LAB3_AML

Andreas C Charitos[andch552]

10/5/2019

Outline of lab

The robot moves along the horizontal axis according to the following SSM:

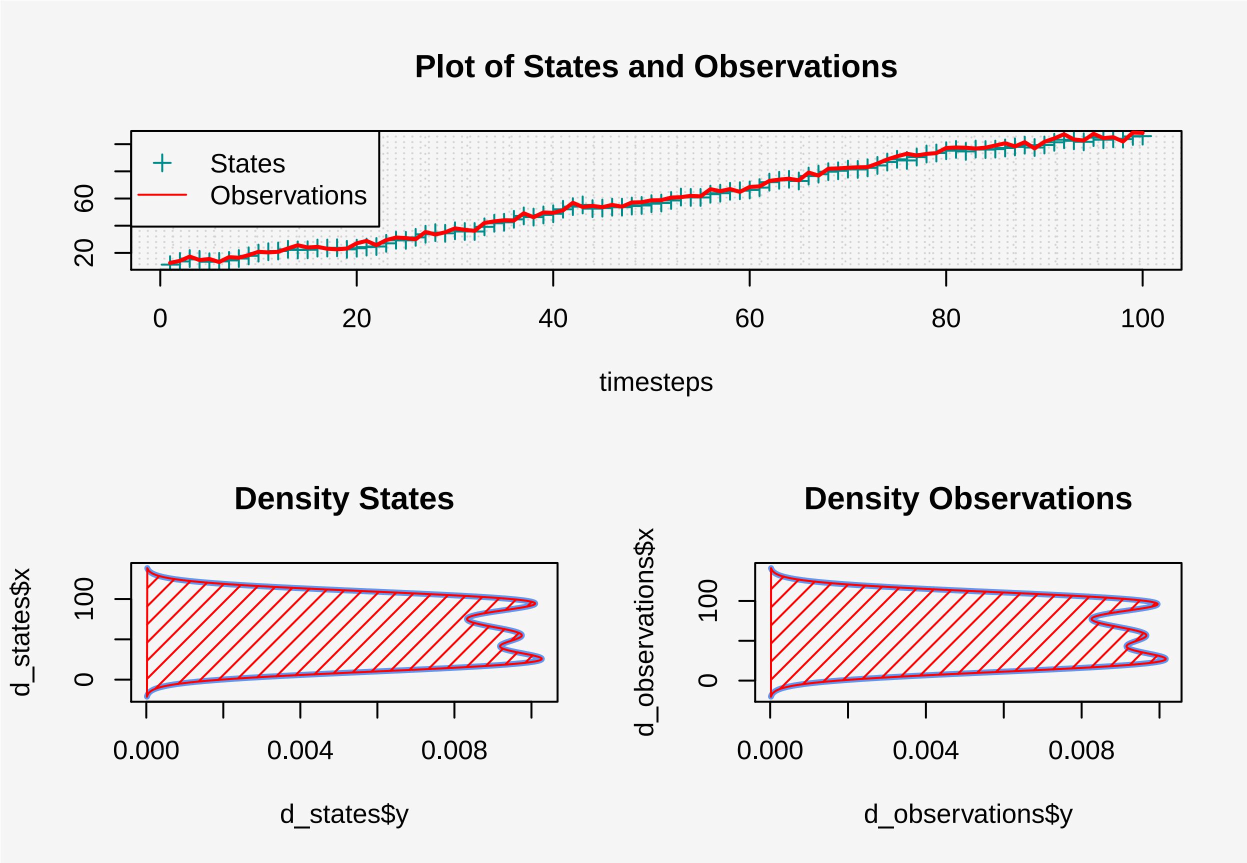
$$p(z_t|z_{t-1}) = (N(z_t|z_{t-1},1) + N(z_t|z_{t-1}+1,1) + N(z_t|z_{t-1}+2,1))/3 \quad //Tranition\ model$$
$$p(x_t|z_t) = (N(x_t|z_t,1) + N(x_t|z_t-1,1) + N(x_t|z_t+1,1))/3 \quad //Emission\ model$$

