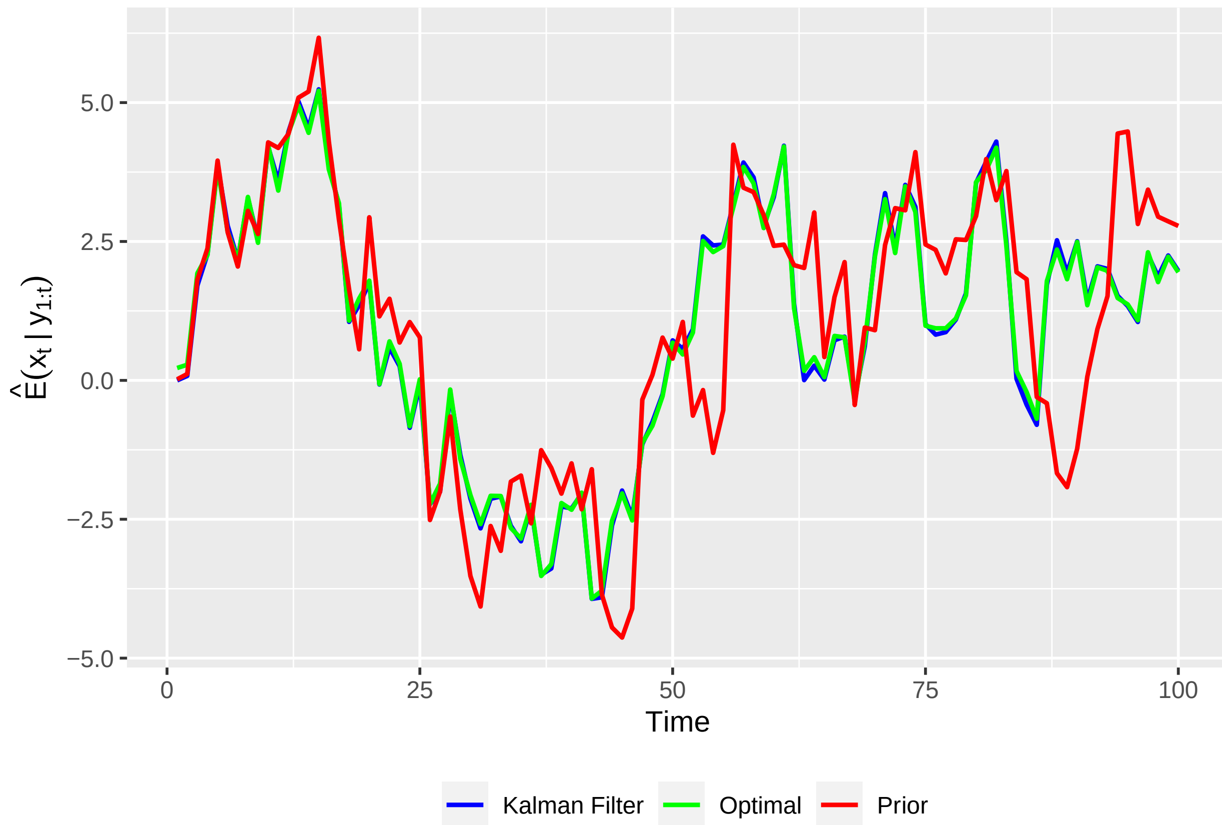
```
plot_Kalman_Filter
```




```
#-------------------------------------------------------------------
#                            Q3-Q4-Q5
#-------------------------------------------------------------------

#-------------------------------------------------------------------
#                 Sequential Importance Sampler (SIS)
#-------------------------------------------------------------------
set.seed(123456)
SIS_func<-function(T,phi,sigma2_V,sigma2_W,x,y,N){
g_Prior<-function(y,x)dnorm(y,mean=x,sd=sqrt(sigma2_W))
p<-function(y,x)dnorm(y,mean=phi*x,sd=sqrt(sigma2_V+sigma2_W))

X_hat_prior<-sample_var_prior<-numeric(T)
```

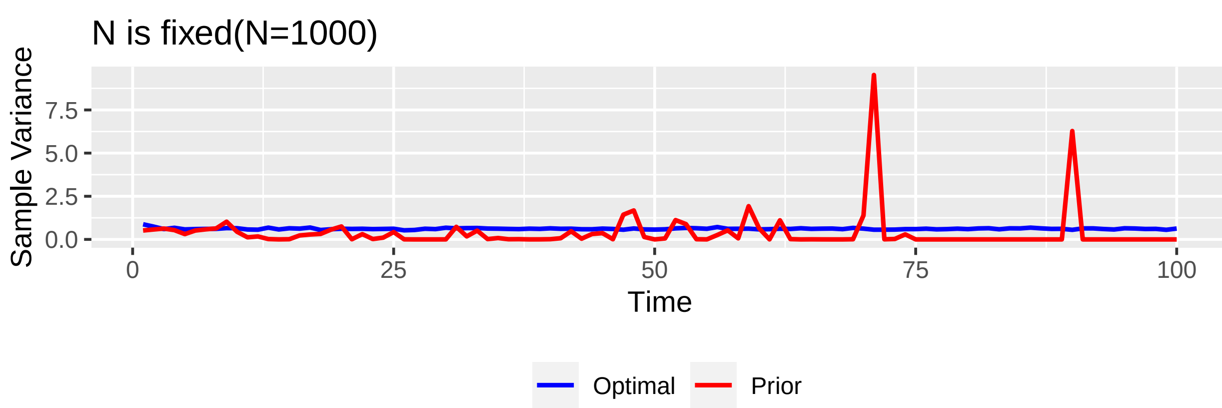
```r
X_mat_prior<-wei_prior<-matrix(NA,nrow=T,ncol=N)
X_mat_prior[1,]<-rnorm(N,0,1)
wei_prior[1,]<-g_Prior(y[1],X_mat_prior[1,])
X_hat_prior[1]<-sum(wei_prior[1,]*X_mat_prior[1,])/sum(wei_prior[1,])
sample_var_prior[1]<-sum(wei_prior[1,]*((X_mat_prior[1,]-X_hat_prior[1])^2))/(sum(wei_prior[1,]))
ESS_prior<-numeric(T)
ESS_prior[1]<-((sum(wei_prior[1,]))^2)/sum(wei_prior[1,]^2)
#-------------------------------------
X_hat_opt<-sample_var_opt<-numeric(T)
X_mat_opt<-wei_opt<-matrix(NA,nrow=T,ncol=N)
sigma_prop_opt<-(sigma2_V*sigma2_W)/(sigma2_V+sigma2_W)
mean_prop_opt<-sigma_prop_opt*(((phi*x)/sigma2_V)+(y[1]/sigma2_W))
X_mat_opt[1,]<-rnorm(N,mean_prop_opt,sqrt(sigma_prop_opt))
wei_opt[1,]<-p(y[1],x)
X_hat_opt[1]<-sum(wei_opt[1,]*X_mat_opt[1,])/sum(wei_opt[1,])
sample_var_opt[1]<-sum(wei_opt[1,]*((X_mat_opt[1,]-X_hat_opt[1])^2))/(sum(wei_opt[1,]))
ESS_opt<-numeric(T)
ESS_opt[1]<-((sum(wei_opt[1,]))^2)/sum(wei_opt[1,]^2)

for(t in 2:T){
X_mat_prior[t,]<-rnorm(N,phi*X_mat_prior[t-1,],sigma2_V)
