

Prepared by: A. Agarwal, A. Bouchard, J. Neeman and M. Traskin

1. (a) See Figure 1 for the plot of the particle's true location x_t .
- (b) See Figure 2 for the plot of the observations y_t on top of the particle's true location x_t .
- (c) Recall the state space model (instead of x_t in the text we are using s_t) given by Eq. 15.1 for the state at time t

$$s_{t+1} = As_t + Gw_t,$$

where $w_t \sim \mathcal{N}(0, Q)$ and Eq. 15.2 for the observed value at time t

$$y_t = Cs_t + v_t,$$

where $v_t \sim \mathcal{N}(0, R)$. In our case $s_t = (x_t^1, x_t^2, \dot{x}_t^1, \dot{x}_t^2)^T$ with

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ -0.02 & 0 & 0.98 & 0 \\ 0 & -0.02 & 0 & 0.98 \end{pmatrix},$$

$$G = I_4,$$

$$Q = \begin{pmatrix} 0_2 & 0_2 \\ 0_2 & 0.05I_2 \end{pmatrix},$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

and $R = 10I_2$. Since we are given that $s_1 \sim \mathcal{N}(0, 5I_4)$, we can run the Rauch-Tung-Striebel algorithm or the two-filter smoother to compute $\mathbf{P}(s_1|y_1, \dots, y_T)$. This estimated distribution (or its mean or mode) is the estimate of the particle's initial state.

- (d) Using the notation introduced above and the Rauch-Tung-Striebel algorithm we estimate $\mathbf{P}(s_1|y_1, \dots, y_T)$ as being normal with mean

$$\hat{s}_{1|T} = (-0.05340117, -0.10154811, 1.86017743, -2.17046699)^T$$

and covariance matrix

$$P_{1|T} = \begin{pmatrix} 3.6123712 & 0.0000000 & -0.2715015 & 0.0000000 \\ 0.0000000 & 3.6123712 & 0.0000000 & -0.2715015 \\ -0.2715015 & 0.0000000 & 0.3283129 & 0.0000000 \\ 0.0000000 & -0.2715015 & 0.0000000 & 0.3283129 \end{pmatrix}.$$

2. (a) We start by noting that the conditional distribution of y_T given x_T is rather straightforward. So it is useful to first derive the conditional distribution of x_T given x_1 .

$$\begin{aligned} x_T &= Ax_{T-1} + w_{T-1} \\ &= A(Ax_{T-2} + w_{T-2}) + w_{T-1} \\ &= A^2x_{T-2} + Aw_{T-2} + w_{T-1} \\ &\vdots \\ &= A^{T-1}x_1 + \sum_{t=2}^T A^{T-t}w_{t-1} \end{aligned}$$

