

COVID-19 Time Series Analysis

Brandon Croom

11/20/2020

General Overview

The COVID-19 pandemic started in November 2019 and continued to spread across the world. The first cases in the United States were seen in late January/early February 2020, by late March 2020 states across the US began the process of issuing stay-at-home orders in order to limit the spread of the virus. The pandemic spread across the US in an extremely quick fashion forcing various stay-at-home restrictions to be put in place at the state level. As of this writing, many states had eased stay-at-home restrictions but were enacting other mandates, such as limiting gatherings and requiring face coverings where social distancing couldn't be practiced, as the the number of cases has continued to increase.

In North Carolina, Governor Roy Cooper and Dr. Mandy Cohen, Secretary of the North Carolina Department of Health and Human Services (NCDHHS) worked to ensure the best results for the public interest. Initially, stay-at-home orders were enforced to tamp down the virus spread. As various indicators began to trend in the appropriate direction, North Carolina moved into a phased re-opening plan. The initial re-opening steps still requested stay-at-home but did begin to allow businesses to open to limited capacity. After a few weeks the state moved into Phase Two, which allowed higher capacity in businesses, the beginning of school re-openings, and outdoor gatherings with a limited number of people and social distancing to occur. In the July 2020 timeframe, the state moved to Phase 3a. In this phase, small indoor gatherings and a few additional businesses, such as bars, were allowed to open. Following the national trend of increased virus cases, in November 2020 the Governor enacted a mandatory face covering requirement in all indoor spaces and outdoor spaces where social distancing could not be practiced.

Evaluating and communicating to the general public the spread and impact of the virus has taken many forms. There have been many different metrics utilized over the course of the pandemic. Prior to testing being widely available, metrics included looking at estimated cases based on hospital bed capacities in use or counts of symptomatic individuals. These metrics were good at the outset however some metrics like symptomatic counts underrepresented populations due to certain individuals being asymptomatic but contracting the virus. As testing has become more prevalent and reliable the most common metric currently used is percent positive.

Percent Positive Metric

The percent positive metric is defined as the percentage of all COVID-19 tests performed that were performed. This metric allows for understanding if enough testing is being performed and what the current level of transmission of COVID-19 is within a community (or how hard you have to look to find cases). In essence, the percent positive metric can be thought of as a relative risk gauge. If the percent positive is high in an area there is a higher chance for exposure to the virus than if the percent positive rate is lower. If the percent positive is low that does not indicate herd immunity or no risk, it simply means that the level of transmission is low in that community or that not enough testing has been performed in the community.

On the negative side, the percent positive metric does not inform individuals of actual numbers of COVID-19 cases. Not all who have COVID-19 get tested and there are asymptomatic carrier who do not know they

should get tested. These individuals would not be reflected in the percent positive metric and thus it cannot be leveraged as a total number of cases type calculation. Additionally, the percent positive metric can be influenced on a daily basis based on numbers of tests performed (or not performed) that day or any changes in testing numbers. For example, should a state report a high number of positive cases one day and then realize there was a reporting error or worse that the testing procedures invalidated the results, the percentage positive metric can move greatly over time. Additionally, there is no single version of percent positive that tells an individual what is good or bad. For example, one state may say that at a 3% positivity rate all school learning must take place remotely. In another state that requirement may be at 15%. Thus it becomes a relative metric for the location that an individual is in.

All in all the percent positive metric does provide a good indicator of relative risk for exposure to COVID-19. Even with the concerns outlined, the metric is currently being leveraged by policy makers to enact policy for reopening schools and businesses and by public health officials to understand the current state of the pandemic. As this metric is continued to be leveraged, education about what this metric represents should be discussed. This continued reinforcement of the correct definition will help the general public understand the metric and hopefully reduce confusion around it.

Over the course of this document, a time-series based analysis will be performed to assist Governor Cooper and Dr. Cohen in understanding how COVID-19 can be forecast. The reader will be able to review all code and commentary around the code will be provided. The analysis will cover the entire US and North Carolina specifically. Multiple time series models will be analyzed in order to evaluate the modeling techniques and also provide guidance on how to approach the pandemic.

Document Organization

This document is organized in the following sections:

- Data Description - provides an overview of the data
- Data Preparation and Exploratory Data Analysis - provides information on the data cleanliness and structure
- Time Series Analysis - provides information on the time series analysis of the data

The Time Series Analysis section is broken into two sub parts:

- United States Analysis
- North Carolina Analysis

Each of the time series subsections may be read independently. The sections do not build on each other. Thus a reader consuming the entire document may find repeat verbiage. This is done to allow the independence to occur.

Data Description

In order to perform the analysis, COVID-19 data will be needed. There have been many R packages developed to make obtaining COVID-19 data easy to obtain. For the purposes of this analysis two different packages will be used.

The first package is COVID19US (<https://cran.r-project.org/web/packages/covid19us/index.html>). This package is a wrapper around the COVID Tracking Project API (<https://covidtracking.com/data/api>). The COVID Tracking Project (<https://covidtracking.com/>) compiles data from each state in the United States on a daily basis to provide the most up to date information. This site has a high focus on testing information and the testing outcomes than actual case numbers. The data is sourced from state public health authorities or official statements. There is human interaction with the data that ensures accuracy and all data from each

state is provided a grade to allow for understanding reliability. For North Carolina, the site grades the data as an A+ (<https://covidtracking.com/data/#state-nc>). The data from this site will allow for calculating the percent positive rate that will be used for analysis.

The second package is COVDATA (<https://kjhealy.github.io/covdata/>). This package provides data related to hospitalizations, mobility information and other case information for the US and European countries. This data set works to maintain the consistency with the source data sets it sources data from, thus data is provided as is from those sites. Data cleanliness will need to be checked to ensure that does not impact analysis. As of this writing the data for this package was up to date through November 25, 2010.

Data Preparation and Exploratory Data Analysis

The necessary R-packages needed for analysis are first loaded. These packages will allow for dataset loading as well and functionality for the timeseries analysis.

```
# Load Necessary Libraries
library(tswge)
library(covid19us) #The Covid Project Data Set
library(covdata)
library(tidyverse)
library(nnfor)
library(forecast)
library(vars)
library(RColorBrewer)
```

In this section, constants and variables are defined. Defining values in this way allows the reader, should they so chose, to change items once and assess the impact. For example, should the reader want to change the SHORT_TERM_FORECAST_HORIZON from 7 days to 5 days, thay can simply update that value and re-execute all the cells in this file.

```
# Define Constants

SHORT_TERM_FORECAST_HORIZON = 7
LONG_TERM_FORECAST_HORIZON = 90
SAMPLE_SIZE_WINDOW = 100
NN_REPS = 50
DATE_FORMAT = "%m%d%Y"
HOSPITAL_DATE_FORMAT = "%Y-%m-%d"
STATE_VAL = "NC"
BACK_OFF_WINDOW = 21

#Define Dataframe Headers
COL_DATE_VAL = "Date_Val"
COL_POSITIVE_CASES = "Positive_Cases"
COL_TOTAL_TESTS = "Total_Tests"
COL_PERCENT_POSITIVE = "Percent_Positive"
COL_HOSPITAL_COUNT = "Hospitalized_Count"

#COVIDUS DataSet Measures
HOSPITALIZED_CURRENTLY_MEASURE = "hospitalized_currently"

# Define Placeholder Variables. These will be used in functions to ensure consistency
US_Seasons = 0
```

```

US_Diffs = 0

NC_Seasons = 0
NC_Diffs = 0

```

In this section helper functions will be defined. These helper functions are repeated numerous times in this analysis. As with the constant definitions above this section provides a single place to modify analysis, if necessary.

```

# Define Helper functions
RollingASECalc = function(values, phis, thetas, seasons, diffs, sampleSize, horizon){

ASEHolder = numeric()

for( i in 1:(length(values)-(sampleSize + horizon) + 1))
{
  forecasts = fore.aruma.wge(values[i:(i+(sampleSize-1))],phi = phis, theta = thetas, s = seasons, d = 0)

  ASE = mean((values[(sampleSize+i):(sampleSize + i + (horizon) - 1)] - forecasts$f)^2)

  ASEHolder[i] = ASE
}

return(ASEHolder)
}

RollingASECalcMLP = function(values, sampleSize, horizon){

ASEHolder = numeric()

for( i in 1:(length(values)-(sampleSize + horizon) + 1))
{
  forecasts = forecast(values[i:(i+(sampleSize-1))], h = horizon)

  ASE = mean((values[(sampleSize+i):(sampleSize+ i + (horizon) - 1)] - forecasts$mean)^2)

  ASEHolder[i] = ASE
}

return(ASEHolder)
}

GetRollingWindowASEInfo = function(modelName, ASEVals){

  hist(RollingASEUS7Day,main=paste("Histogram of ", modelName, " Windowed ASE"))
  WindowedASE_Mean = mean(ASEVals)
  WindowedASE_Median = median(ASEVals)

  summary(ASEVals)

#Print Mean WindowedASE
  print(paste(modelName, " Windowed ASE Mean: ",WindowedASE_Mean))
}

```

```

#Print Median WindowedASE
print(paste(modelName, " Windowed ASE Median: ",WindowedASE_Median))
}

RollingASECalcVAR = function(val1, val2, pVal, sampleSize, horizon){

ASEHolder = numeric()

for( i in 1:(length(val1)-(sampleSize + horizon) + 1))
{

X = cbind(val1[1:(i+(sampleSize-1))],val2[i:(i+(sampleSize-1))])
#names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)

lsfit=VAR(X,p=pVal,type="const")
preds=predict(lsfit,n.ahead=horizon)

ASE = mean((val1[(sampleSize+i):(sampleSize+ i + (horizon) - 1)] - preds$fcst$y1[,1])^2)

ASEHolder[i] = ASE
}

return(ASEHolder)
}

```

This section of code is where the data sets are loaded for analysis from the COVID19US package. Two data frames are defined to hold all of the data:

- df_US_Percent_Positive - contains a daily listing of US percent positive values
- df_NC_Percent_Positive - contains a daily listing of NC percent positive values

As previously stated the percent positive metric is the number of positive cases divided by the total number of cases. For analysis purposes the values are still decimal values and have not been converted to percentages.

```

#Percent Positive Definition Articles
#https://www.cnn.com/2020/11/17/health/covid-19-percent-positive-explainer/index.html
#https://www.jhsph.edu/covid-19/articles/covid-19-testing-understanding-the-percent-positive.html#:~:text

# Load the US Daily Data
df_US_Daily_Data = get_us_daily()

#Filter the data and create a dataframe that is just percent positive cases
df_US_Percent_Positive = data.frame(df_US_Daily_Data$date, df_US_Daily_Data$positive, df_US_Daily_Data$negative)
names(df_US_Percent_Positive) = c(COL_DATE_VAL, COL_POSITIVE_CASES, COL_TOTAL_TESTS, COL_PERCENT_POSITIVE)

#Sort the data by date descending for the initial realization. The data comes in reverse order and we want it in ascending order
df_US_Percent_Positive = df_US_Percent_Positive %>% arrange(as.Date(Date_Val,format=DATE_FORMAT))

# Load the daily NC data
df_NC_Daily_Data = get_states_daily(state=STATE_VAL)

#Filter the data and create a dataframe that is just percent positive cases
df_NC_Percent_Positive = data.frame(df_NC_Daily_Data$date, df_NC_Daily_Data$positive, df_NC_Daily_Data$negative)
names(df_NC_Percent_Positive) = c(COL_DATE_VAL, COL_POSITIVE_CASES, COL_TOTAL_TESTS, COL_PERCENT_POSITIVE)

```

```

names(df_NC_Percent_Positive) = c(COL_DATE_VAL, COL_POSITIVE_CASES, COL_TOTAL_TESTS, COL_PERCENT_POSITI

#Sort the data by date descending for the initial realization (Order doesn't really matter)
df_NC_Percent_Positive = df_NC_Percent_Positive %>% arrange(as.Date(Date_Val,format=DATE_FORMAT))

```

The following chunk of code loads in the hospitalization data from the COVIDUS package. As with the percent positive data, two data frames will be created:

- df_US_Hospital - contains a daily listing of hospitalizations for the total US
- df_NC_Hospital - contains a daily listing of hospitalizations for all of NC

With the hospitalization data there are some missing valued identified. In tracking back the data, it looks as though these missing values were early in the COVID-19 data reporting. For analysis purposes the assumption will be made that there were no reported hospitalizations during these specific dates and the values will be set to zero.

```

#https://kjhealy.github.io/covdata/articles/codebook.html
#data current as of 11/25/2020

#Get the number of hospitalized patients in the US
df_US_Hospitalizations_Total = covus %>% filter(measure %in% c(HOSPITALIZED_CURRENTLY_MEASURE))

#Build short dataframe and sort the data
df_US_Hospital = data.frame(df_US_Hospitalizations_Total$date,df_US_Hospitalizations_Total$count)
names(df_US_Hospital) = c(COL_DATE_VAL,COL_HOSPITAL_COUNT)

df_US_Hospital = df_US_Hospital %>% arrange(as.Date(Date_Val,format=HOSPITAL_DATE_FORMAT))

# for missing values assume no hospitalizations
df_US_Hospital[is.na(df_US_Hospital)] = 0

# summarize the data so there are single dates
df_US_Hospital <- df_US_Hospital %>%
  mutate(Date_Val = as.Date(Date_Val, format=HOSPITAL_DATE_FORMAT)) %>%
  group_by(Date_Val) %>%
  summarise(total_count=sum(Hospitalized_Count))

names(df_US_Hospital) = c(COL_DATE_VAL,COL_HOSPITAL_COUNT)

#Get the number of hospitalized patients in NC
df_NC_Hospitalizations_Total = covus %>% filter(state==STATE_VAL & measure %in% c(HOSPITALIZED_CURRENTLY_MEASURE))

#Build short dataframe and sort the data
df_NC_Hospital = data.frame(df_NC_Hospitalizations_Total$date,df_NC_Hospitalizations_Total$count)
names(df_NC_Hospital) = c(COL_DATE_VAL,COL_HOSPITAL_COUNT)

df_NC_Hospital = df_NC_Hospital %>% arrange(as.Date(Date_Val,format=DATE_FORMAT))

# for missing values assume no hospitalizations
df_NC_Hospital[is.na(df_NC_Hospital)] = 0

```

In the final chunk of code the four data frames defined above are combined into two data frames: one for total US and one for NC. Combining the dataframes in this way makes analysis easier and also allows for keeping

the data in sync. As noted previously, the percent positive data is updated daily while the hospitalization data lags. Combining the dataframes in this way ensures the data will be mapped to the lagged data

```
# align hospitalization and percent positive dataframes to ensure we're aligned completely. Hospitaliza  
df_US_Final = merge(df_US_Percent_Positive,df_US_Hospital)  
df_NC_Final = merge(df_NC_Percent_Positive,df_NC_Hospital)
```

As a final step in EDA, verify that there are no missing values and provide a quick look at summary statistics to ensure that the data seems reasonable.

```
summary(df_US_Final)  
  
##      Date_Val      Positive_Cases      Total_Tests      Percent_Positive  
##  Min.   :2020-01-22  Min.   : 0  Min.   : 1  Min.   :0.00000  
##  1st Qu.:2020-04-08  1st Qu.: 432490  1st Qu.: 2335944  1st Qu.:0.06308  
##  Median :2020-06-24  Median : 2368309  Median : 29871099  Median :0.07531  
##  Mean   :2020-06-24  Mean   : 3621028  Mean   : 51583853  Mean   :0.08237  
##  3rd Qu.:2020-09-09  3rd Qu.: 6302903  3rd Qu.: 91271748  3rd Qu.:0.10001  
##  Max.   :2020-11-25  Max.   :12581196  Max.   :184877012  Max.   :0.19016  
##  
## Hospitalized_Count  
##  Min.   : 0  
##  1st Qu.:28279  
##  Median :36437  
##  Mean   :34649  
##  3rd Qu.:51425  
##  Max.   :89959
```

```
summary(df_NC_Final)  
  
##      Date_Val      Positive_Cases      Total_Tests      Percent_Positive  
##  Min.   :2020-03-04  Min.   : 1  Min.   : 1  Min.   :0.03405  
##  1st Qu.:2020-05-09  1st Qu.: 14562  1st Qu.: 189248  1st Qu.:0.06826  
##  Median :2020-07-15  Median : 91266  Median :1284637  Median :0.07020  
##  Mean   :2020-07-15  Mean   :113813  Mean   :1635293  Mean   :0.09917  
##  3rd Qu.:2020-09-19  3rd Qu.:192915  3rd Qu.:2787919  3rd Qu.:0.07443  
##  Max.   :2020-11-25  Max.   :346506  Max.   :5084569  Max.   :1.00000  
##  
## Hospitalized_Count  
##  Min.   : 0.0  
##  1st Qu.: 512.0  
##  Median : 903.0  
##  Mean   : 806.1  
##  3rd Qu.:1113.0  
##  Max.   :1811.0
```

From the results above for both the US and NC dataframes there are no missing values identified. Additionally, the values for each column in the dataframe look to be reasonable. It can also be seen that the maximum date column aligns to the known data date of November 25, 2020.

Time Series Analysis

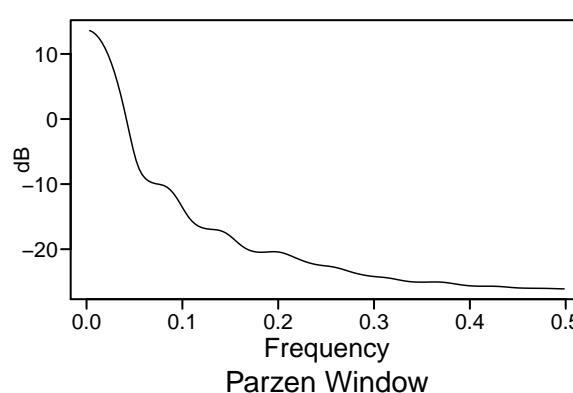
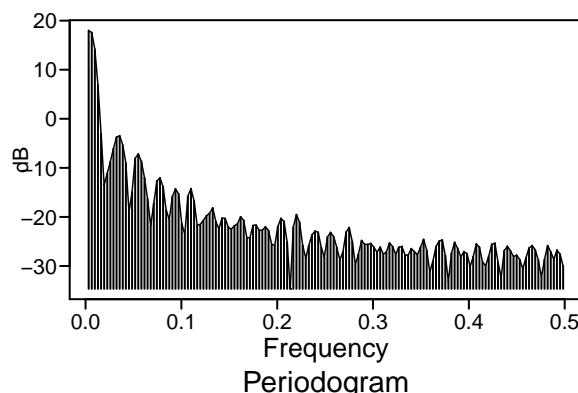
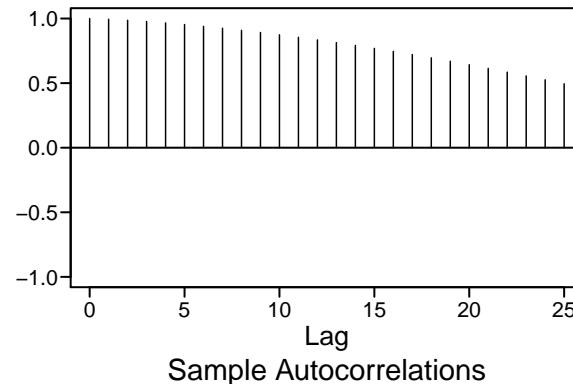
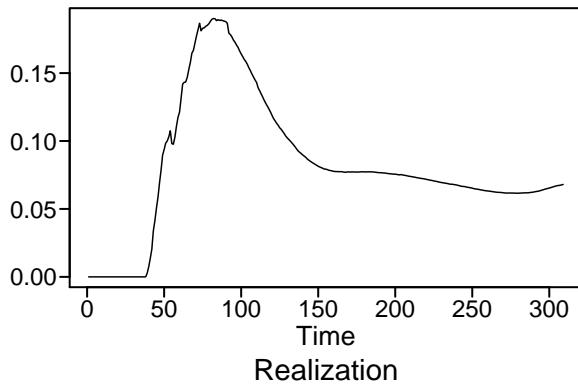
The time series analysis for both the United States (US) and North Carolina (NC) will be performed throughout the remaining sections of this document. The analysis will start at the US level and then move to NC specifically. For each of the geographic regions a univariate and multi-variate analysis will be performed. The univariate analysis will focus only on the single variable percent positive. The multi-variate analysis will investigate any interrelations between hospitalizations and the percent positive rate. Within the univariate and multi-variate analysis different time series modeling techniques will be leveraged.

In order to compare models through this analysis, the rolling window ASE metric will be used. This metric creates a window into the forecasted data and compares it to the actual data. This allows for an analysis of how well the time series model is able to forecast the data. When comparing ASE metrics the lower the metric the better the model.

US Univariate Analysis

ARIMA Analysis The initial time series model that will be created will be an ARIMA-type model. These models are relatively easy to build and provide good insight into the data characteristics. The first step in the ARIMA modeling process is to simply plot the realization of the data along with ACF and the spectral density. The ACF plot will provide an indication of model characteristics that may need to be taken into account. The spectral density plot will also provide an indication of model characteristics and also allows for confirming seasonality within the data, if there is any.

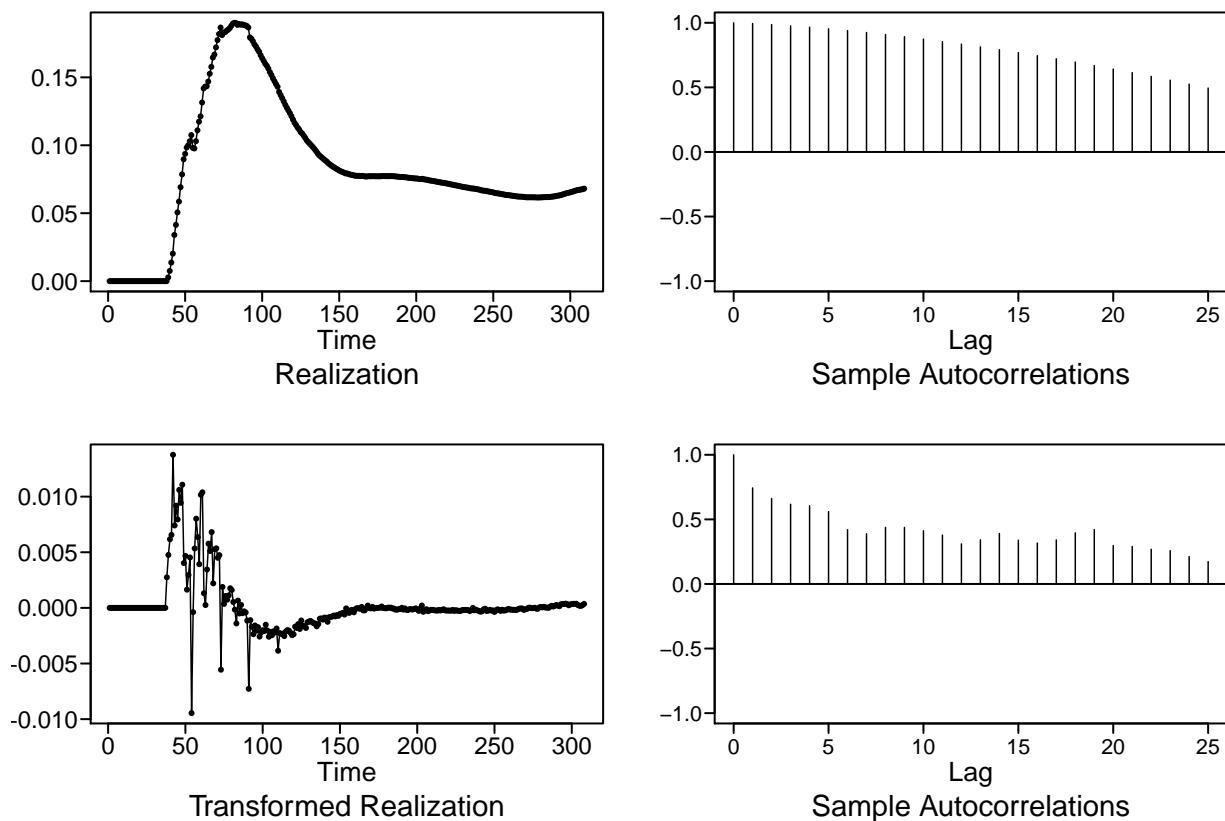
```
# Look at active cases in the US
USPlotTSOut = plott.sample.wge(df_US_Final$Percent_Positive)
```



```
# Spectral Density shows large peak around zero
# ACF shows slow damping. Assume non stationary. Difference the data
```

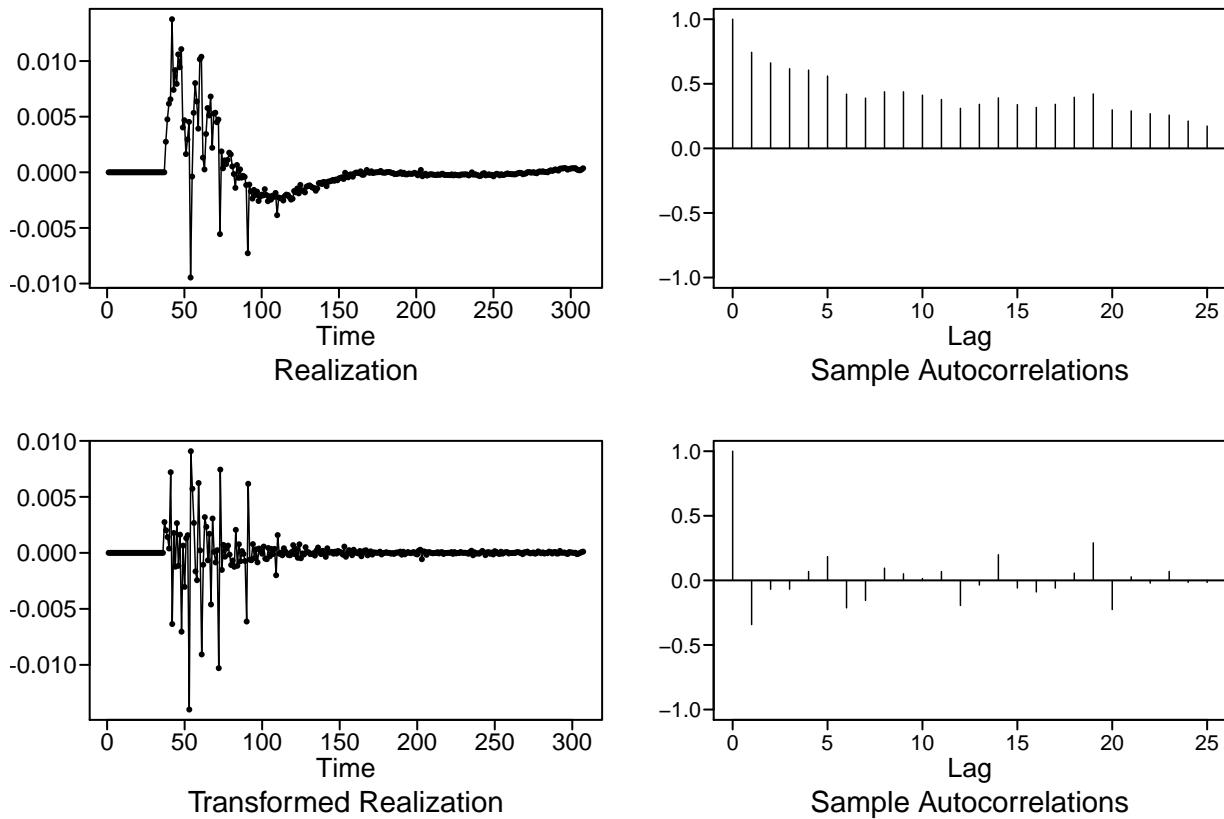
From the realization (top-left plot), there is a large peak and then the data looks to start to level off a bit. The realization does not indicate any wandering behavior, where the data is increasing or decreasing over time, nor do is there any indication of cyclic behavior. Moving to the ACF plot (top-right plot), the data looks to be slowly damping. The spectral density plot (bottom-right plot) indicates a large peak at zero and no additional peaks in the data. This indicator coupled with the slow damping in the ACF plot would indicate that the data may not be stationary at this point in time. In order to attempt to obtain more stationary data a difference calculation will be executed on the data.

```
dfUSCasesDiff = df_US_Final$Percent_Positive
dfUSCasesDiff = artrans.wge(dfUSCasesDiff,phi.tr=1)
```



After differencing the data one time, the realization of the differenced data (bottom-left plot) is starting to look a bit more stationary. In addition, the differenced ACF plot (bottom-right plot) begins to indicate a bit more of a slowly damping behavior. These factors indicate that a second difference of the data may be in order to ensure stationarity.

```
dfUSCasesDiff = artrans.wge(dfUSCasesDiff,phi.tr=1)
```



Executing the second difference of the data, the differenced realization (bottom-left plot) looks much more stationary. This is further confirmed with the ACF plot (bottom-right plot). At this point in time, the data can be considered stationary and modeling steps can move forward. Leveraging the differenced data, the AR(p) and MA(q) component values can be determined. In order to determine the best fit for these components both the AIC and BIC metrics will be used. The AR(p) component will be searched from zero (0) to fifteen (15) and the MA(q) component will be searched from zero (0) to fifteen (15).

```
#Get estimates for AIC - 6,3 - best; 1-3 common
aic5.wge(dfUSCasesDiff,p=0:15,q=0:5)
```

```
## -----WORKING... PLEASE WAIT...
##
##
## Five Smallest Values of aic

##      p      q      aic
## 93    15     2   -13.10160
## 79    13     0   -13.09034
## 77    12     4   -13.09029
## 78    12     5   -13.08535
## 83    13     4   -13.08488
```

```
#Get estimates for BIC - 1,1 - best; 1-3 common
aic5.wge(dfUSCasesDiff,p=0:15,q=0:5,type="bic")
```

```
## -----WORKING... PLEASE WAIT...
```

```

## 
## 
## Five Smallest Values of bic

##      p      q      bic
## 36     5      5 -12.92883
## 79    13      0 -12.92038
## 42     6      5 -12.91324
## 80    13      1 -12.90179
## 85    14      0 -12.90178

```

From the AR(p) and MA(q) selection process above, there is a commonality between p=13 and q=0 across both the AIC and BIC calculations. Given the commonality between both metrics this will be the starting point for the model.

```

# Set the p, q, and d values for later use
USSelPVal = 13
USSelQVal = 0
US_Diffs = 2

```

Now that the p and q values are defined for the data the actual coefficients of the ARIMA model will be estimated. These coefficients will assist in providing a true understanding of the model.

```

#Estimate off of 13,0 model since it's consistent between AIC and BIC
US.est = est.arma.wge(dfUSCasesDiff, p=USSelPVal, q=USSelQVal)

```

```

## 
## Coefficients of Original polynomial:
## -0.4982 -0.1908 -0.1410 0.0094 0.0652 -0.3438 -0.3888 -0.0838 -0.0305 0.0076 0.1051 -0.1985 -0.2957
## 
## Factor          Roots          Abs Recip   System Freq
## 1-0.4388B+0.9180B^2  0.2390+-1.0160i  0.9581    0.2132
## 1+1.2690B+0.8718B^2 -0.7278+-0.7857i  0.9337    0.3689
## 1+0.2486B+0.8640B^2 -0.1438+-1.0662i  0.9295    0.2713
## 1-1.3802B+0.8088B^2  0.8532+-0.7130i  0.8994    0.1108
## 1-1.6764B+0.7887B^2  1.0628+-0.3720i  0.8881    0.0536
## 1+1.6252B+0.7880B^2 -1.0313+-0.4534i  0.8877    0.4341
## 1+0.8508B            -1.1754           0.8508    0.5000
## 
## 
```

```

paste("White Noise Variance: ", US.est$avar)

```

```

## [1] "White Noise Variance:  1.88507978282692e-06"

```

```

paste("Data Mean: ", mean(df_US_Final$Percent_Positive))

```

```

## [1] "Data Mean:  0.0823671283308277"

```

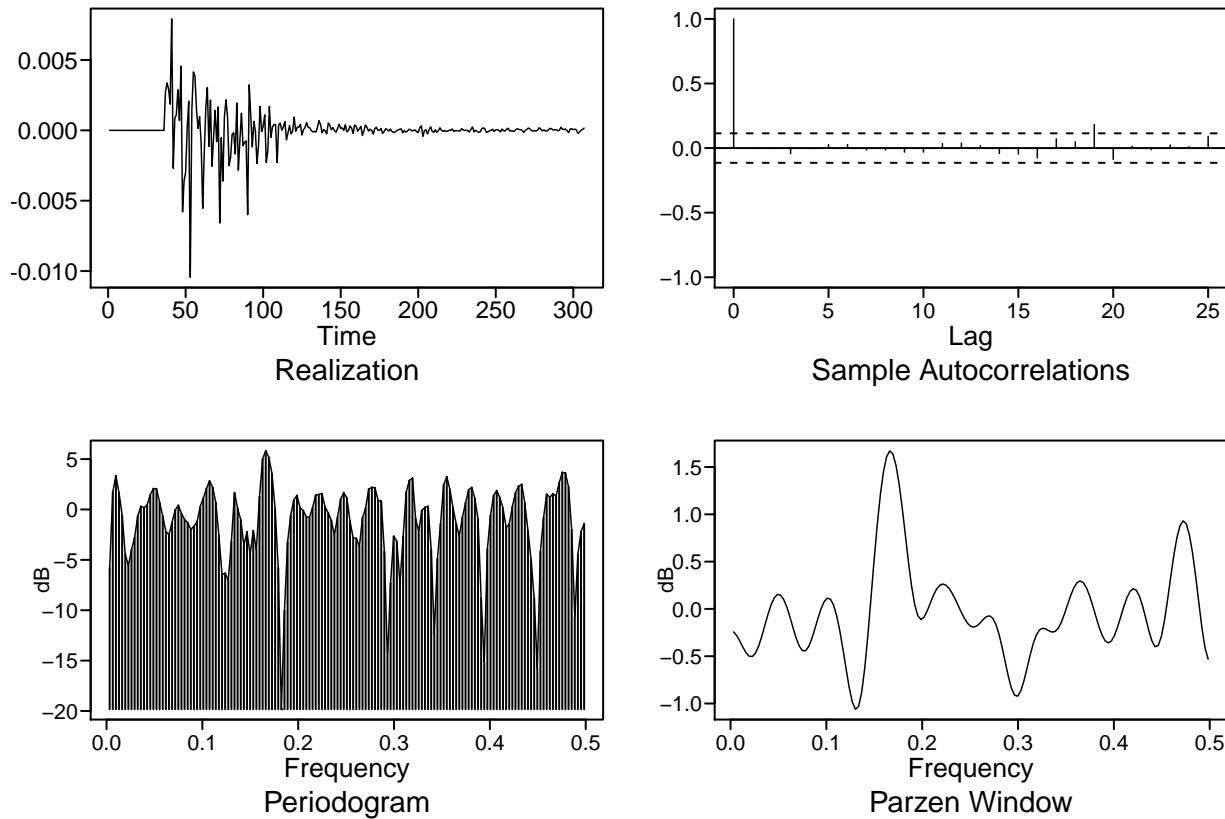
From the output above the coefficients of the model are displayed as well as the factor table of the model. The factor table indicates the influence each individual factor has on the model and whether the root of

that factor is real or complex. From the factor table, there are six (6) complex roots. Of the seven (7) total factors the first three factors have high influence as they are close to 1.

Continuing the model development, the residuals from the model parameter estimates need to be evaluated. These will be evaluated in two ways: visually and through the Ljung-Box test. Each of these methods will help to determine if the residuals in the model are white noise or not. Modeling down to white noise would indicate that there is minimal to no “information” left in the remaining data and thus the model would contain everything. However, even with the residuals not being white noise the model can still be leveraged.

In evaluating the residuals for white noise the first step will be a visual inspection of the residual ACF plot. The residual ACF plot, shown below (top-right), indicates that there is only white noise left in the model. Only one of the autocorrelations is outside the limits which is roughly expected at the shown timeframe.

```
#Test the residuals for white noise using visual test
USResPlots = plotts.sample.wge(US.est$res, arlimits=TRUE)
```



The Ljung-Box test is another test that can be leveraged to test for white noise in the residuals. The Ljung-Box test evaluates the autocorrelations as a group with the null hypothesis indicating that all the autocorrelations together equal zero. In order to determine if the null hypothesis is accepted or rejected, p-values from the Ljung-Box test will be evaluated. Each p-value will be tested against an alpha of 0.05. This would indicate a 95% confidence level. It is best practice to execute the Ljung-Box test twice with differing values of the K parameter. In this case K will be set to 24 and 48 respectively.

```
#Test the residuals for white noise using Ljung-Box test
ljung.wge(US.est$res, p=USSelPVal, q=USSelQVal)
```

```
## Obs -0.003769487 -0.003936467 -0.04289976 -0.003829373 0.02670438 0.02676598 -0.01487848 -0.0151587
```

```

## $test
## [1] "Ljung-Box test"
##
## $K
## [1] 24
##
## $chi.square
## [1] 22.16577
##
## $df
## [1] 11
##
## $pval
## [1] 0.02311999

ljung.wge(US.est$res,p=USSelPVal,q=USSelQVal,K=48)

## Obs -0.003769487 -0.003936467 -0.04289976 -0.003829373 0.02670438 0.02676598 -0.01487848 -0.0151587 ...

## $test
## [1] "Ljung-Box test"
##
## $K
## [1] 48
##
## $chi.square
## [1] 60.69695
##
## $df
## [1] 35
##
## $pval
## [1] 0.004512555

```

The results of the Ljung-Box test executions confirm that for K=24 the p-value is 0.0231 which at alpha = 0.05 would indicate a failure to reject the null hypothesis. At K=48, the p-value is 0.0045 which at alpha=0.05 would indicate a rejection of the null hypothesis. Hence, there is a bit of a mixed result. Visually it seems that the residuals are white noise and one of the Ljung-Box test also supports this. The second Ljung-Box test does not support white noise residual. Given that there does seem to be some indication of white noise in the model residuals, modeling will continue.

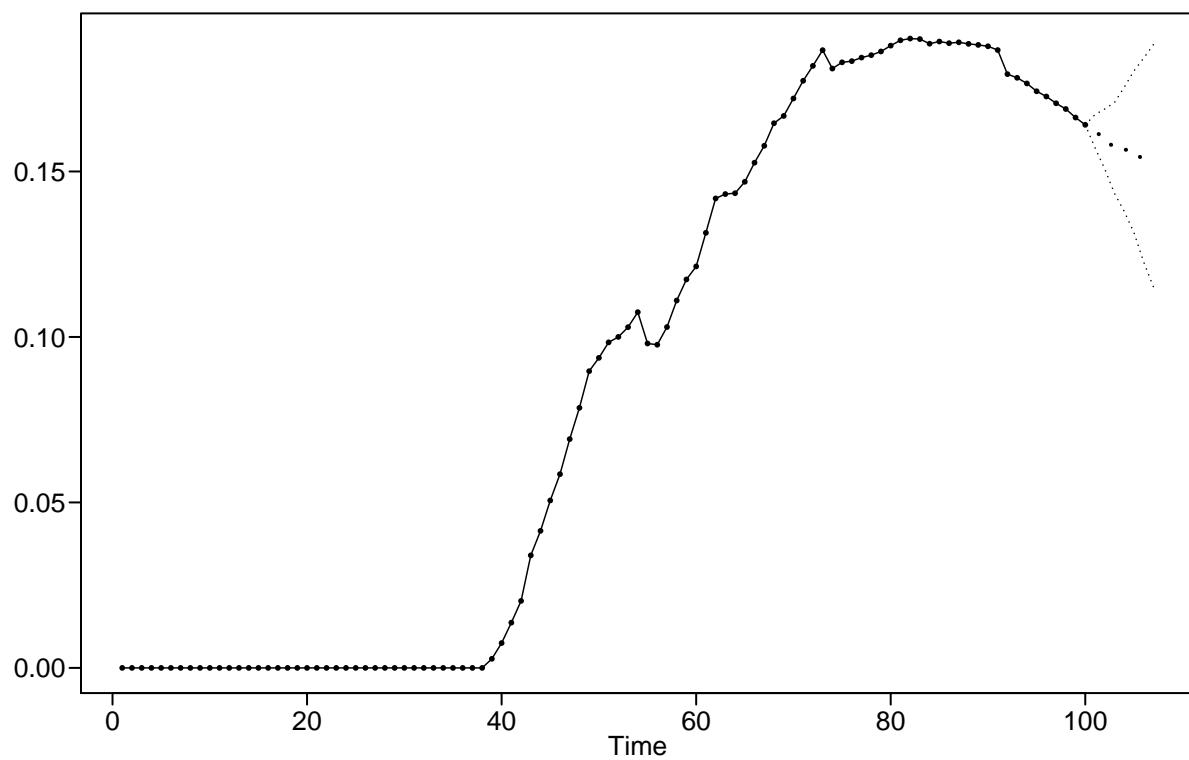
Thus the final model that will be leveraged will be an ARIMA(13,2,0).

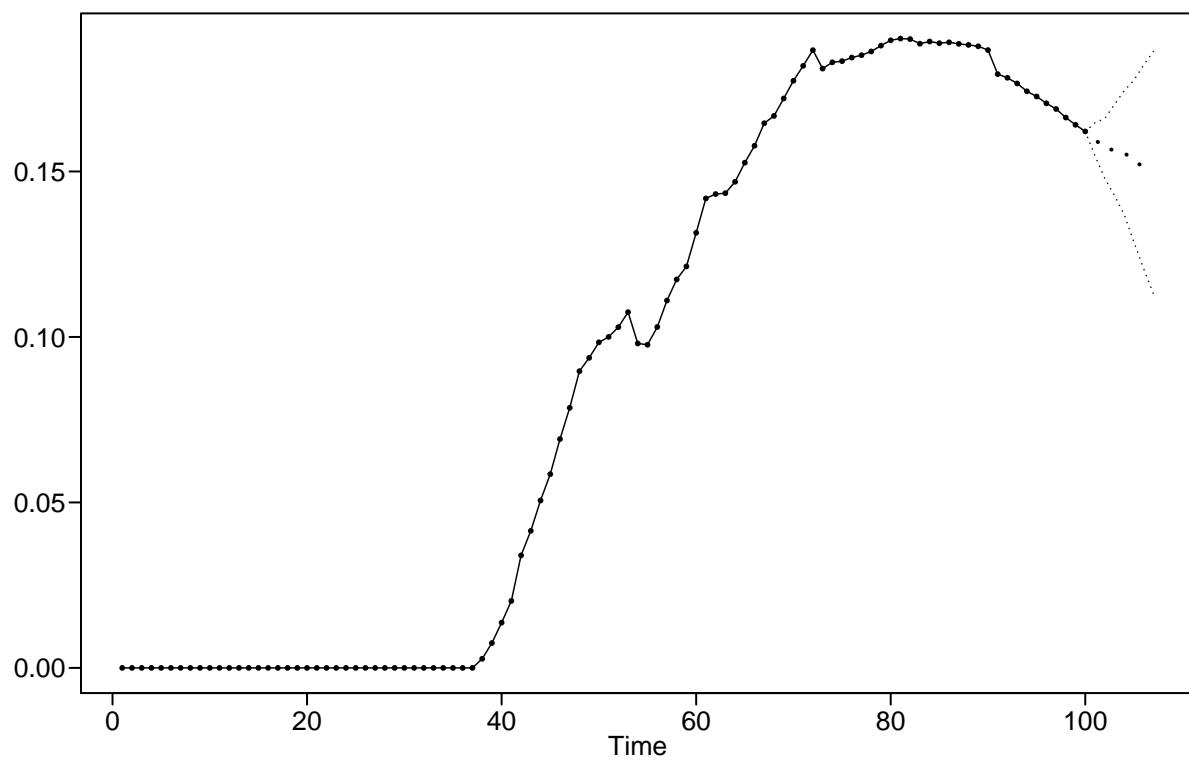
In order to evaluate this model, the data will be forecasted for a short term horizon of 7 days and a long term horizon of 90 days (3 months), The Rolling Window ASE will be leveraged as the evaluation metric for both of these horizons.

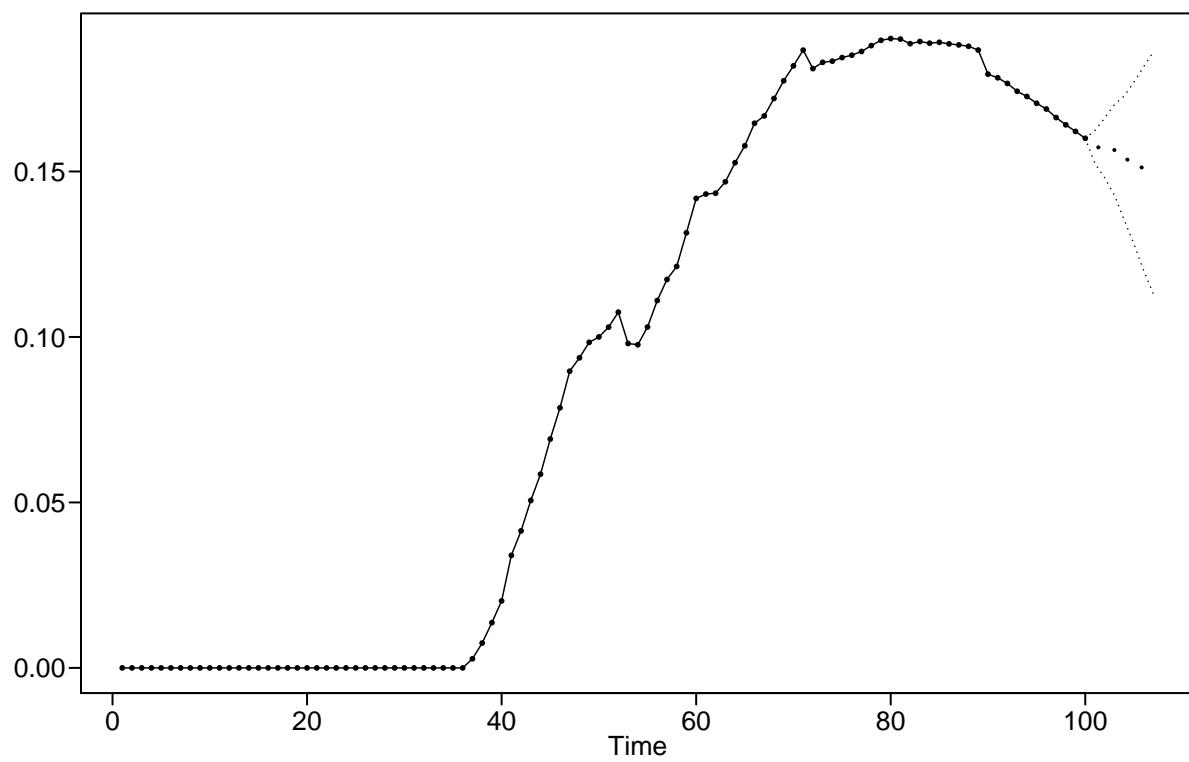
```

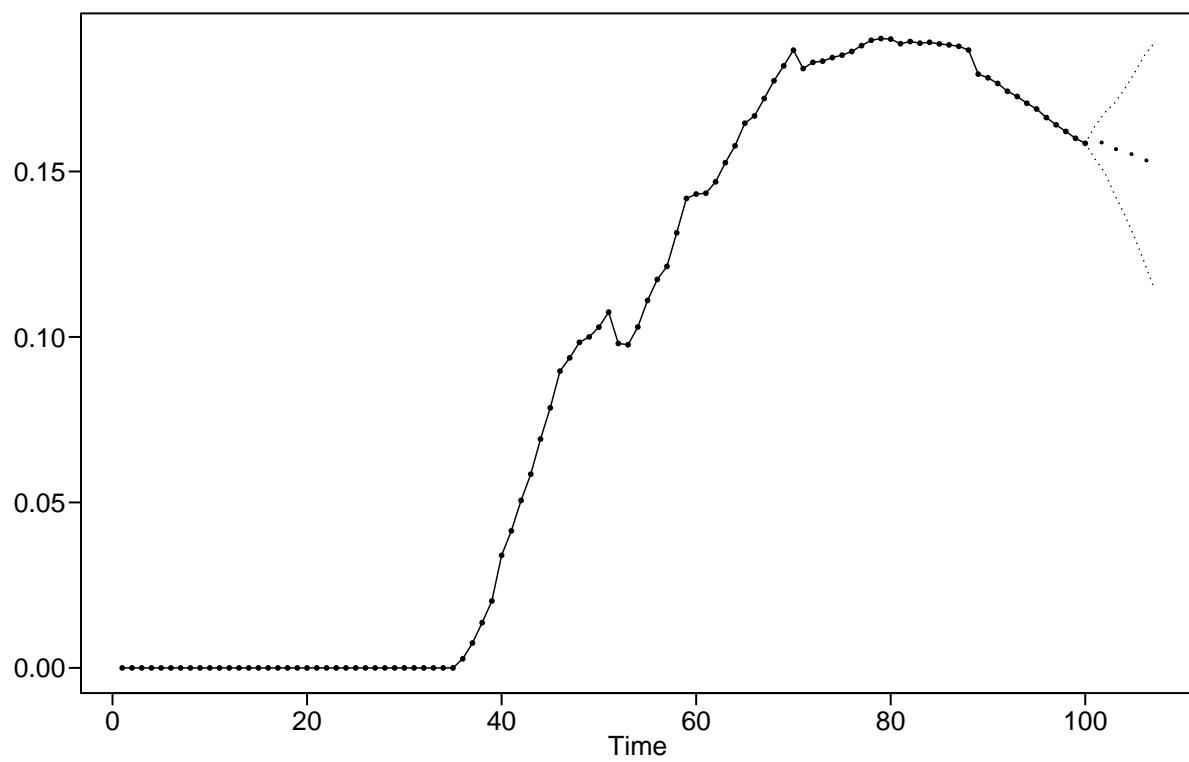
#Rolling Window ASE Calc - 7 Days
RollingASEUS7Day = RollingASECalc(df_US_Final$Percent_Positive,phis=US.est$phi, thetas = US.est$theta, ...

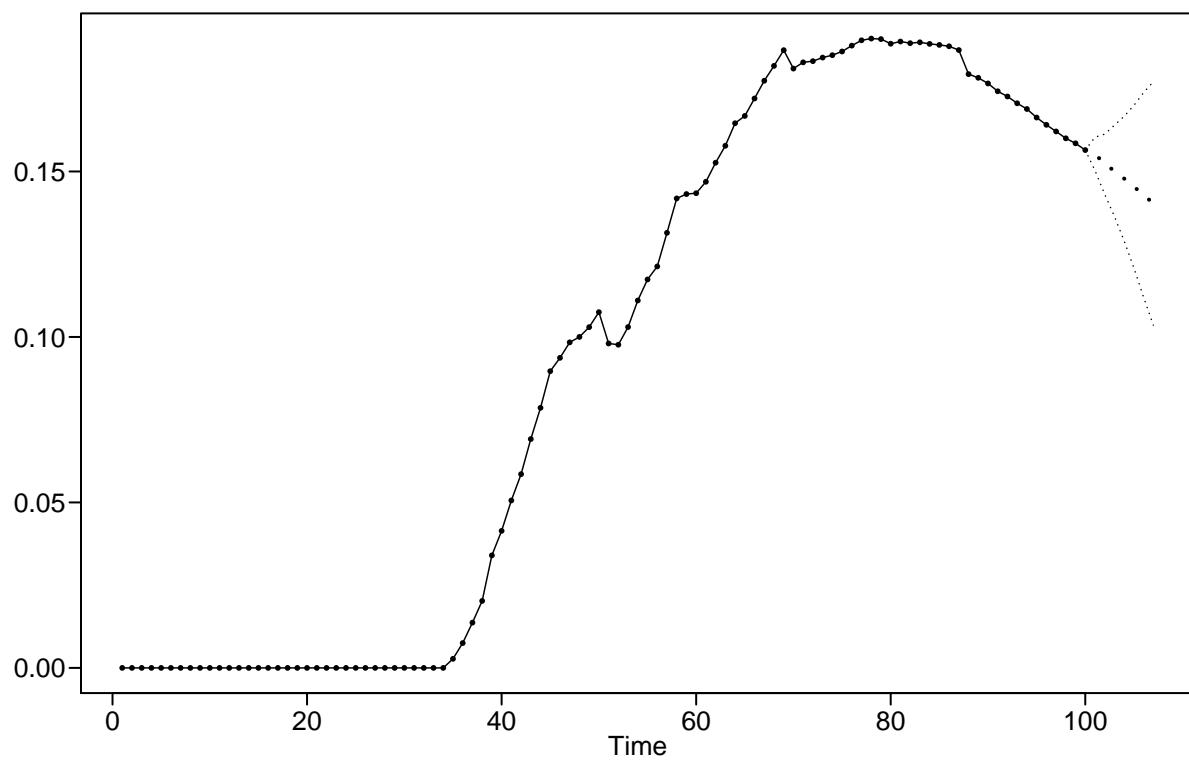
```