wei_prior[t,]<-wei_prior[t-1,]*g_Prior(y[t],X_mat_prior[t,])
X_hat_prior[t]<-(sum(wei_prior[t,]*X_mat_prior[t,]))/sum(wei_prior[t,])
sample_var_prior[t]<-sum(wei_prior[t,]*((X_mat_prior[t,]-X_hat_prior[t])^2))/(sum(wei_prior[t,]))
ESS_prior[t]<-((sum(wei_prior[t,]))^2)/sum(wei_prior[t,]^2)

mean_prop_opt<-sigma_prop_opt*(((phi*X_mat_opt[t-1,])/sigma2_V)+(y[t]/sigma2_W))
X_mat_opt[t,]<-rnorm(N,mean_prop_opt,sqrt(sigma_prop_opt))
wei_opt[t,]<-p(y[t],X_mat_opt[t-1,])
X_hat_opt[t]<-(sum(wei_opt[t,]*X_mat_opt[t,]))/sum(wei_opt[t,])
sample_var_opt[t]<-sum(wei_opt[t,]*((X_mat_opt[t,]-X_hat_opt[t])^2))/(sum(wei_opt[t,]))
ESS_opt[t]<-((sum(wei_opt[t,]))^2)/sum(wei_opt[t,]^2)

}
return(list(X_mat_prior=X_mat_prior,X_mat_opt=X_mat_opt,
            wei_prior=wei_prior,wei_opt=wei_opt,
            Mu_hat_prior=X_hat_prior,s2_hat_prior=sample_var_prior,
            Mu_hat_opt=X_hat_opt,s2_hat_opt=sample_var_opt,ESS_prior=ESS_prior,ESS_opt=ESS_opt))
}
#-----------------------------------------------------------------
#                          Q3
#-----------------------------------------------------------------
set.seed(123456)
estimates_SIS<-SIS_func(100,0.95,1,1,observ_data$X,observ_data$Y,1000)

lable1<-c(rep("Kalman Filter",100),rep("Optimal",100),rep("Prior",100))
estimators<-c(estimates$Kalman_Filter,estimates_SIS$Mu_hat_opt,estimates_SIS$Mu_hat_prior)
result_estim_mu<-data.frame(estimators,lable1)
Estimators1<-factor(lable1)
plot_Mu_hat<-ggplot(result_estim_mu,aes(x=rep(1:100,3),
          y=estimators,group=factor(lable1),colour=Estimators1))+
          geom_line(size=0.8)+
          ylab(expression(hat(E)(x[t]~"|"~y[1:t])))+xlab("Time")+
```

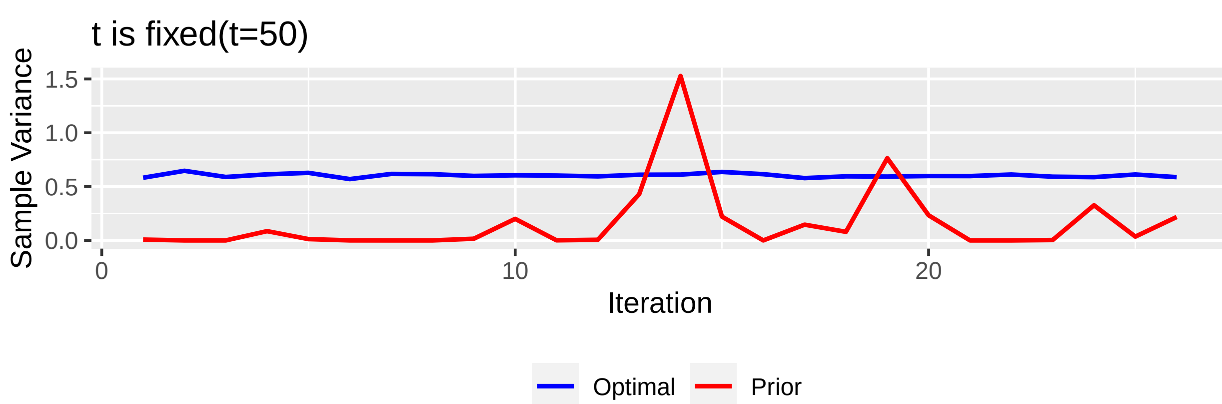
```r