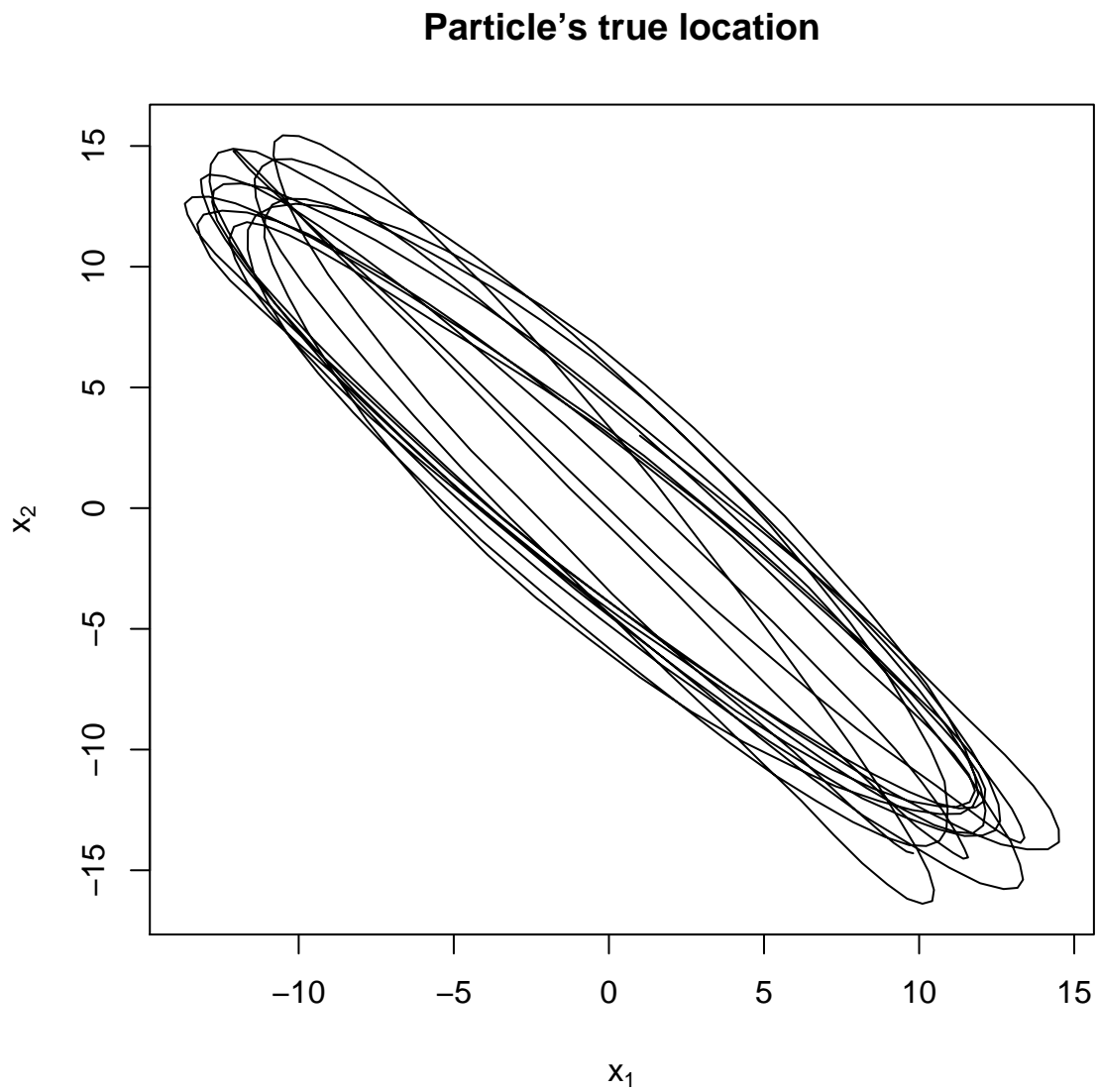


Figure 1: Plot of the particle's true location x_t .