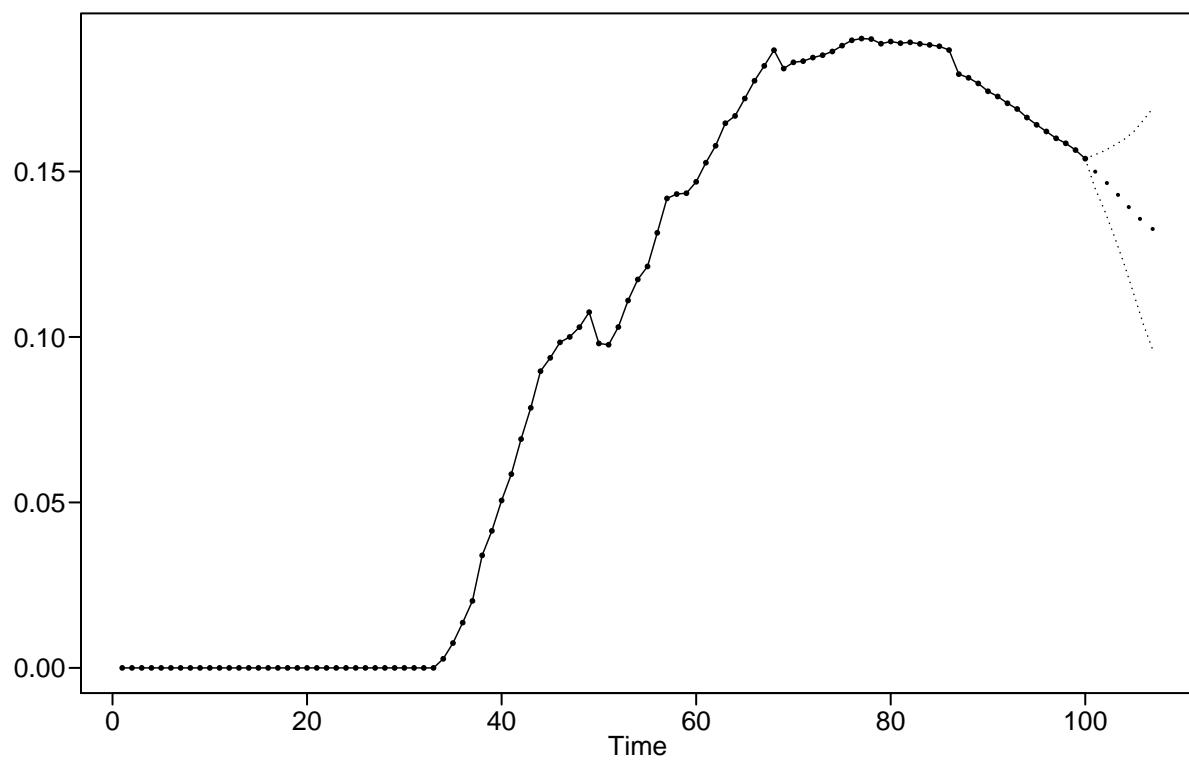


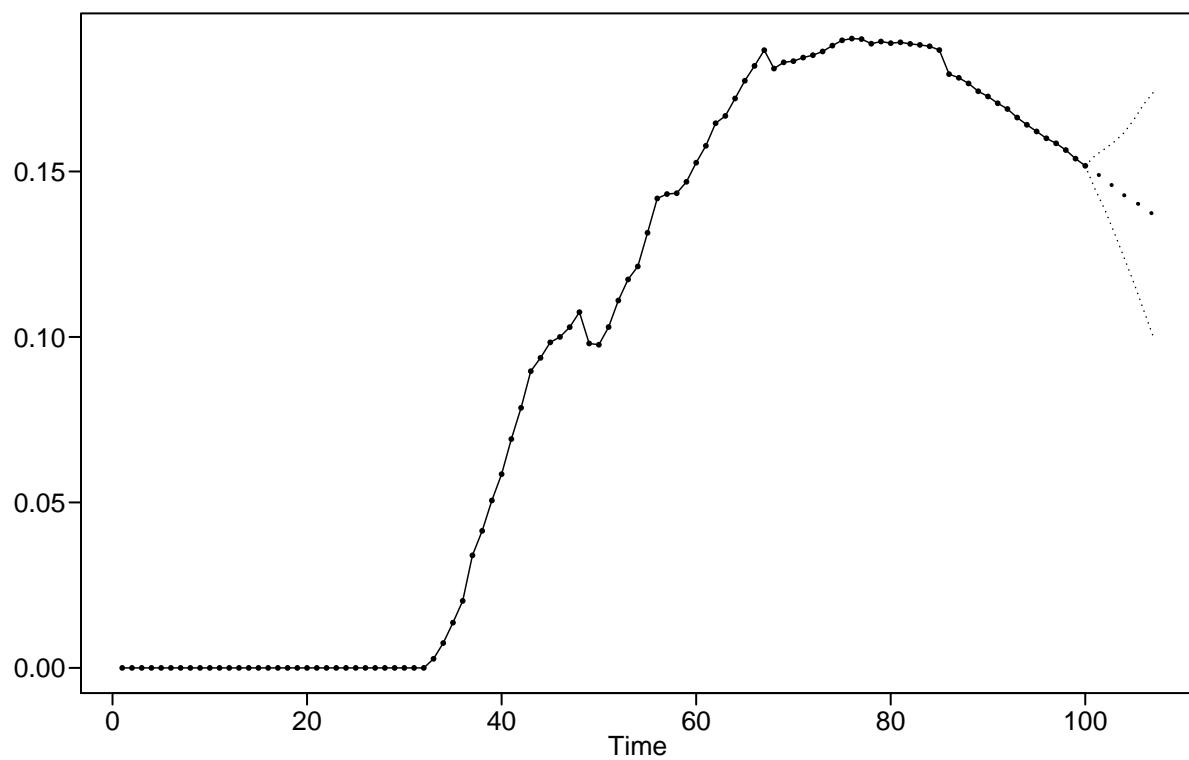


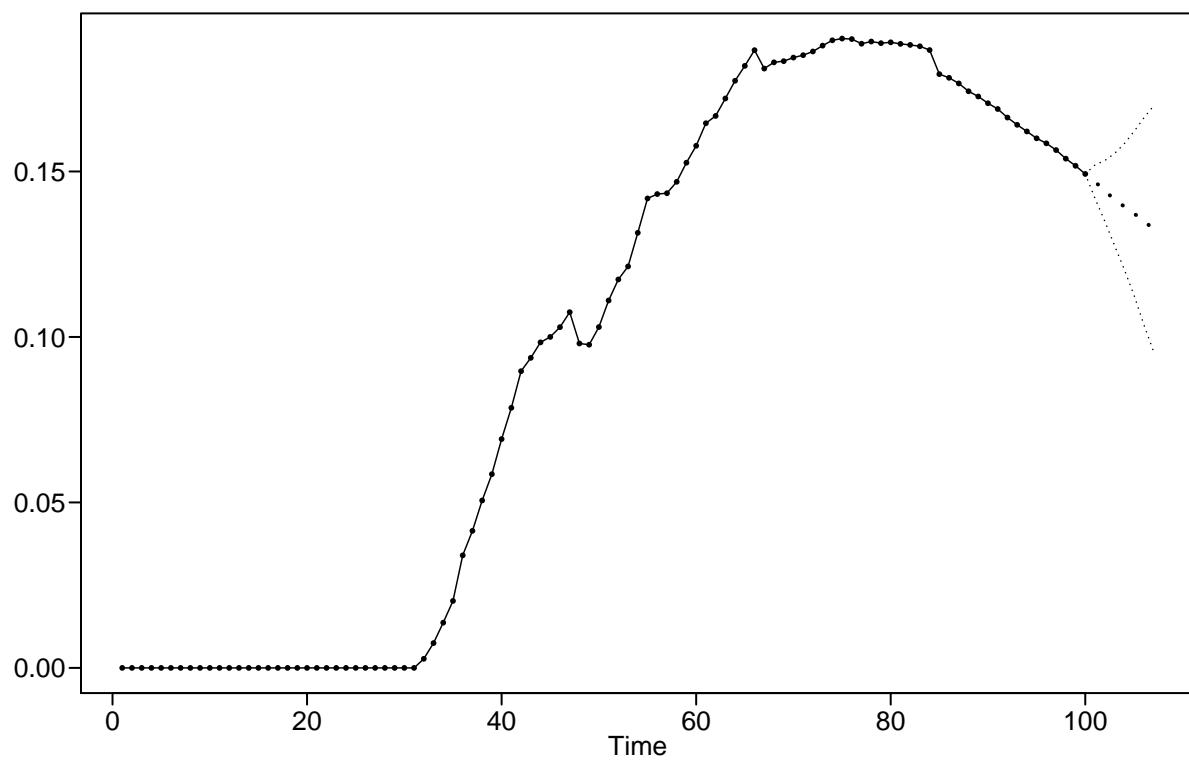


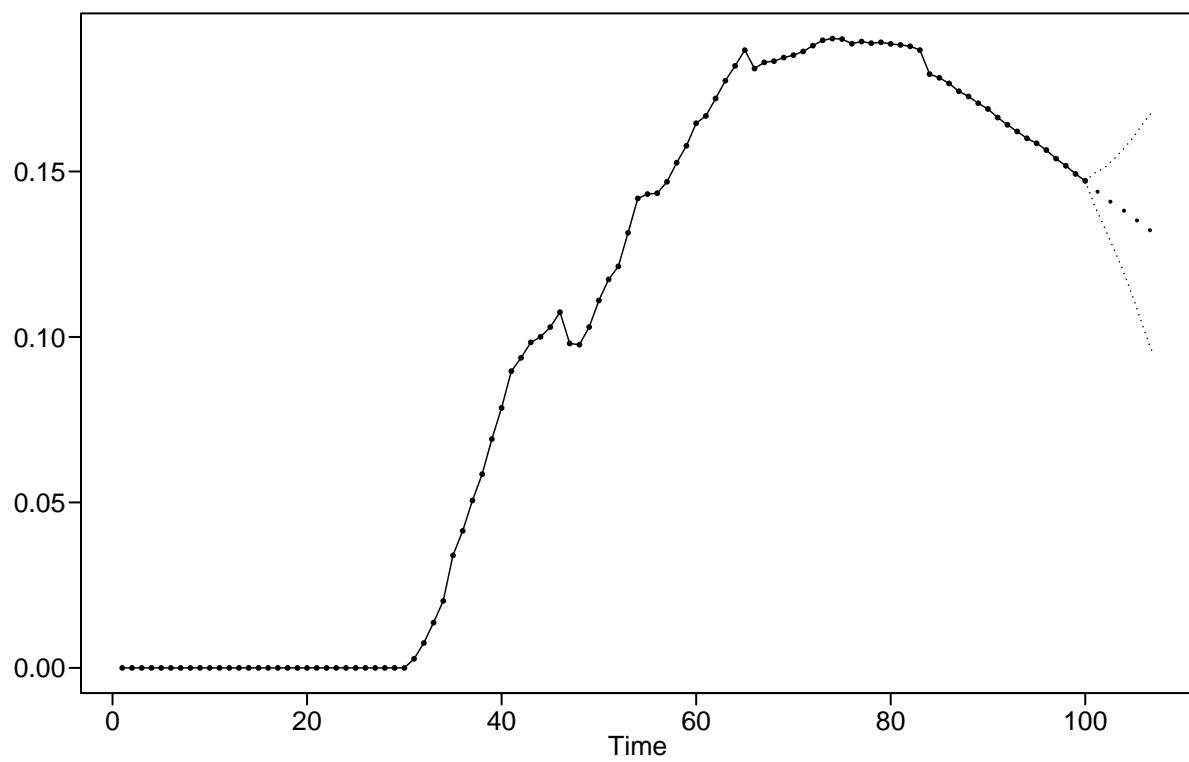


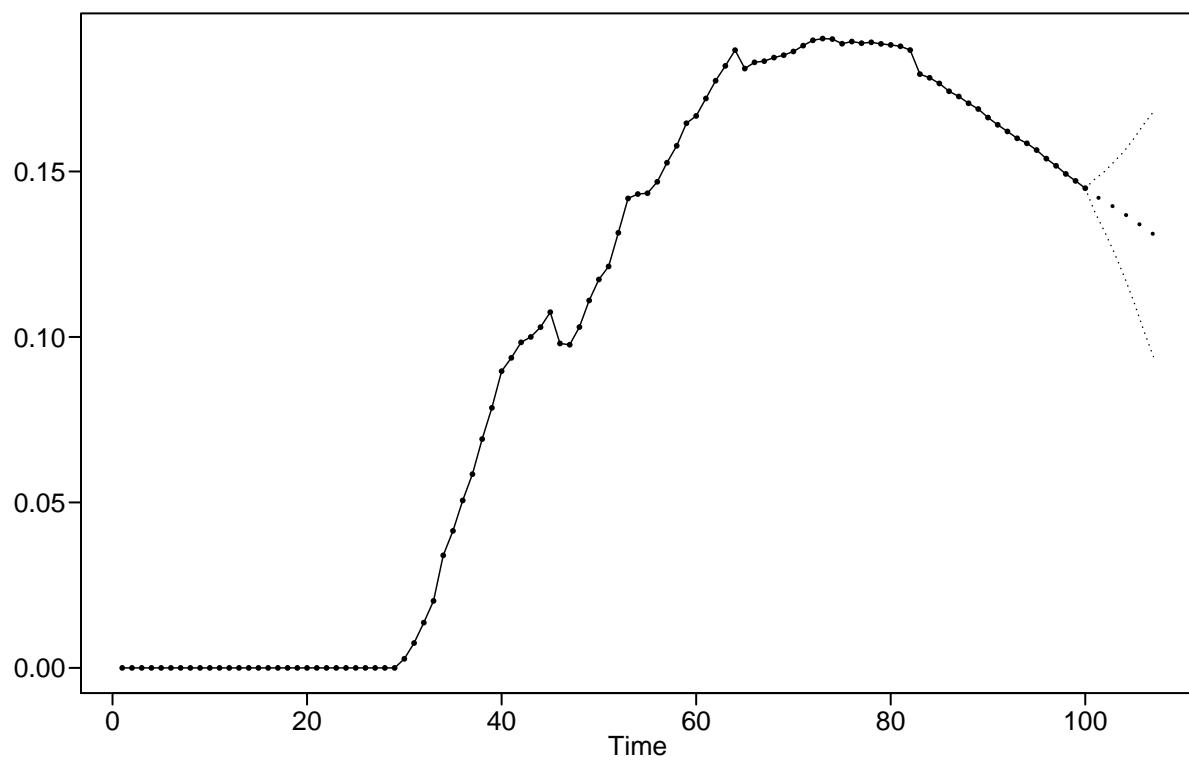


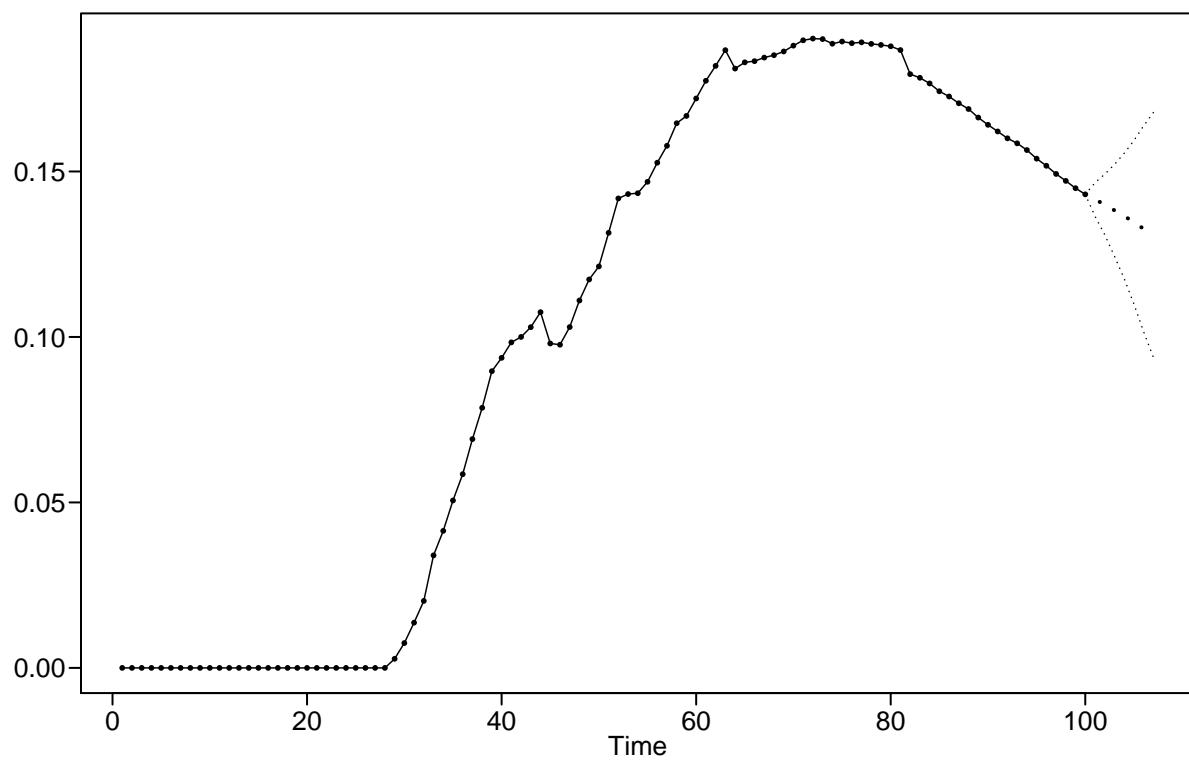


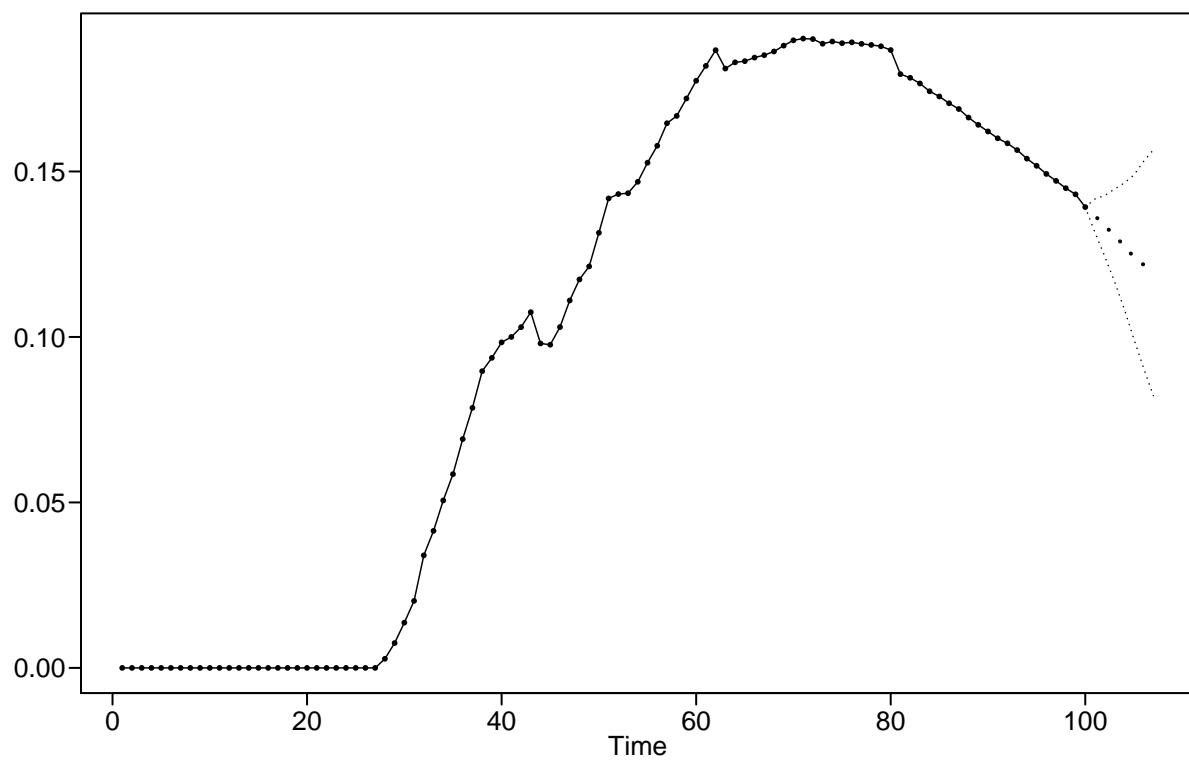


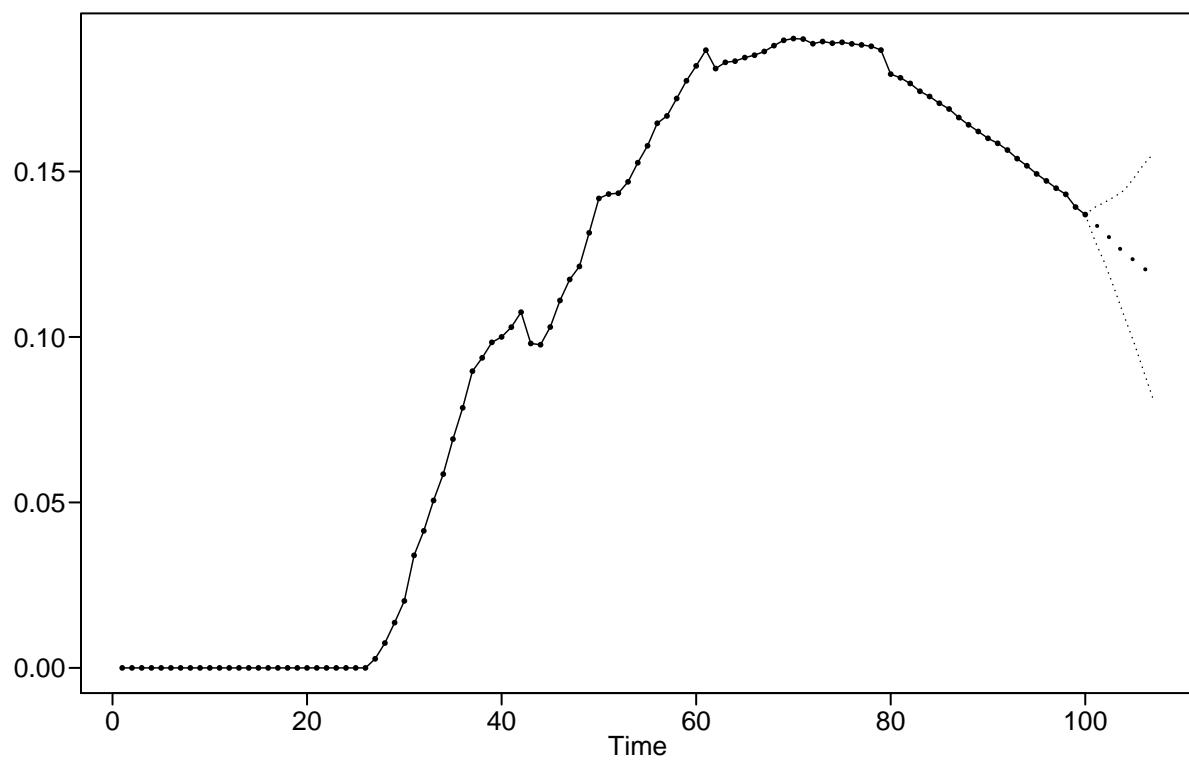


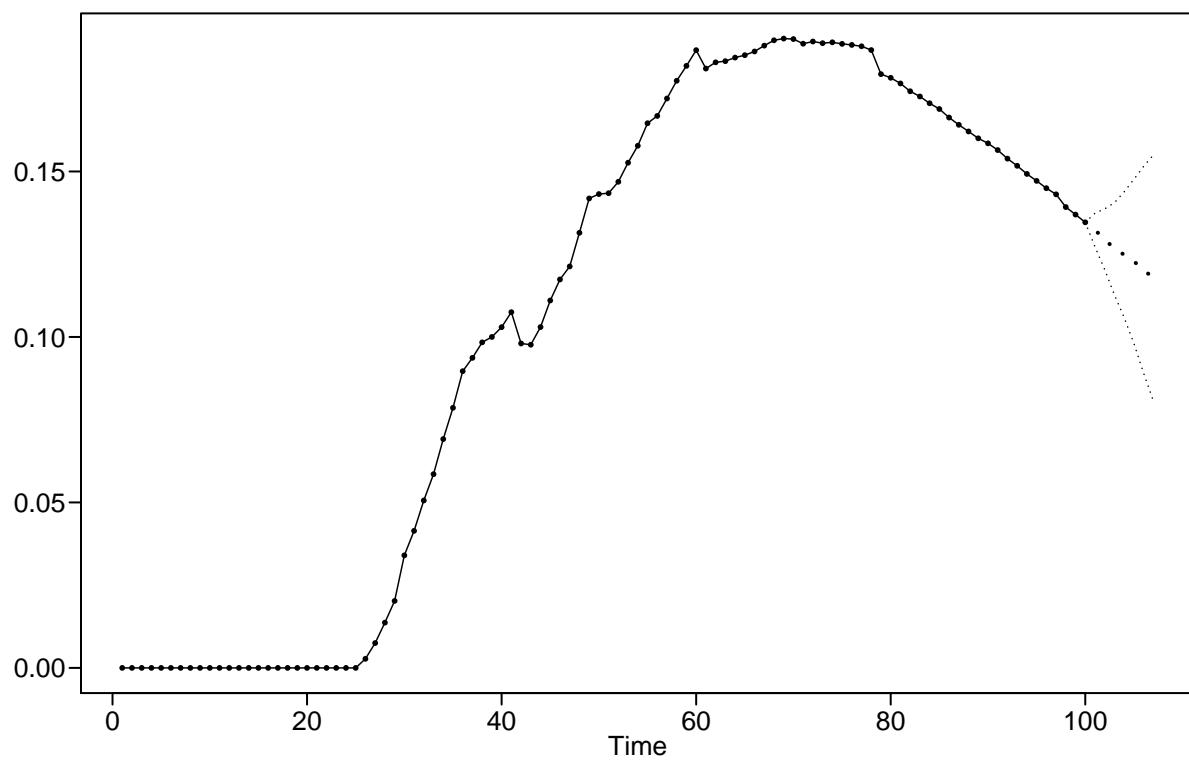


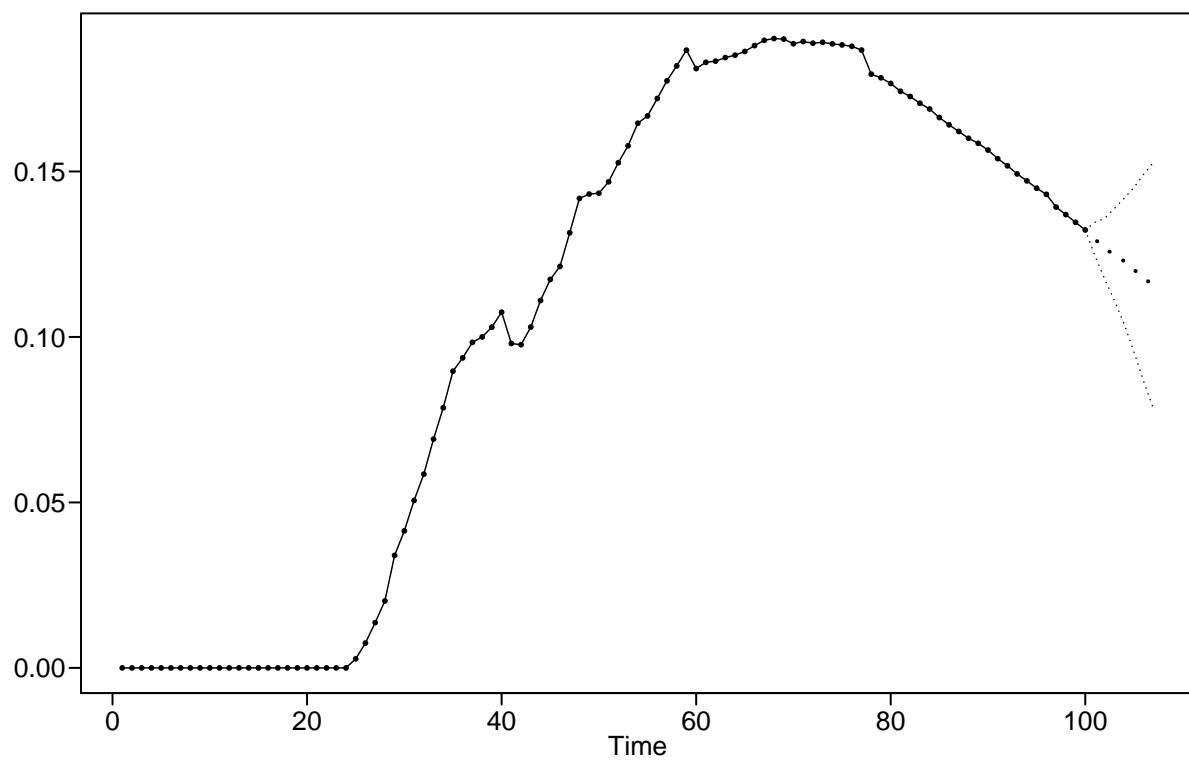


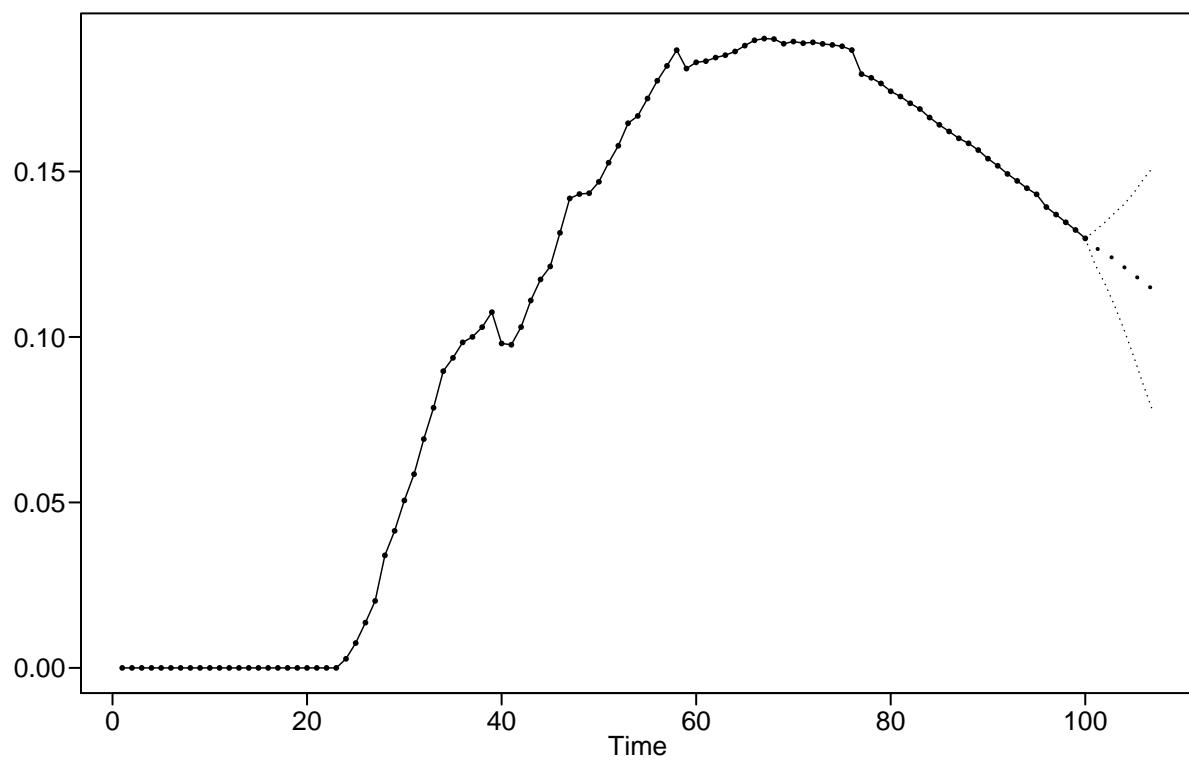


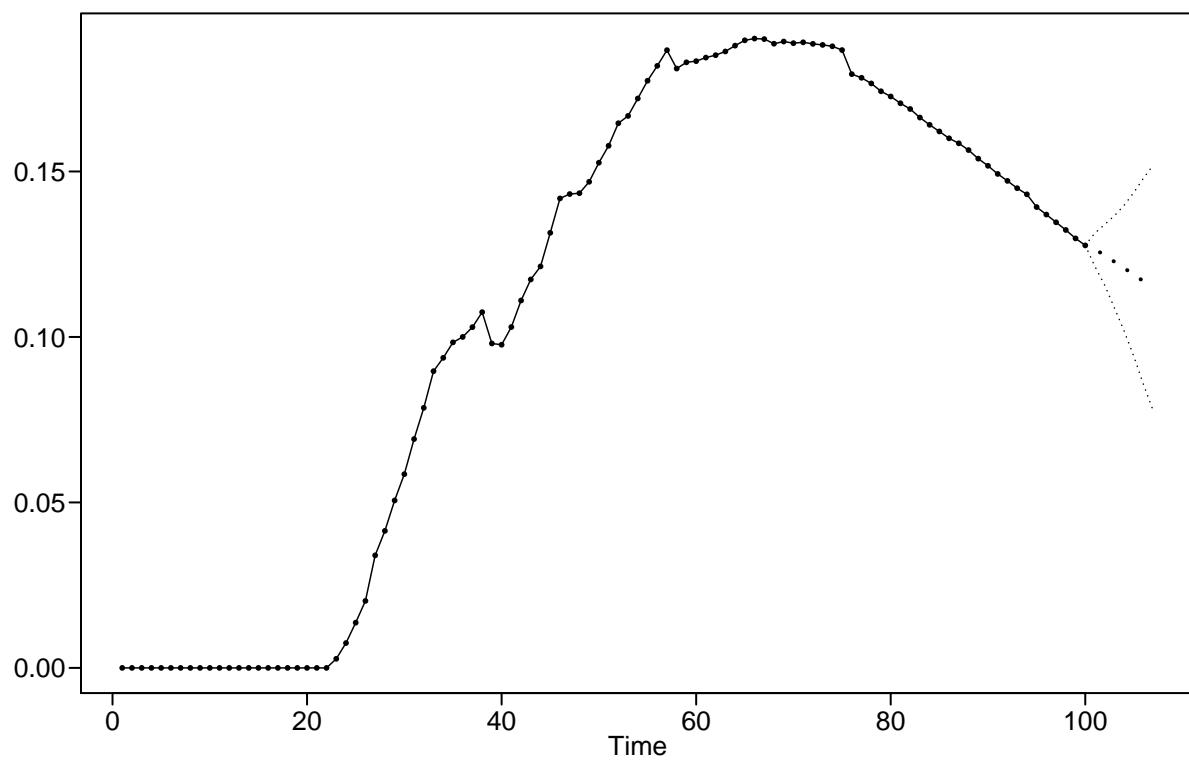


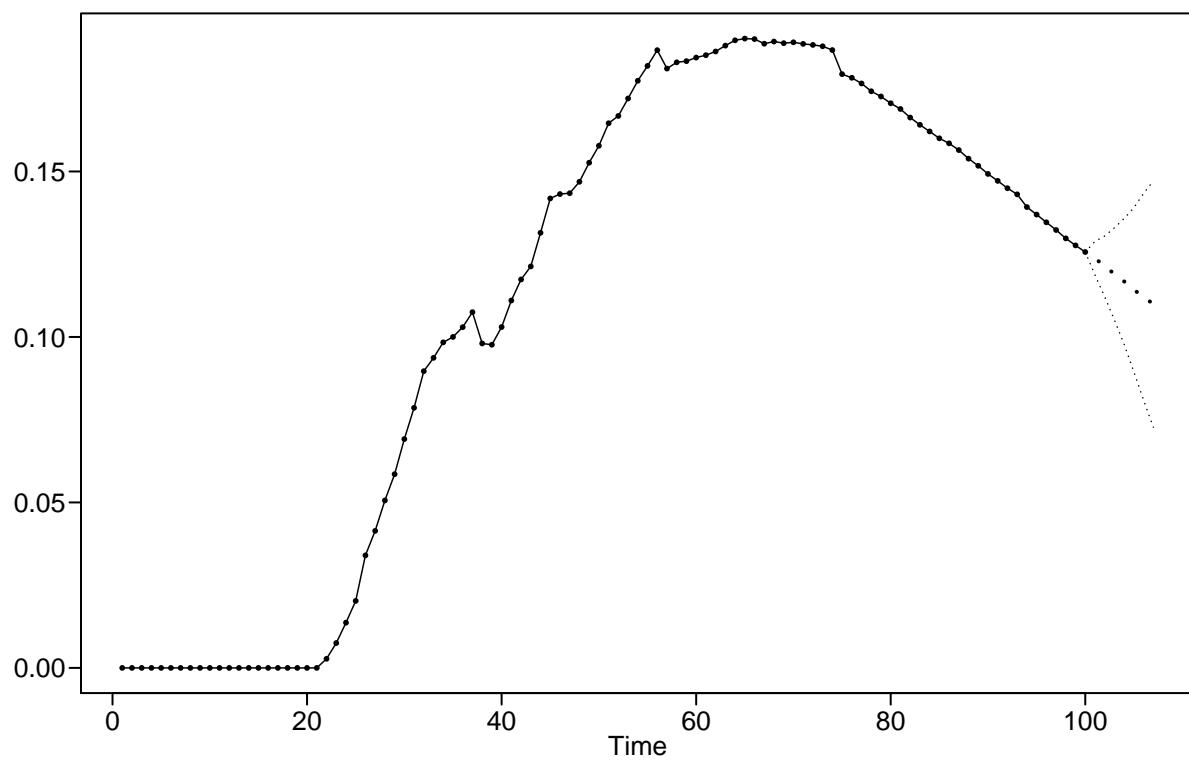


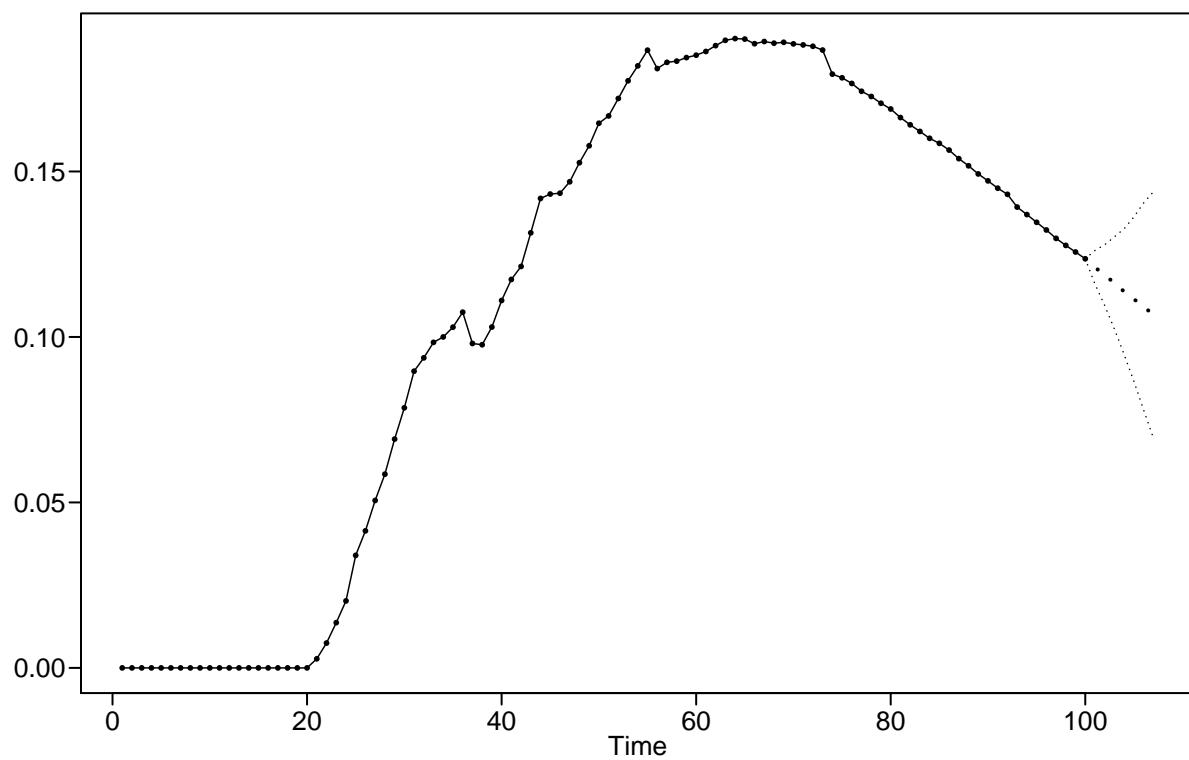


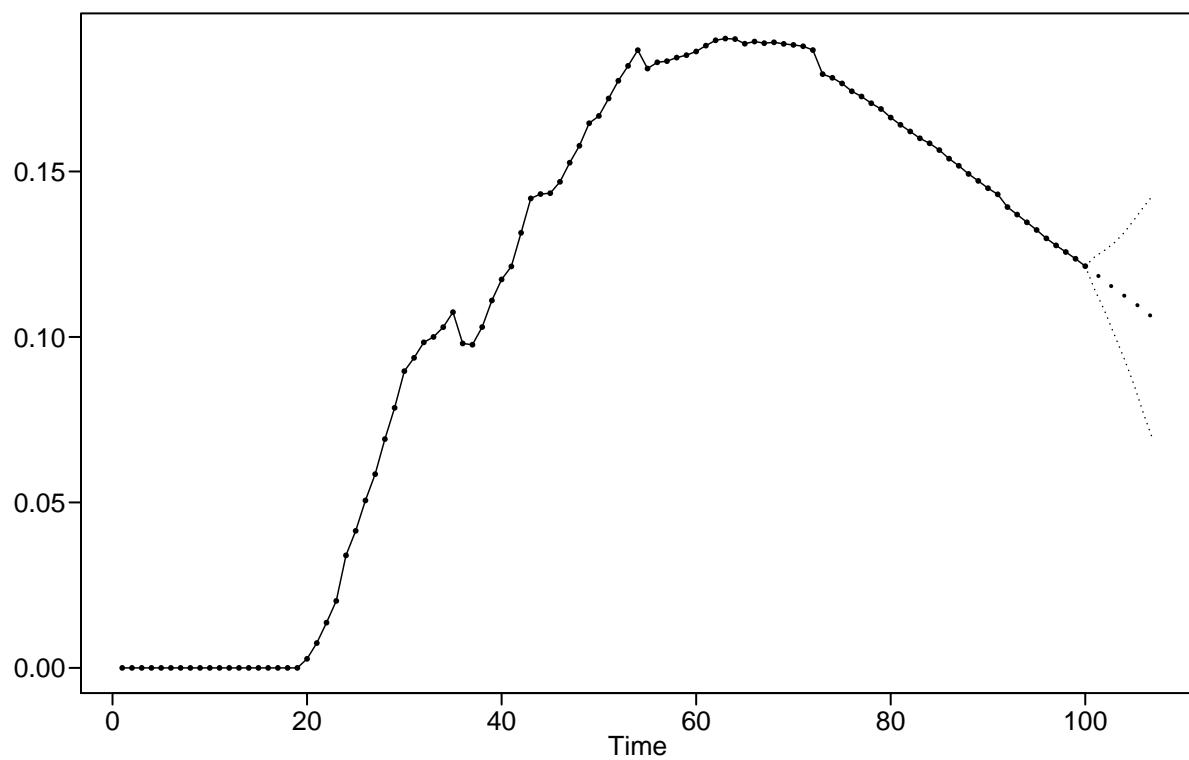


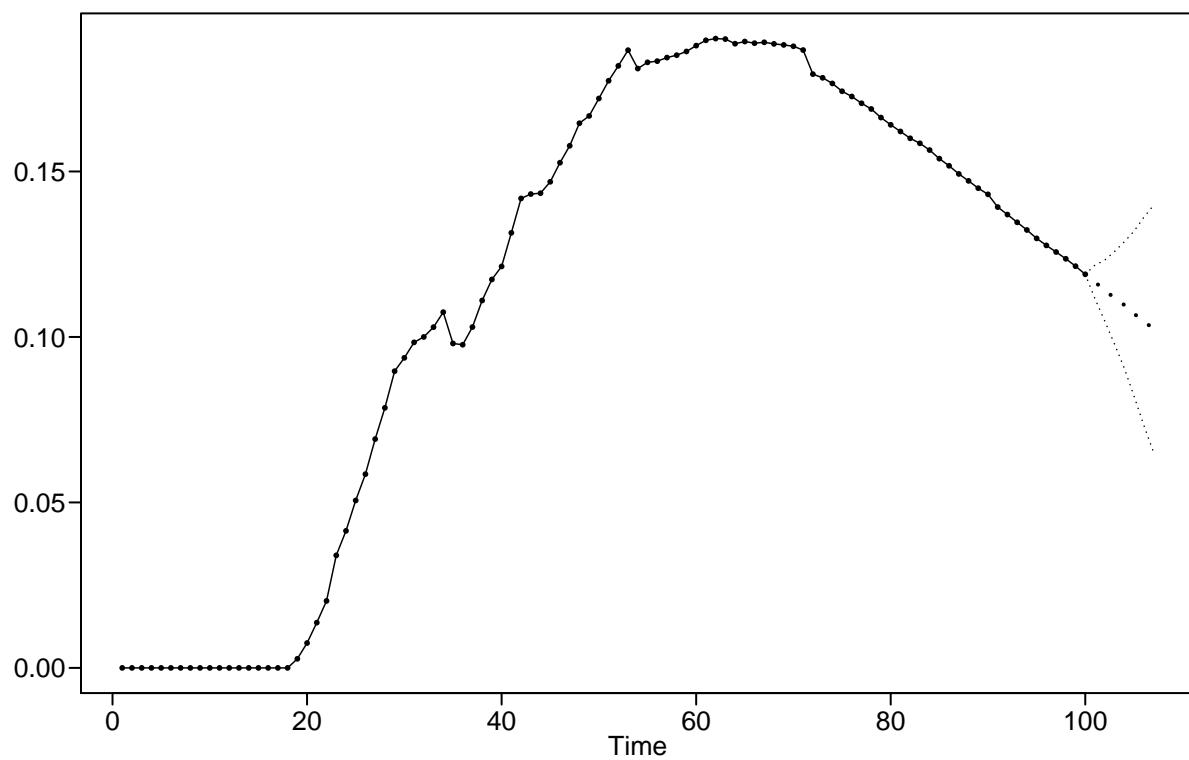


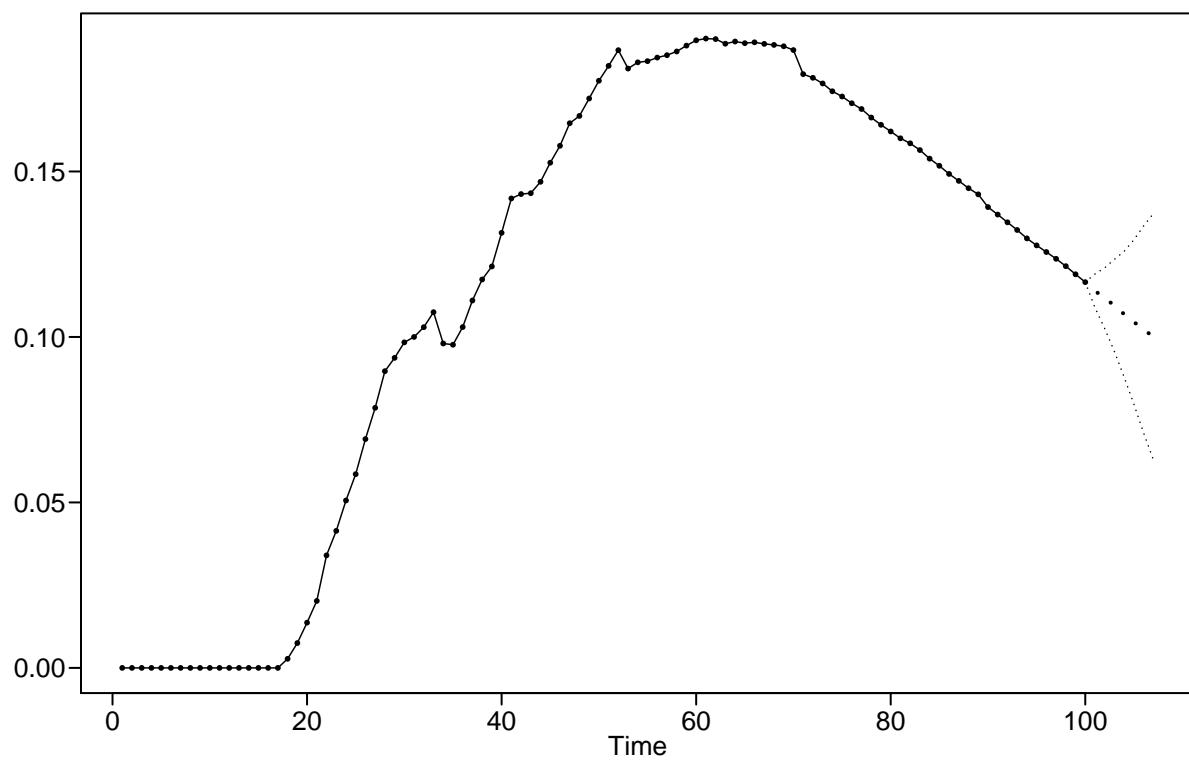


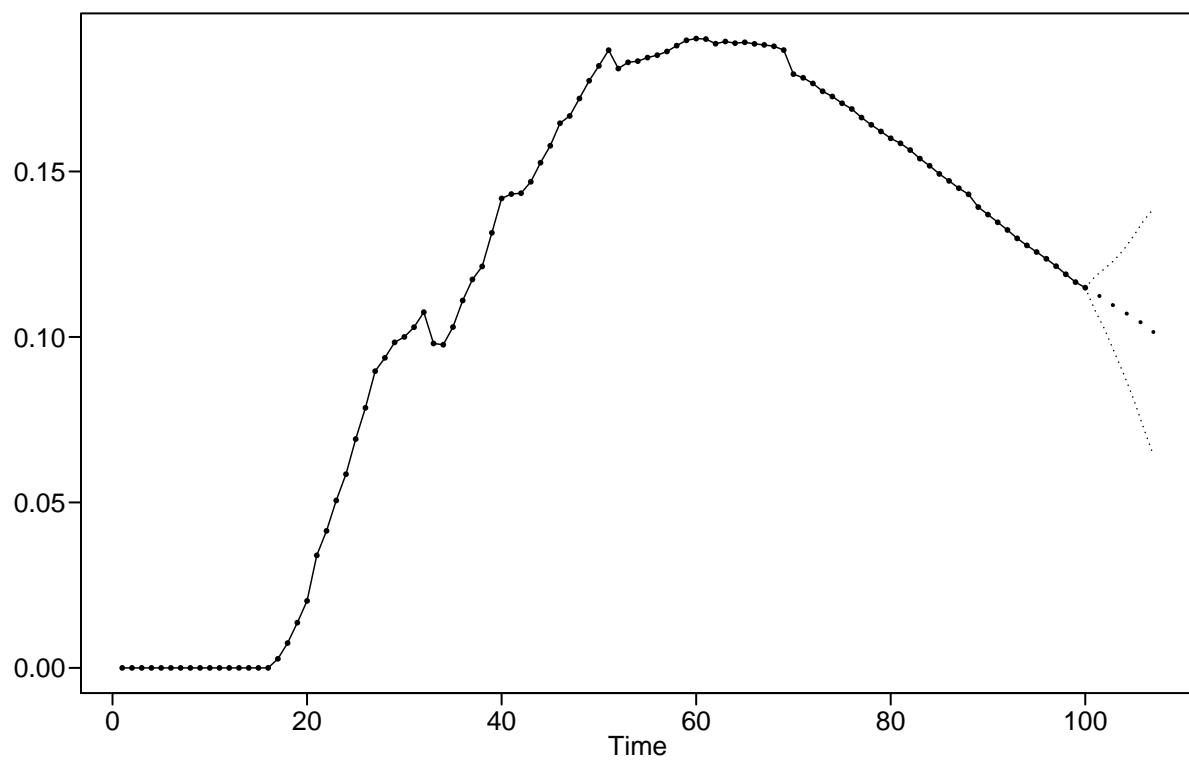


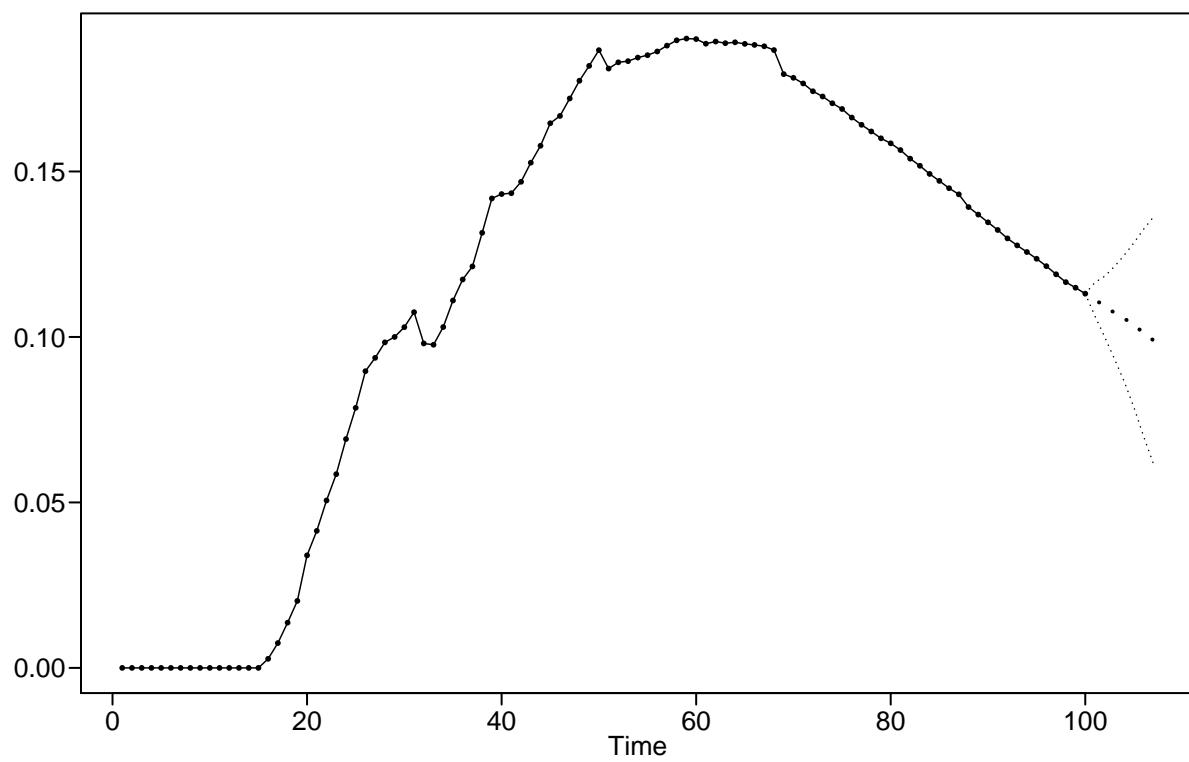


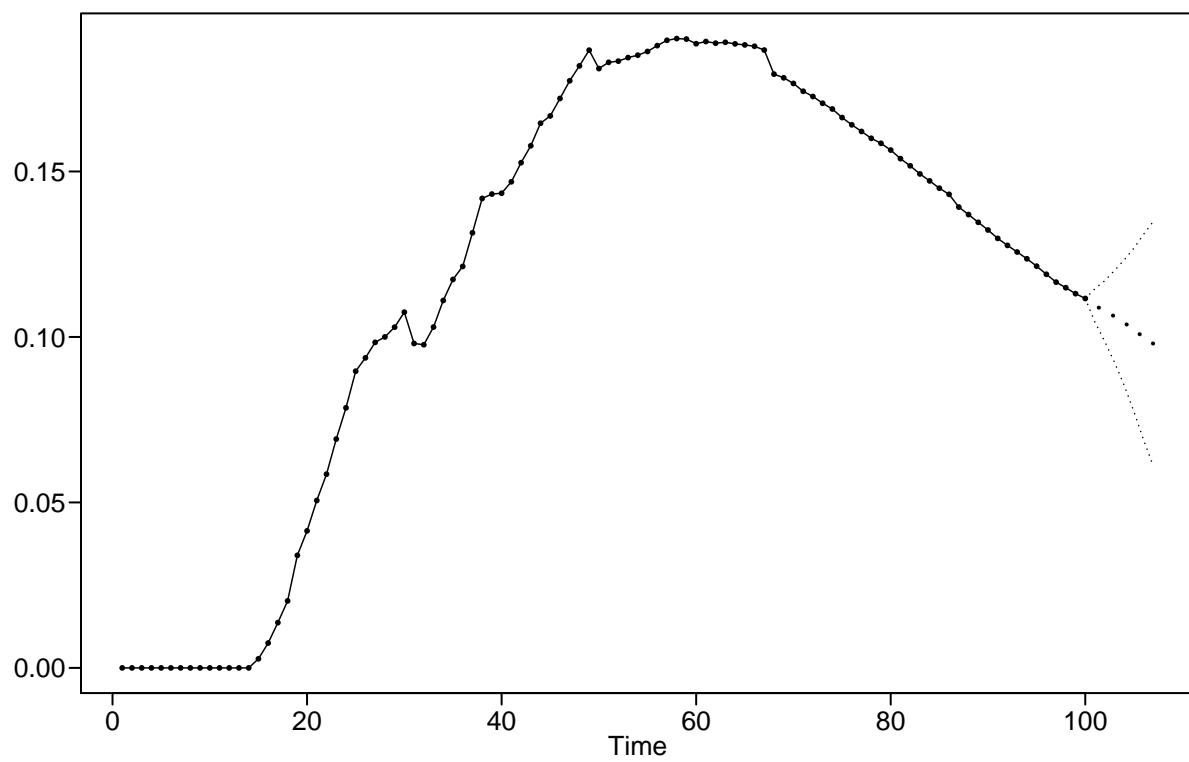


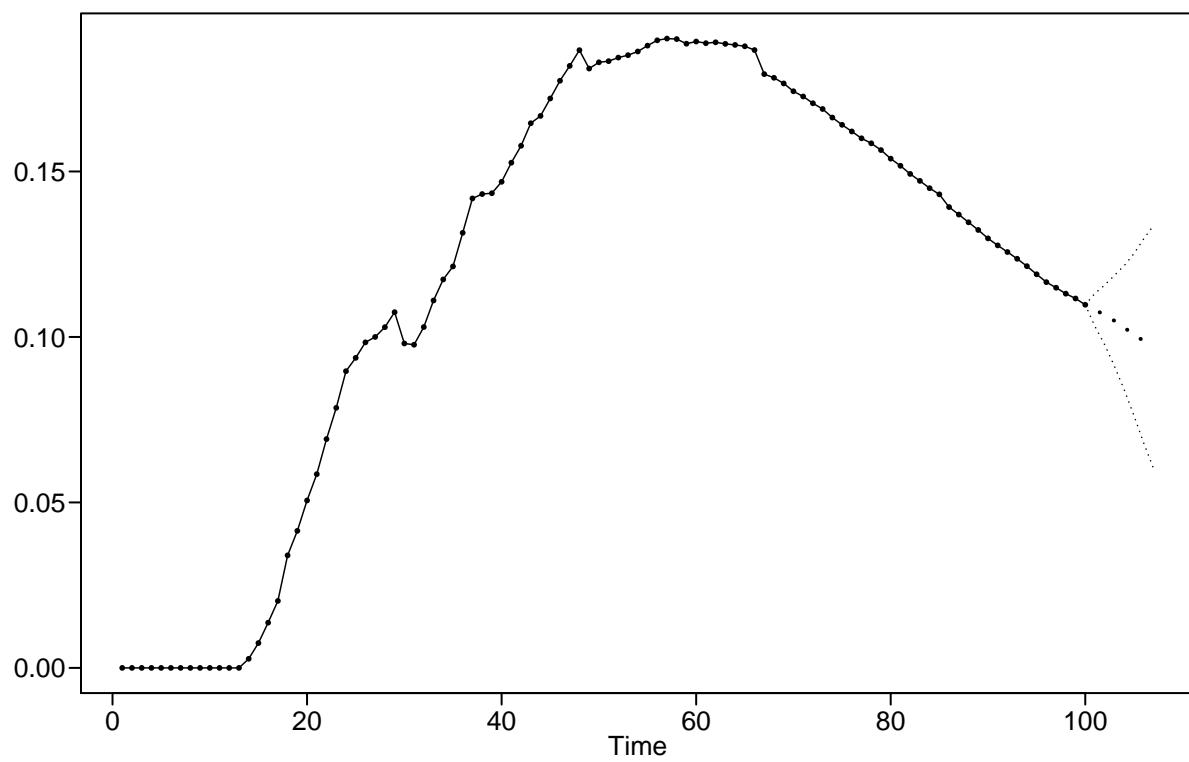


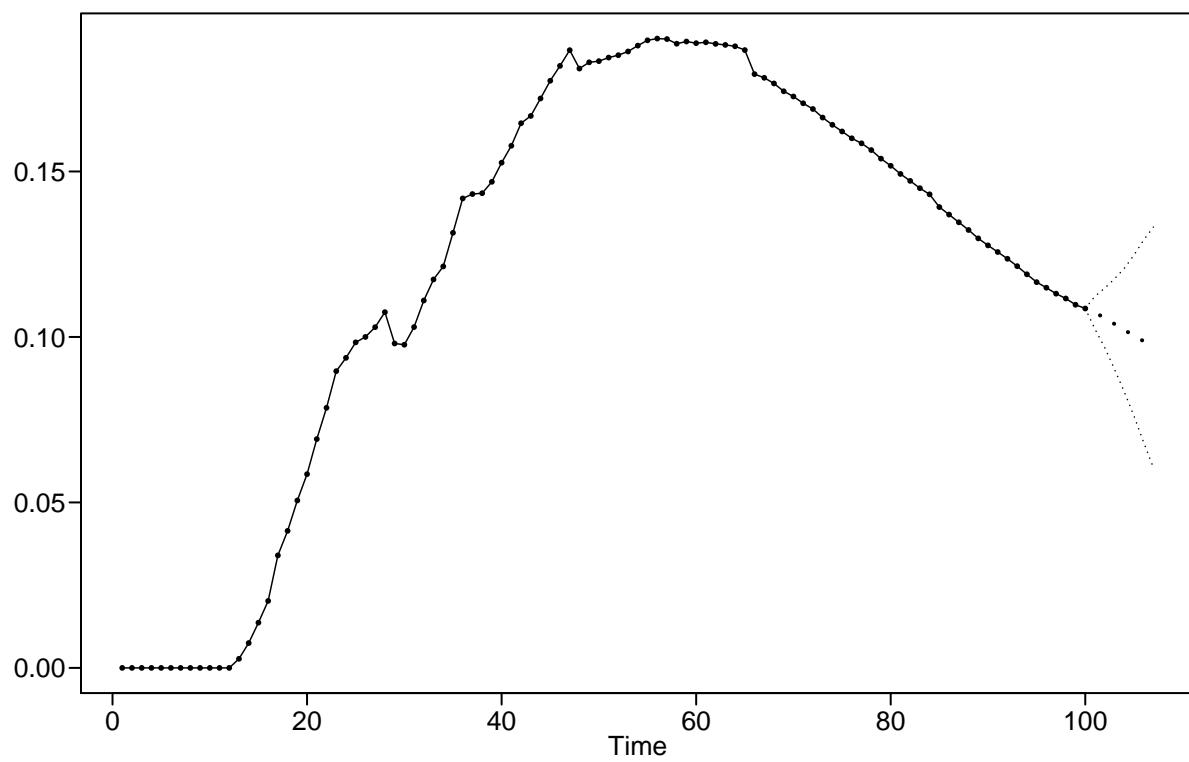


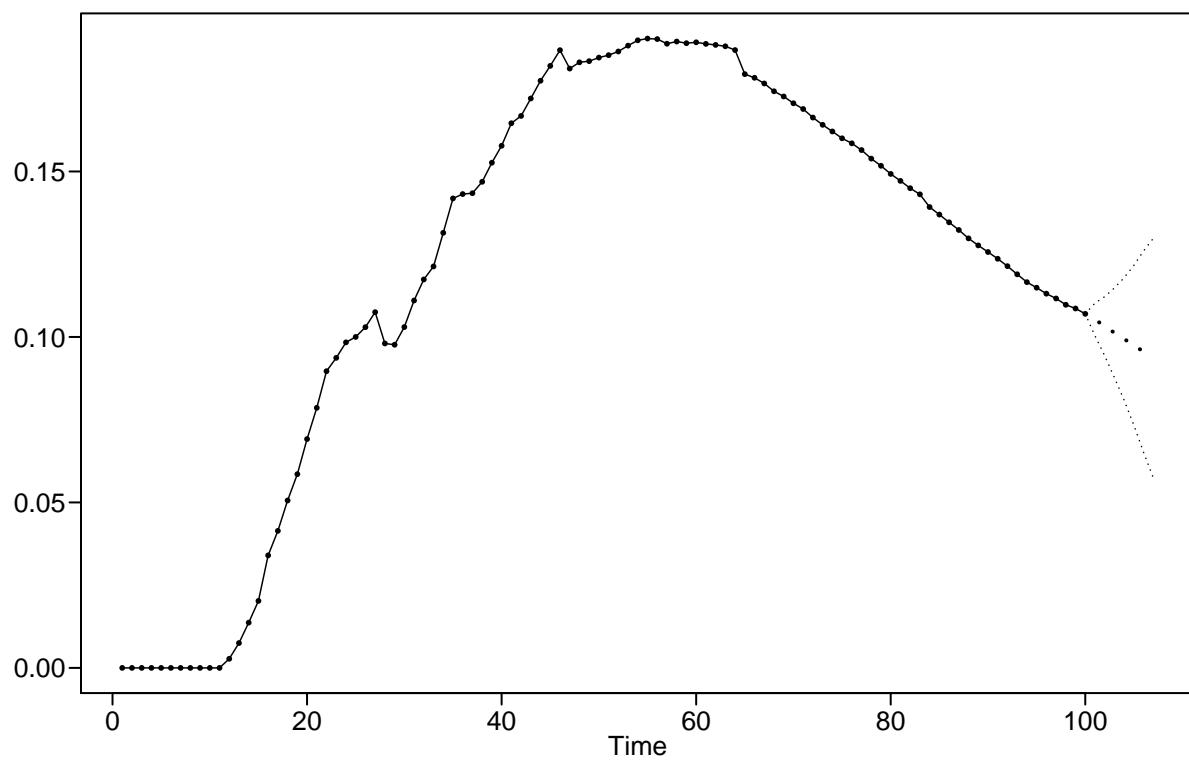


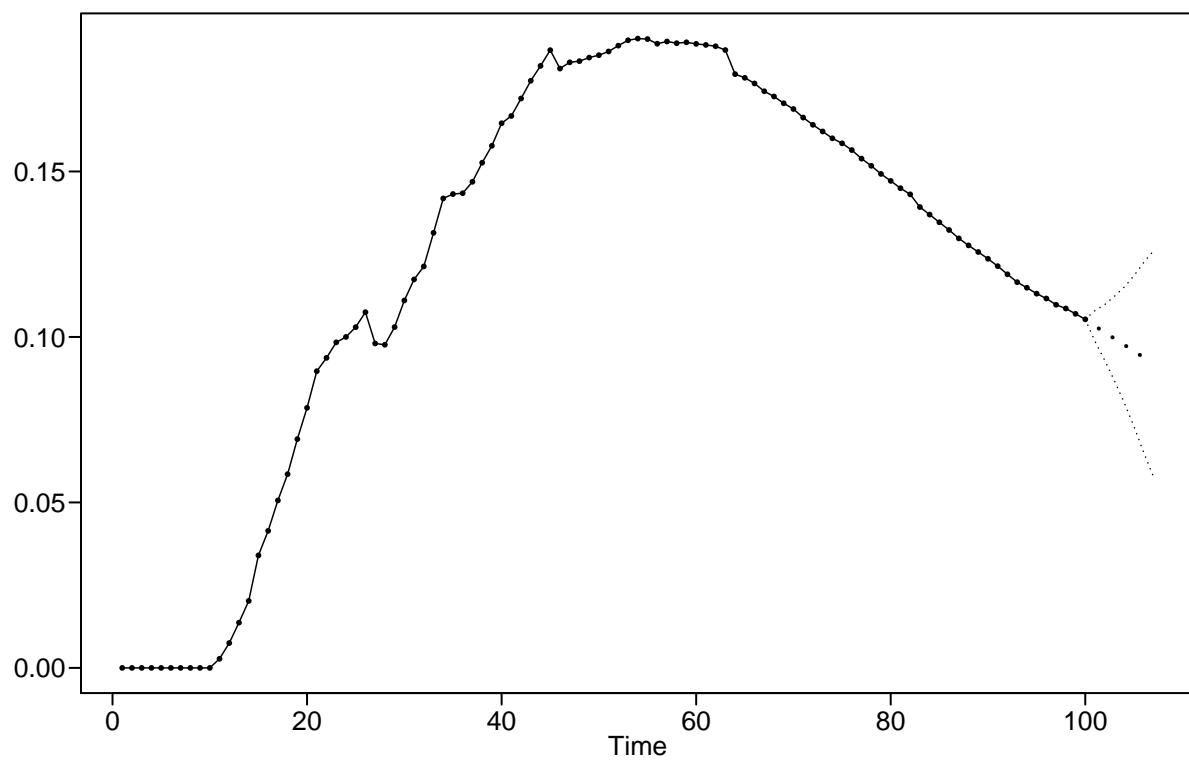


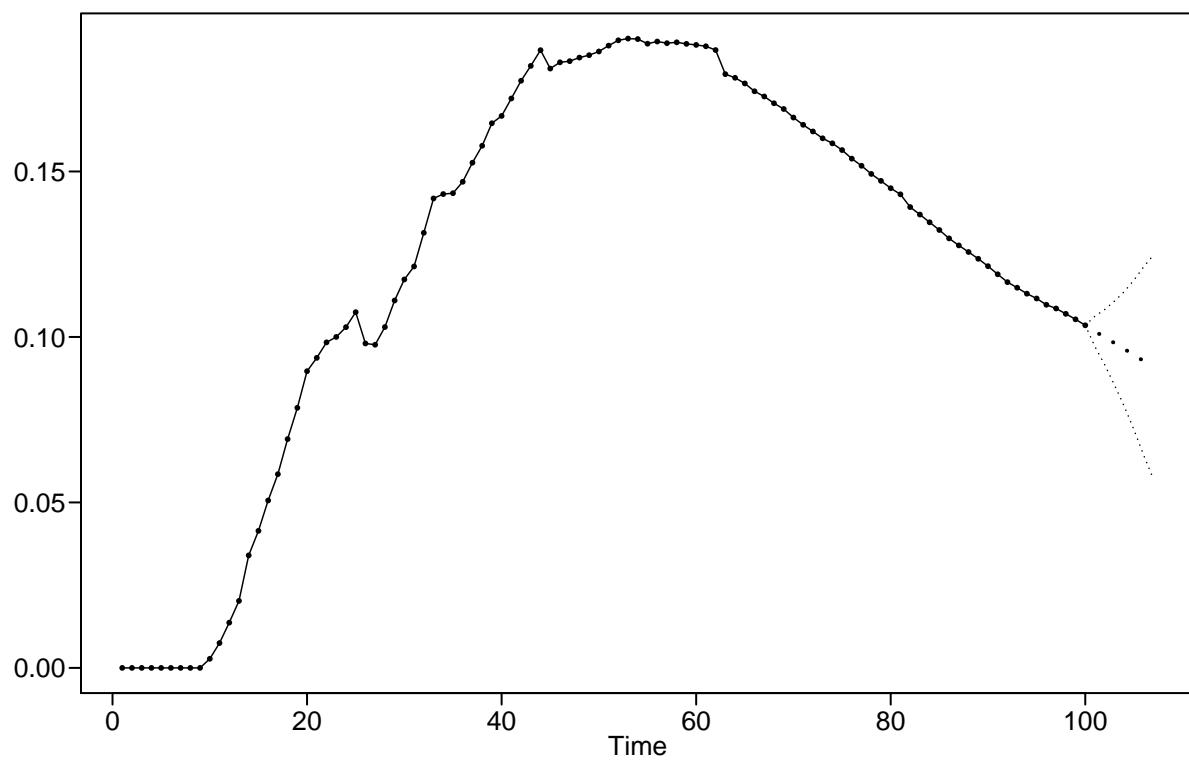


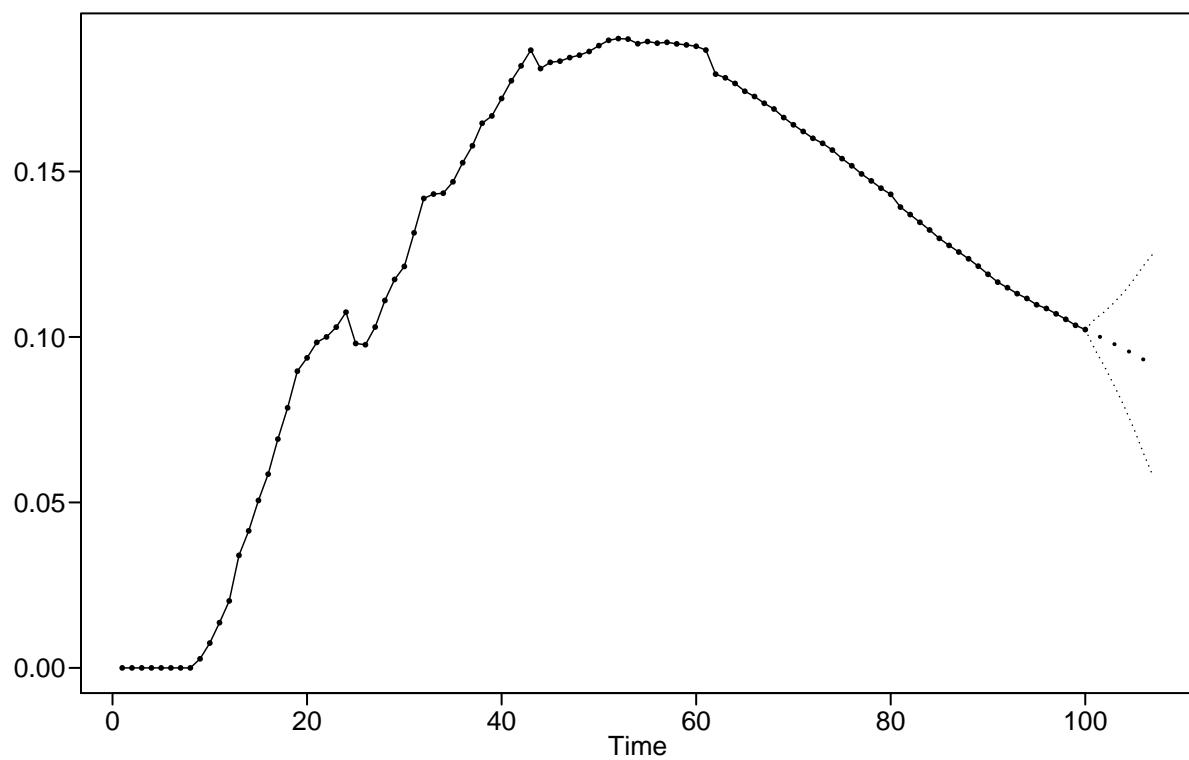


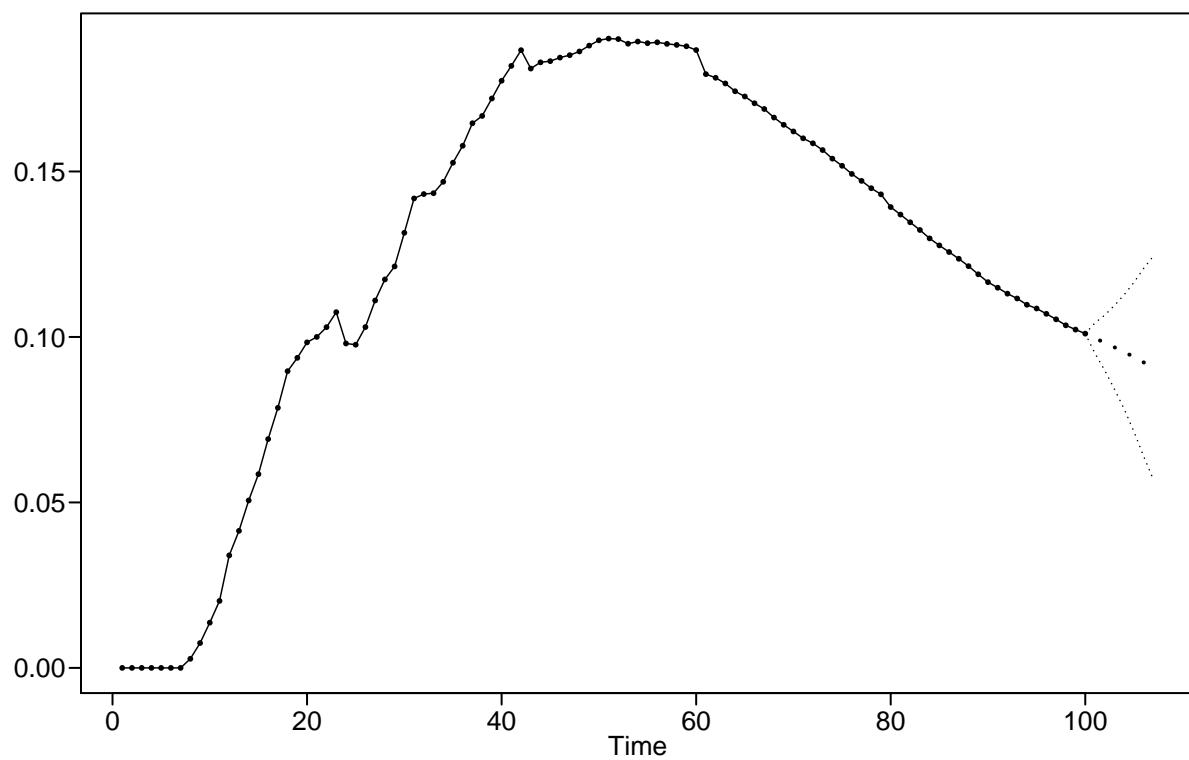


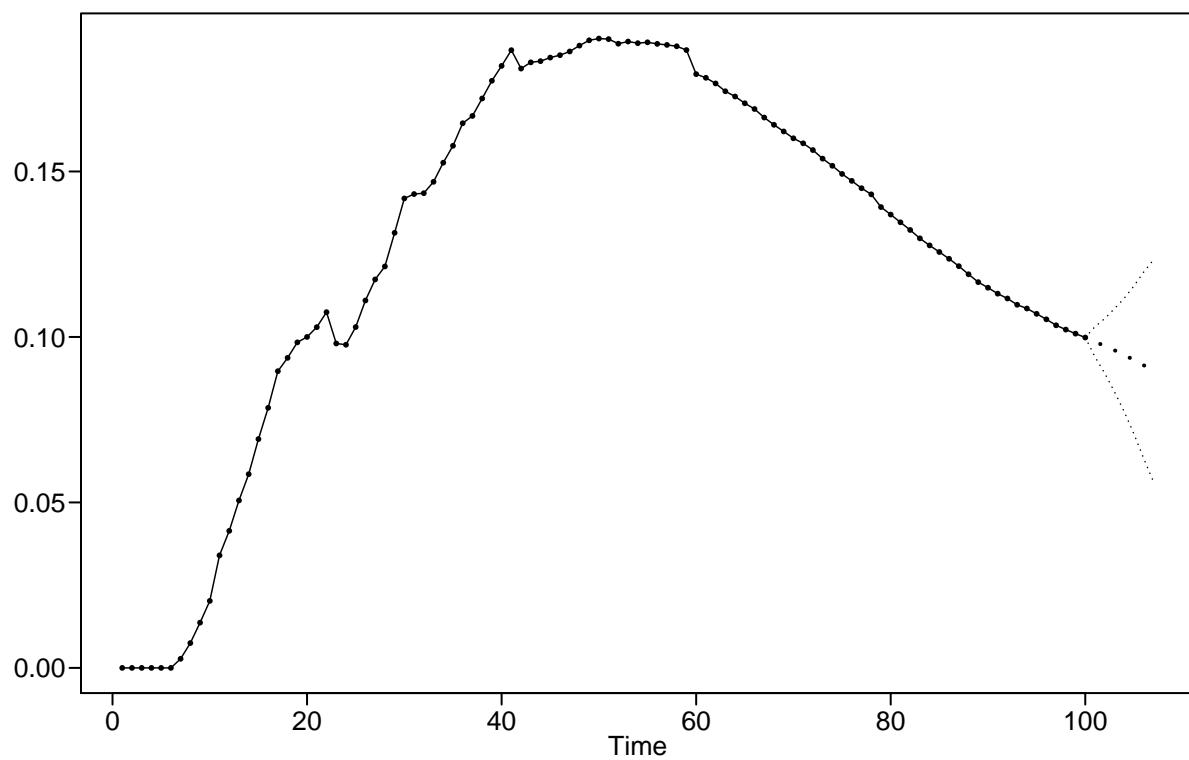


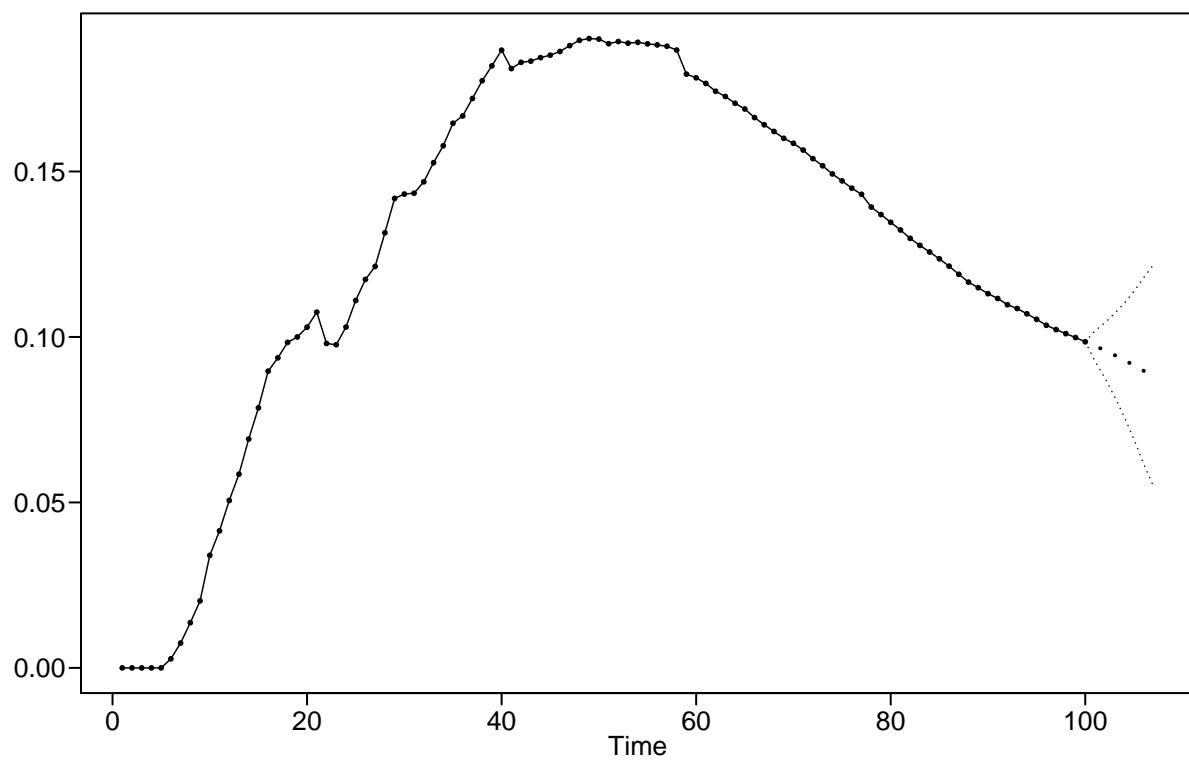


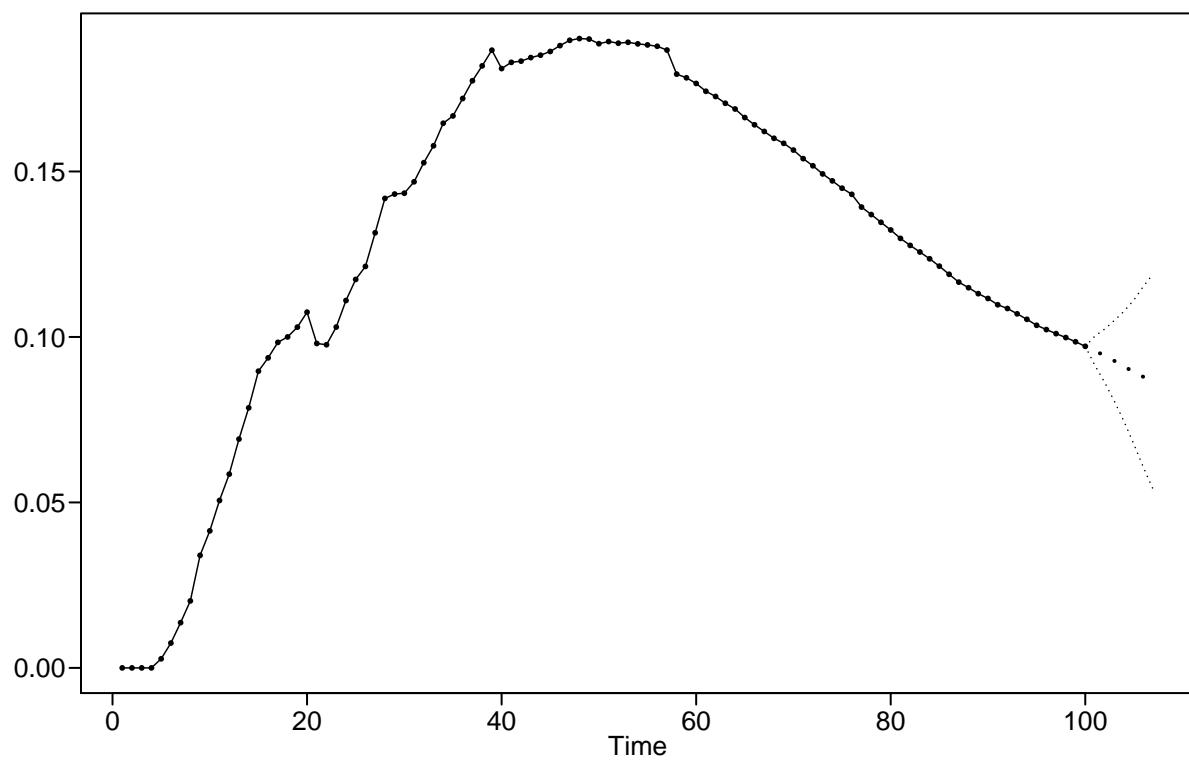


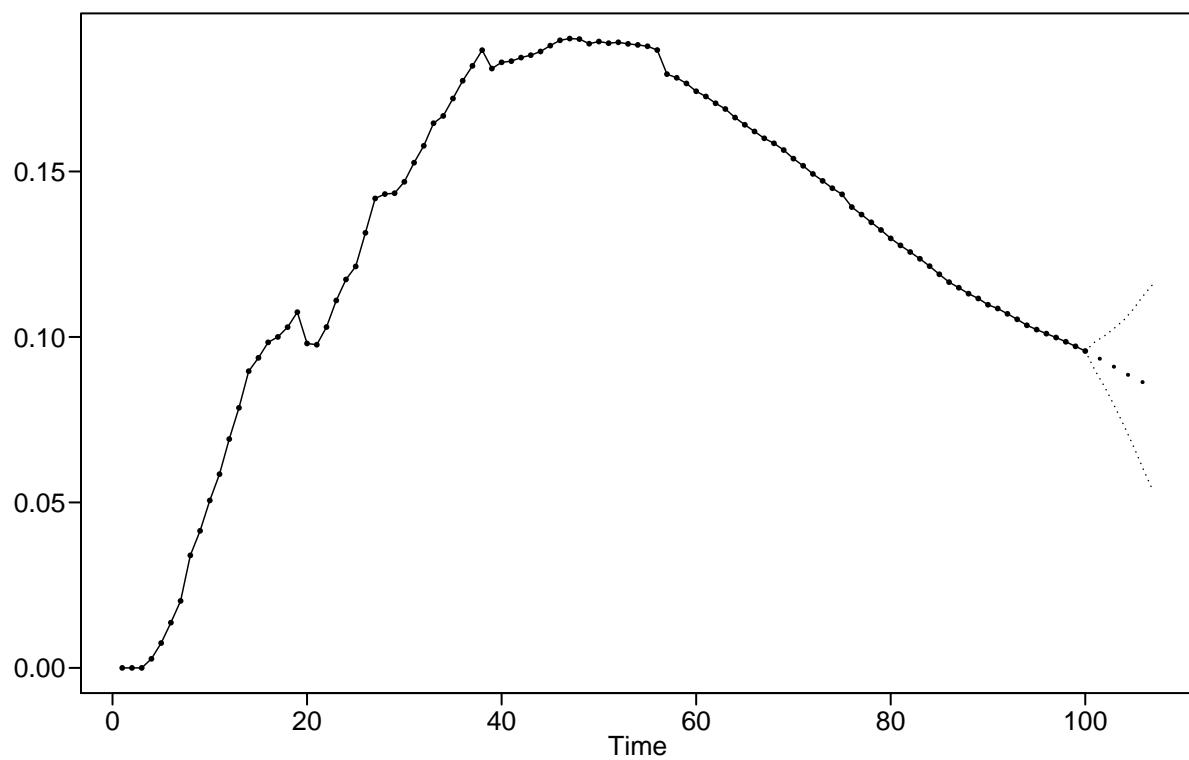


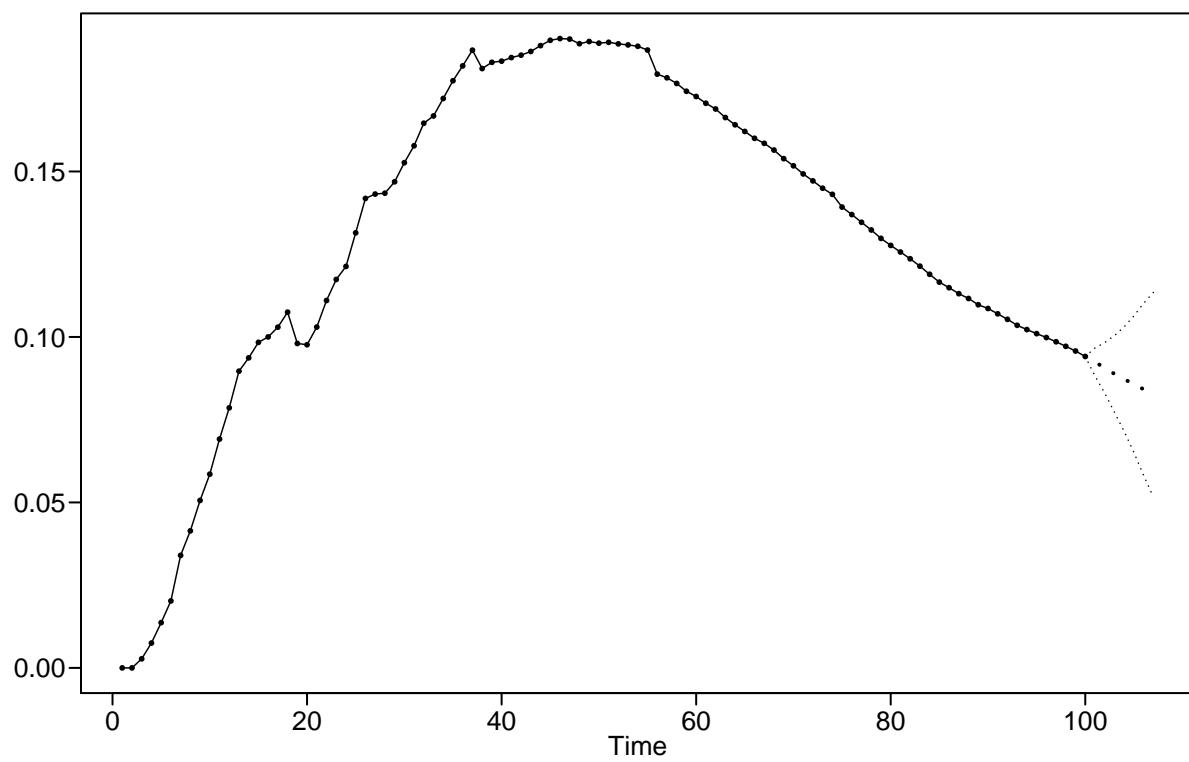


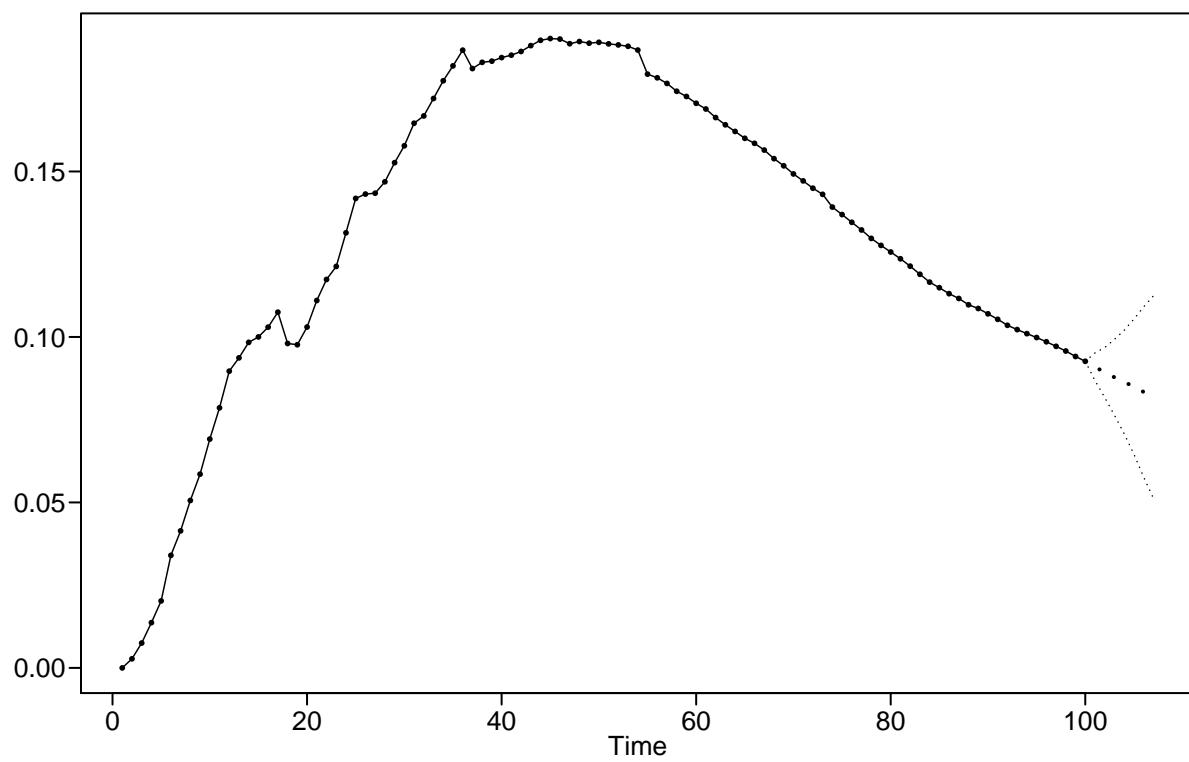


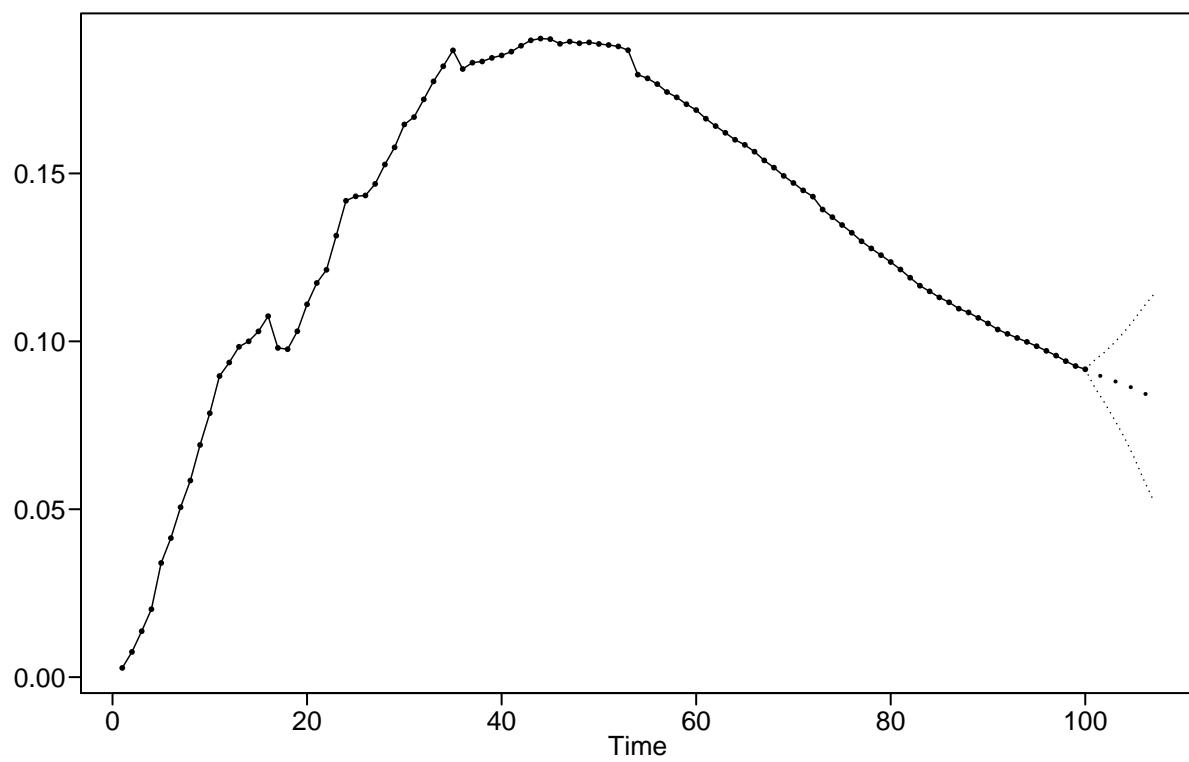


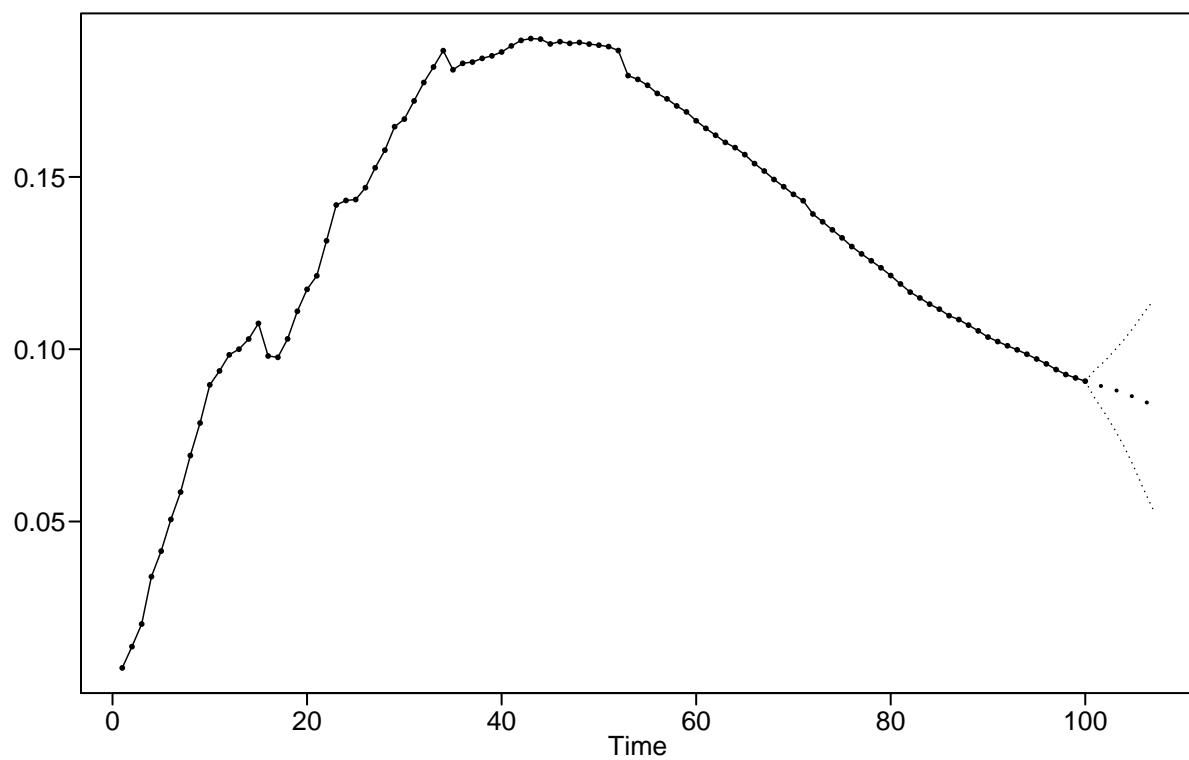


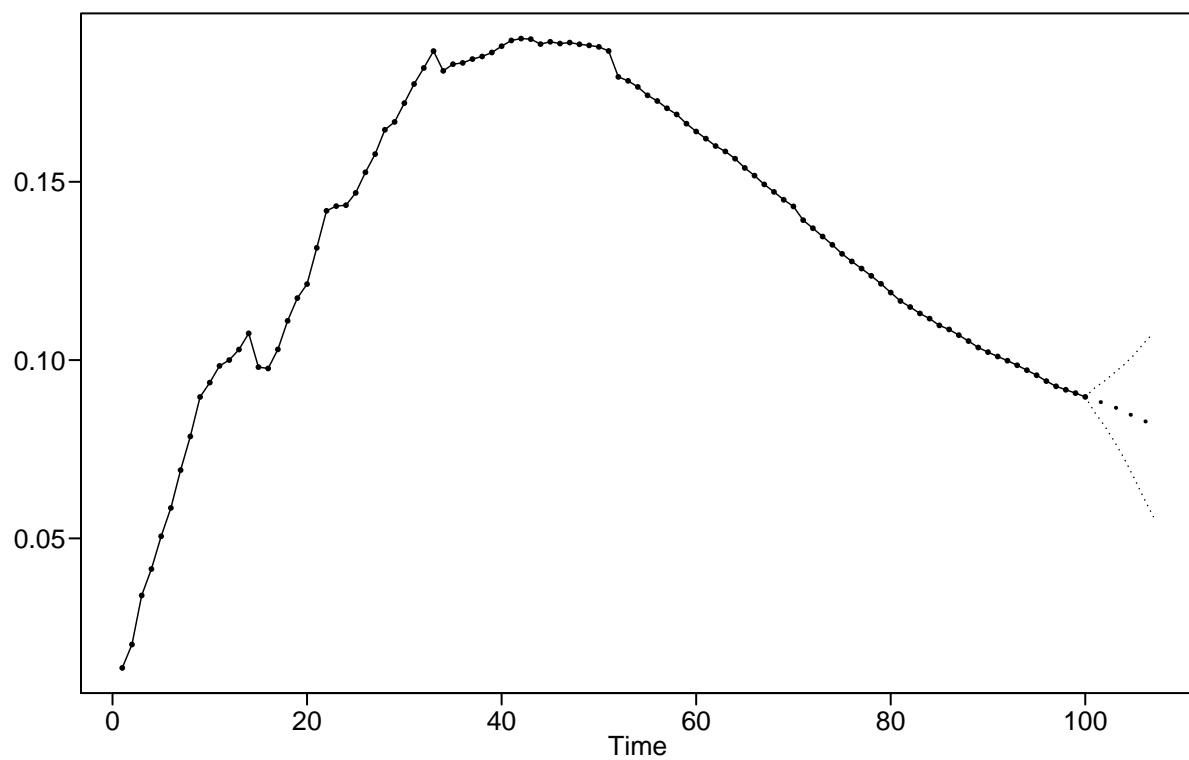