$$p(z_1) = Uniform(0,100)$$

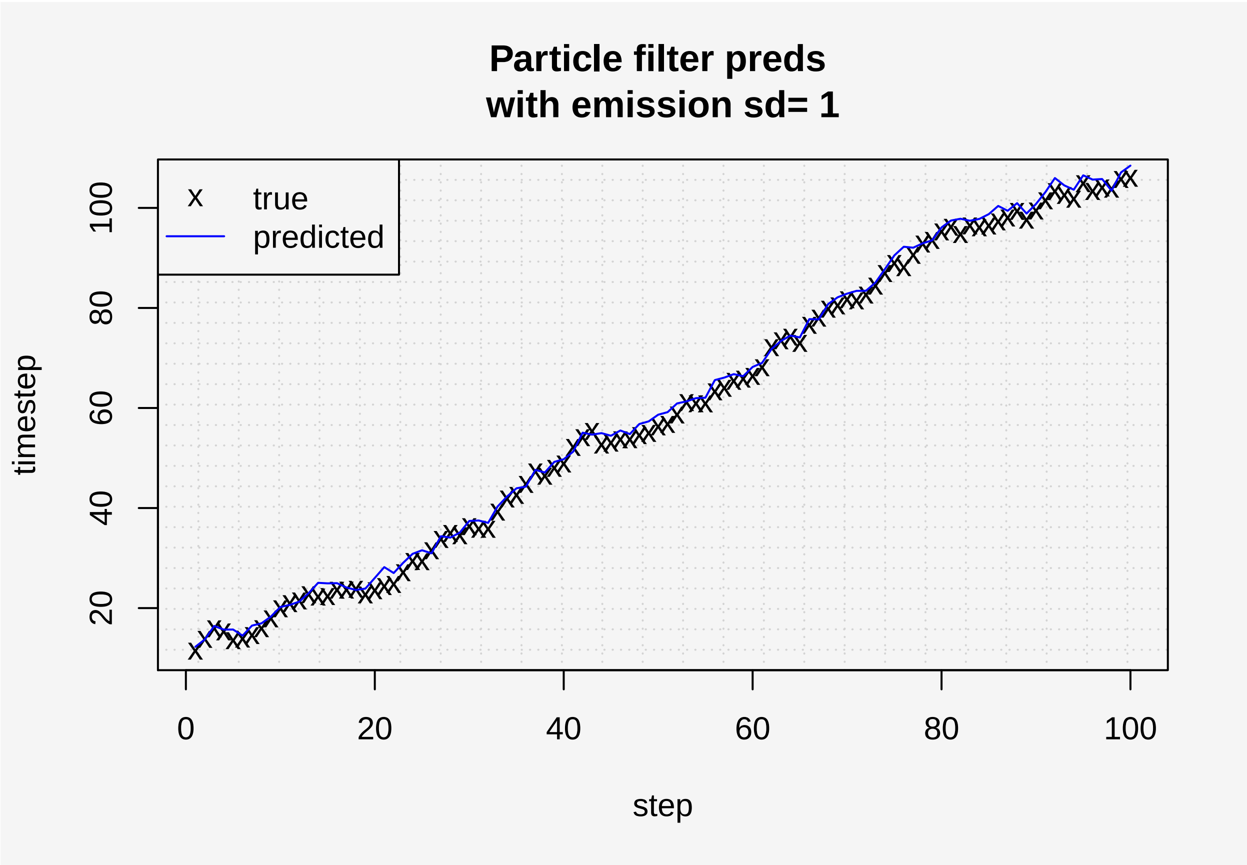
Question 1

Implement the SSM above. Simulate it for T = 100 time steps to obtain $z_{1:100}$ (i.e.,states) and $x_{1:100}$ (i.e., observations). Use the observations (i.e., sensor readings) to identify the state $(i.e., robot location) via particle filtering. Use 100 particles. Show the particles, the expected location and the true location for the first and last time steps, as well as for two intermediate time steps of your choice.

Plot of the states and observations

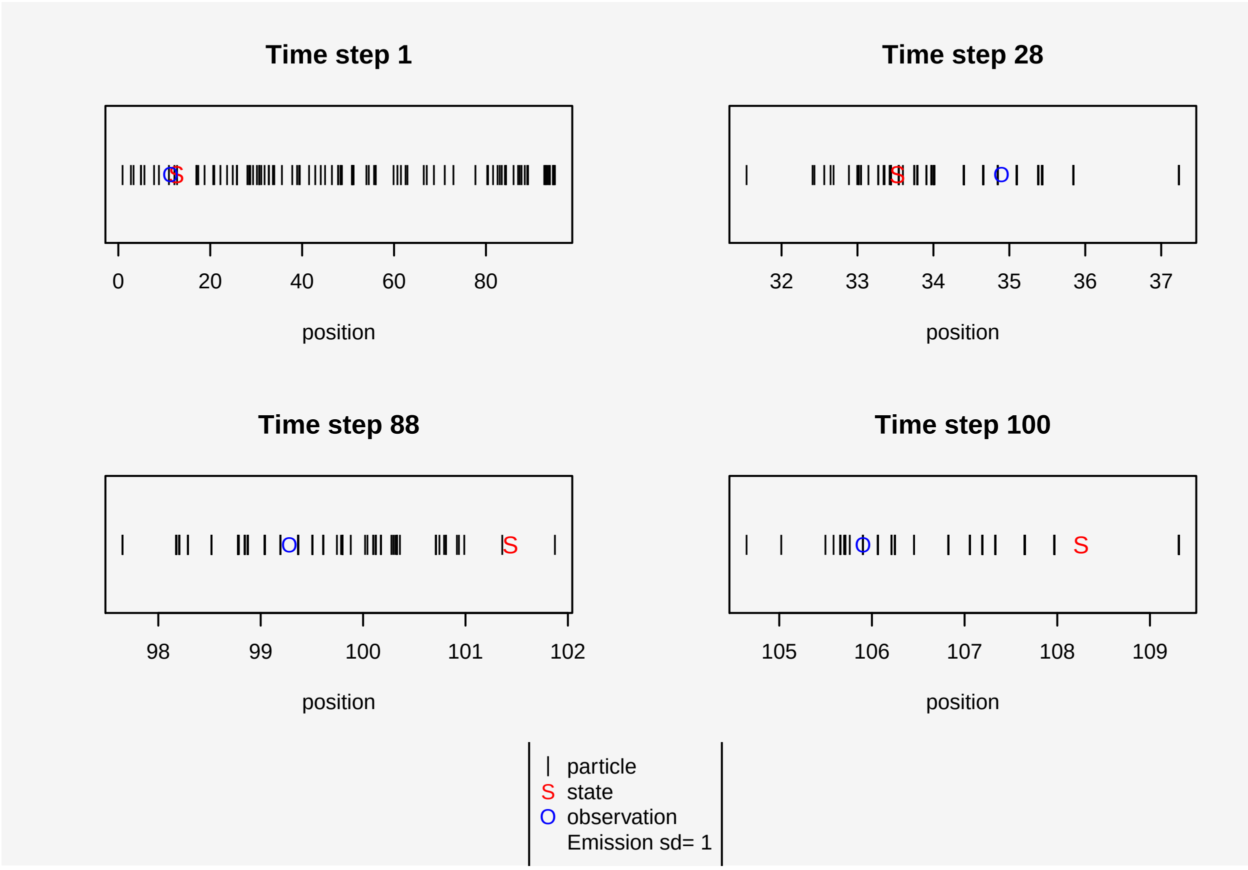


Plot of particle filter predictions with emission sd=1



As we can see from the plot the predictions generated with particle filter approximate the true states very well. This is reasonable since the sd of the emission model is small.