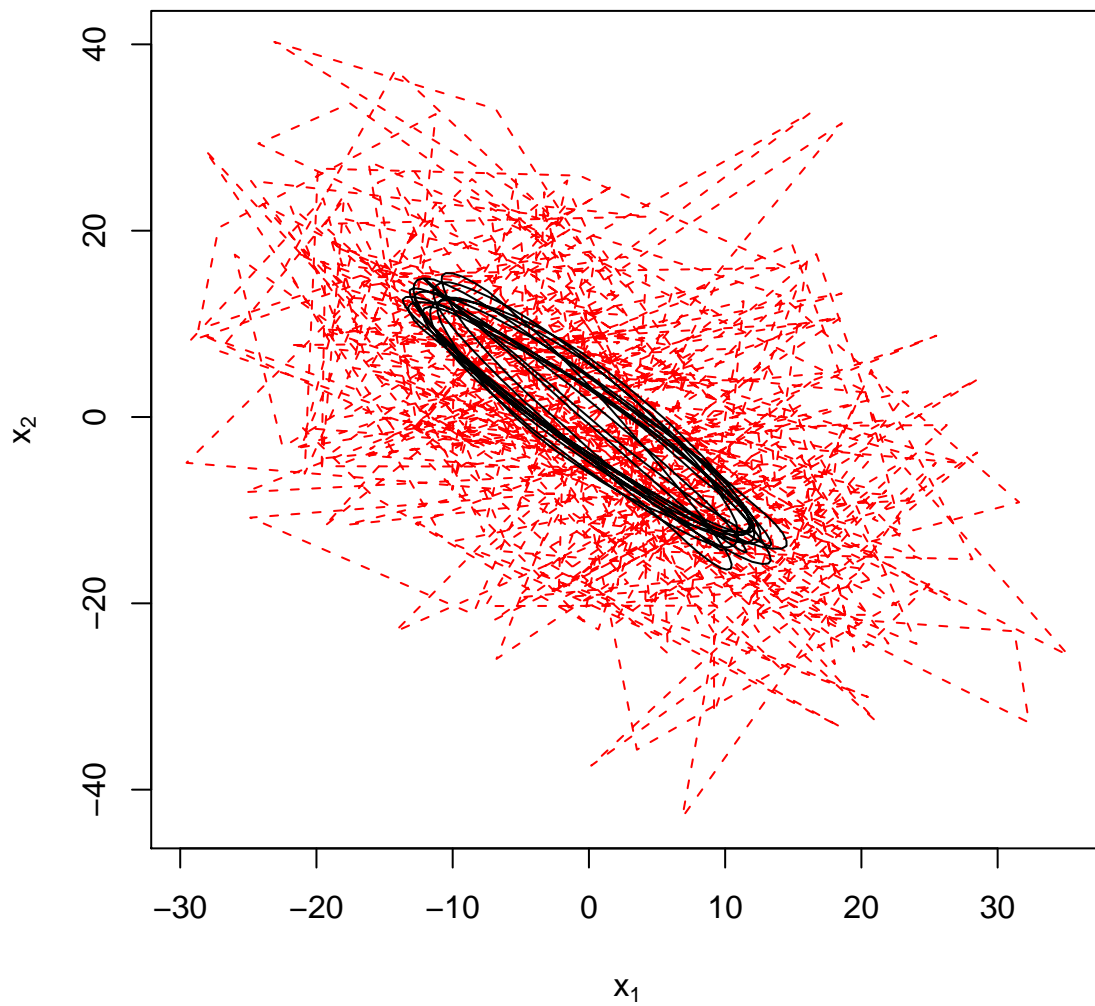
Particle's true location and observations

Figure 2: Plot of the observations y_t (red dashed lines) on top of the particle's true location x_t (black solid lines).