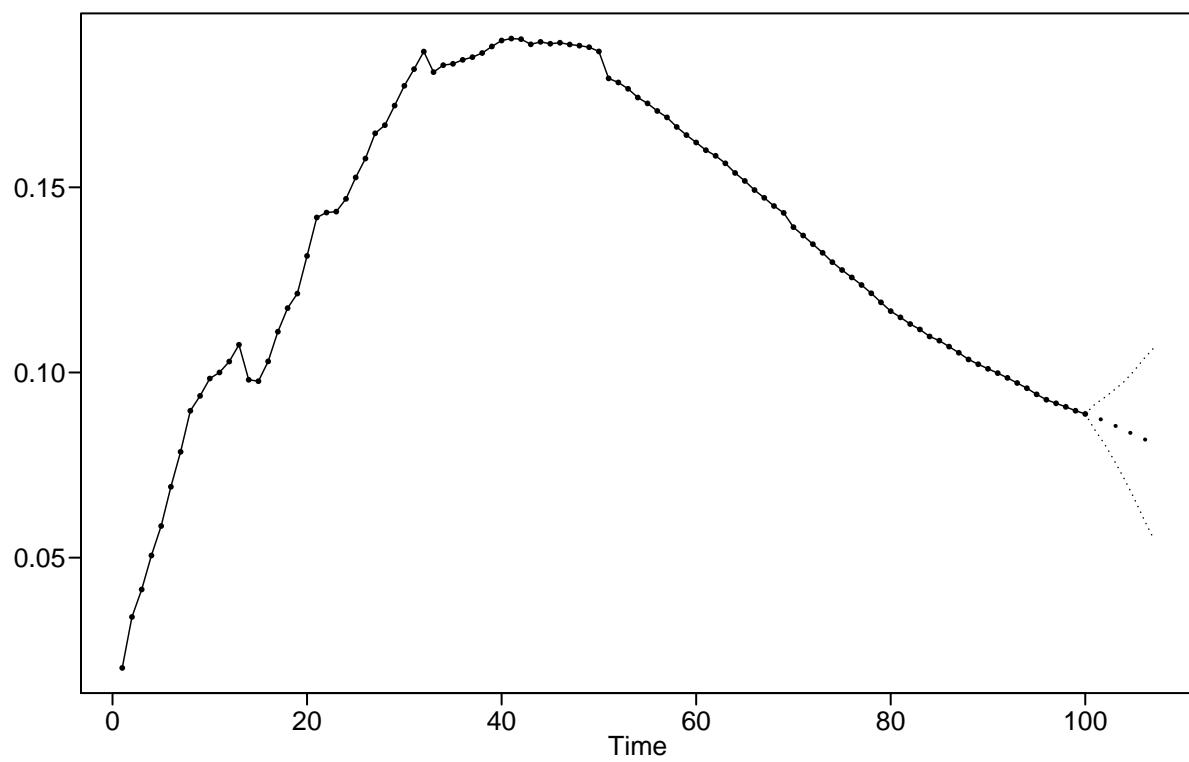


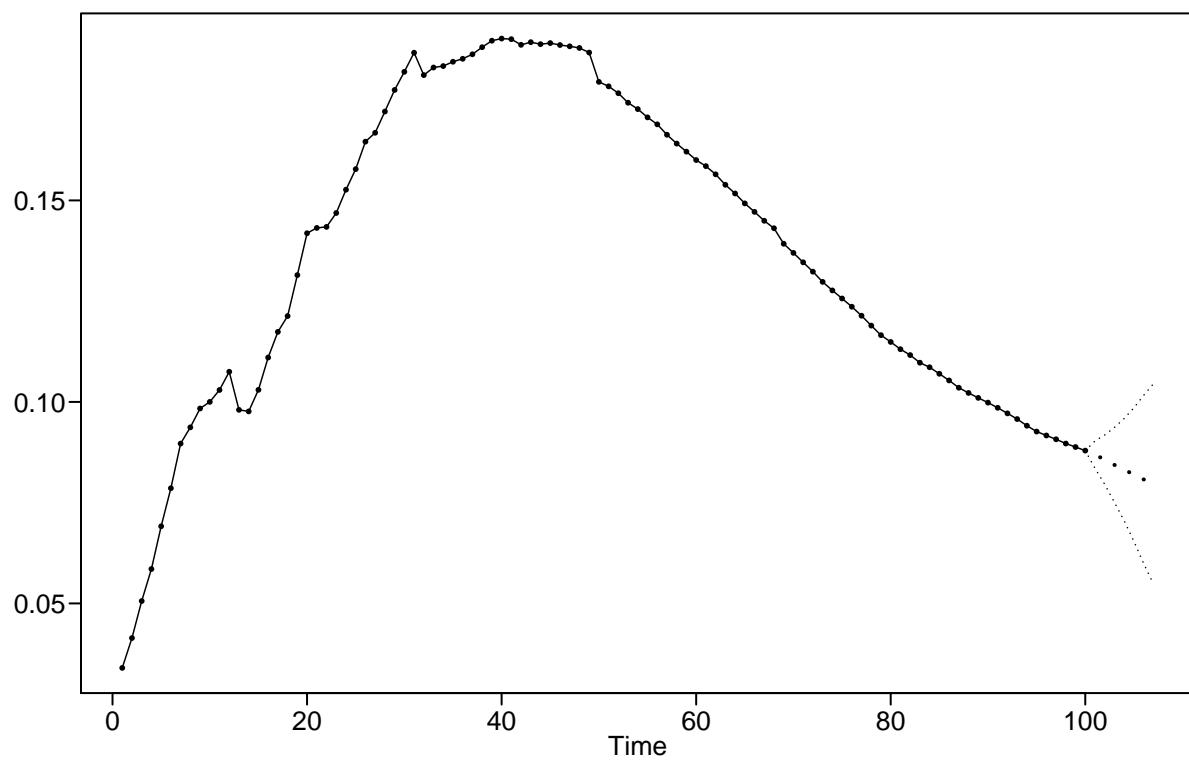


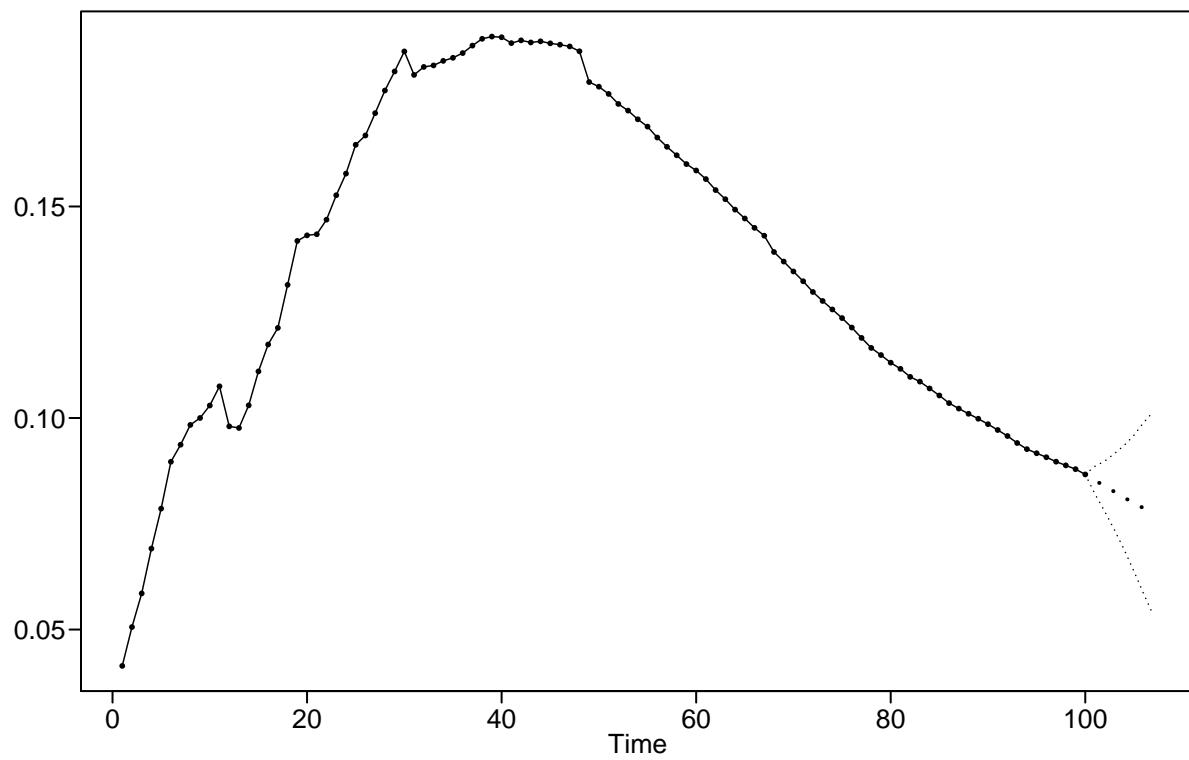


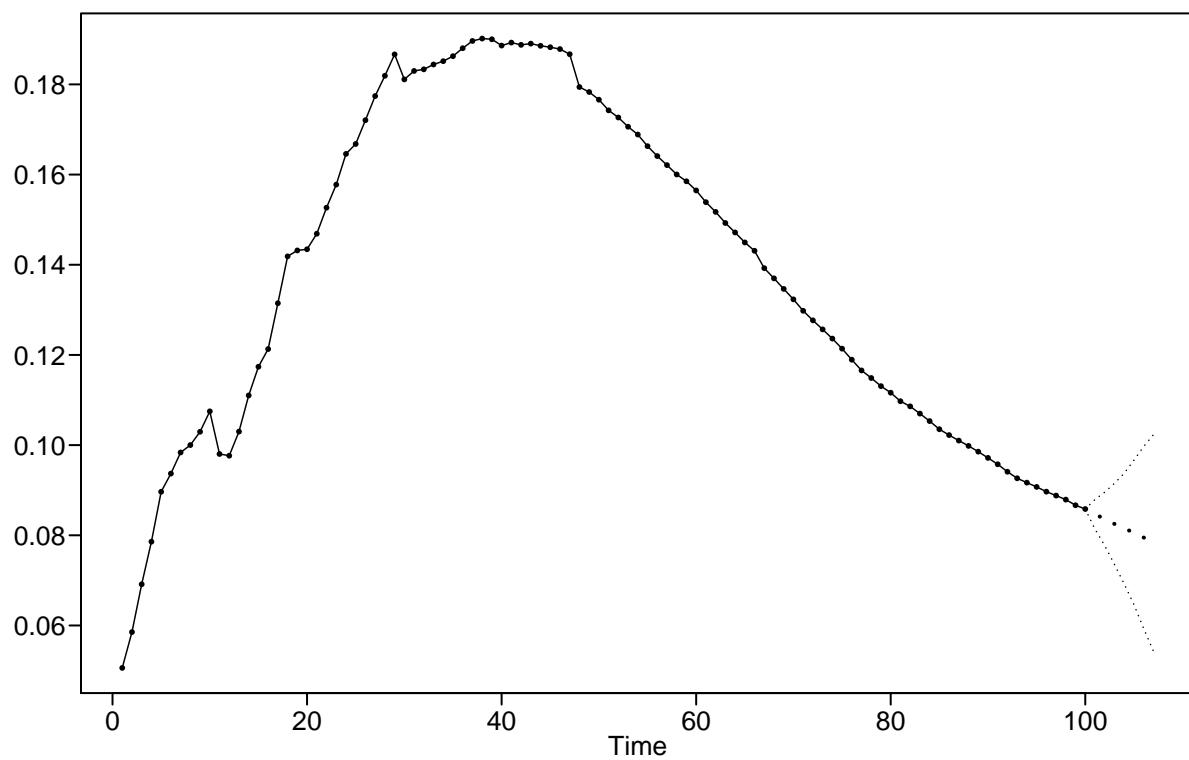


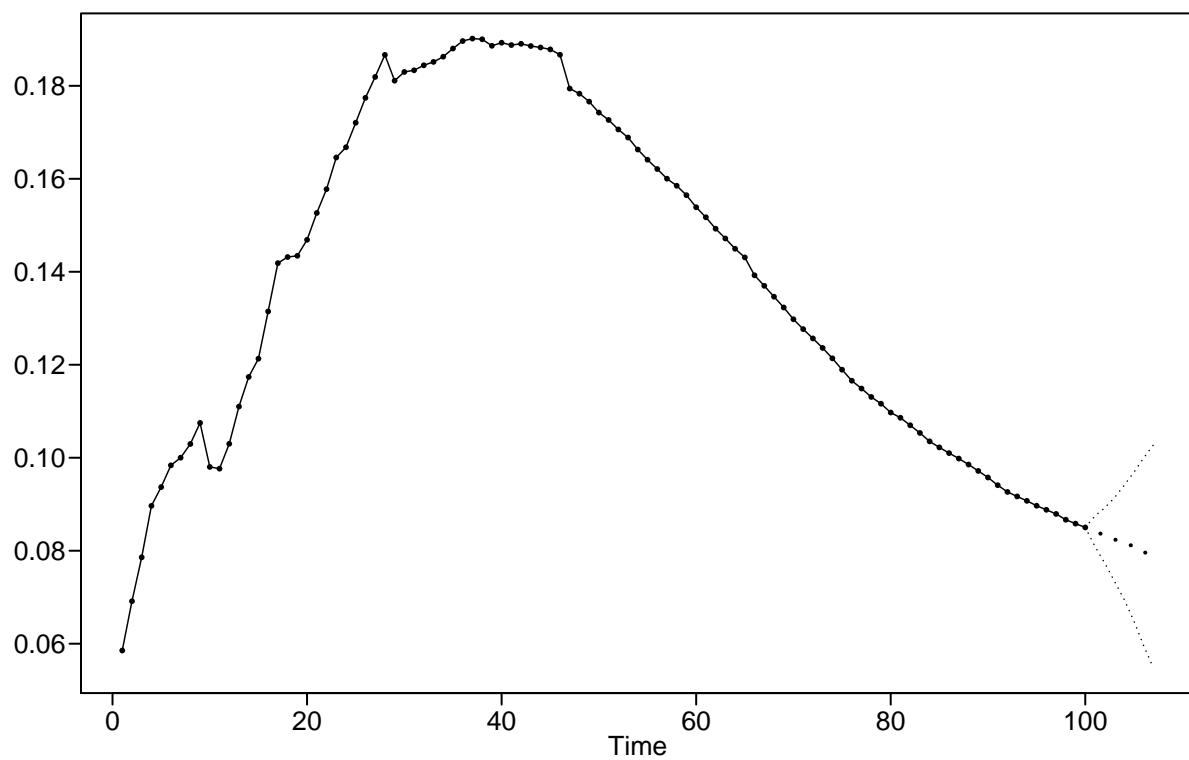


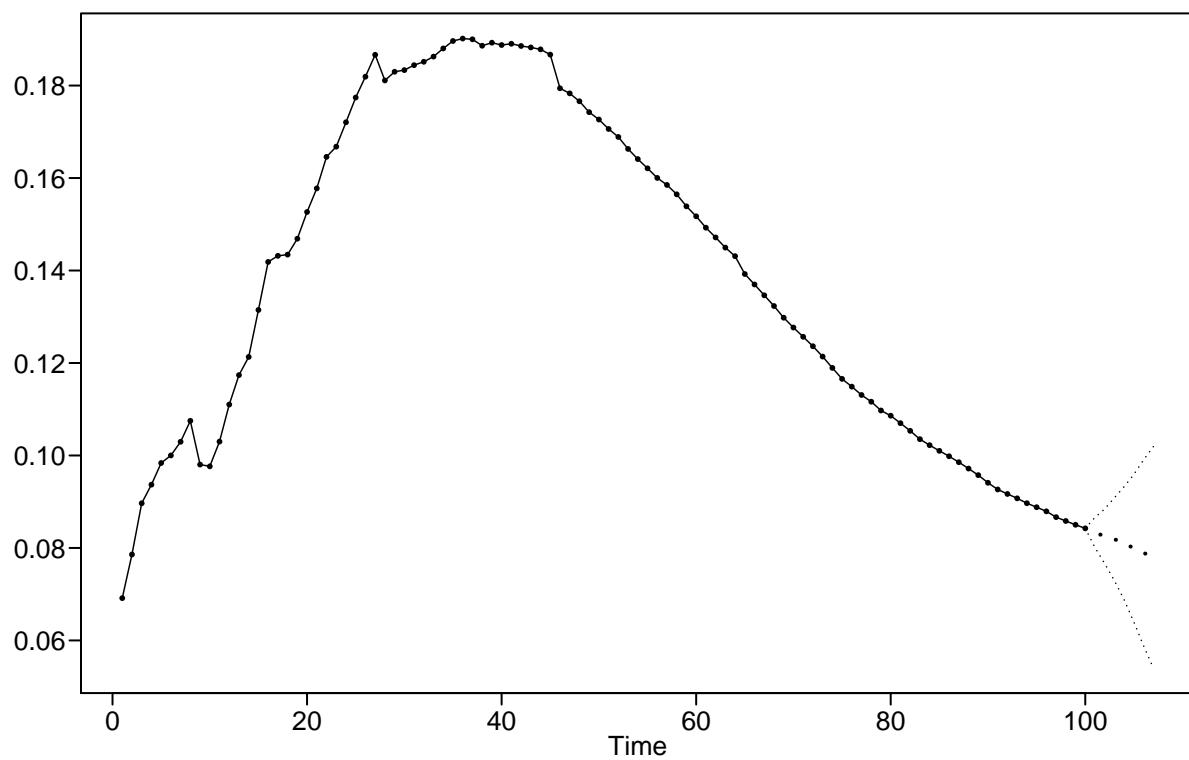


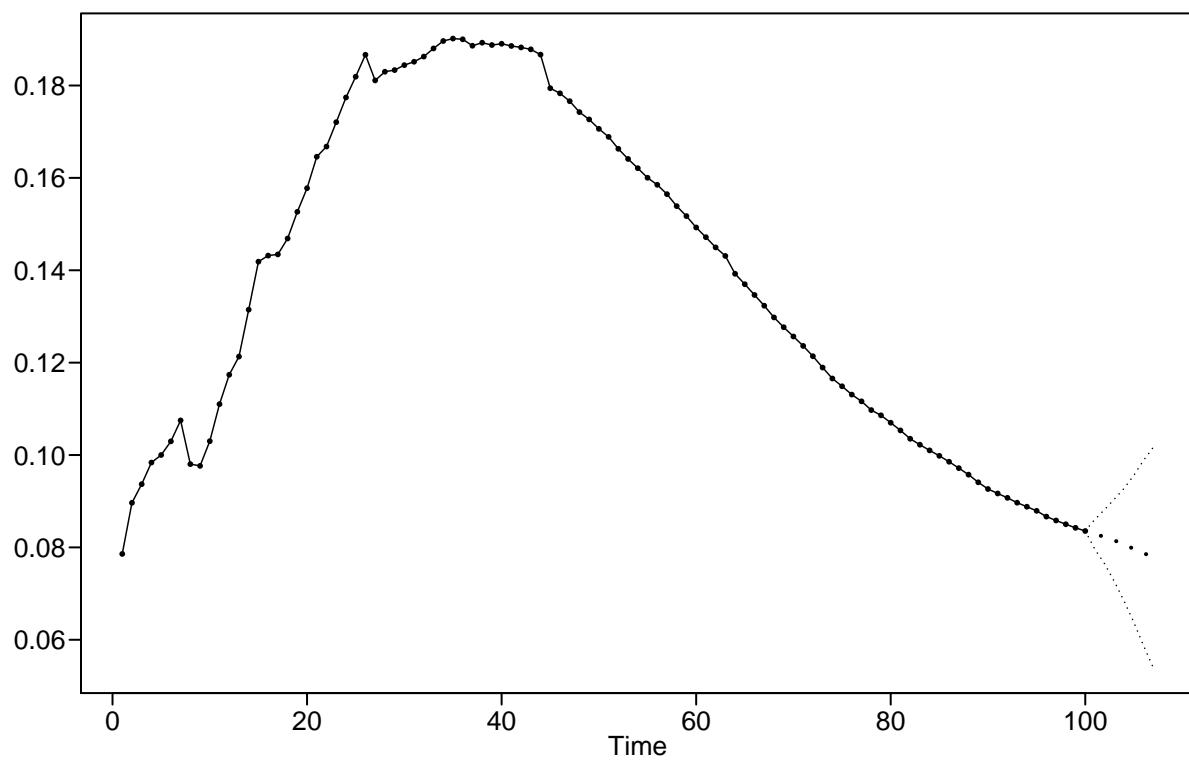


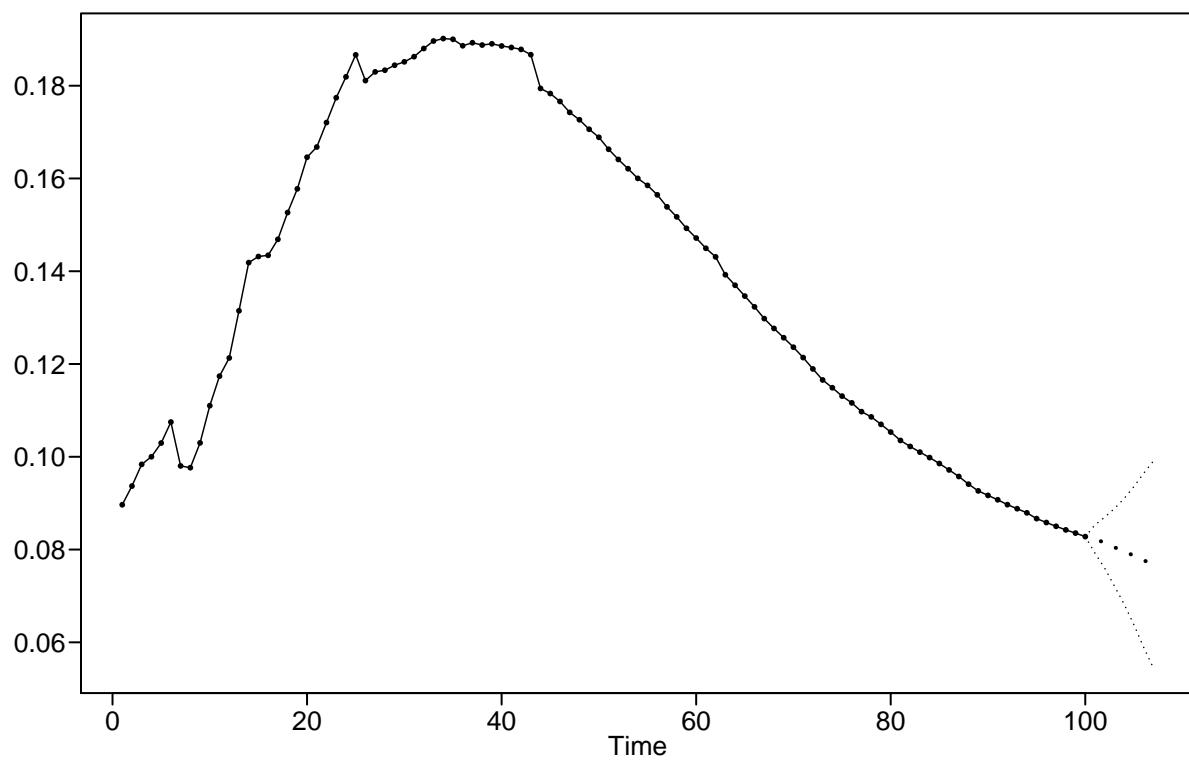


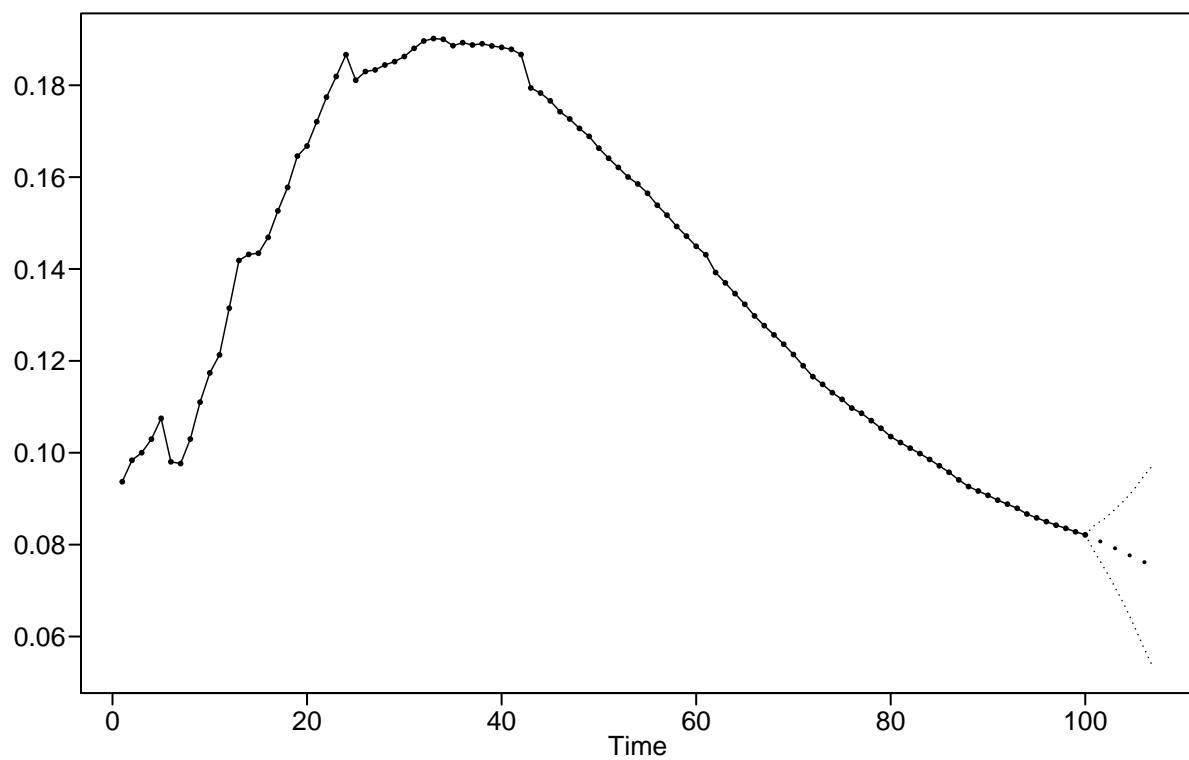


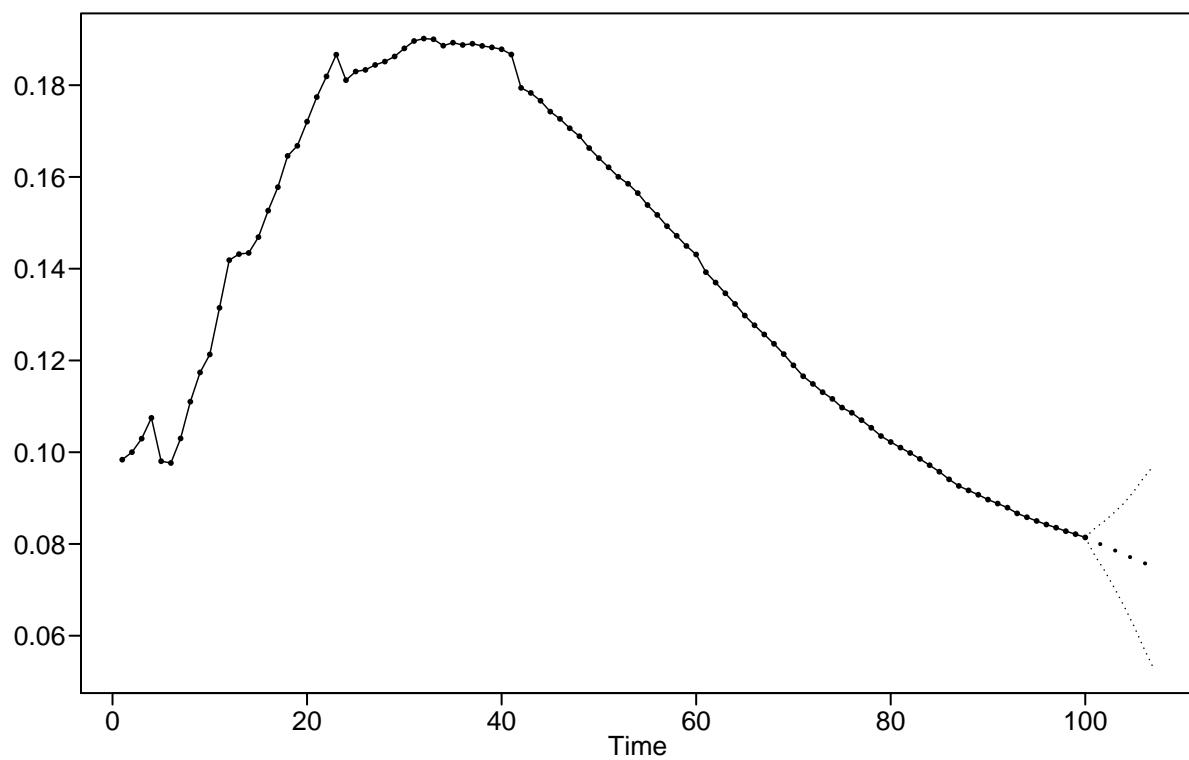


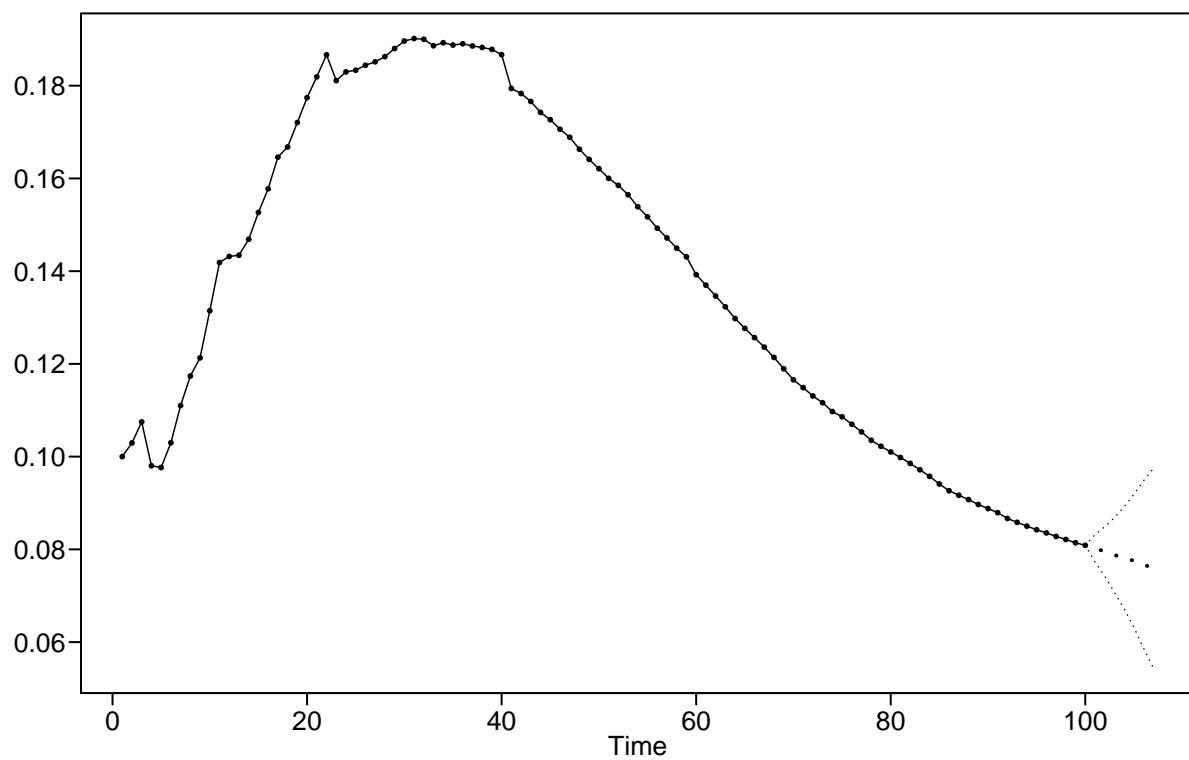


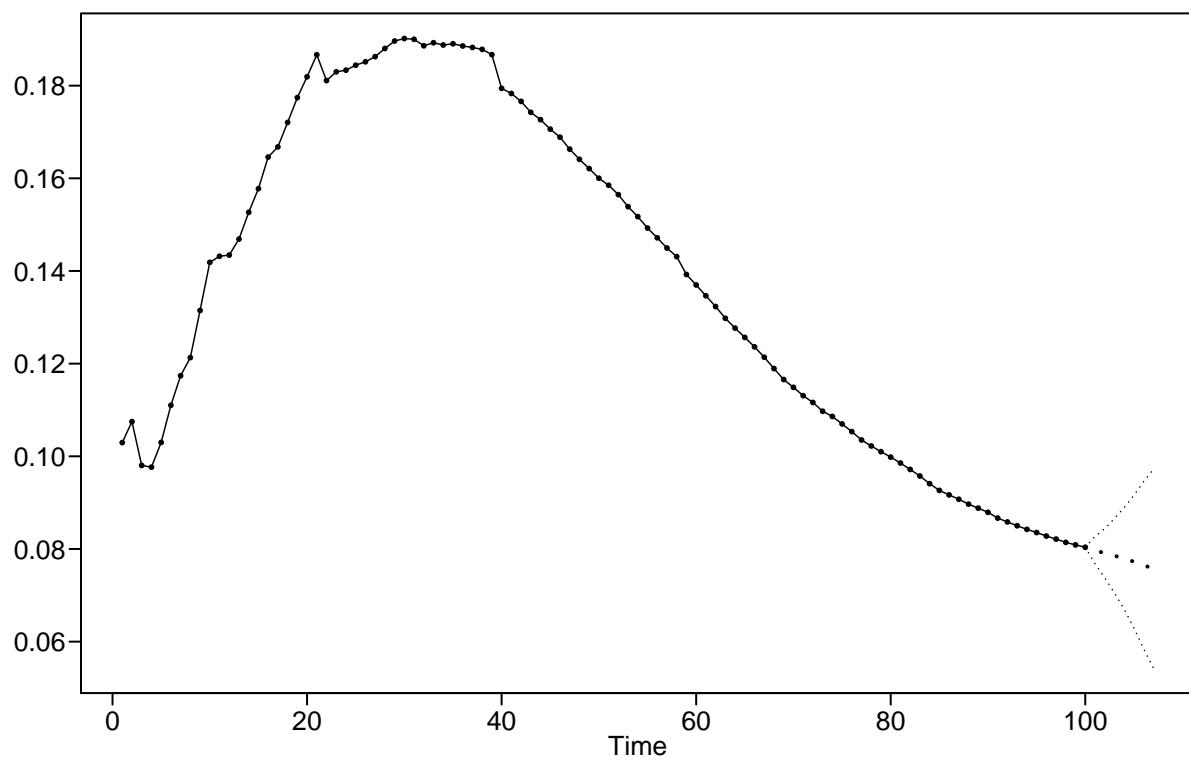


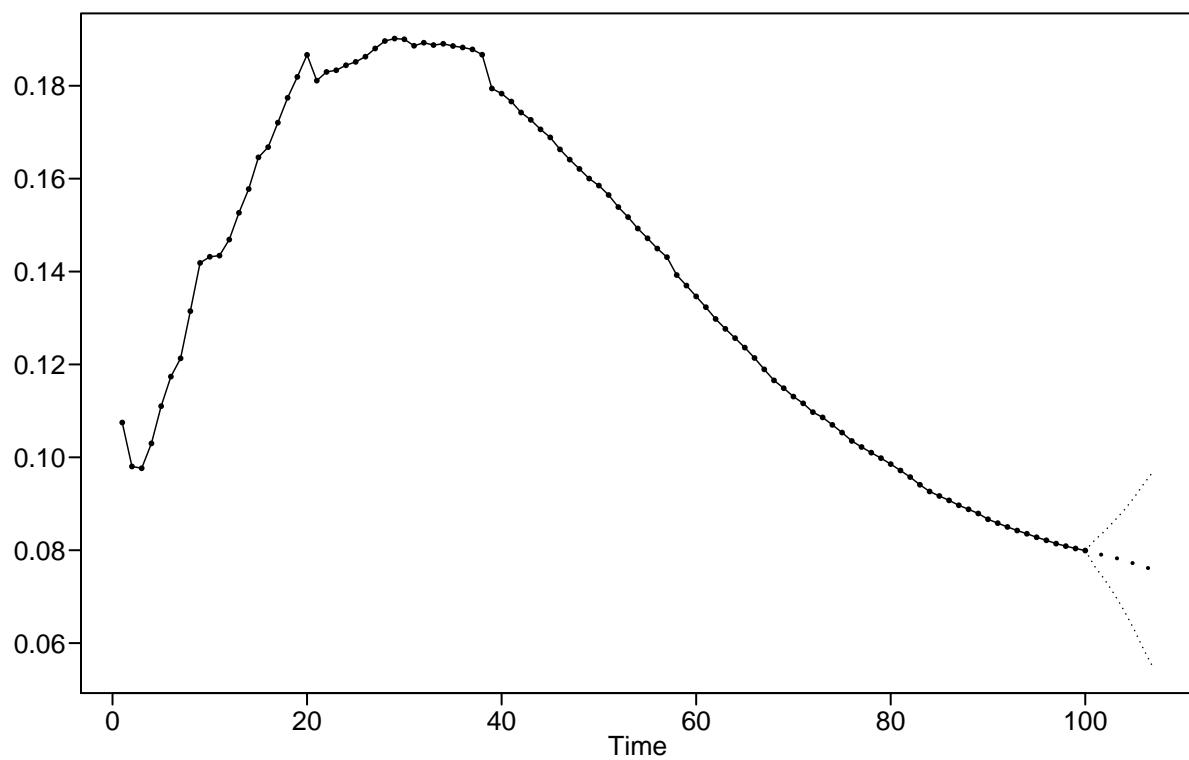


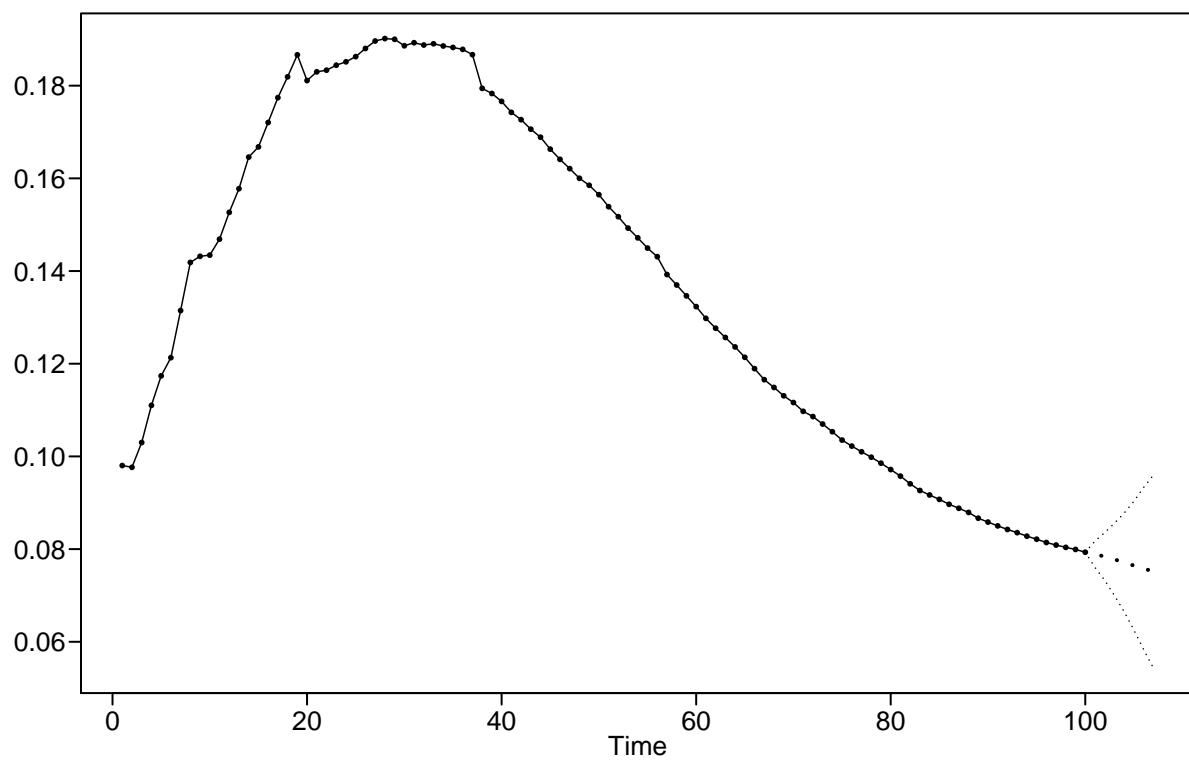


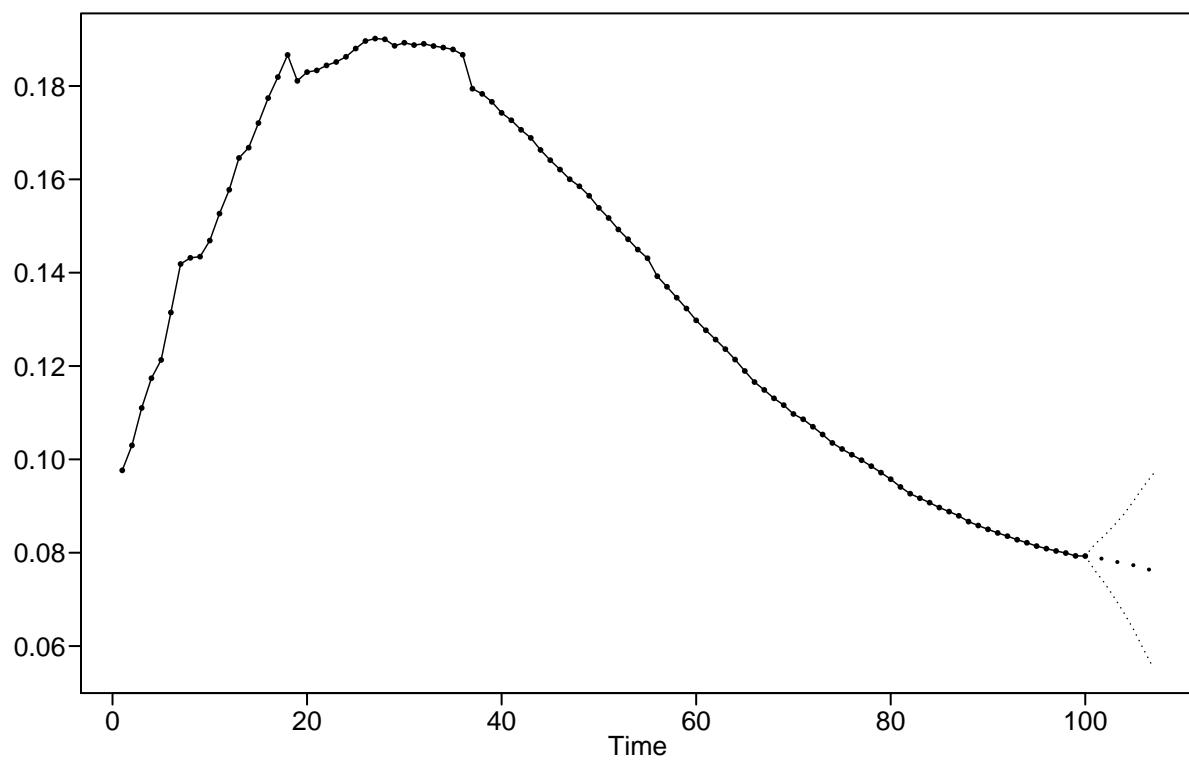


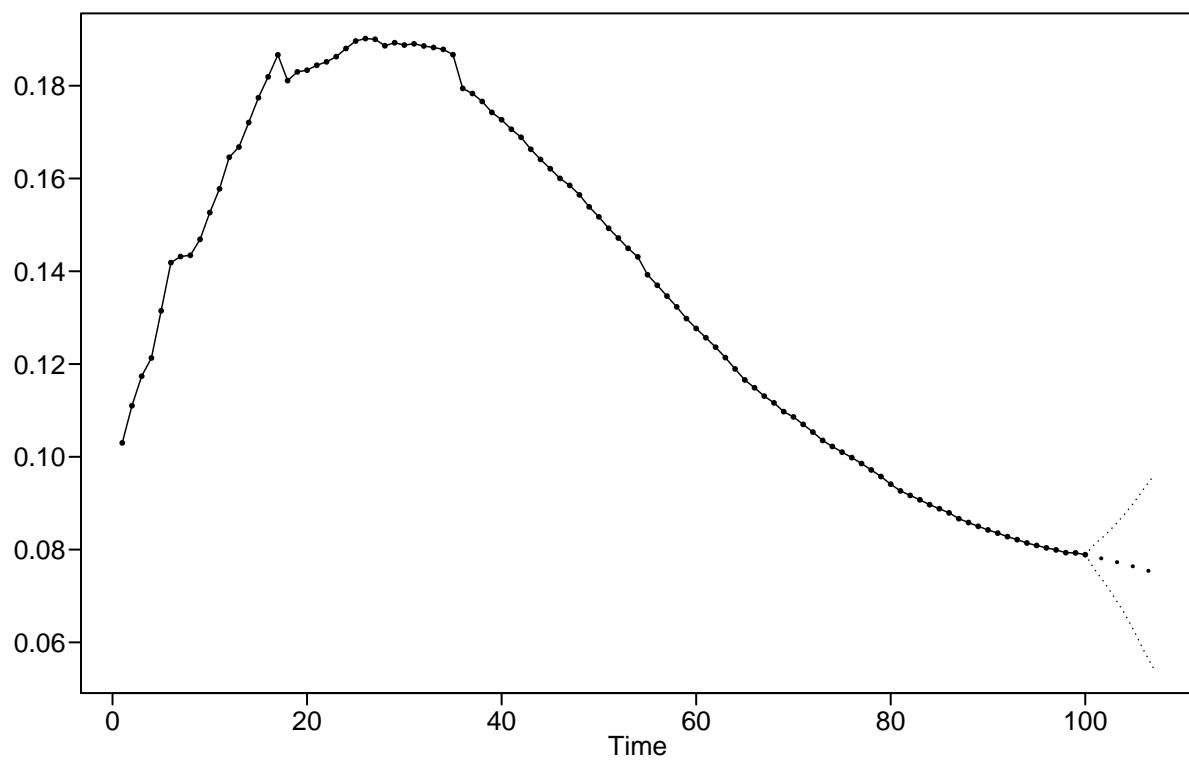


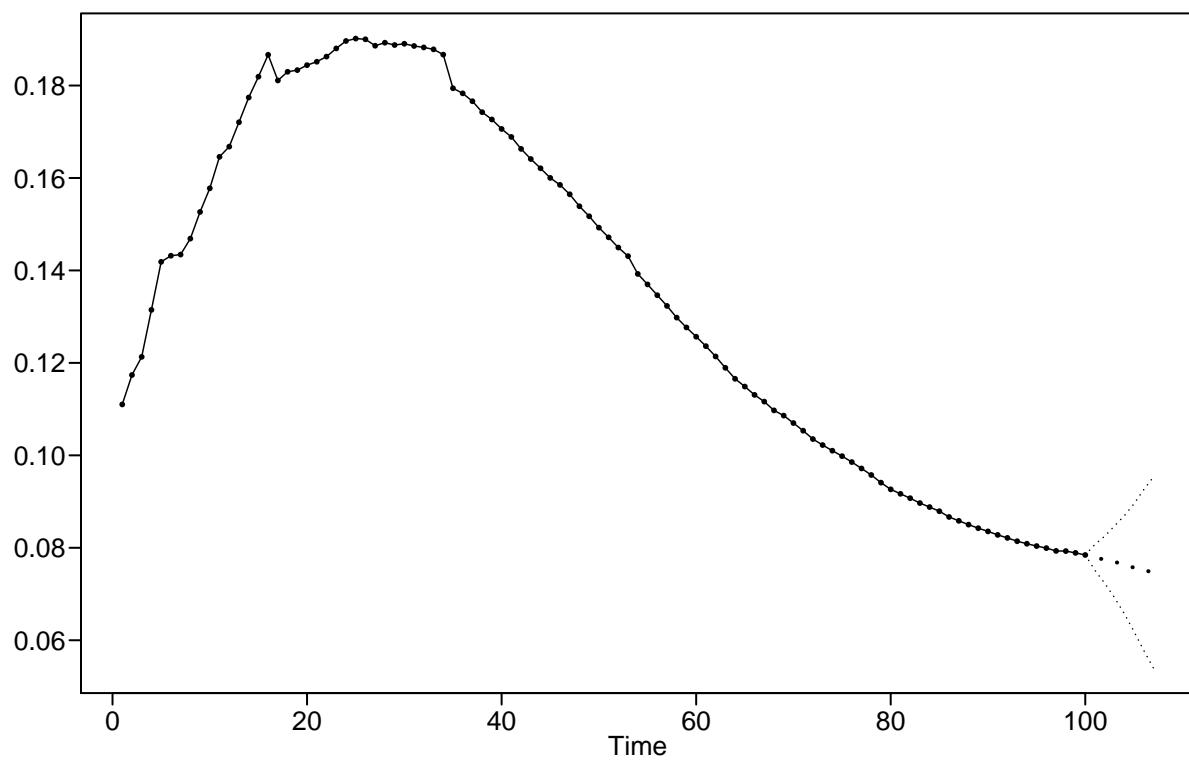


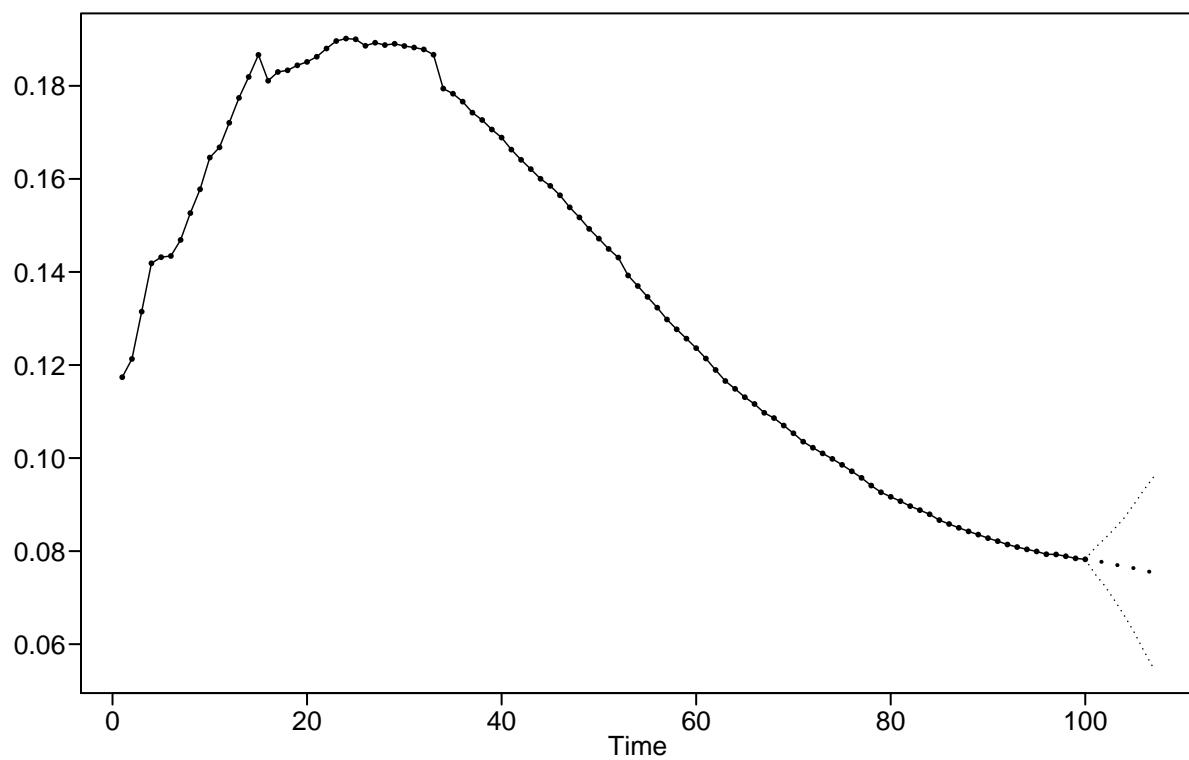


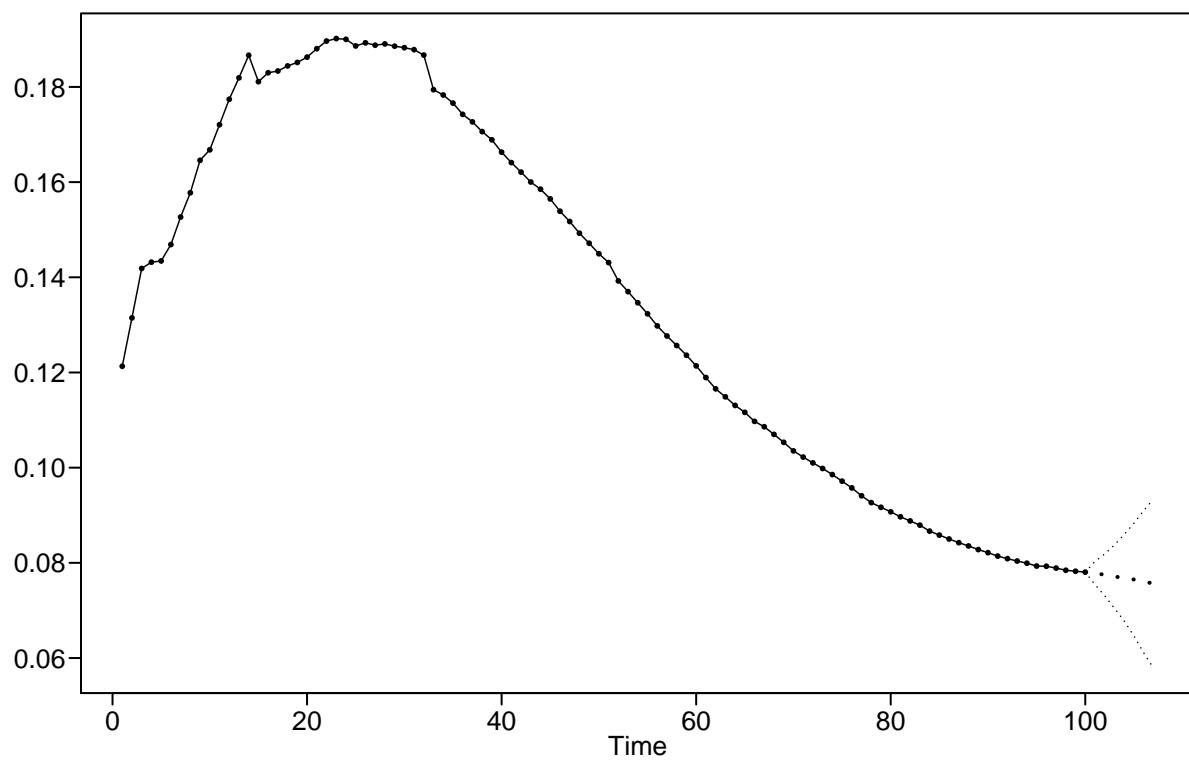


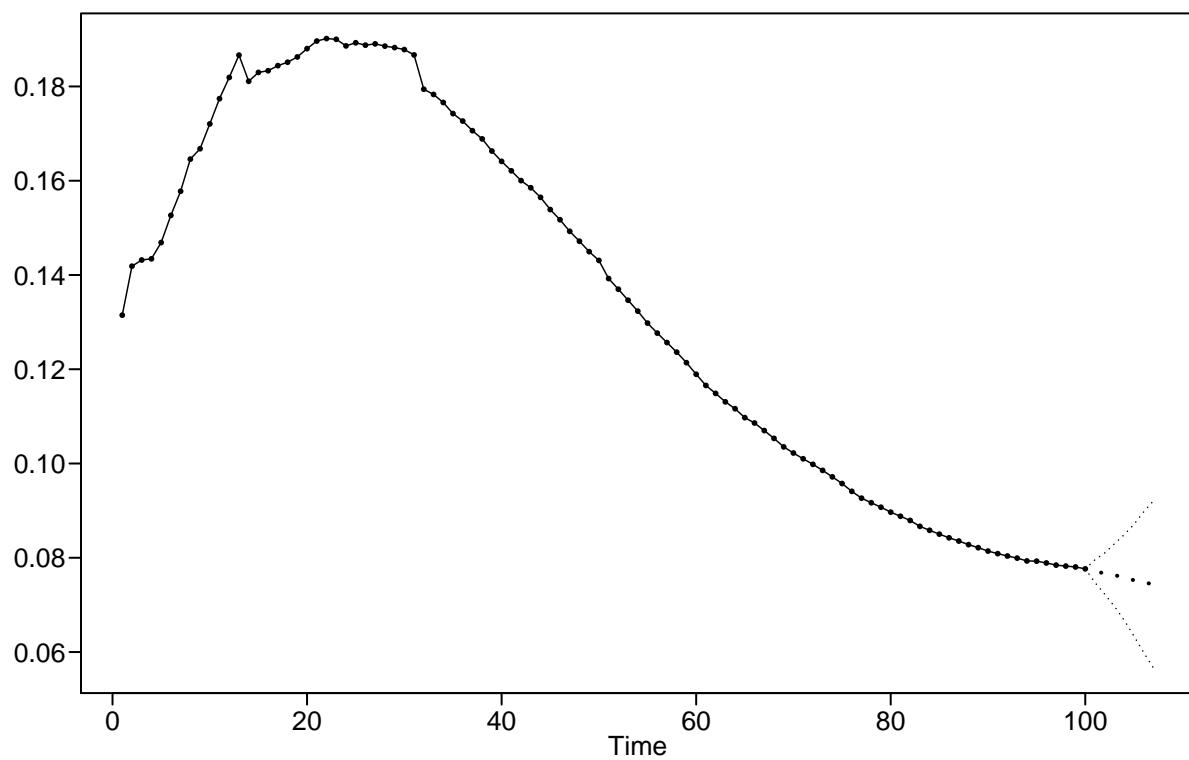


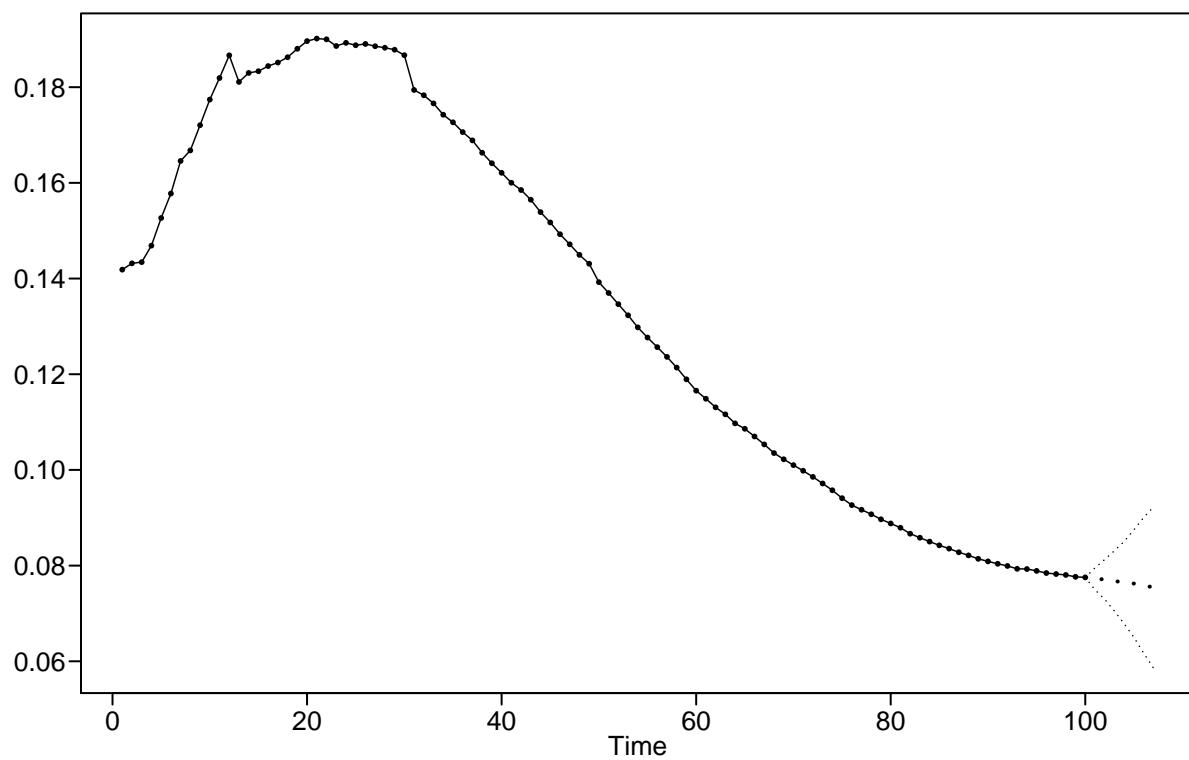


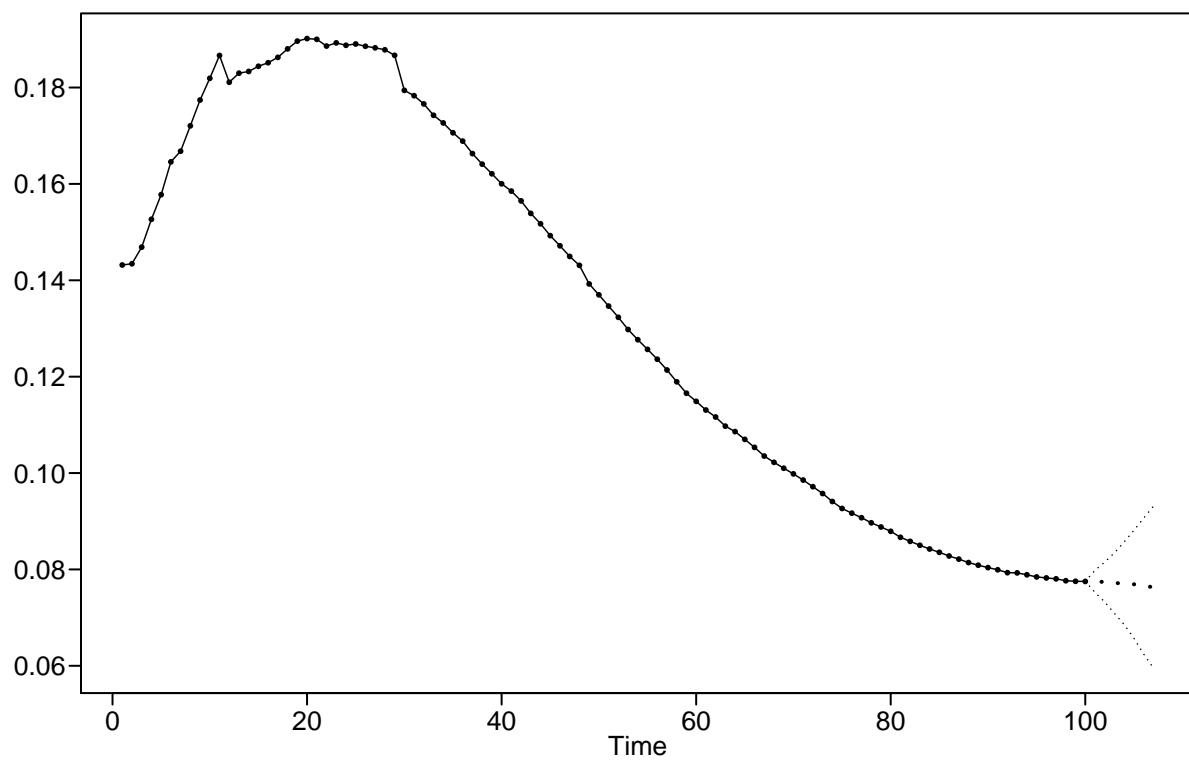


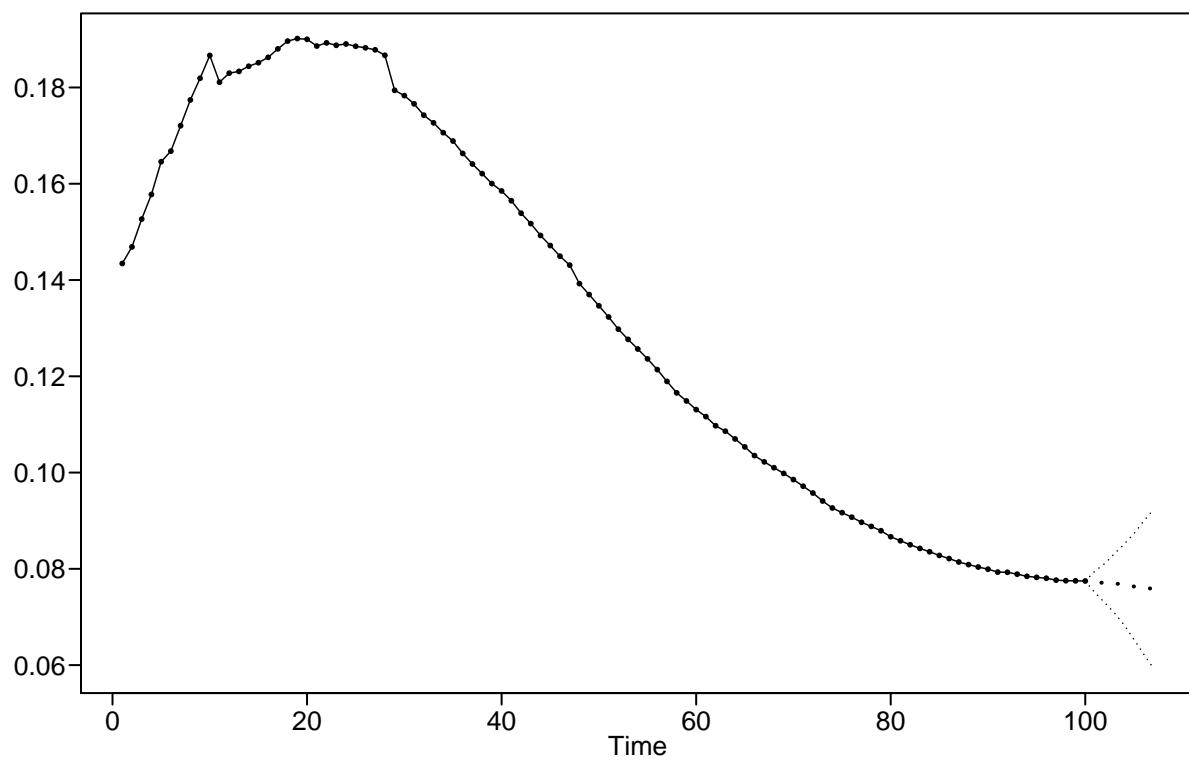


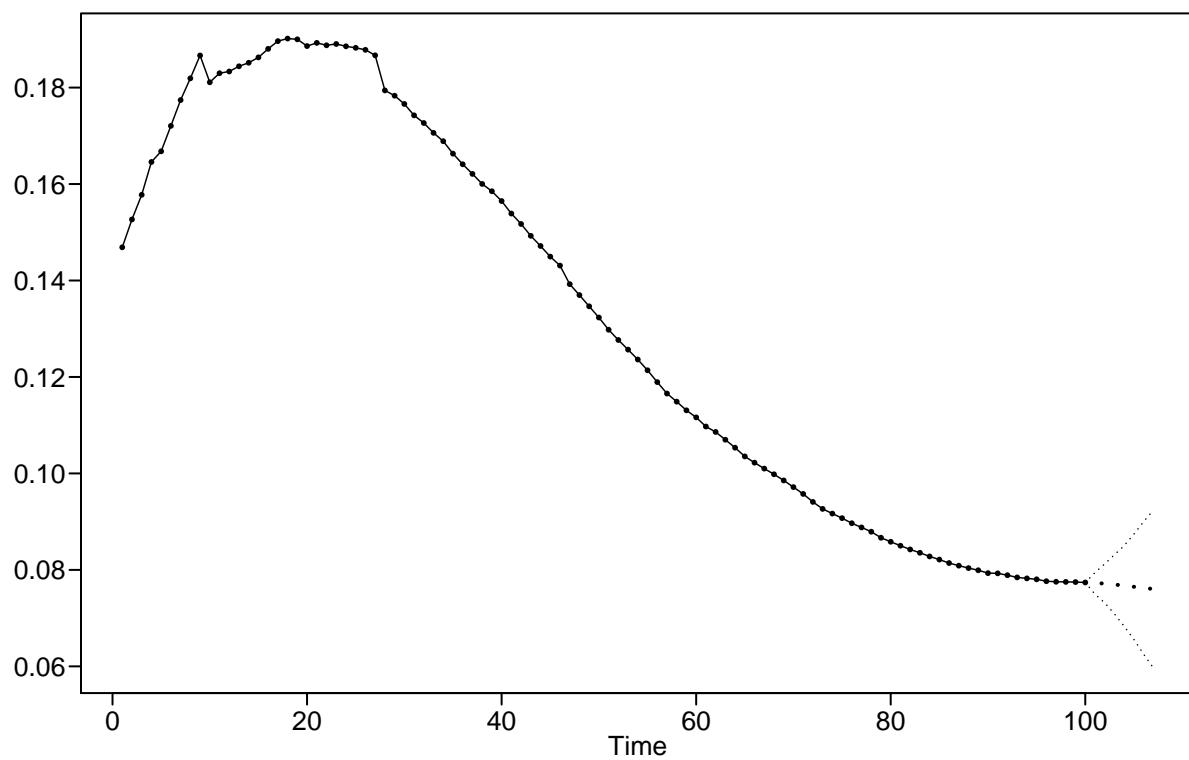


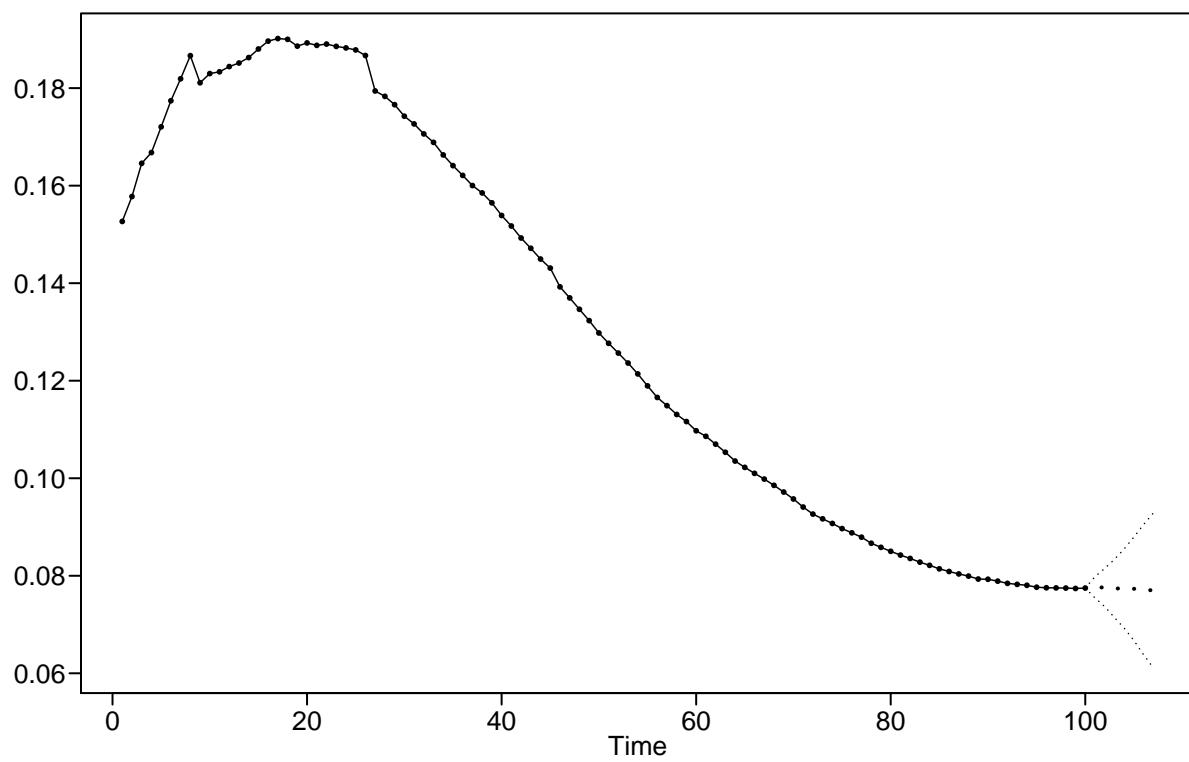


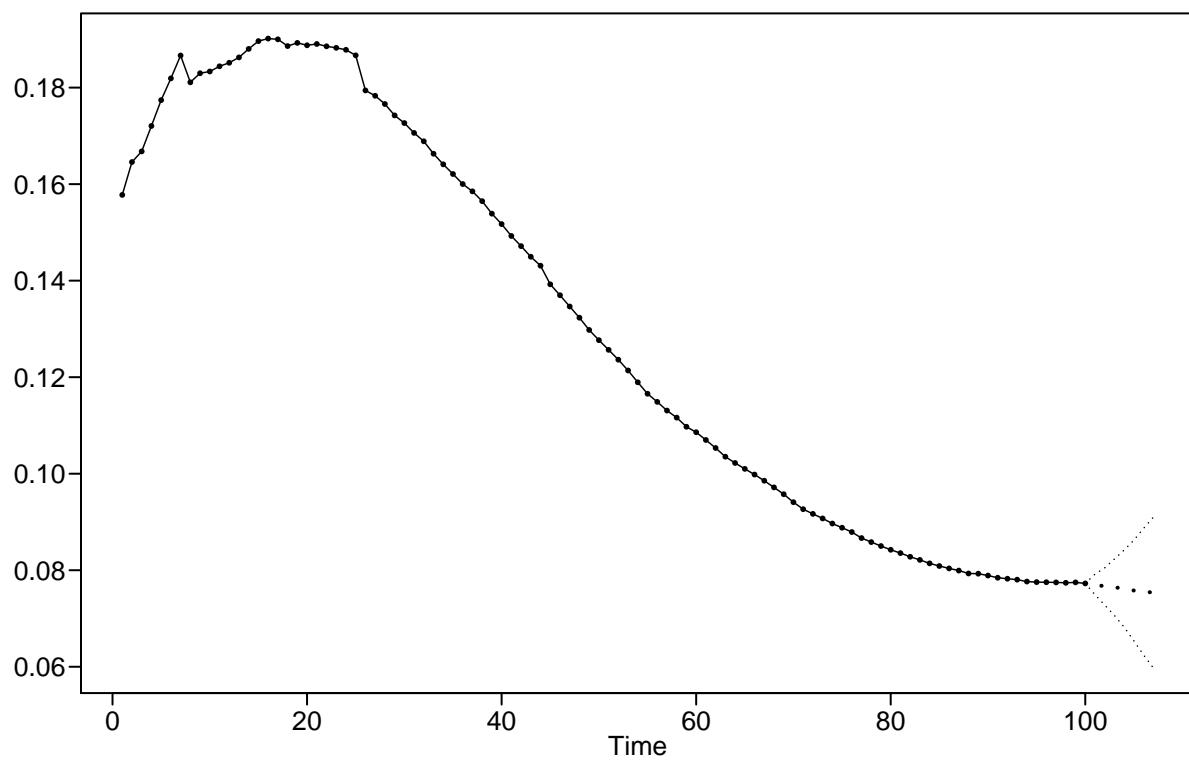


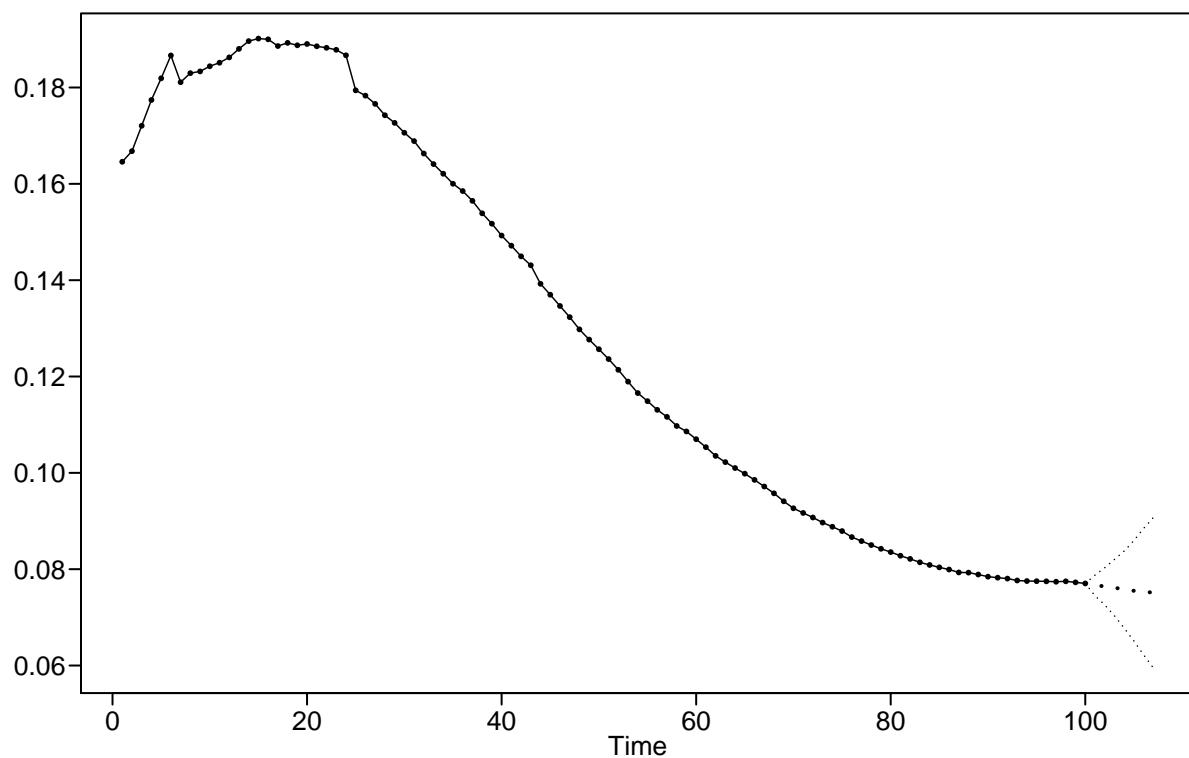


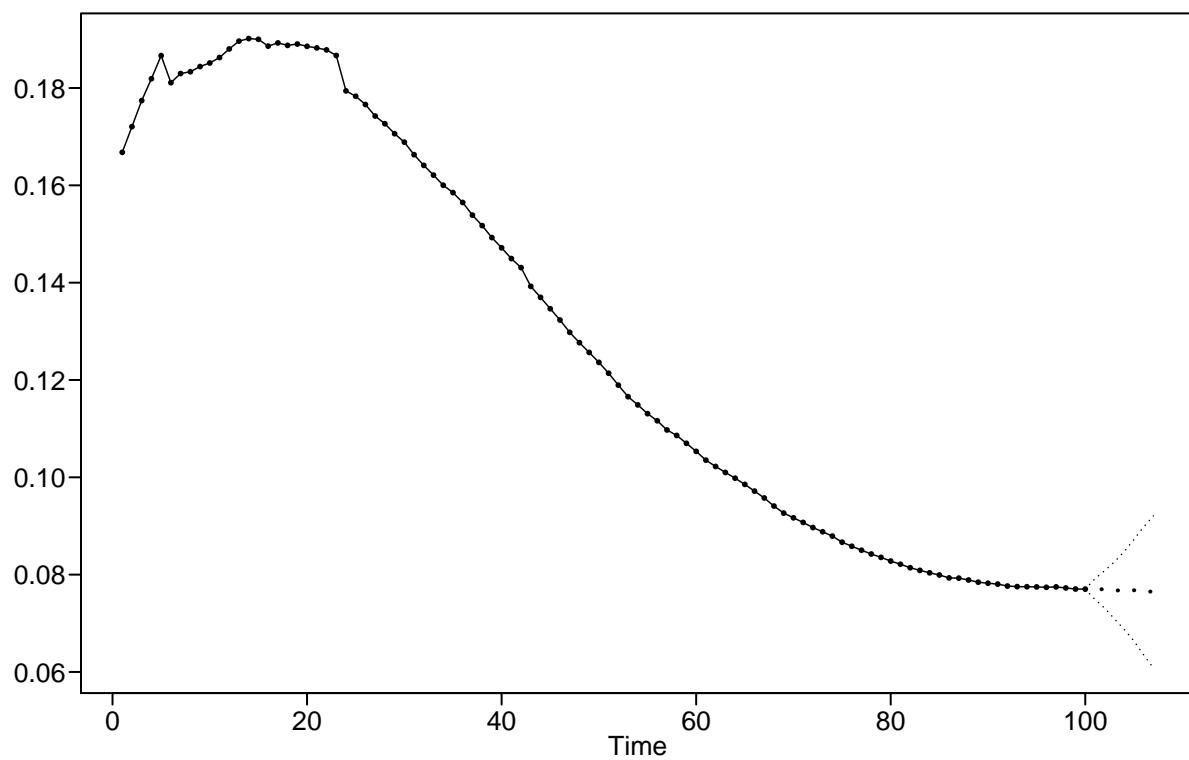


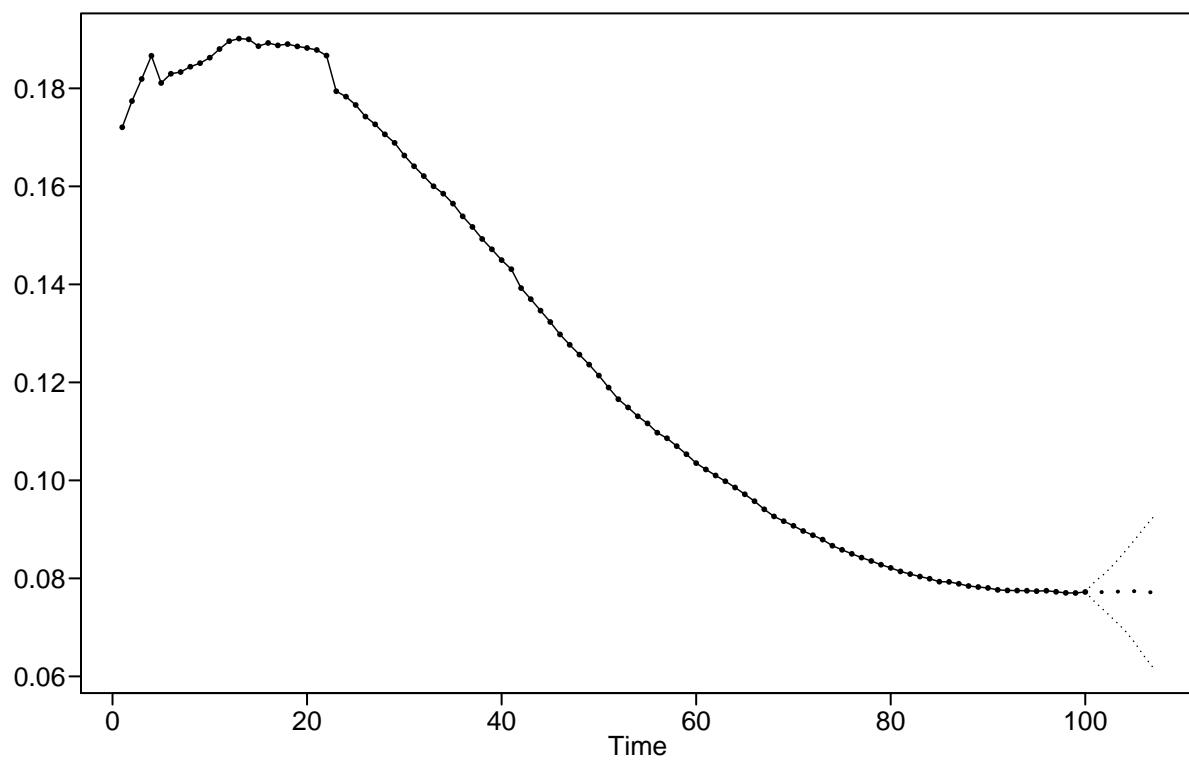


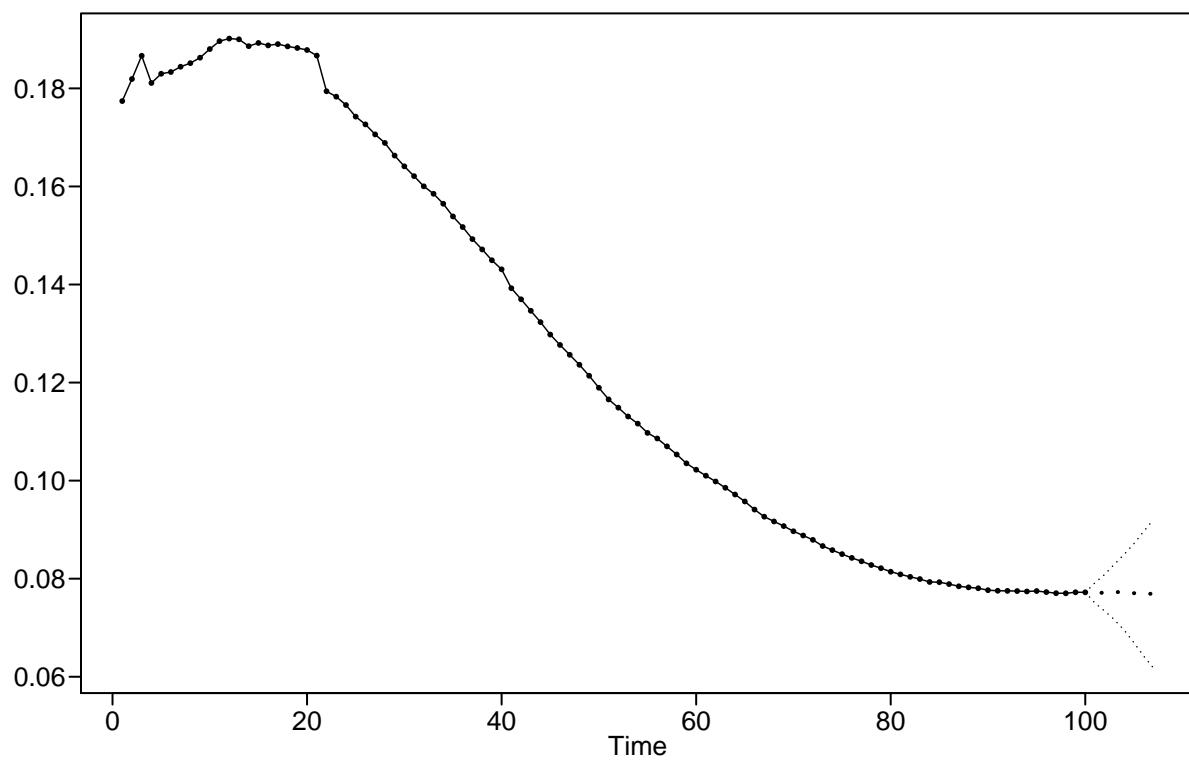


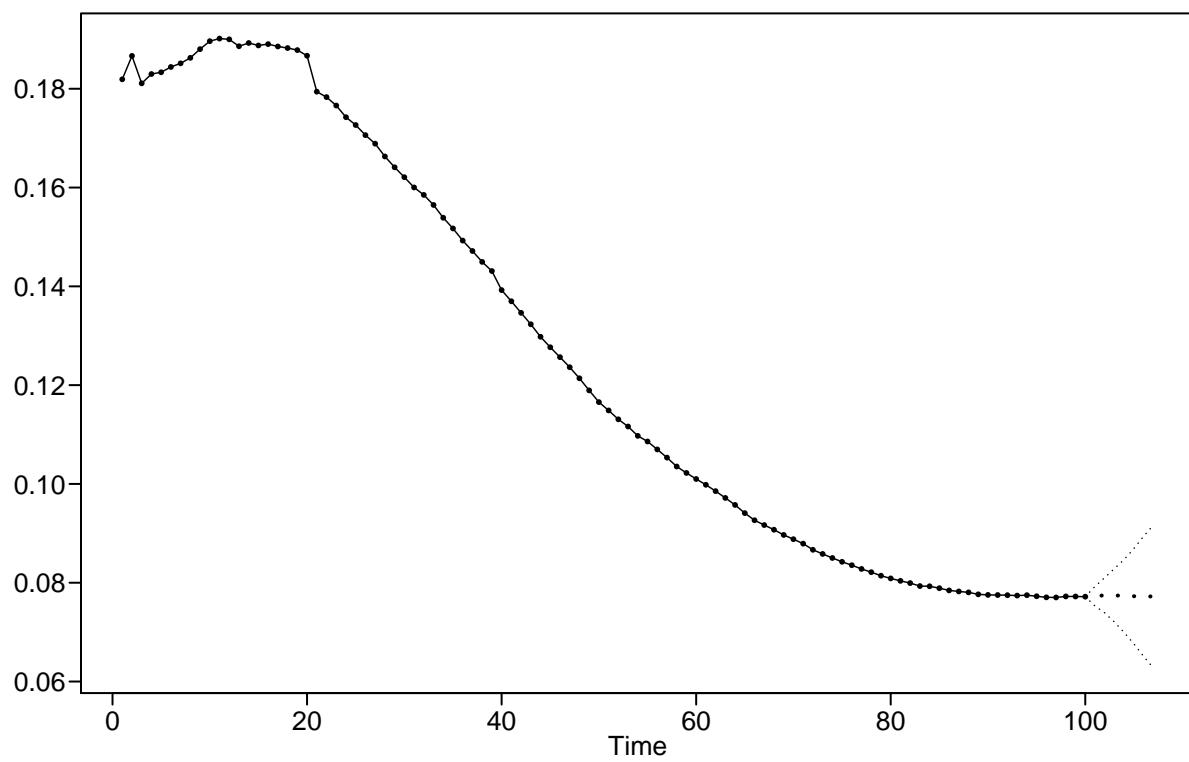


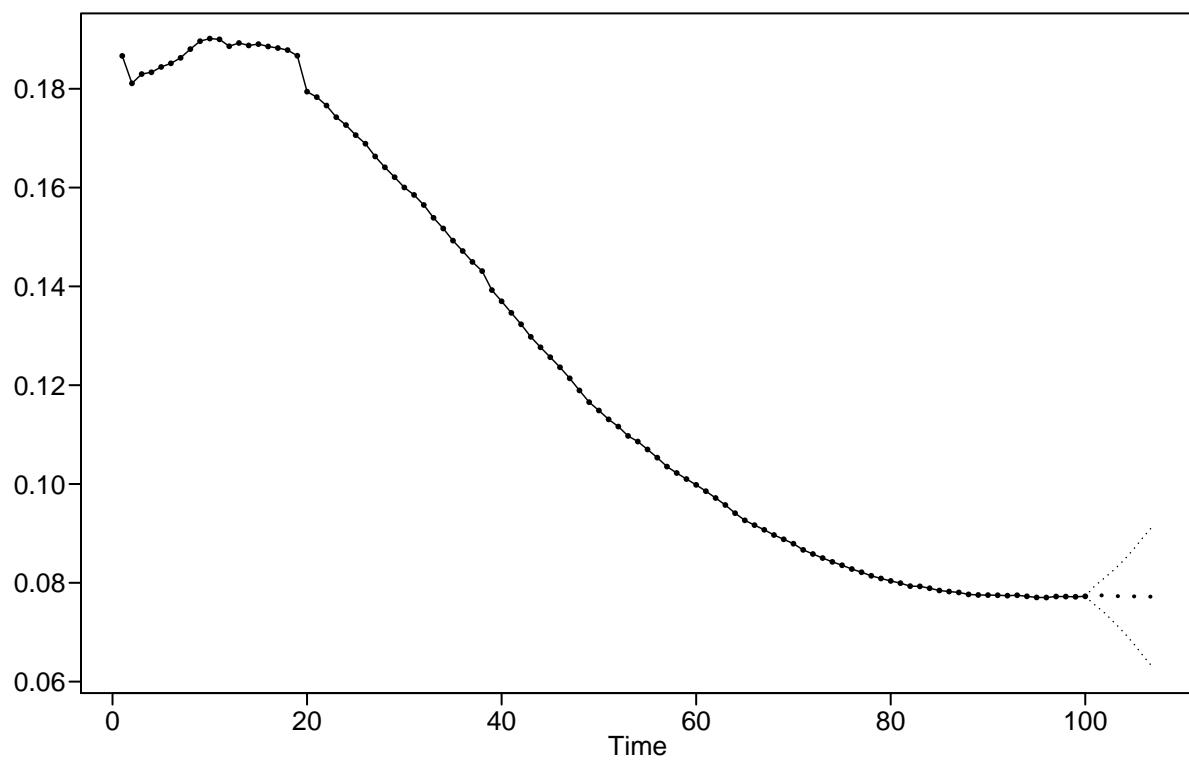


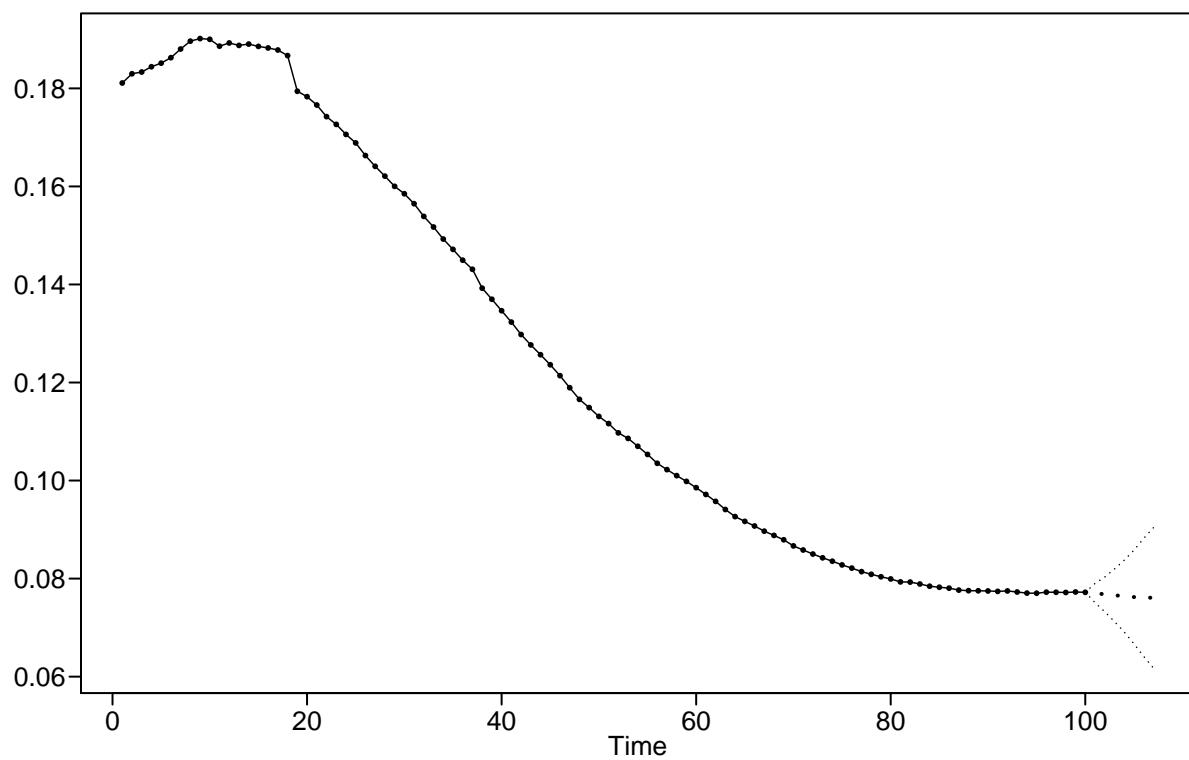


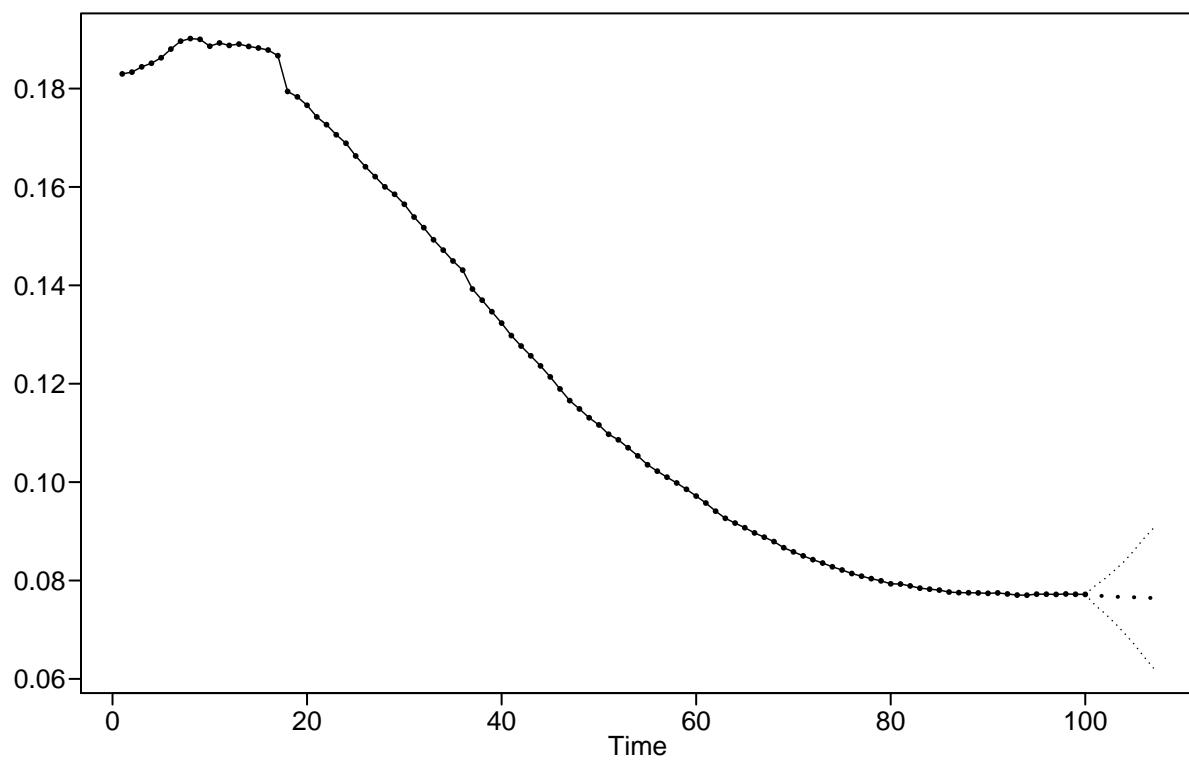


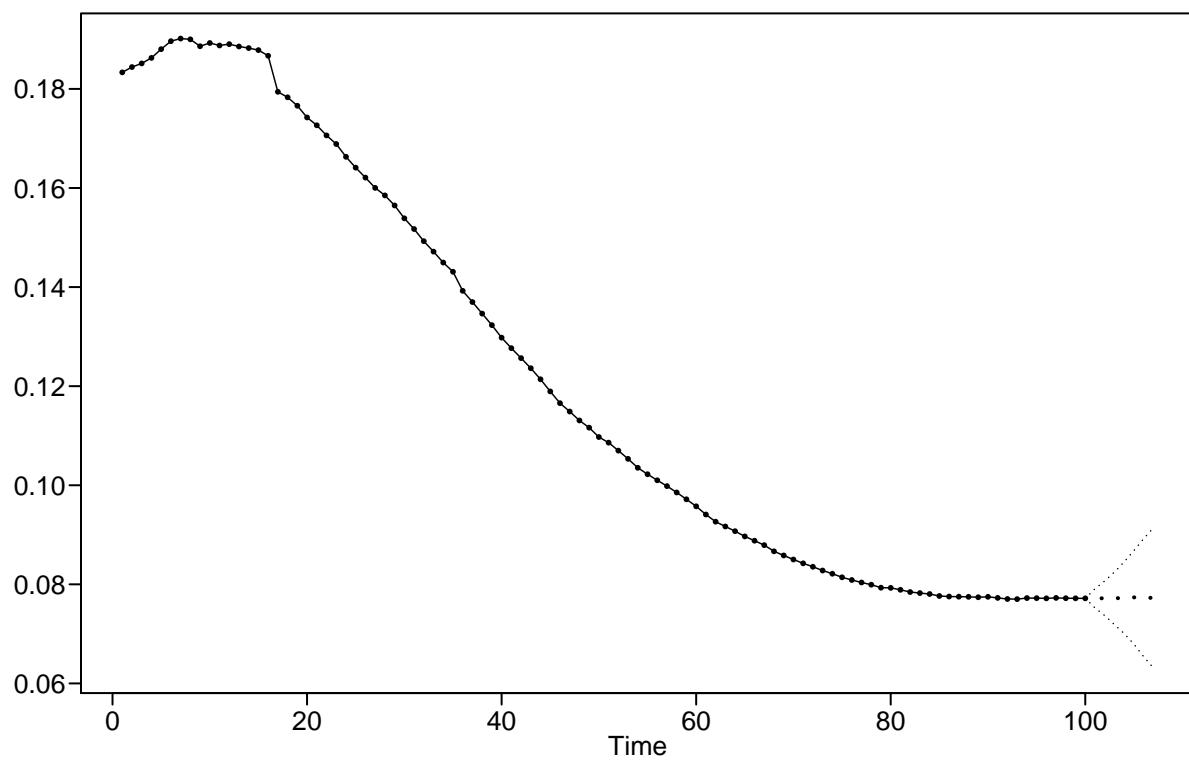