          scale_colour_manual(name="",values=c("Kalman Filter"="blue","Optimal"="green","Prior"="red")
          labels=c("Kalman Filter","Optimal","Prior")) +
          theme(legend.position="bottom")
plot_Mu_hat
```




```r
#---------------------------------------------------------------------------
#                              Q4
#---------------------------------------------------------------------------
set.seed(123456)
N_seq<-seq(500,3000,100)
var_samp_prior<-var_samp_opt<-matrix(NA,100,length(N_seq))
for(i in 1:length(N_seq)){
var_initi<-SIS_func(100,0.95,1,1,observ_data$X,observ_data$Y,N_seq[i])
var_samp_prior[,i]<-var_initi$s2_hat_prior
var_samp_opt[,i]<-var_initi$s2_hat_opt
}
lable21<-c(rep("Optimal",100),rep("Prior",100))
lable22<-c(rep("Optimal",26),rep("Prior",26))
result_estimators_var1<-data.frame(c(var_samp_opt[,6],var_samp_prior[,6]),lable21)
result_estimators_var2<-data.frame(c(var_samp_opt[50,],var_samp_prior[50,]),lable22)
variance1<-factor(lable21)
variance2<-factor(lable22)

plot_s2_N_fix<-ggplot(result_estimators_var1,aes(x=rep(1:100,2),
          y=result_estimators_var1[,1],group=factor(lable21),colour=variance1))+
          geom_line(size=0.8)+
```

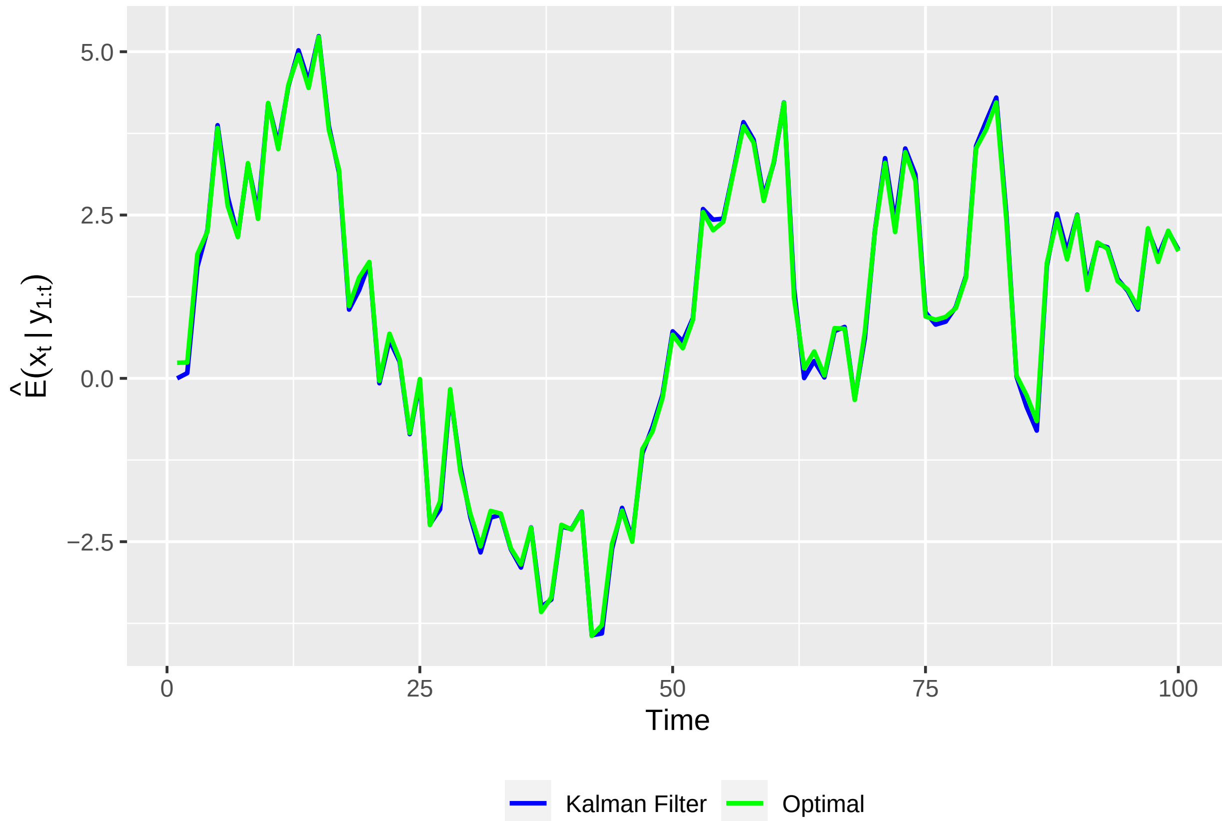
```
           ylab("Sample Variance")+xlab("Time")+
           scale_colour_manual(name="",values=c("Optimal"="blue","Prior"="red"),
           labels=c("Optimal","Prior"))+ggtitle("N is fixed(N=1000)")+
           theme(legend.position="bottom")

plot_s2_t_fix<-ggplot(result_estimators_var2,aes(x=rep(1:26,2),
           y=result_estimators_var2[,1],group=factor(lable22),colour=variance2))+
           geom_line(size=0.8)+
           ylab("Sample Variance")+xlab("Iteration")+