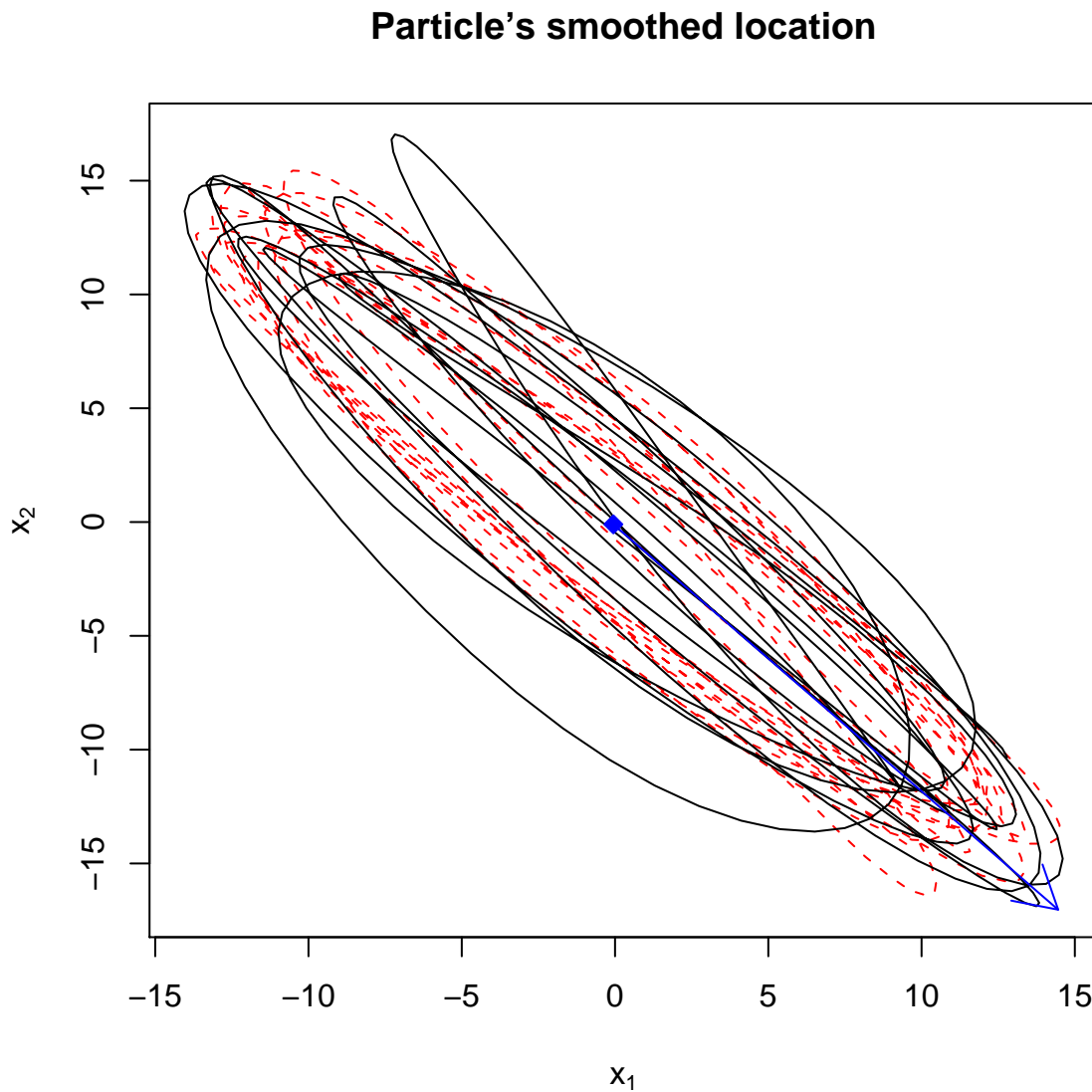


Figure 3: Plot of the particle's smoothed location \tilde{x}_t (black solid line). The true location (red dashed line) is given for comparison. The estimated initial position is given by a diamond with arrow pointing in the direction of the MAP initial velocity (the velocity vector was scaled by a factor of 7.8 to make the arrow visible). Assumed $v_t \sim \mathcal{N}(0, 100I_2)$.

We also recall that if a vector v is distributed according to $\mathcal{N}(0, \Sigma)$, then Av is distributed as $\mathcal{N}(0, A\Sigma A')$. Using the fact that the sum of independent normal variables is a normal variable with the sums of means and variances, we get:

$$x_T | x_1 \sim \mathcal{N}(A^{T-1}x_1, \sum_{t=2}^T A^{T-t}QA^{T-t'})$$

if $T \geq 2$, where $x_T | X_1$ denotes the conditional distribution of x_T conditioned on x_1 .

Now

$$\begin{aligned} y_T &= Cx_T + v_T \\ &= \begin{cases} Cx_1 + v_1 & \text{if } T = 1 \\ CA^{T-1}x_1 + \sum_{t=2}^T CA^{T-t}w_{t-1} + v_T & \text{if } T > 1 \end{cases} \end{aligned}$$

Hence the conditional distribution of y_T conditioned on x_1 is

$$\begin{aligned} &\mathcal{N}(Cx_1, \sigma^2) && \text{if } T = 1 \\ \mathcal{N}(CA^{T-1}x_1, \sum_{t=2}^T CA^{T-t}QA^{T-t'}C' + \sigma^2) &&& \text{if } T > 1 \end{aligned}$$

- (b) From part (a), using the observation matrix we can rewrite the variance of y_T conditioned on x_1 as $\text{tr}(O_{T-1}QO'_{T-1})$ for $T > 1$. Now we use a well known fact about the KL-Divergence between two normal distributions

$$\text{KL}(\mathcal{N}(\mu_1, \sigma^2), \mathcal{N}(\mu_2, \sigma^2)) = \frac{1}{2\sigma^2}(\mu_1 - \mu_2)^2.$$

Hence the KL-Divergence between the distributions of y_T conditioned on x_1 and \tilde{x}_1 for $T > 1$ is given by

$$\frac{1}{2(\text{tr}(O_{T-1}QO'_{T-1}) + \sigma^2)}(CA^{T-1}x_1 - CA^{T-1}\tilde{x}_1)^2.$$