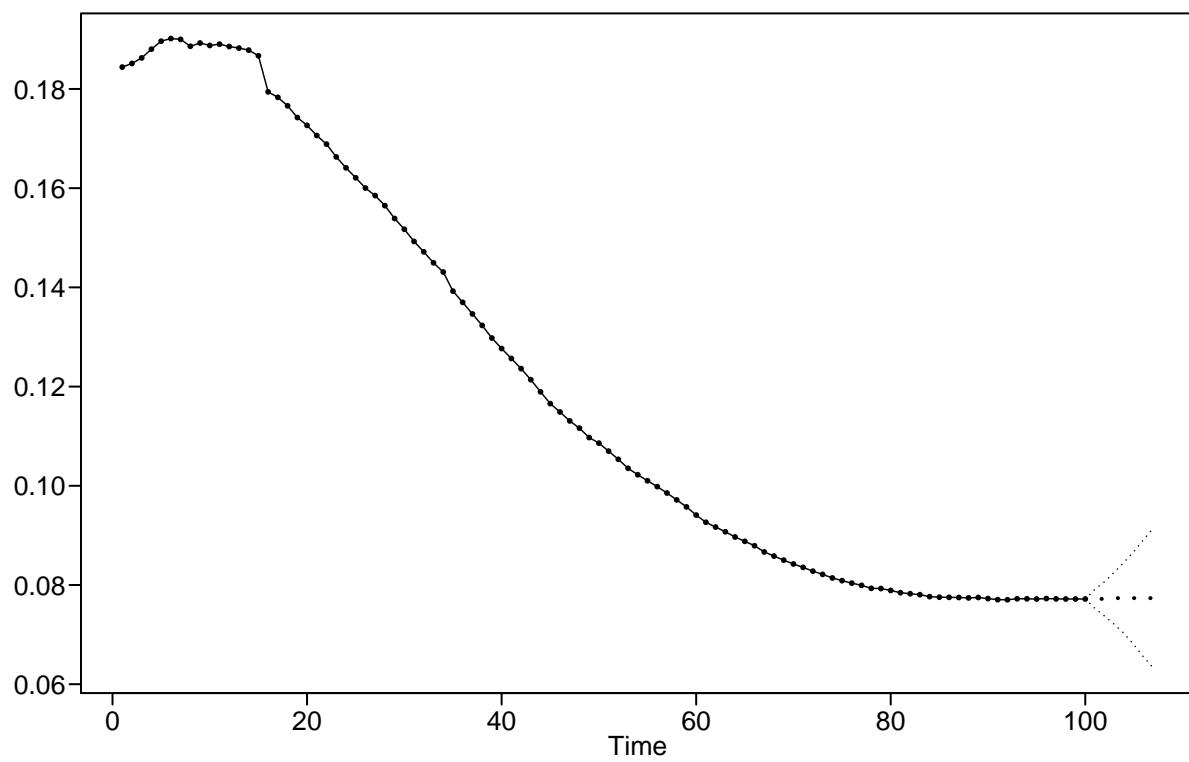


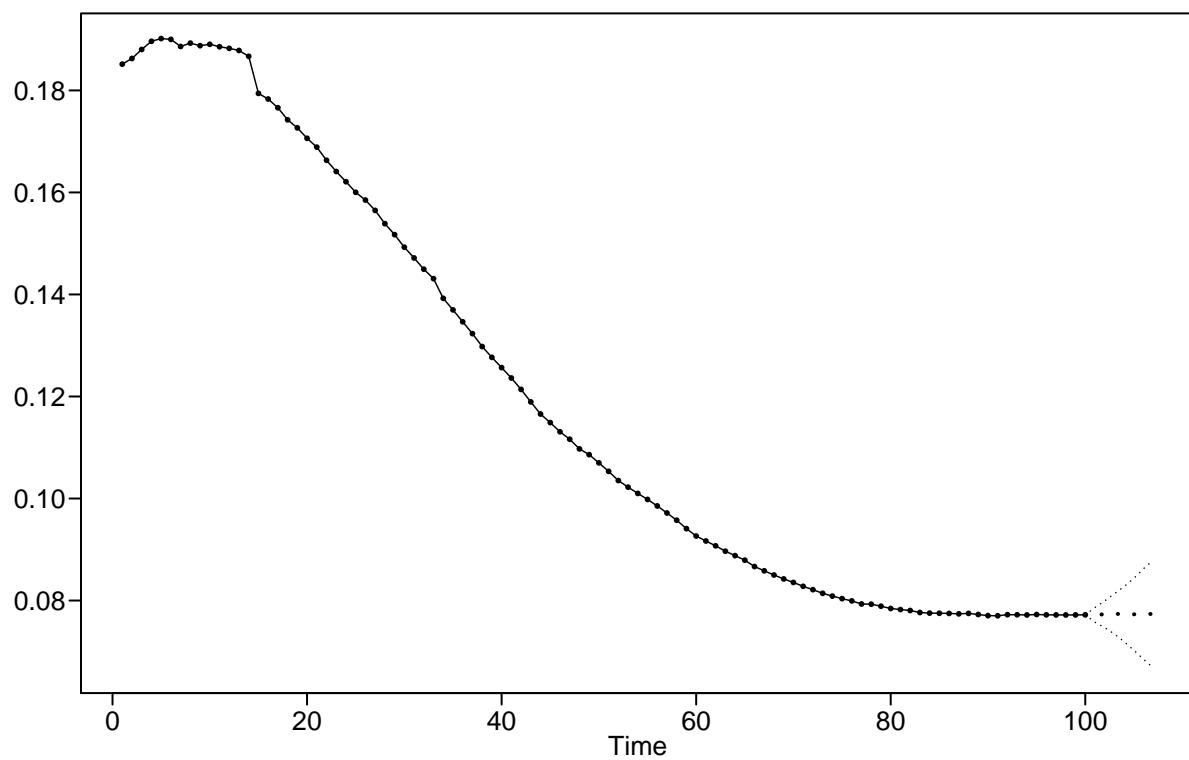


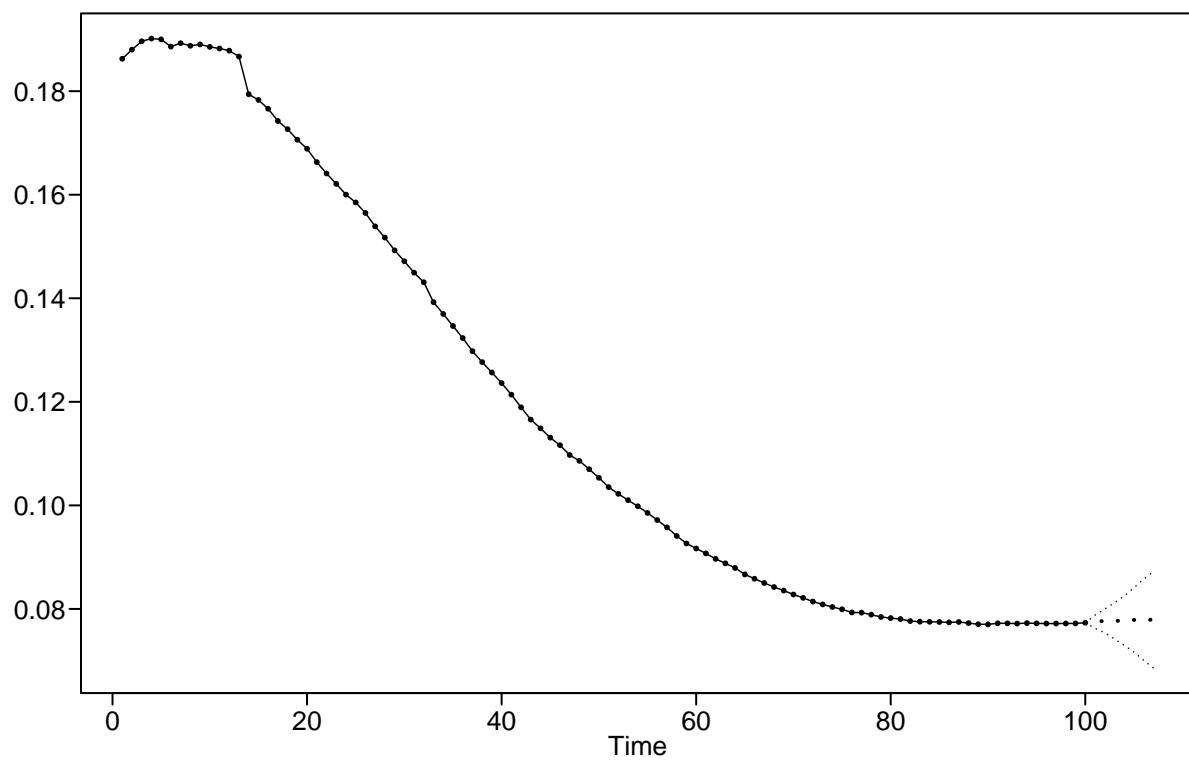


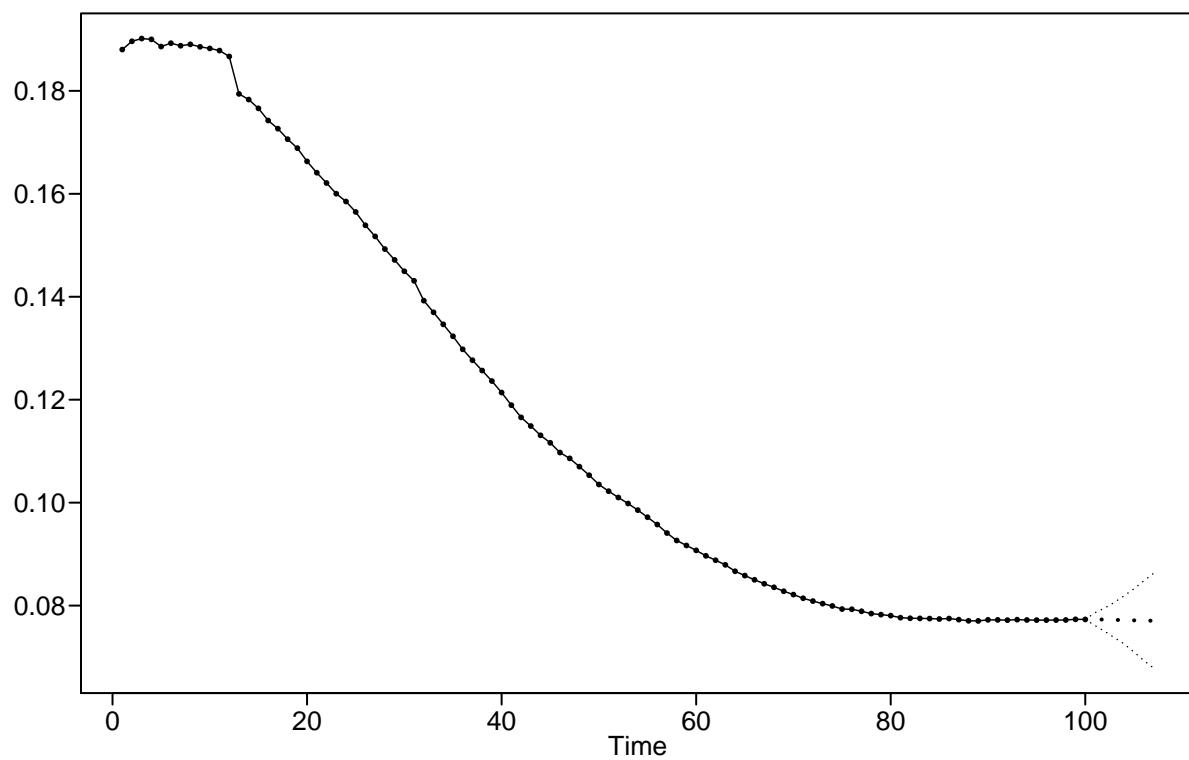


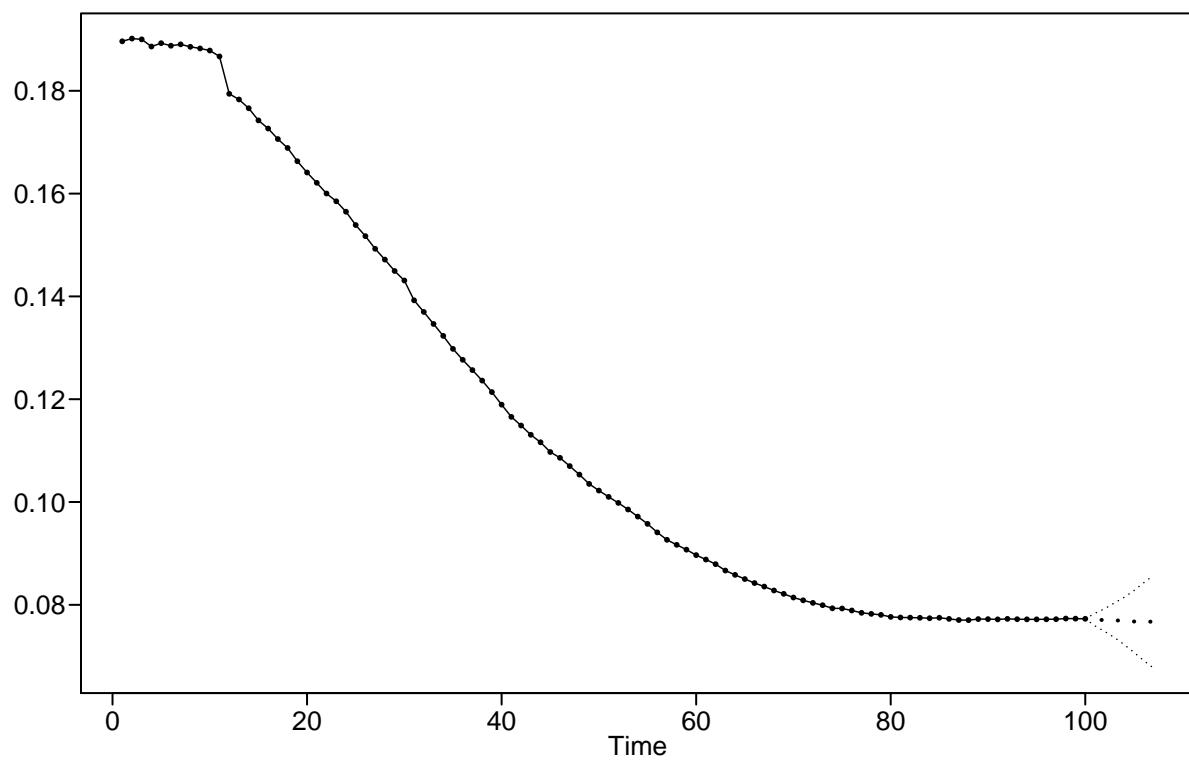


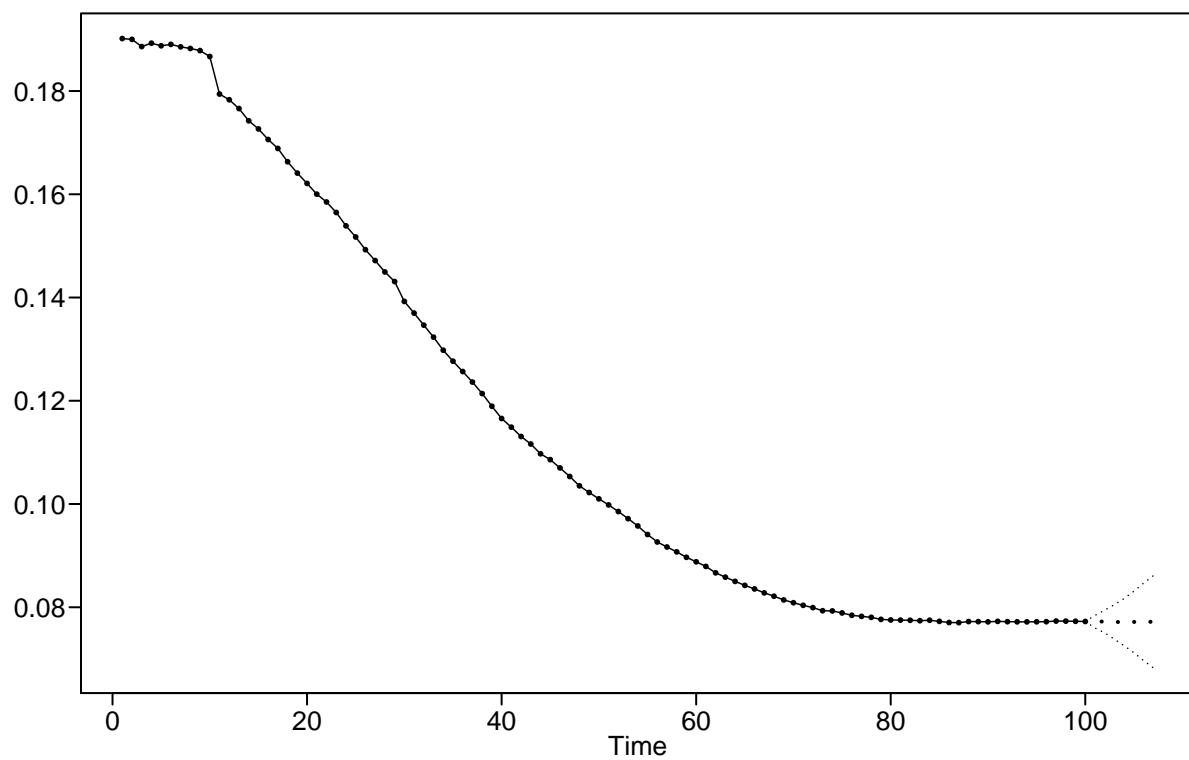


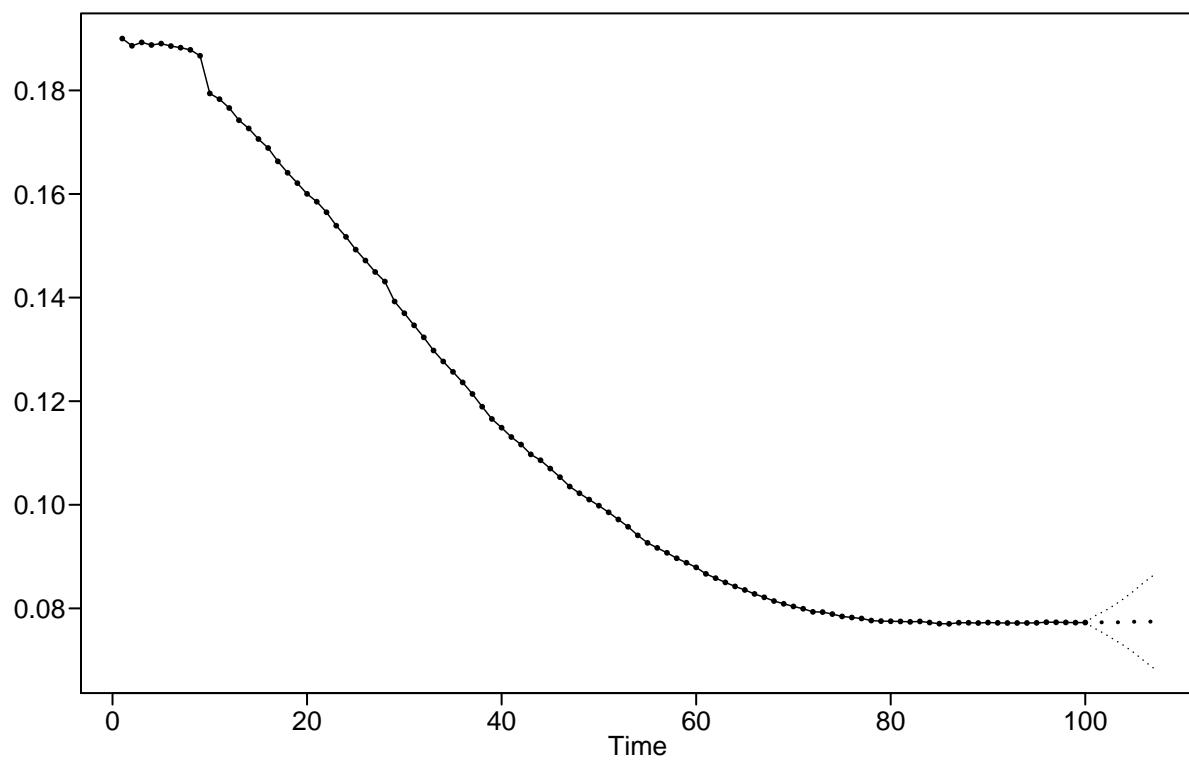


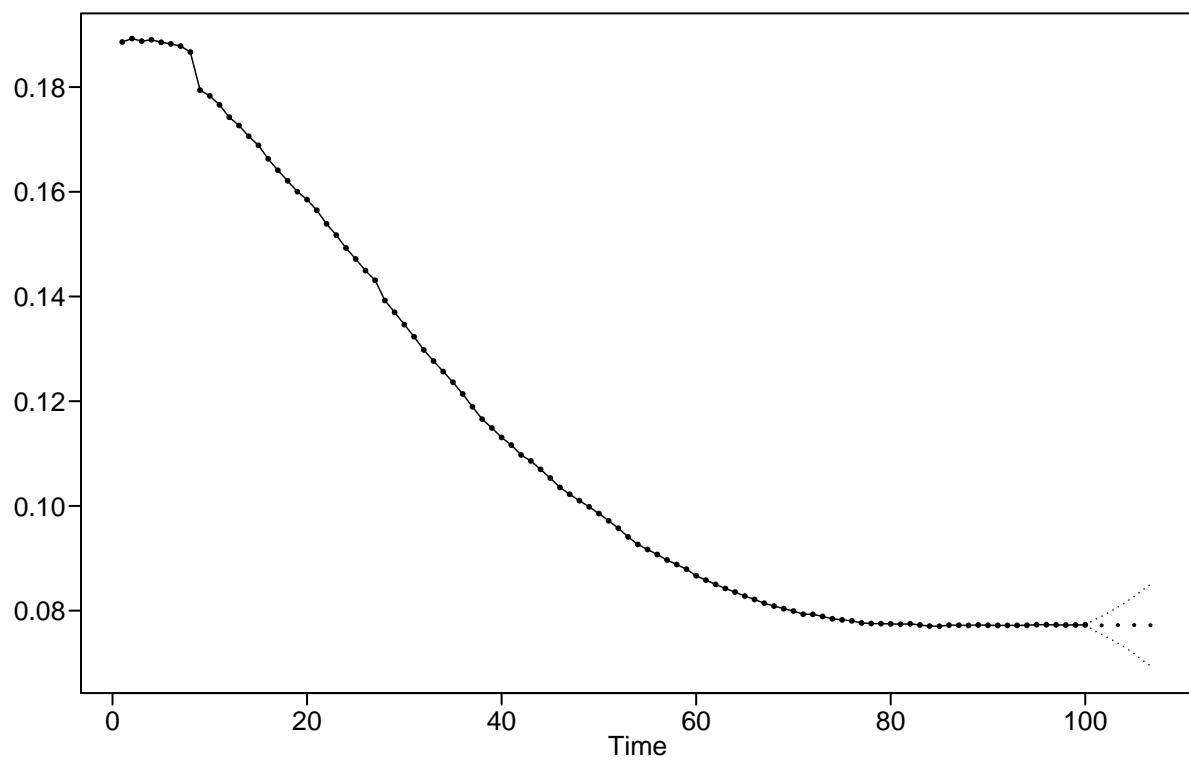


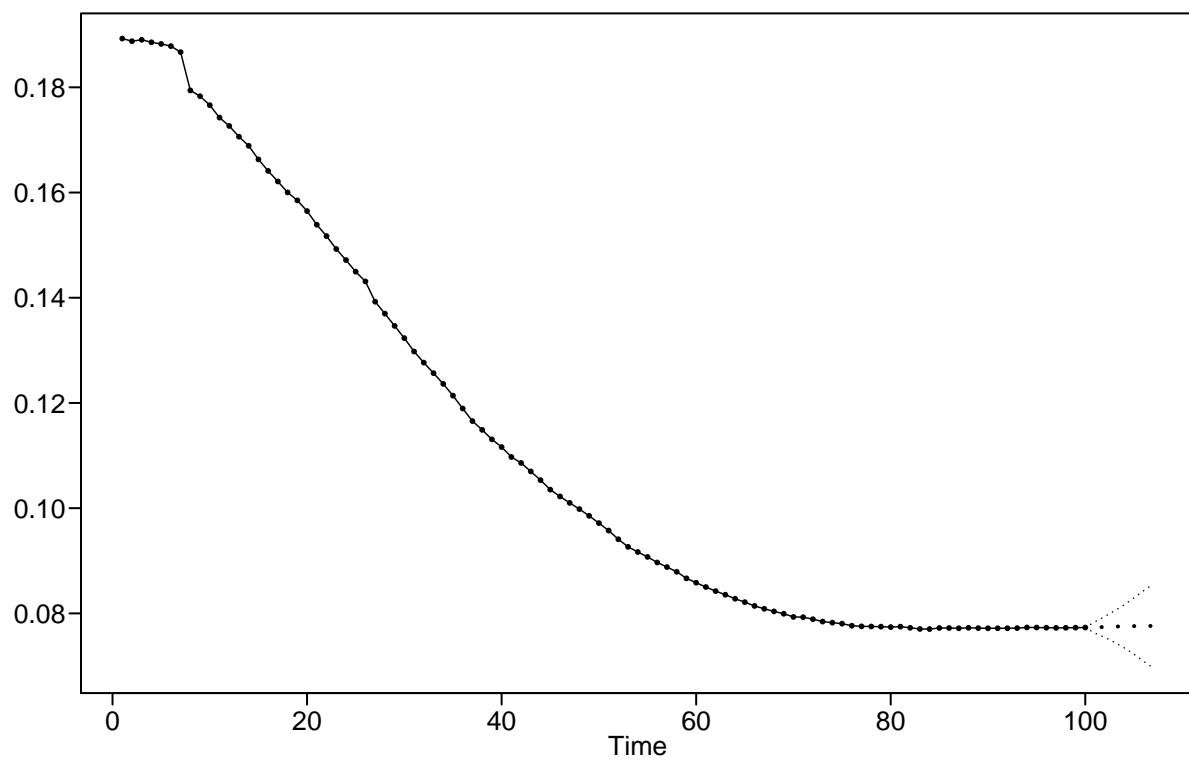


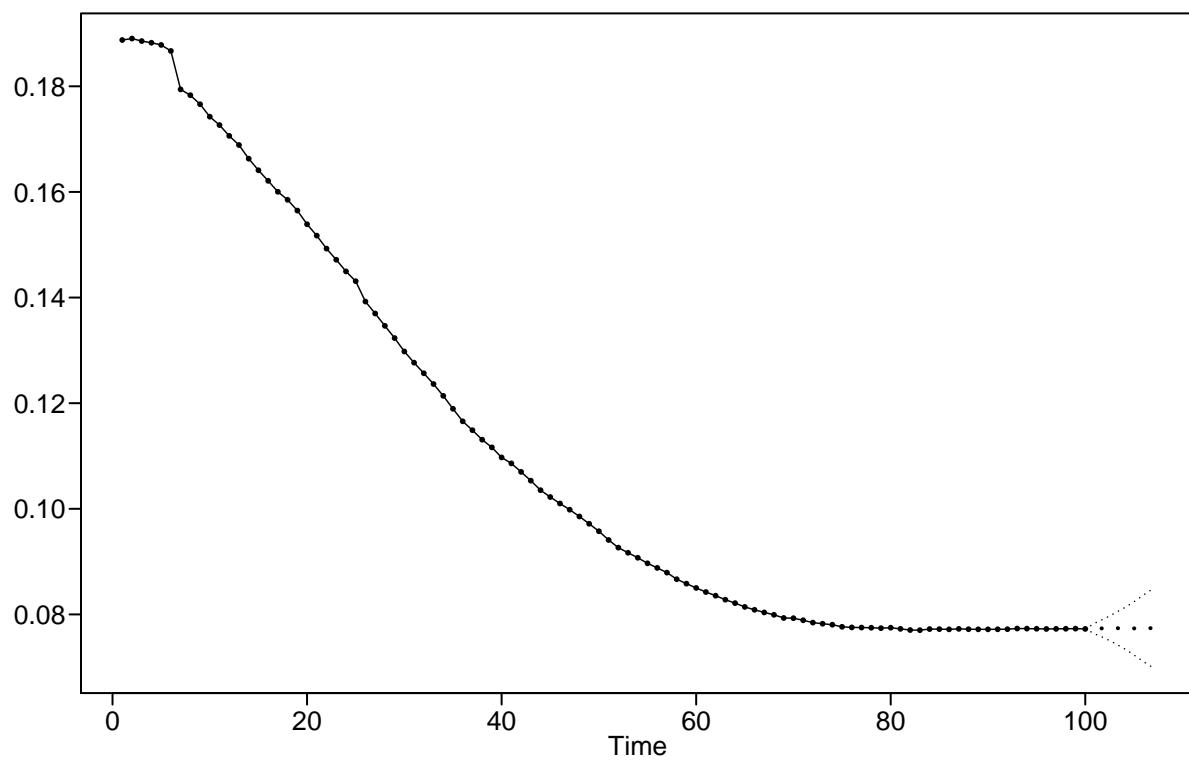


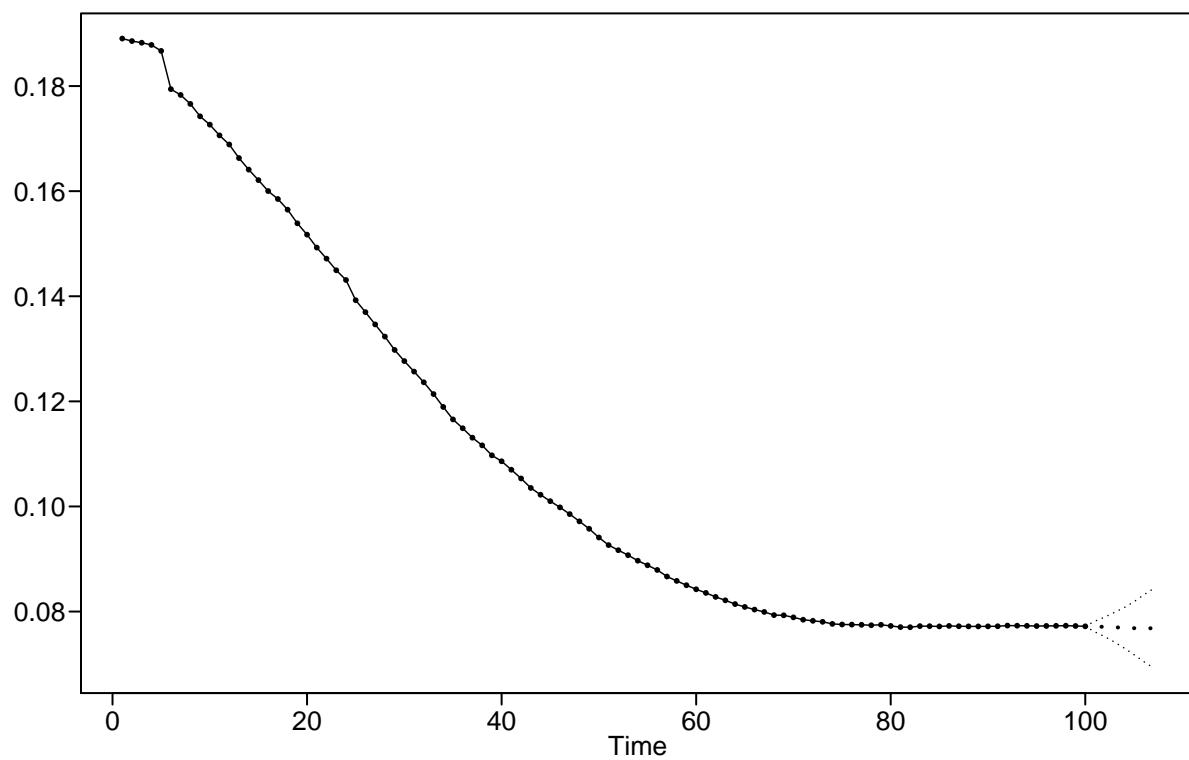


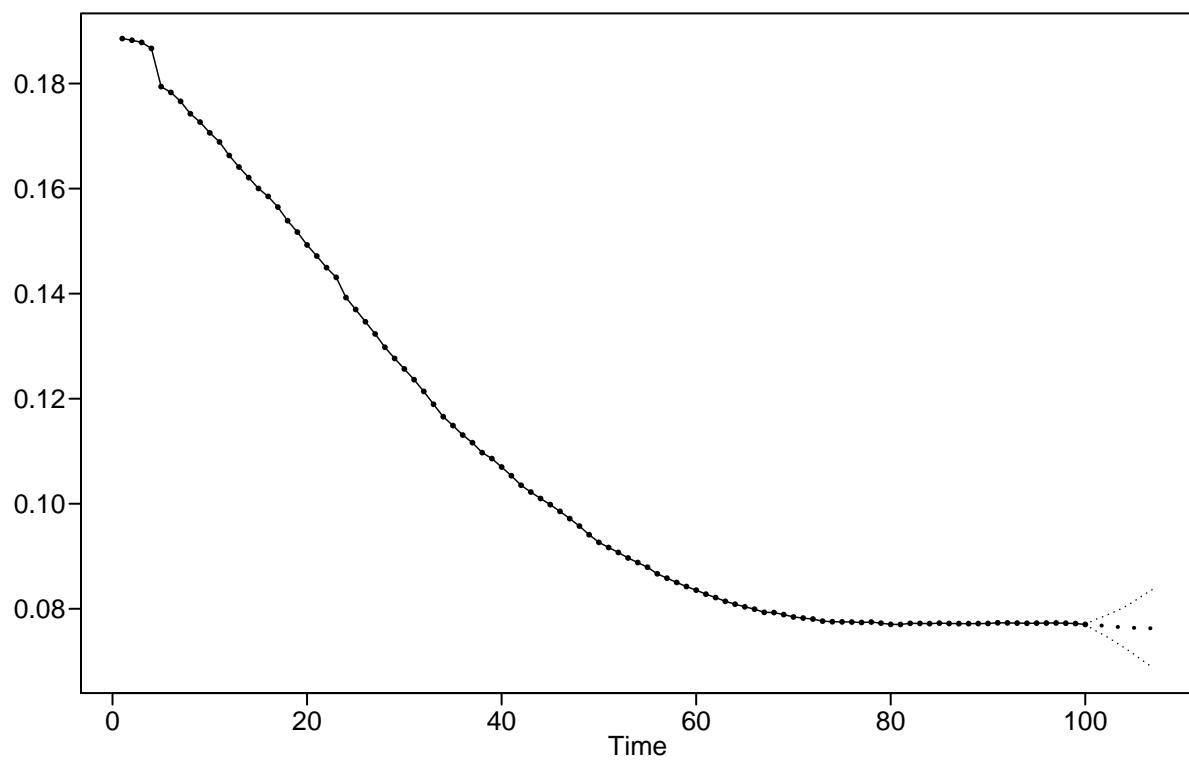


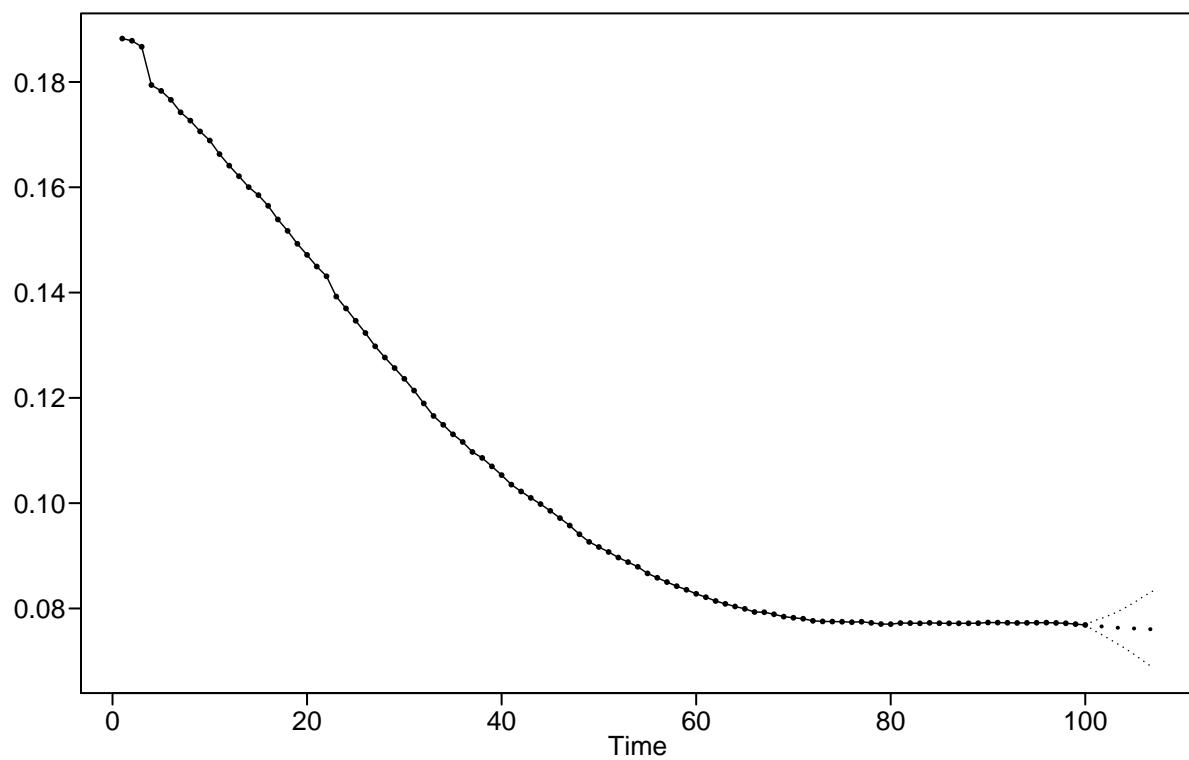


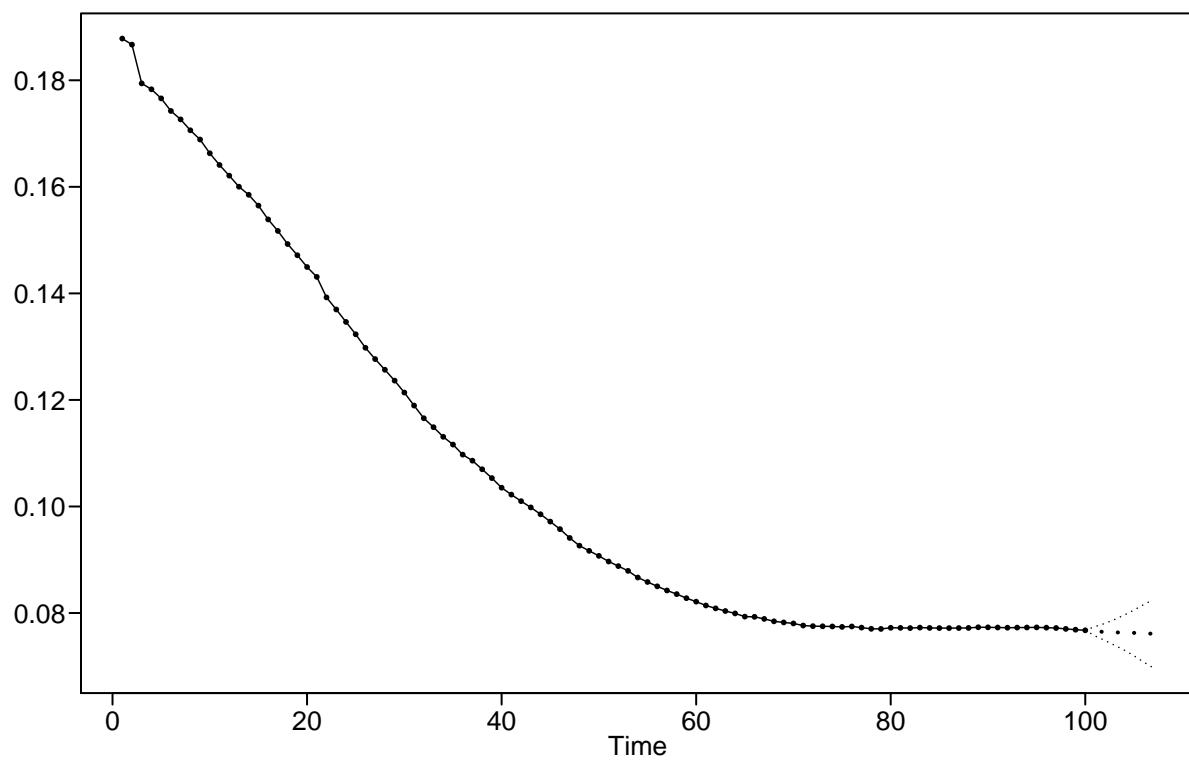


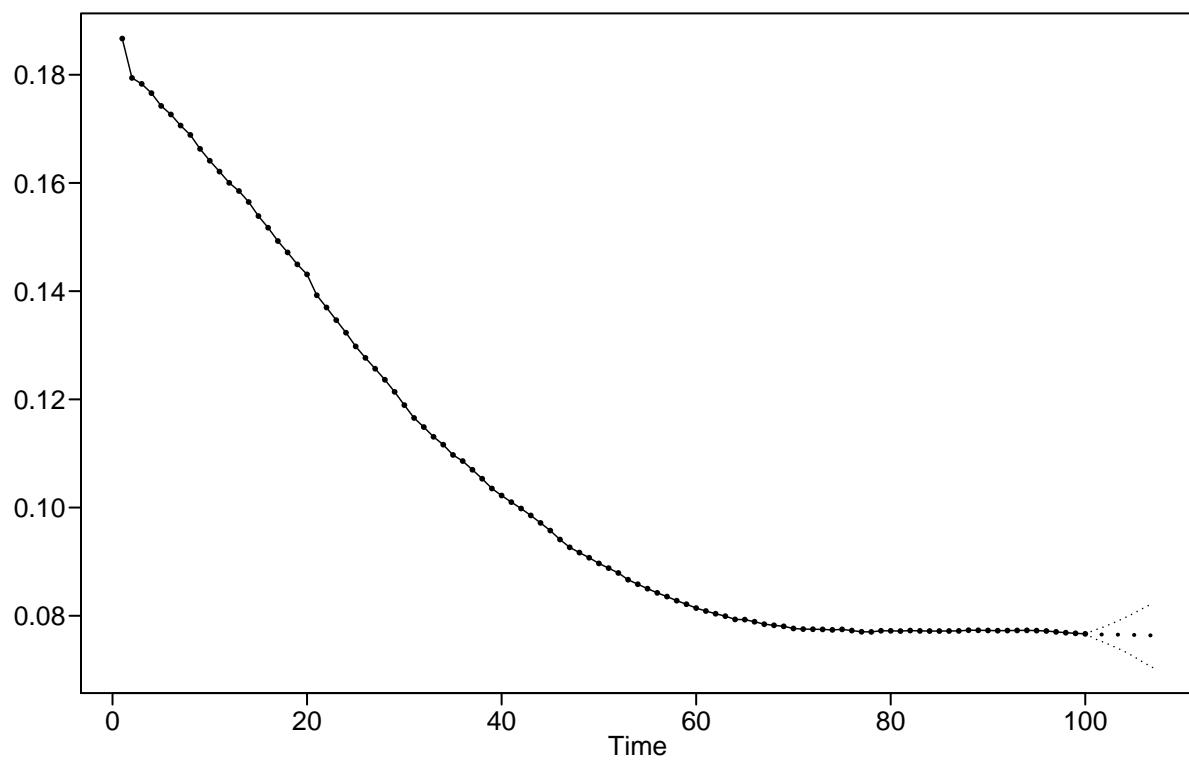


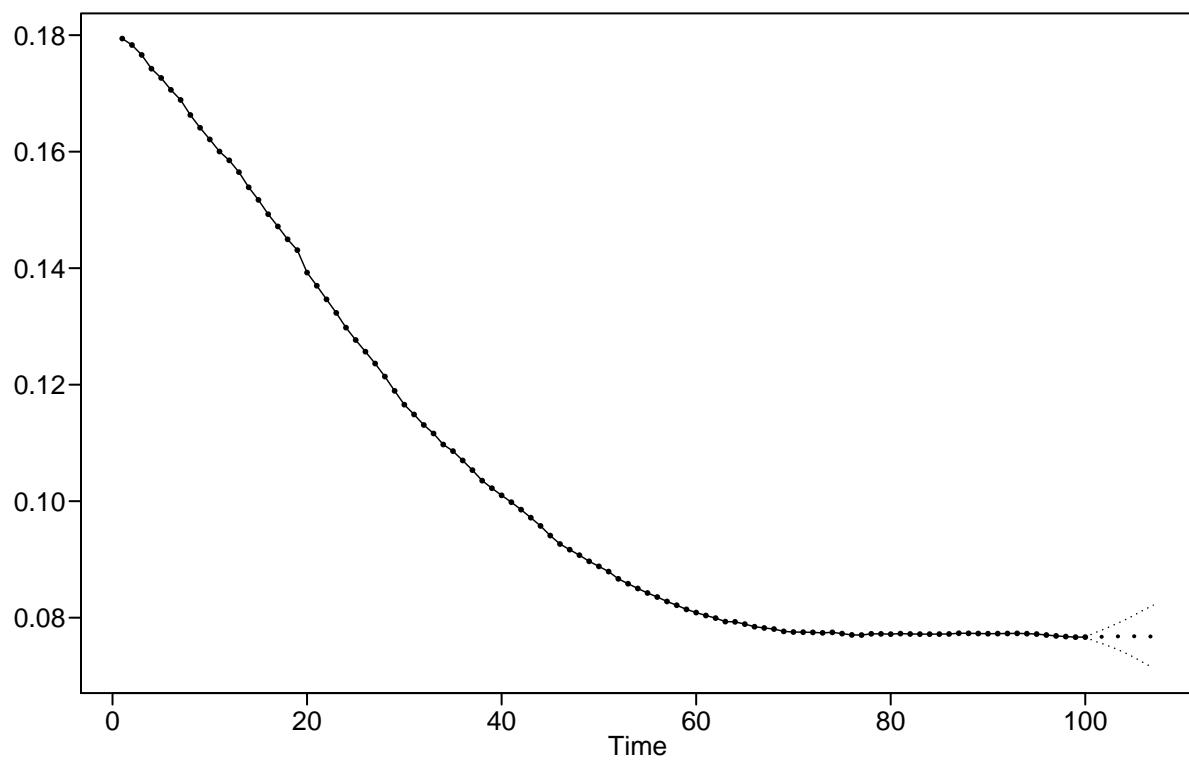


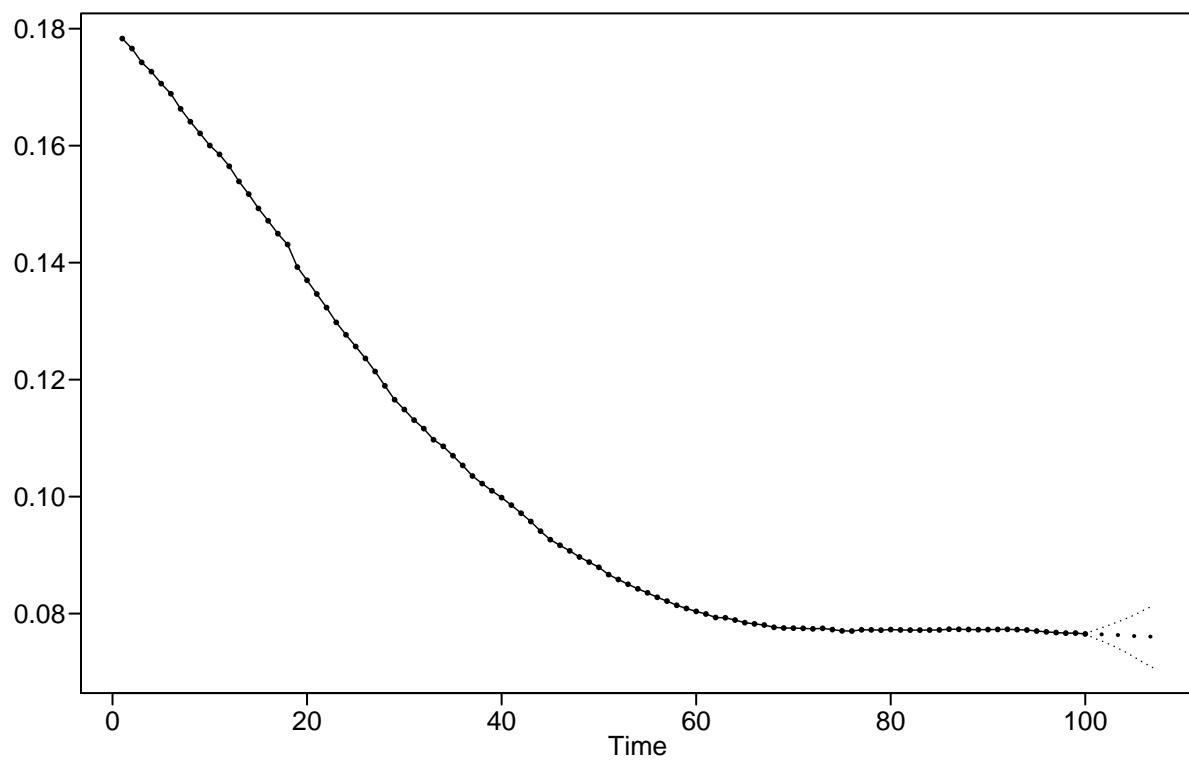


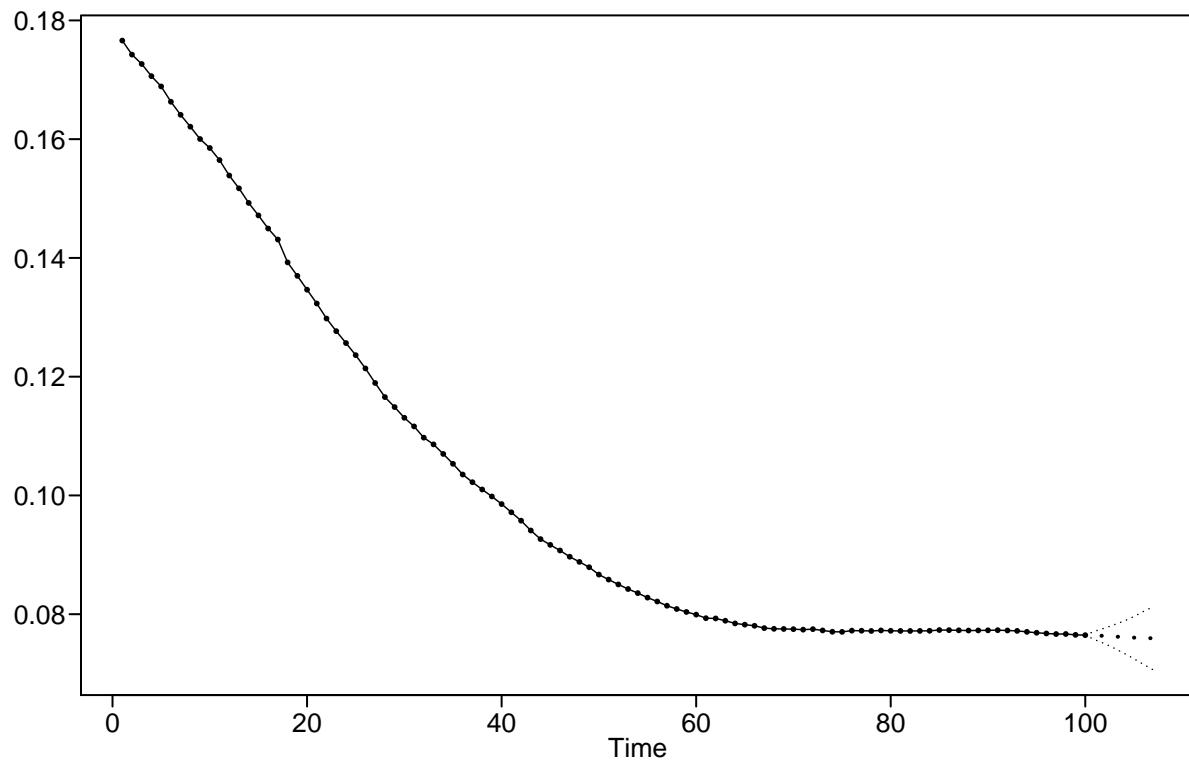


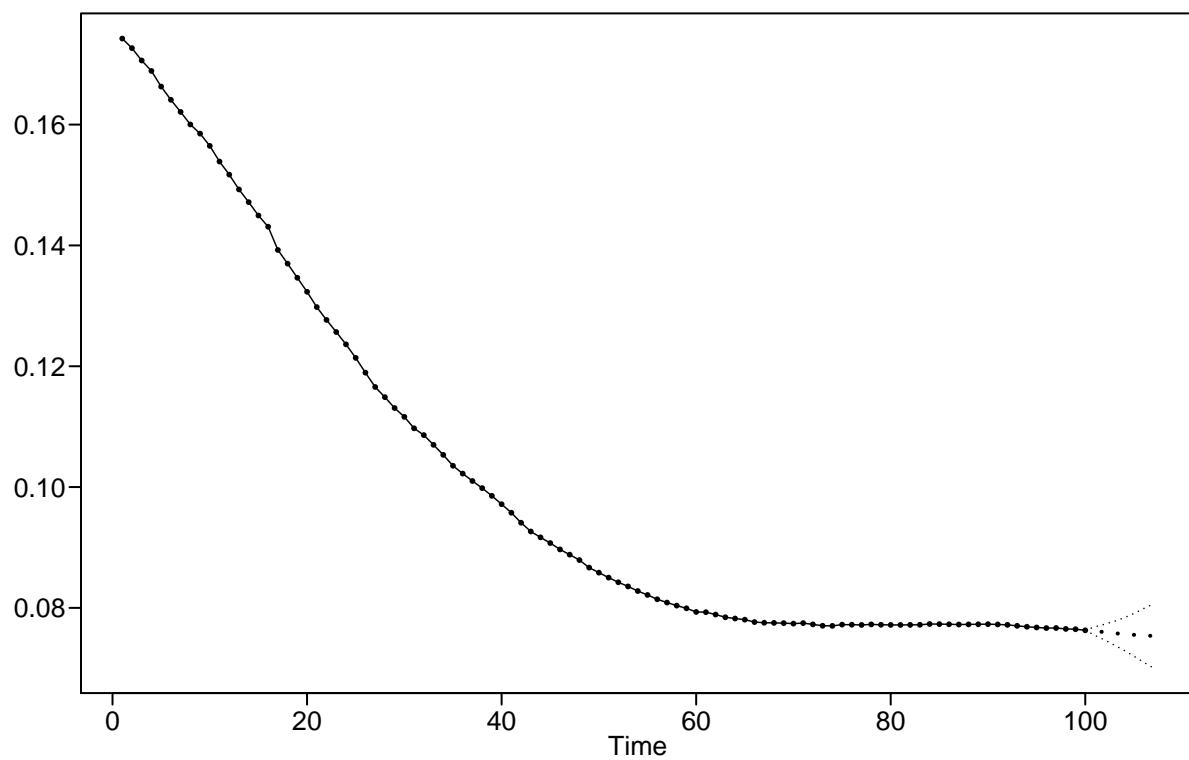


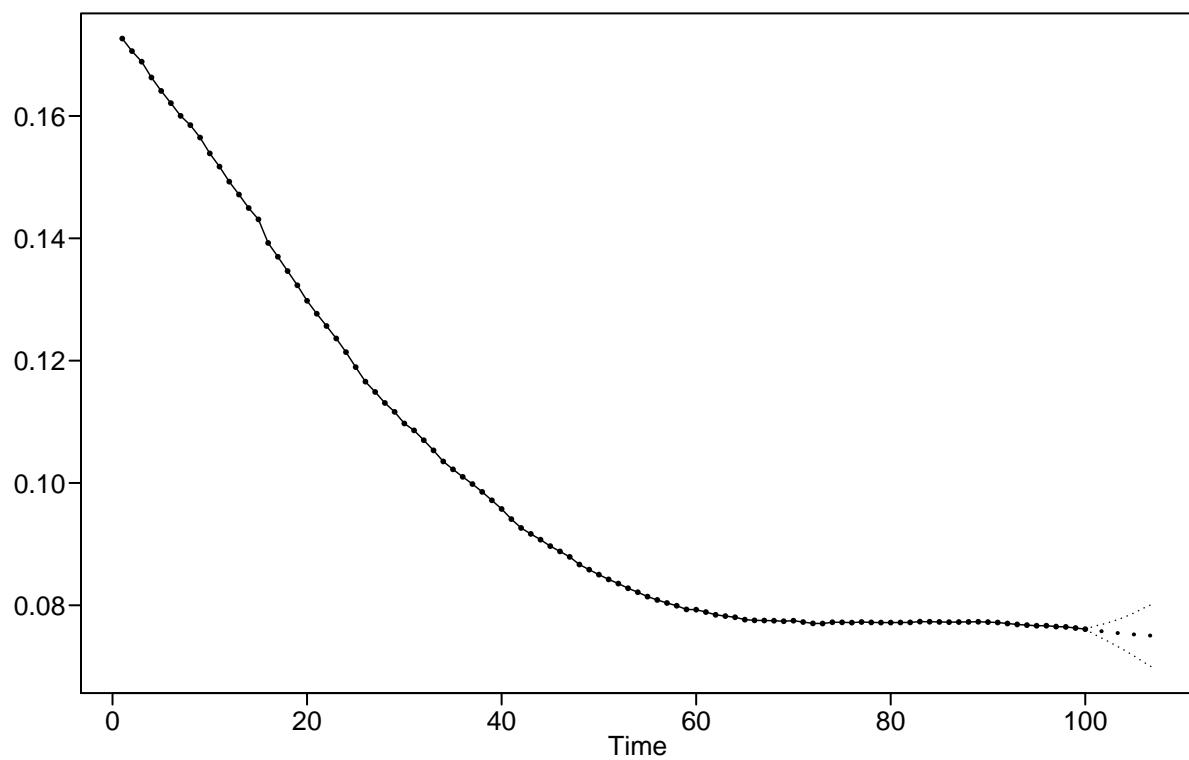


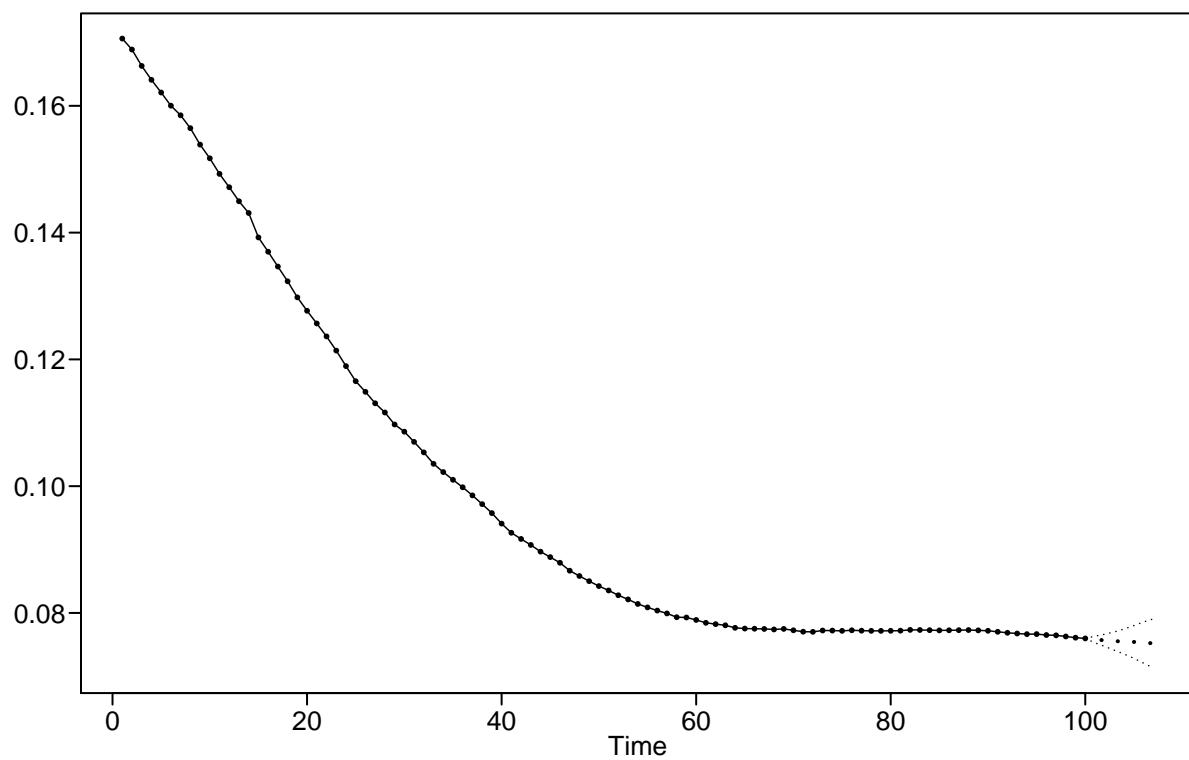


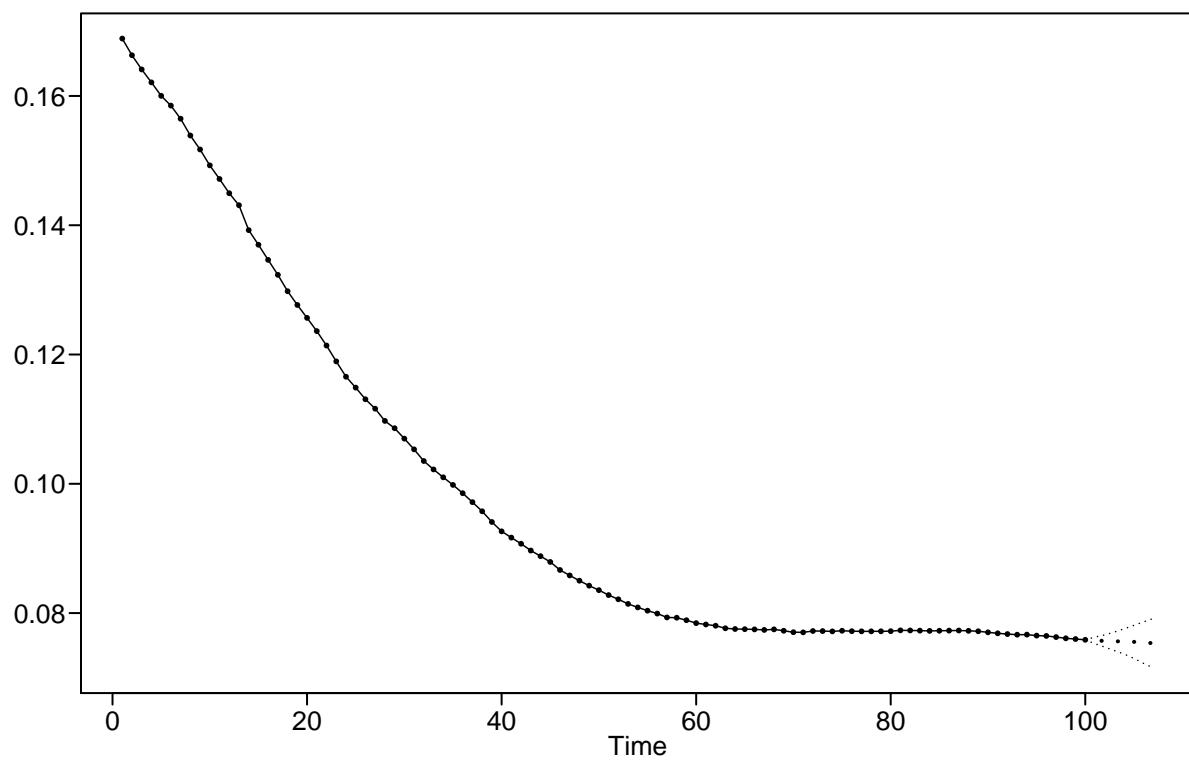


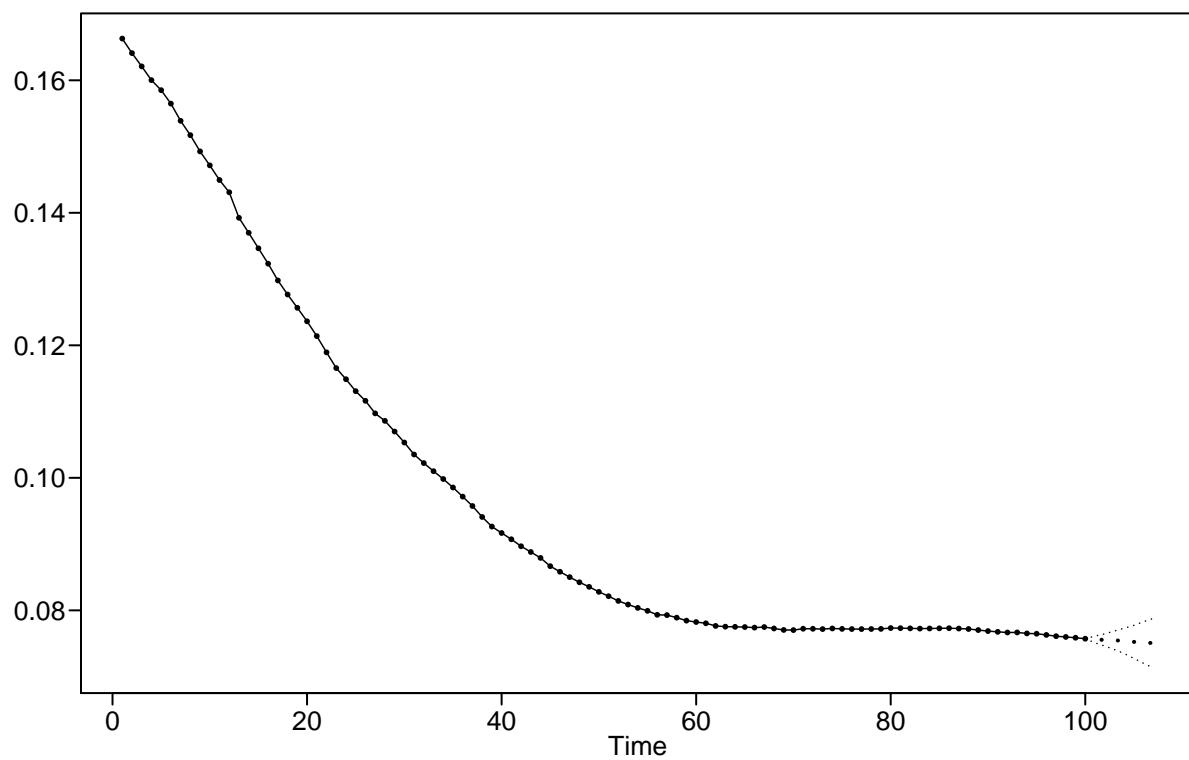


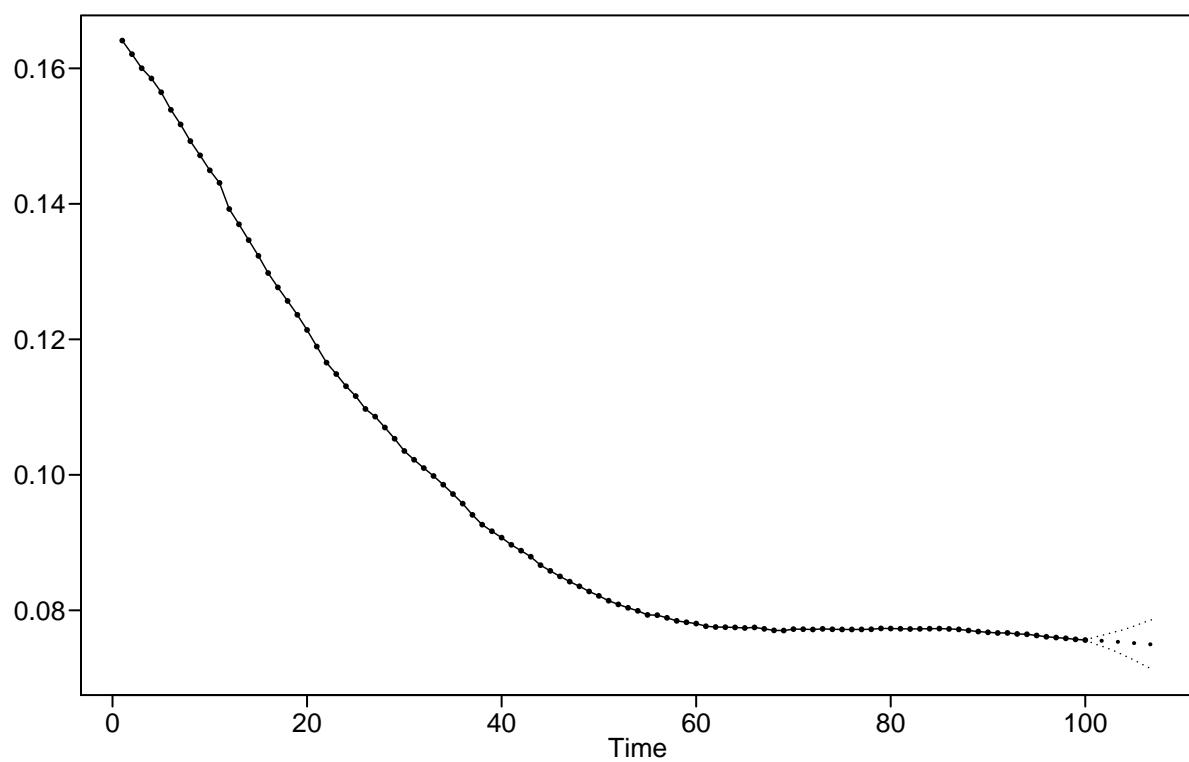


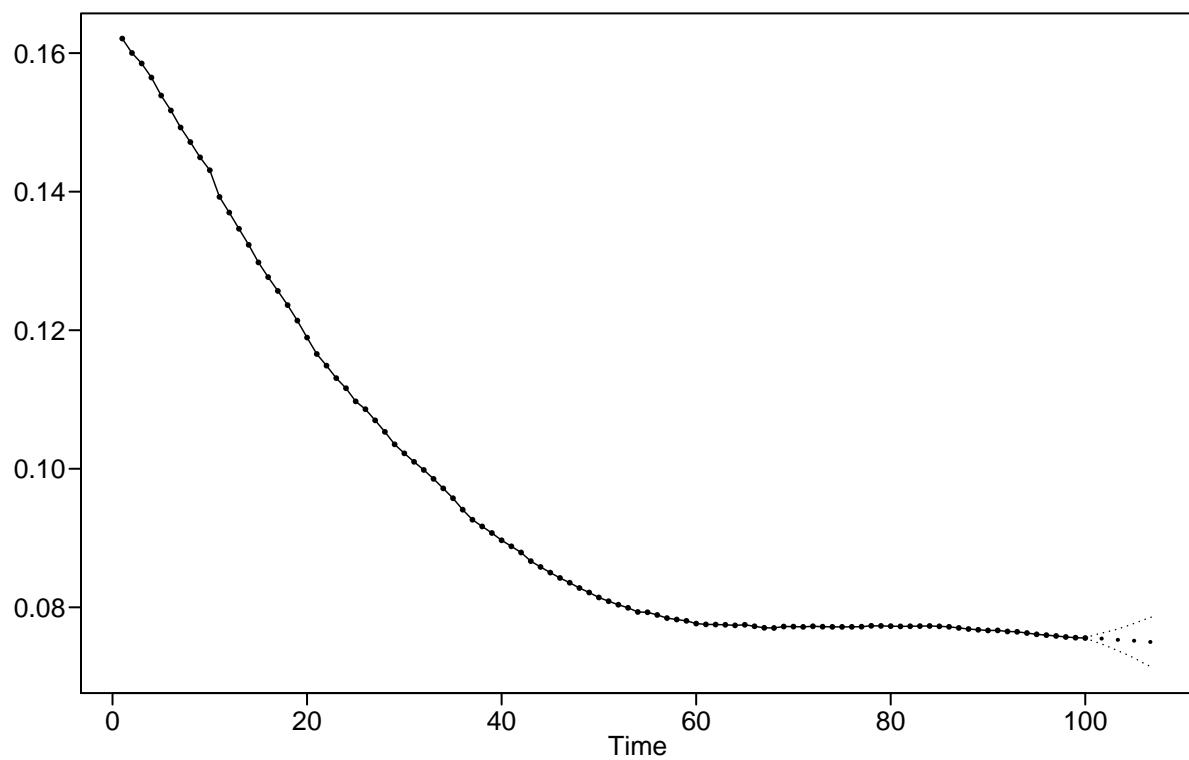


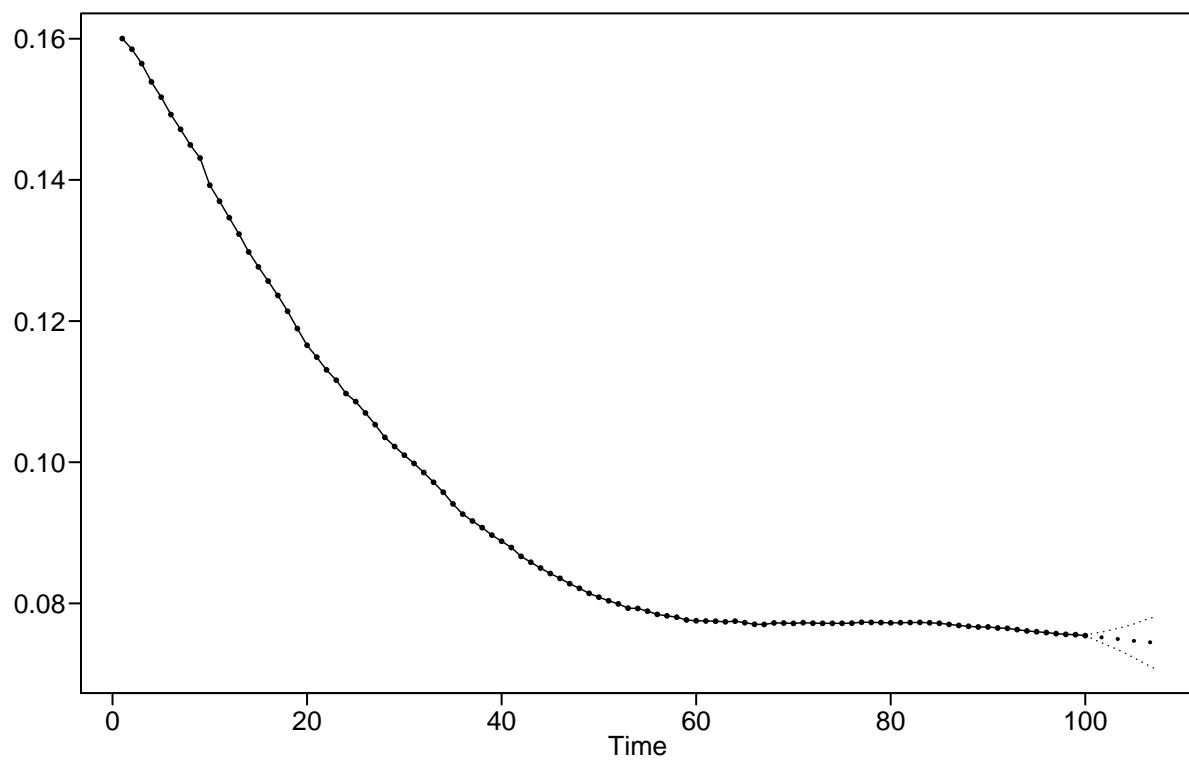


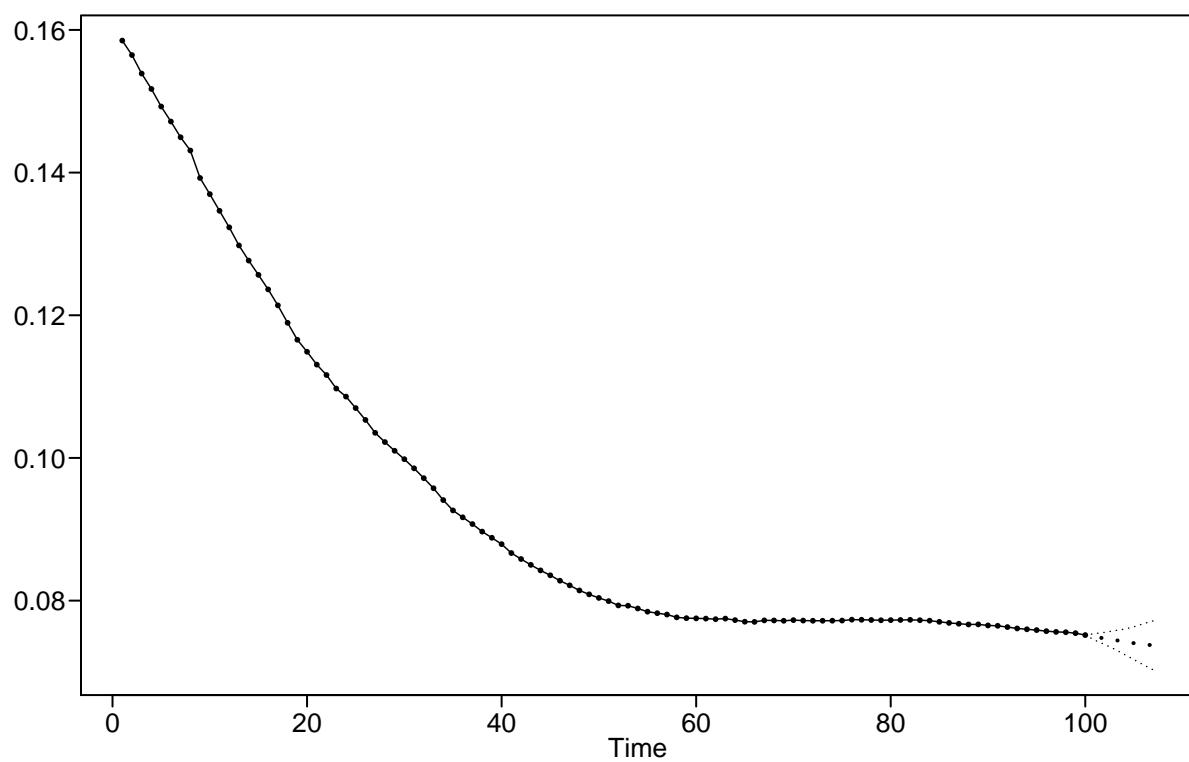


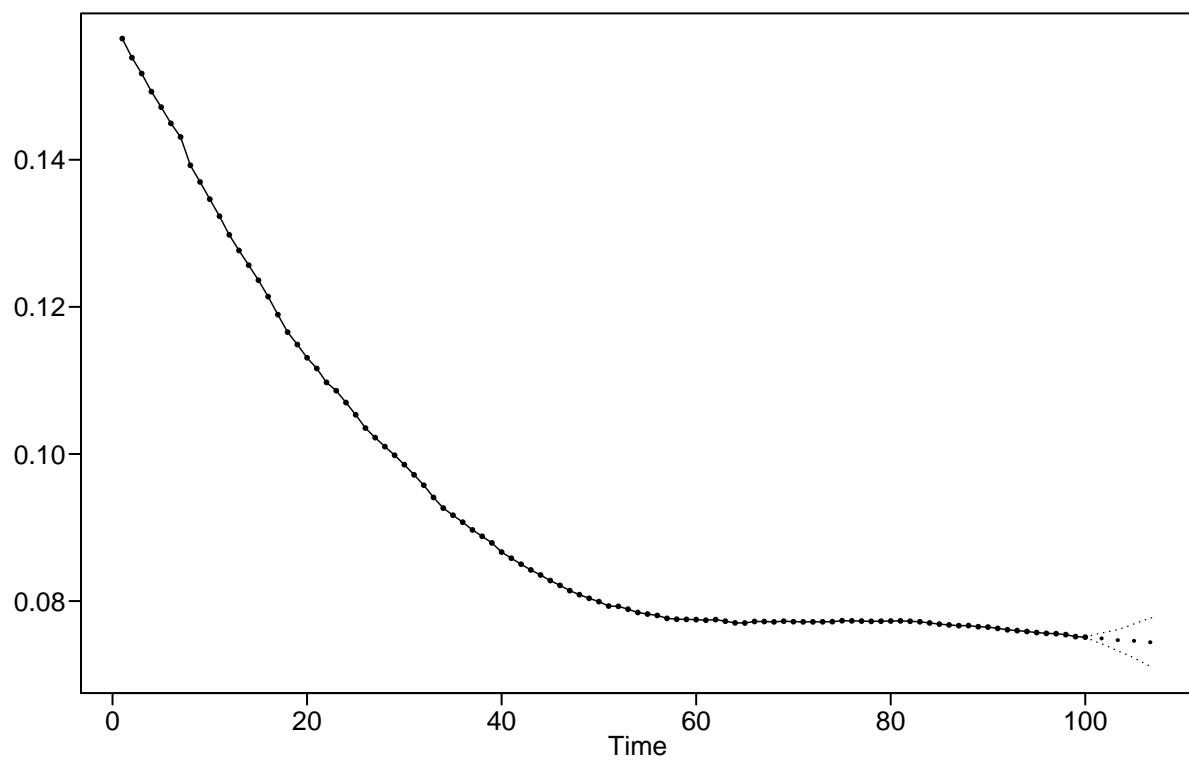


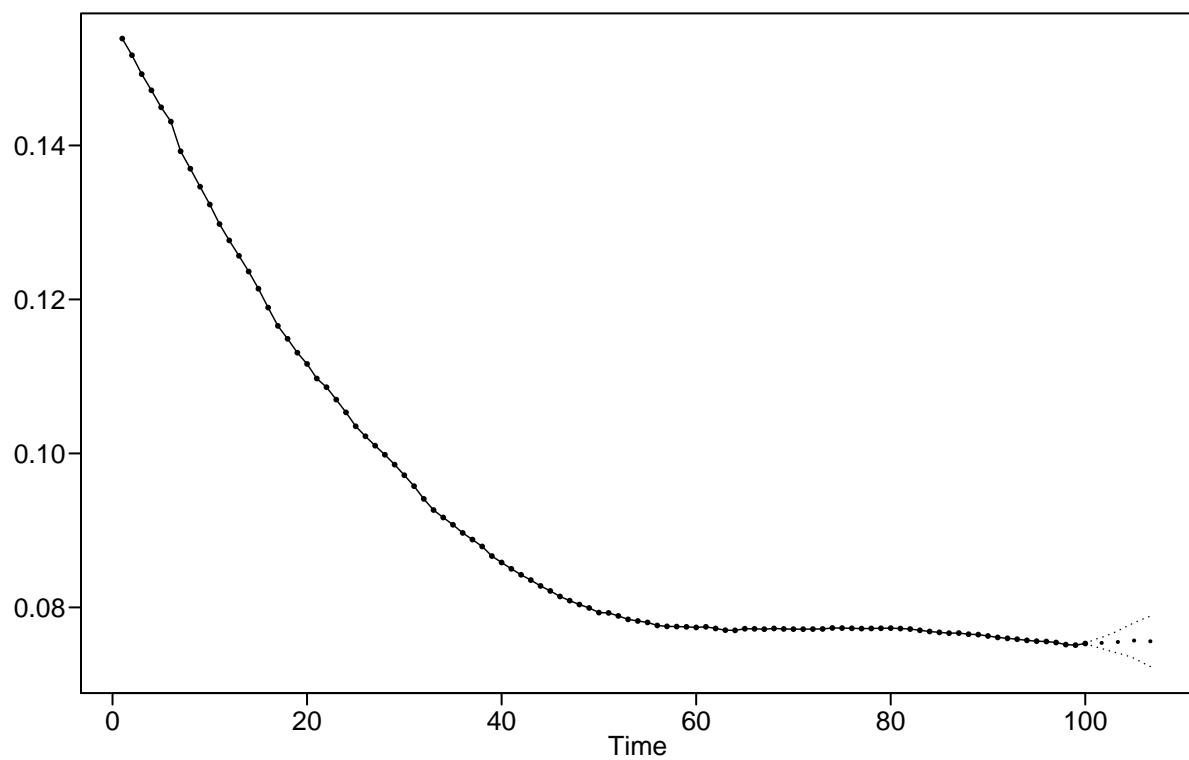


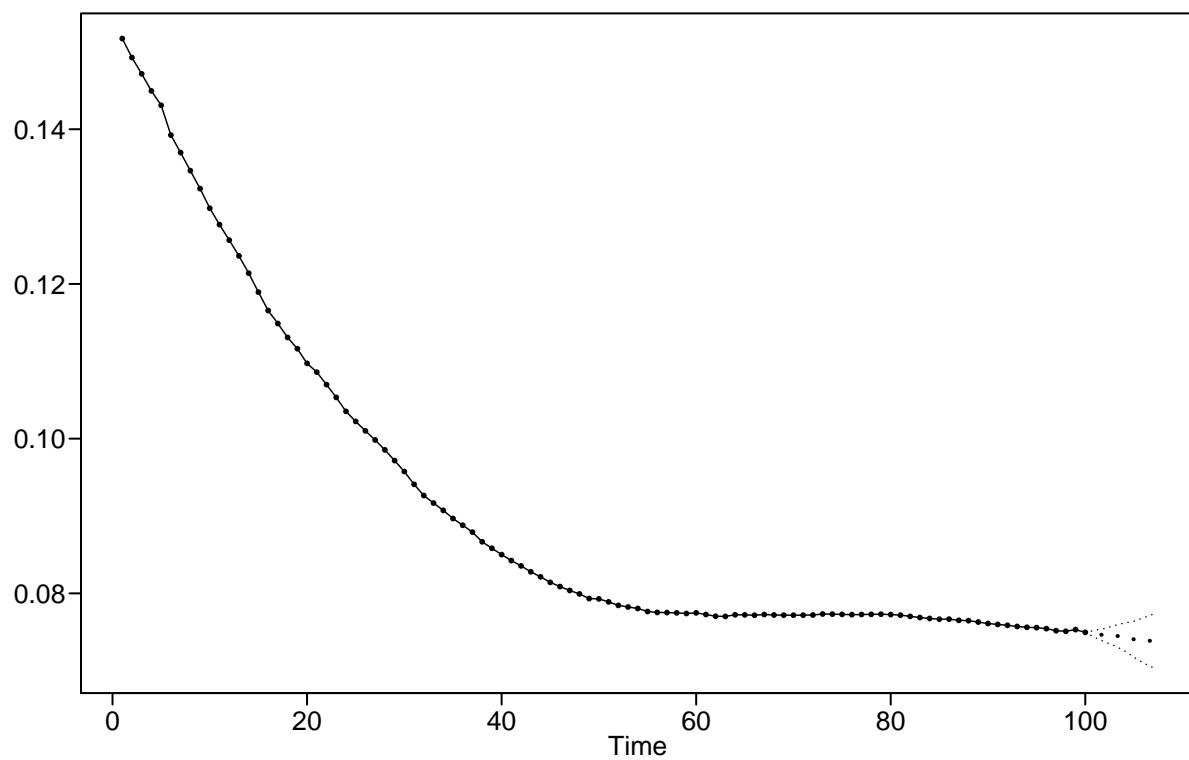


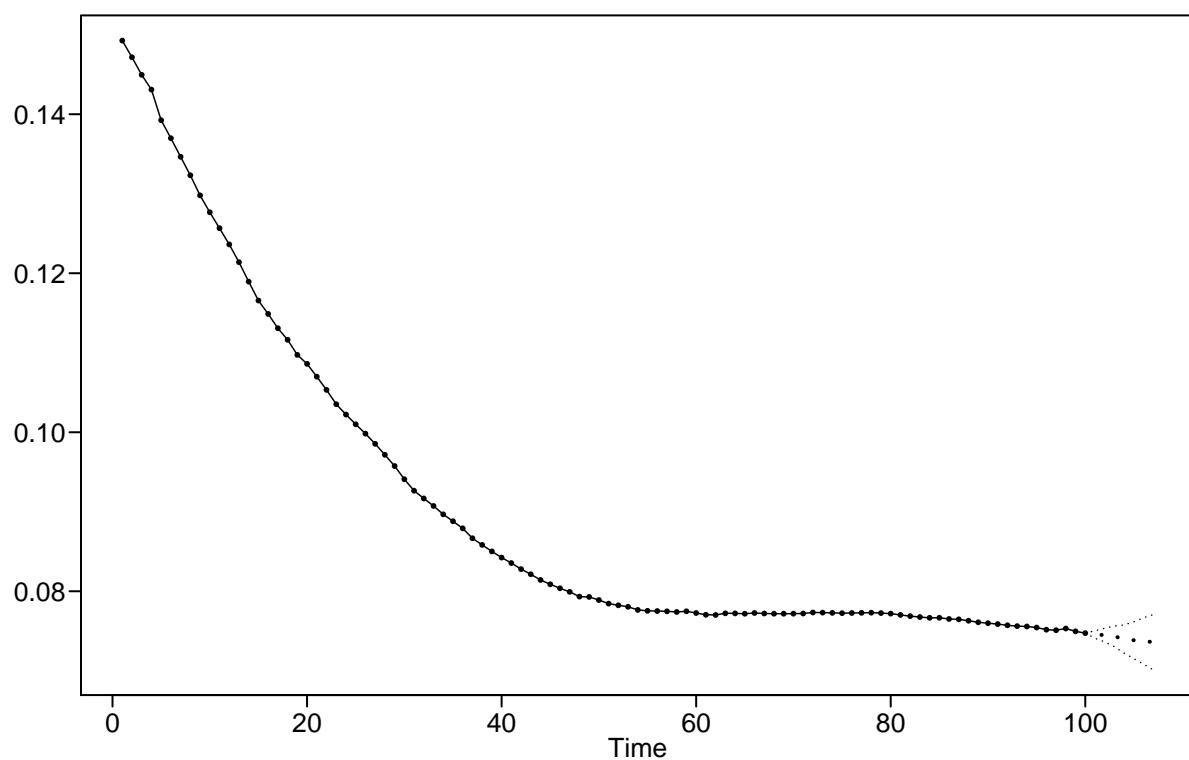


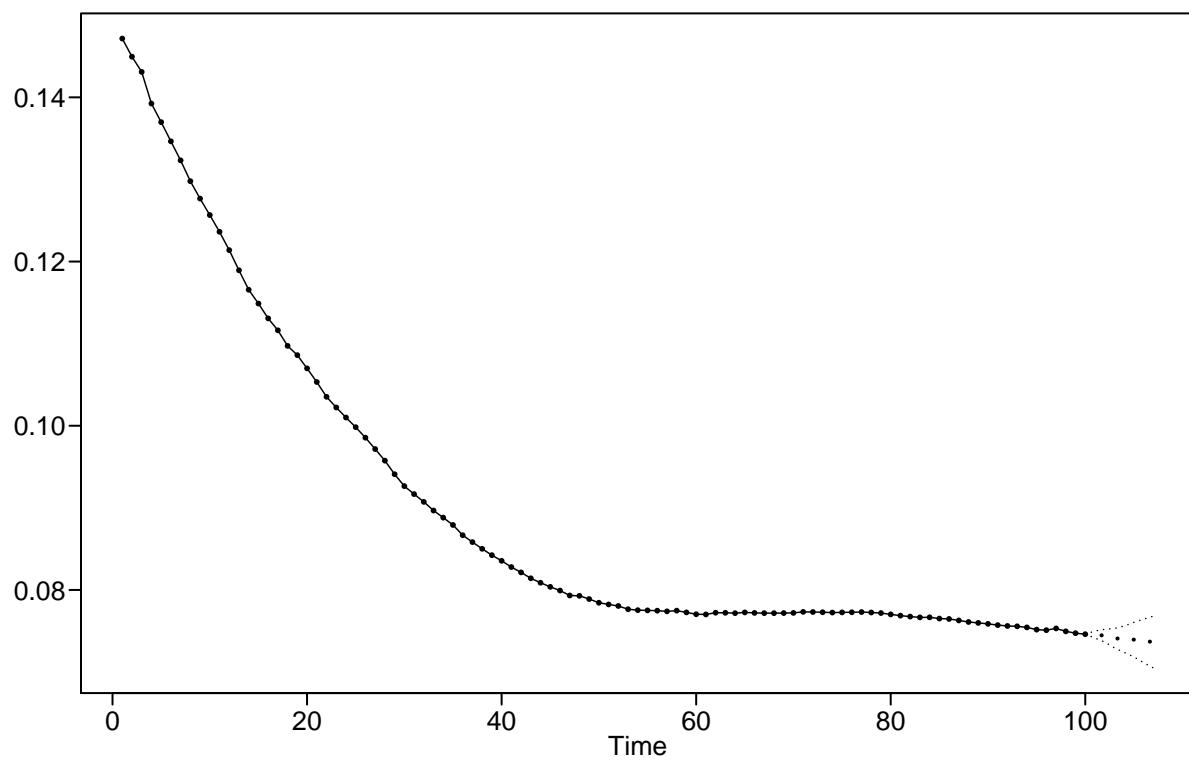


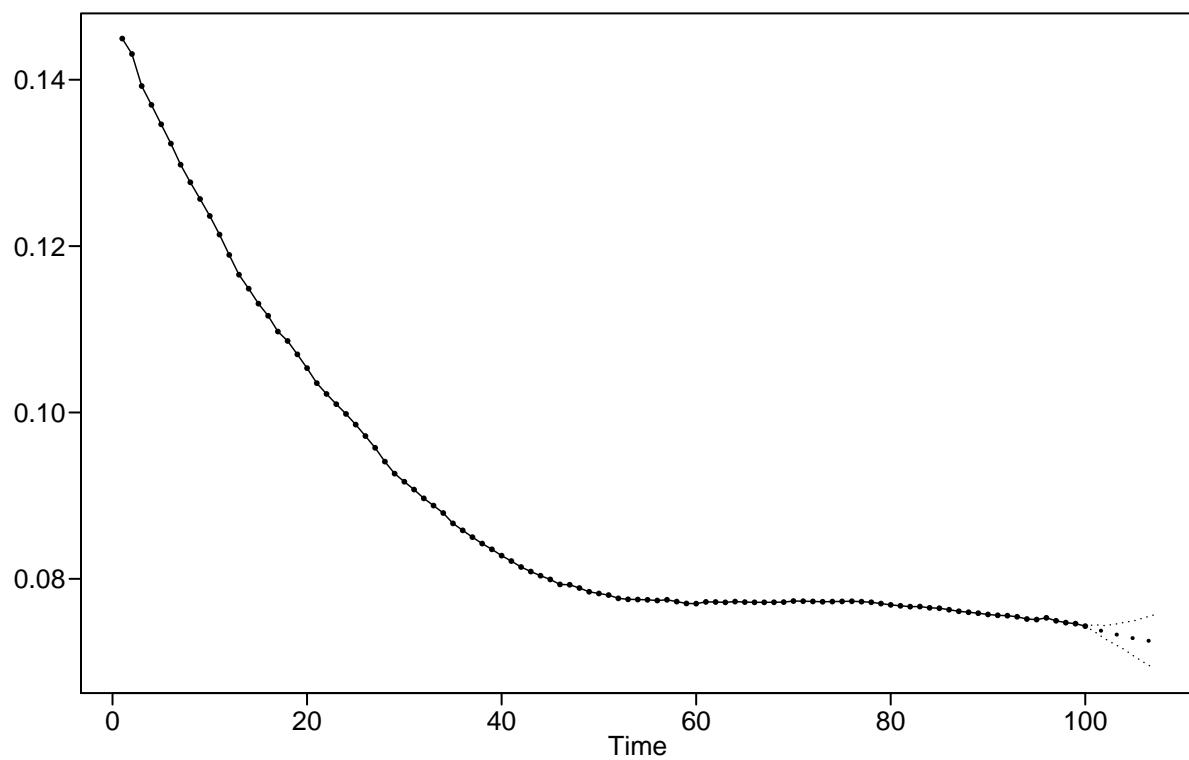


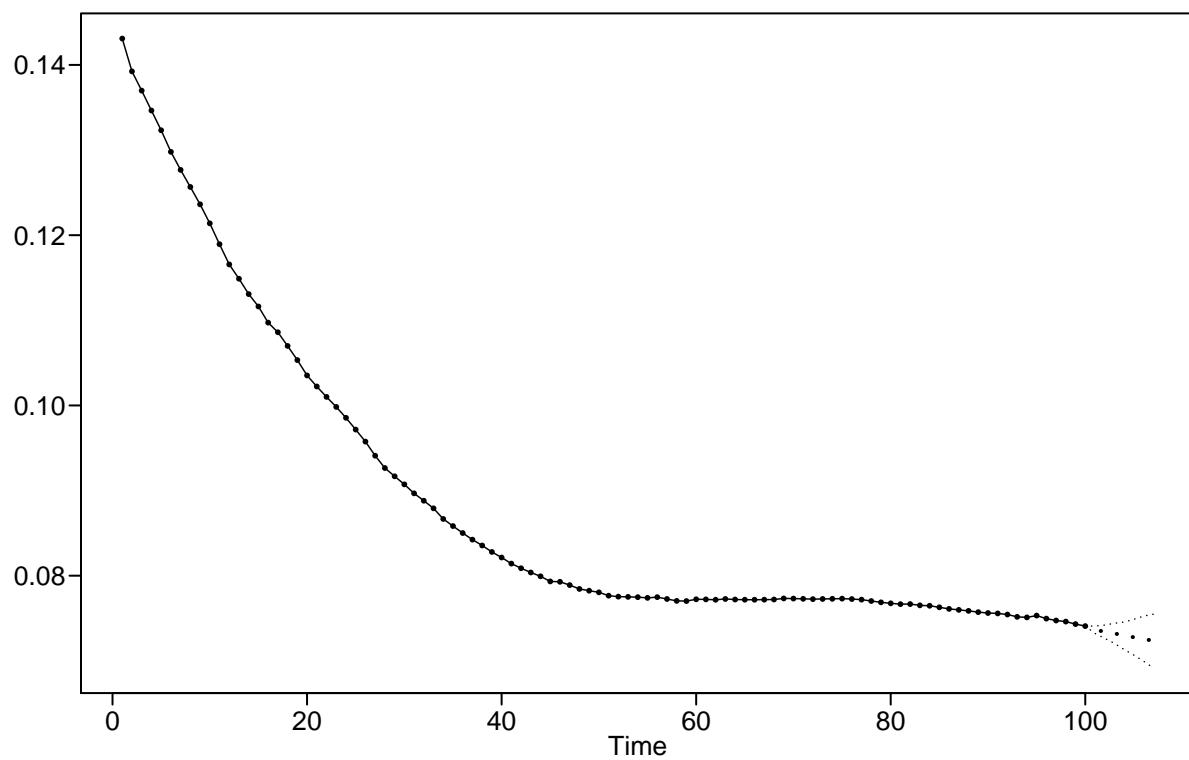


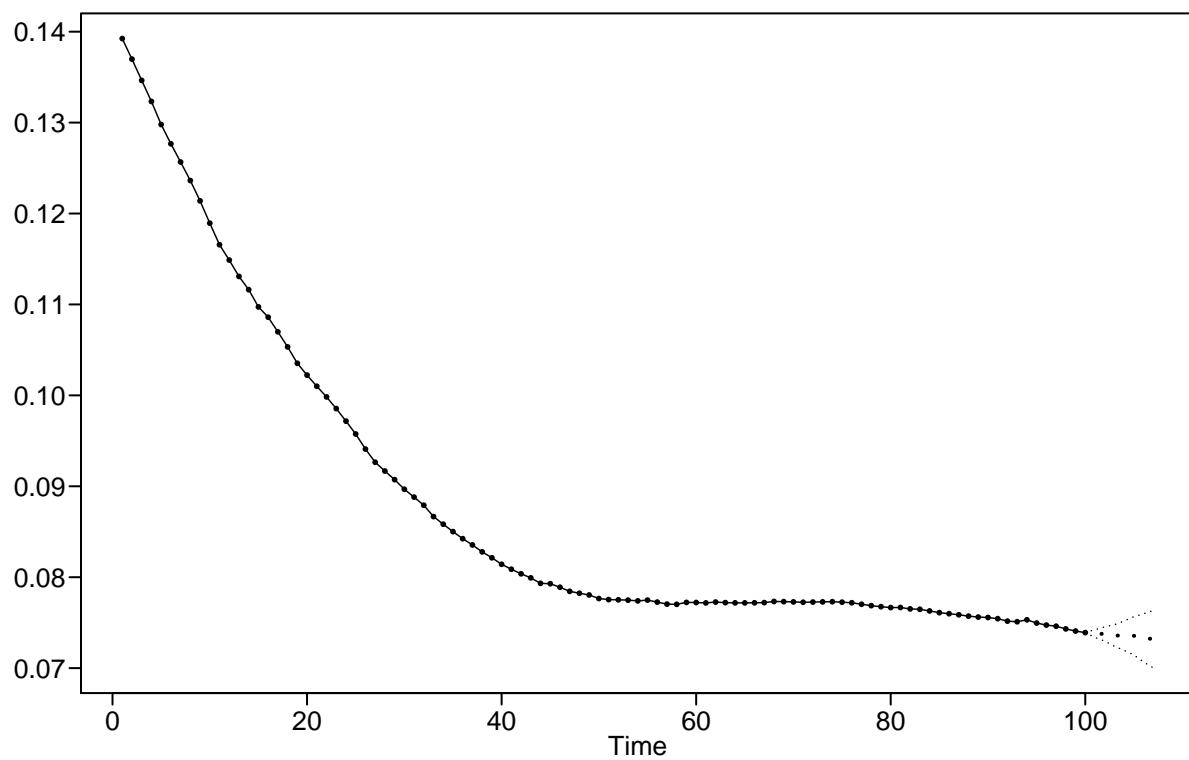


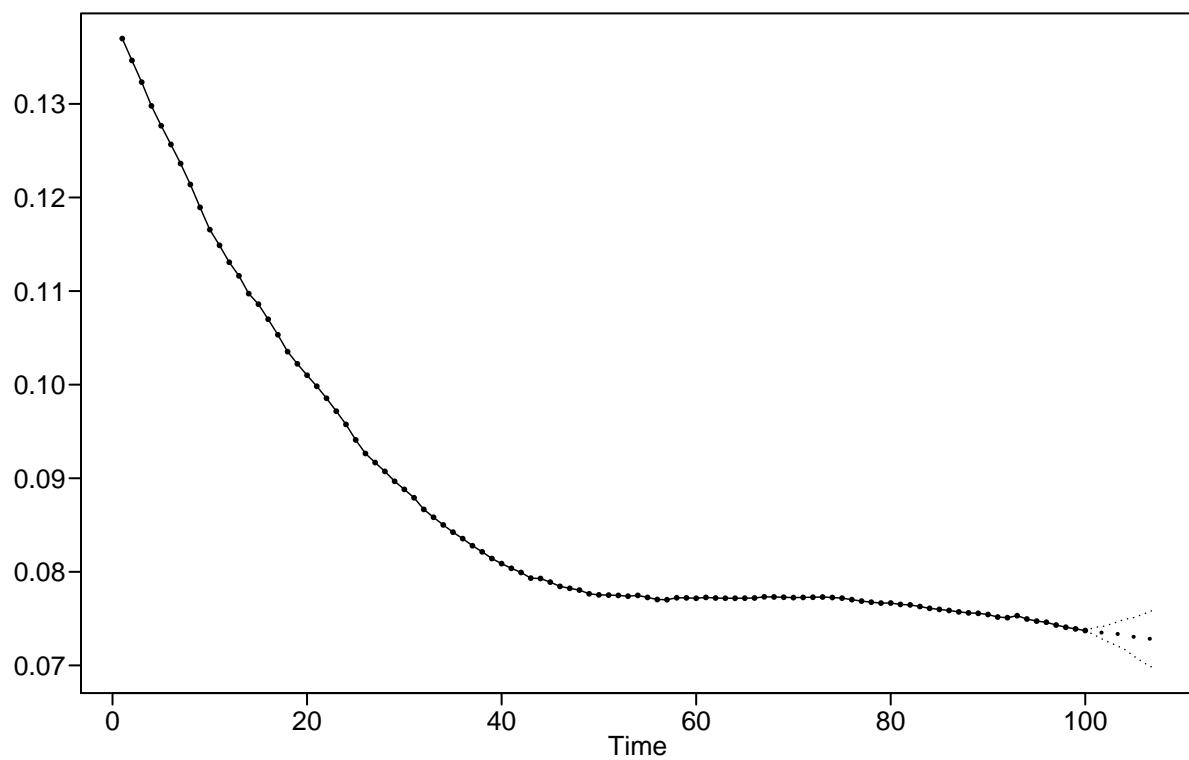