           scale_colour_manual(name="",values=c("Optimal"="blue","Prior"="red"),
           labels=c("Optimal","Prior"))+ggtitle("t is fixed(t=50)")+
           theme(legend.position="bottom")

grid.arrange(plot_s2_N_fix,plot_s2_t_fix,ncol=1)
```






```
#-------------------------------------------------------------------
#                             Q5
#-------------------------------------------------------------------
estimates_SIS2<-SIS_func(100,0.95,1,1,observ_data$X,observ_data$Y,1000)
lable3<-c(rep("Kalman Filter",100),rep("Optimal",100))
estimators3<-c(estimates$Kalman_Filter,estimates_SIS2$Mu_hat_opt)
result_estim_mu2<-data.frame(estimators3,lable3)
Estimators3<-factor(lable3)
plot_Mu_hat_opt<-ggplot(result_estim_mu2,aes(x=rep(1:100,2),
           y=estimators3,group=factor(lable3),colour=Estimators3))+
           geom_line(size=0.8)+
```

```
        ylab(expression(hat(E)(x[t]~"|"~y[1:t])))+xlab("Time")+
        scale_colour_manual(name="",values=c("Kalman Filter"="blue","Optimal"="green","Prior"="red")
        labels=c("Kalman Filter","Optimal","Prior")) +
        theme(legend.position="bottom")
plot_Mu_hat_opt
```




```
#------------------------------------------------------------------
p_hat_opt_func<-function(t,x_SIS,w,x,N){
delta<-wei_new<-numeric(N)
for(i in 1:N){
deltaa<-as.numeric(x_SIS[t,i]<=x[t])
#if(length(deltaa[deltaa==0])==0){
delta[i]<-sum(deltaa)
#}else{
#delta[i]<-0
#}
wei_new[i]<-w[t,i]
}
estim_p<-sum(delta*w[t,])/sum(w[t,])
return(list(estim_p=estim_p))
}


estimates_SIS2<-SIS_func(100,0.95,1,1,observ_data$X,observ_data$Y,1000)
p_hat_value<-p_exact<-numeric(100)
X_init<-rnorm(1,0,sqrt(1/2))
```

```r
for(t in 1:100){
p_hat_value[t]<-p_hat_opt_func(t,estimates_SIS2$X_mat_opt,