By Cauchy-Schwartz inequality

$$(CA^{T-1}x_1 - CA^{T-1}\tilde{x}_1)^2 \leq \|C\|^2 \|A^{T-1}x_1 - A^{T-1}\tilde{x}_1\|^2.$$

Using the fact that $\|C\| = 1$ and $\|A^t v\| \leq \alpha^t \|v\|$, this further simplifies to

$$(CA^{T-1}x_1 - CA^{T-1}\tilde{x}_1)^2 \leq \alpha^{3(T-1)} \|x_1 - \tilde{x}_1\|^2.$$

Thus we get the desired bound

$$\text{KL}((y_T | x_1) || (y_T | \tilde{x}_1)) \leq \frac{\alpha^{2(T-1)} \|x_1 - \tilde{x}_1\|^2}{2(\text{tr}(O_{T-1}QO'_{T-1}) + \sigma^2)}$$

for $T > 1$. For $T = 1$, the KL-Divergence is simply bounded by $\frac{\|x_1 - \tilde{x}_1\|^2}{2\sigma^2}$.

A Code for problem 1

```
plotDataPS <- function( fileName, width, height, plotFunction, ... )
```

```
{
  postscript( paste( fileName, "eps", sep = "." ), width = width,
    height = height, paper = "special", horizontal = FALSE )
  plotFunction( ... )
  dev.off()
}

plotDataPDF <- function( fileName, width, height, plotFunction, ... )
{
  pdf( paste( fileName, "pdf", sep = "." ), width = width,
    height = height, paper = "special" )
  plotFunction( ... )
  dev.off()
}

prepareDataPlot <- function( mainTitle = "Particle's true location",
  data = NULL )
{
  if( is.null( data ) )
  {
    data <- read.table( "../data/hw5-2.data" )
  }
  else if( is.character( data ) )
  {
    data <- read.table( data )
  }
  else
  {
    data <- data.frame( data )
  }
  par( mar = c( 4, 4, 4, 1 ) + 0.1 )
  plot( data[[1]], data[[2]], type = "n", xlab = expression( x[1] ),
    ylab = expression( x[2] ), main = mainTitle )
}

plotTrueData <- function( colour = "black", linetype = "solid" )
{
  data <- read.table( "../data/hw5-2.true" )
  lines( data[[1]], data[[2]], col = colour, lty = linetype )
}

plotObservedData <- function()
{
  data <- read.table( "../data/hw5-2.data" )
  lines( data[[1]], data[[2]], lty = "dashed", col = "red" )
}

plotPartA <- function()
{
```

```

    prepareDataPlot( data = "../data/hw5-2.true" )
    plotTrueData()
}

plotPartB <- function()
{
  prepareDataPlot( "Particle's true location and observations" )
  plotObservedData()
  plotTrueData()
}

# function kalmanFilter
#
# Parameters:
# data - the matrix or dataframe of observations.
# A - the transition matrix for the state (see model description below).
# GQG - the covariance of the state noise (see model description below).
# C - the transition matrix for the observation (see model description below).
# R - the covariance of the observation noise (see model description below).
# P0 - the initial covaraince matrix (prior).
#
# Return value:
# A list of lists, where each list has components
# sii = \E( s_i|y_1, \ldots, y_i)
# Pii = \E( ( s_i - \E( s_i|y_1, \ldots, y_i) )
#       ( s_i - \E( s_i|y_1, \ldots, y_i) )^T|y_1, \ldots, y_i)
# siim1 = \E( s_i|y_1, \ldots, y_{i-1})
# Piim1 = \E( ( s_i - \E( s_i|y_1, \ldots, y_{i-1}) )
#            ( s_i - \E( s_i|y_1, \ldots, y_{i-1}) )^T|y_1, \ldots, y_{i-1})
#
# Description:
# This function implements the filtering step (the Kalman filter) for the state
# space model. The model is given by
# s_{t+1} = A s_t + w_t,
# y_t = C s_t + v_t,
# where w_t \sim \mathcal{N}(0, GQG^T) and v_t \sim \mathcal{N}(0,R). P0 is the covariance
# matrix of s_1.
kalmanFilter <- function( data, A, GQG, C, R, P0 )
{
  y <- as.matrix( data )
  T <- dim( y )[1]
  stateDim <- dim( A )[2]
  res <- list( sii = matrix( 0, nrow = T, ncol = stateDim ),
              Pii = array( 0, dim = c( T, stateDim, stateDim ) ),
              siim1 = matrix( 0, nrow = T, ncol = stateDim ),
              Piim1 = array( 0, dim = c( T, stateDim, stateDim ) ) )
  res$sii1[1,] <- rep( 0, stateDim )
  res$Piim1[1,,] <- P0
  for( i in 1:T )