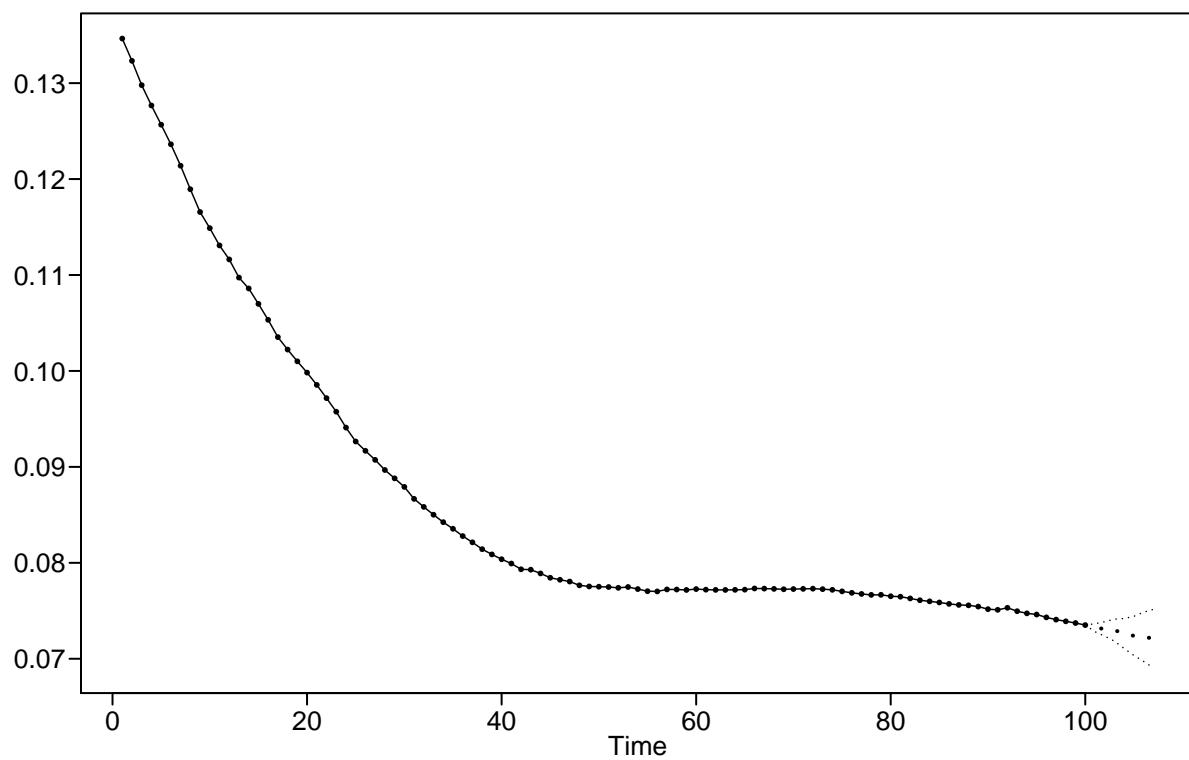


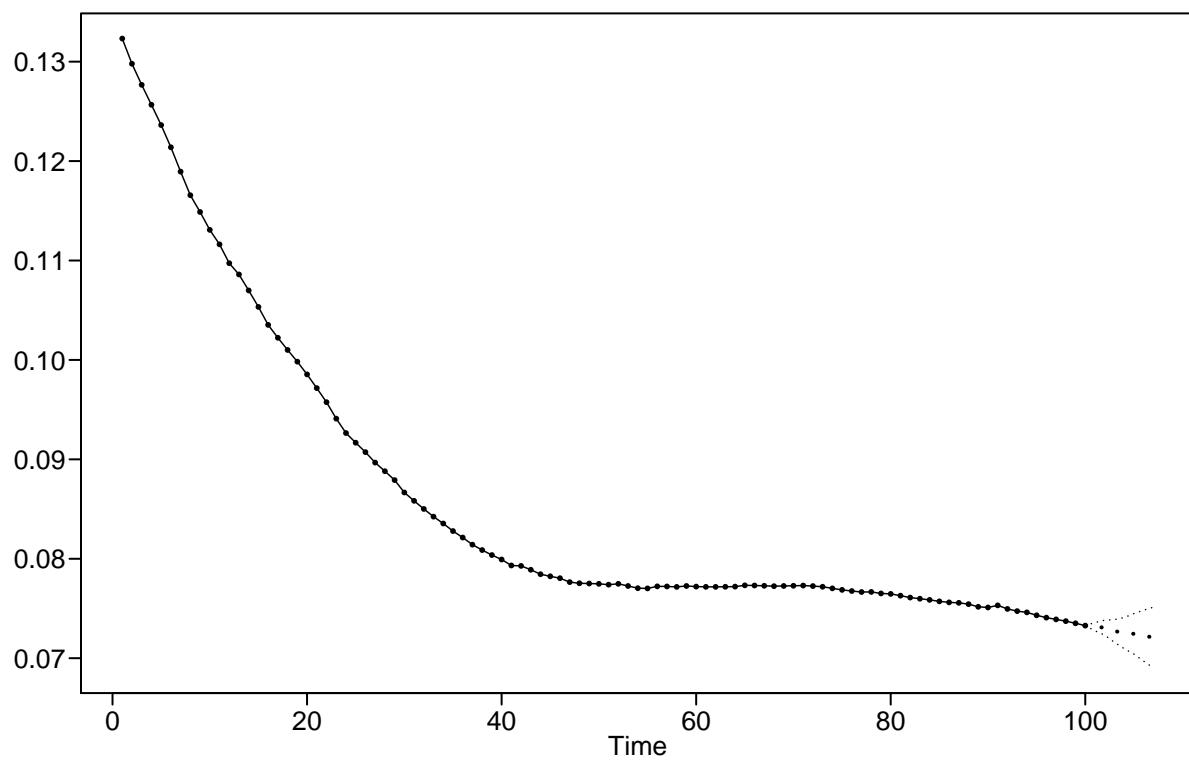


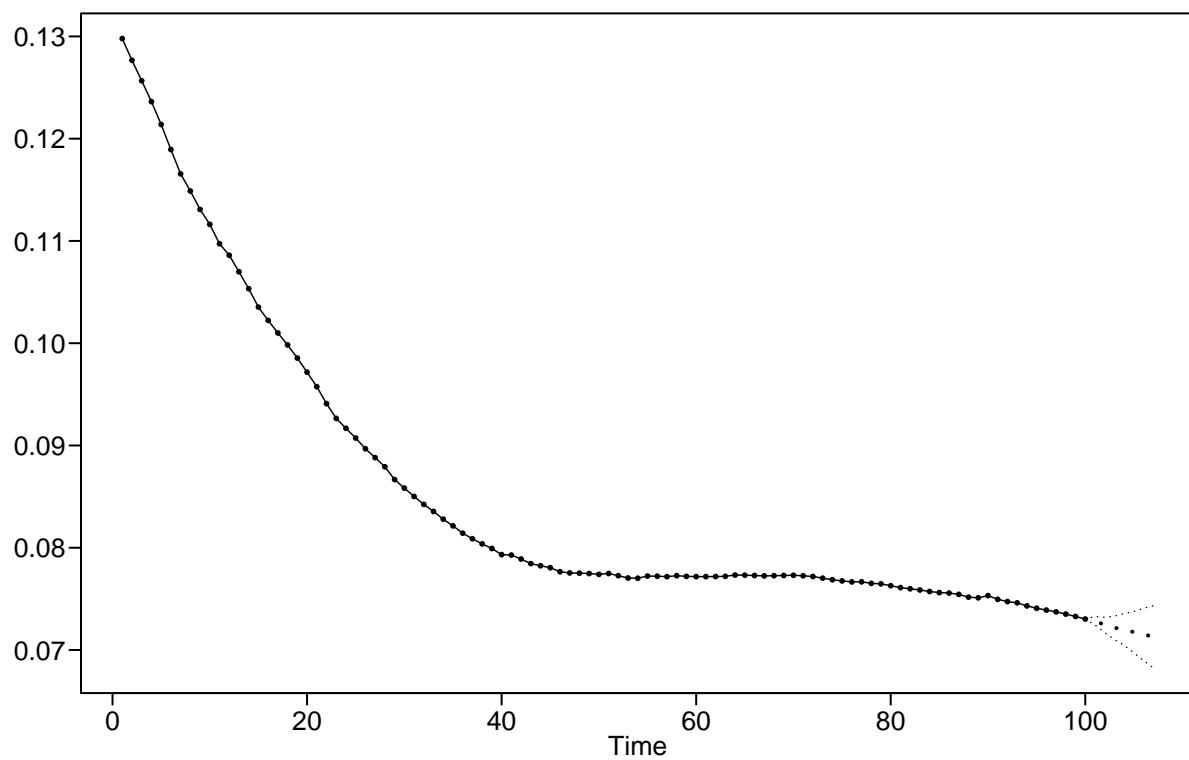


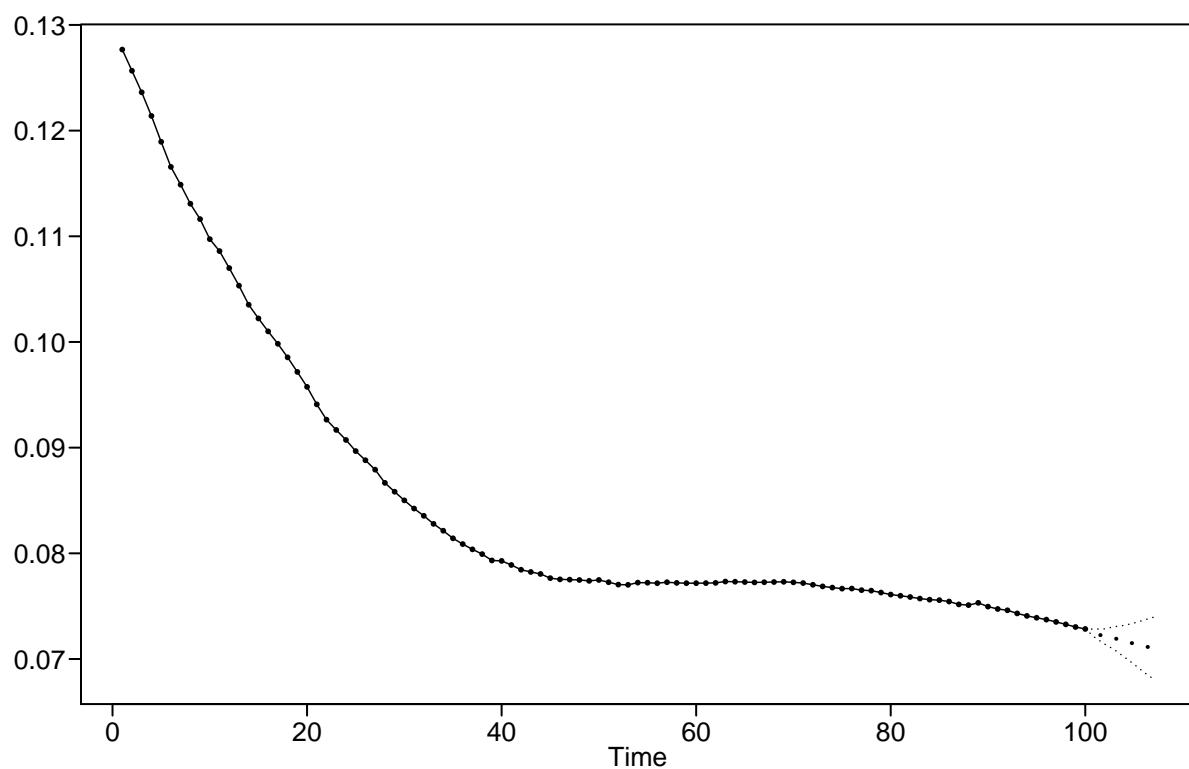


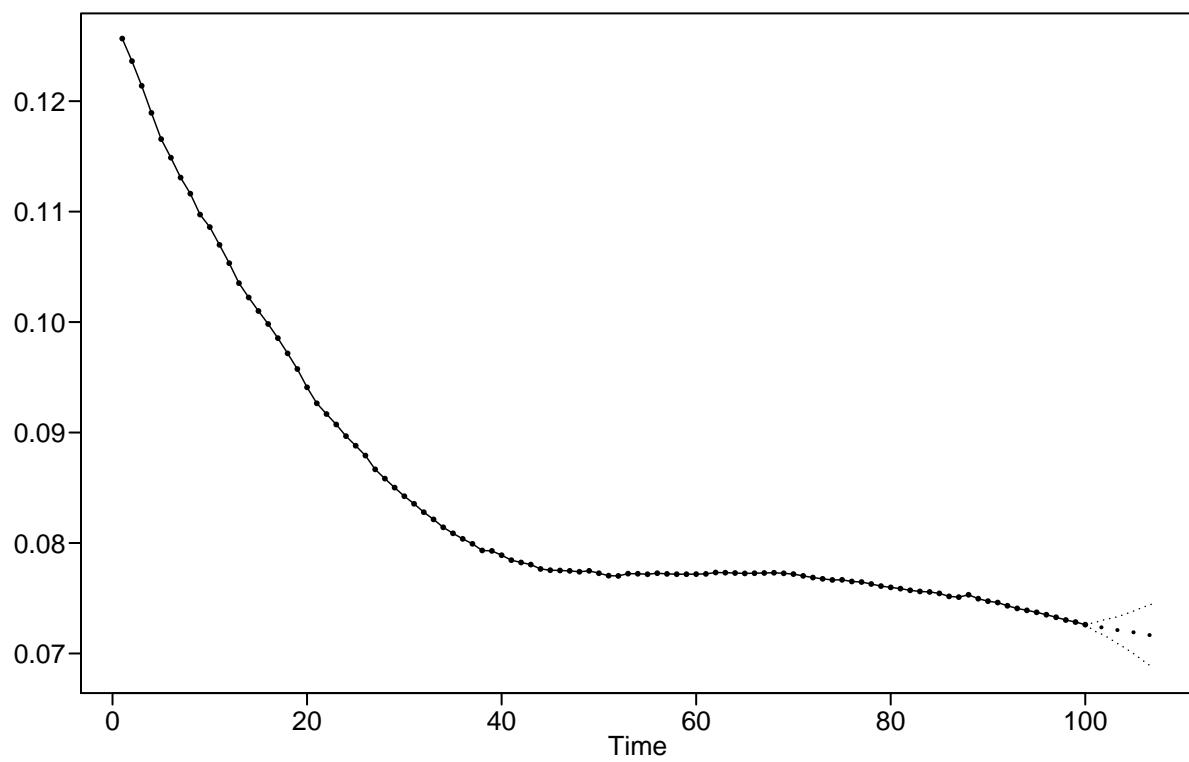


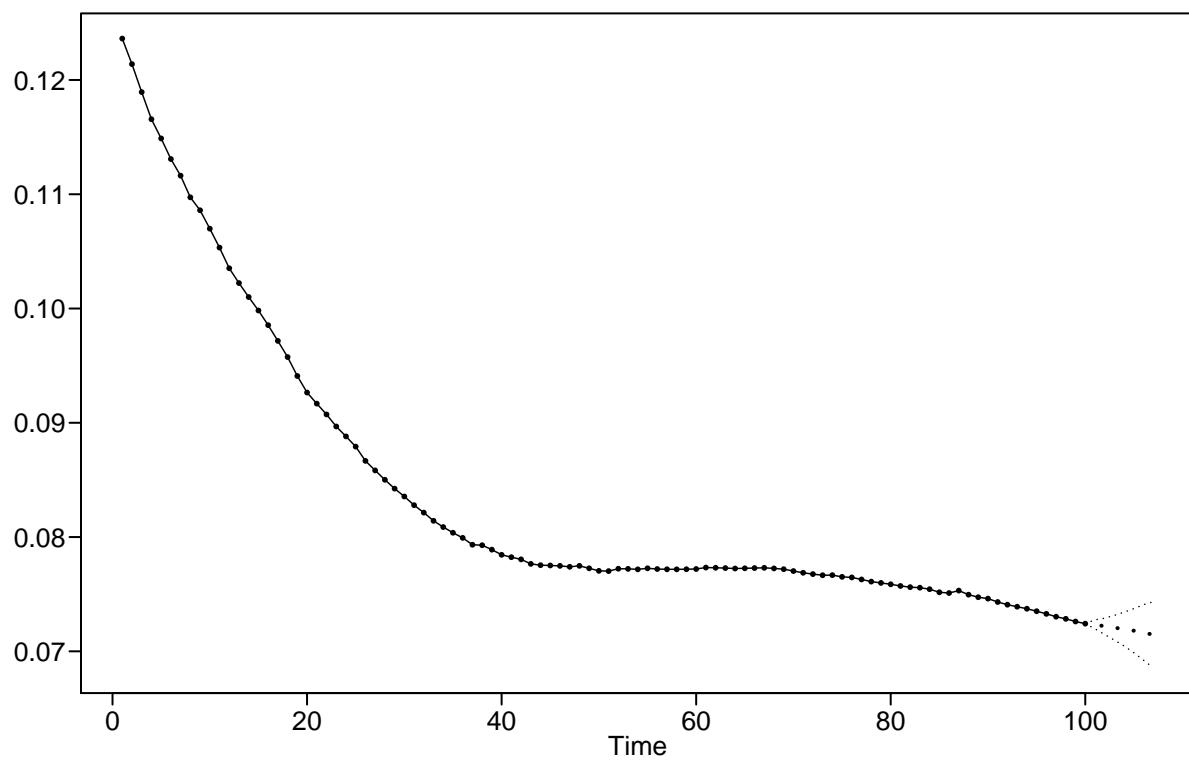


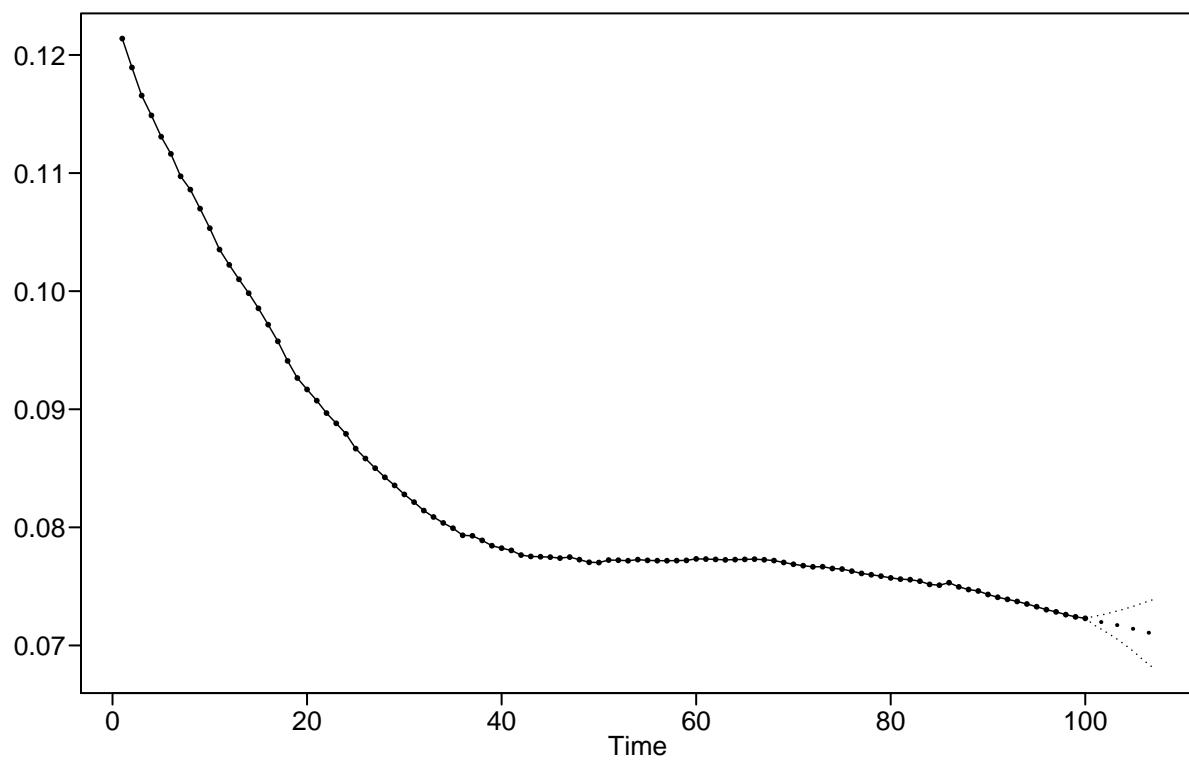


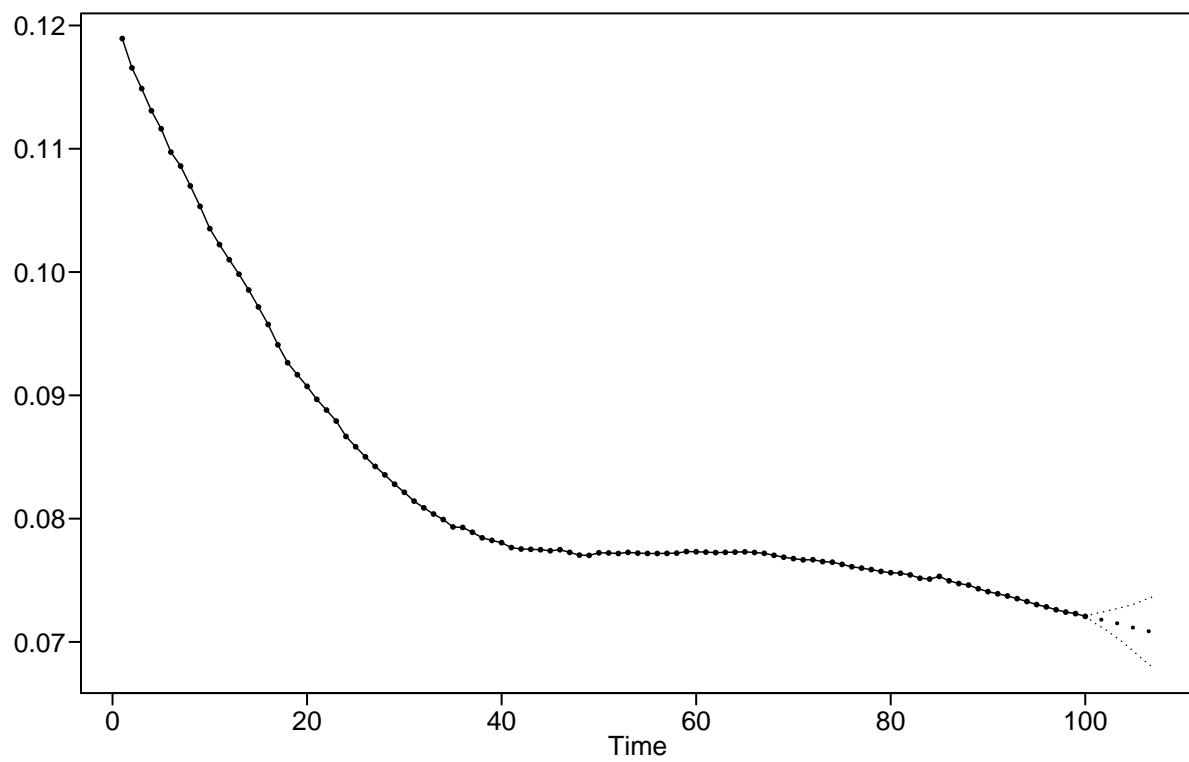


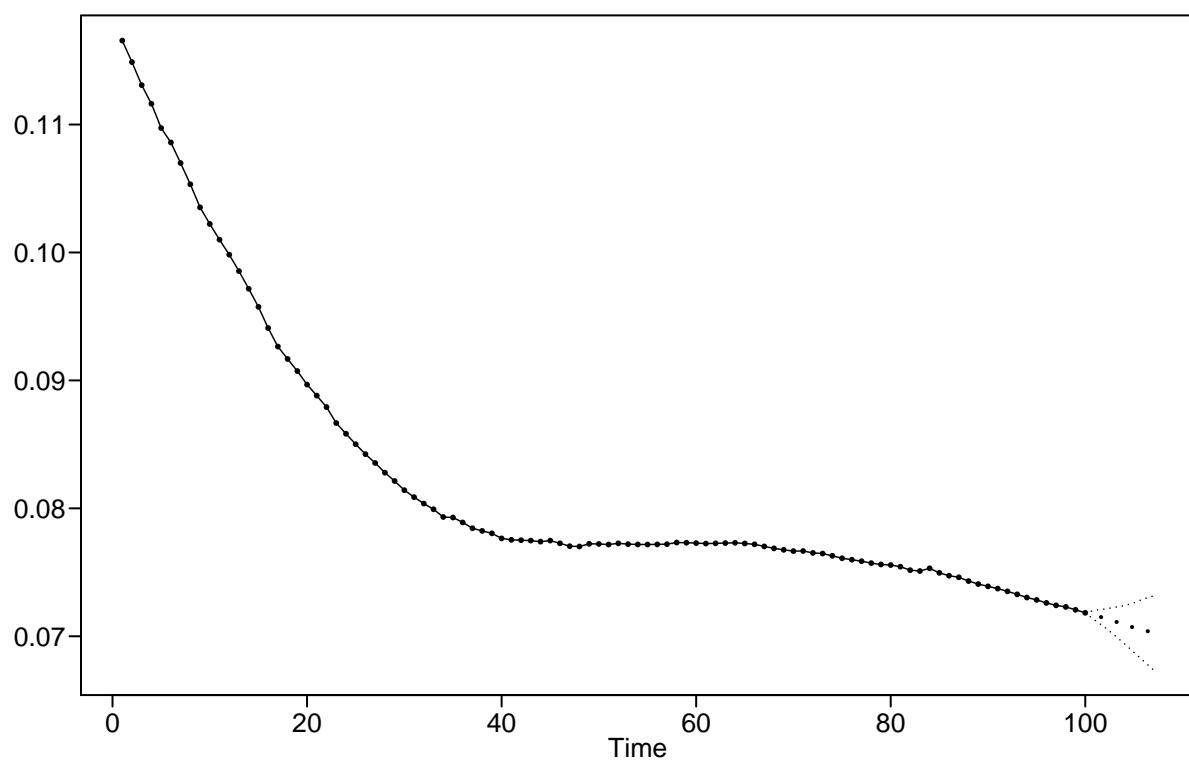


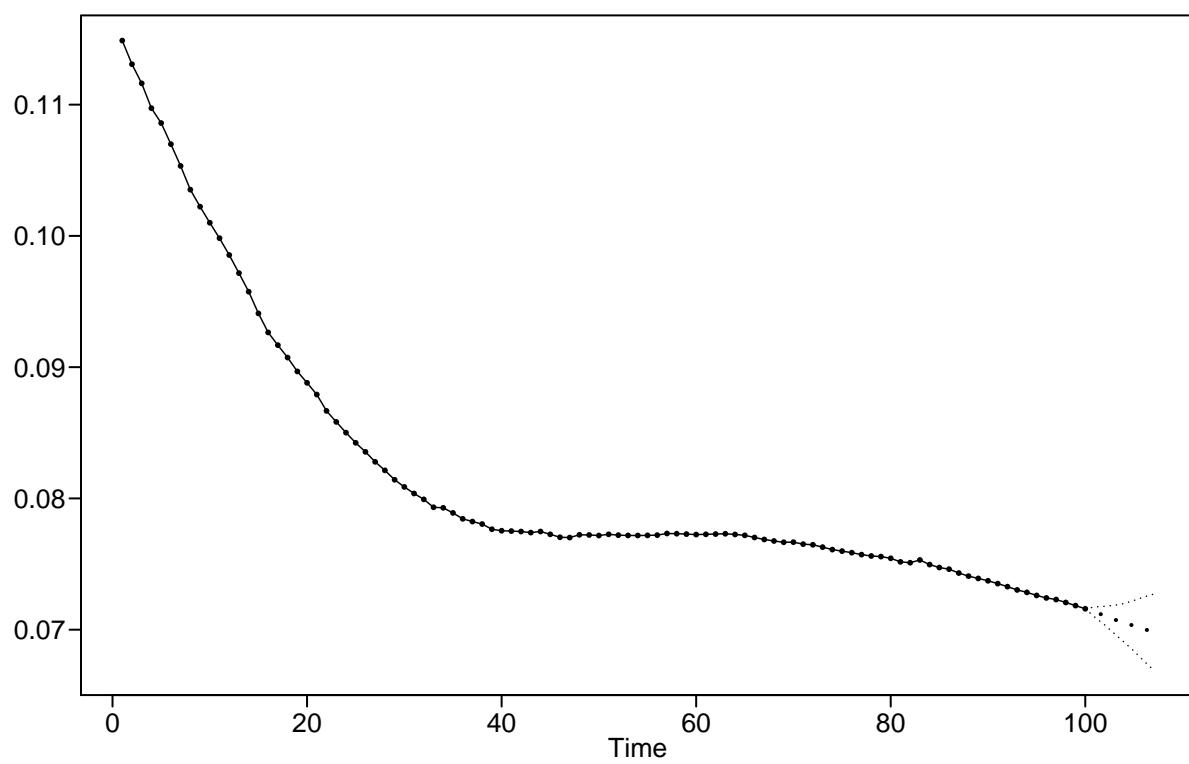


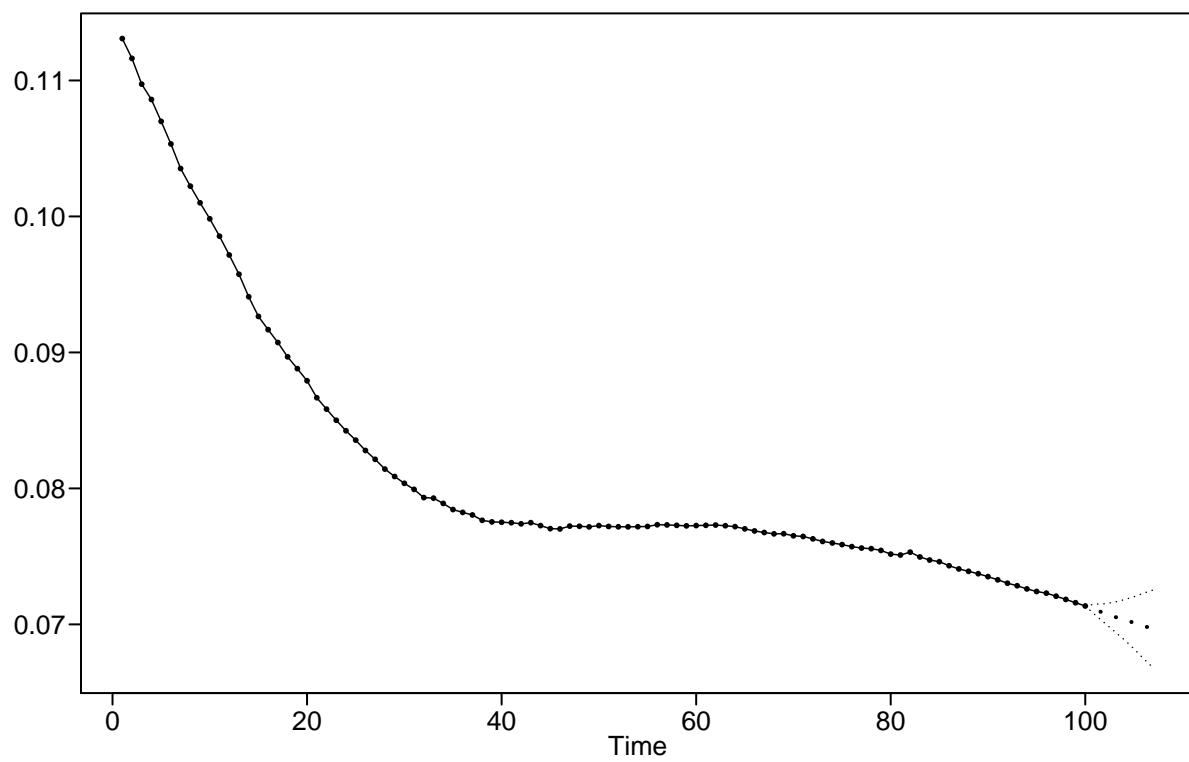


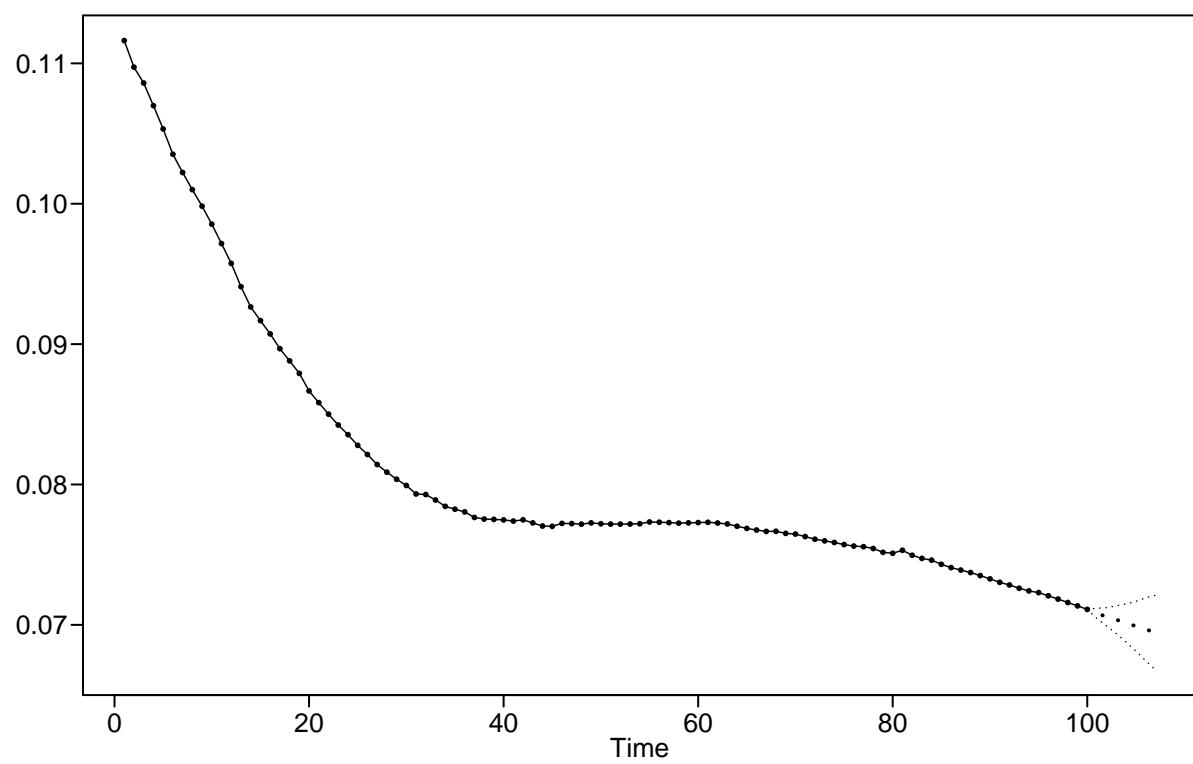


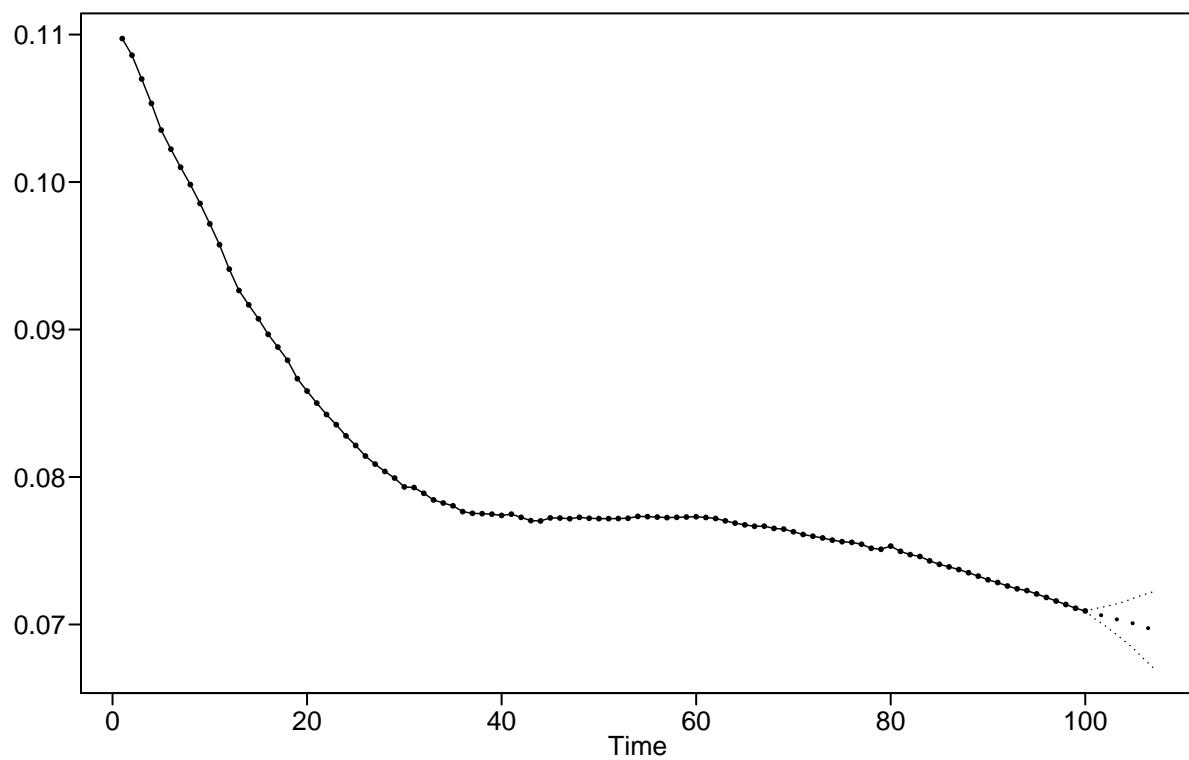


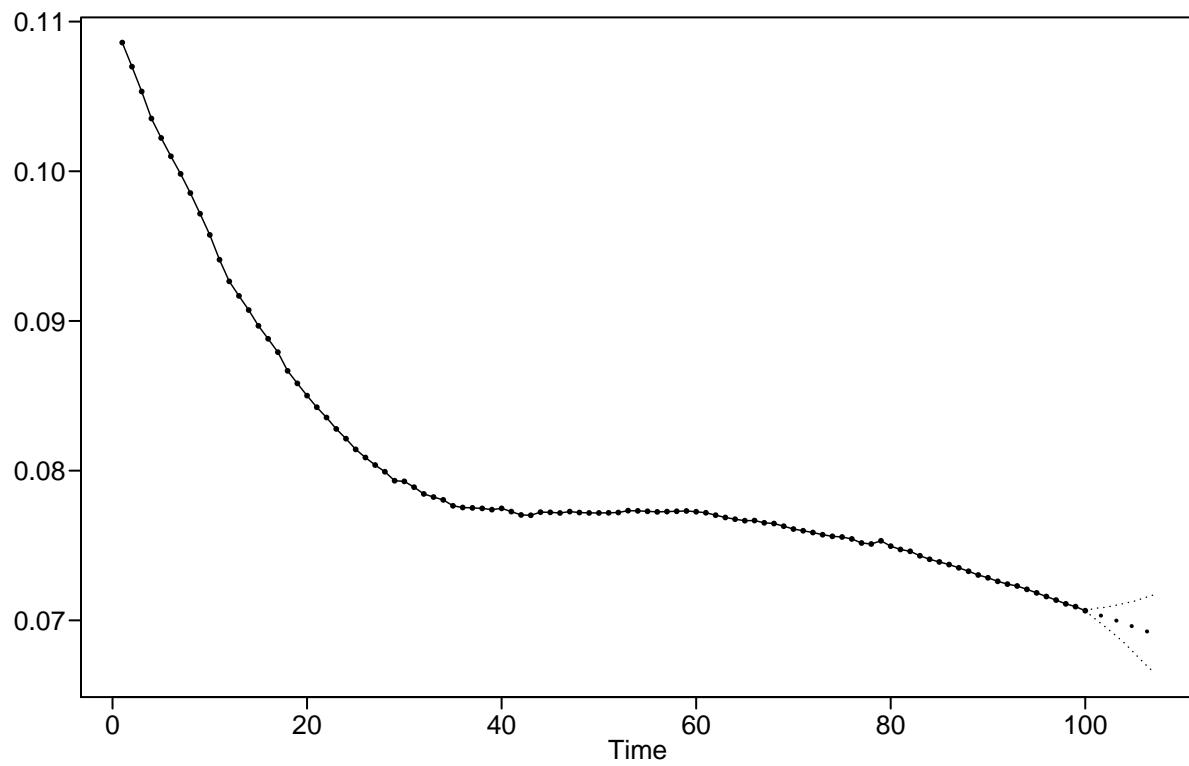


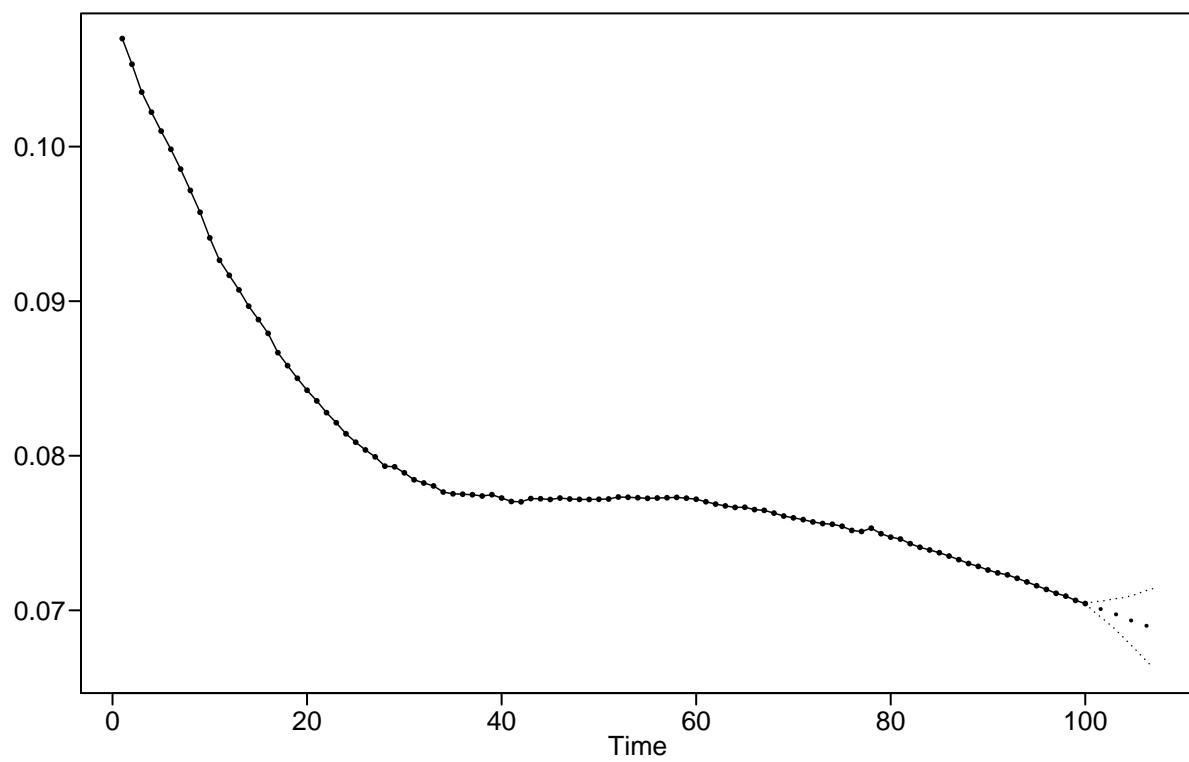


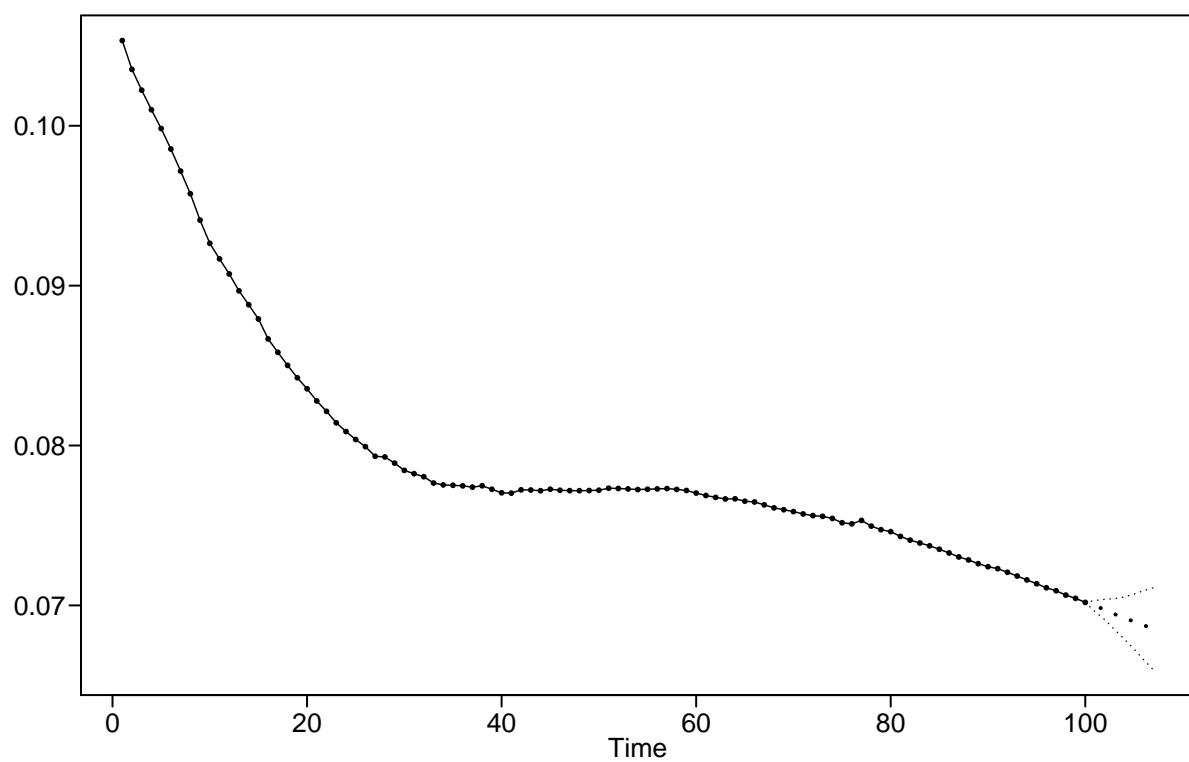


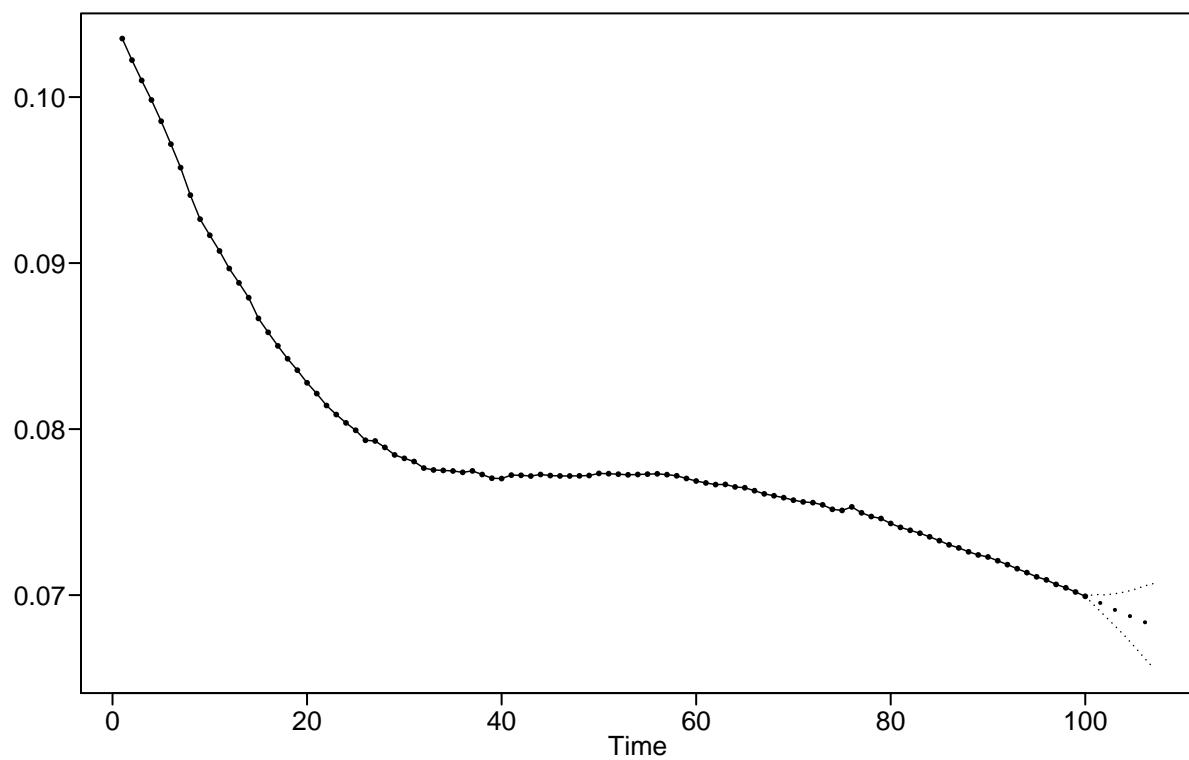


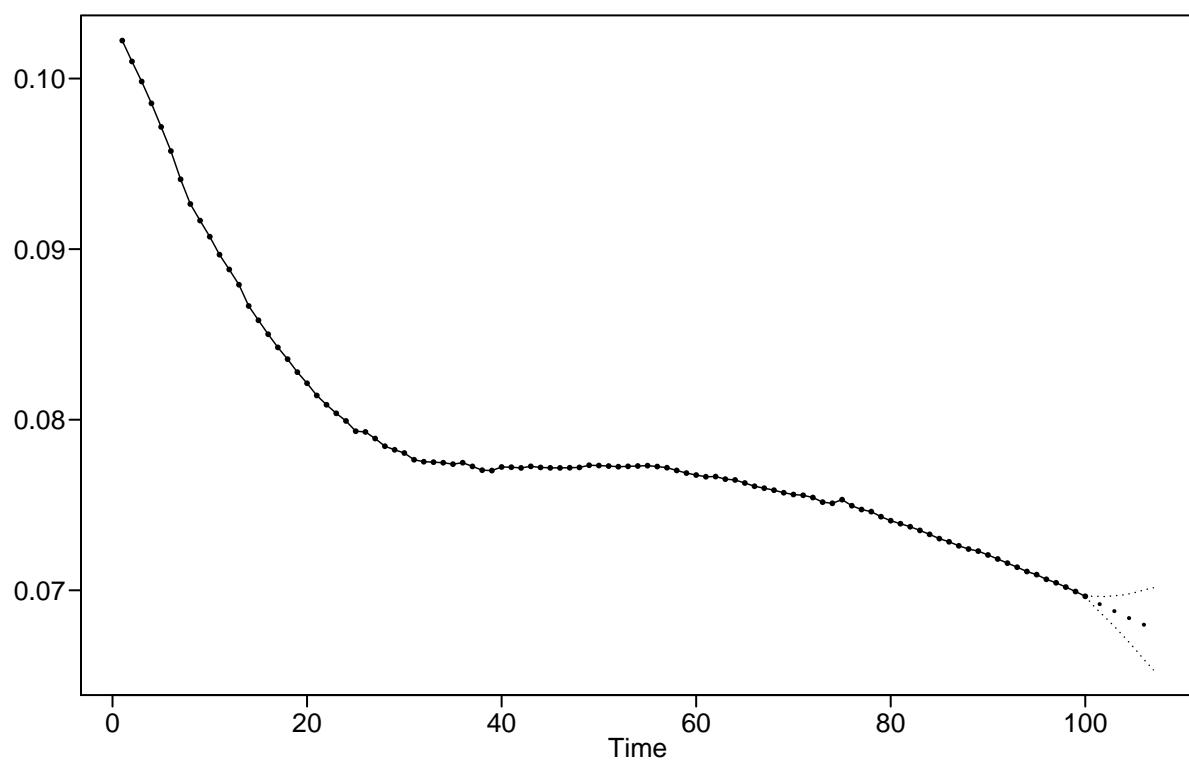


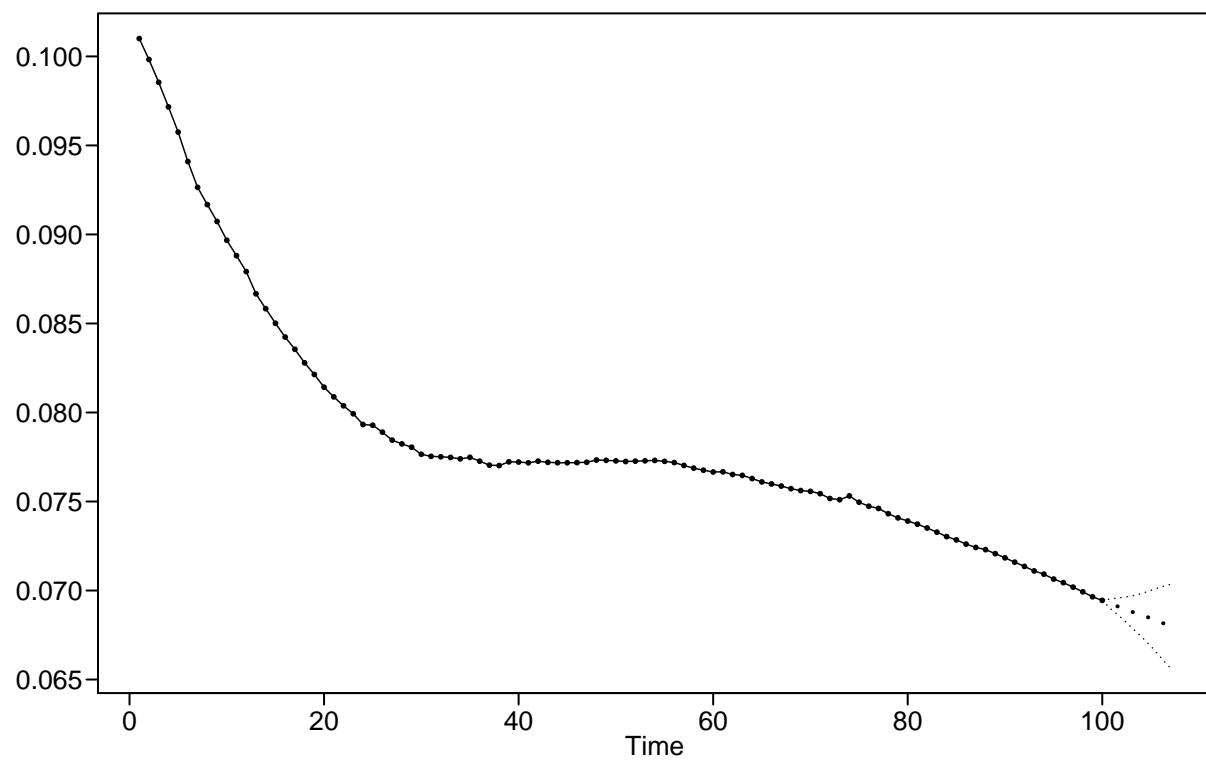


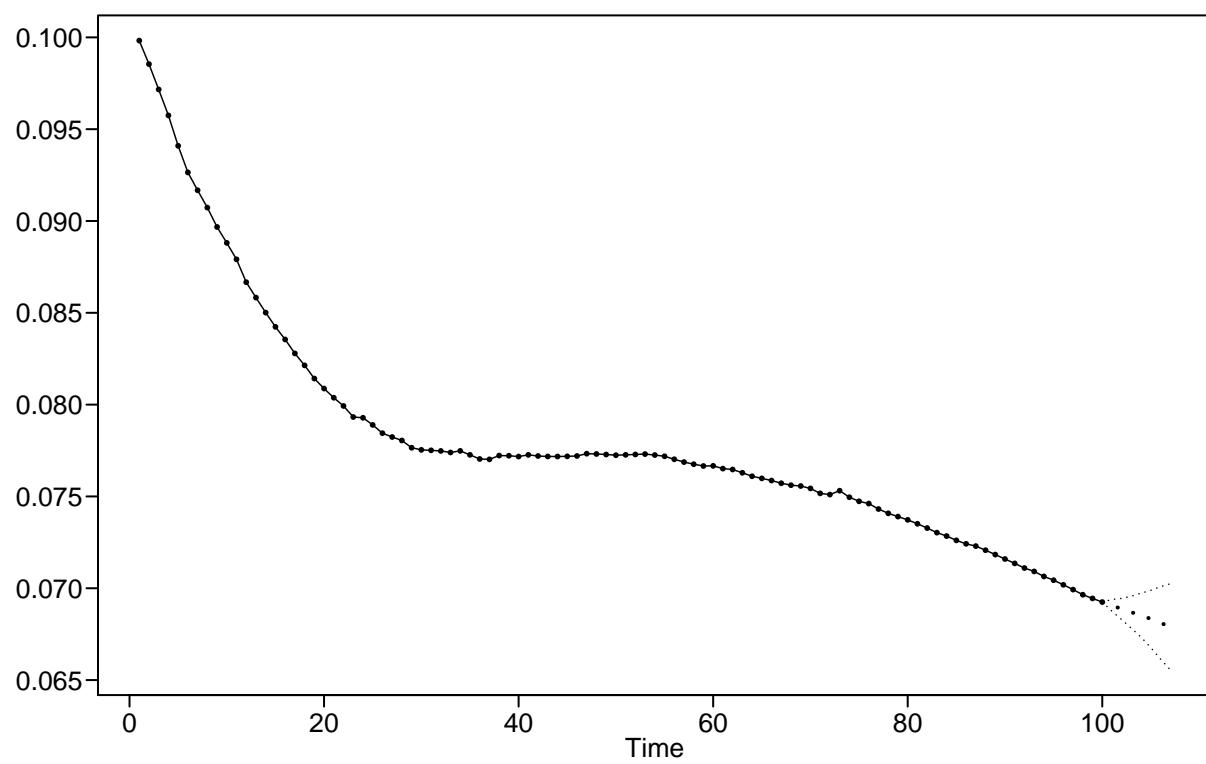


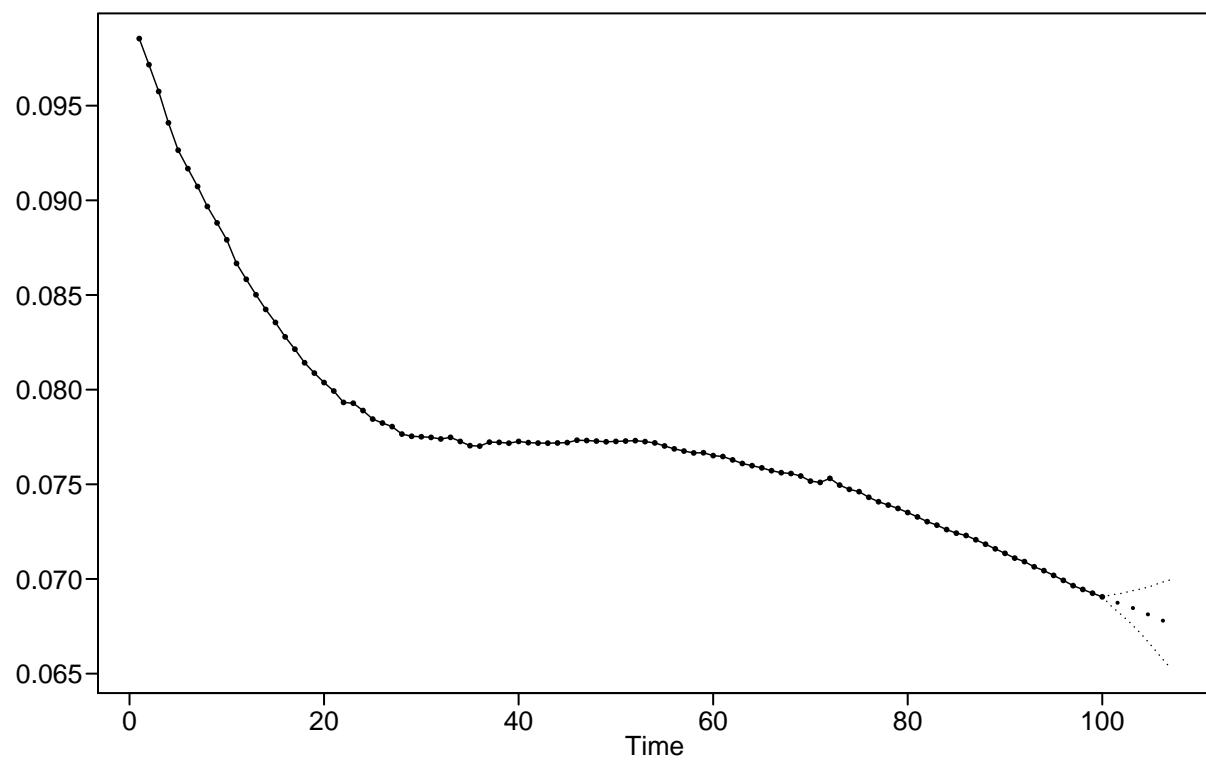


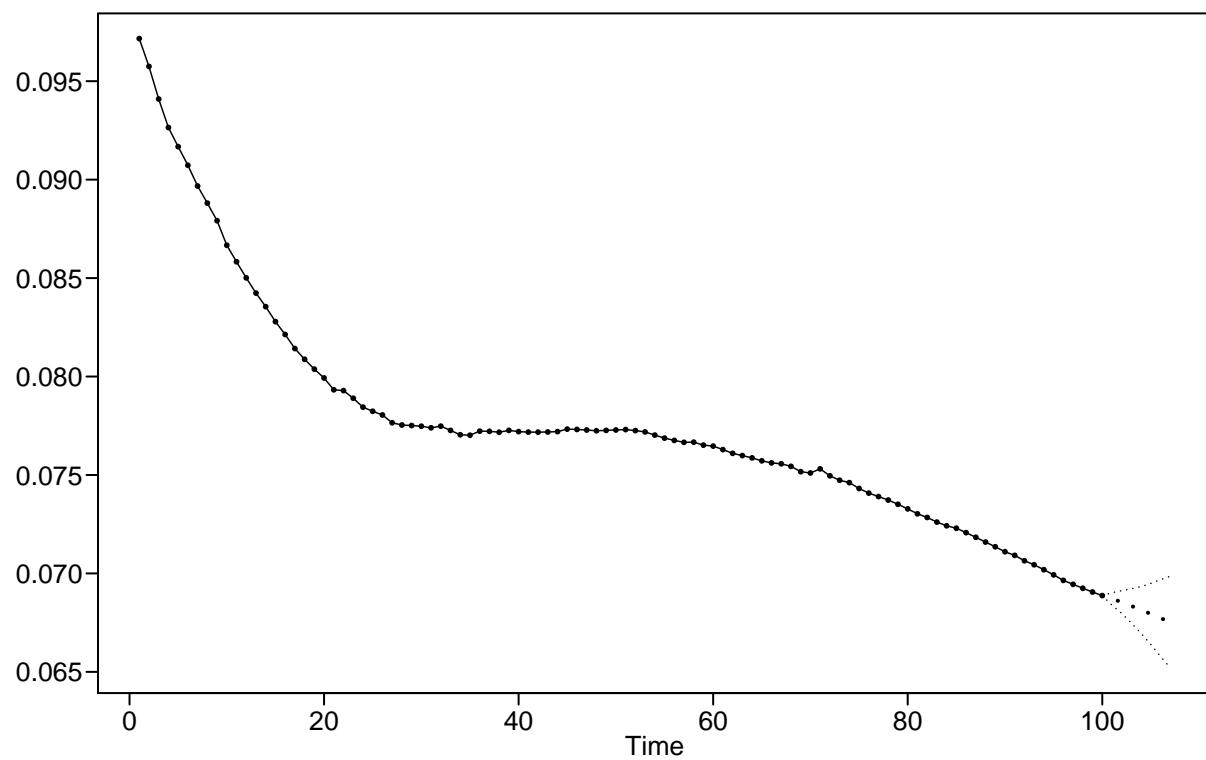


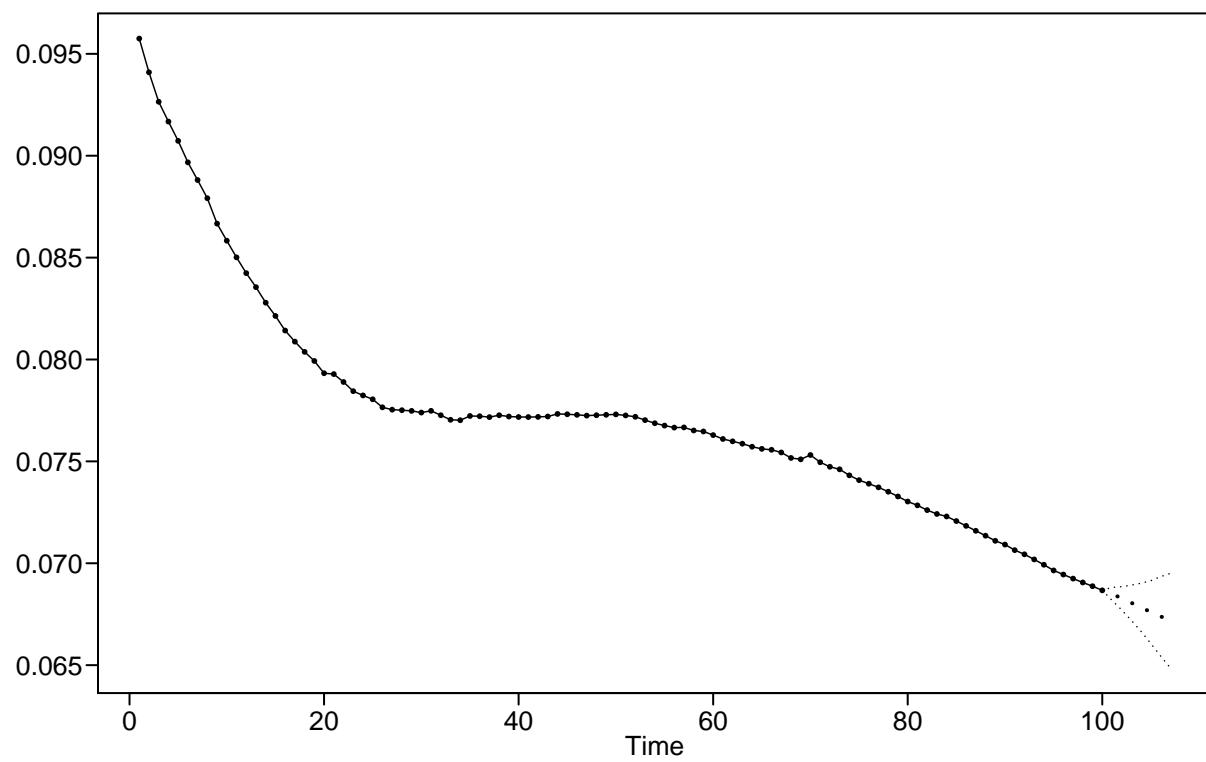


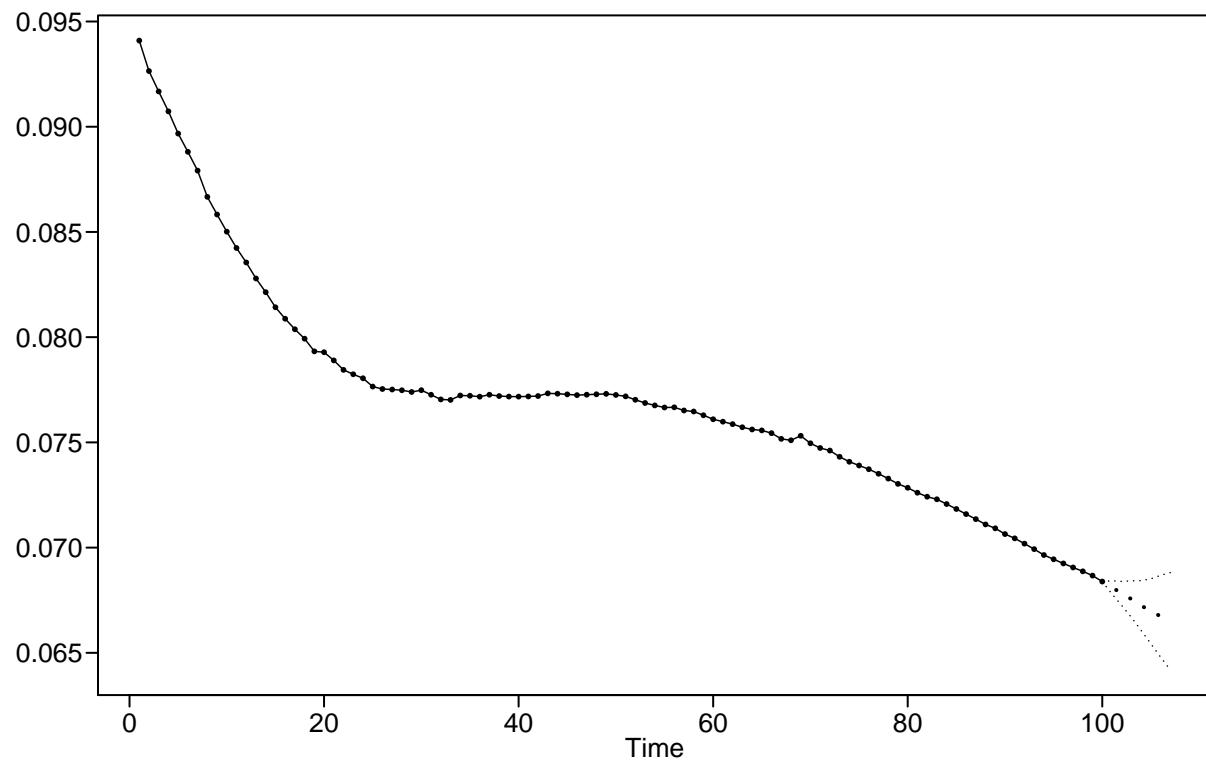


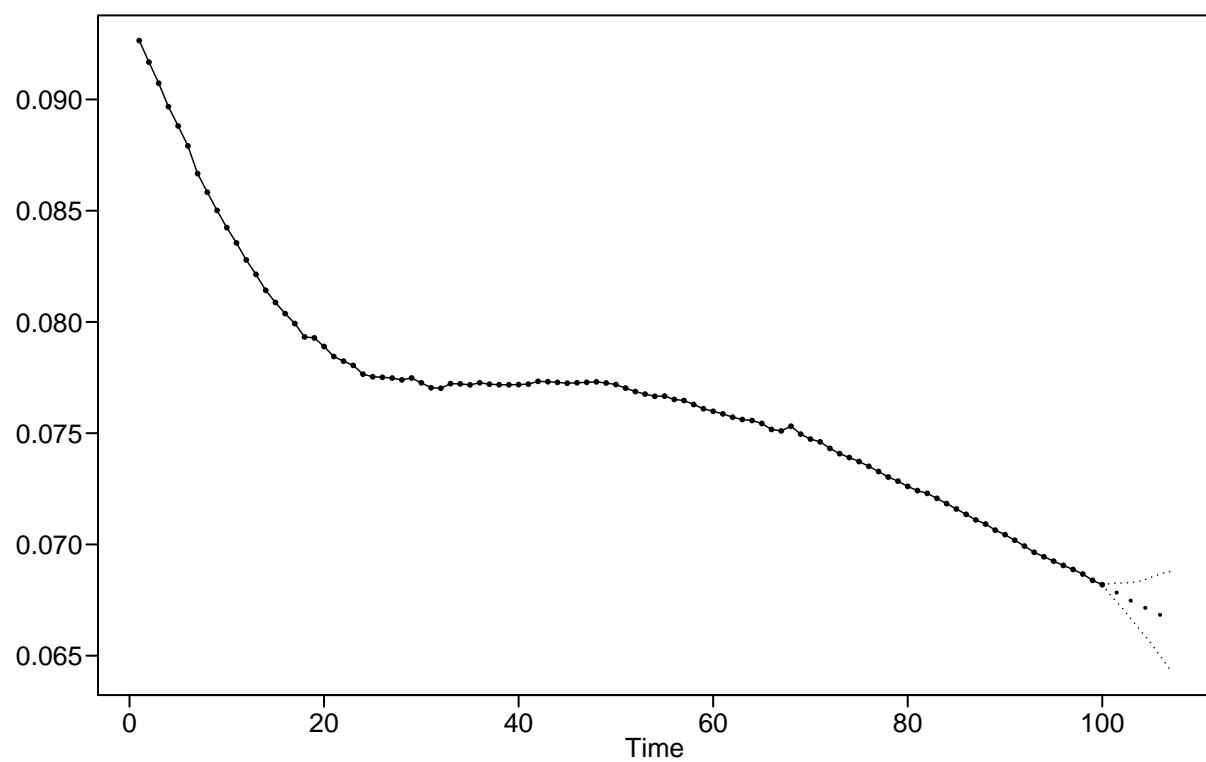


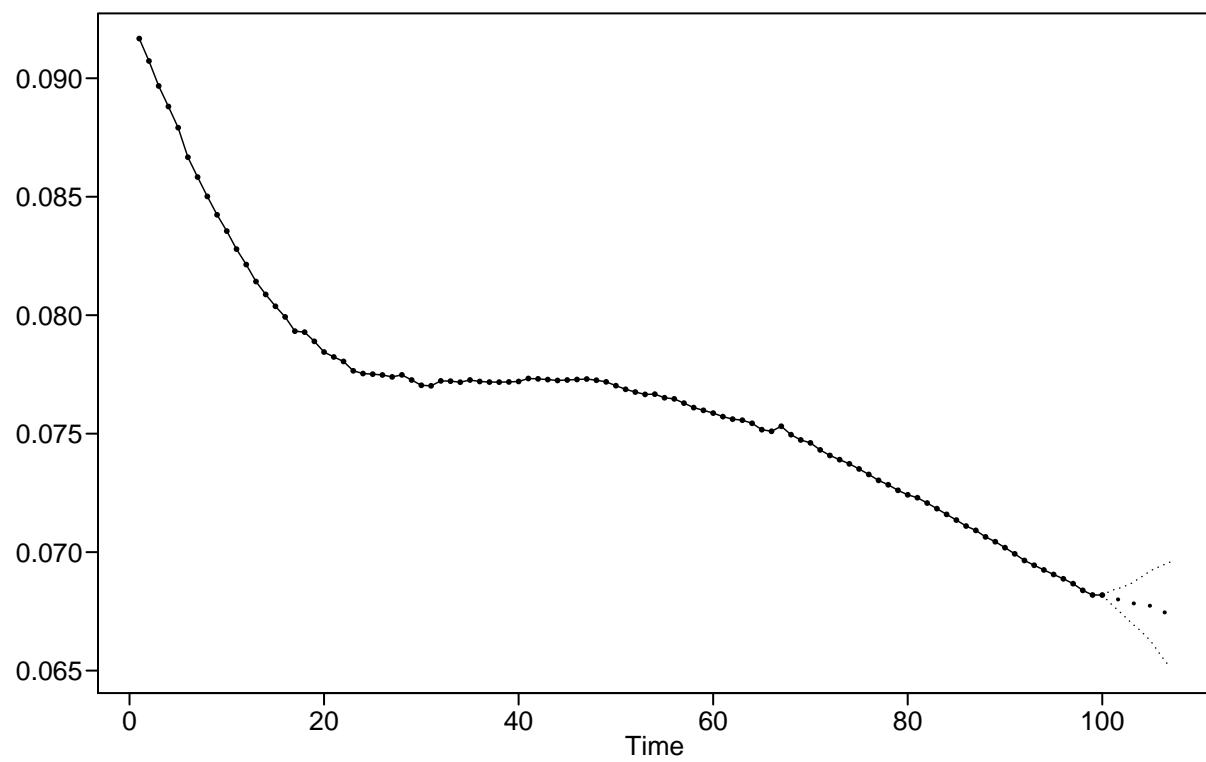


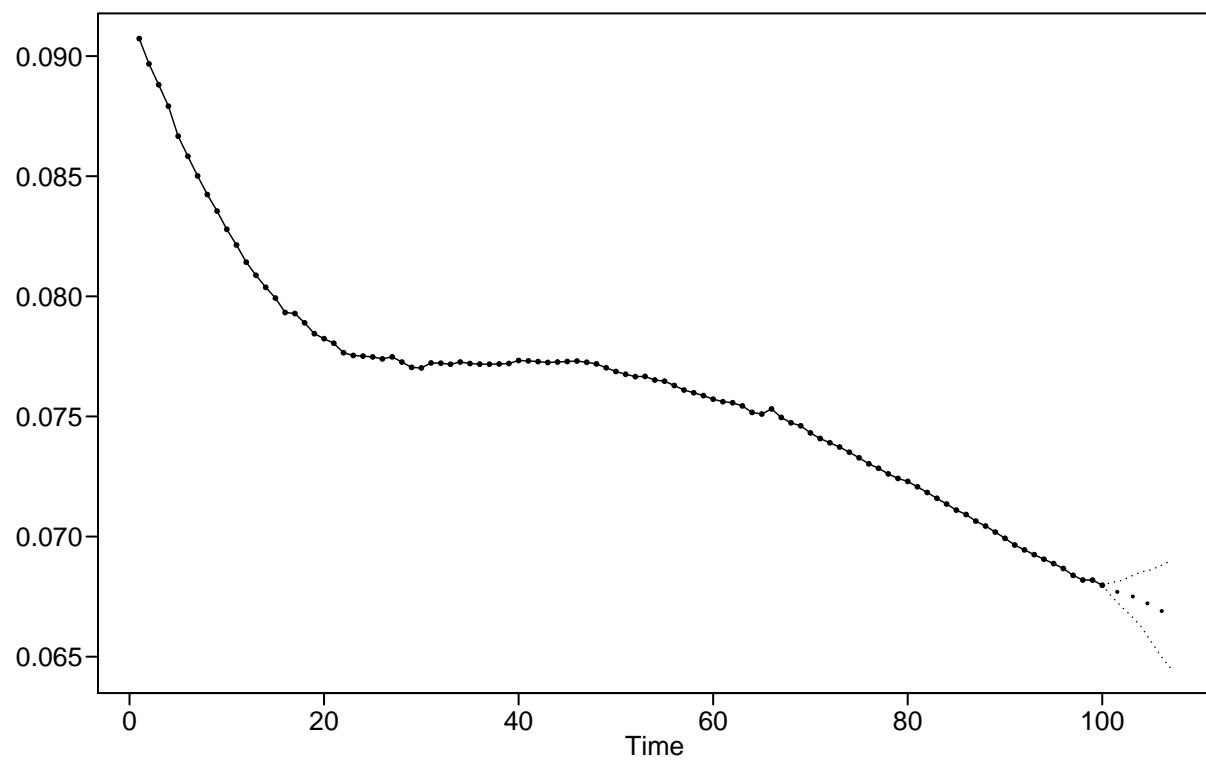


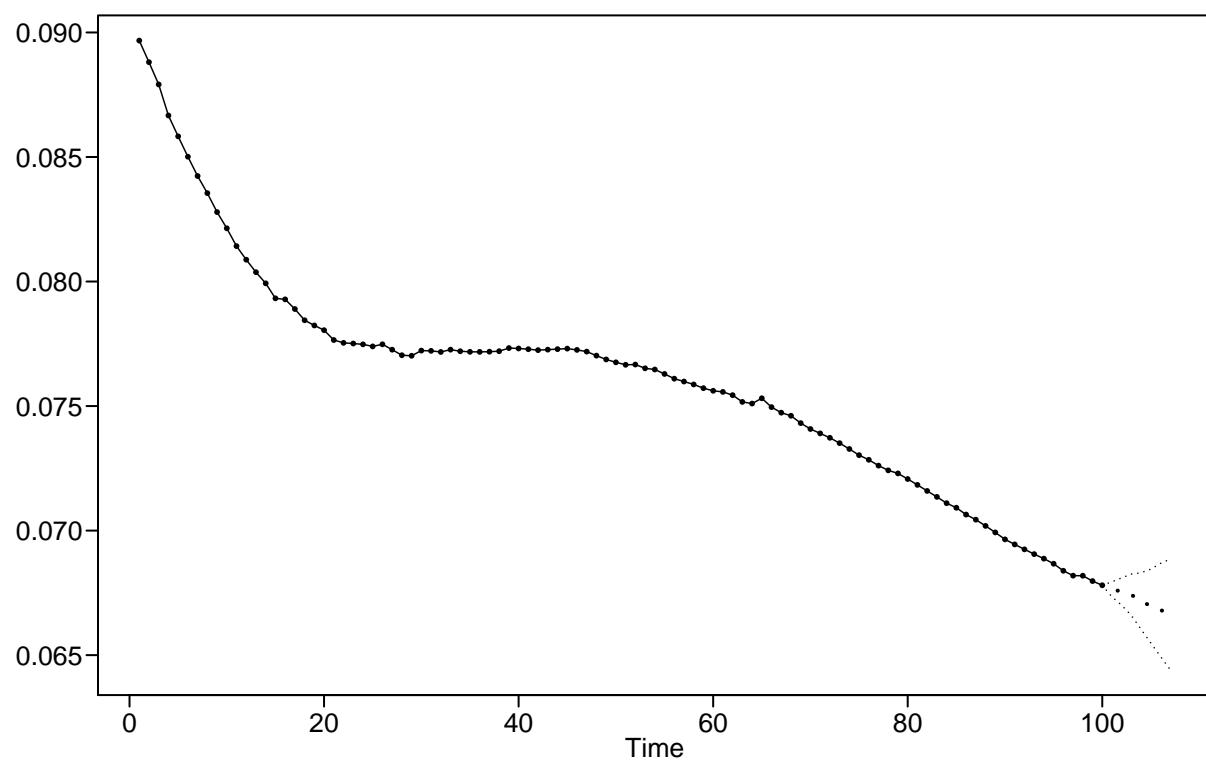


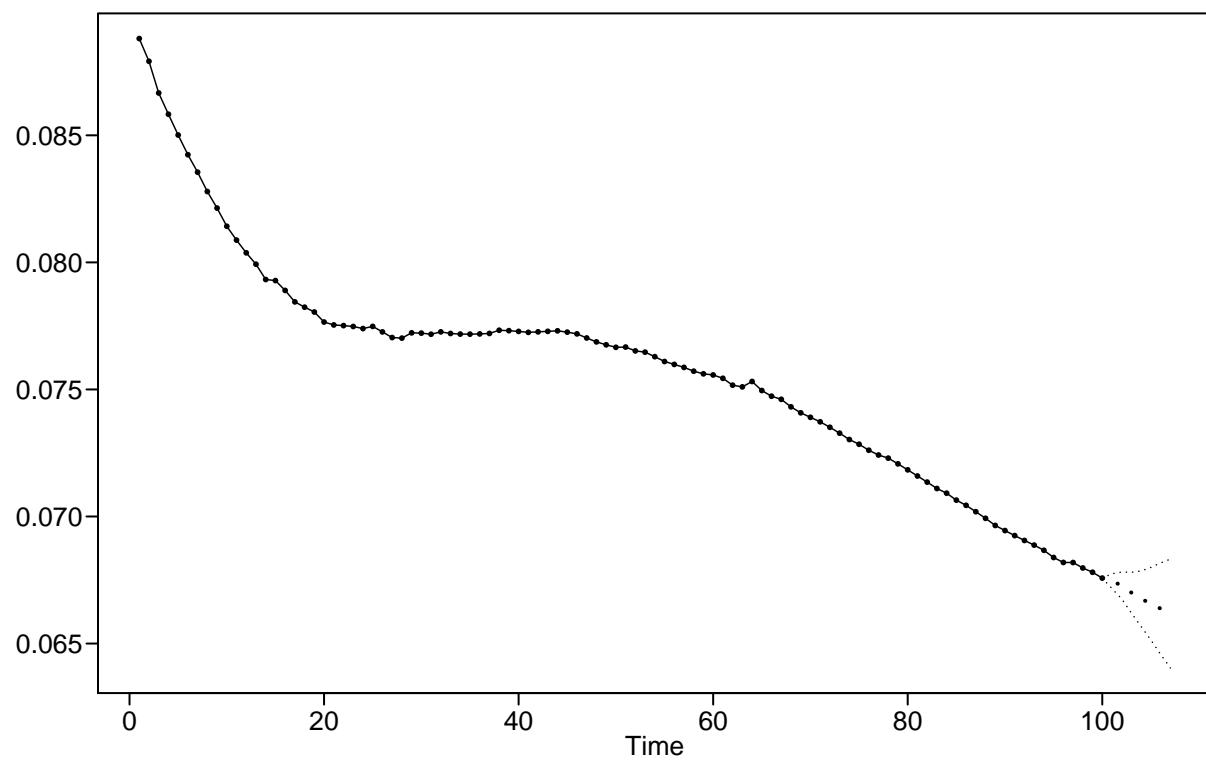


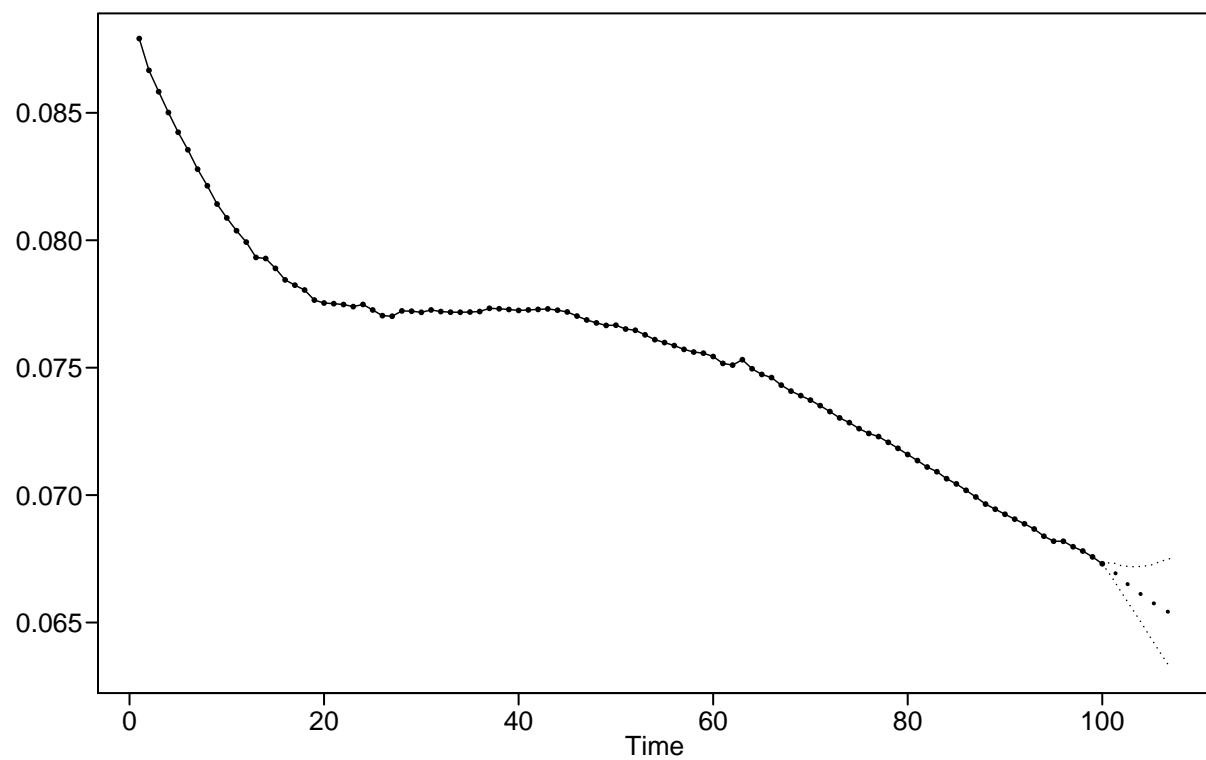


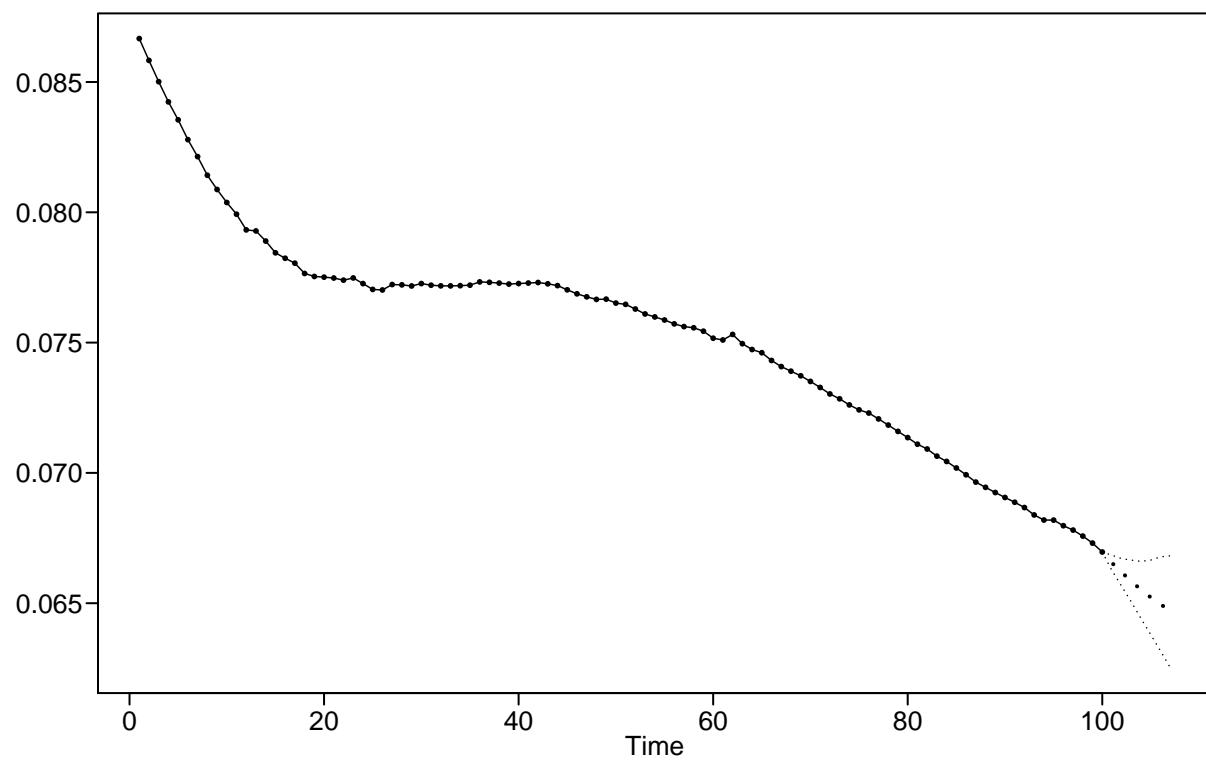


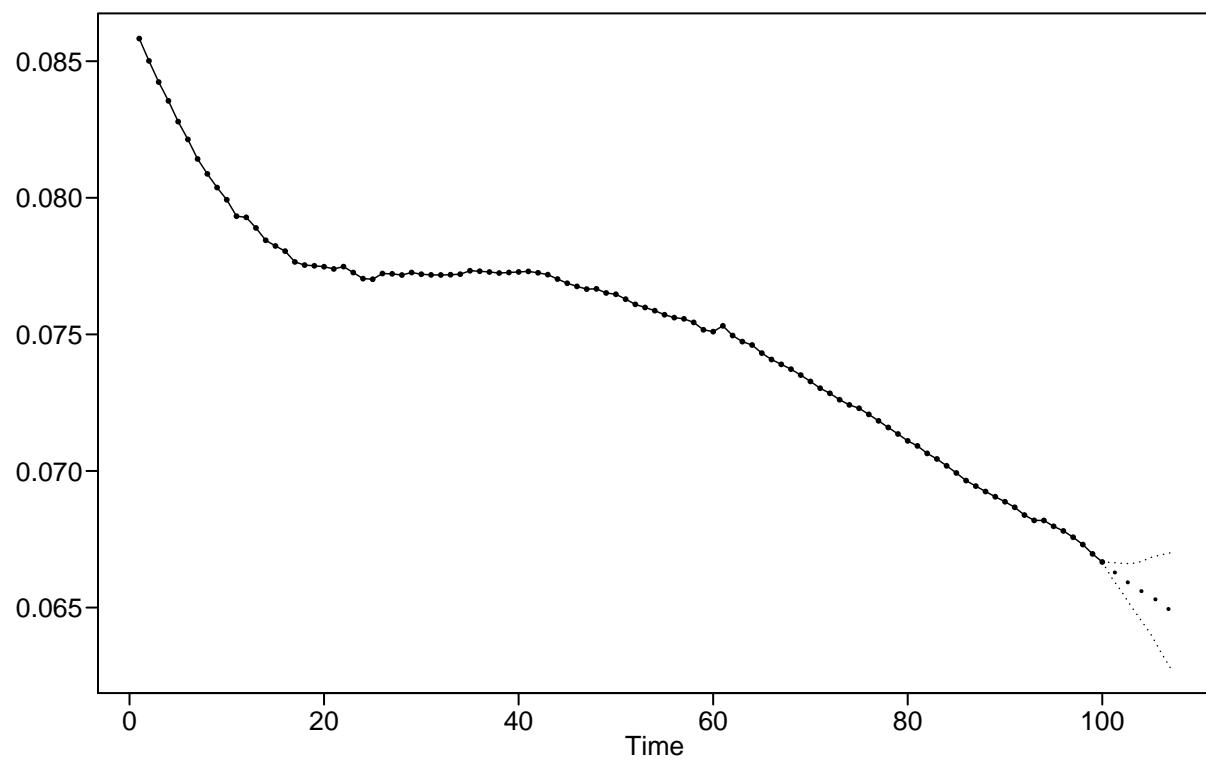


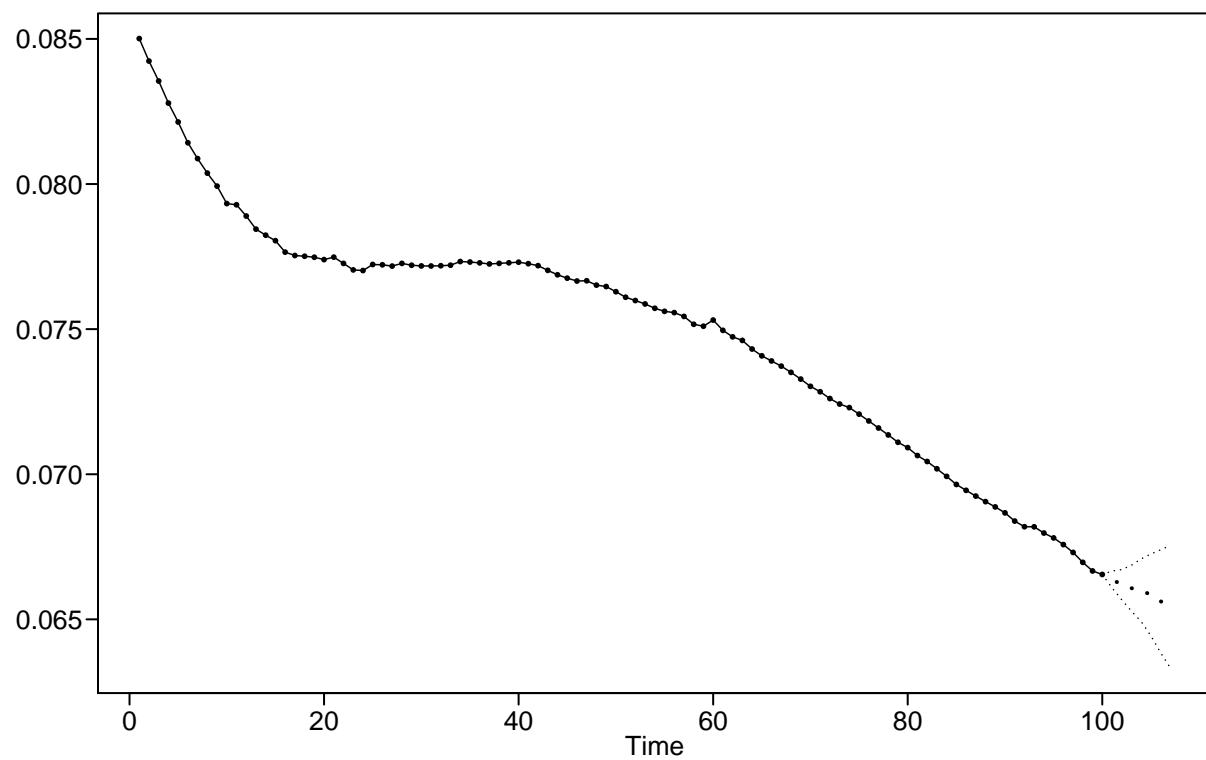


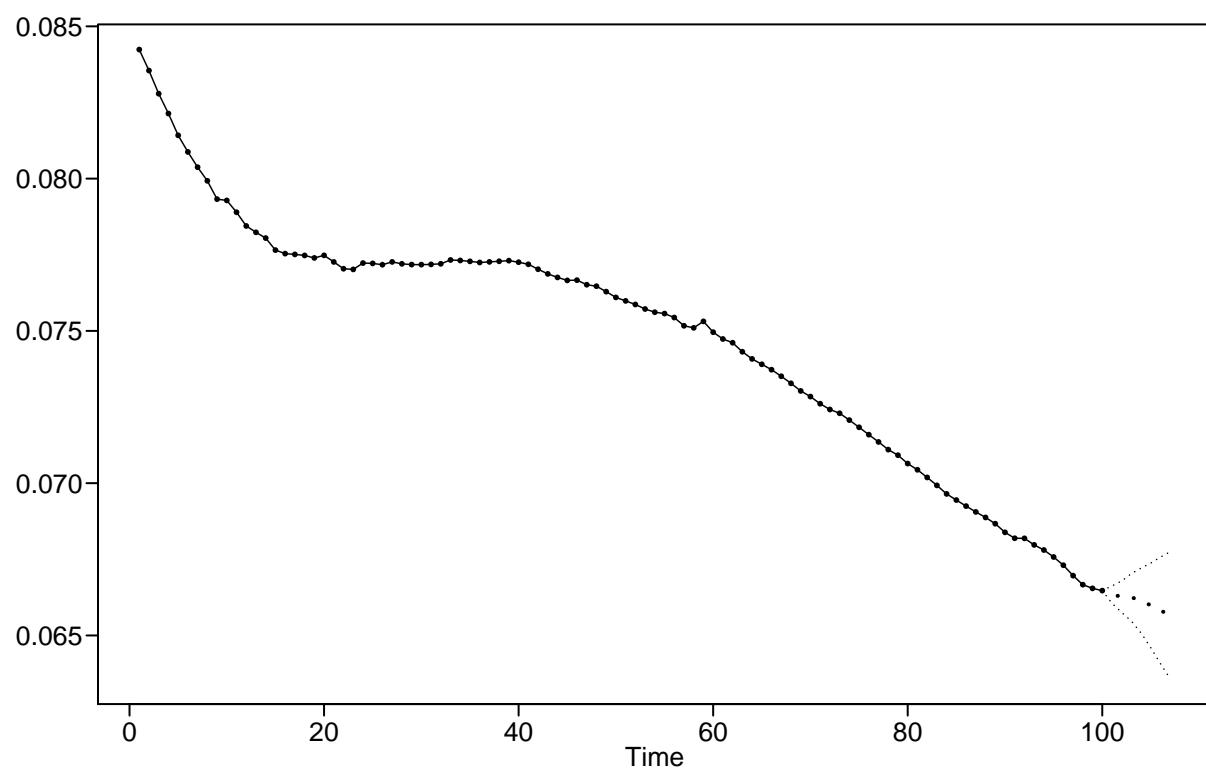


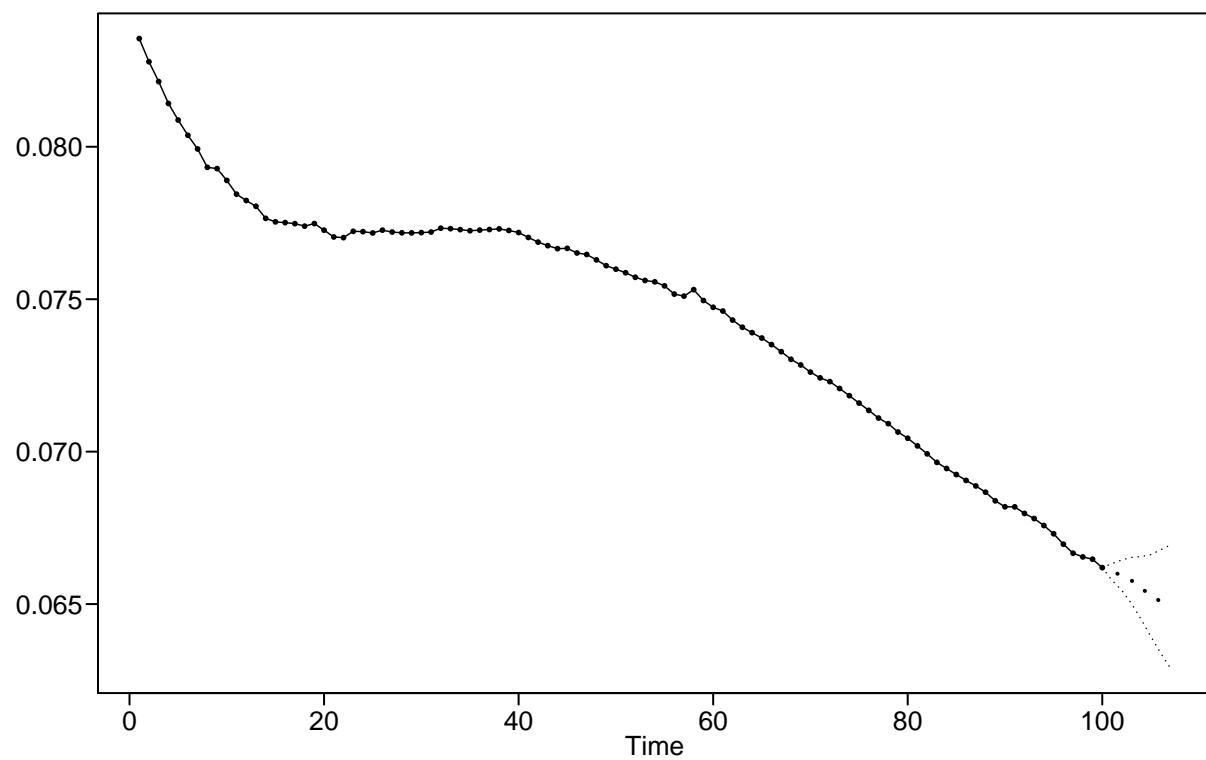


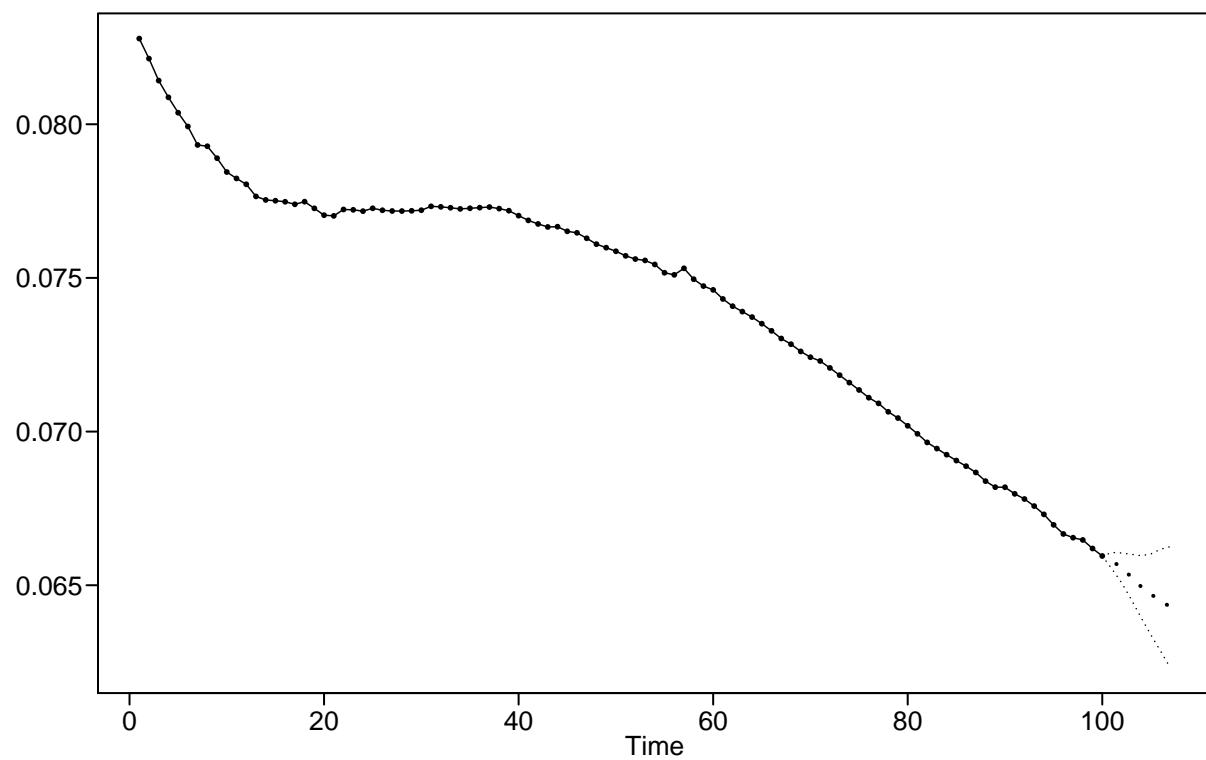