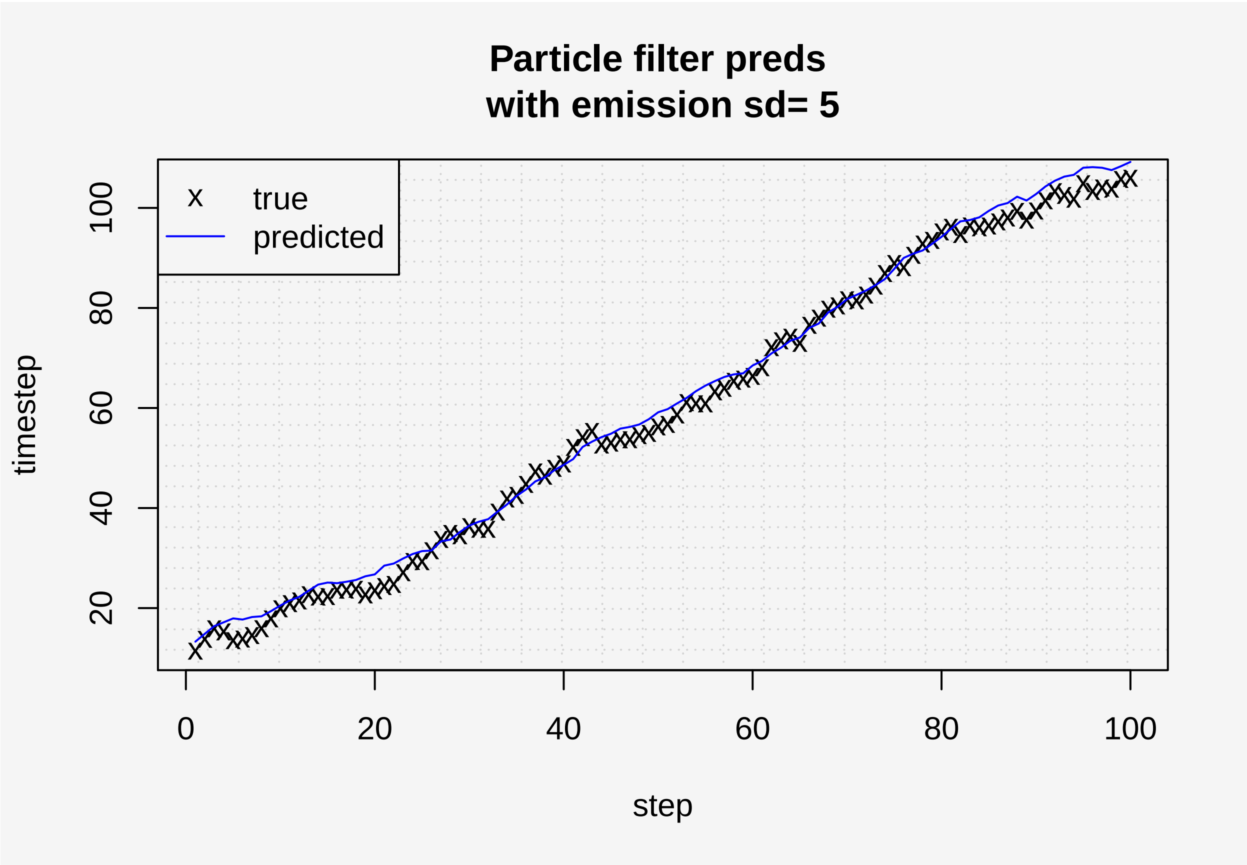
Plot of timesteps



Question 2

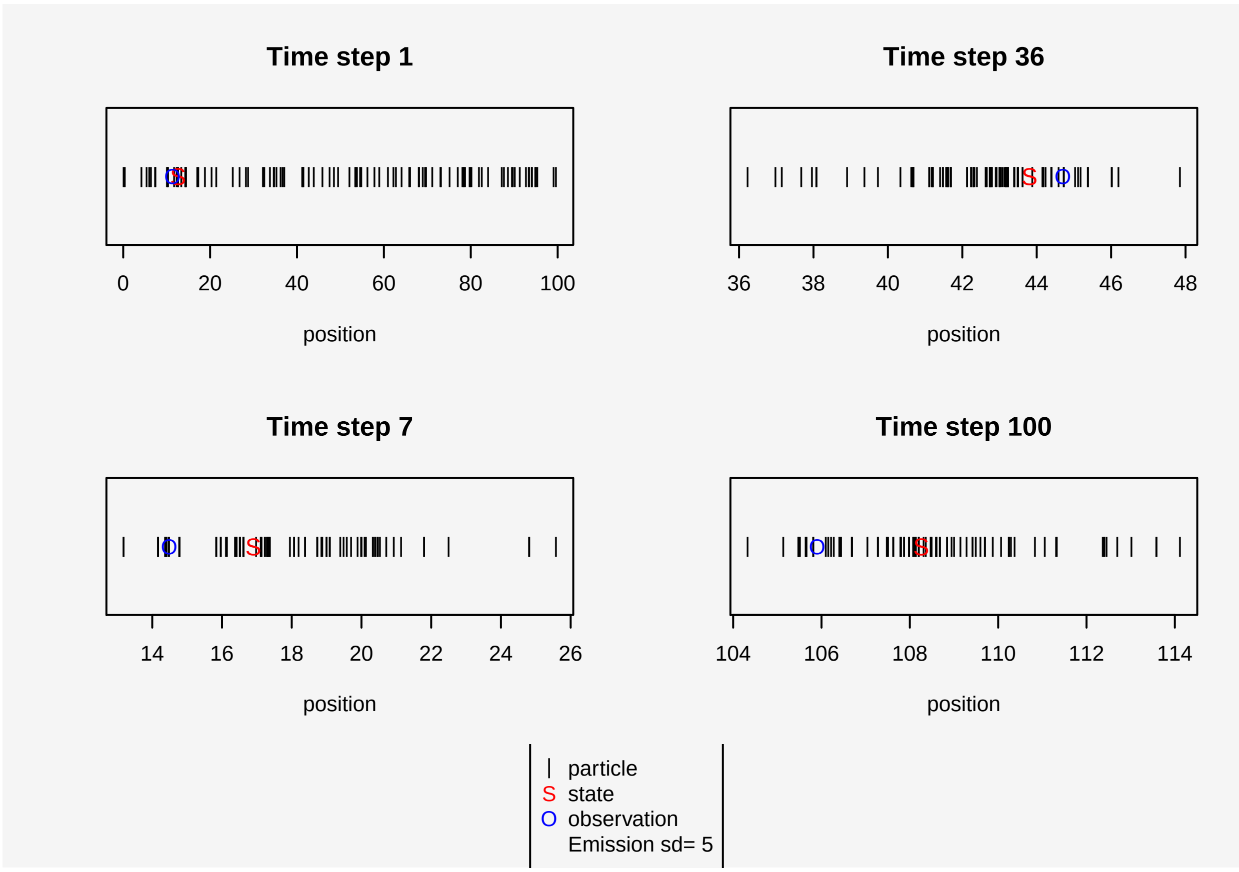
Repeat the exercise above replacing the standard deviation of the emission model with 5 and then with 50. Comment on how this affects the results.

Plot of particle filter predictions with emission sd=5