```

```

{
  P <- res$Piim1[i,,]
  siim1 <- res$siim1[i,]
  res$sii[i,] <- as.vector( siim1 + P %>% t(C) %>%
    solve( C %>% P %>% t(C) + R, y[i,] - C %>% siim1 ) )
  res$Pii[i,,] <- P - P %>% t(C) %>%
    solve( C %>% P %>% t(C) + R, C %>% P )
  if( i < T )
  {
    res$siim1[i+1,] <- as.vector( A %>% res$sii[i,] )
    res$Piim1[i+1,,] <- A %>% res$Pii[i,,] %>% t(A) + GQG
  }
}
res
}

RTSsmoother <- function( KFOutput, A )
{
  T <- dim( KFOutput$sii )[1]
  stateDim <- dim( A )[2]
  res <- list( siT = matrix( 0, nrow = T, ncol = stateDim ),
    PiT = array( 0, dim = c( T, stateDim, stateDim ) ) )
  res$siT[T,] <- KFOutput$sii[T,]
  res$PiT[T,,] <- KFOutput$Pii[T,,]
  for( i in (T-1):1 )
  {
    L <- t( solve( KFOutput$Piim1[i+1,,] , A %>% KFOutput$Pii[i,,] ) )
    res$siT[i,] <- as.vector( KFOutput$sii[i,] +
      L %>% ( res$siT[i+1,] - KFOutput$siim1[i+1,] ) )
    res$PiT[i,,] <- KFOutput$Pii[i,,] +
      L %>% ( res$PiT[i+1,,] - KFOutput$Piim1[i+1,,] ) %>% t(L)
  }
  res
}

plotPartE <- function( smoothed )
{
  prepareDataPlot( "Particle's smoothed location", smoothed )
  plotTrueData( "red", "dashed" )
  lines( smoothed$siT[,1], smoothed$siT[,2] )
  x <- smoothed$siT[1,]
  points( x[1], x[2], col = "blue", bg = "blue", pch = 23 )
  arrows( x[1], x[2], x[1] + 7.8*x[3], x[2] + 7.8*x[4], col = "blue" )
}

main <- function()
{
  #Do part (a) - plot the particle's true location
  plotDataPS( "../graphics/hw5pla", 6, 6, plotPartA )
}

```

```
plotDataPDF( "../graphics/hw5p1a", 6, 6, plotPartA )
#Do part (b) - plot the observed location on top of the true location
plotDataPS( "../graphics/hw5p1b", 6, 6, plotPartB )
plotDataPDF( "../graphics/hw5p1b", 6, 6, plotPartB )
#Do part (d)
A <- rbind( c( 1, 0, 1, 0 ),
            c( 0, 1, 0, 1 ),
            c( -0.02, 0, 0.98, 0 ),
            c( 0, -0.02, 0, 0.98 ) )
GQG <- 0.05 * diag( c( 0, 0, 1, 1 ) )
C <- rbind( c( 1, 0, 0, 0 ),
            c( 0, 1, 0, 0 ) )
R <- 100 * diag( 1, 2 )
data <- read.table( "../data/hw5-2.data" )
PO <- diag( 5, 4 )
KFOutput <- kalmanFilter( data, A, GQG, C, R, PO )
smoothedState <- RTSsmoother( KFOutput, A )
print( "Filtered estimate of the initial state" )
print( KFOutput$sii[1,] )
print( "MAP estimate of the initial state" )
print( smoothedState$siT[1,] )
print( "Covariance matrix for the initial state" )
print( smoothedState$PiT[1,,] )
#Do part (e)
plotDataPS( "../graphics/hw5p1e", 6, 6, plotPartE, smoothedState )
plotDataPDF( "../graphics/hw5p1e", 6, 6, plotPartE, smoothedState )
}
```