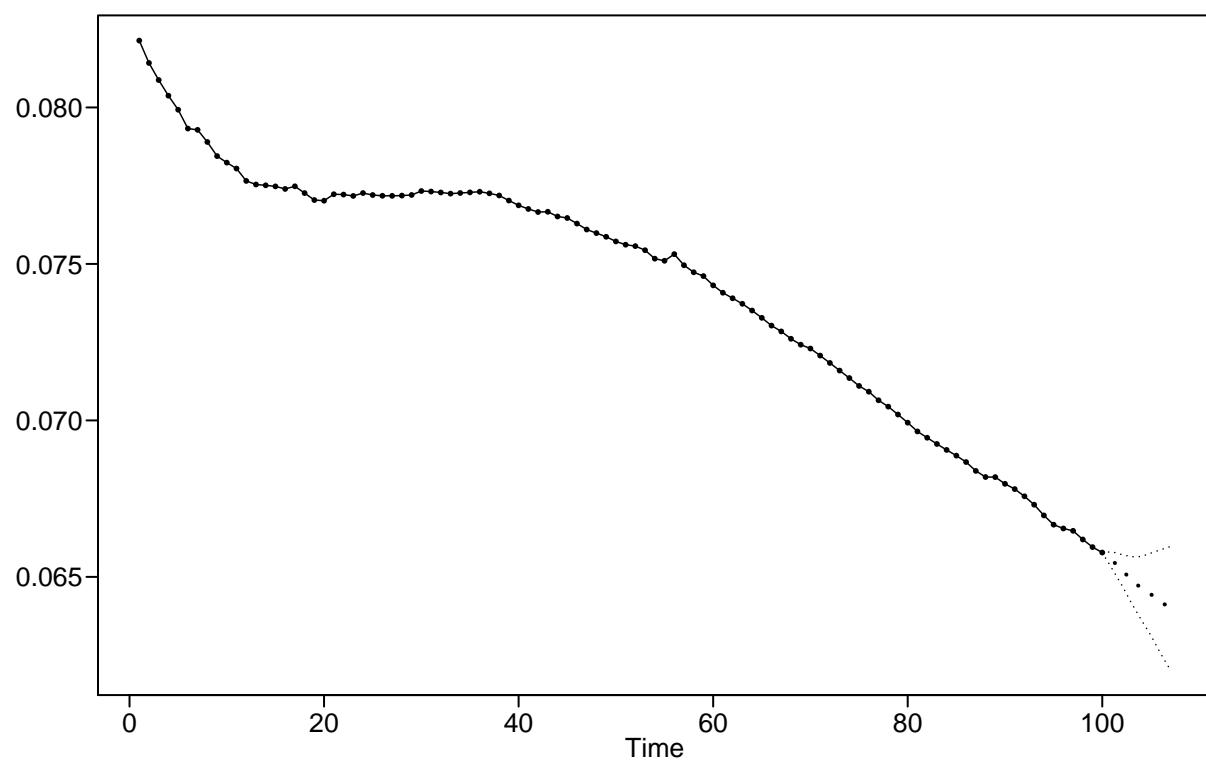


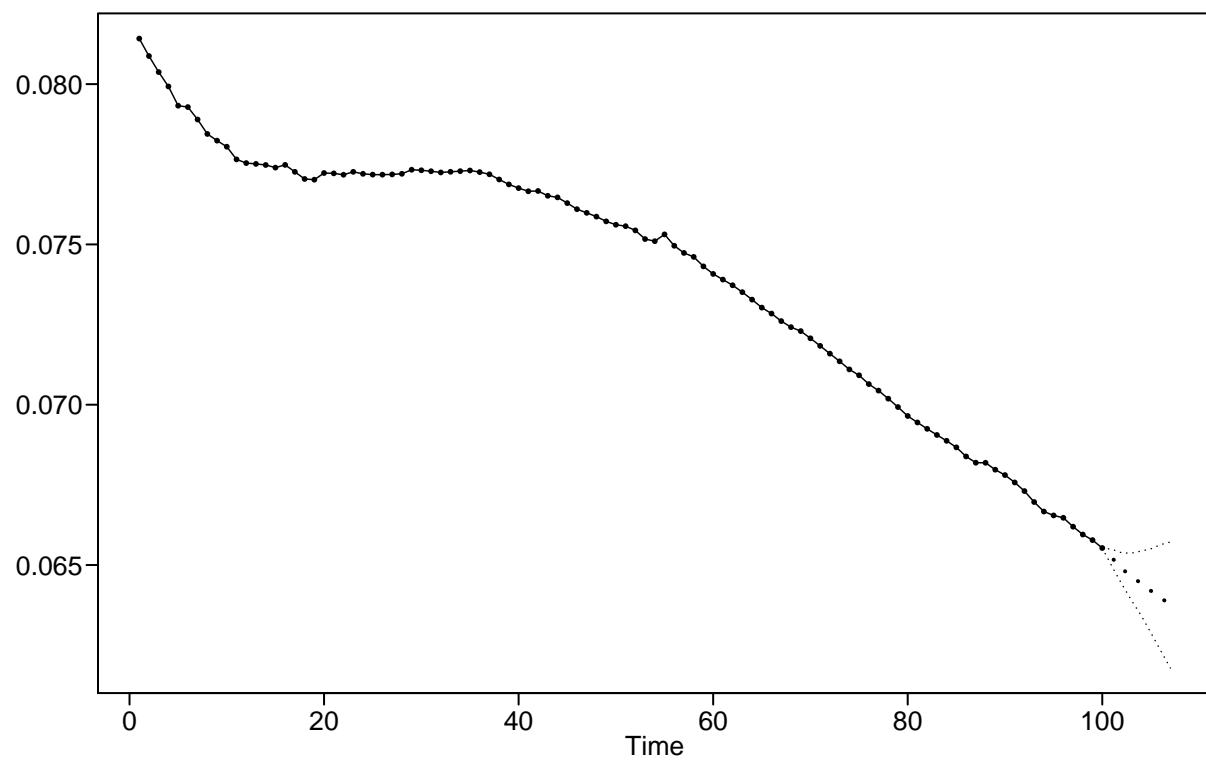


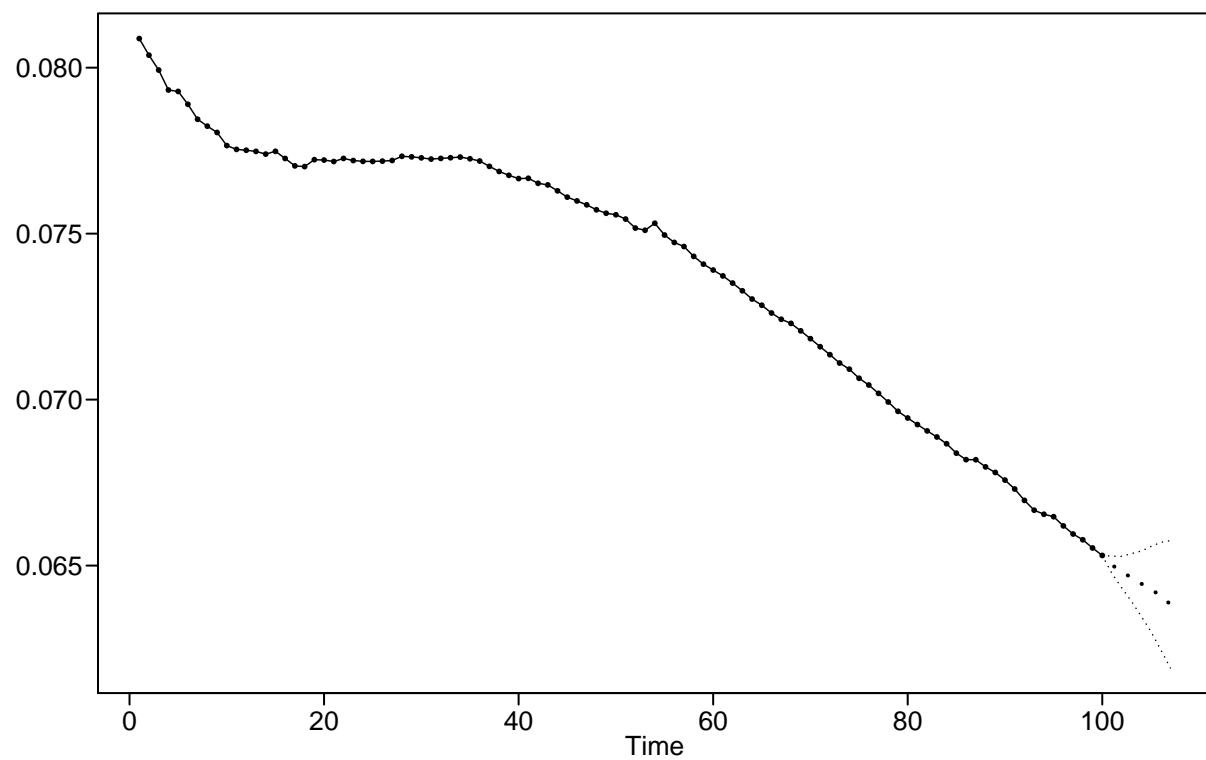


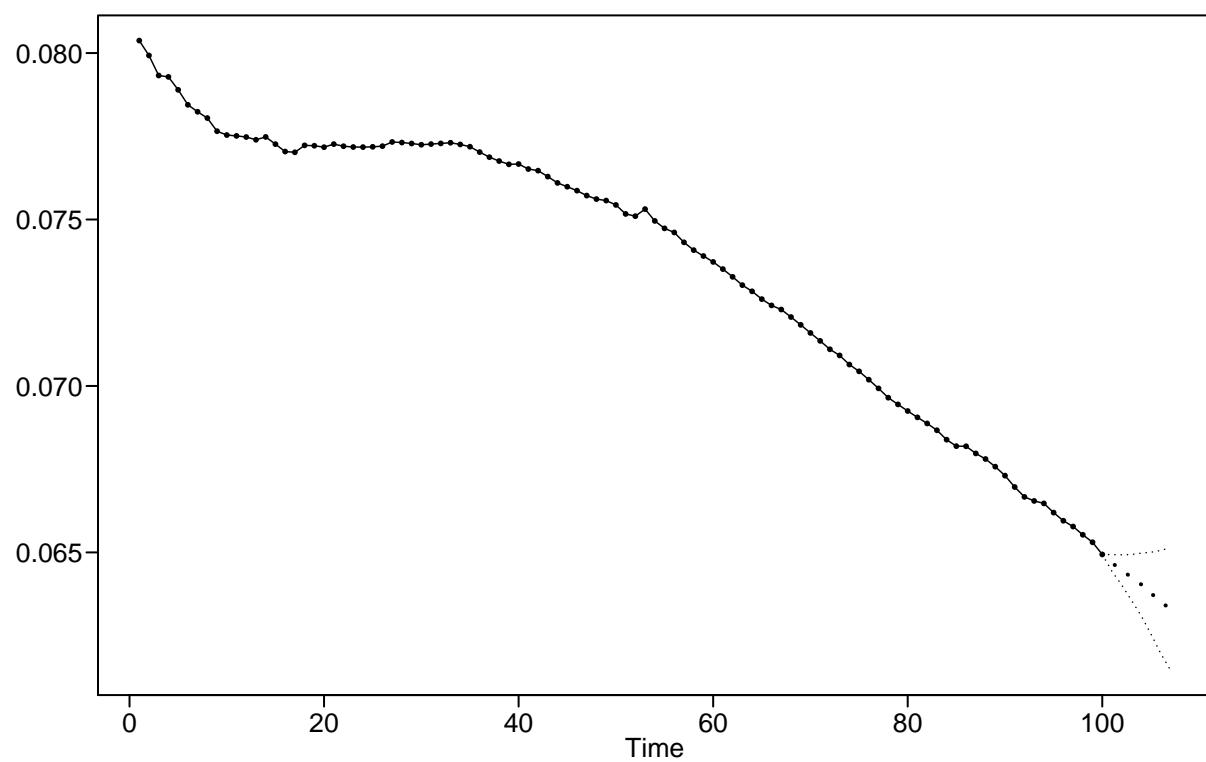


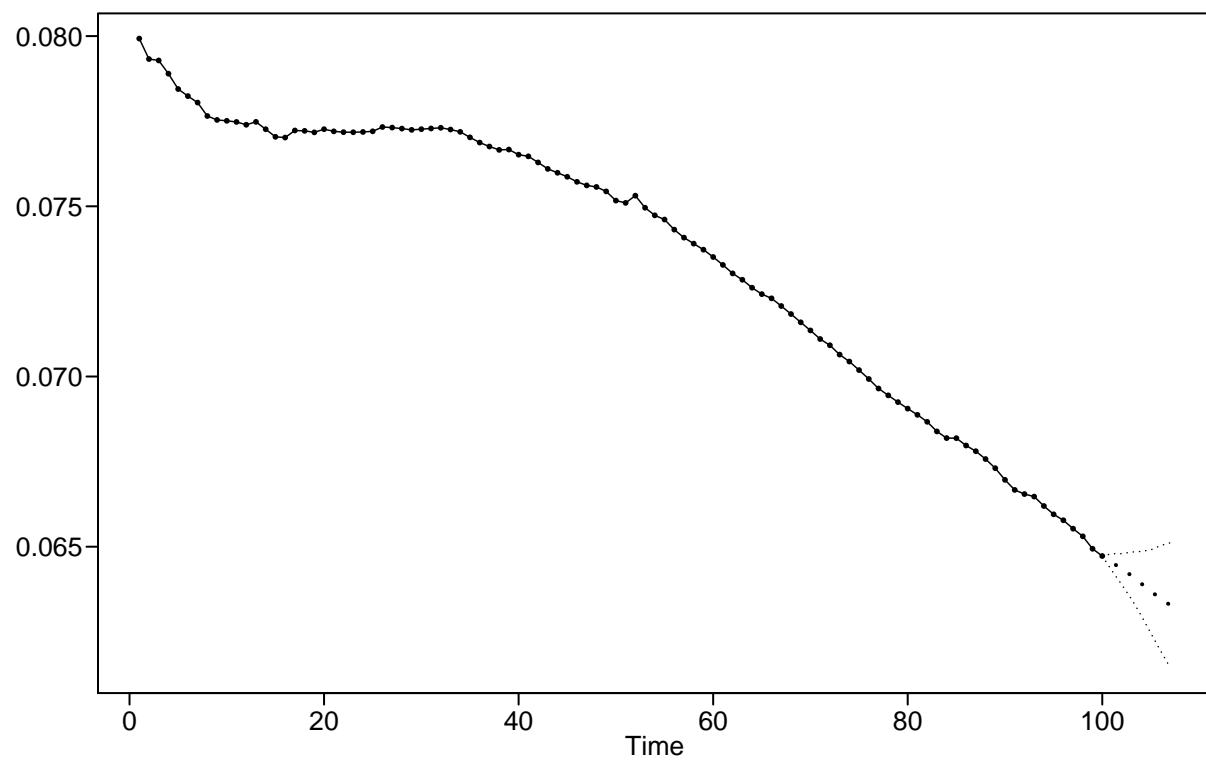


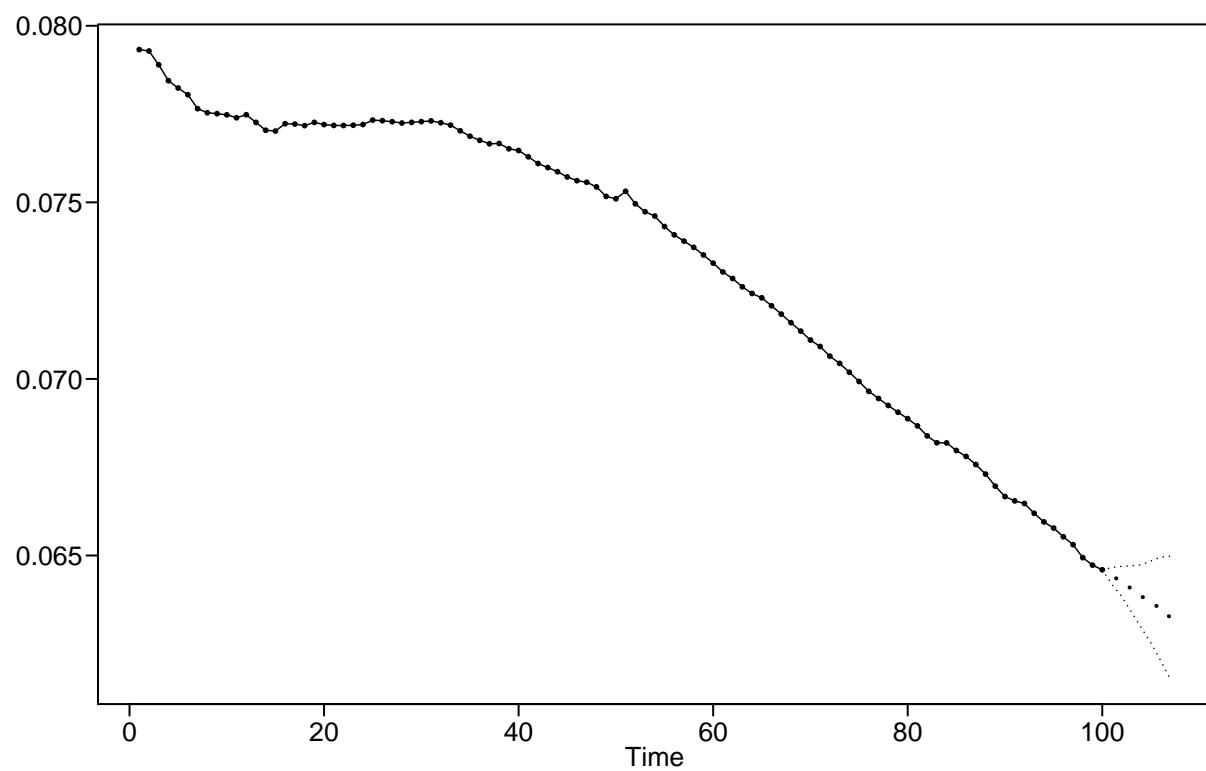


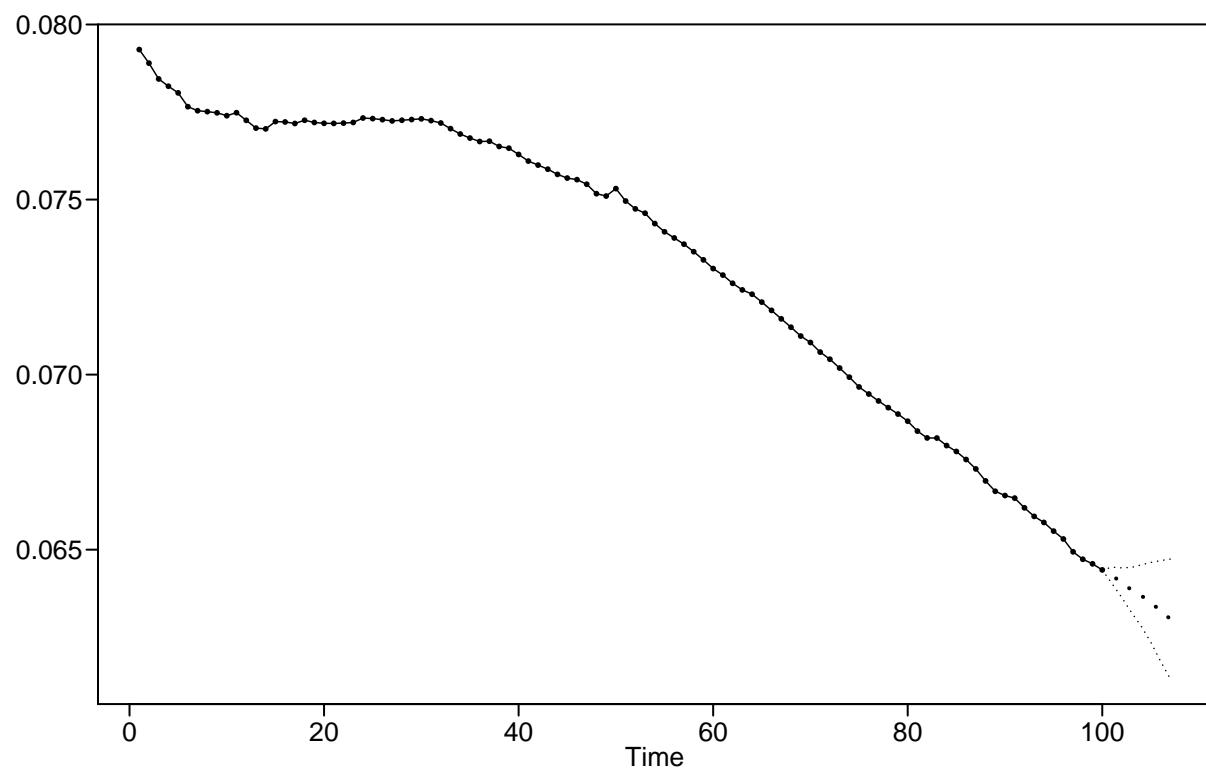


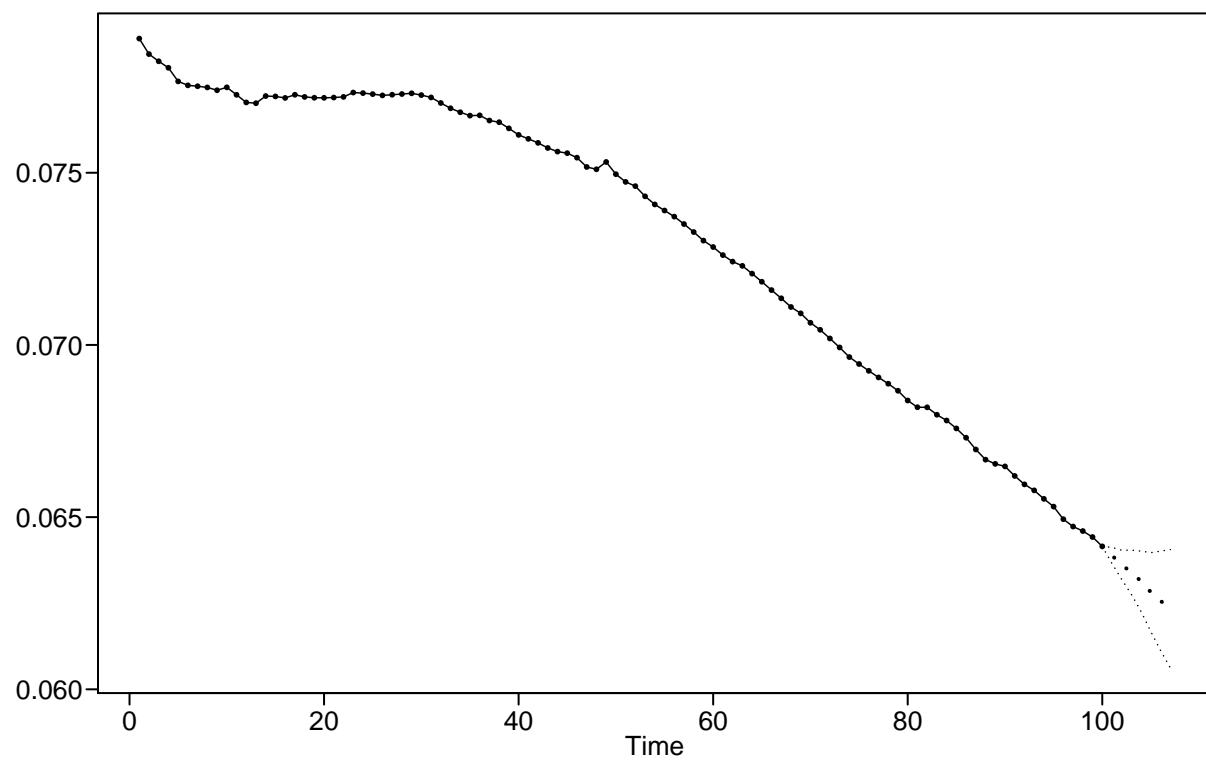


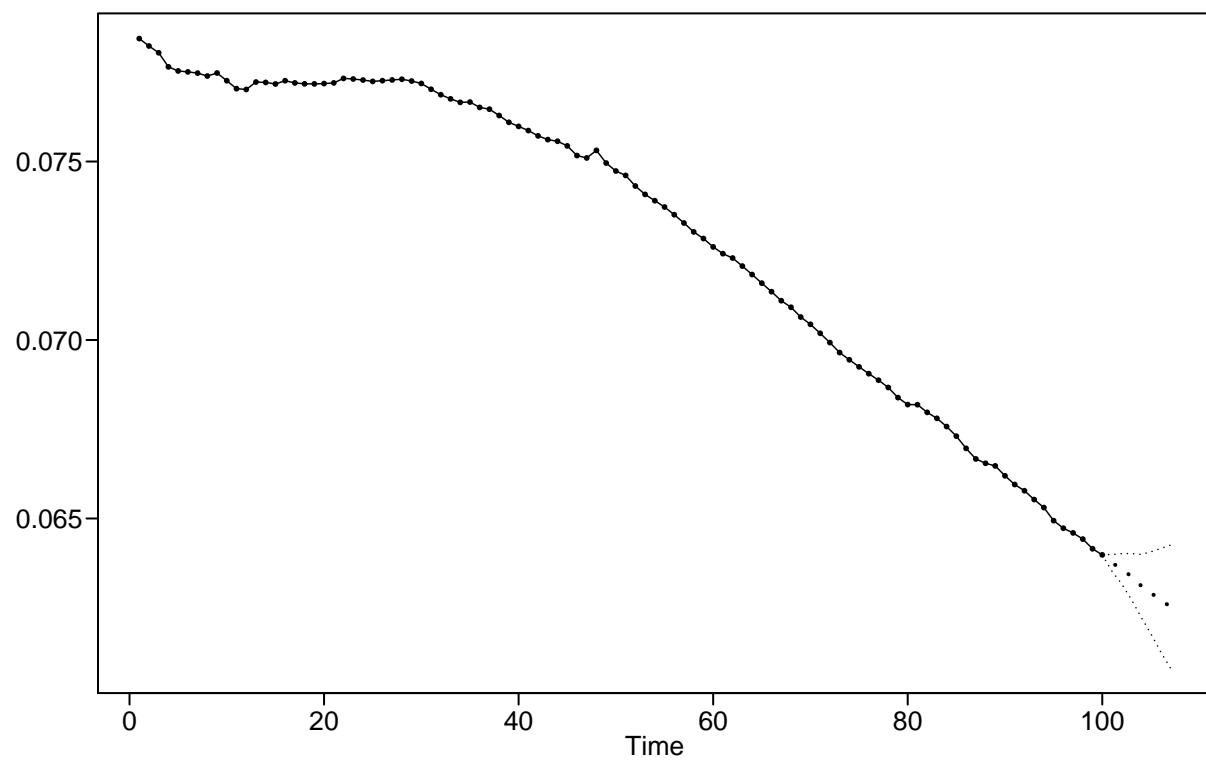


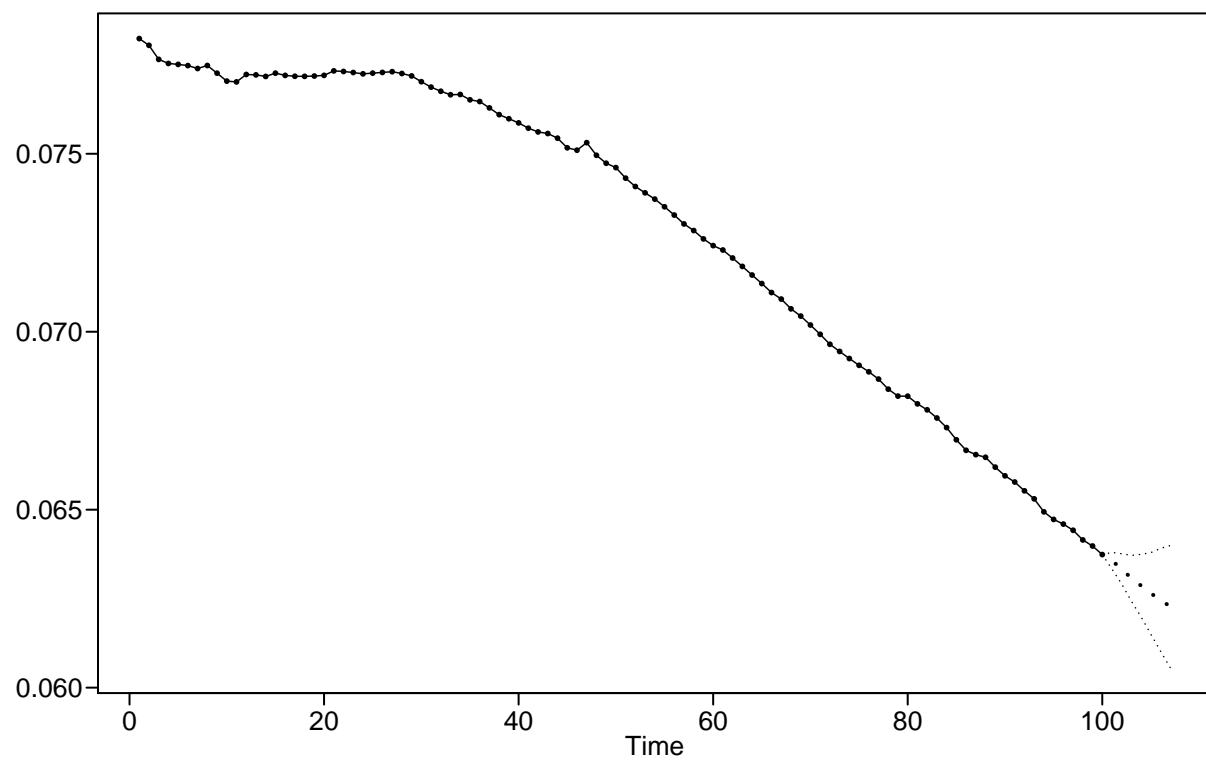


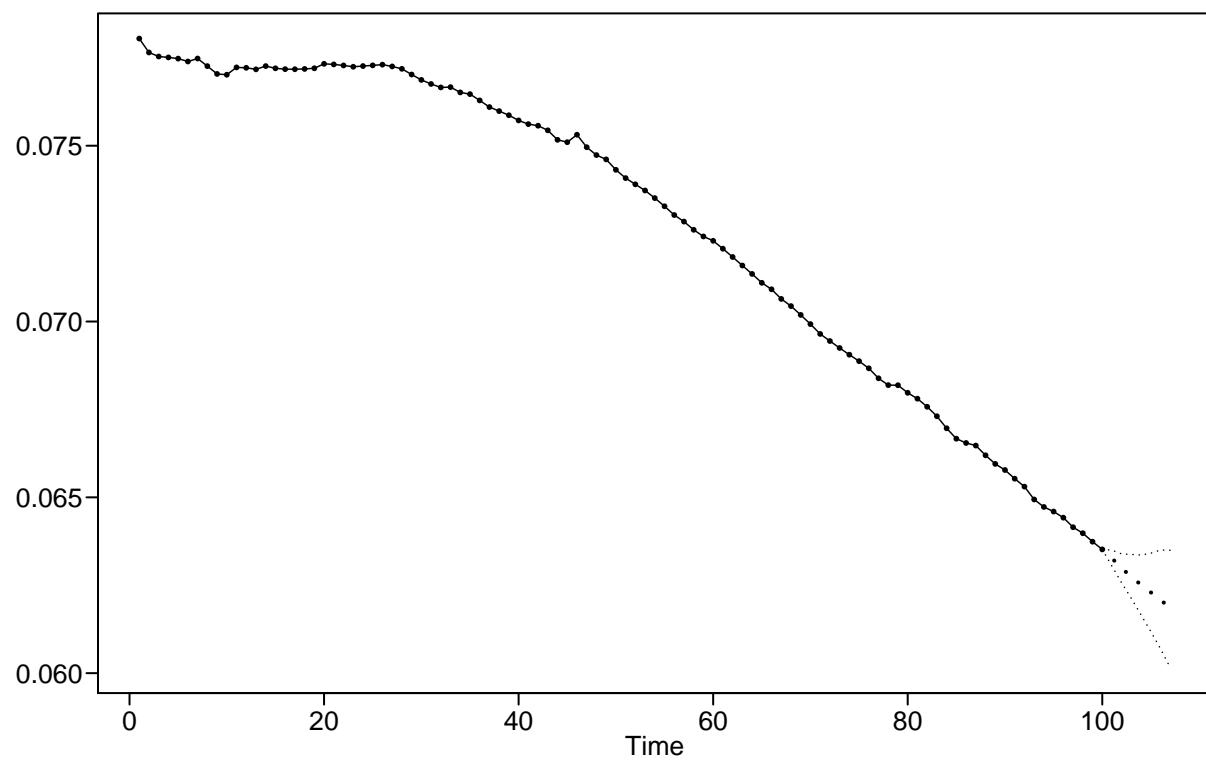


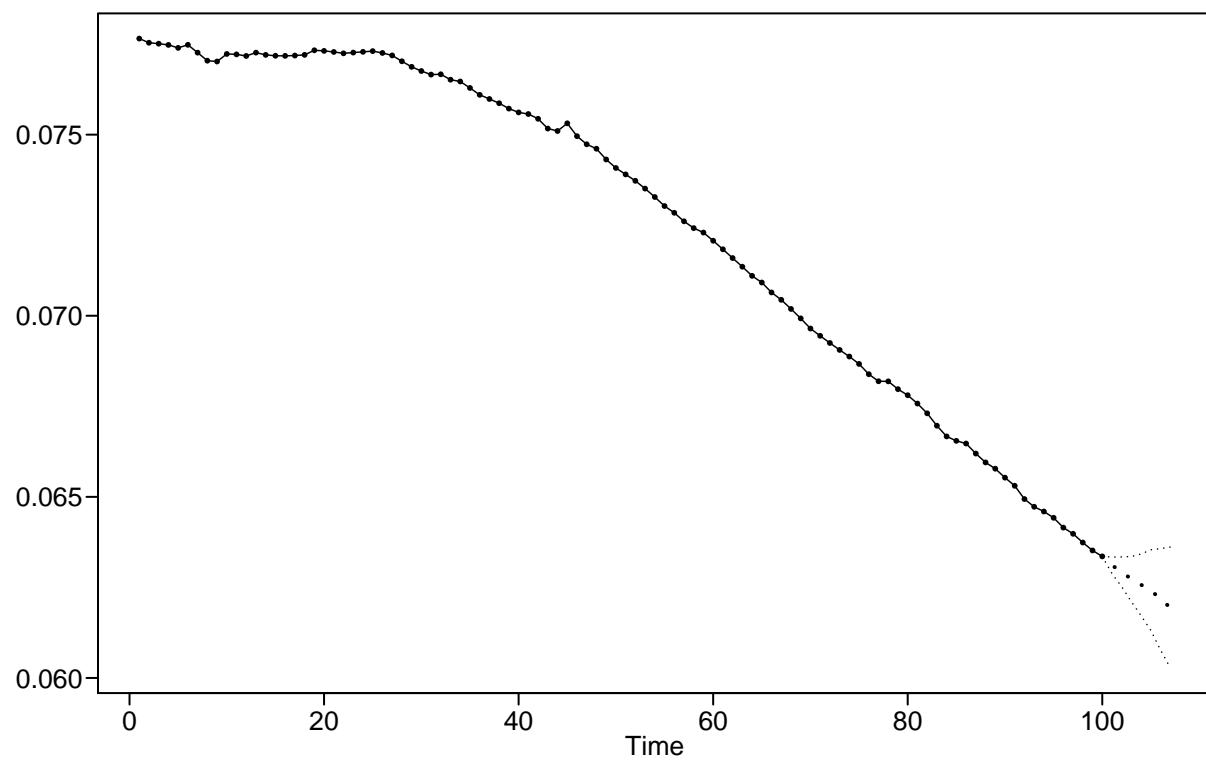


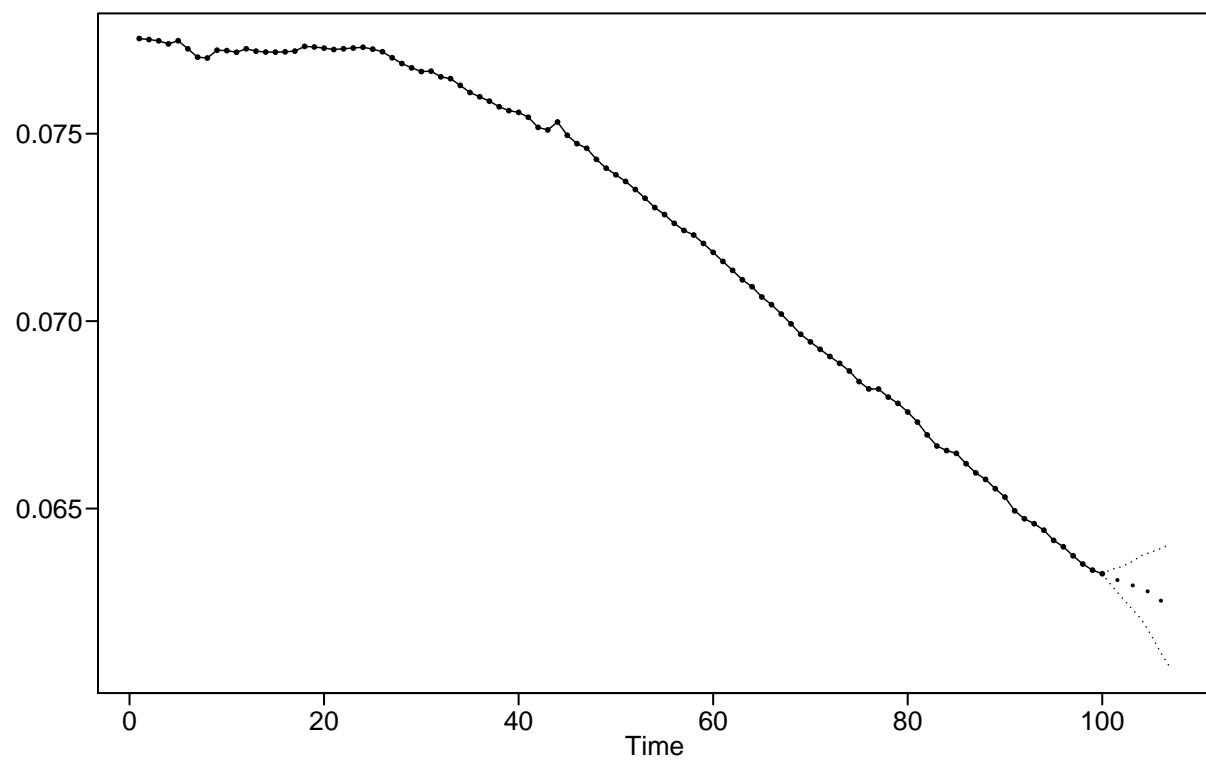


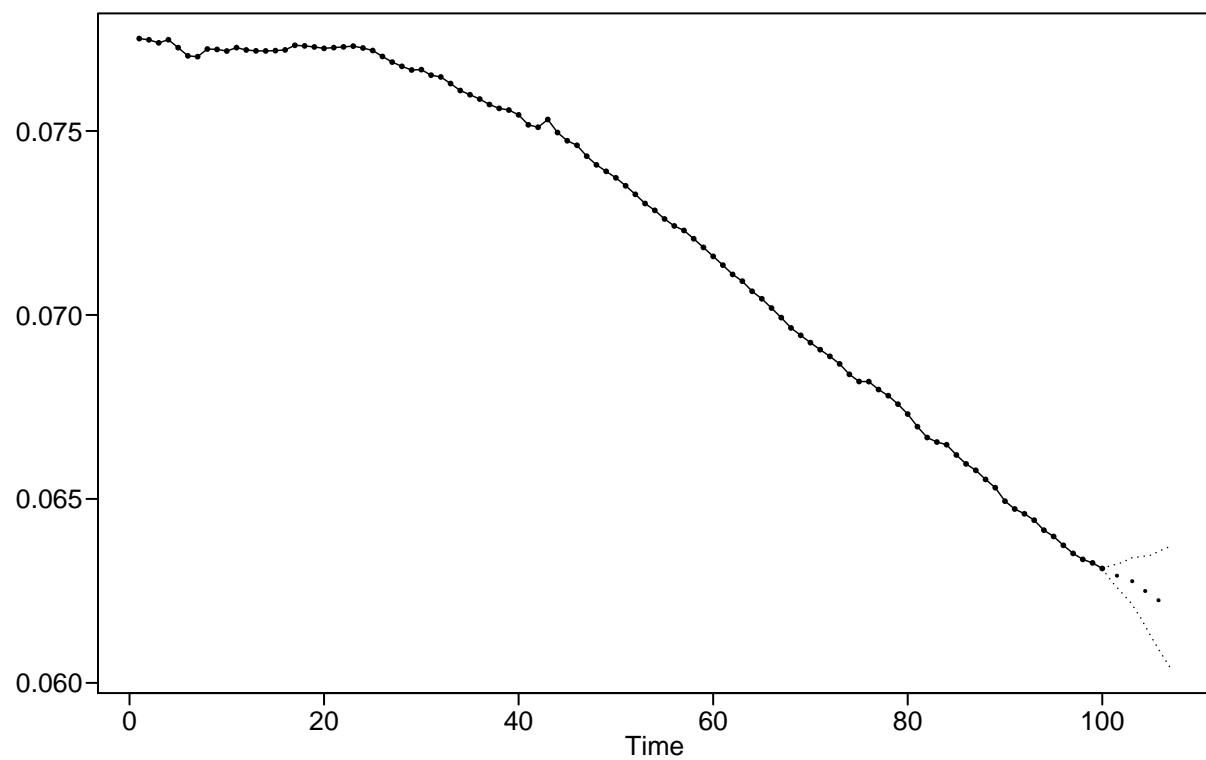


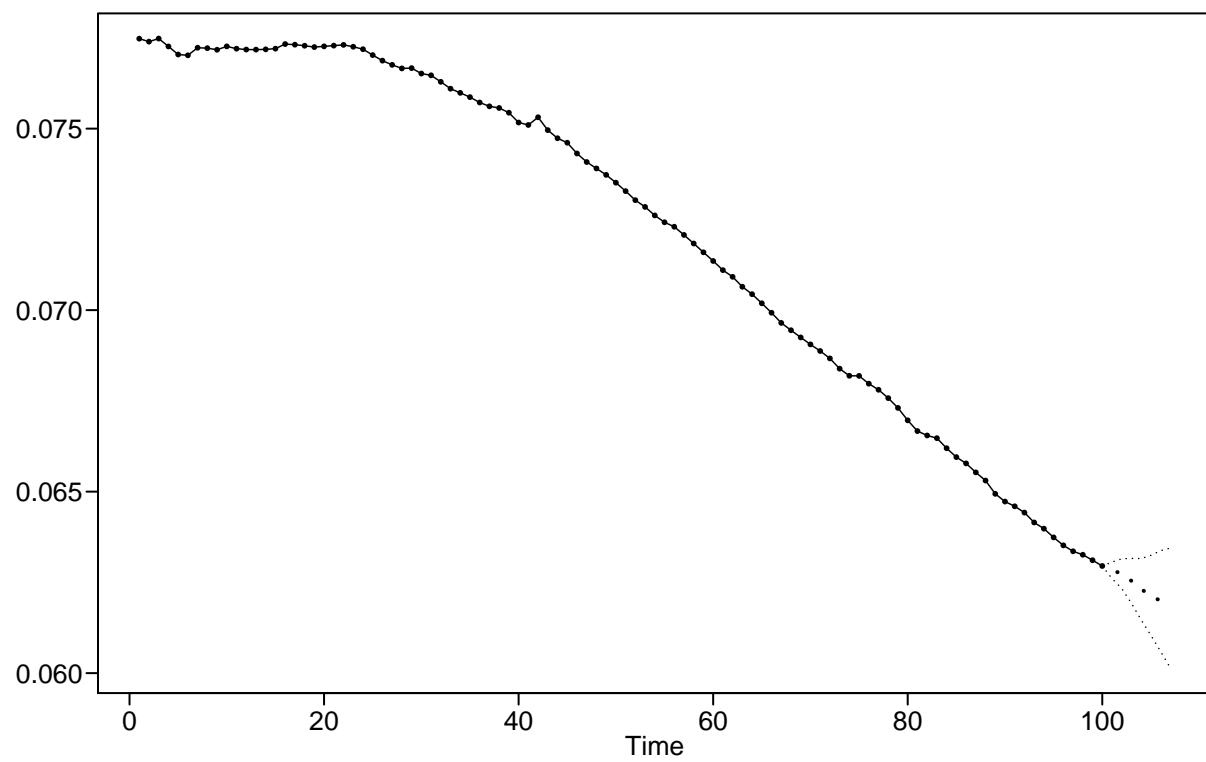


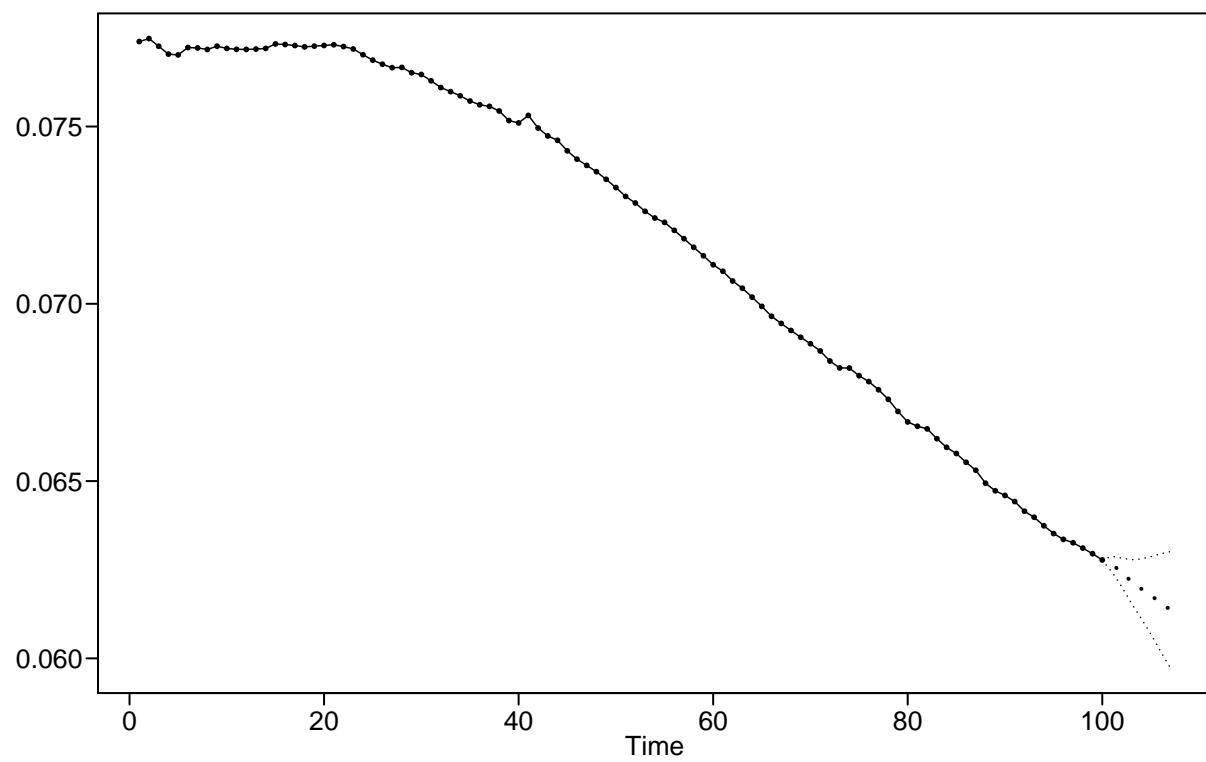


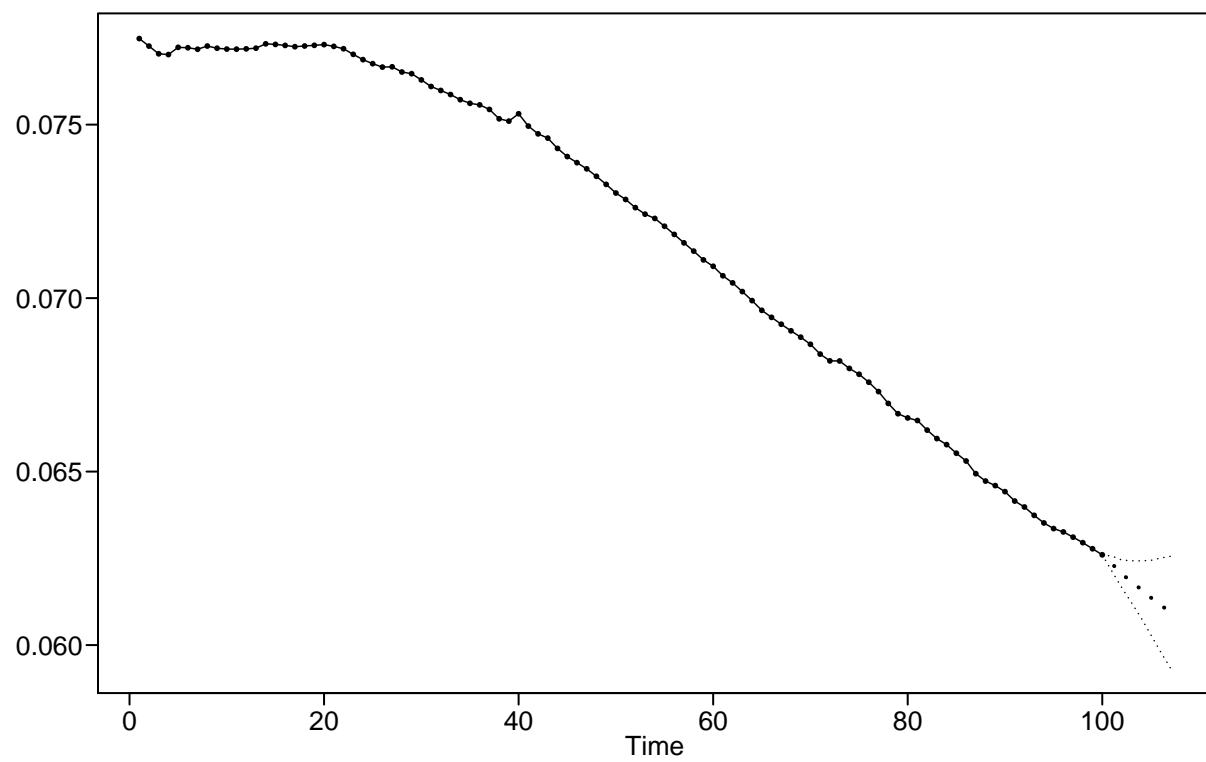


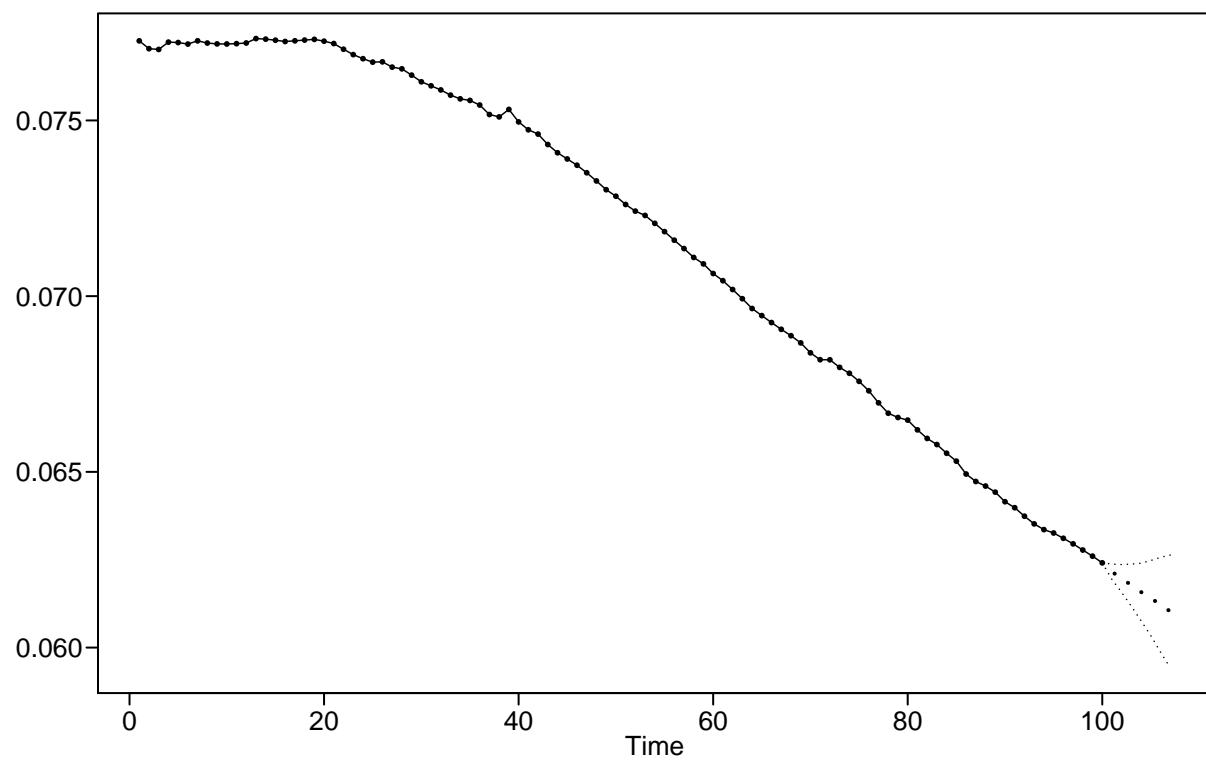


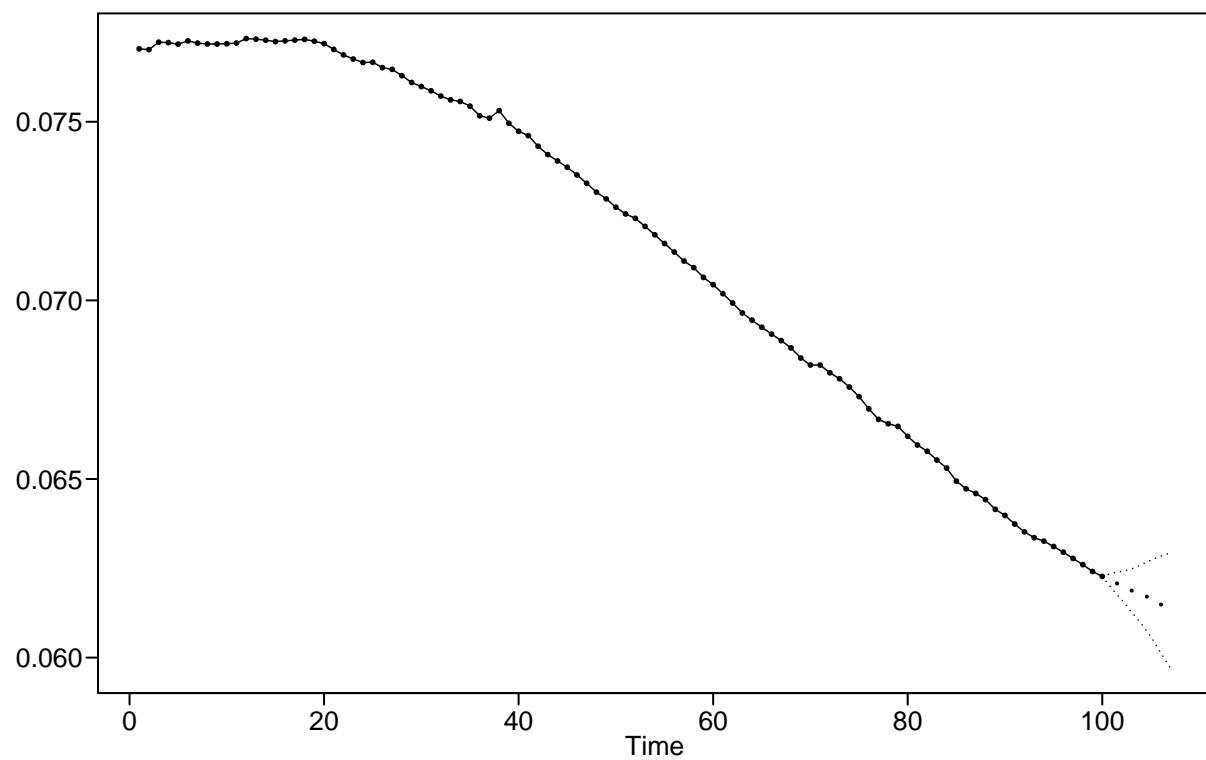


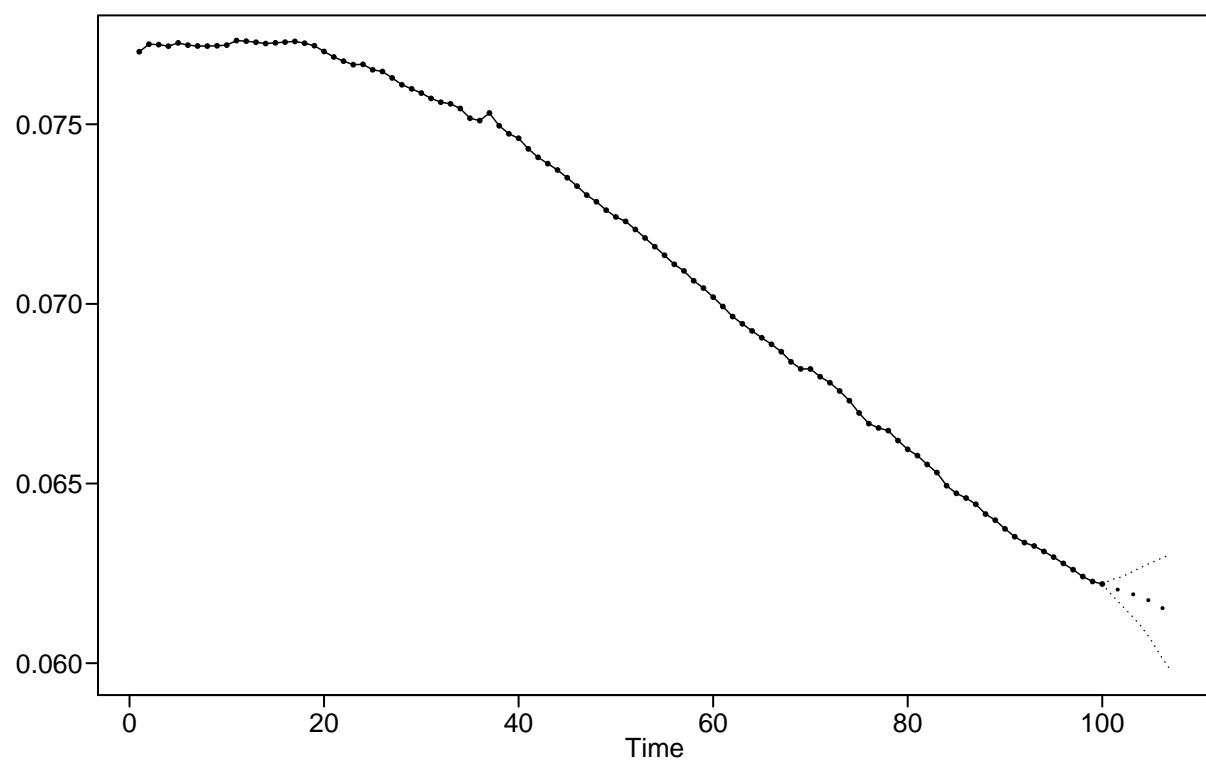


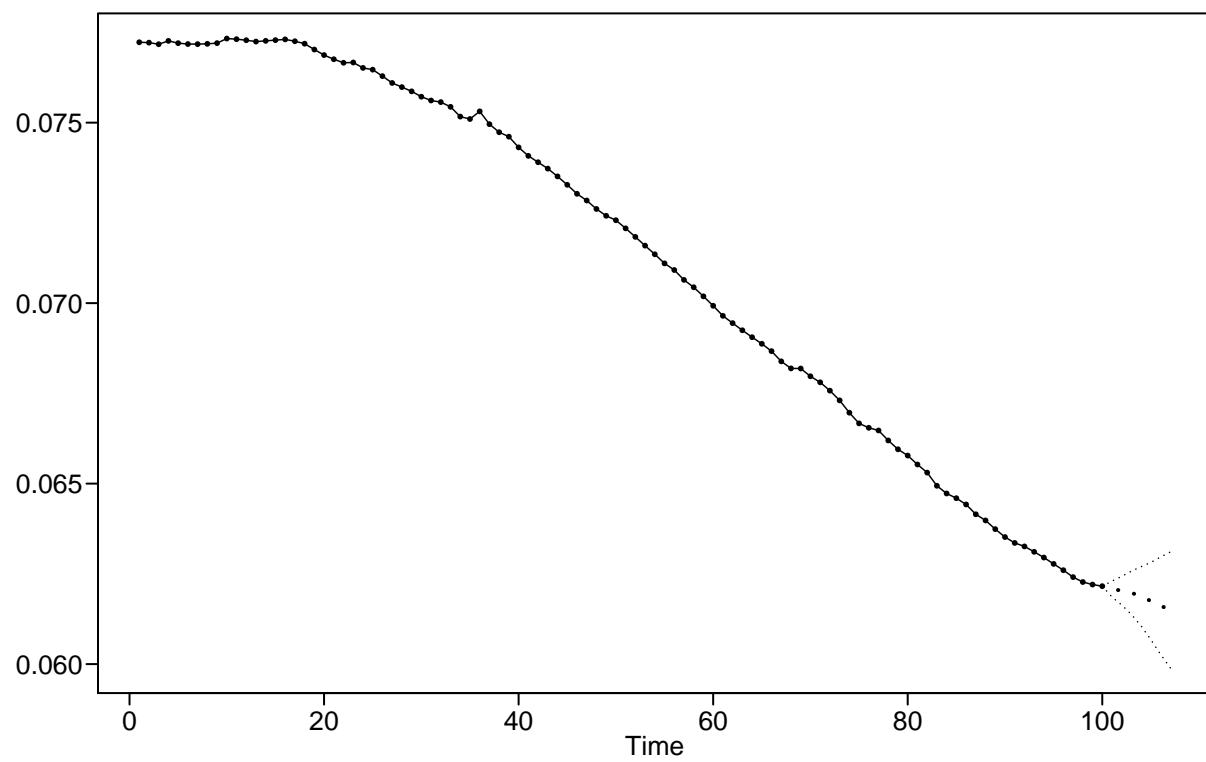


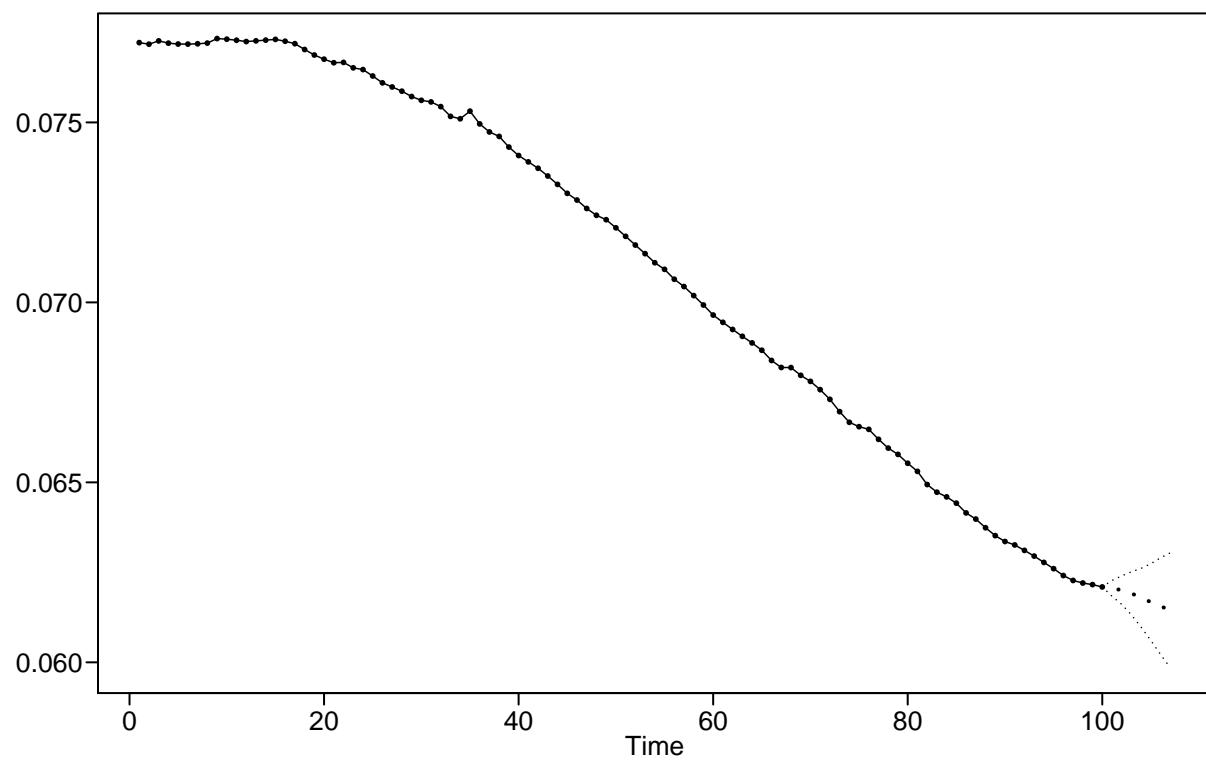


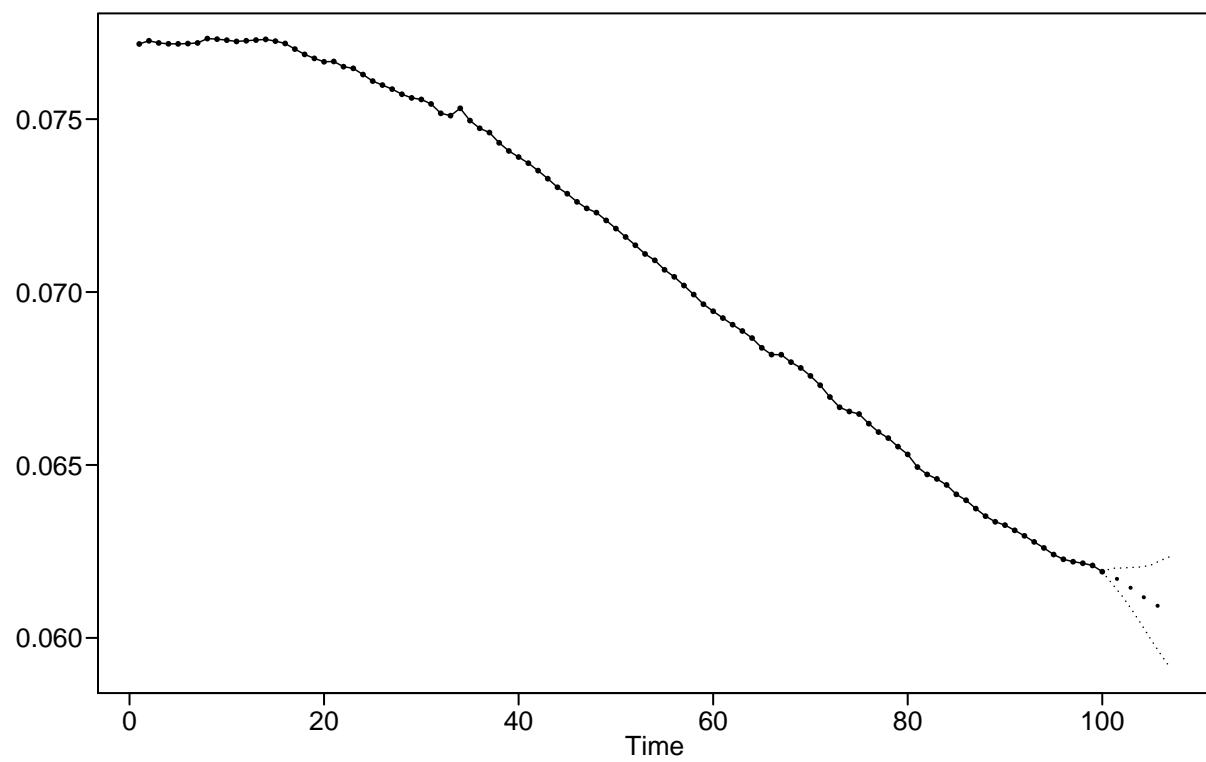


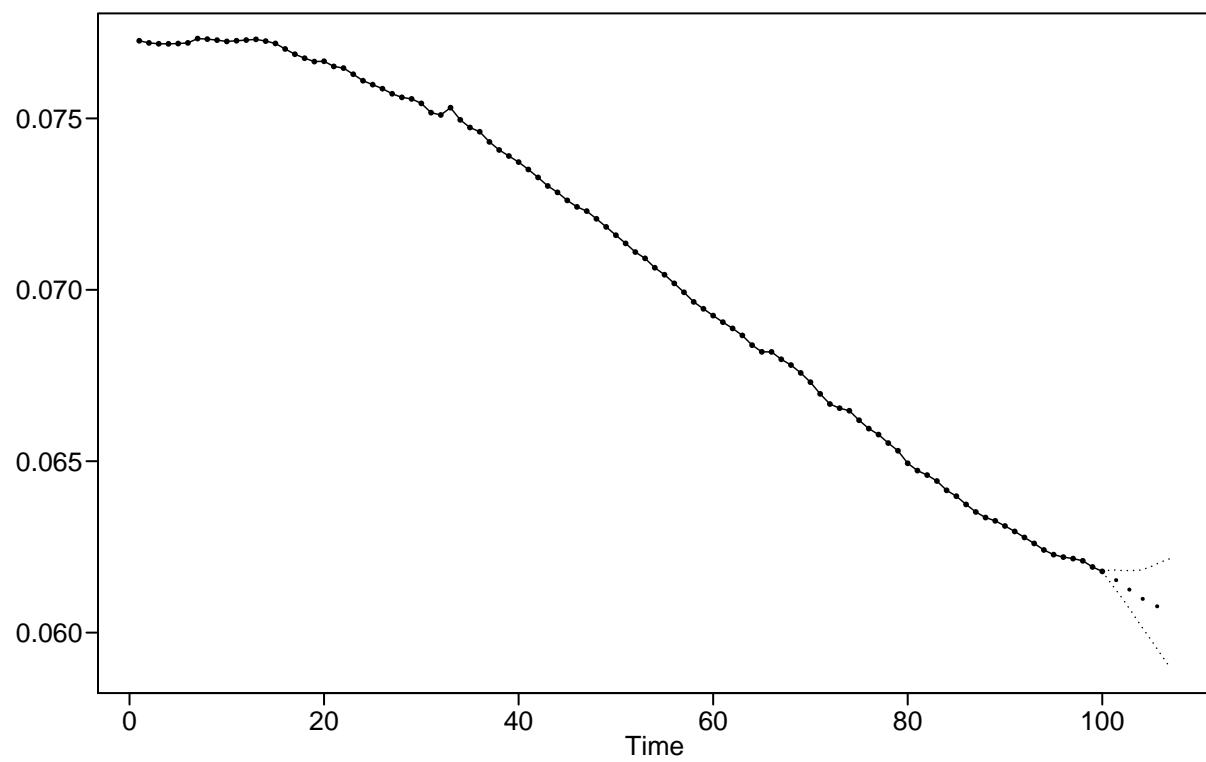


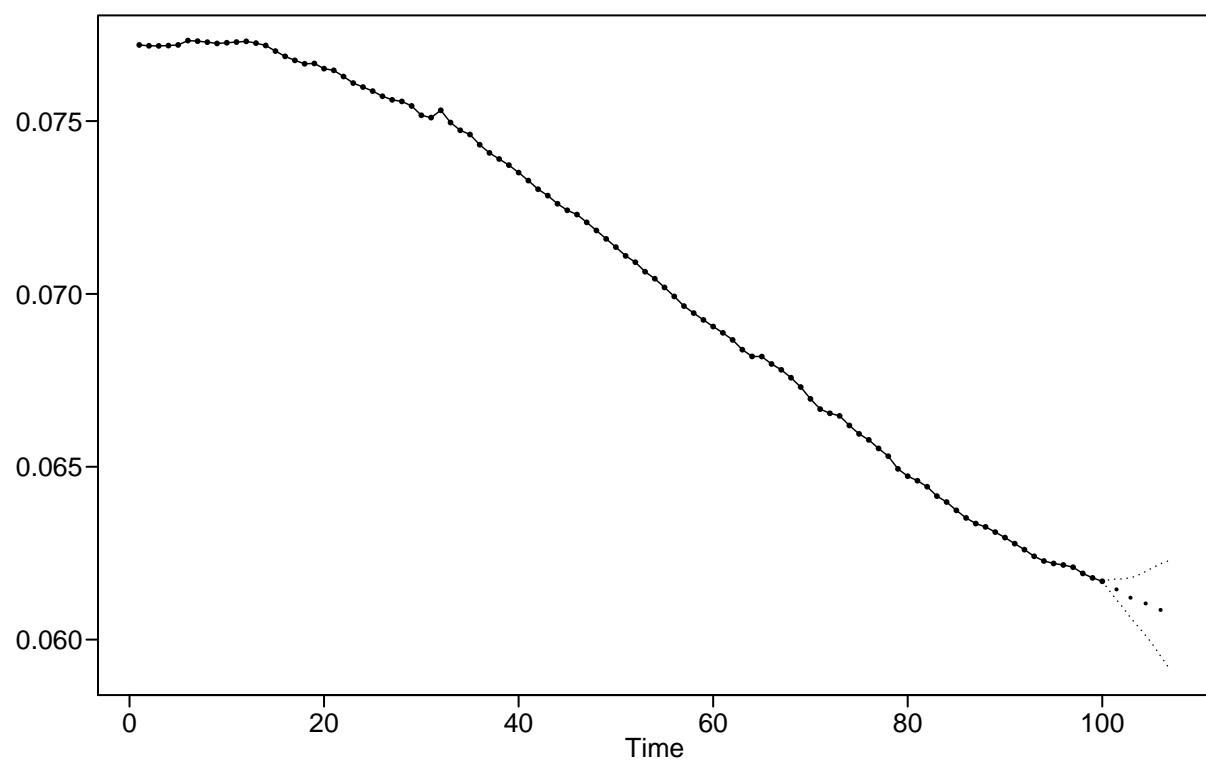


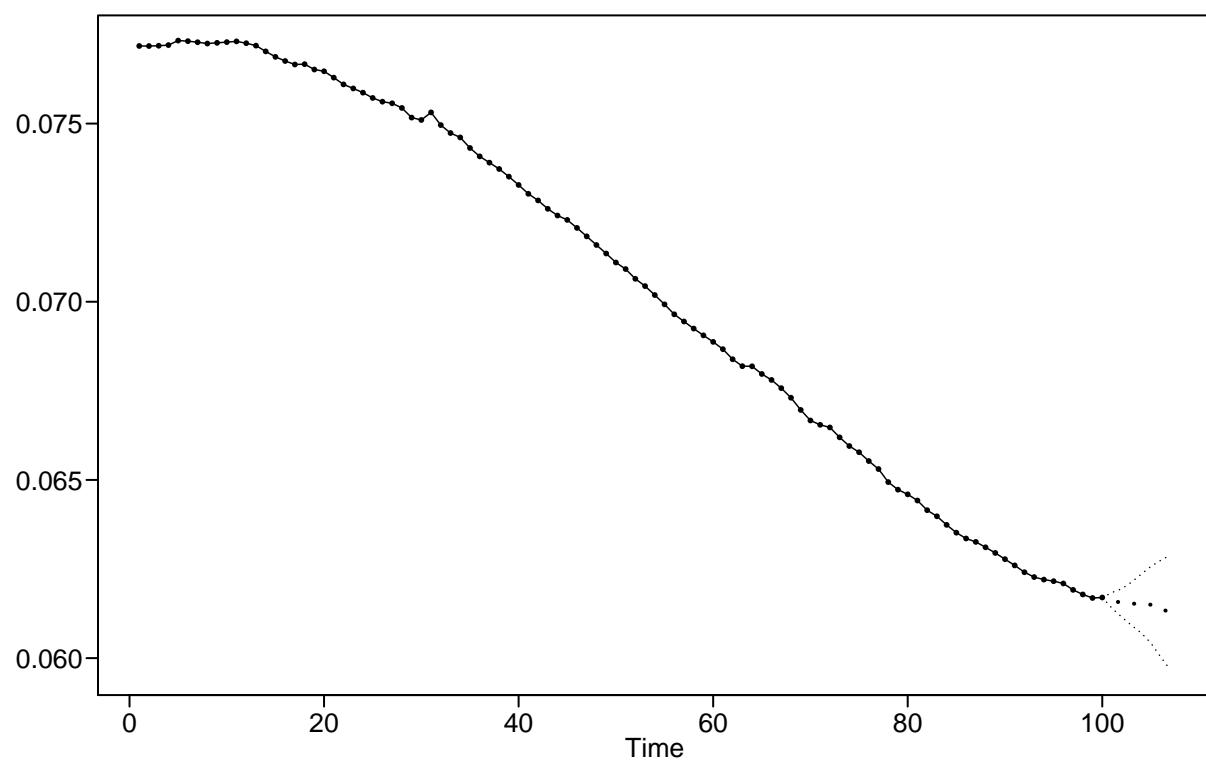


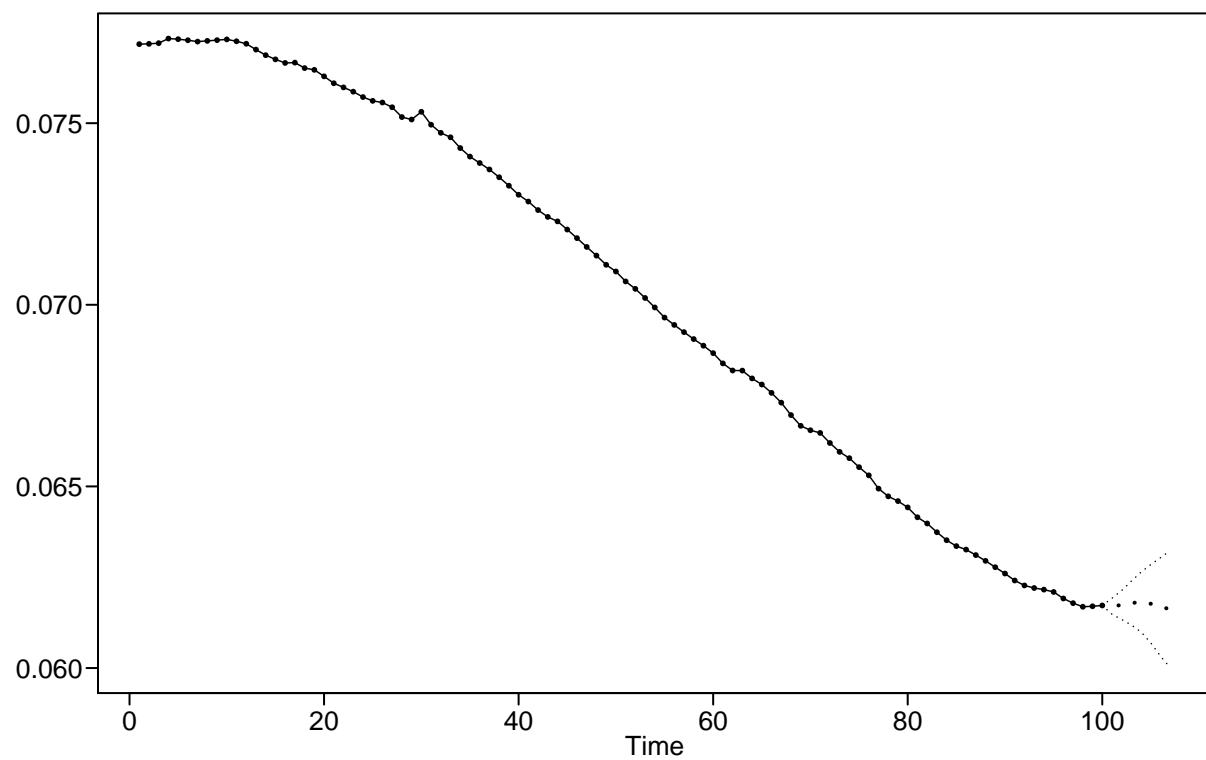


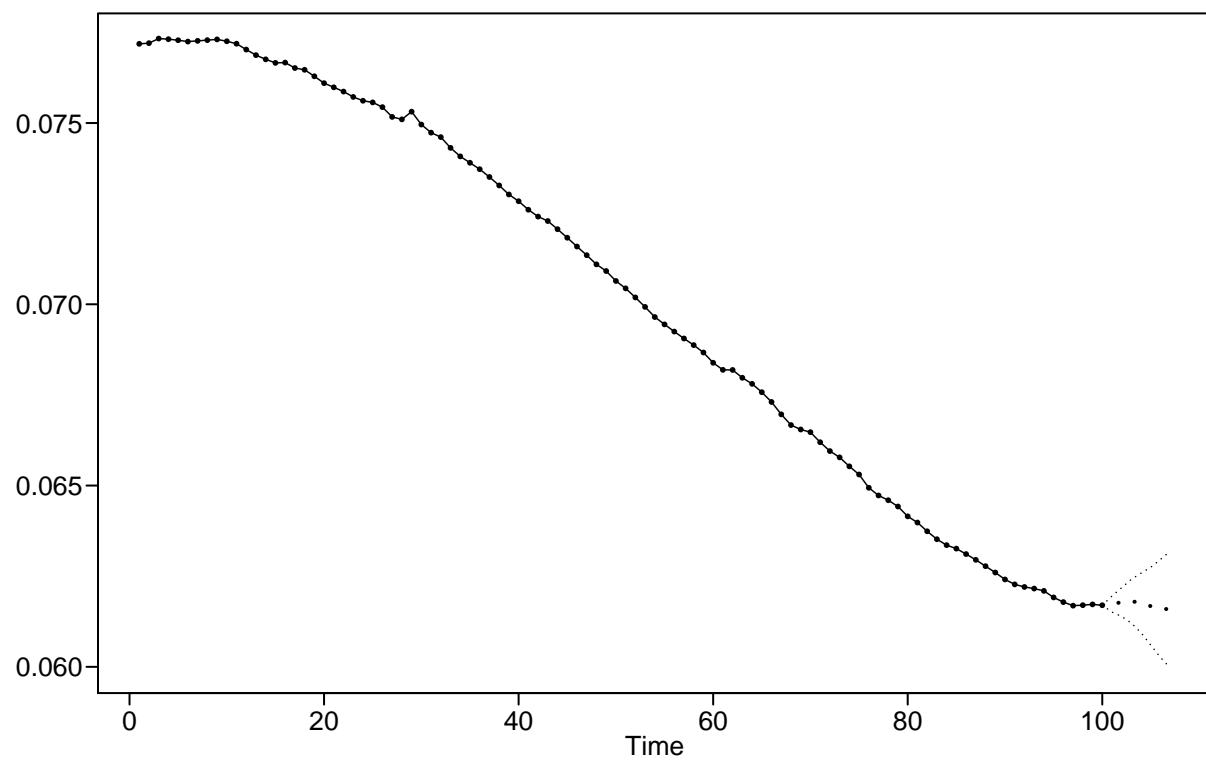


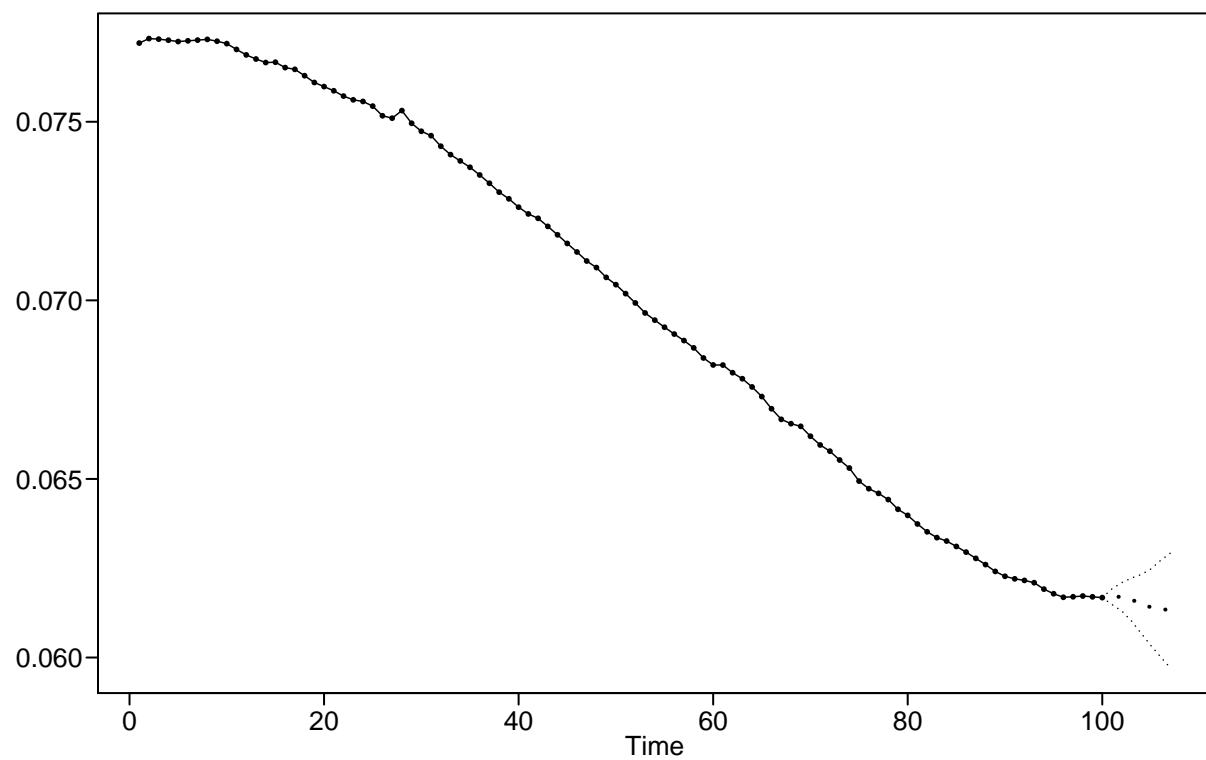


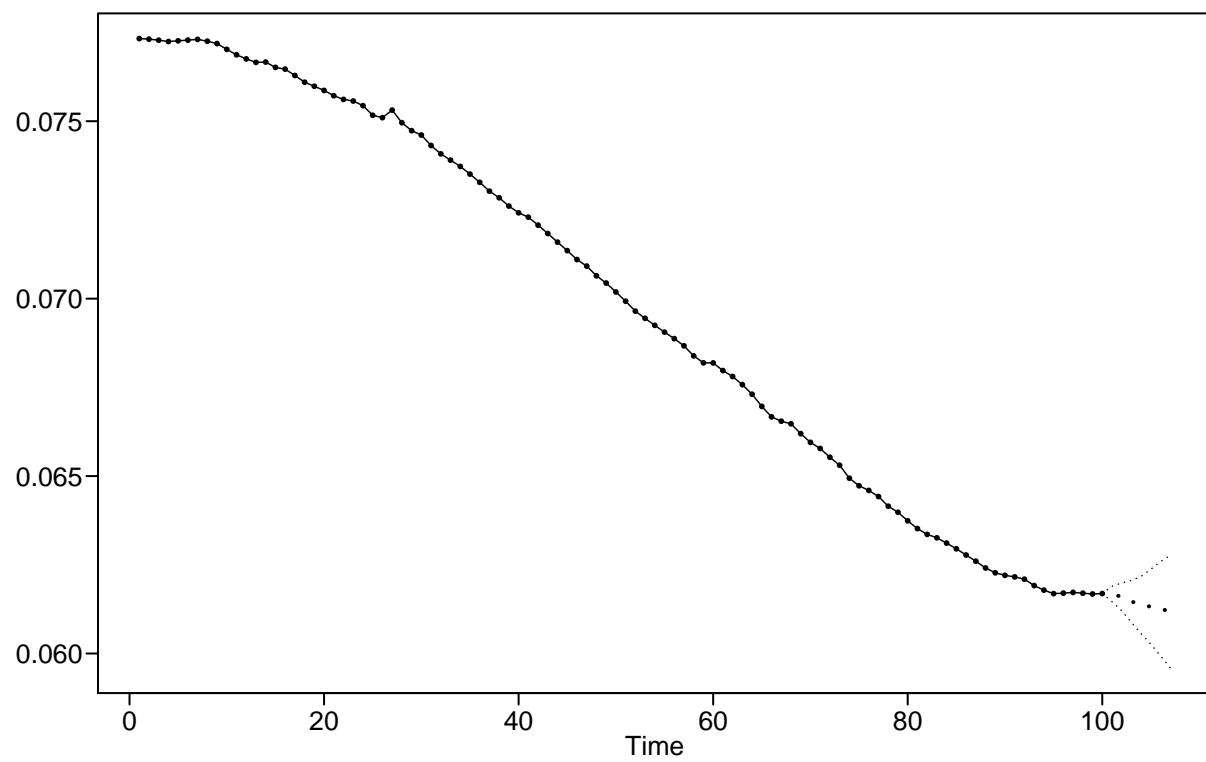


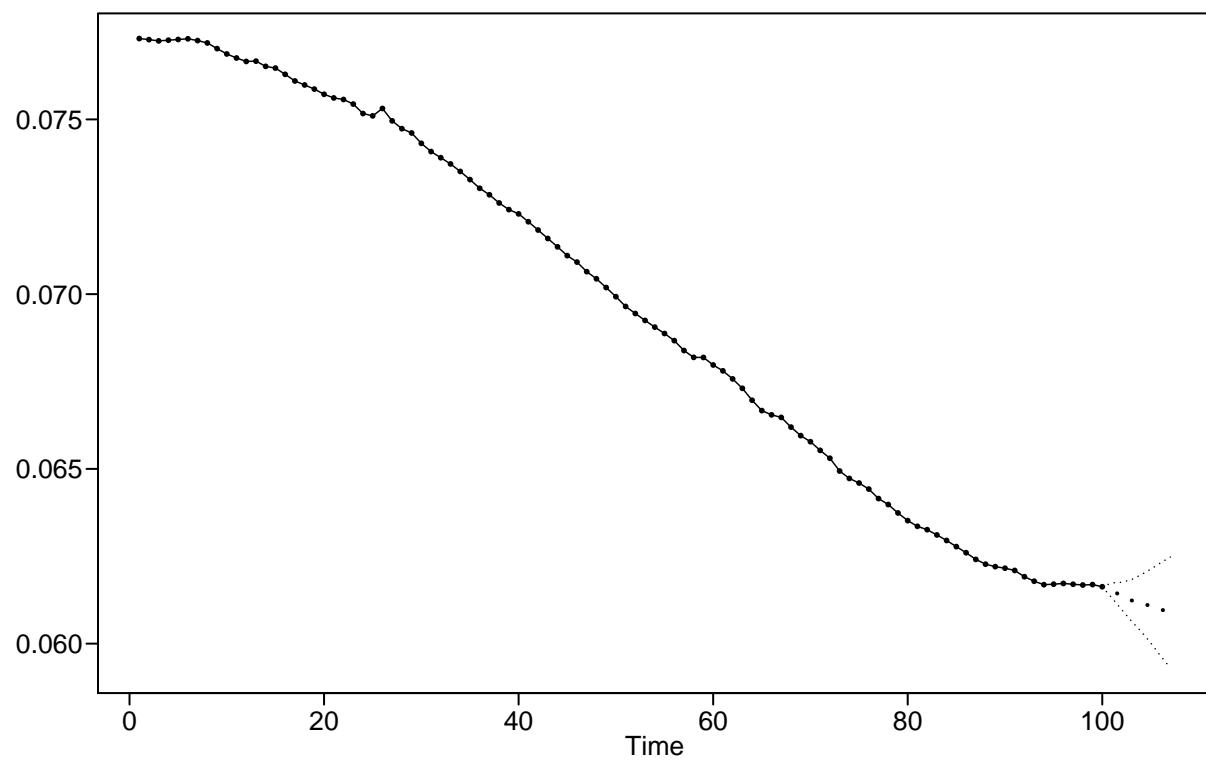


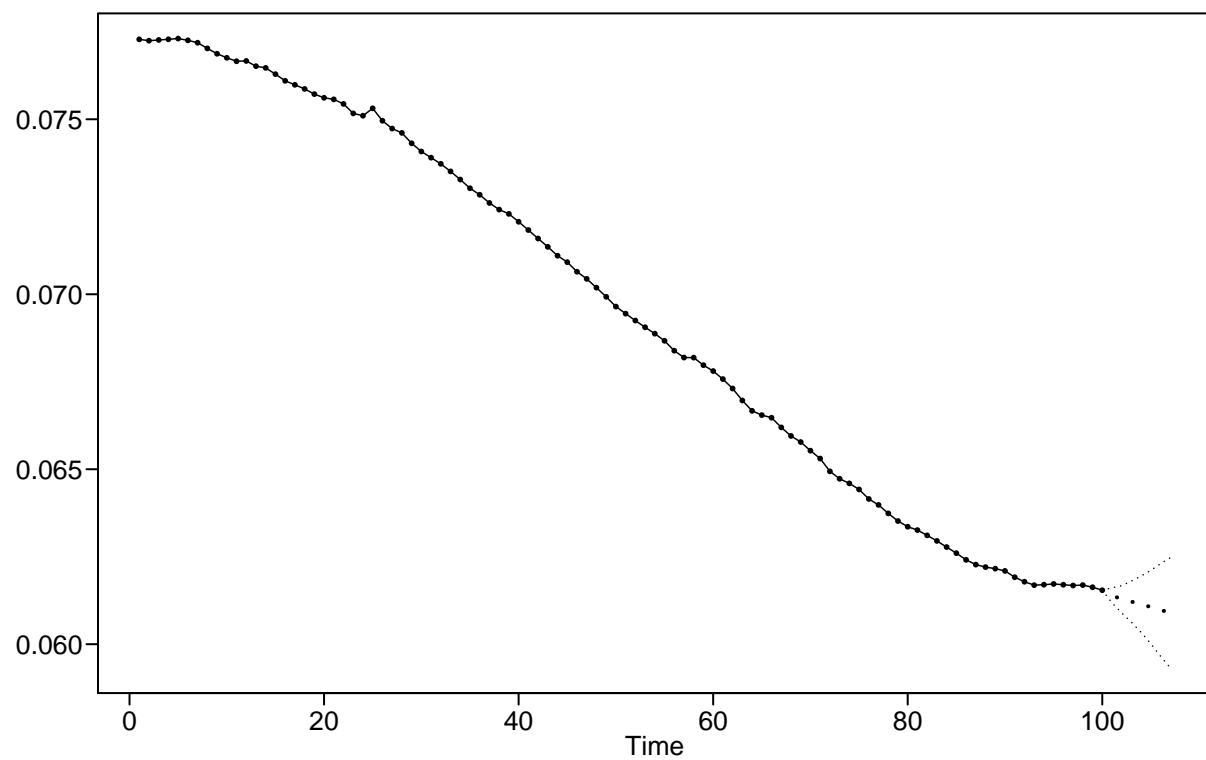


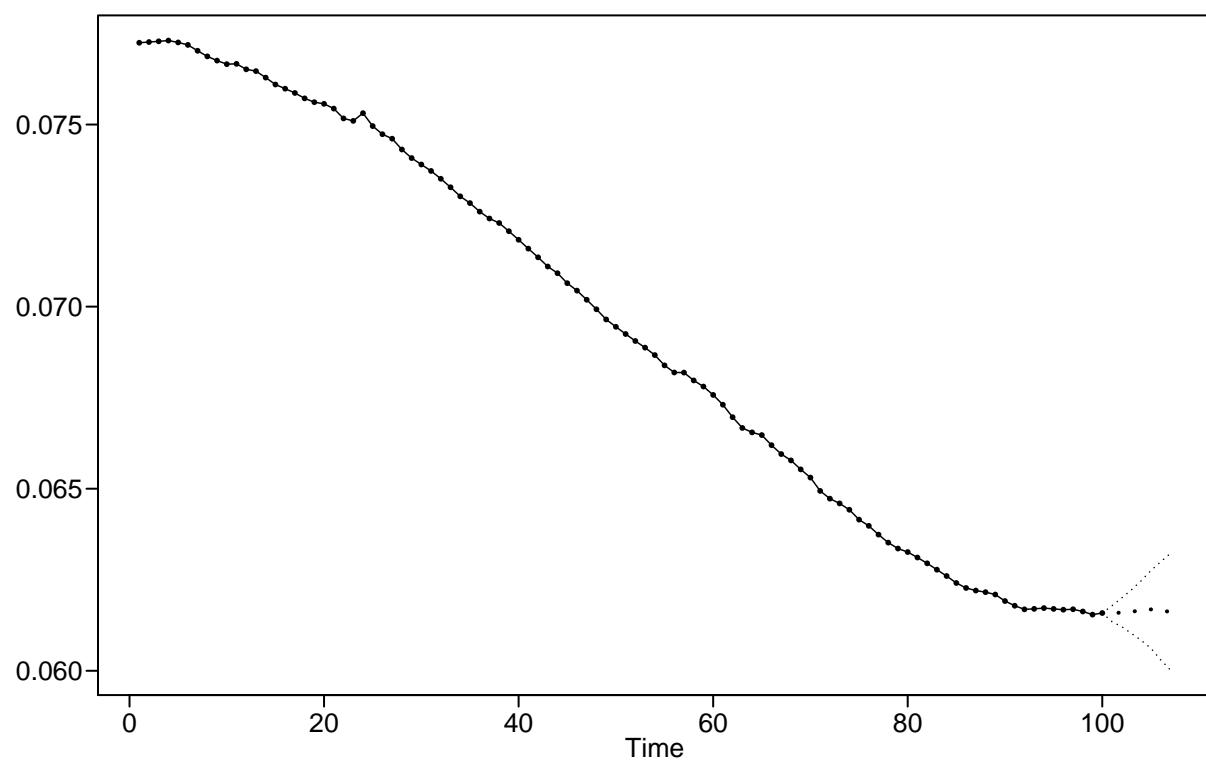


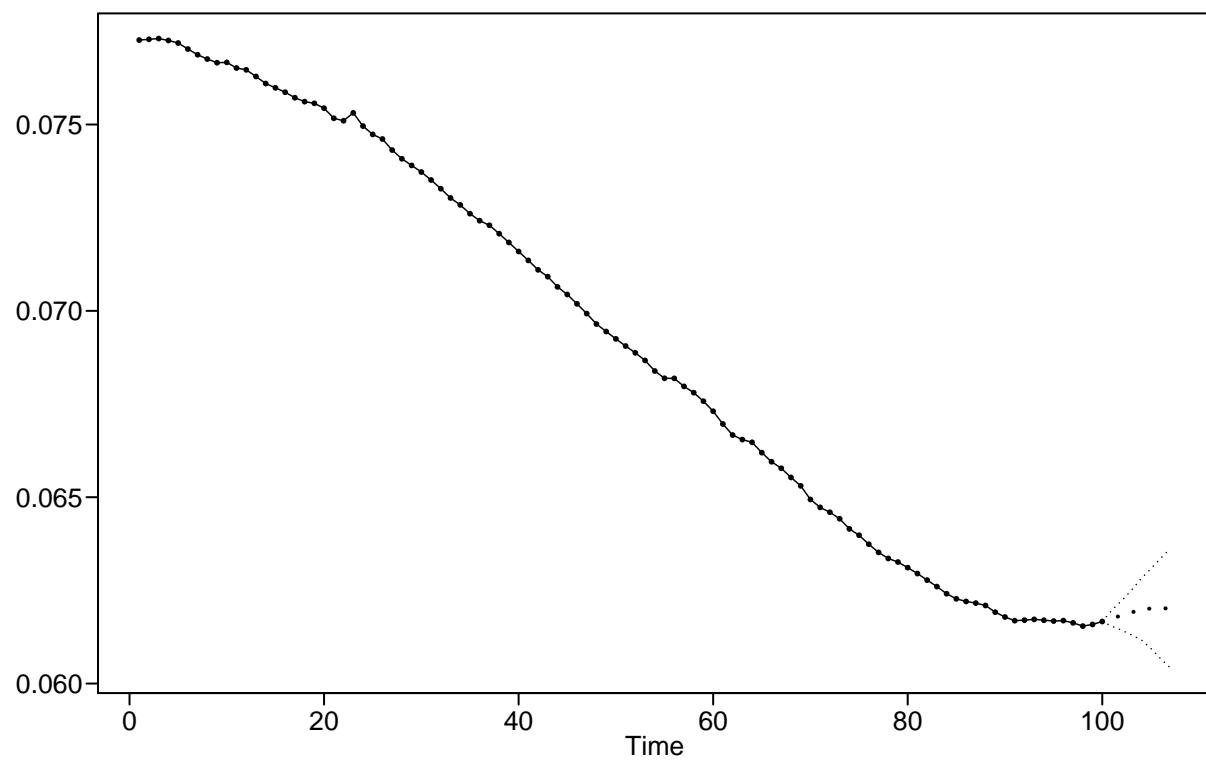


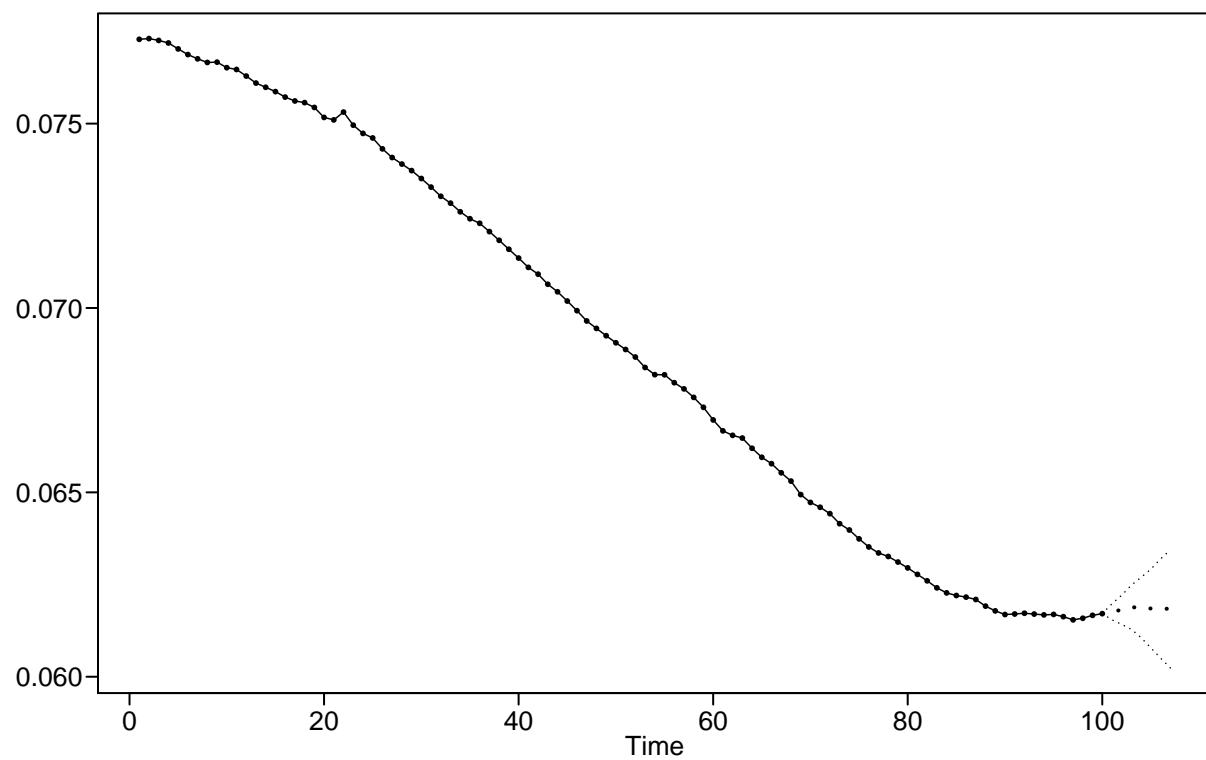