              estimates_SIS2$wei_opt,observ_data$X,1000)$estim_p
sigma<-1/2
b<-0.95*X_init
mean_init<-sigma*(observ_data$Y[t]+b)
p_exact[t]<-pnorm(observ_data$X[t],mean=mean_init,sd=sqrt(sigma))
X_init<-observ_data$X[t]
}
p_hat_value
```

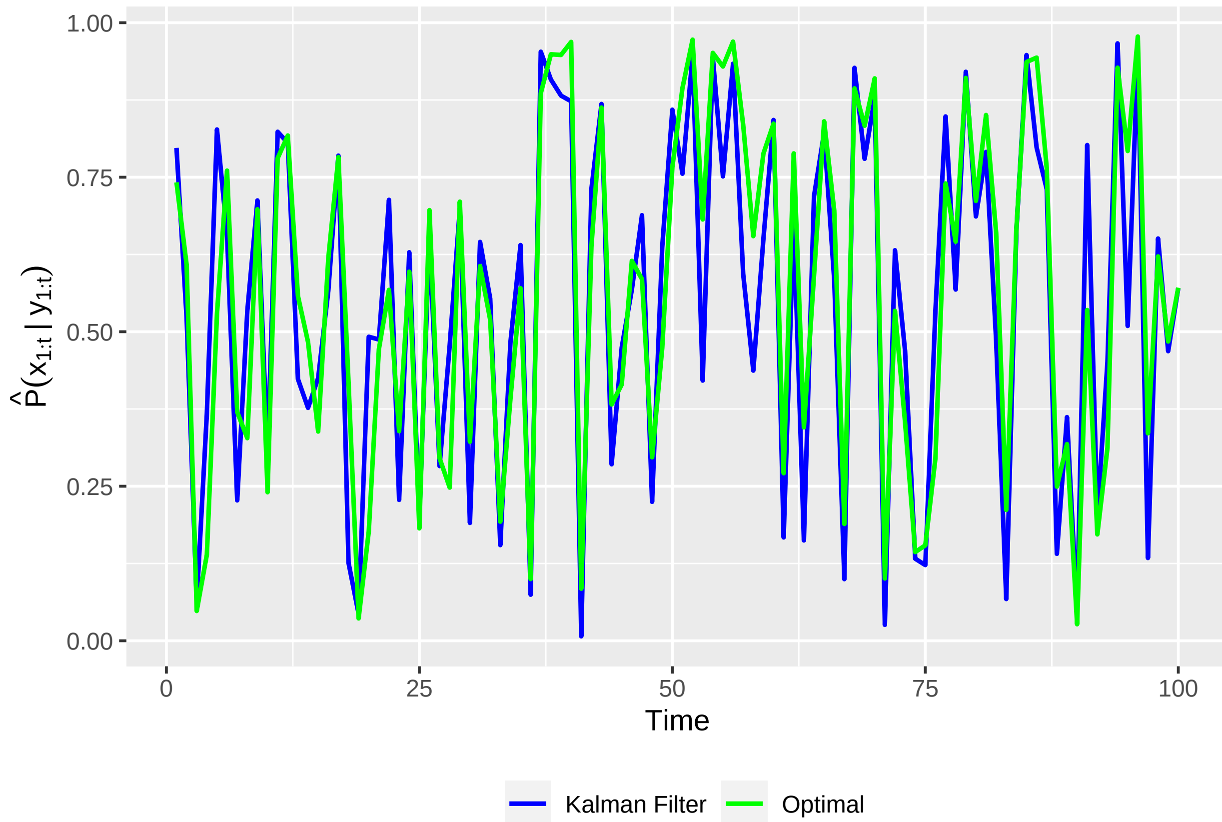
```
##   [1] 0.74175104 0.60826862 0.04816334 0.13922066 0.52912223 0.76073323
##   [7] 0.37020660 0.32783309 0.69815713 0.24030161 0.78086589 0.81752825
##  [13] 0.55794192 0.48329890 0.33850259 0.61833248 0.78256892 0.41814466
##  [19] 0.03619375 0.17679272 0.47172435 0.56774468 0.33918296 0.59710977
##  [25] 0.18190053 0.69678523 0.29531634 0.24782926 0.71047459 0.32207473
##  [31] 0.60630867 0.51922674 0.19266958 0.39156136 0.57040030 0.09981435
##  [37] 0.88586554 0.94894124 0.94782483 0.96890914 0.08422412 0.63735191
##  [43] 0.86275084 0.38236851 0.41473141 0.61471165 0.58512137 0.29689841
##  [49] 0.47339721 0.76613086 0.89430426 0.97261379 0.68171429 0.95121411
##  [55] 0.92911771 0.96945547 0.83488065 0.65466230 0.78821327 0.83675190
##  [61] 0.27105562 0.78858337 0.34541970 0.59045100 0.84044420 0.69589060
##  [67] 0.18890638 0.89377751 0.83328547 0.90998372 0.10078725 0.53371504
##  [73] 0.35329787 0.14344941 0.15443038 0.29540687 0.73993038 0.64571456
##  [79] 0.91069425 0.71165969 0.85066054 0.65992063 0.21168953 0.66637885
##  [85] 0.93646770 0.94355996 0.76451902 0.24935354 0.31822392 0.02673329
##  [91] 0.53510133 0.17214635 0.31213340 0.92728611 0.79240807 0.97768086
##  [97] 0.33576273 0.62195058 0.48420636 0.57126468
```

```r
p_exact
```

```