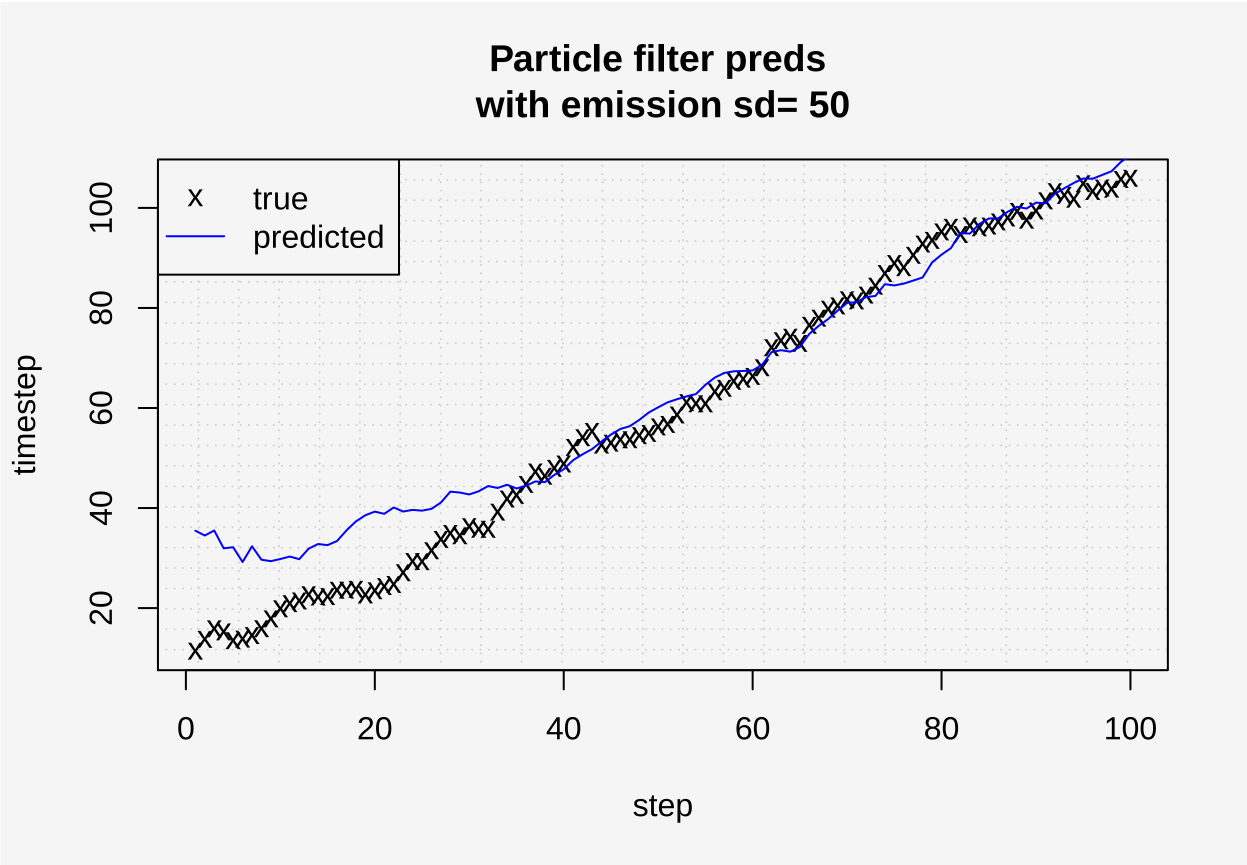


As we can see from the plot the predictions generated with particle filter have some small deviation from the true states. This is reasonable since the sd of the emission model is higher compares to the ones generated with sd=1.