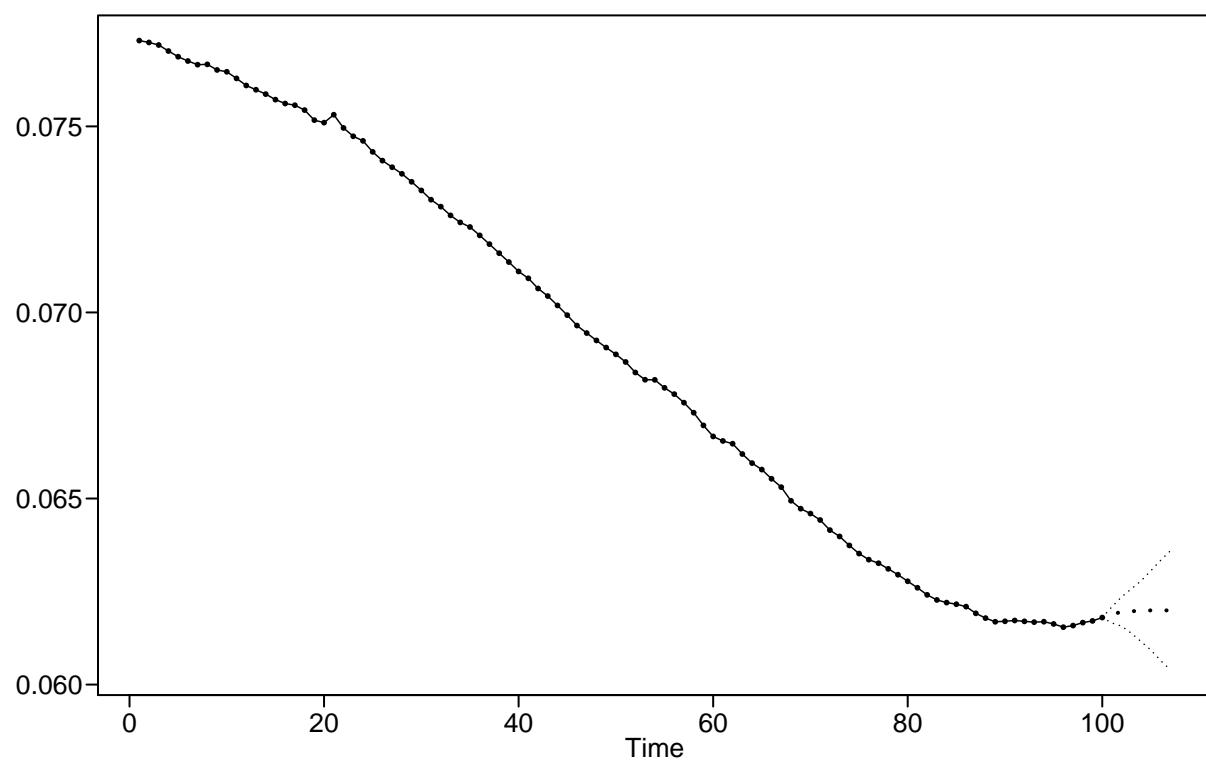


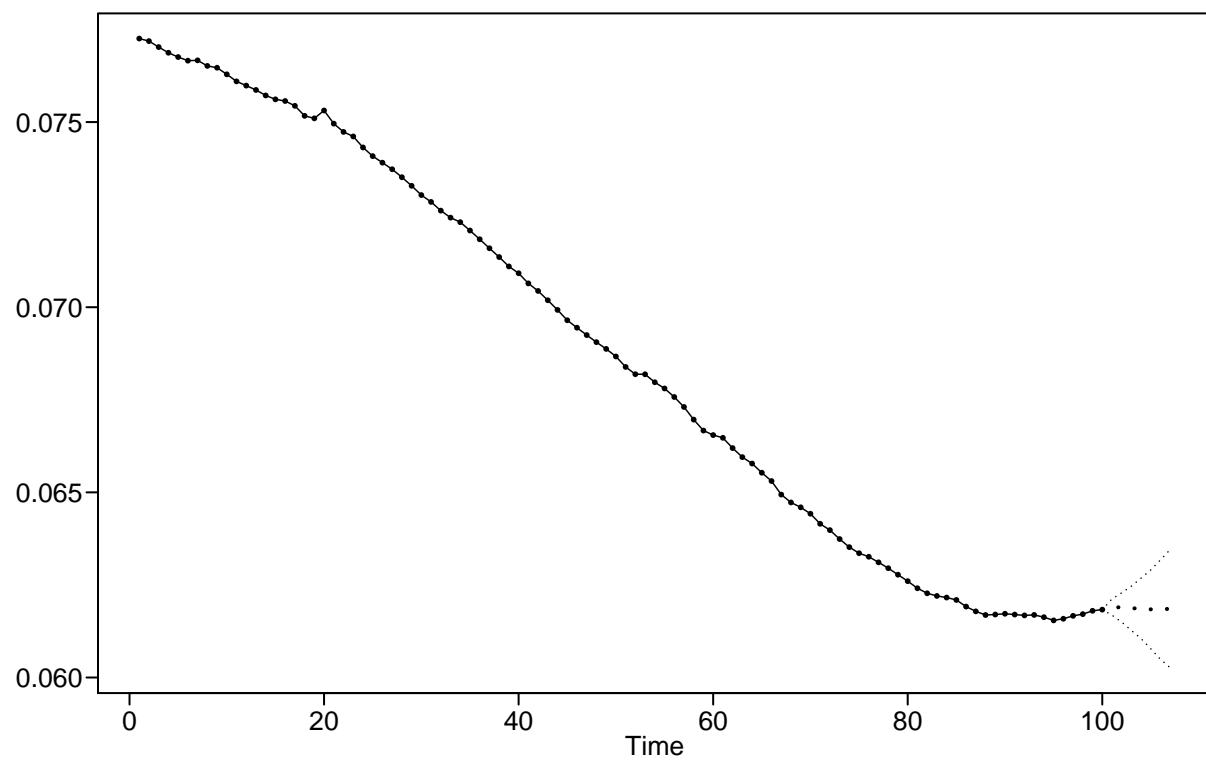


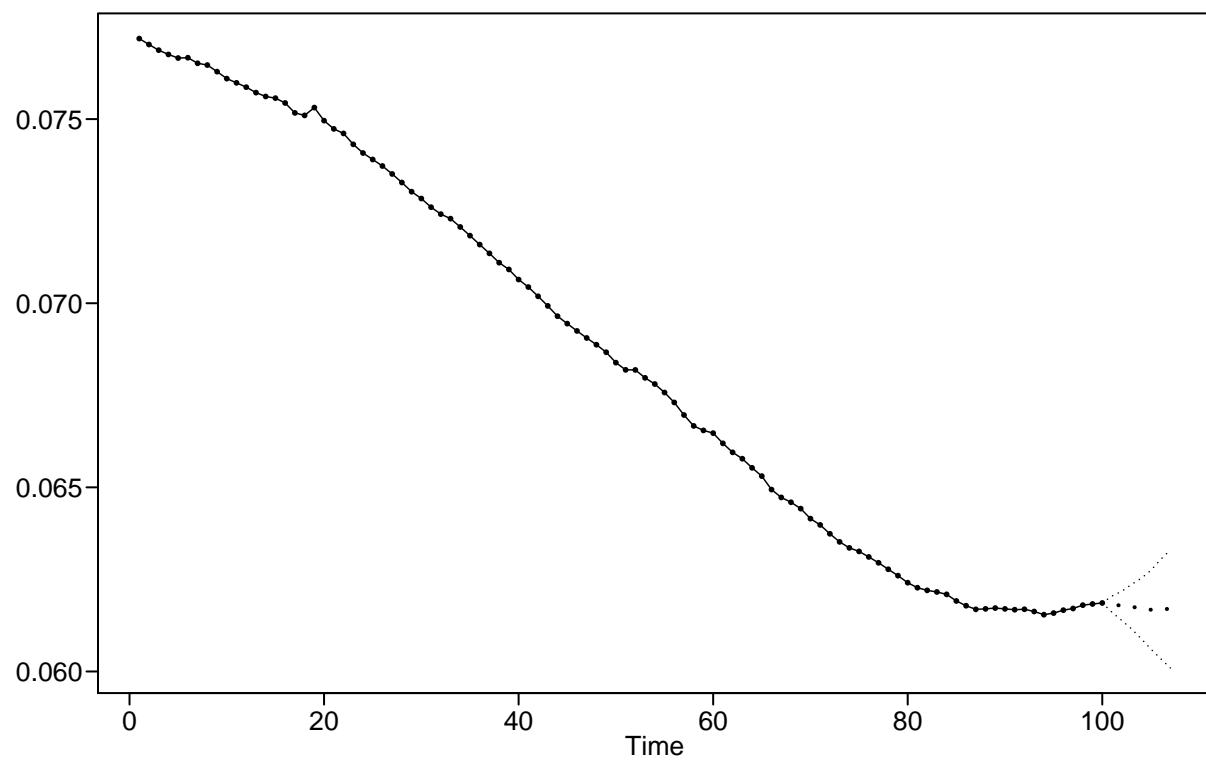


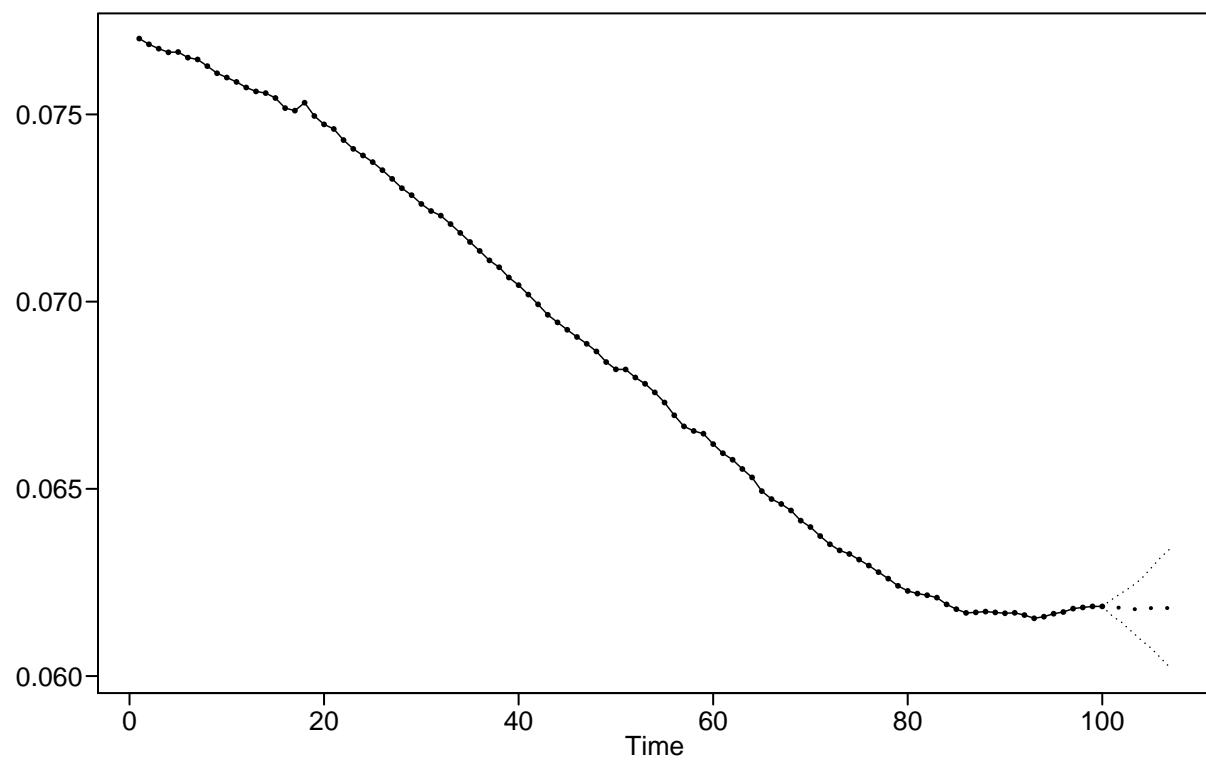


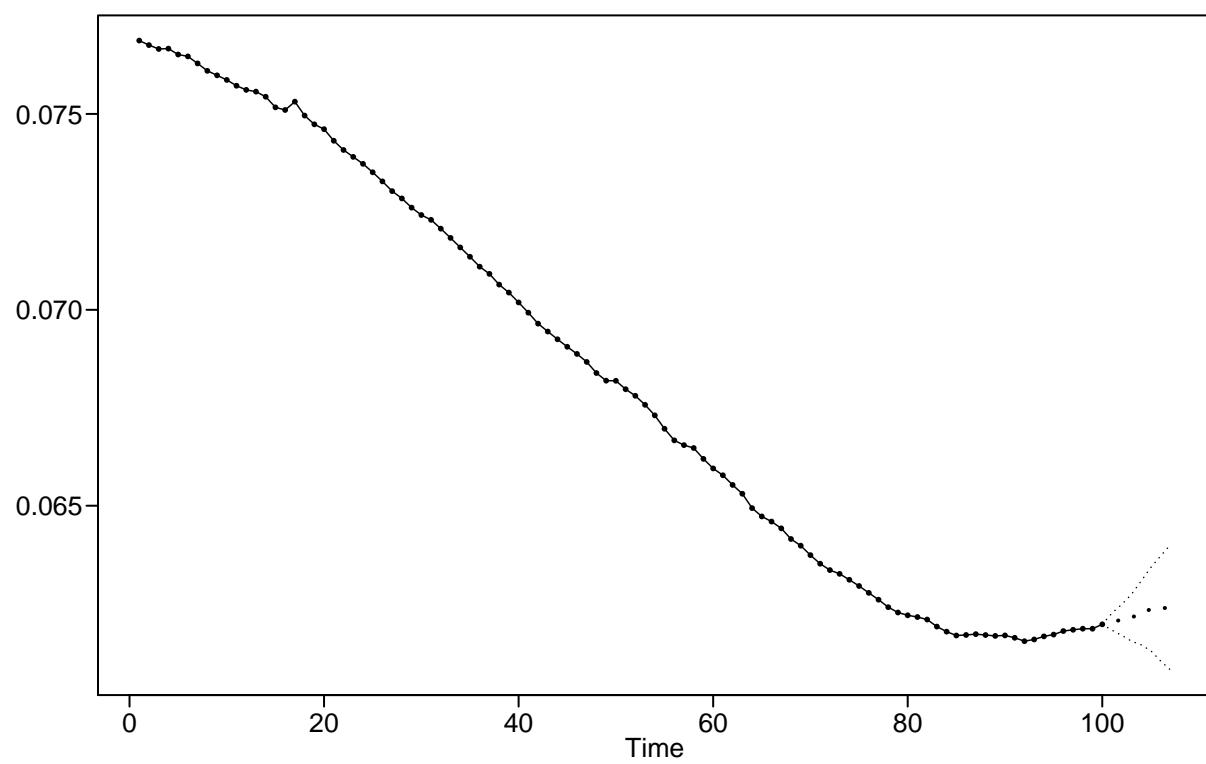


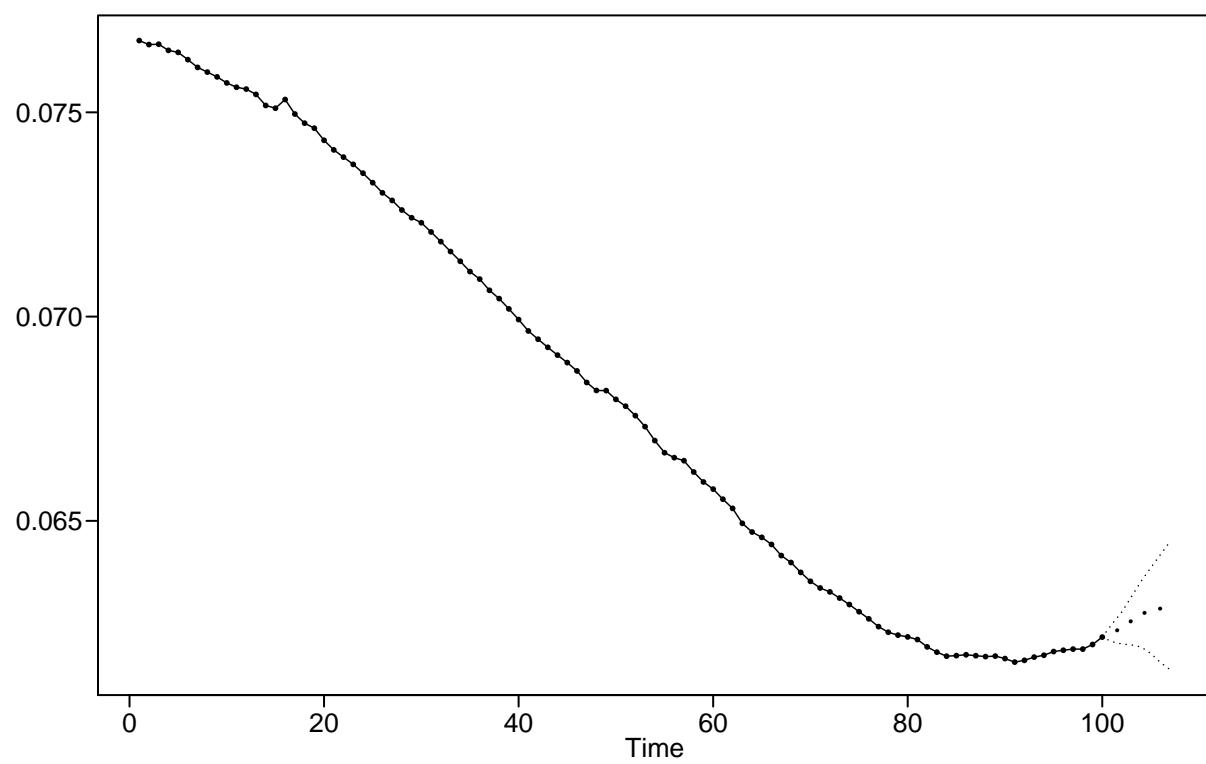


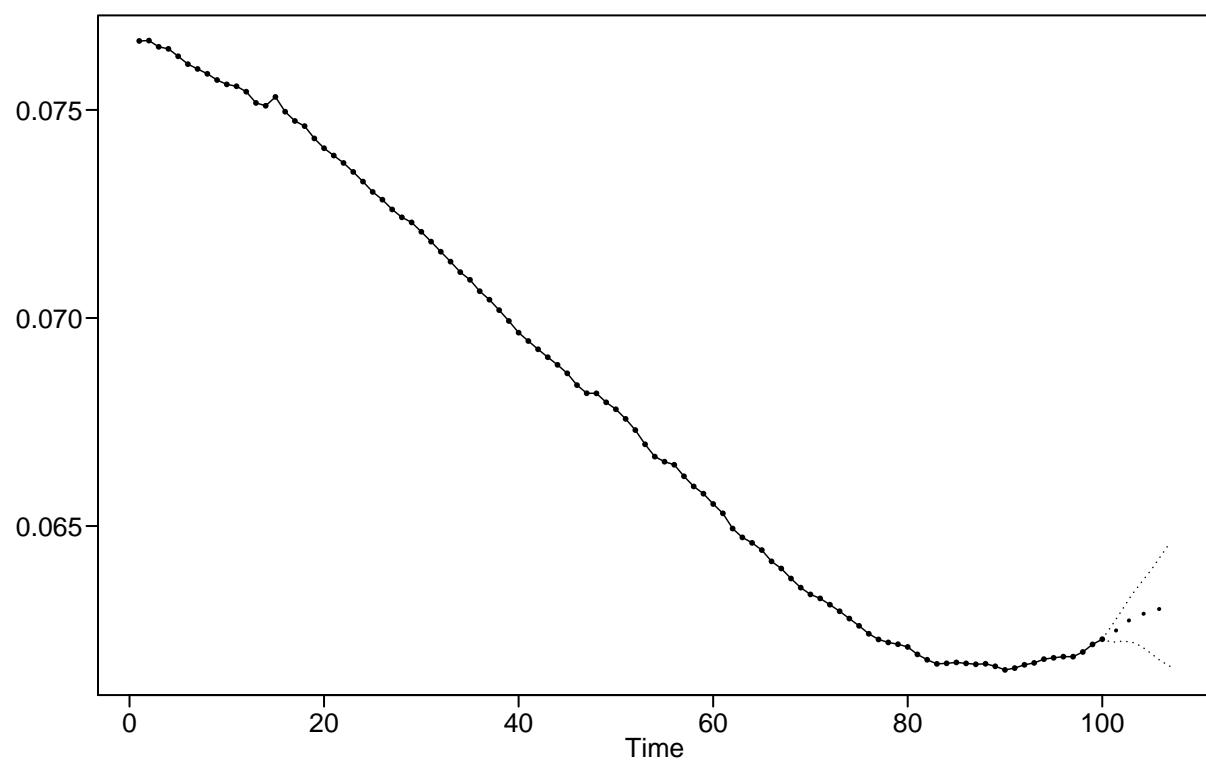


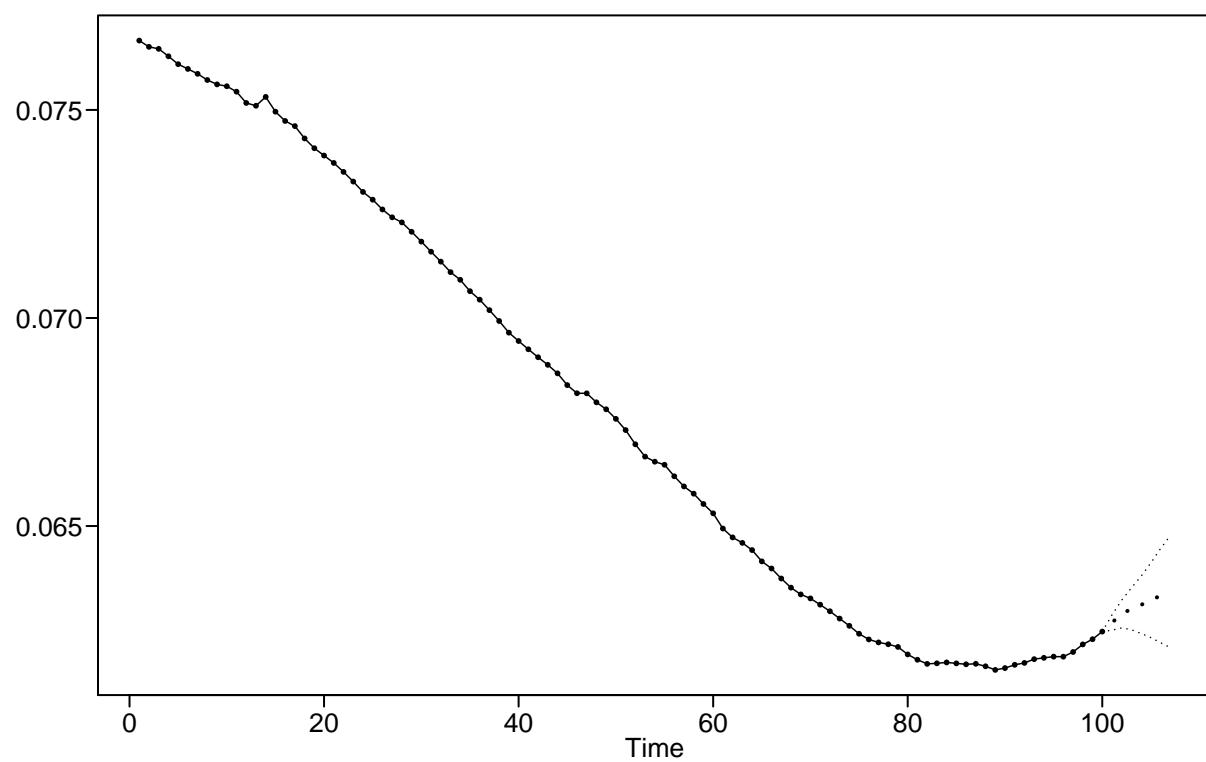


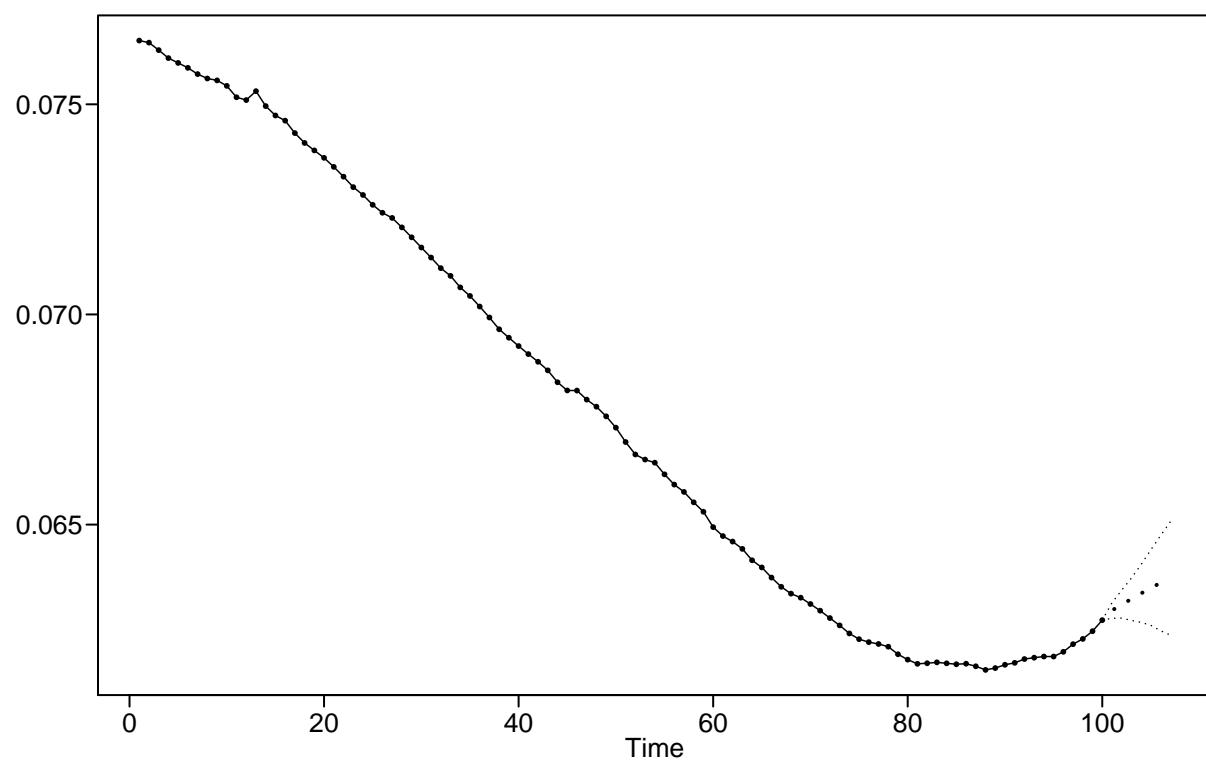


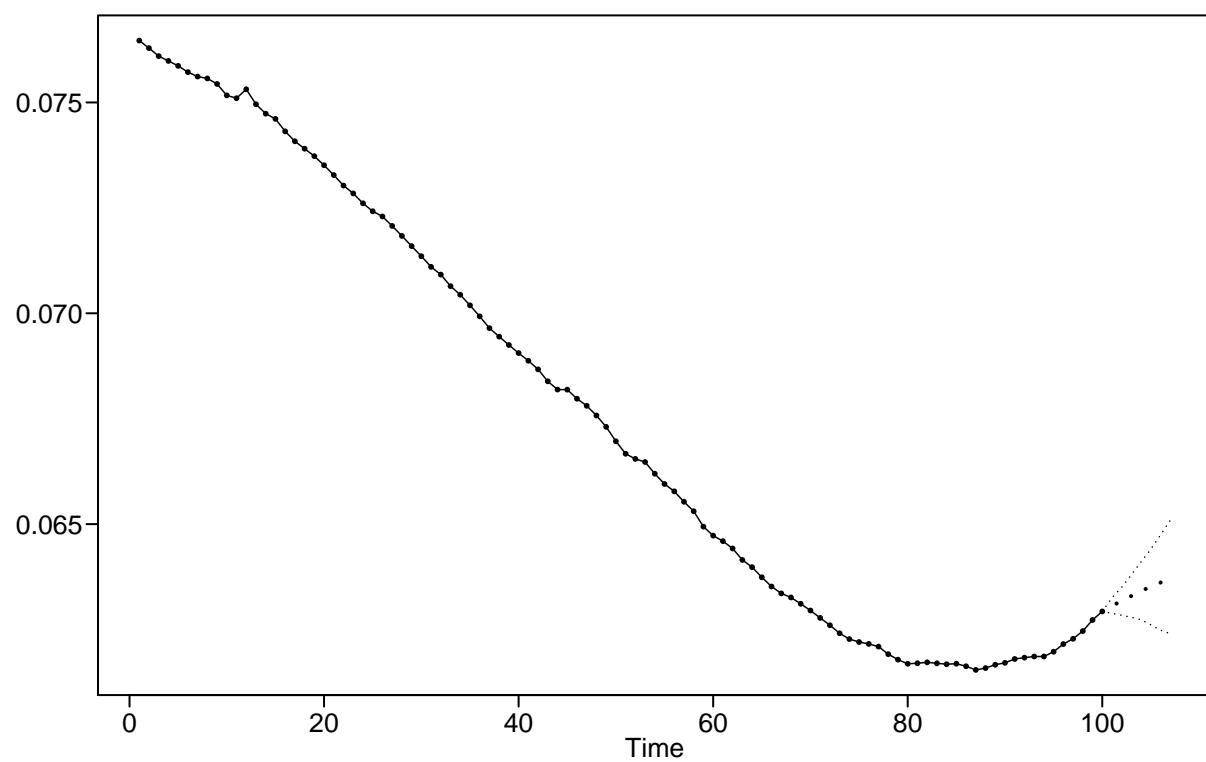


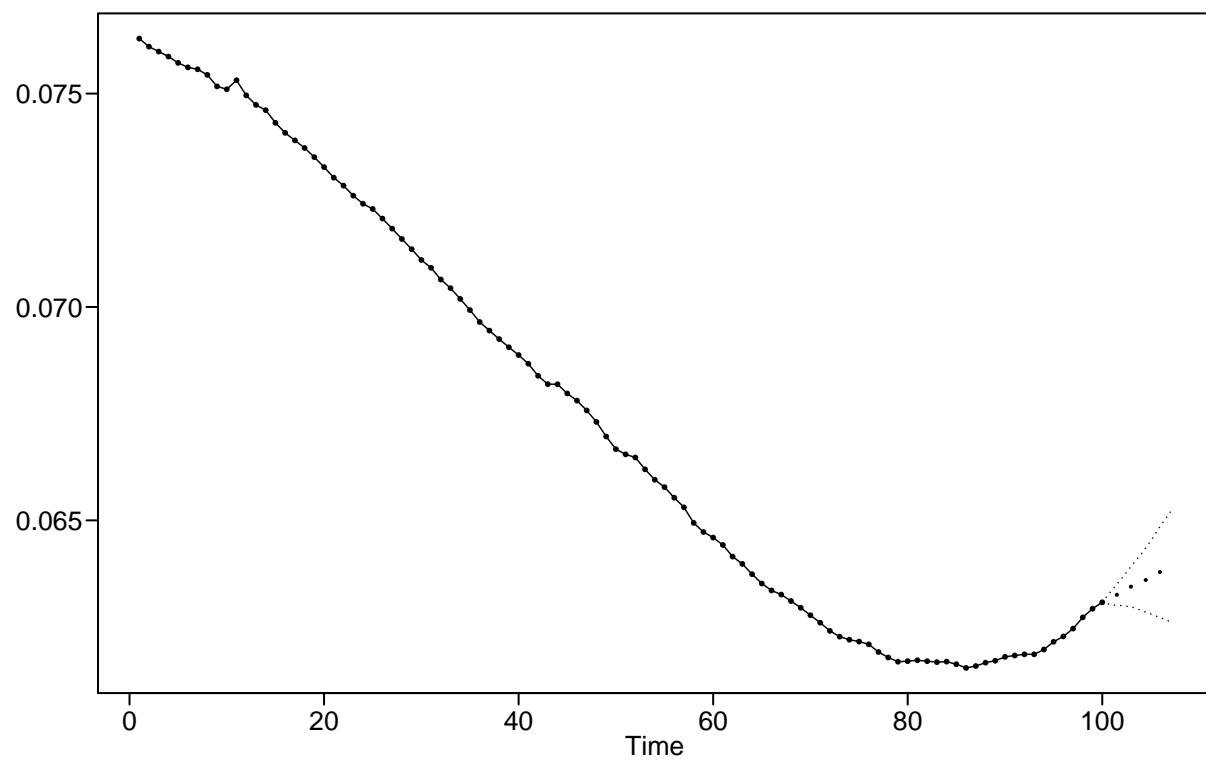


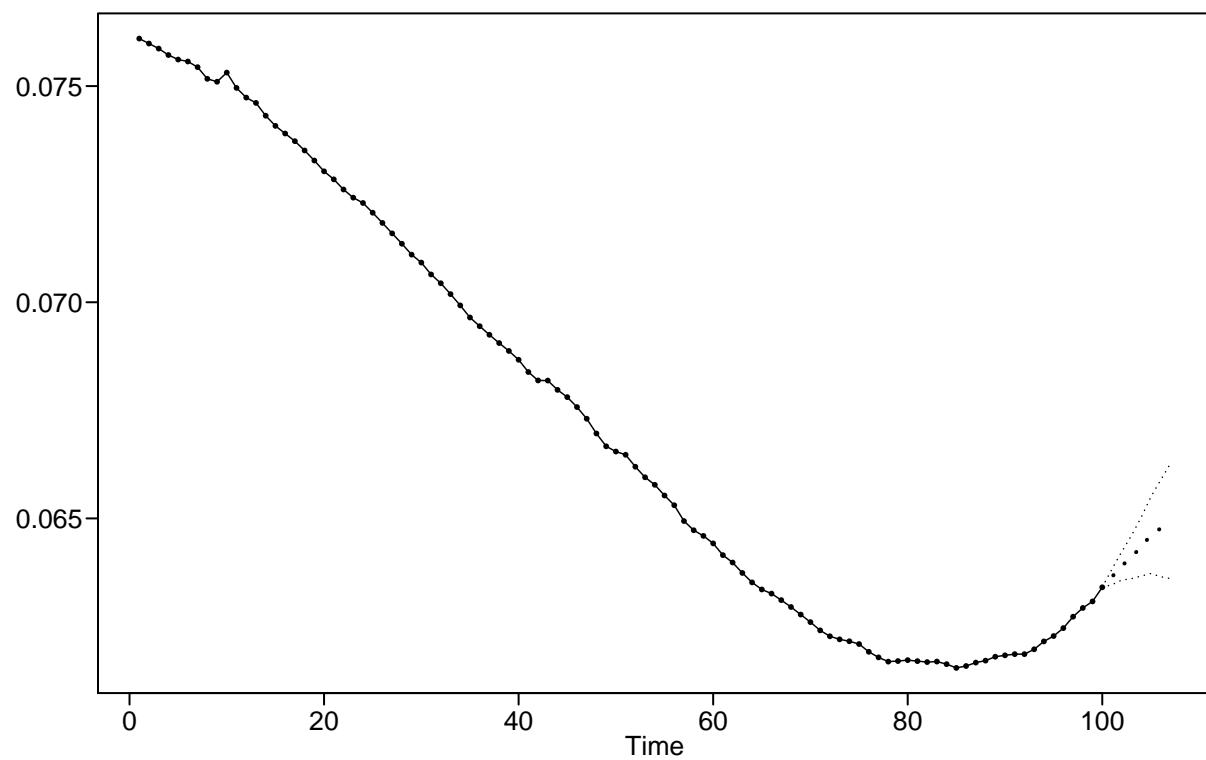


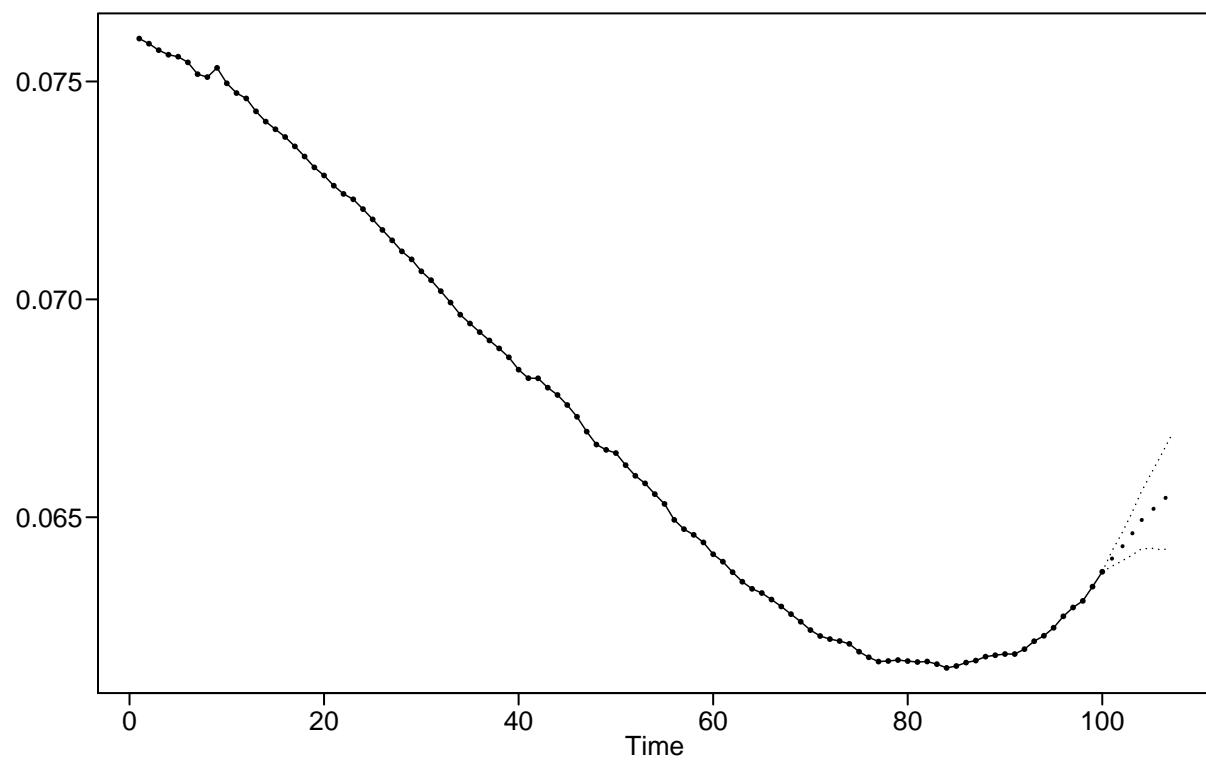


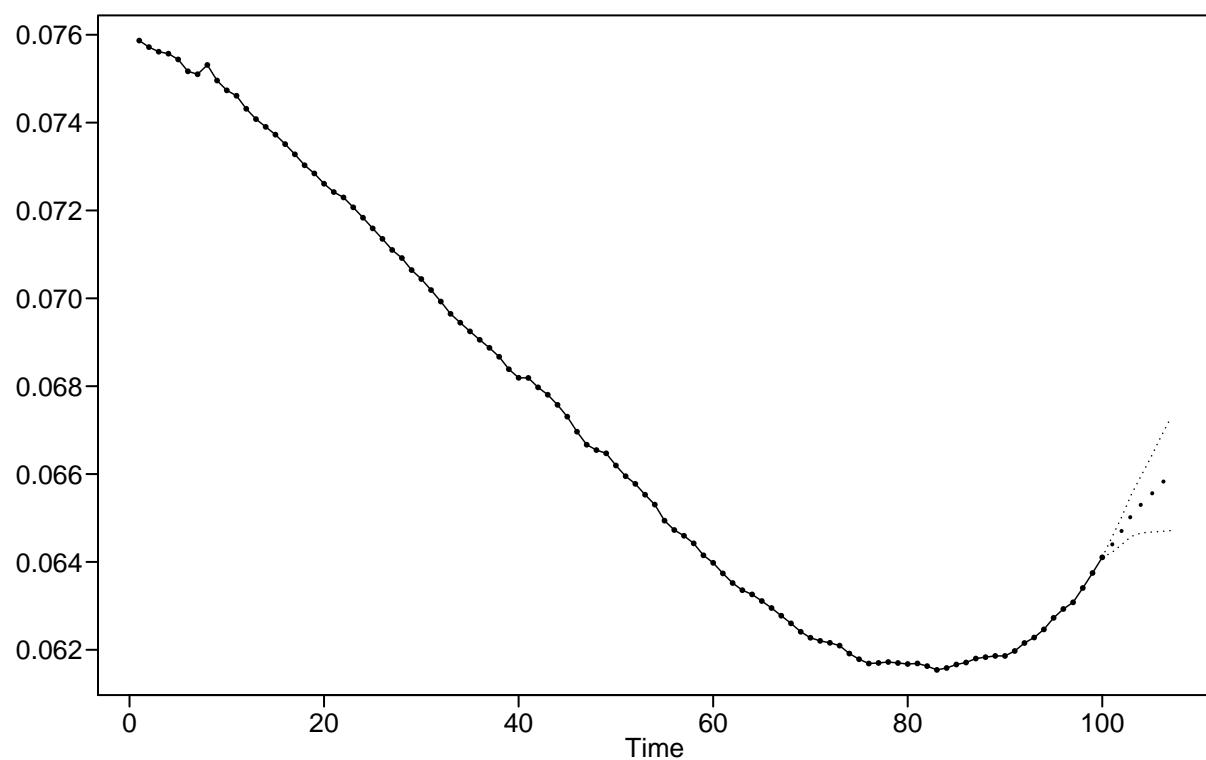


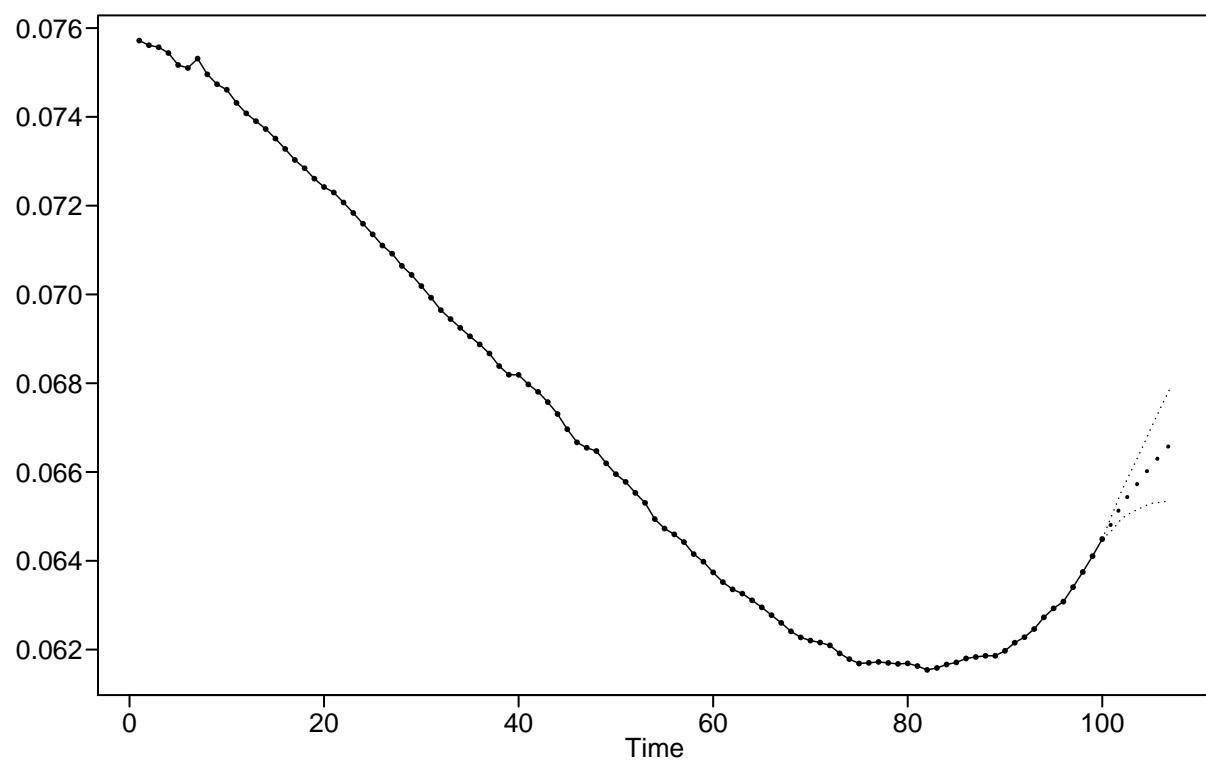


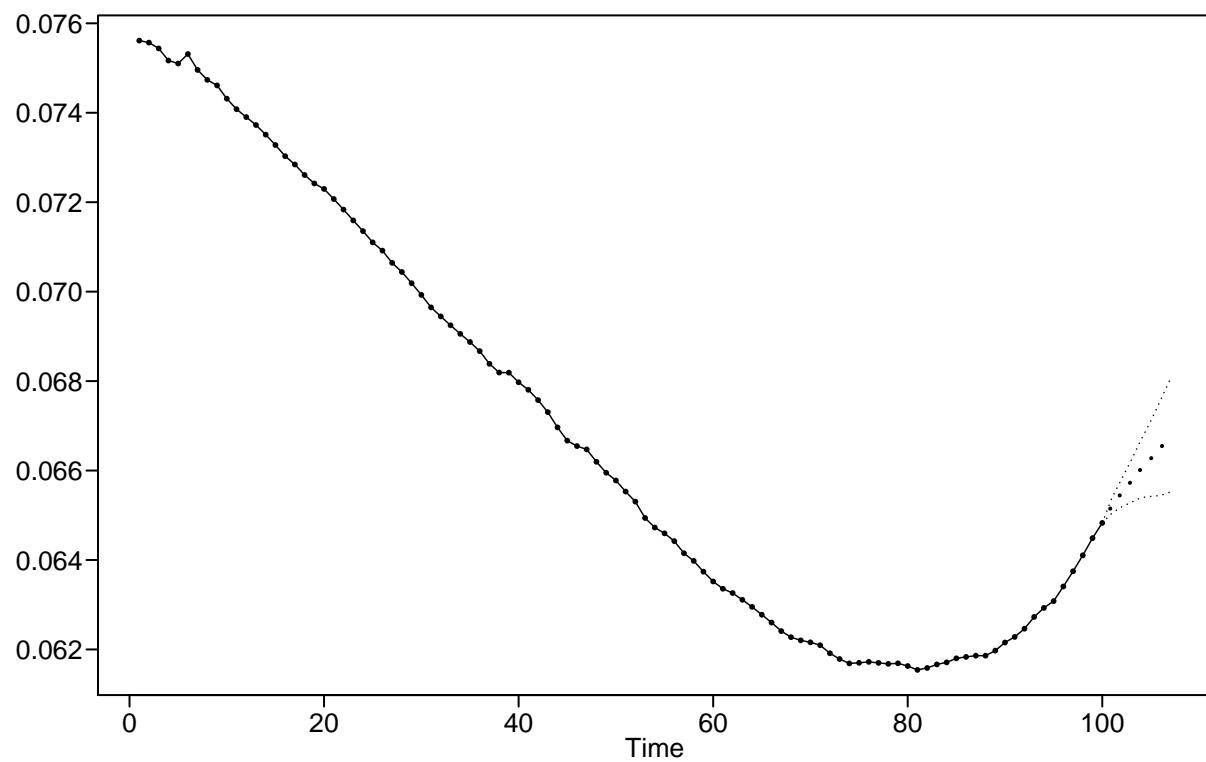


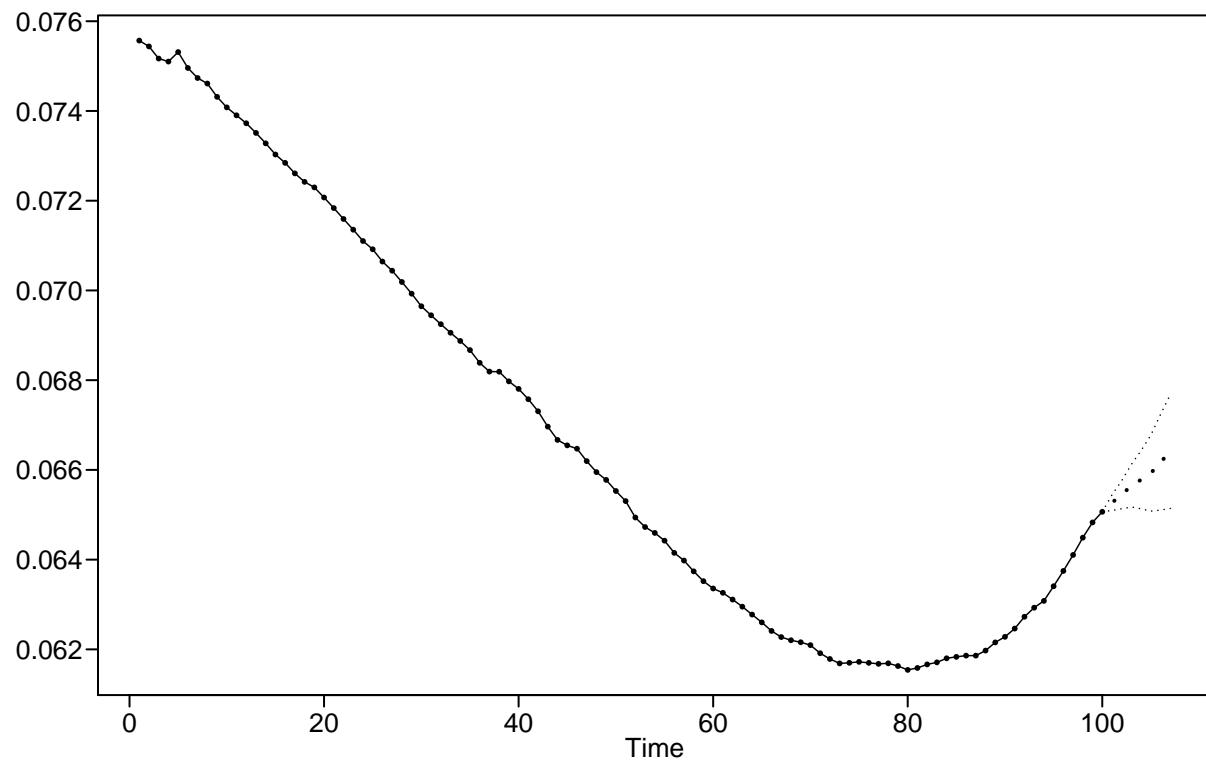


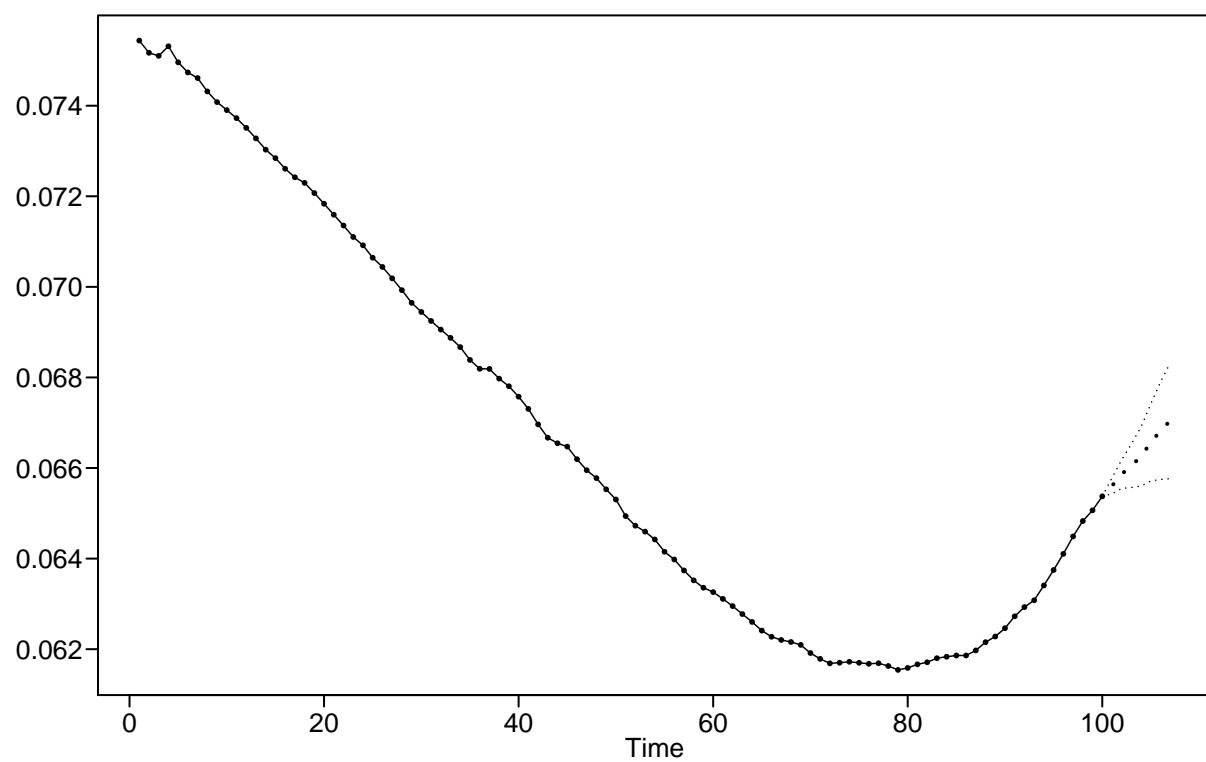


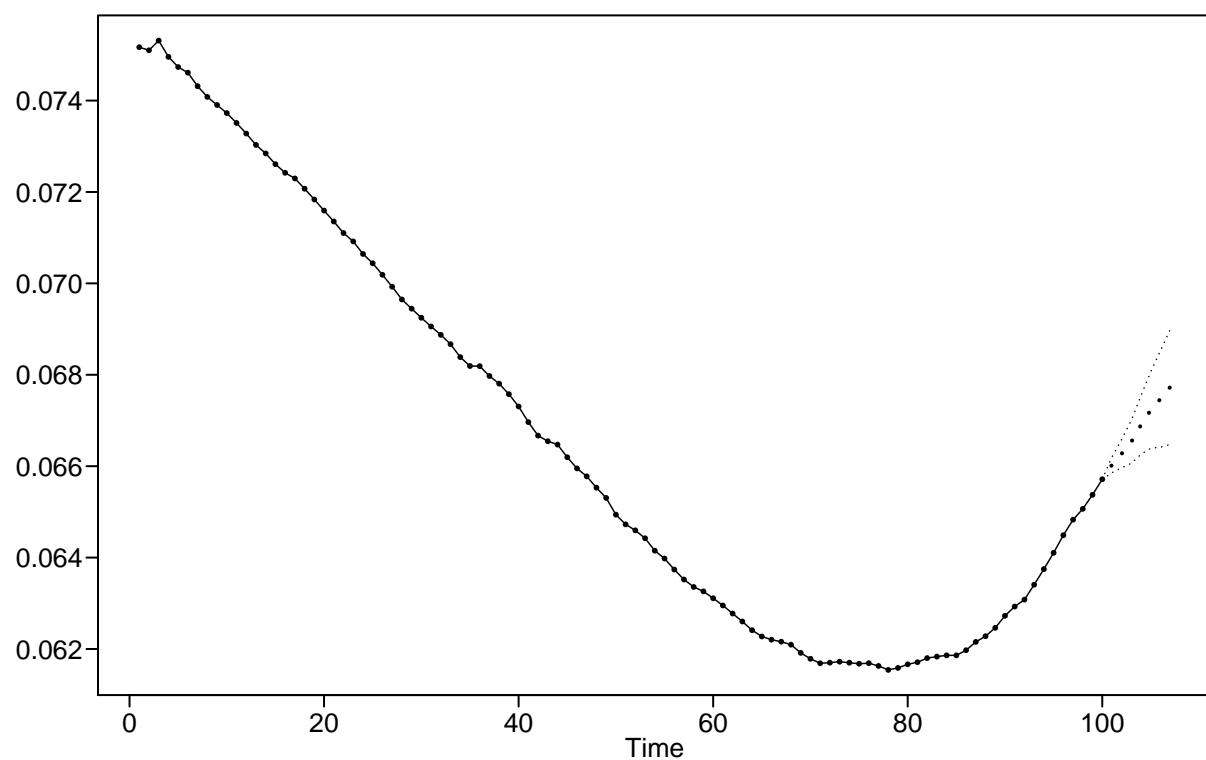


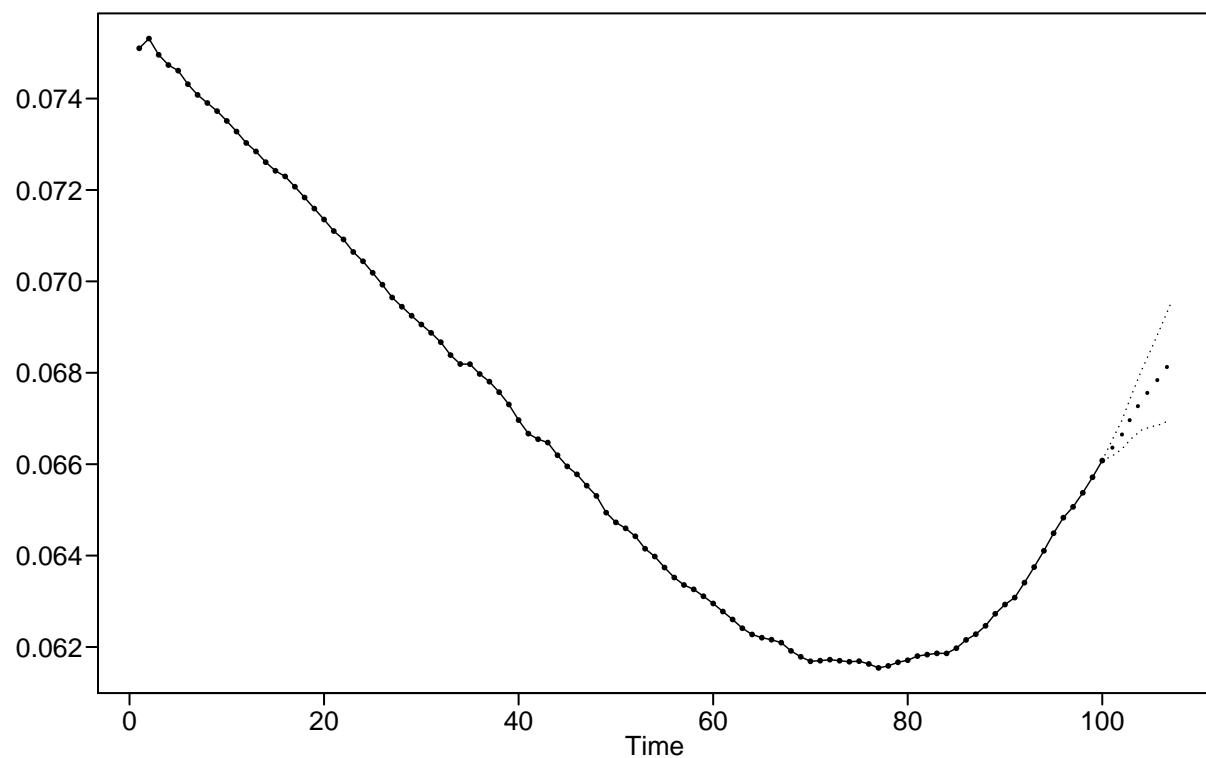






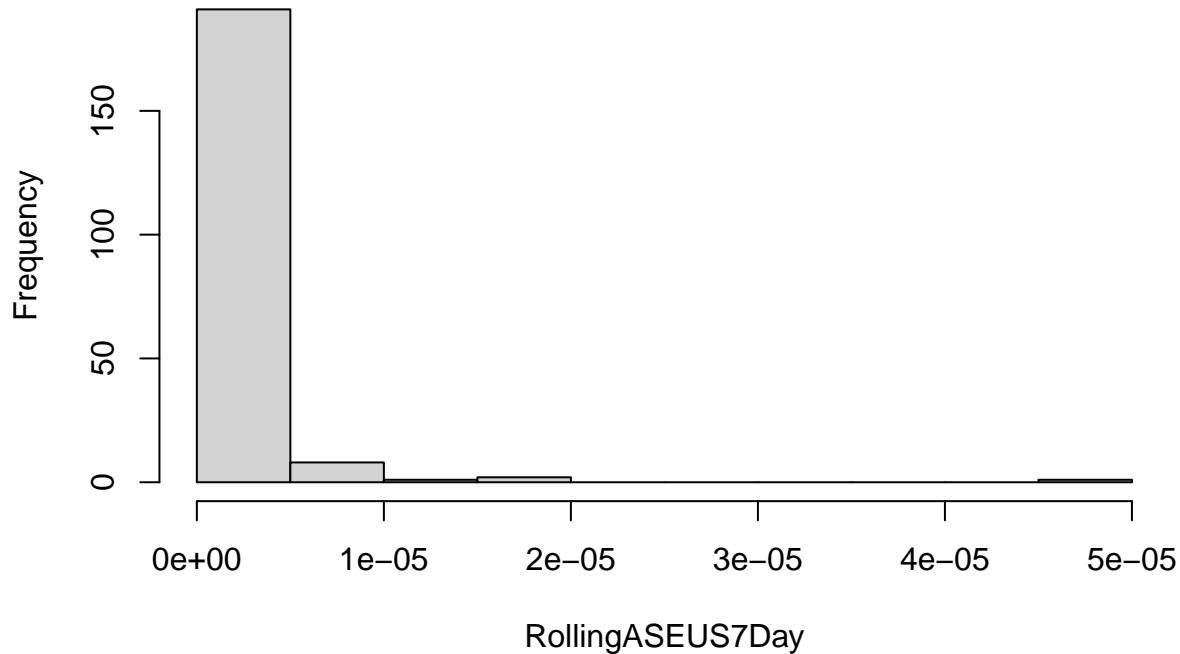






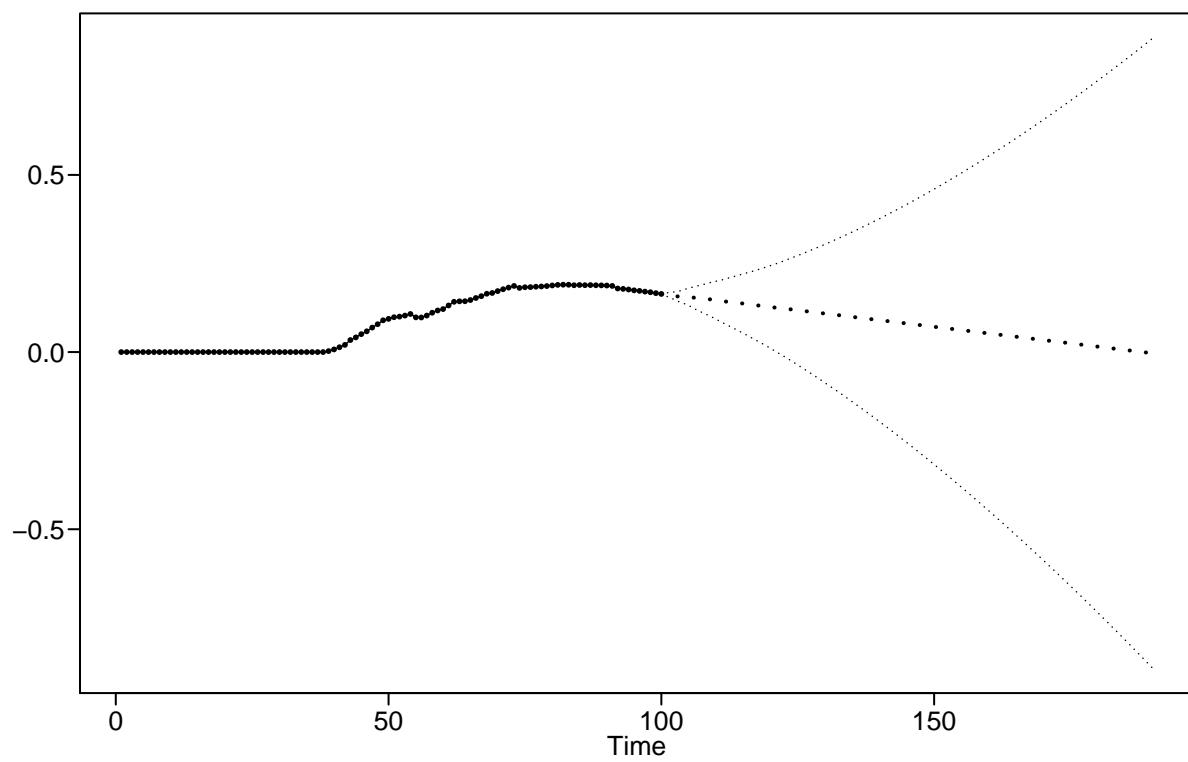
```
modelName = "US Model - 7 Days"  
GetRollingWindowASEInfo(modelName, RollingASEUS7Day)
```

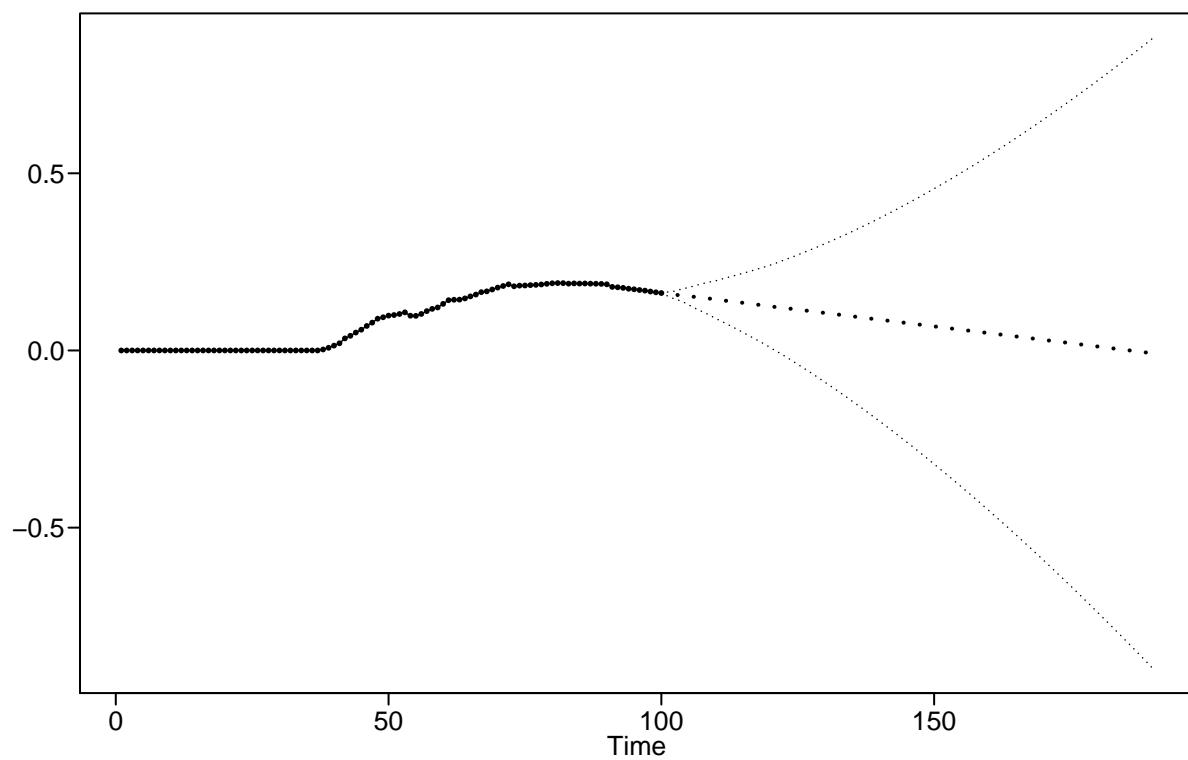
Histogram of US Model – 7 Days Windowed ASE

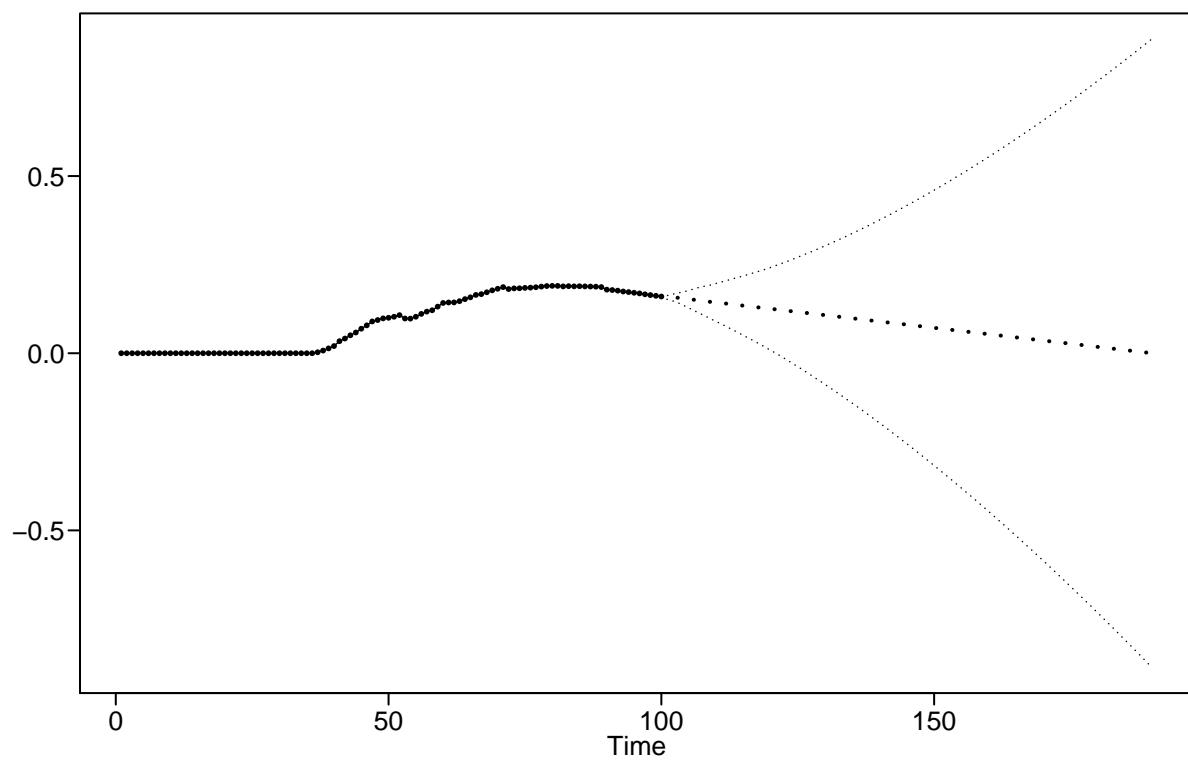


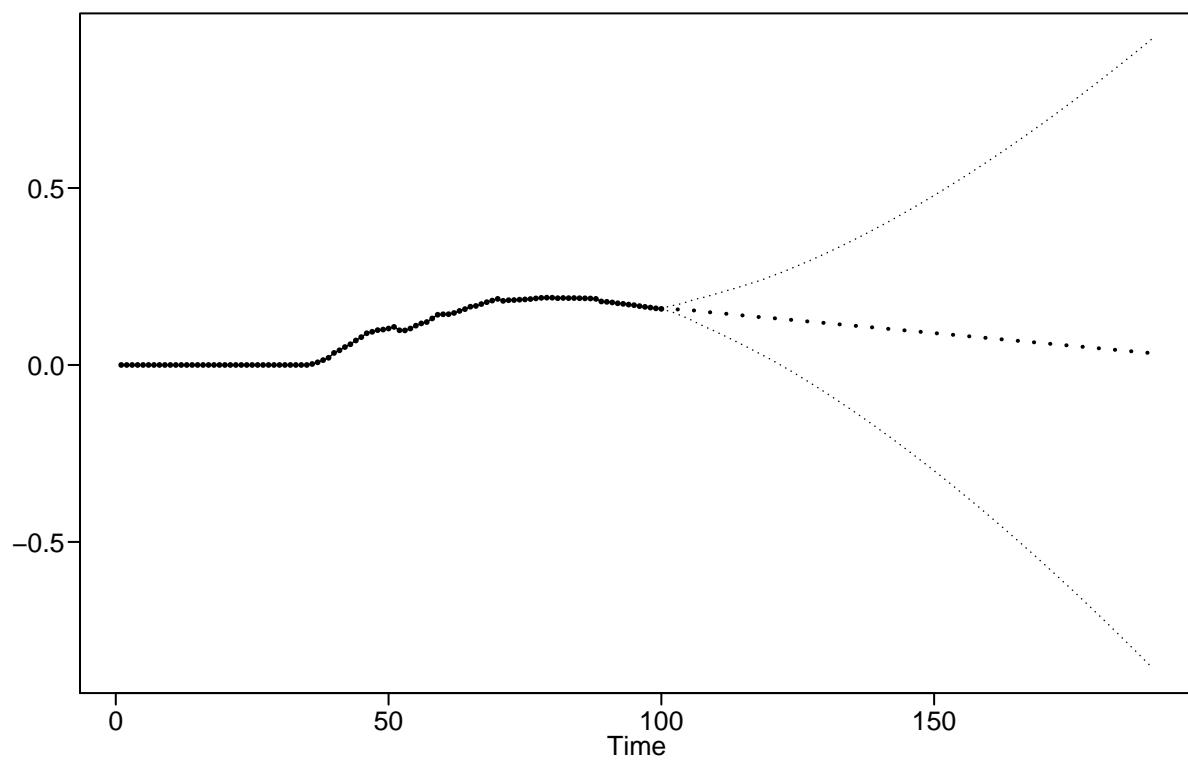
```
## [1] "US Model - 7 Days Windowed ASE Mean: 1.26780054730084e-06"  
## [1] "US Model - 7 Days Windowed ASE Median: 1.56336762628859e-07"
```

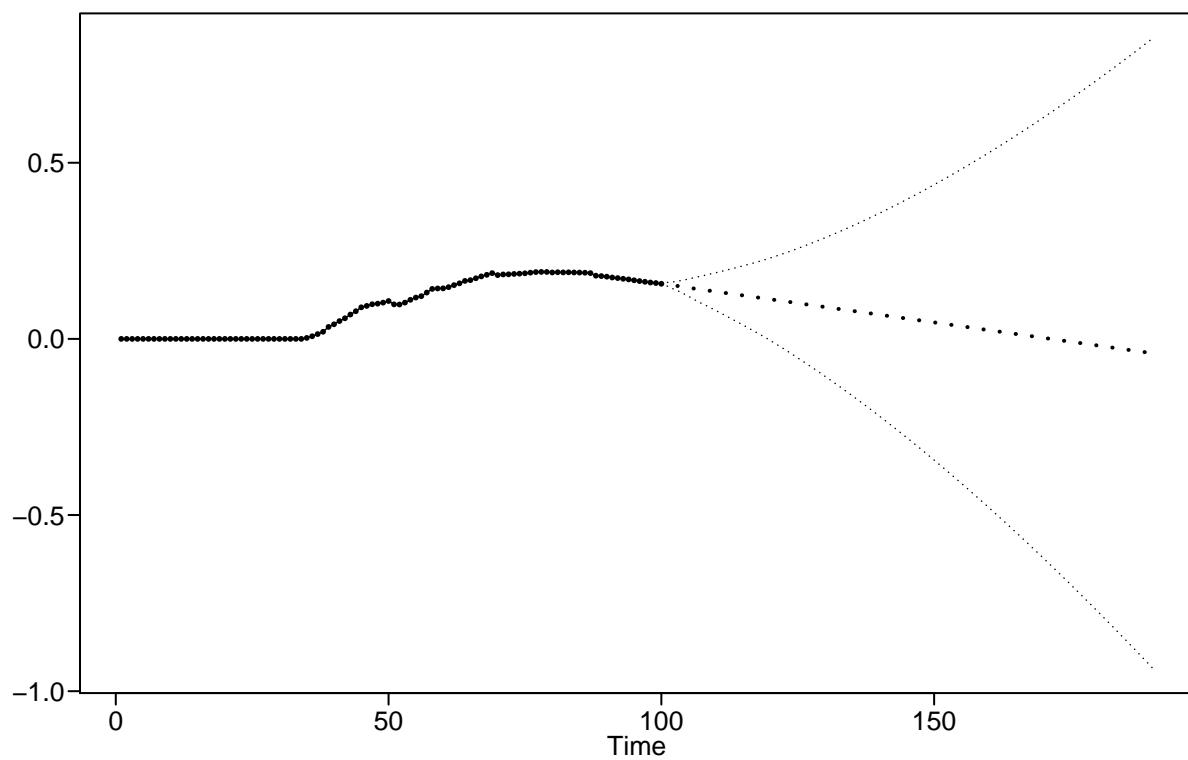
```
#Rolling Window ASE Calc - 90 Days  
RollingASEUS90Day = RollingASECalc(df_US_Final$Percent_Positive,phis=US.est$phi, thetas = US.est$theta,
```