##   [1] 0.797485386 0.522260966 0.062919014 0.367694597 0.827306040 0.656535279
##   [7] 0.227147639 0.534151957 0.712426480 0.300967353 0.823529485 0.805463486
##  [13] 0.423448499 0.376791133 0.425134014 0.566092895 0.784932031 0.126143505
##  [19] 0.043407628 0.491846321 0.487420461 0.713423659 0.227932771 0.628574134
##  [25] 0.198134163 0.652915429 0.282649199 0.478407225 0.699081314 0.190675084
##  [31] 0.645309781 0.553511341 0.154743794 0.482721454 0.640230408 0.074539751
##  [37] 0.953083922 0.908068789 0.881986868 0.872869961 0.007206084 0.730569767
##  [43] 0.868547112 0.285565154 0.474592682 0.568203713 0.688472321 0.224760575
##  [49] 0.636796359 0.859278474 0.755723222 0.948429677 0.421020993 0.944536404
##  [55] 0.751340645 0.933603560 0.593683618 0.437105908 0.650034531 0.842655022
##  [61] 0.167249995 0.680636809 0.162381302 0.719252079 0.821866579 0.583341584
##  [67] 0.099675443 0.927010014 0.779836557 0.878925481 0.025754683 0.631823263
##  [73] 0.468931167 0.132813461 0.122455669 0.534555873 0.848353623 0.568327027
##  [79] 0.920630435 0.686537026 0.790930058 0.485450466 0.067500203 0.663041513
##  [85] 0.947662273 0.798041747 0.730559723 0.140662184 0.361866764 0.045097413
##  [91] 0.802100004 0.201809056 0.451272884 0.966564139 0.509444680 0.950694480
##  [97] 0.133903042 0.650819677 0.468560348 0.570771829
```

```r
lable3<-c(rep("Kalman Filter",100),rep("Optimal",100))
estimators3<-c(p_exact,p_hat_value)
result_estim_mu2<-data.frame(estimators3,lable3)
Estimators3<-factor(lable3)
plot_p_hat_opt<-ggplot(result_estim_mu2,aes(x=rep(1:100,2),
           y=estimators3,group=factor(lable3),colour=Estimators3))+
```

```r
          geom_line(size=0.8)+
          ylab(expression(hat(P)(x[1:t]~"|"~y[1:t])))+xlab("Time")+
          scale_colour_manual(name="",values=c("Kalman Filter"="blue","Optimal"="green","Prior"="red")
          labels=c("Kalman Filter","Optimal","Prior")) +