Plot of the timesteps

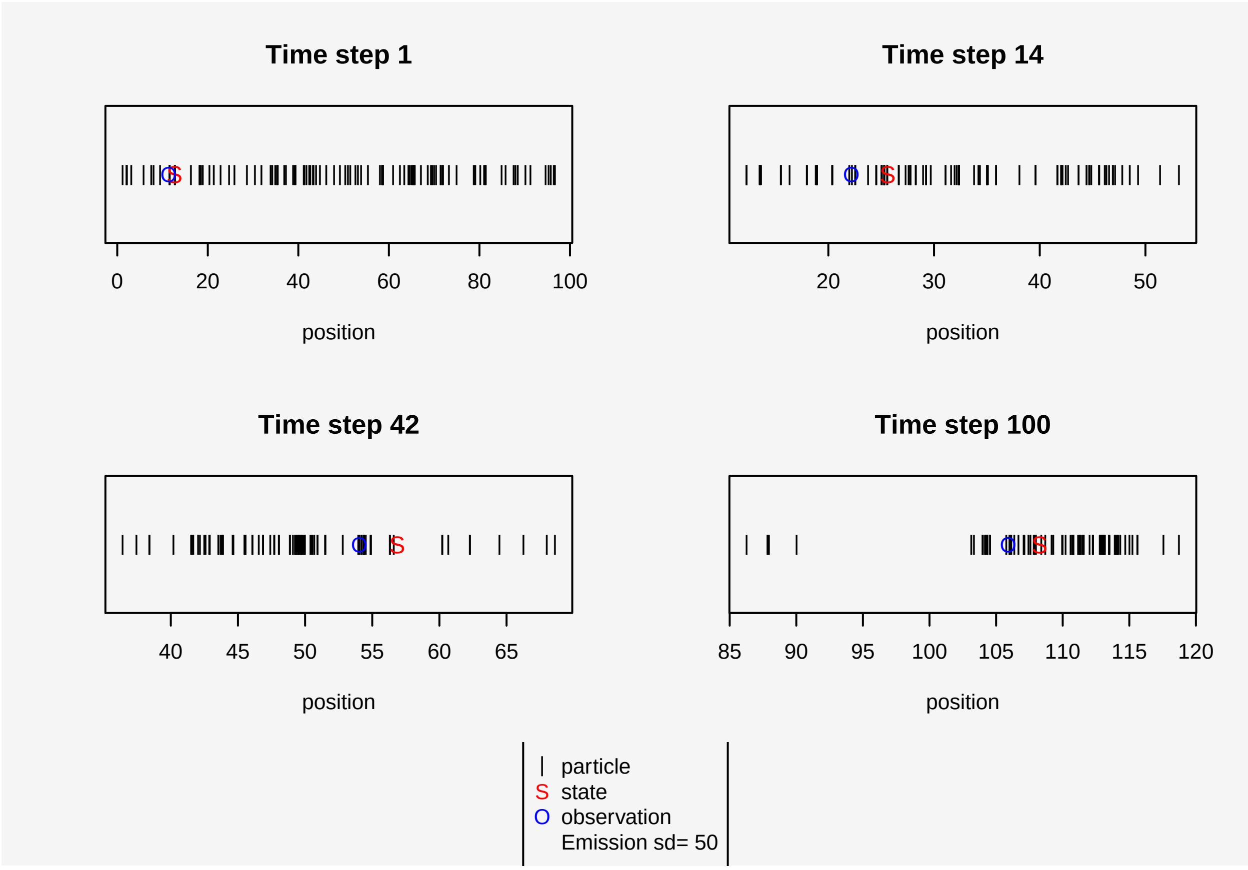


Plot of particle filter predictions with emission sd=50



As we can see from the plot the predictions generated with particle filter have some deviation from the true states . This is reasonable since the sd of the emission model is higher compares to the ones generated with sd=1 or 5.