          theme(legend.position="bottom")
plot_p_hat_opt
```




```r
#-----------------------------------------------------------------------
#                            Q6
#-----------------------------------------------------------------------
#-----------------------------------------------------------------------
#               Metropolis Sampler(MH Sampler)
#-----------------------------------------------------------------------

MH_sampler<-function(T,phi,sigma2_V,sigma2_W,x,y,N){

density_func<-function(x,y){
sum_value<-log(dnorm(x[1],mean=0,sd=1))+log(dnorm(y[1],mean=x[1],sd=sqrt(sigma2_W)))
for(t in 2:T){
sum_value<-sum_value+log(dnorm(x[t],mean=phi*x[t-1],sd=sqrt(sigma2_V)))
          +log(dnorm(y[t],mean=x[t],sd=sqrt(sigma2_W)))
}
sum_value
}
X_mat_MH<-matrix(NA,nrow=T,ncol=N)
X_mat_MH_init<-estimates_SIS$Mu_hat_opt
```

```
k<-0
for(i in 1:N){
X_star<-rnorm(T,X_mat_MH_init,sqrt(0.01))
F1<-density_func(X_star,y)
F2<-density_func(X_mat_MH_init,y)
rho<-F1-F2
if(runif(1)<exp(rho)){
X_mat_MH[,i]<-X_star
k<-k+1
}else{
X_mat_MH[,i]<-X_mat_MH_init
}
X_mat_MH_init<-X_mat_MH[,i]
}
X_hat_MH<-apply(X_mat_MH,1,mean)
return(list(Mu_hat_MH=X_hat_MH,accept_rate=k/N))
}
estimates_MH<-MH_sampler(100,0.95,1,1,observ_data$X,observ_data$Y,1000)
estimates_MH$accept_rate
```

```
## [1] 0.515
```

```
lable3<-c(rep("Kalman Filter",100),rep("MH",100))
estimators_MH<-c(estimates$Kalman_Filter,estimates_MH$Mu_hat_MH)
result_estim_mu_MH<-data.frame(estimators_MH,lable3)
Estimators3<-factor(lable3)
plot_Mu_hat_MH<-ggplot(result_estim_mu_MH,aes(x=rep(1:100,2),
           y=estimators_MH,group=factor(lable3),colour=Estimators3))+
           geom_line(size=0.8)+
           ylab(expression(hat(E)(x[t]~"|"~y[1:t])))+xlab("Time")+
           scale_colour_manual(name="",values=c("Kalman Filter"="blue","MH"="red"),
           labels=c("Kalman Filter","MH"))+ggtitle("Metropolis approach")+
           theme(legend.position="bottom")

#----------------------------------------------------------------------
#                Gibbs Sampler(Gibbs Sampler)
#----------------------------------------------------------------------
Gibbs_sampler<-function(T,phi,sigma2_V,sigma2_W,x,y,N){
X_gibbs<-matrix(0,nrow=T,ncol=N)
mean_init<-0
sigma_init<-(1/(1+(1/(sigma2_W^2))))
X_init<-rnorm(N,mean_init,sqrt(sigma_init))
for(t in 1:T){
b<-phi*X_init
sigma<-1/(1/sigma2_W^2+1/sigma2_V^2)
mean_init<-sigma*(y[t]/sigma2_W^2+b/sigma2_V^2)
X_gibbs[t,]<-rnorm(N,mean=mean_init,sd=sqrt(sigma))
X_init<-X_gibbs[t,]
}
X_hat_Gibbs<-apply(X_gibbs,1,mean)
return(list(Mu_hat_Gibbs=X_hat_Gibbs))
}
estimates_Gibbs<-Gibbs_sampler(100,0.95,1,1,observ_data$X,observ_data$Y,1000)
```

```r
lable4<-c(rep("Gibbs",100),rep("Kalman Filter",100))
estimators_Gibbs<-c(estimates$Kalman_Filter,estimates_Gibbs$Mu_hat_Gibbs)
result_estim_mu_Gibbs<-data.frame(estimators_Gibbs,lable4)
Estimators4<-factor(lable4)
plot_Mu_hat_Gibbs<-ggplot(result_estim_mu_Gibbs,aes(x=rep(1:100,2),
           y=estimators_Gibbs,group=factor(lable4),colour=Estimators4))+
           geom_line(size=0.8)+
           ylab(expression(hat(E)(x[t]~"|"~y[1:t])))+xlab("Time")+
           scale_colour_manual(name="",values=c("Gibbs"="red","Kalman Filter"="blue"),
           labels=c("Gibbs","Kalman Filter"))+ggtitle("Gibbs approach")+
           theme(legend.position="bottom")
#----------------------------------------------------------------------

grid.arrange(plot_Mu_hat_MH,plot_Mu_hat_Gibbs,ncol=1)
```