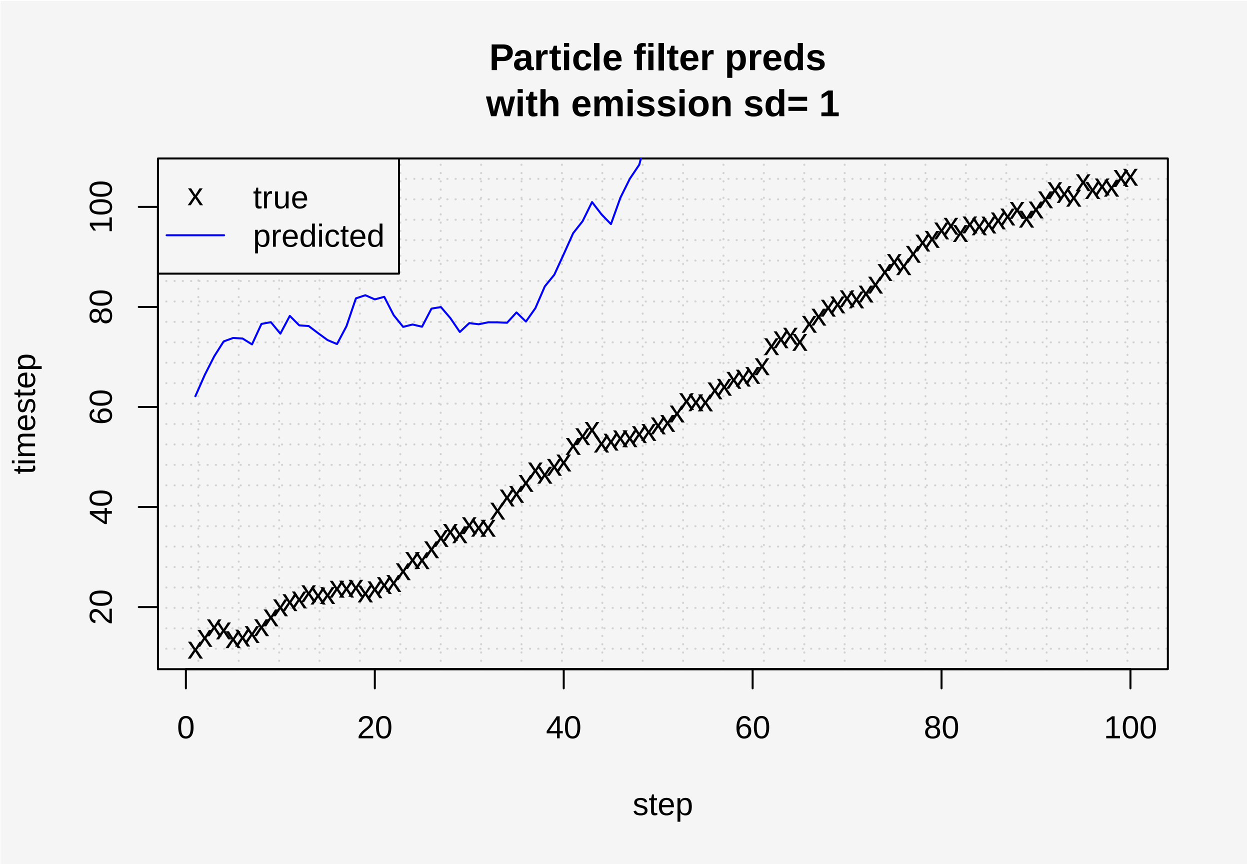
Plot of the timesteps



Question 3

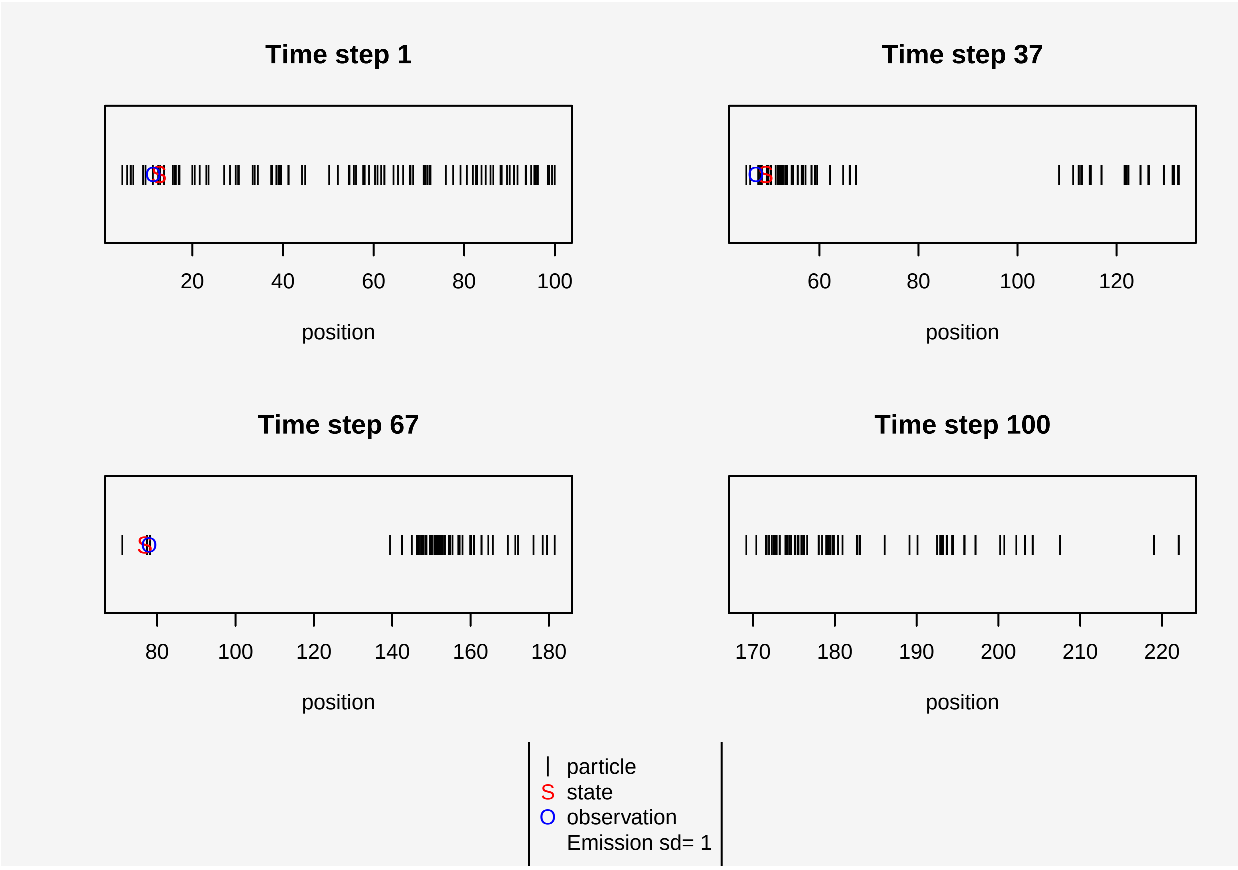
Finally, show and explain what happens when the weights in the particle filter are always equal to 1, i.e. there is no correction.

Plot of particle filter predictions with emission sd=1



The particle filter predictions with the weights fixed to 1 don't approximate the true states well and deviate a lot from the true states.This is reasonable since the observations don't have any influence for the predictions because the particles in each timestep are not updated according to the weigths in order to find particles that are good canditates for the observation step.

Plot of timesteps



Apprendix

```
## ----echo=FALSE-------------------------------------------------------
knitr::opts_chunk$set(echo = F)


## -----------------------------------------------------------------------
# Question1 -------------------------------------------------------------
set.seed(1234)

transition_func=function(zt,sd_trans){
  # function that samples a component and returns
  # random sample from transition model
  U=sample(1:3,1)
  #
  if(U==1){
    zt= rnorm(1,zt,sd_trans)
    }else if(U==2){
      zt = rnorm(1,zt+1,sd_trans)
    }else{
      zt= rnorm(1,zt+2,sd_trans)
    }
  return(zt)
}

emisssion_func=function(zt,sd_emiss){
  # function that samples a component and returns
  # random sample from emission model
  U=sample(1:3,1)
  #
  if(U==1){
      xt= rnorm(1,zt,sd_emiss)
    }else if(U==2){
      xt = rnorm(1,zt+1,sd_emiss)
    }else{
      xt= rnorm(1,zt+2,sd_emiss)
    }
  return(xt)
}


make_model=function(nTimes,upper_limit,sd_trans,sd_emiss){
  # function to simulate states and observations
  zt=double(nTimes)
  #
  xt=double(nTimes)
  #
  for(time in 1:nTimes){
    if(time==1){
      zt[time]=runif(1,0,upper_limit)
    }else{
    zt[time]=transition_func(zt[(time-1)],sd_trans)
    }
```

```
    xt[time]=emisssion_func(zt[time],sd_emiss)
  }
  return(list(states=zt,observations=xt))
}



## ---------------------------------------------------------------------------
x.sim=make_model(100,100,1,1)
par(bg="whitesmoke")
layout(matrix(c(1,1,2,3), 2, 2, byrow = TRUE))
plot(1:100,x.sim$states,col="darkcyan",pch=3,
     panel.first = grid(25,25),ylab="",xlab="timesteps",
     main="Plot of States and Observations")
lines(1:100,x.sim$observations,col="red",lwd=2)
legend("topleft",legend=c("States","Observations"),
       col = c("darkcyan","red"),lty=c(NA,1),pch=c(3,NA))
#
d_states=density(x.sim$states)
plot(y=d_states$x,x=d_states$y,col="cornflowerblue",lwd=3,
     main = "Density States",type="l")
polygon(y=d_states$x,x=d_states$y,col="red",density=15)
d_observations=density(x.sim$observations)
plot(y=d_observations$x,x=d_observations$y,col="cornflowerblue",lwd=3,
     main="Density Observations",type="l")
polygon(y=d_observations$x,x=d_observations$y,col="red",density=15)


## ---------------------------------------------------------------------------
# Partical Filter ------------

observations=x.sim$observations # observations of the simulation
states=x.sim$states # states from the simulation

emission_model=function(zt,mu,sd_emiss){
  # function that returns the probability-weight from the emission model
  result=( dnorm(zt,mu,sd_emiss)+dnorm(zt,(mu-1),sd_emiss)+dnorm(zt,(mu+1),sd_emiss) ) / 3
  return(result)
}

particle_func=function(obs,nTimes,nParts,sd_trans,sd_emiss,I){
  # function that returns the particles and weights given
  # observations, timesteps, #particles, sd of transition,sd of emission,
  # switch betwwen update and not update weights (I=1 means update)
  init_particles=runif(nTimes,0,100)
  particle_mat=matrix(0,nrow=nTimes+1,
                      ncol=length(init_particles))
  particle_mat[1,]=init_particles
  X_bar=double(nTimes)
  wt=matrix(0,nrow=nTimes,ncol=nParts)
  #
  for(i in 1:(nTimes)){
    #cat("---------iteration: ",i)
```

```
    #cat("\n")
    for( j in 1:length(init_particles)){
      X_bar[j]=transition_func(particle_mat[i,j],sd_trans)
      wt[i,j]=emission_model(obs[i], X_bar[j], sd_emiss)
      #X_bar[j]=future_particle
    }
    if(I==1){
    wt[i,]=wt[i,]/sum(wt[i,])
    }else{
      wt[i,]=rep(1,nTimes)
    }
    #
    particle_mat[(i+1),]=sample(X_bar,100,prob=wt[i,],replace = T)

  }
  return(list(particles=particle_mat,weigts=wt))

}

plot_st=function(true_states,predicted_states,sd_title){
  # plot function for the true and predicted staates for given emisison sd
  plot(true_states, pch="x",xlab="step",
       ylab="timestep",
       main=(paste("Particle filter preds \nwith emission sd=",sd_title)),lwd=2,
       panel.first = grid(25,25))
  lines(predicted_states, type="l", col="blue")
  legend("topleft",legend=c("true","predicted"),
         col=c("black","blue"),pch=c("x",NA), lty=c(NA,1))
}

plot_parts=function(particles,timestep,obs,states){
  # plot for the particles for given timestep
  plot(particles[timestep,],rep(0, length(particles[timestep,])),pch="|",yaxt='n',
     main = paste("Time step", timestep), xlab = "position", ylab = "")
  points(obs[timestep], 0,col="red",pch="S",cex=1.1)
  points(states[timestep], 0,col="blue",pch="O")

  }



## ------------------------------------------------------------------------
sd1=1 # set the emission sd
# sample from particle func
res=particle_func(obs=observations,
                  nTime=100,nParts=100,sd_trans =1,sd_emiss=sd1,I=1)
# calculate the row mean for each timestep and discard the first (initial)
pred_states<-rowMeans(res$particles)[-1]

# plot for states and filter particle preds
par(bg="whitesmoke")
plot_st(x.sim$states,pred_states,sd1)
# -------------------------------------------------------------------------
```

```r
## ---------------------------------------------------------------------------
# plot of particles for different timesteps
par(bg="whitesmoke")
layout(matrix(c(1,2,3,4,5,5), ncol=2, byrow=TRUE),heights=c(3, 3),
       respect=F)
plot_parts(res$particles,1,observations,states)
plot_parts(res$particles,28,observations,states)
plot_parts(res$particles,88,observations,states)
plot_parts(res$particles,100,observations,states)
par(mai=c(0,0,0,0))
plot.new()
legend("center",legend = c("particle","state","observation",paste("Emission sd=",sd1)),
       col = c("black","red","blue",NA),pch=c("|","S","O",NA))
# ----------------------------------------------------------------------------


## ---------------------------------------------------------------------------
# Question 2 -----------------------------------------------------------------

# For emission sd = 5 -------------
sd5=5 # emission sd =5
#
res_sd5=particle_func(obs=observations,
                      nTime=100,nParts=100,sd_trans =1,sd_emiss=sd5,I=1)
# filter particles predictions with emission sd=5
pred_states_sd5<-rowMeans(res_sd5$particles)[-1]
#
par(bg="whitesmoke")
plot_st(x.sim$states,pred_states_sd5,sd5)




## ---------------------------------------------------------------------------
#
par(bg="whitesmoke")
layout(matrix(c(1,2,3,4,5,5), ncol=2, byrow=TRUE),heights=c(3, 3),
       respect=F)
plot_parts(res_sd5$particles,1,observations,states)
plot_parts(res_sd5$particles,36,observations,states)
plot_parts(res_sd5$particles,7,observations,states)
plot_parts(res_sd5$particles,100,observations,states)
par(mai=c(0,0,0,0))
plot.new()
legend("center",
       legend = c("particle","state","observation",paste("Emission sd=",sd5)),
       col = c("black","red","blue",NA),pch=c("|","S","O",NA))



## ---------------------------------------------------------------------------
```

```r
# For emission sd = 50 --------------

sd50=50 # emission sd =50
res_sd50=particle_func(obs=observations,
                       nTime=100,nParts=100,sd_trans =1,sd_emiss=sd50,I=1)
# filter particle predictions eith emission sd=50
pred_states_sd50<-rowMeans(res_sd50$particles)[-1]
#
par(bg="whitesmoke")
plot_st(x.sim$states,pred_states_sd50,sd50)


## -----------------------------------------------------------------------
#
par(bg="whitesmoke")
layout(matrix(c(1,2,3,4,5,5), ncol=2, byrow=TRUE),heights=c(3, 3),
       respect=F)
plot_parts(res_sd50$particles,1,observations,states)
plot_parts(res_sd50$particles,14,observations,states)
plot_parts(res_sd50$particles,42,observations,states)
plot_parts(res_sd50$particles,100,observations,states)
par(mai=c(0,0,0,0))
plot.new()
legend("center",
       legend = c("particle","state","observation",paste("Emission sd=",sd50)),
       col = c("black","red","blue",NA),pch=c("|","S","O",NA))
# ------------------------------------------------------------------------


## -----------------------------------------------------------------------
# Question 3 -------------------------------------------------------------
# use non updated weights for particle filtering with emission sd=1
res_sd1_w=particle_func(obs=observations,
                        nTime=100,nParts=100,sd_trans =1,sd_emiss=sd1,I=2)
# make predictions for each time step(-->rows)
pred_states_sd1_w<-rowMeans(res_sd1_w$particles)[-1]
#
par(bg="whitesmoke")
plot_st(x.sim$states,pred_states_sd1_w,sd1)
#


## -----------------------------------------------------------------------
par(bg="whitesmoke")
layout(matrix(c(1,2,3,4,5,5), ncol=2, byrow=TRUE),heights=c(3, 3),
       respect=F)
plot_parts(res_sd1_w$particles,1,observations,states)
plot_parts(res_sd1_w$particles,sample(2:99,1),observations,states)
plot_parts(res_sd1_w$particles,sample(2:99,1),observations,states)
plot_parts(res_sd1_w$particles,100,observations,states)
par(mai=c(0,0,0,0))
plot.new()
legend("center",
```

```r
       legend = c("particle","state","observation",paste("Emission sd=",sd1)),
       col = c("black","red","blue",NA),pch=c("|","S","O",NA))

# ------------------------------------------------------------------------
# END


## ----code = readLines(knitr::purl("~/Courses/Advanced ML/Labs/Lab3/Lab3_AML.Rmd",documentation = 1)),
## NA
```