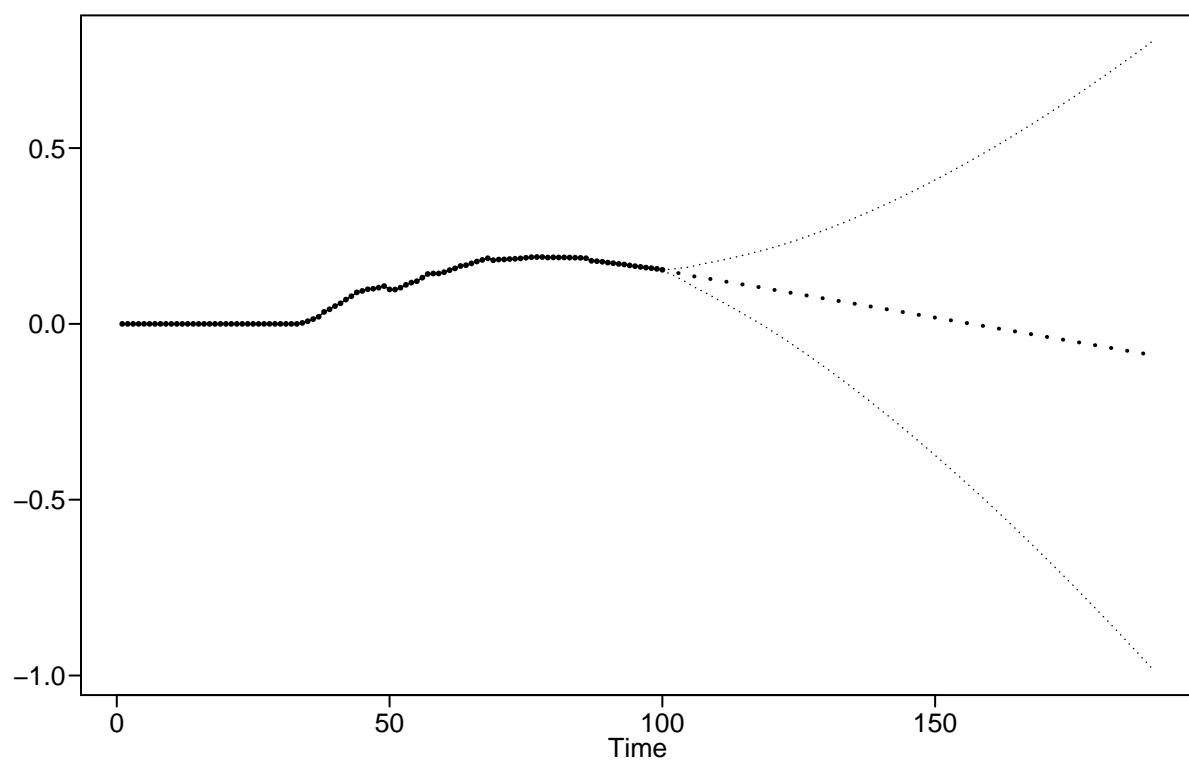


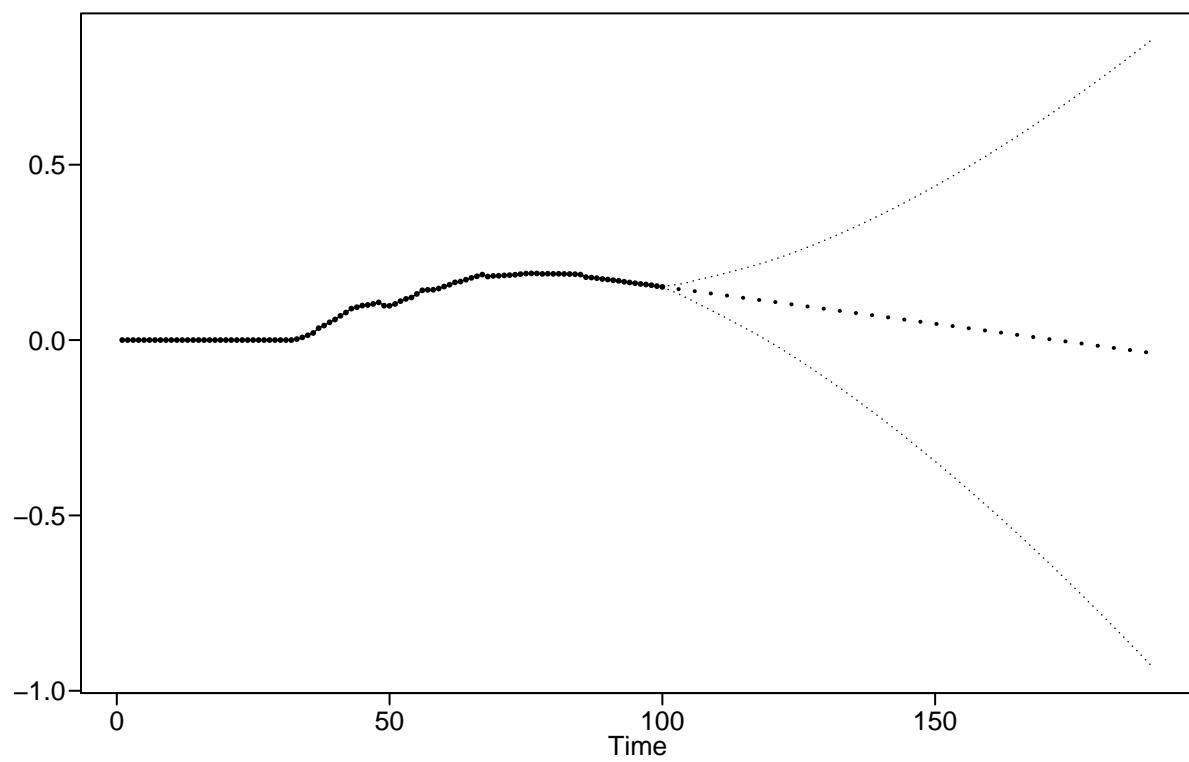


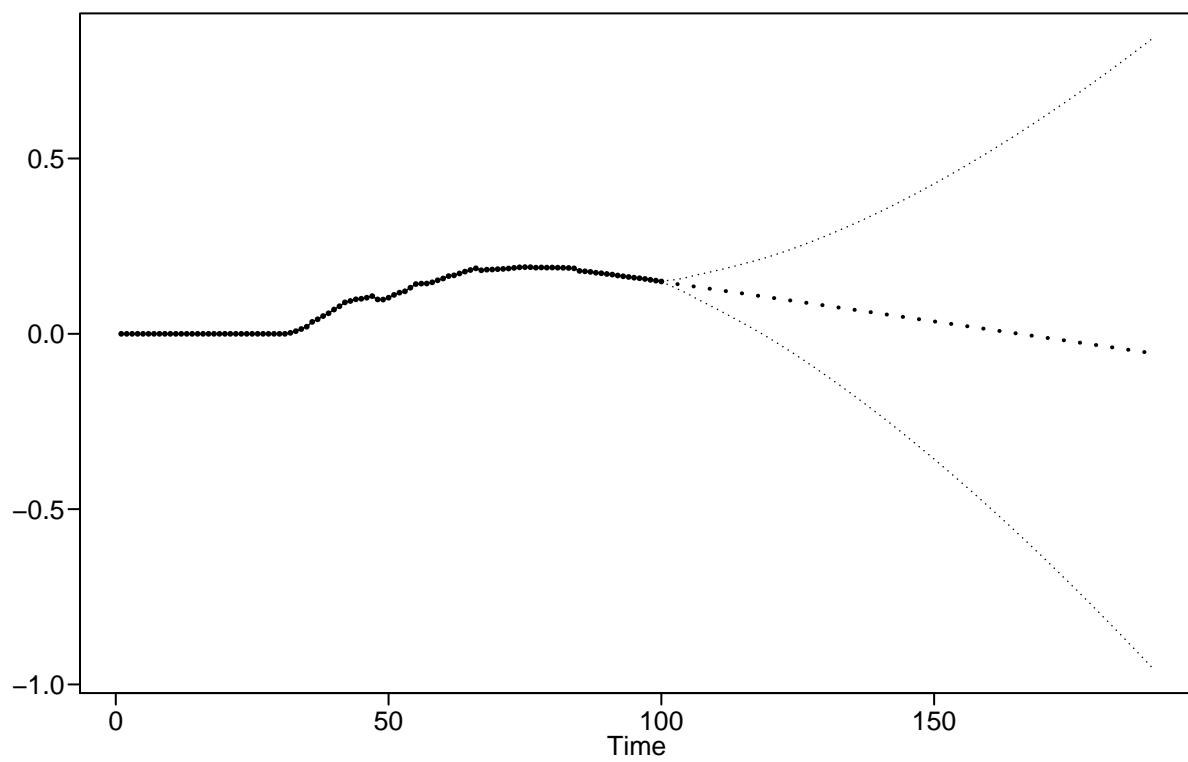


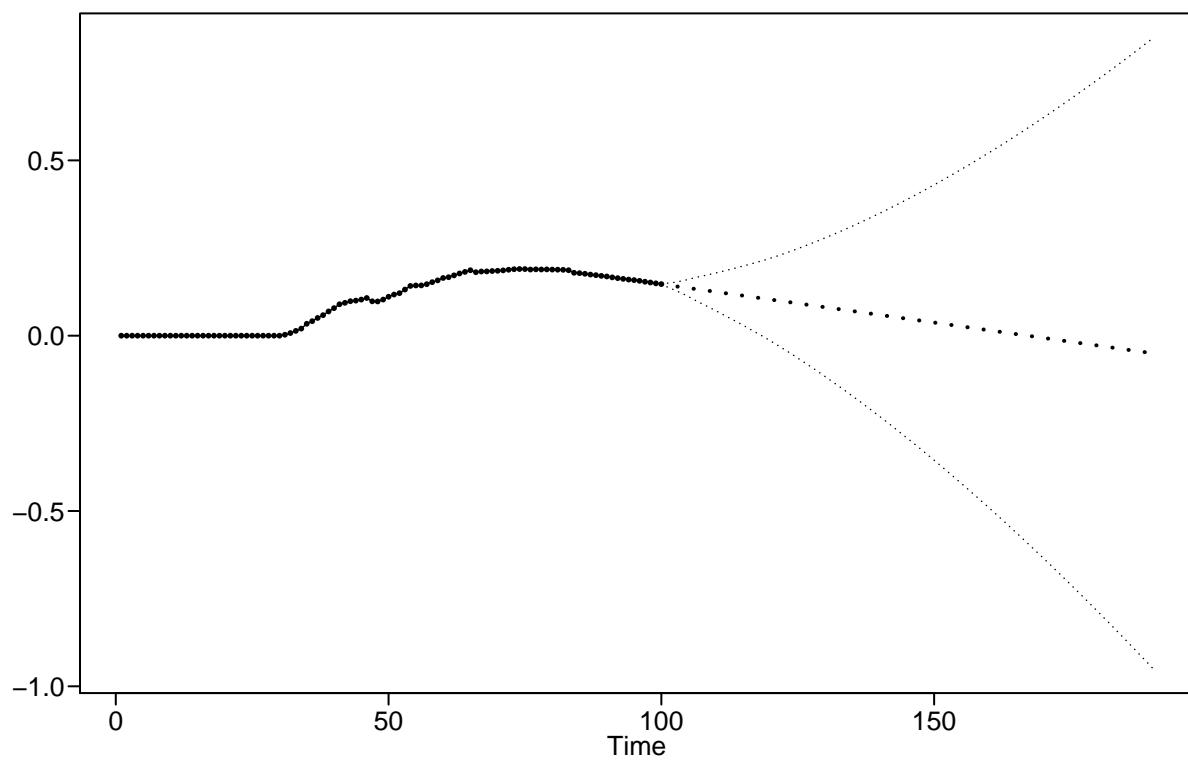


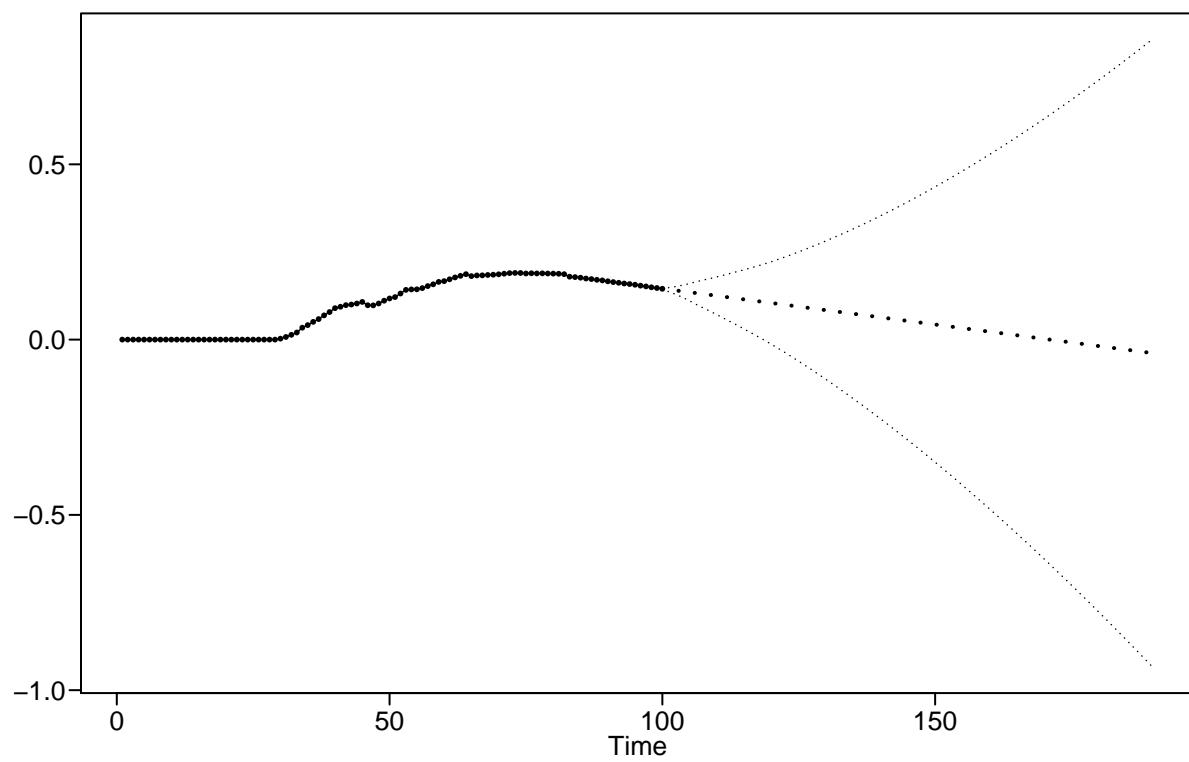


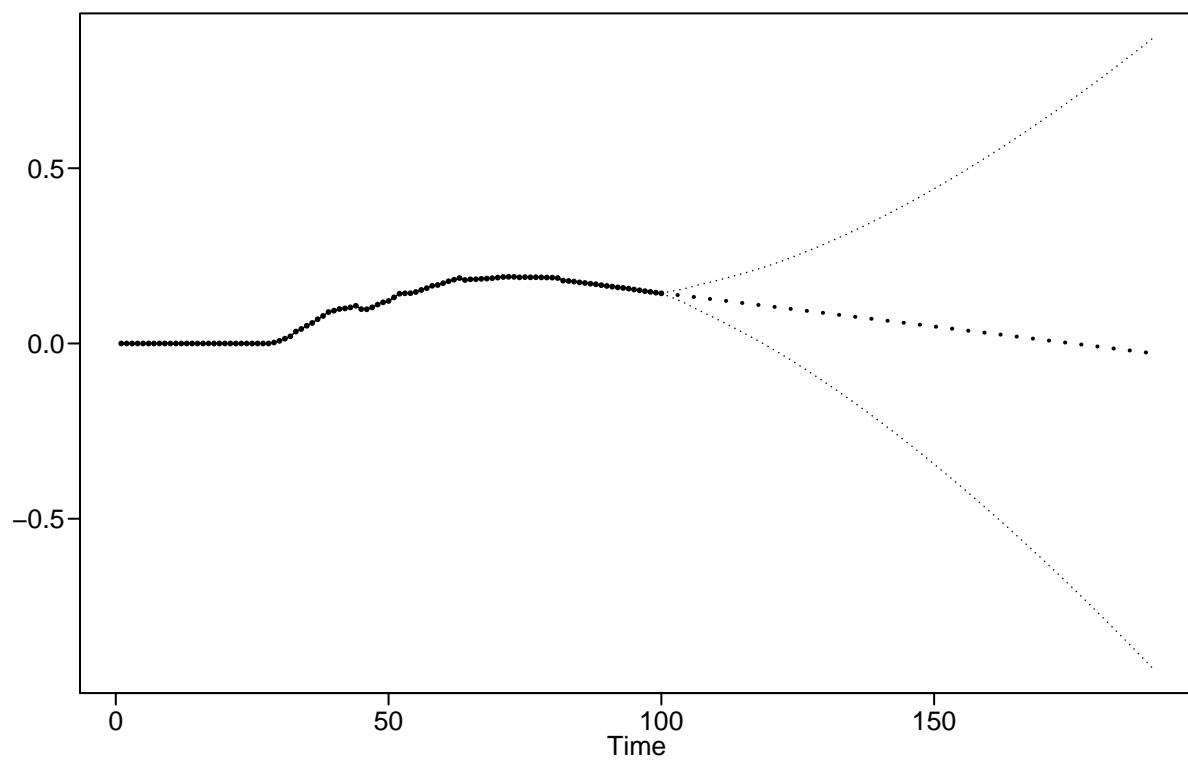


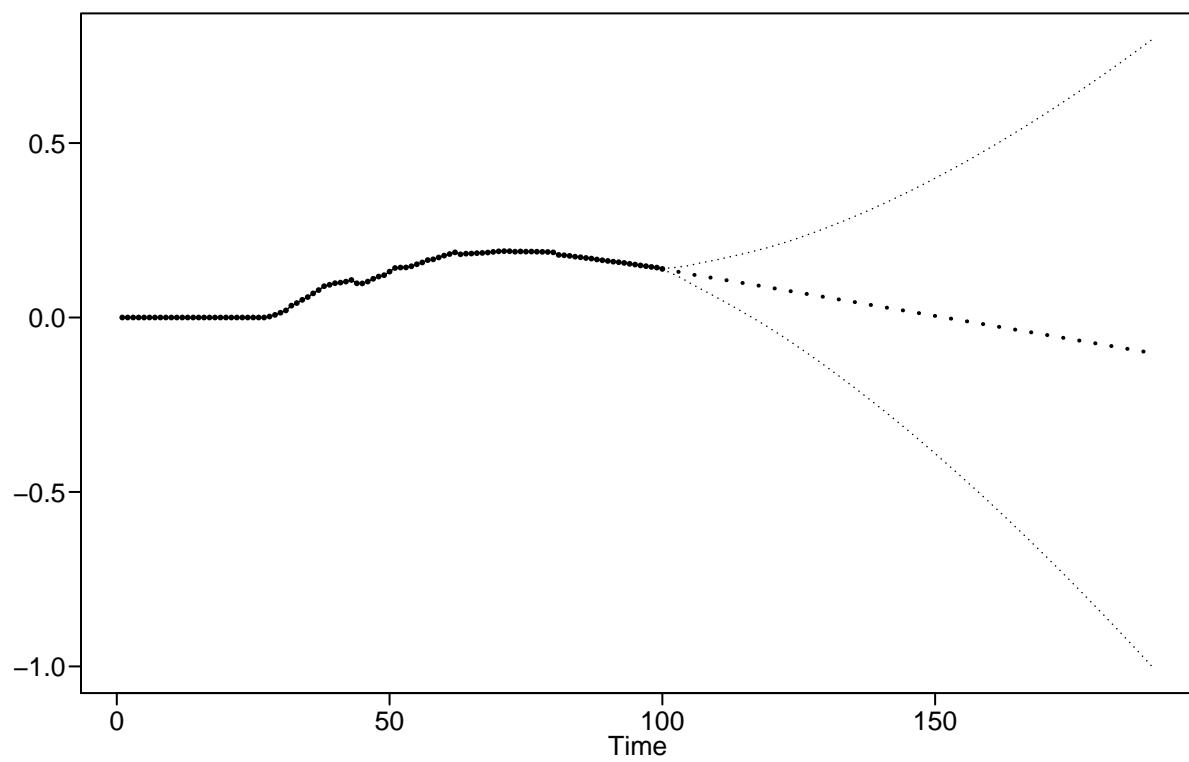


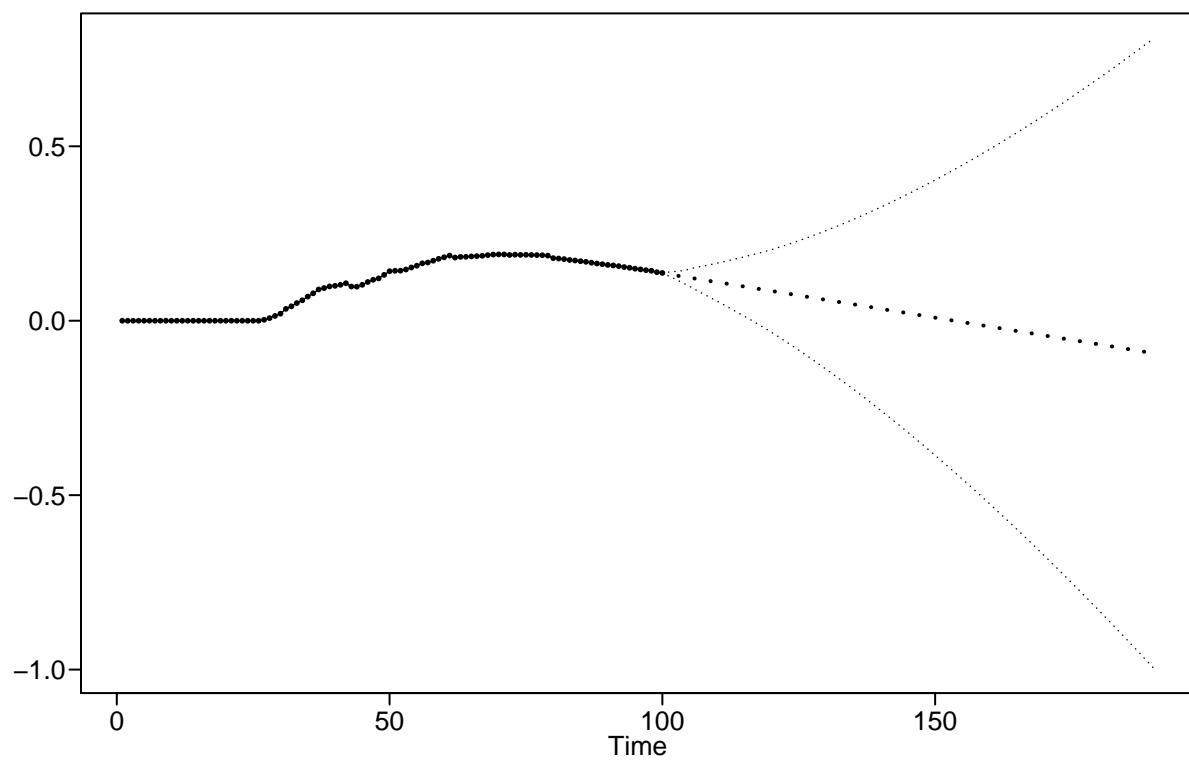


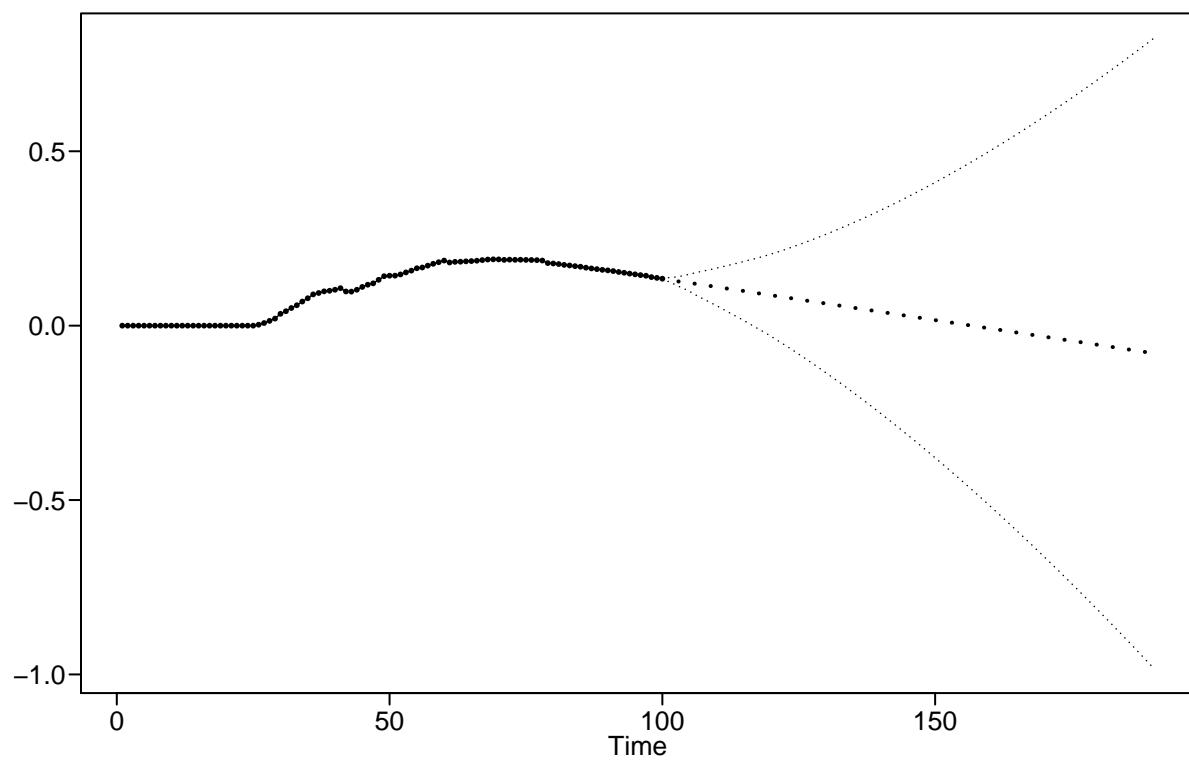


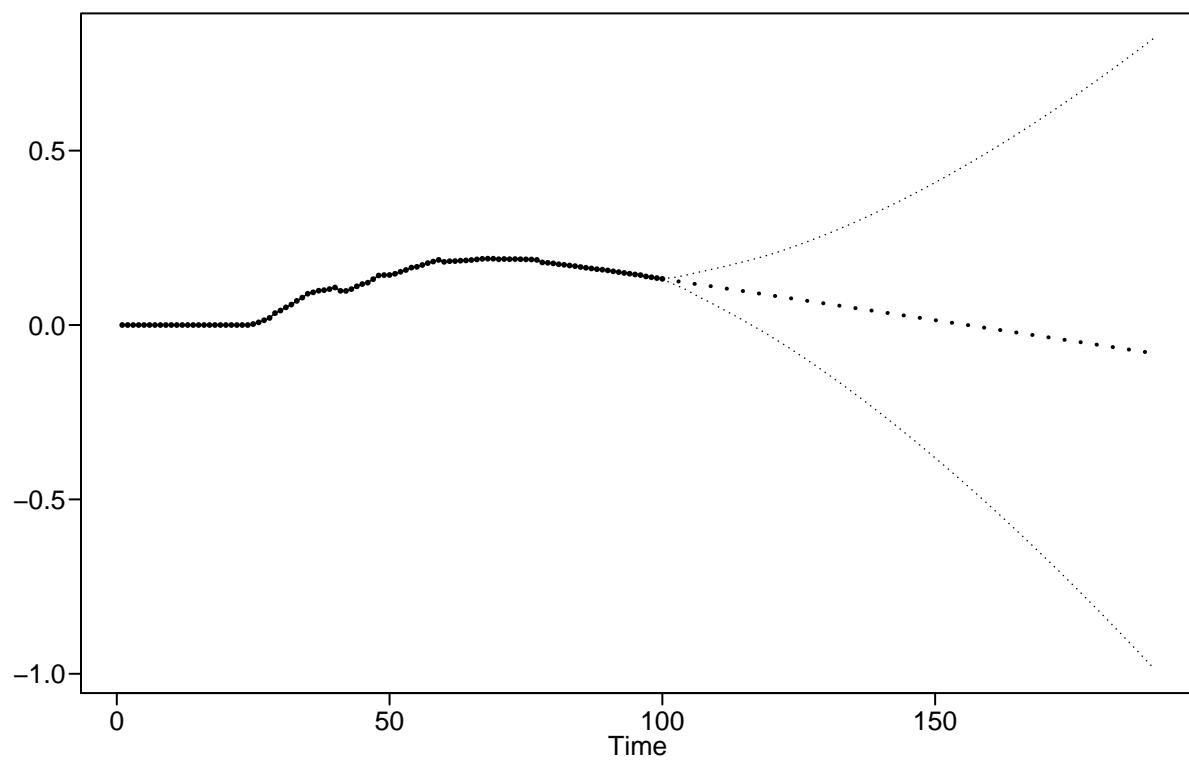


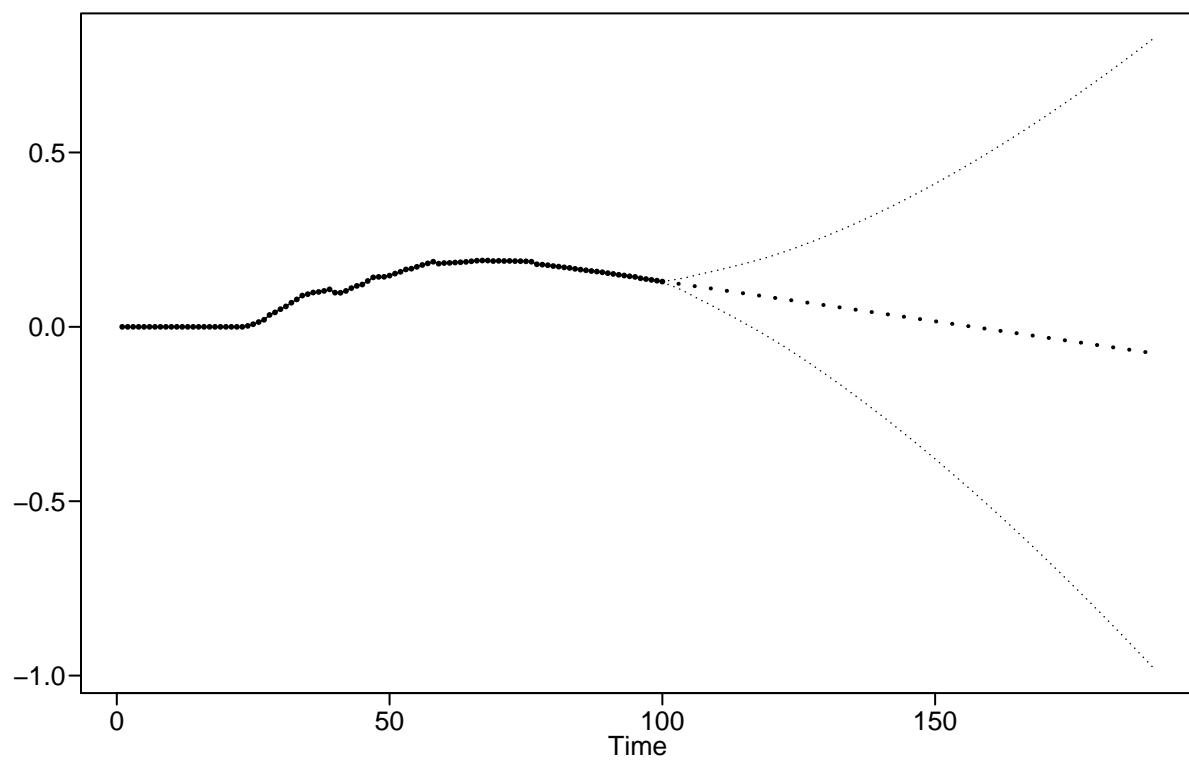


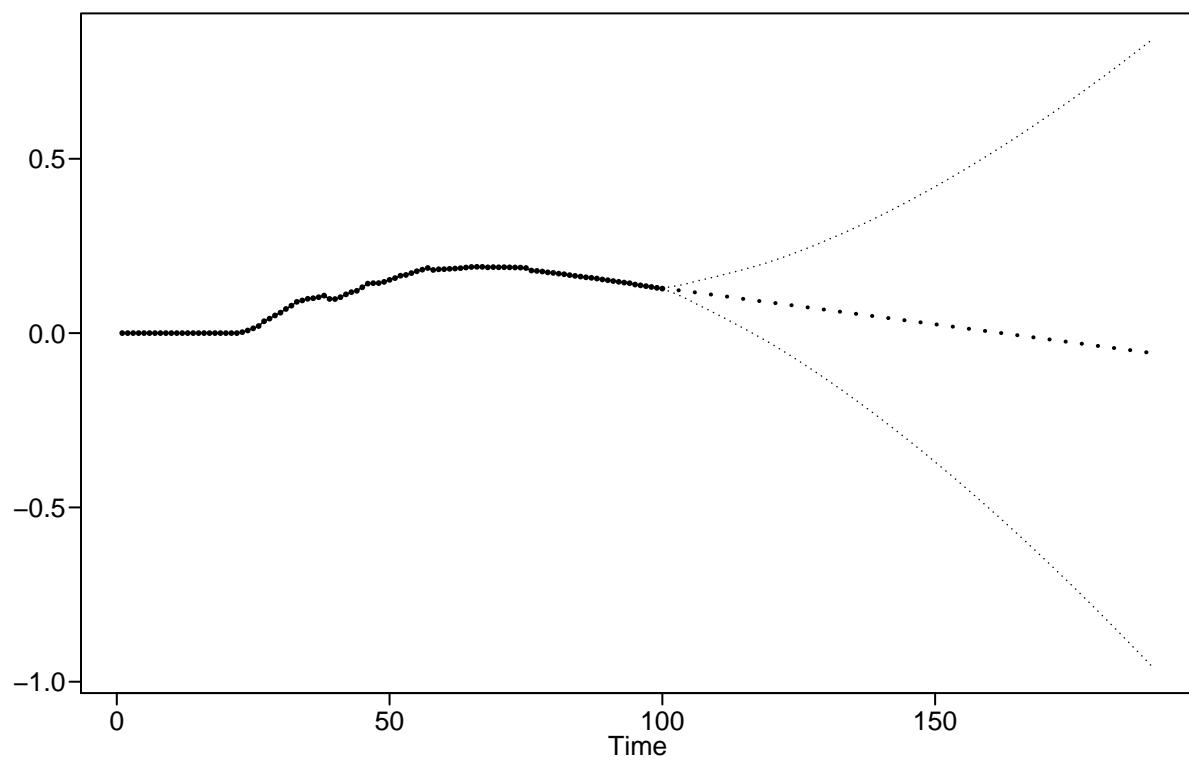


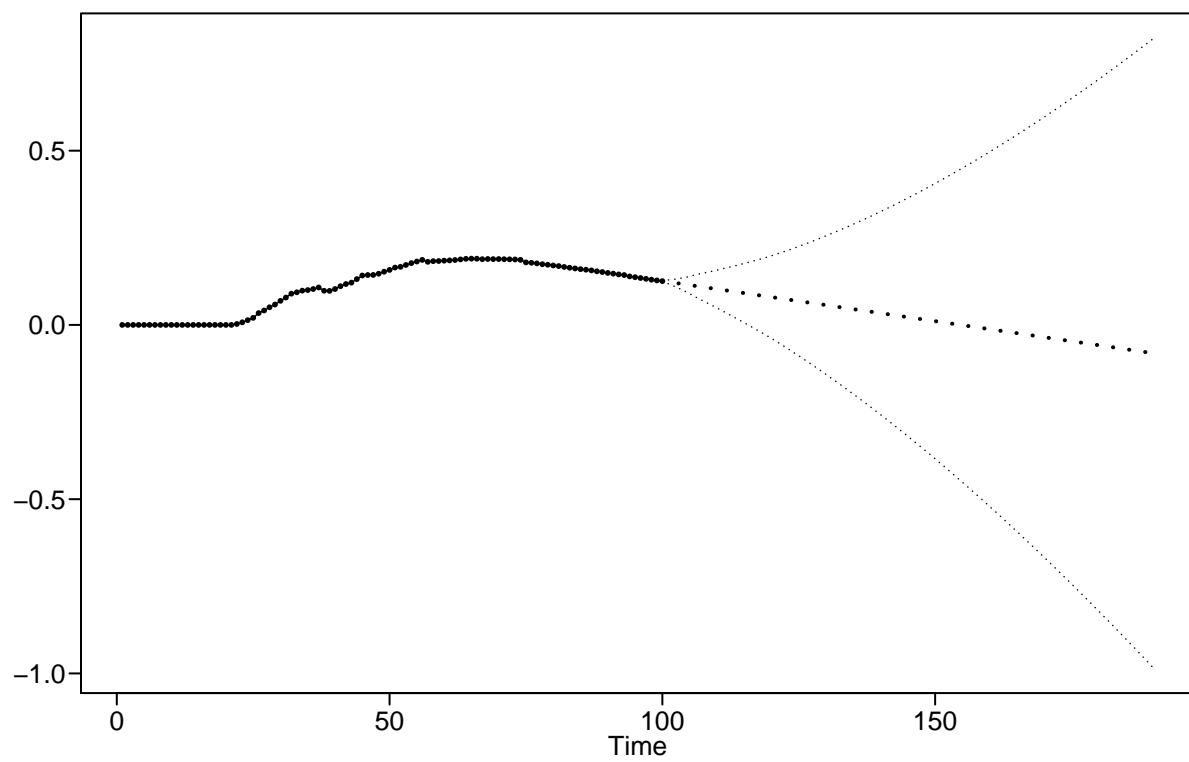


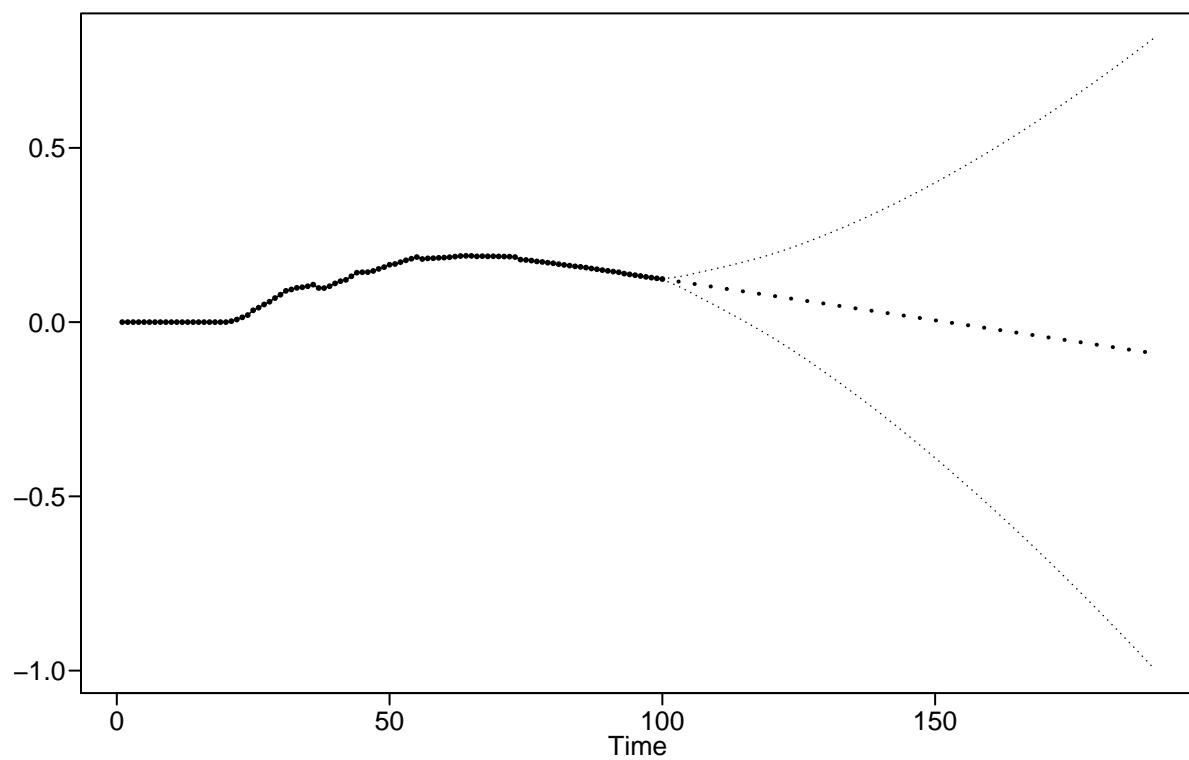


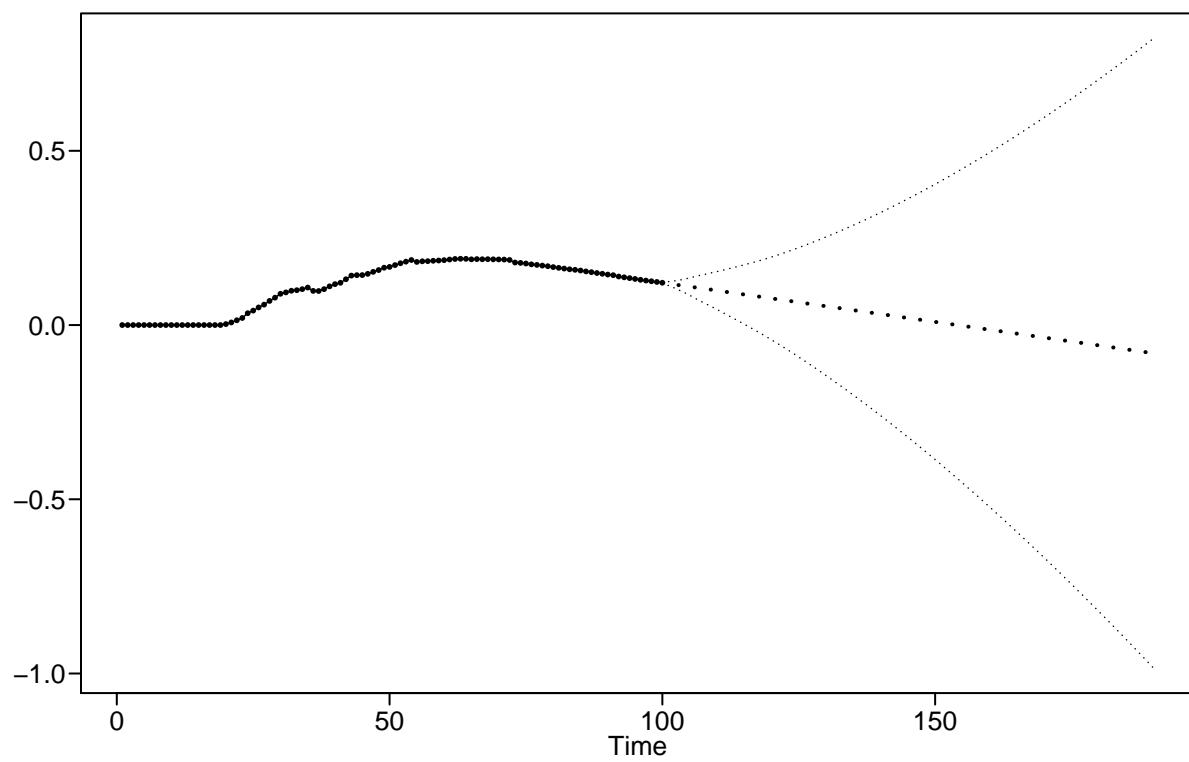


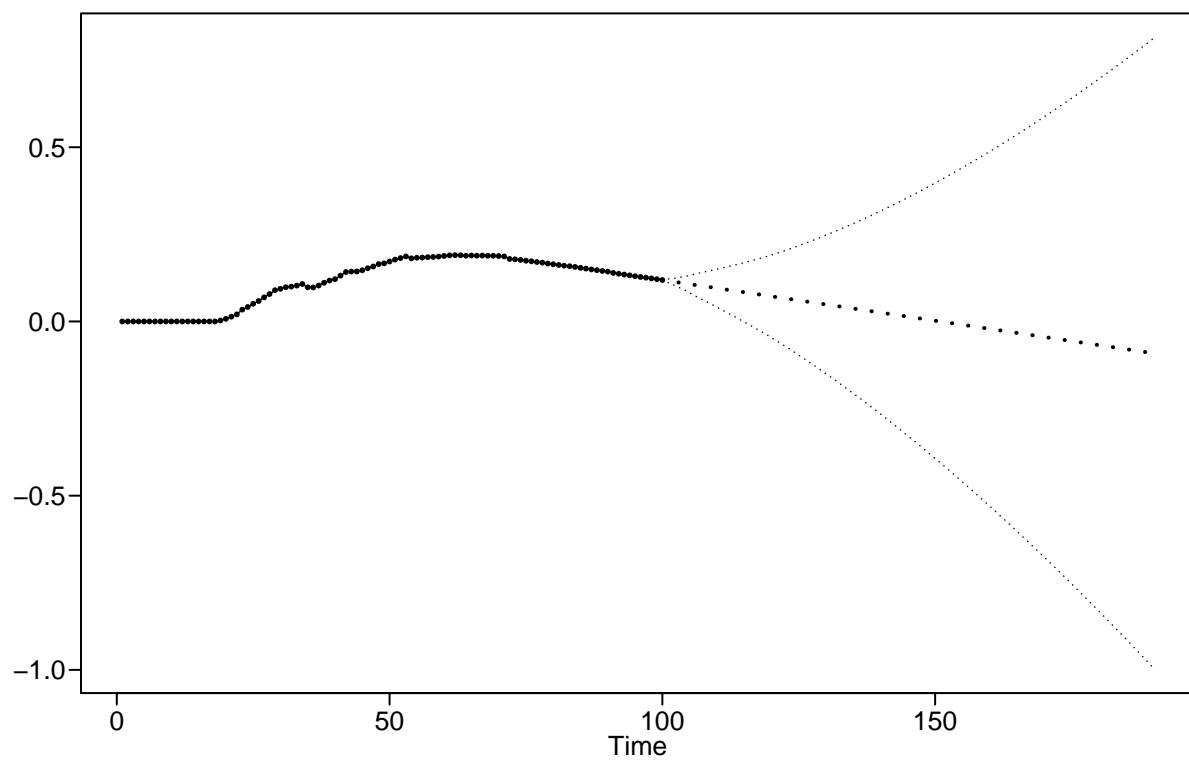


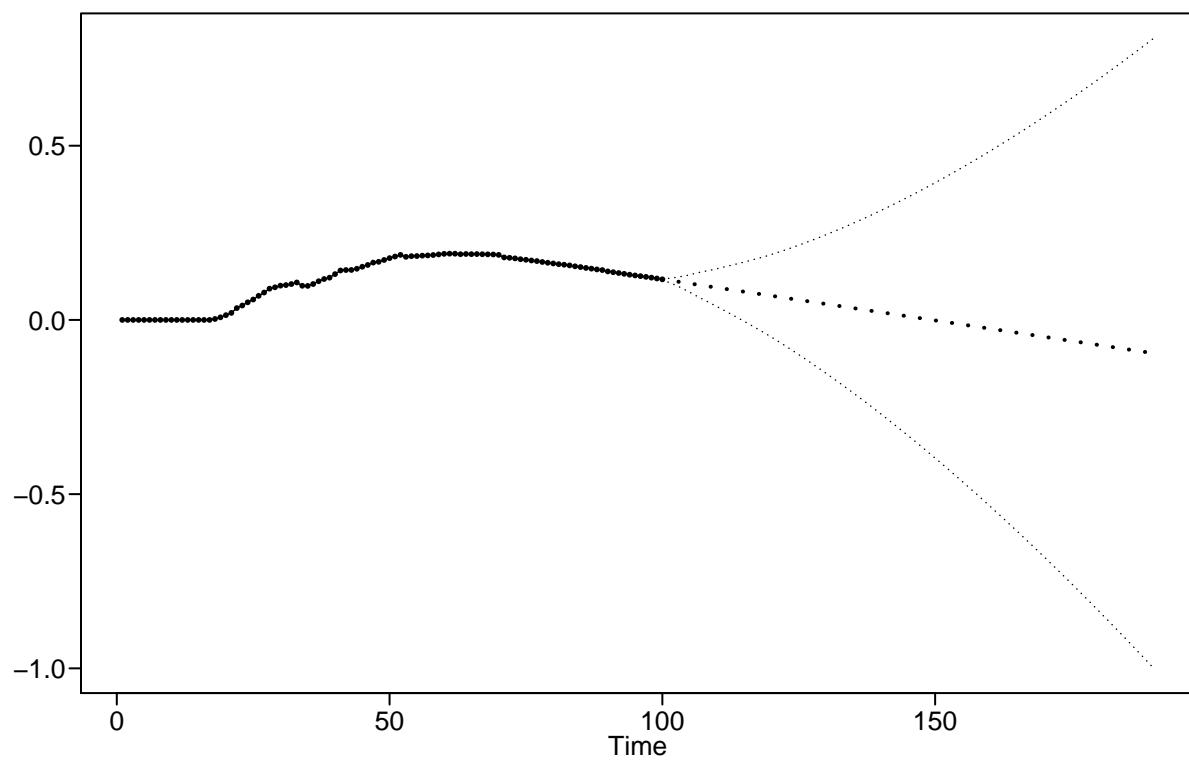


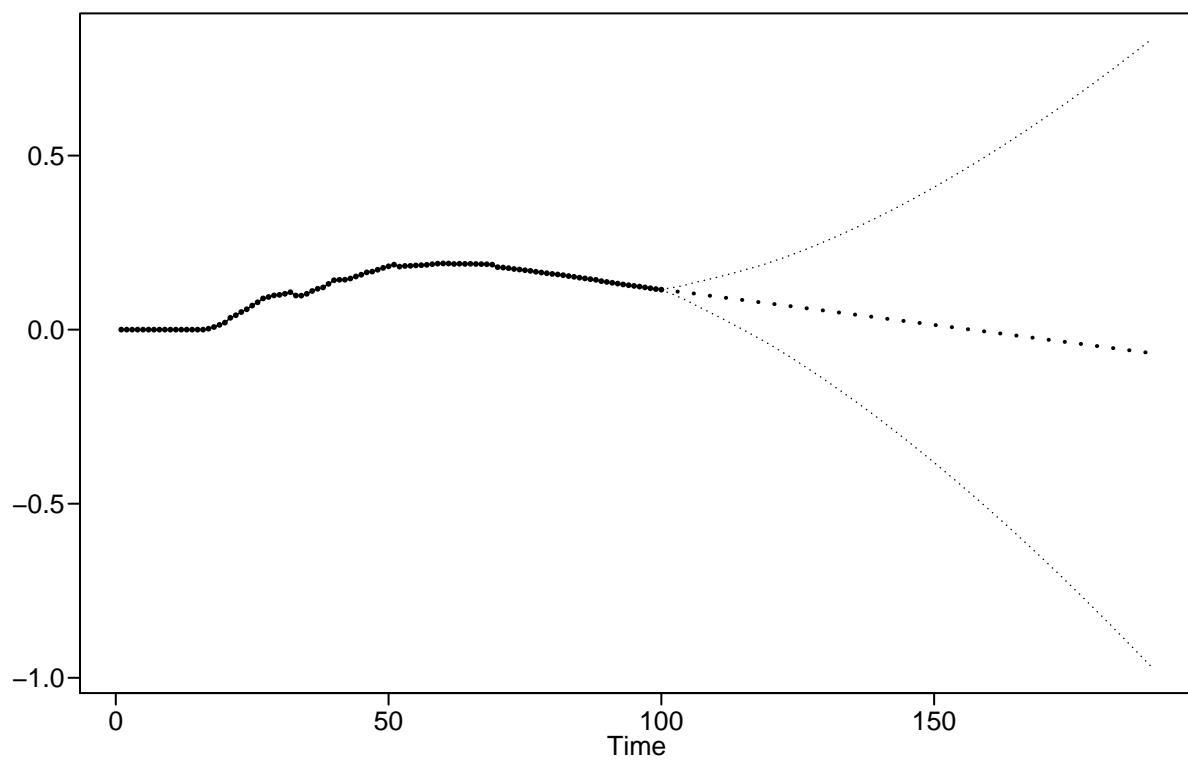


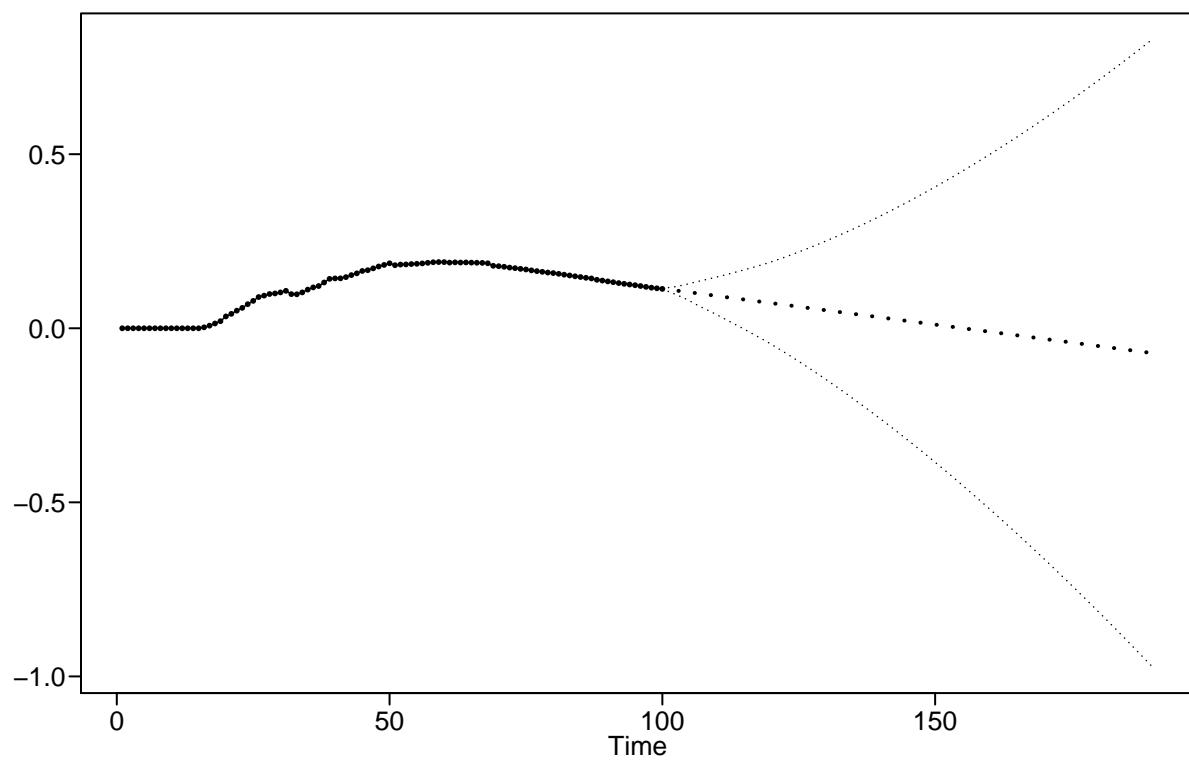


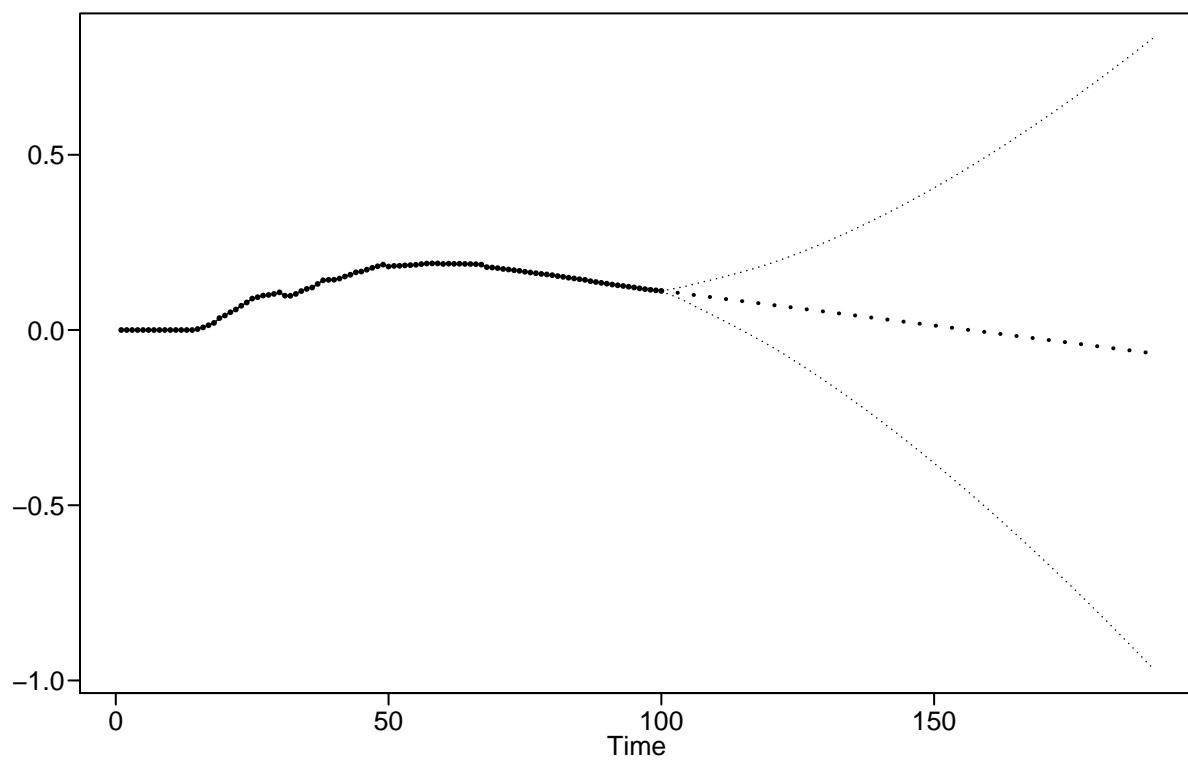


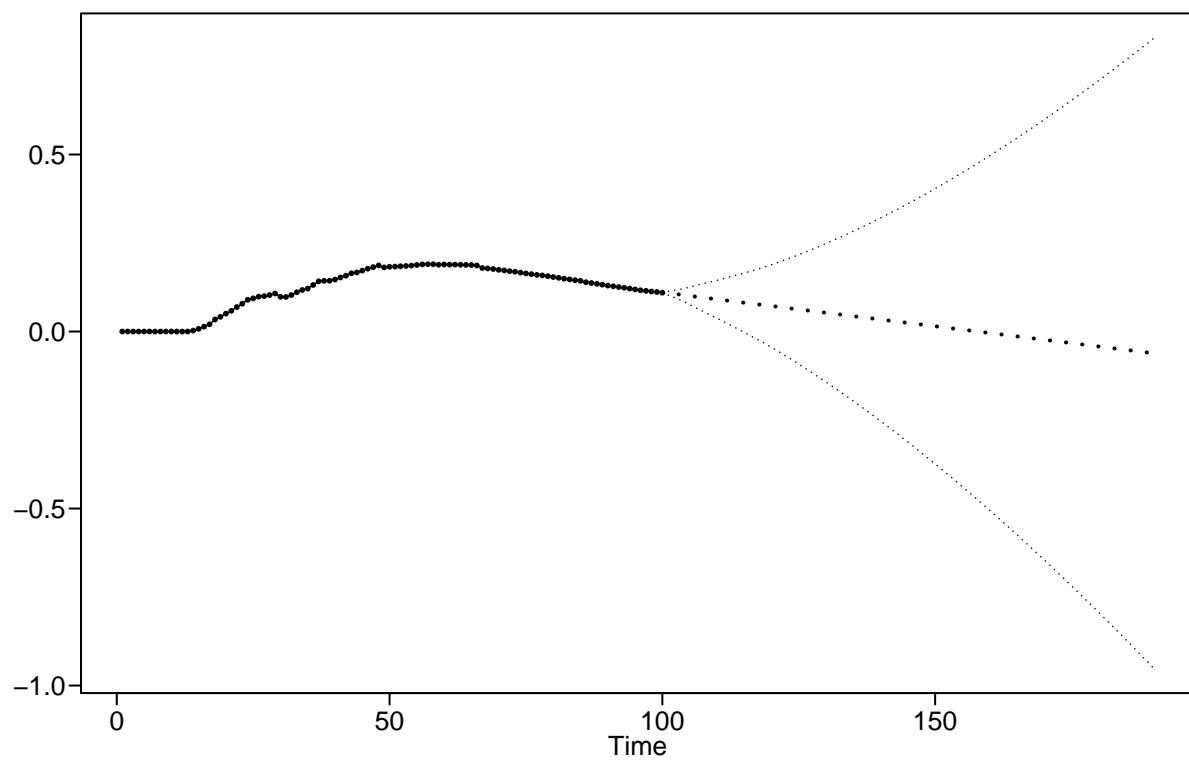


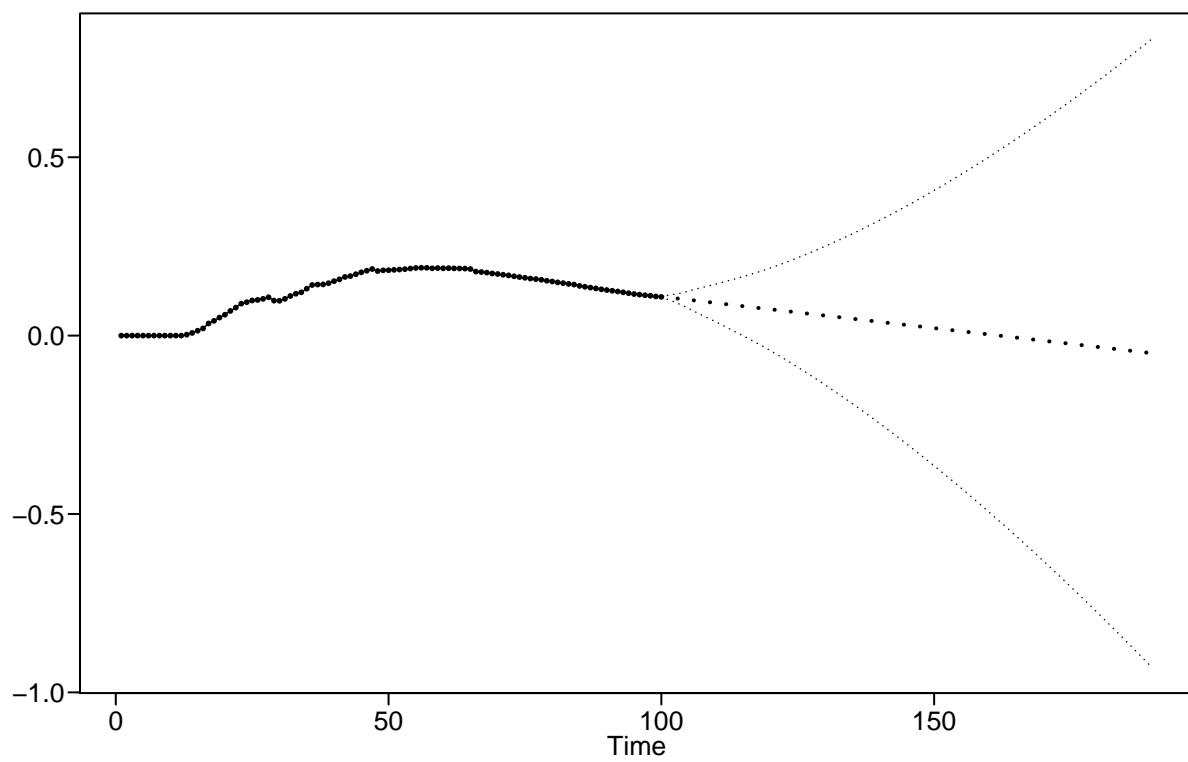


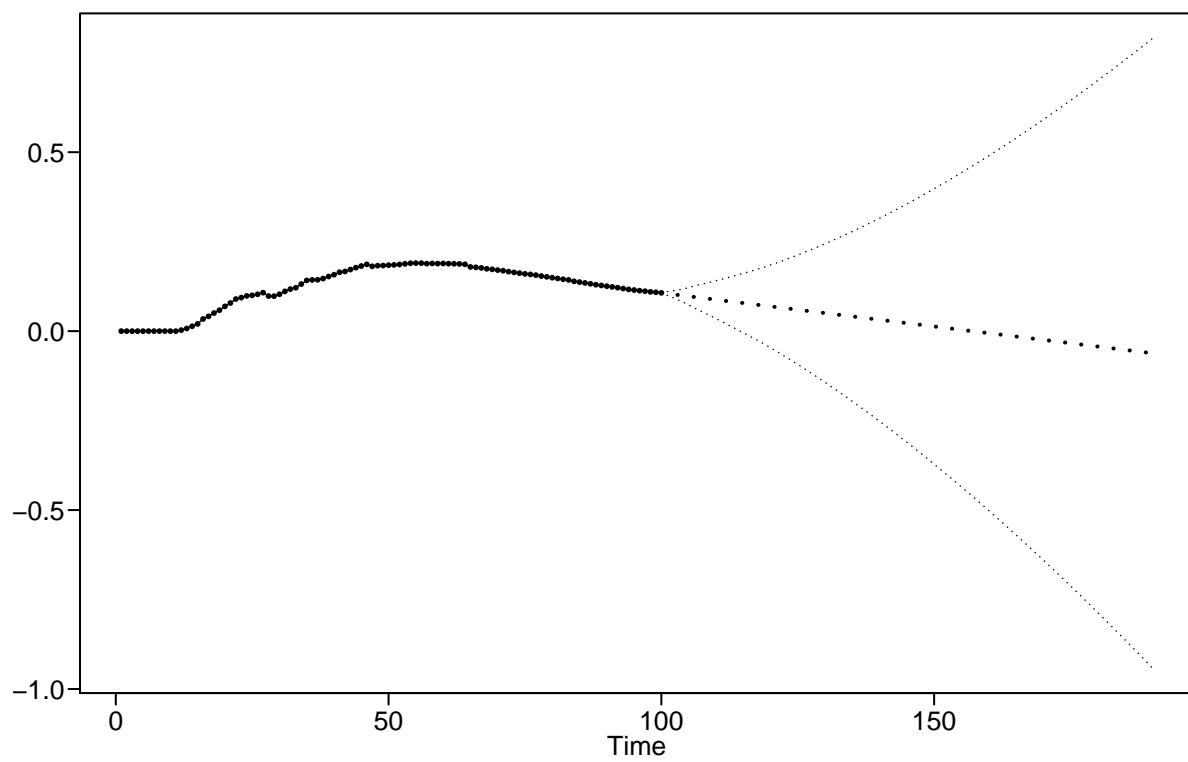


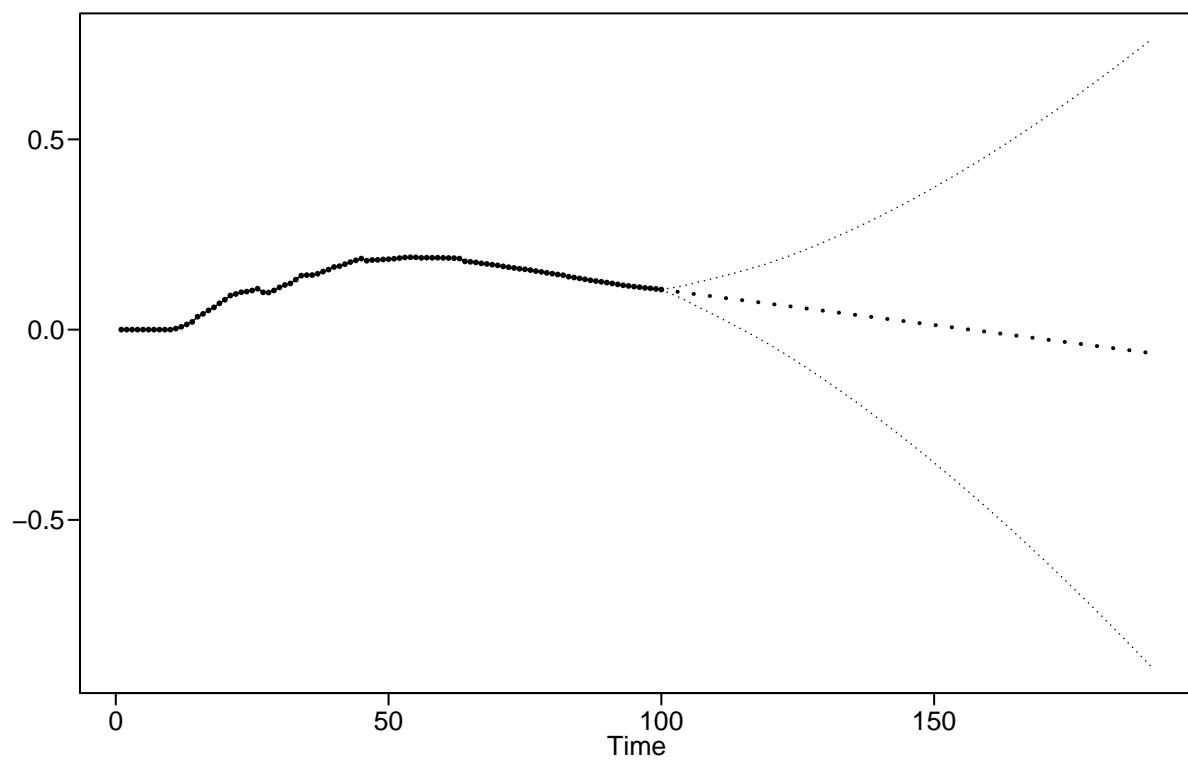


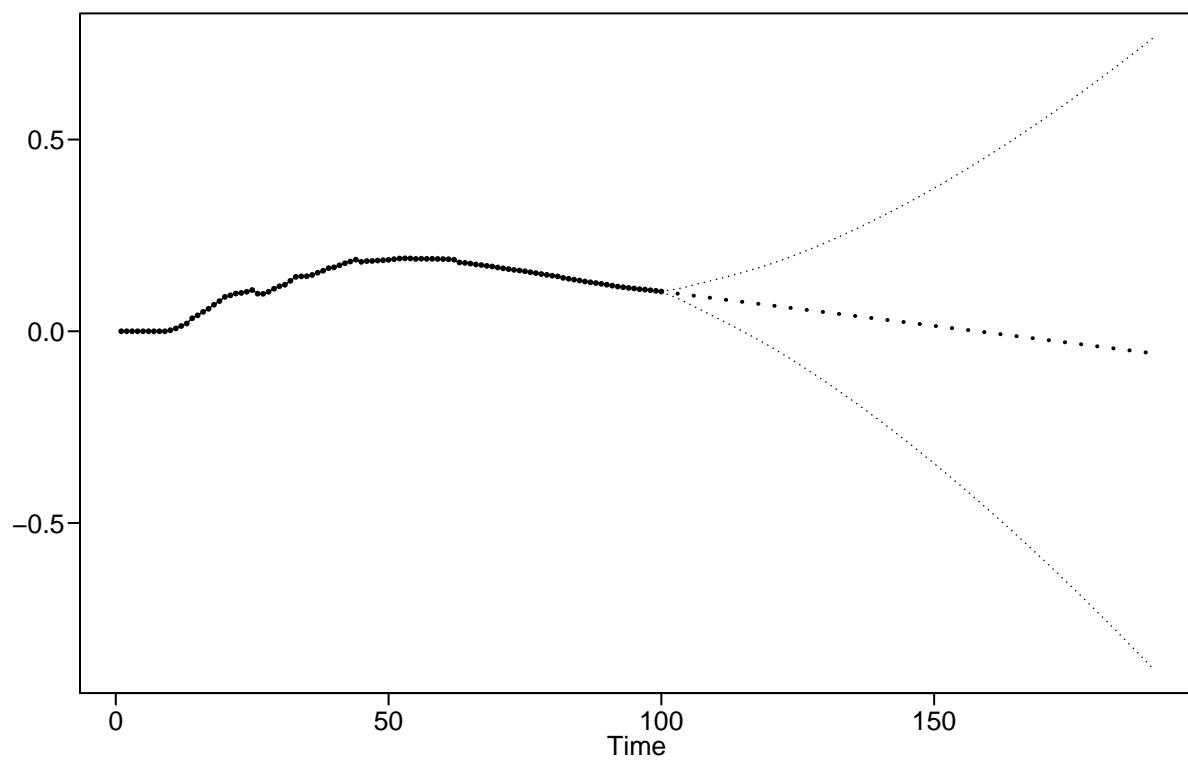


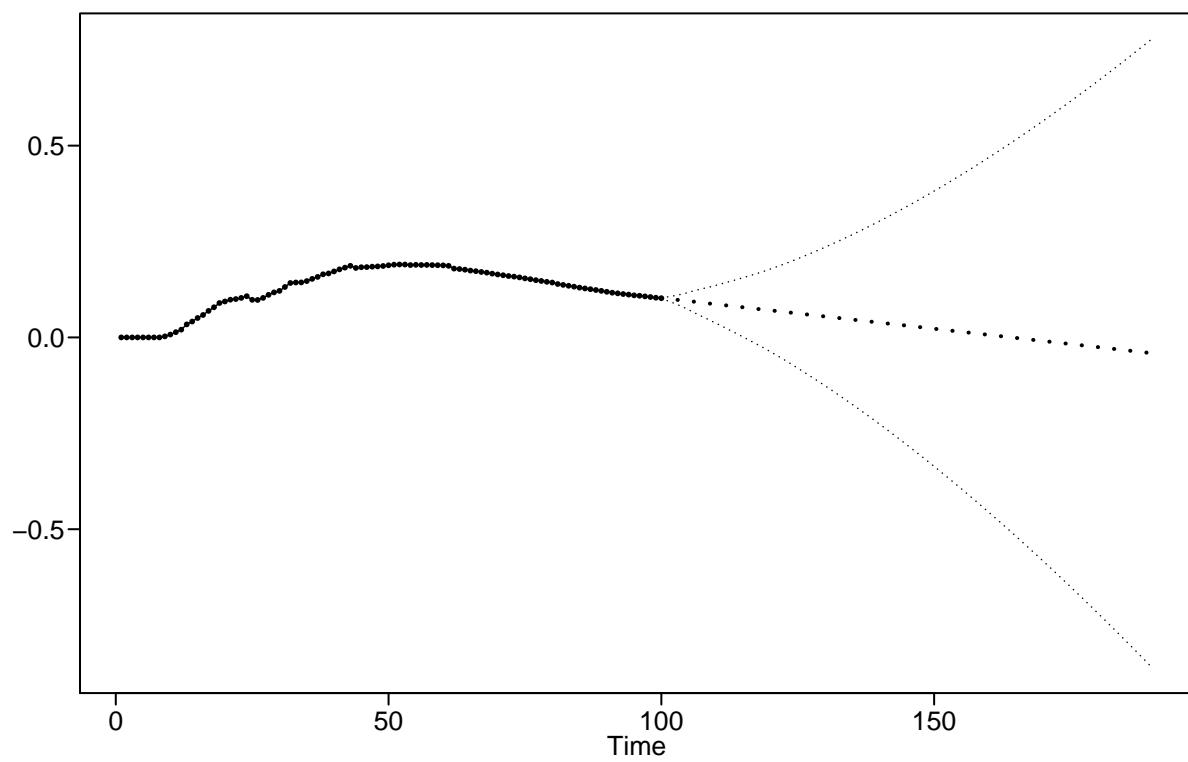


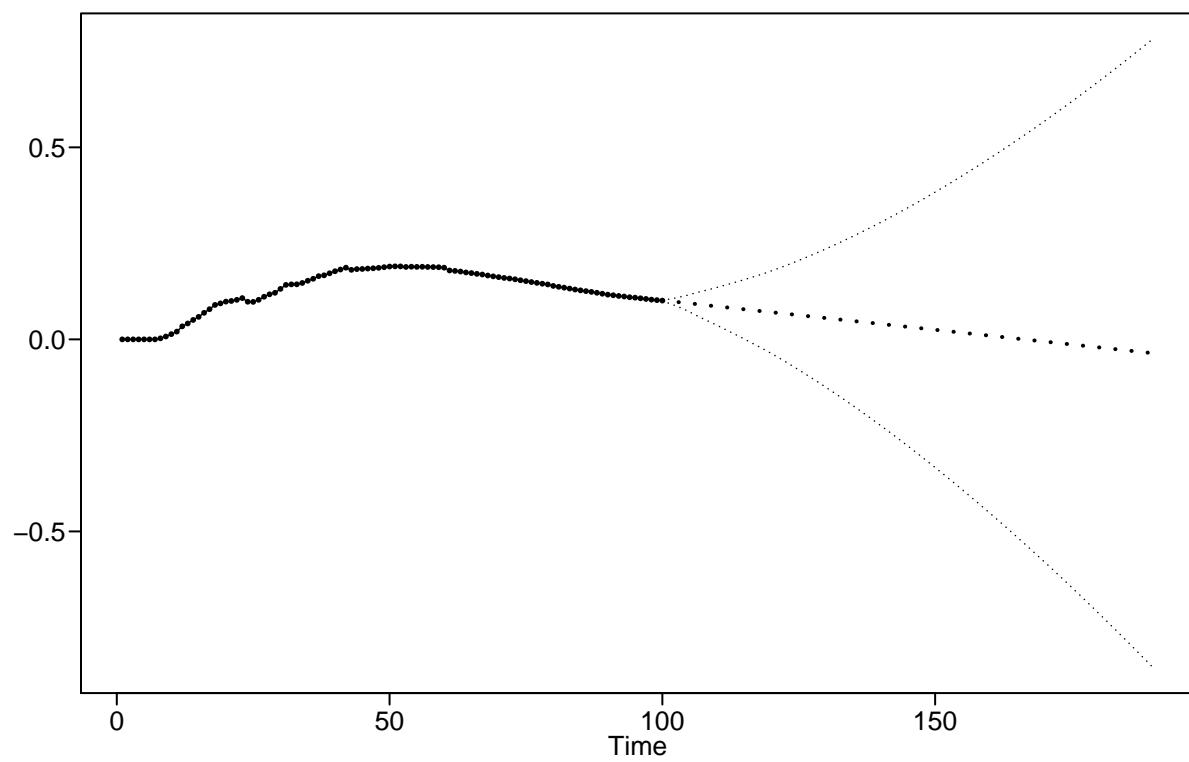


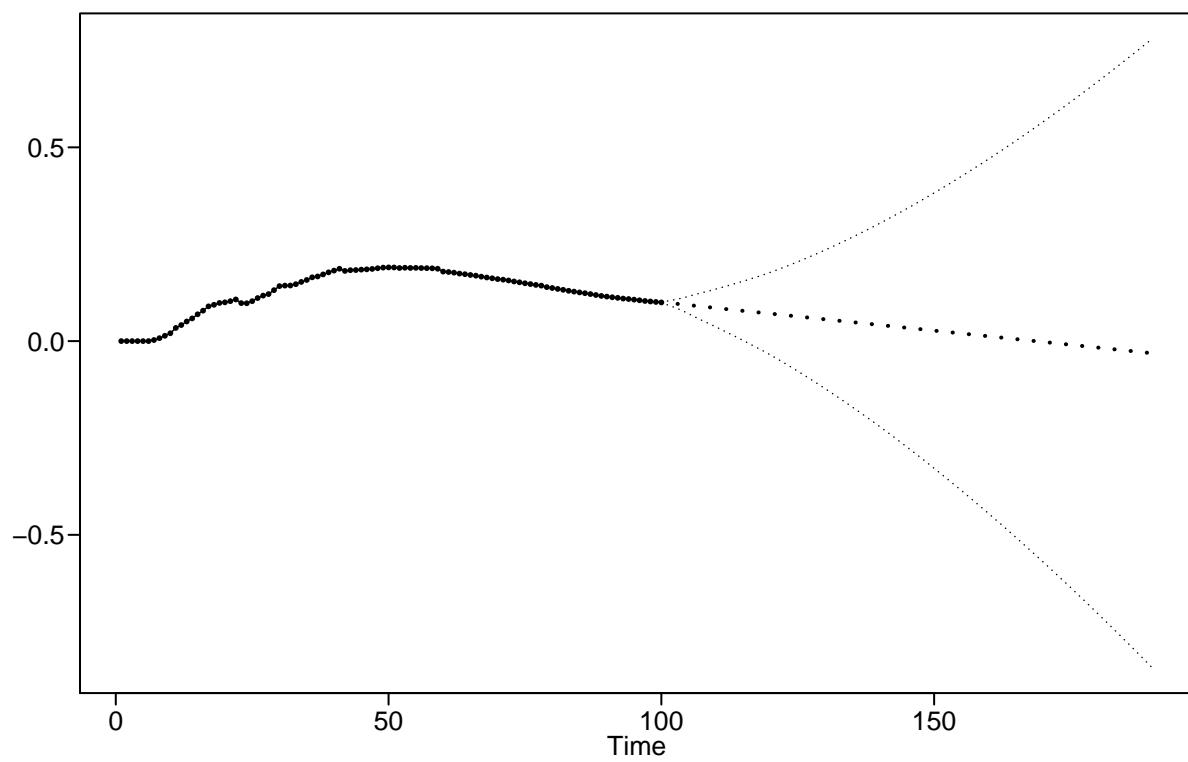


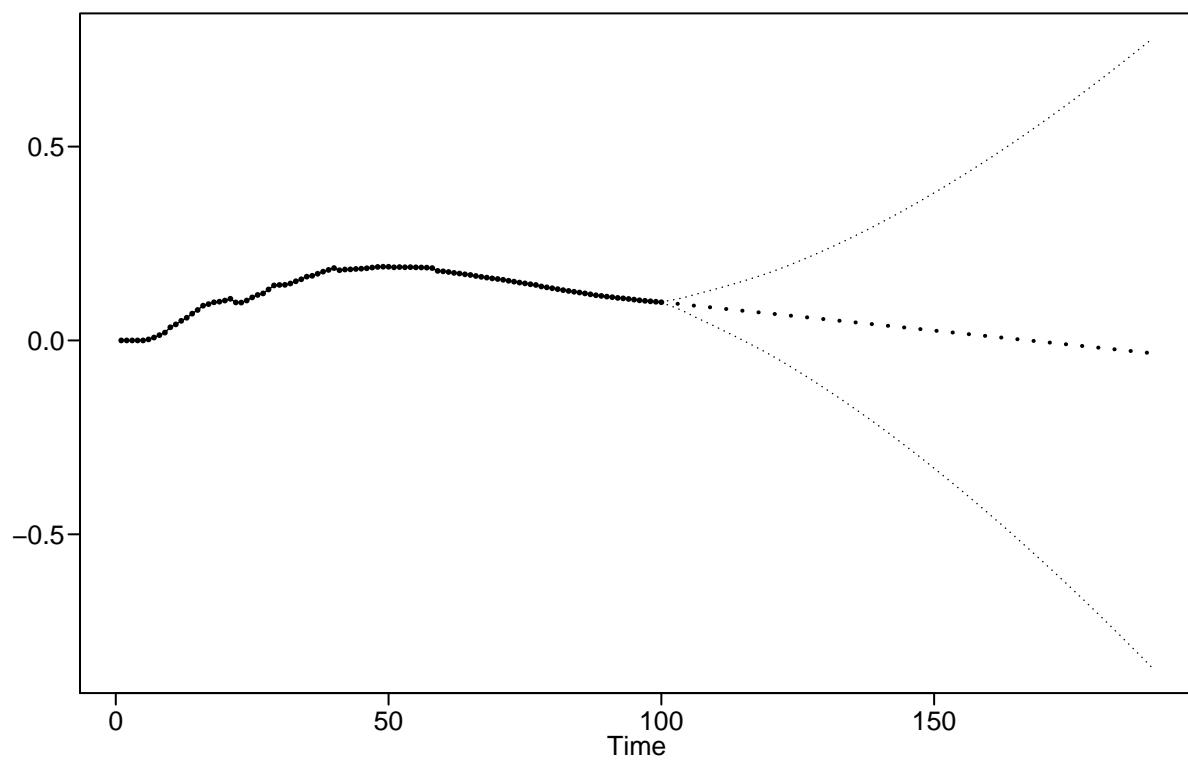


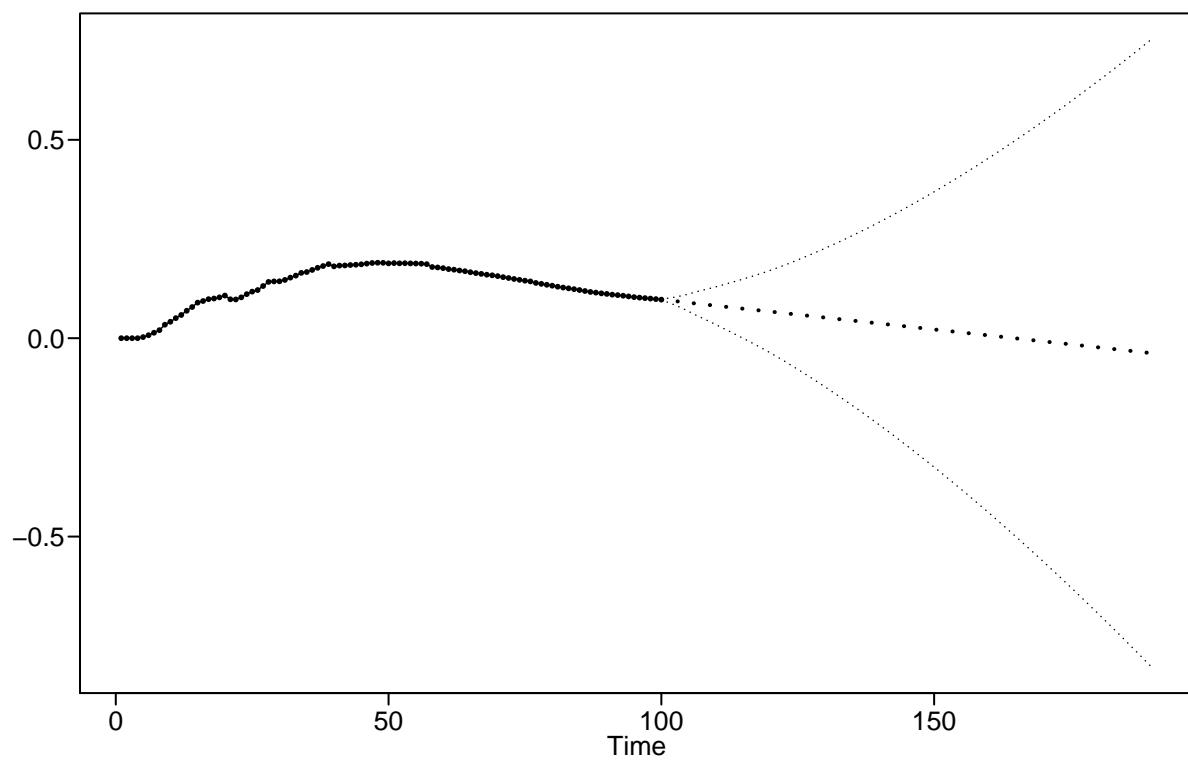


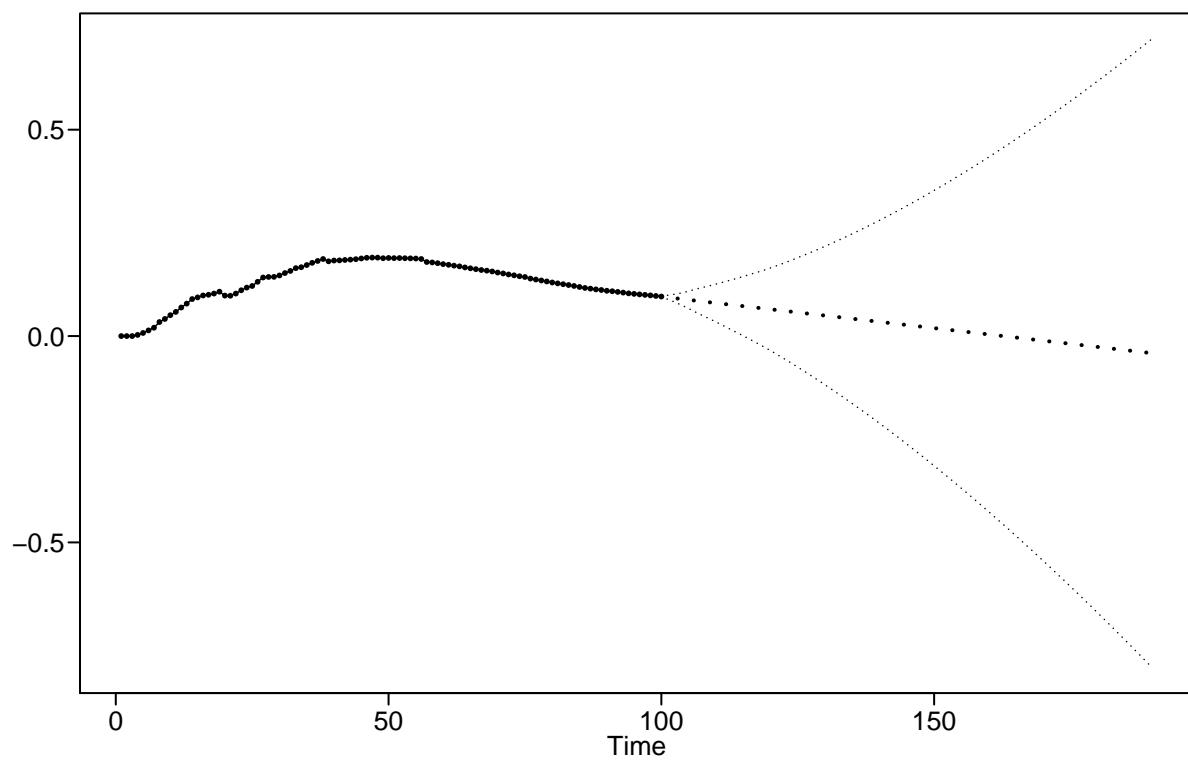


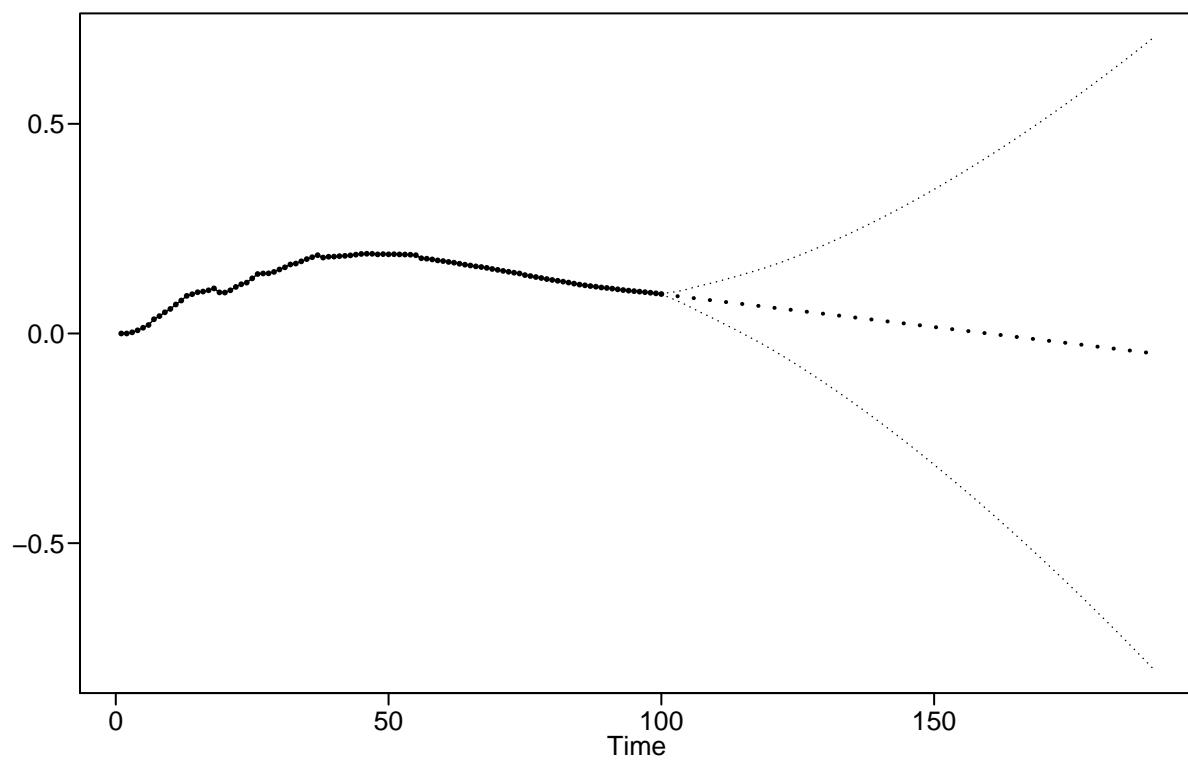


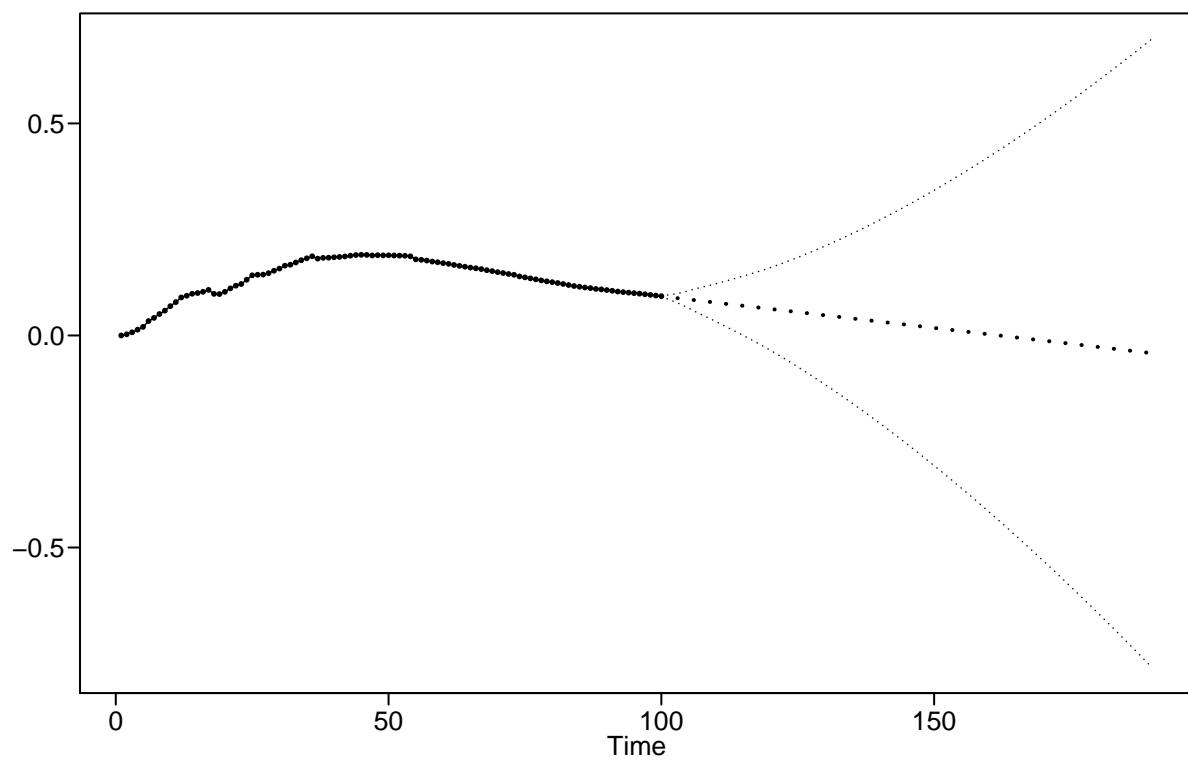


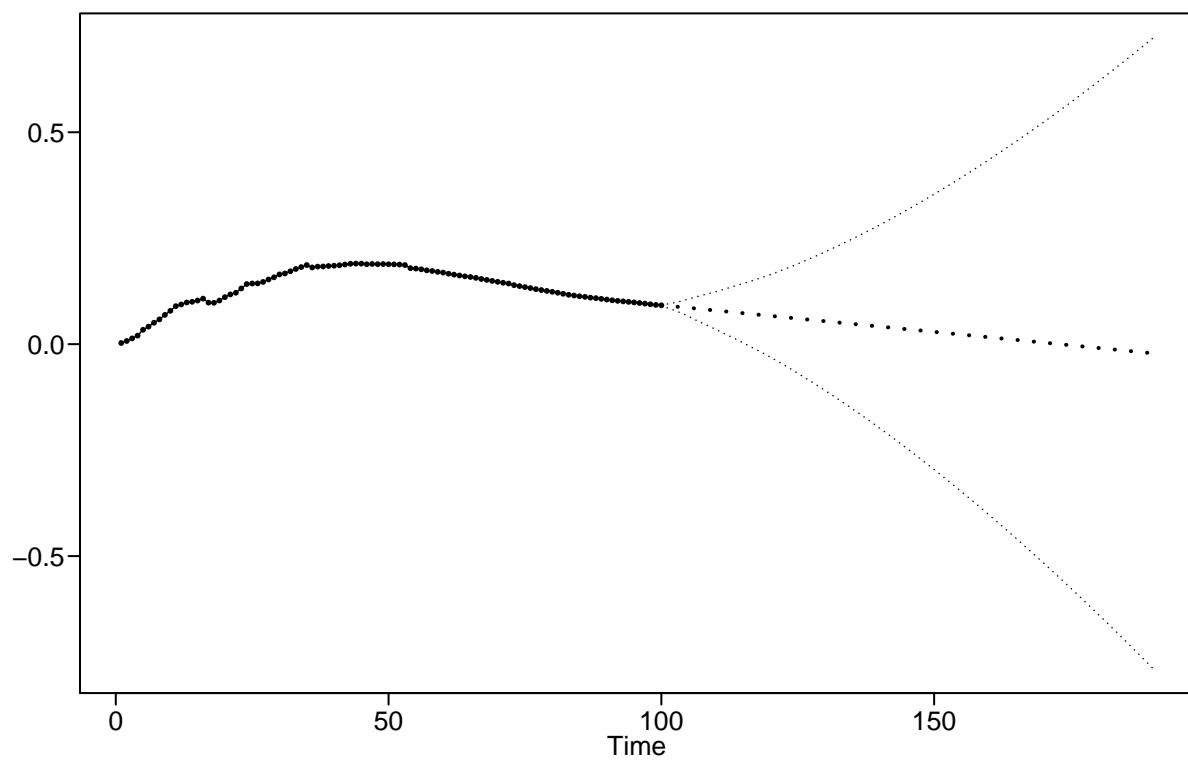


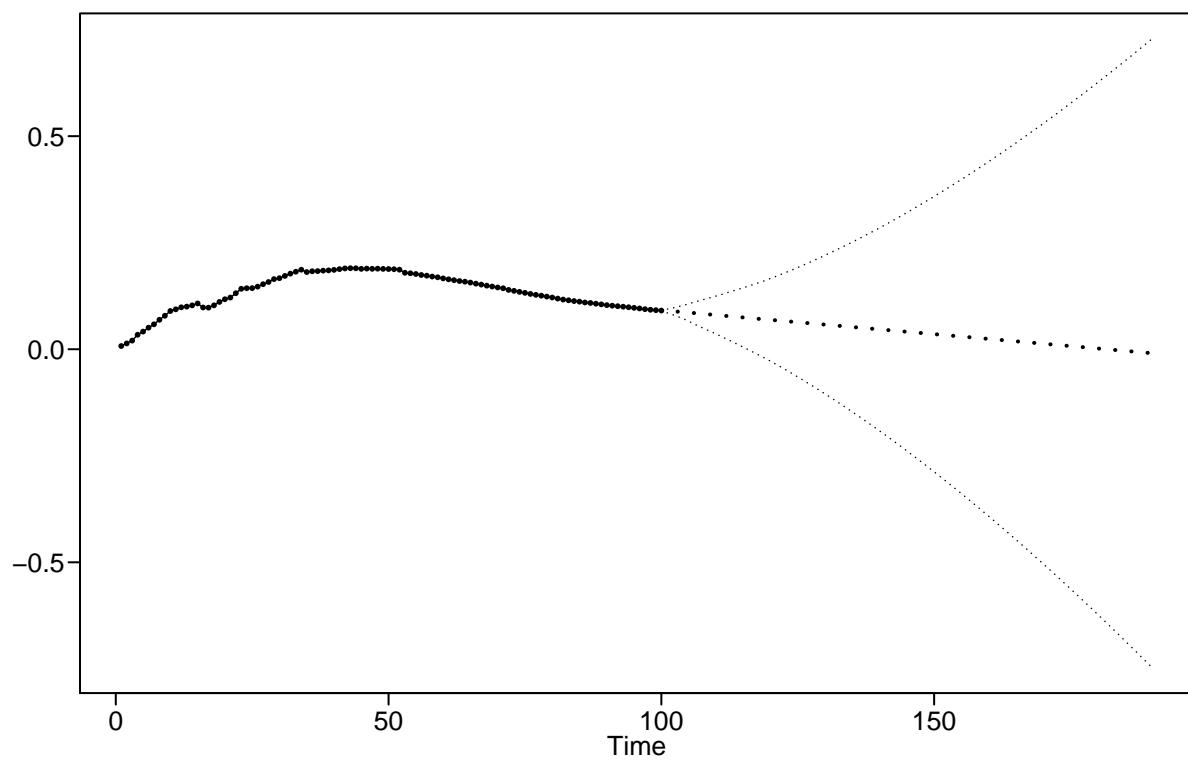


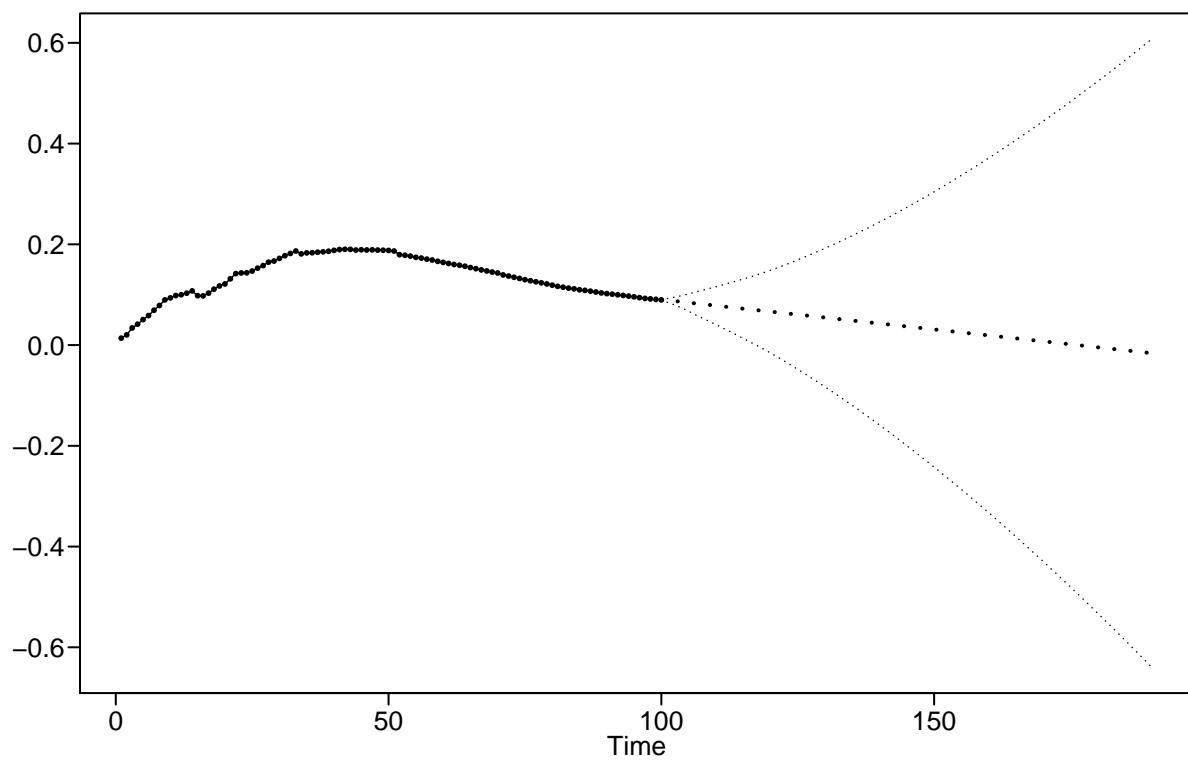


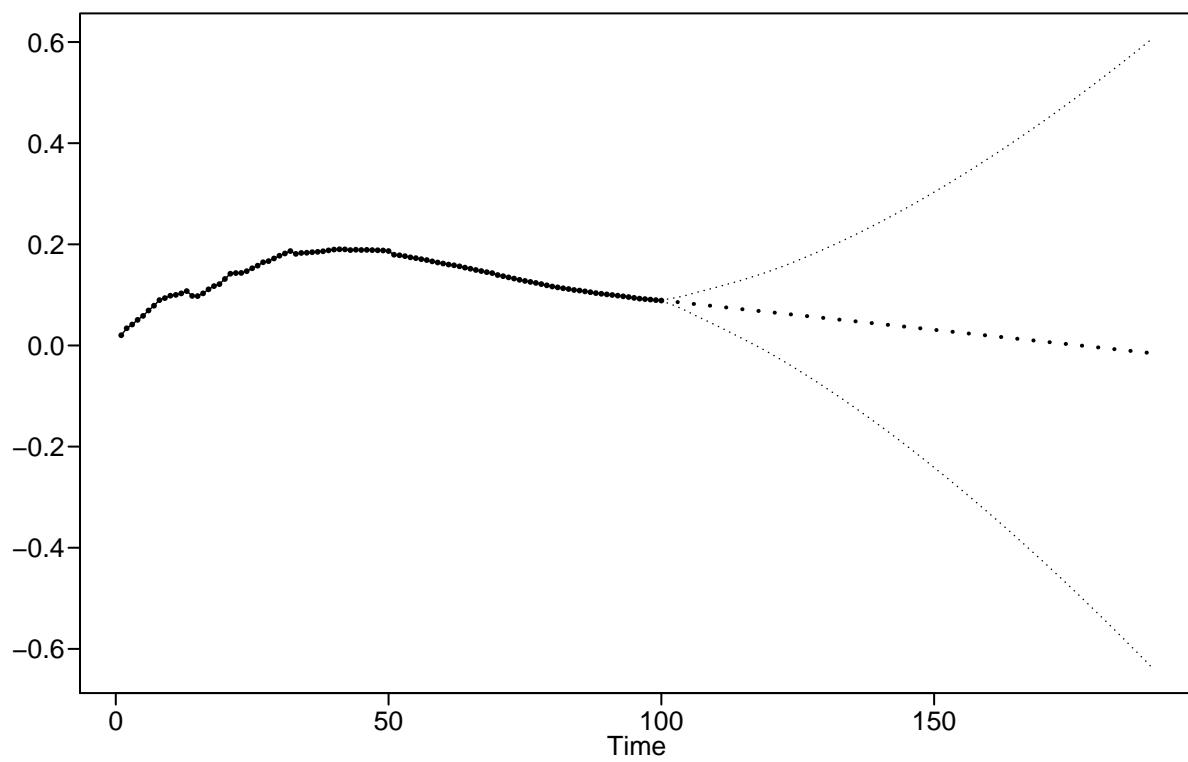


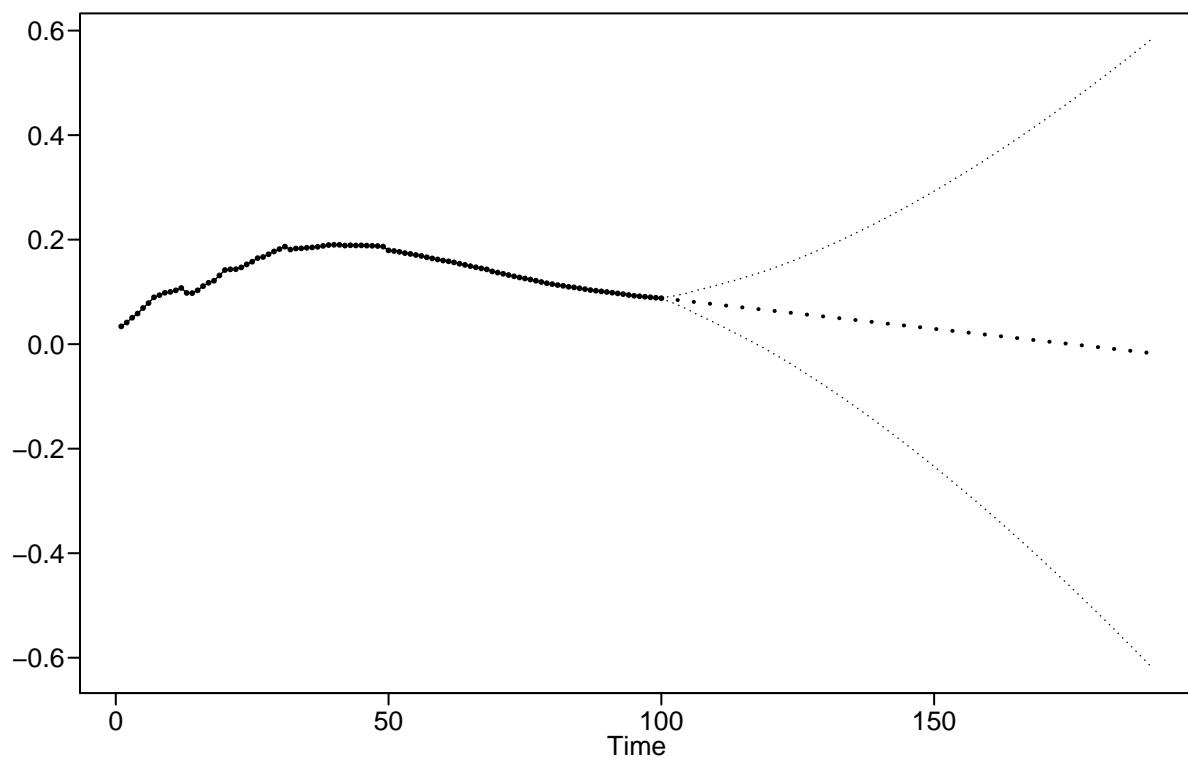


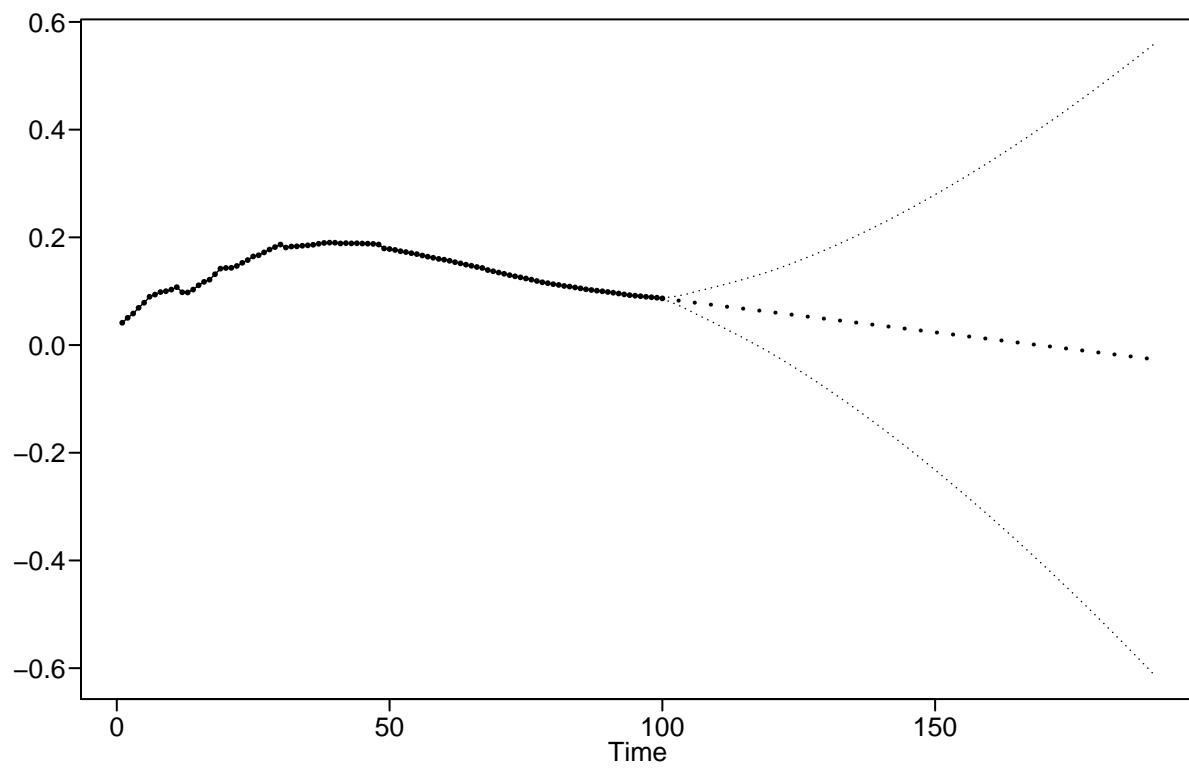


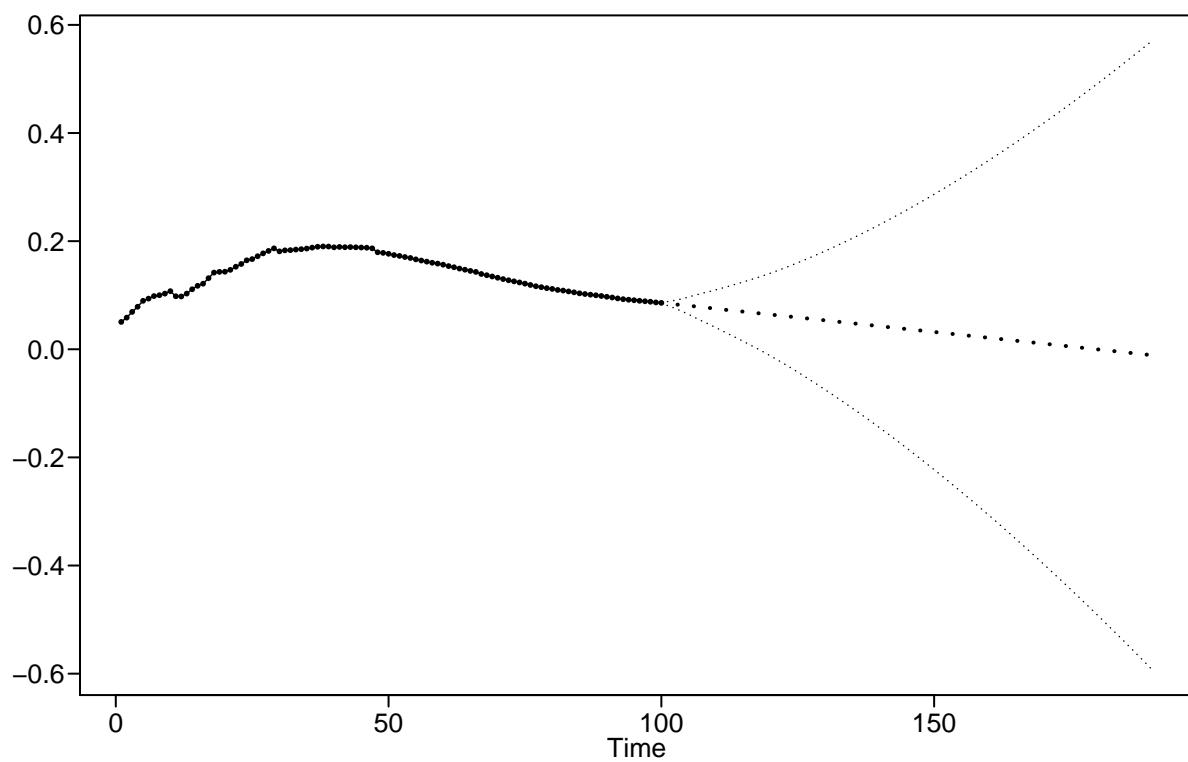


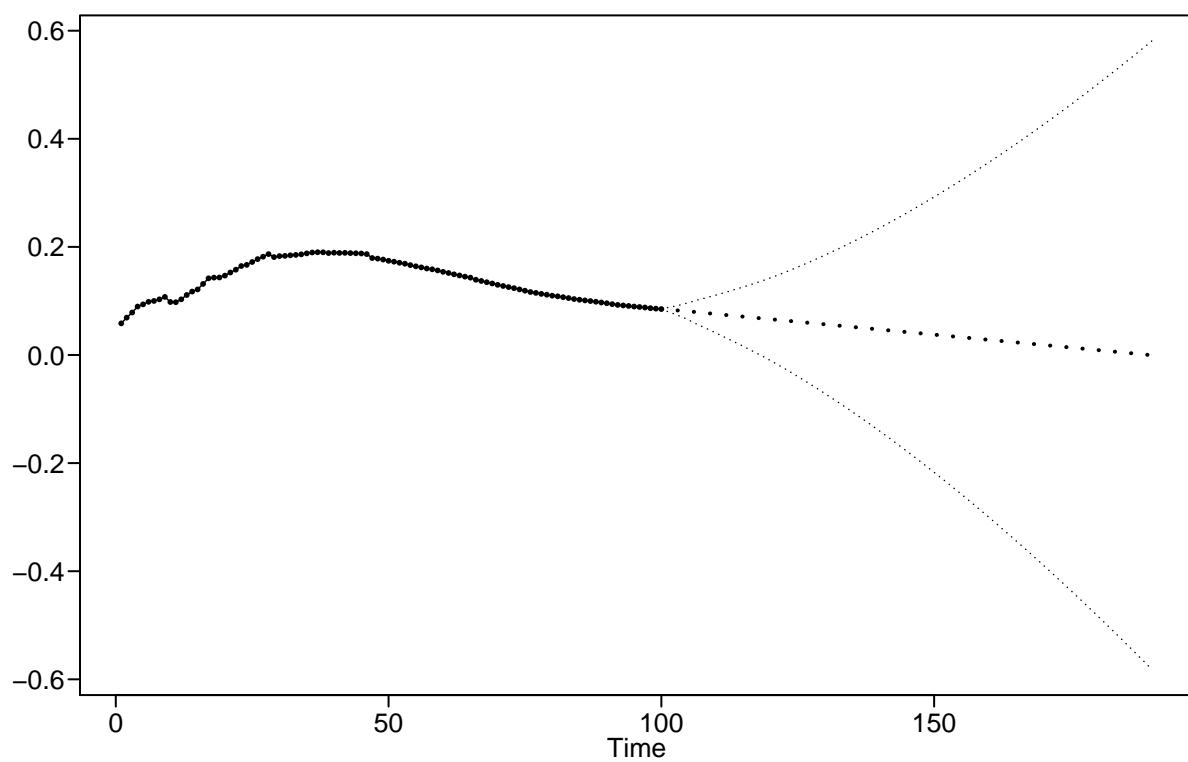


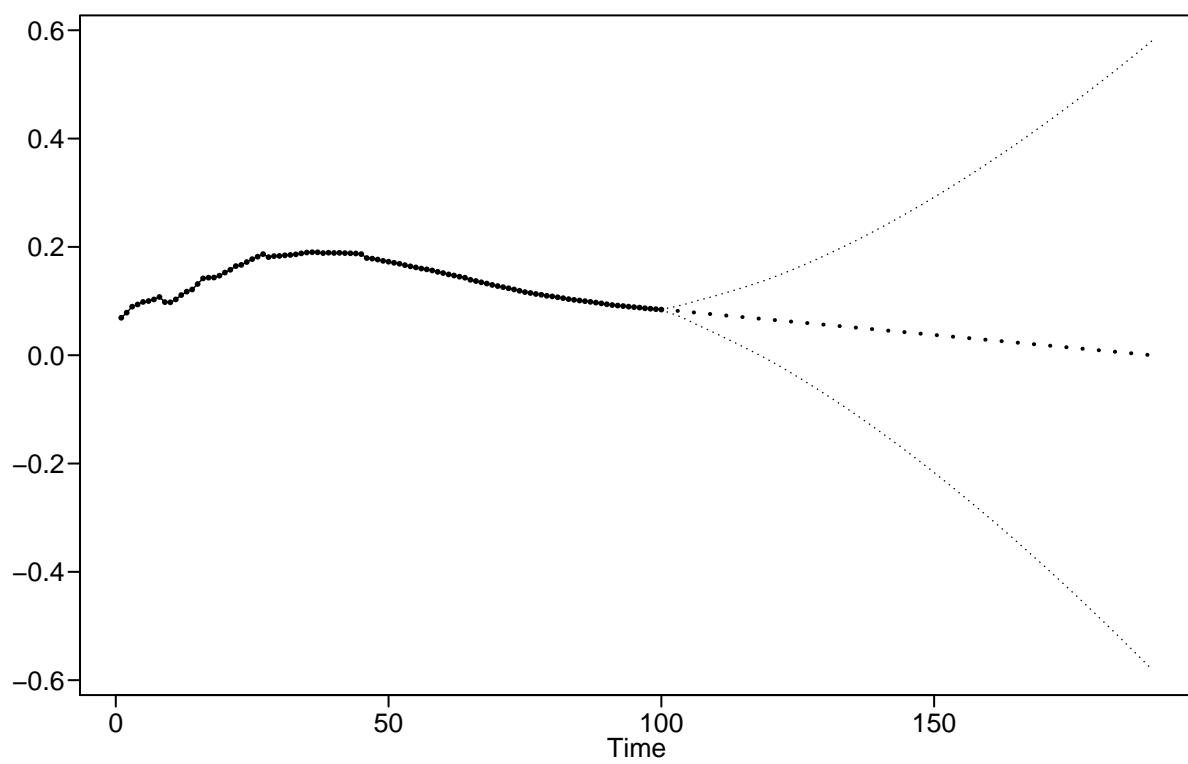


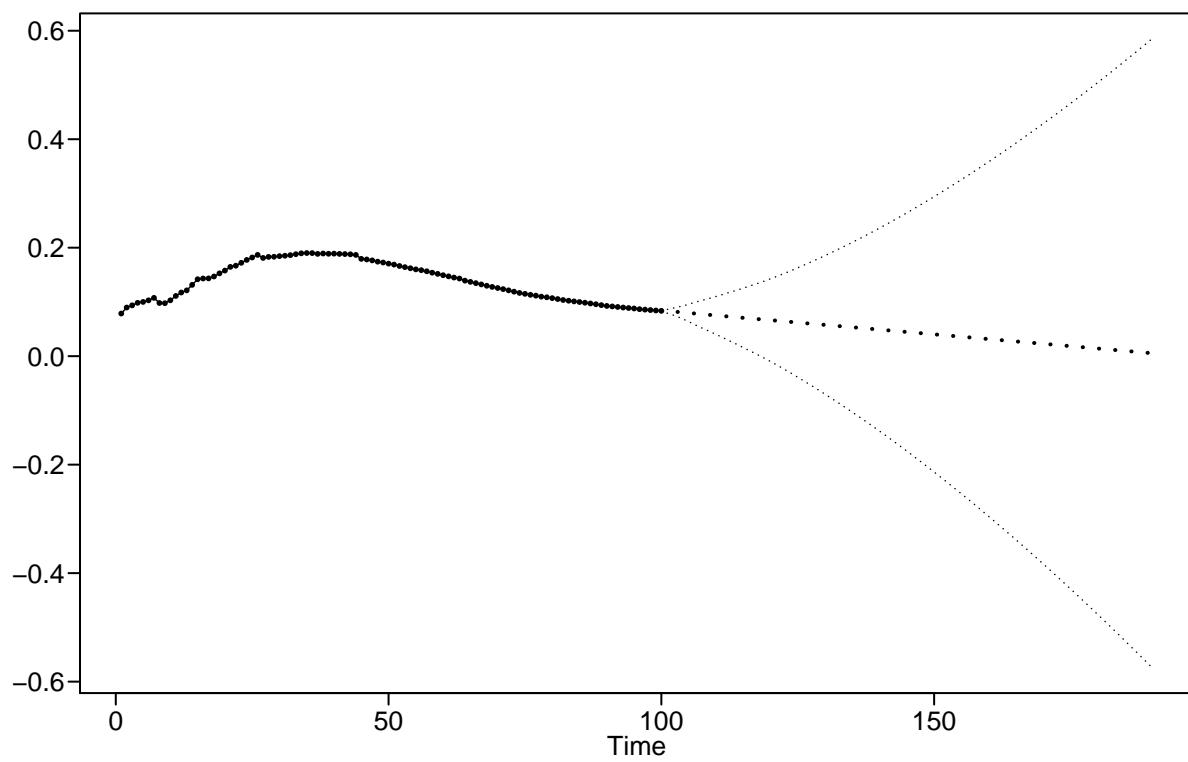


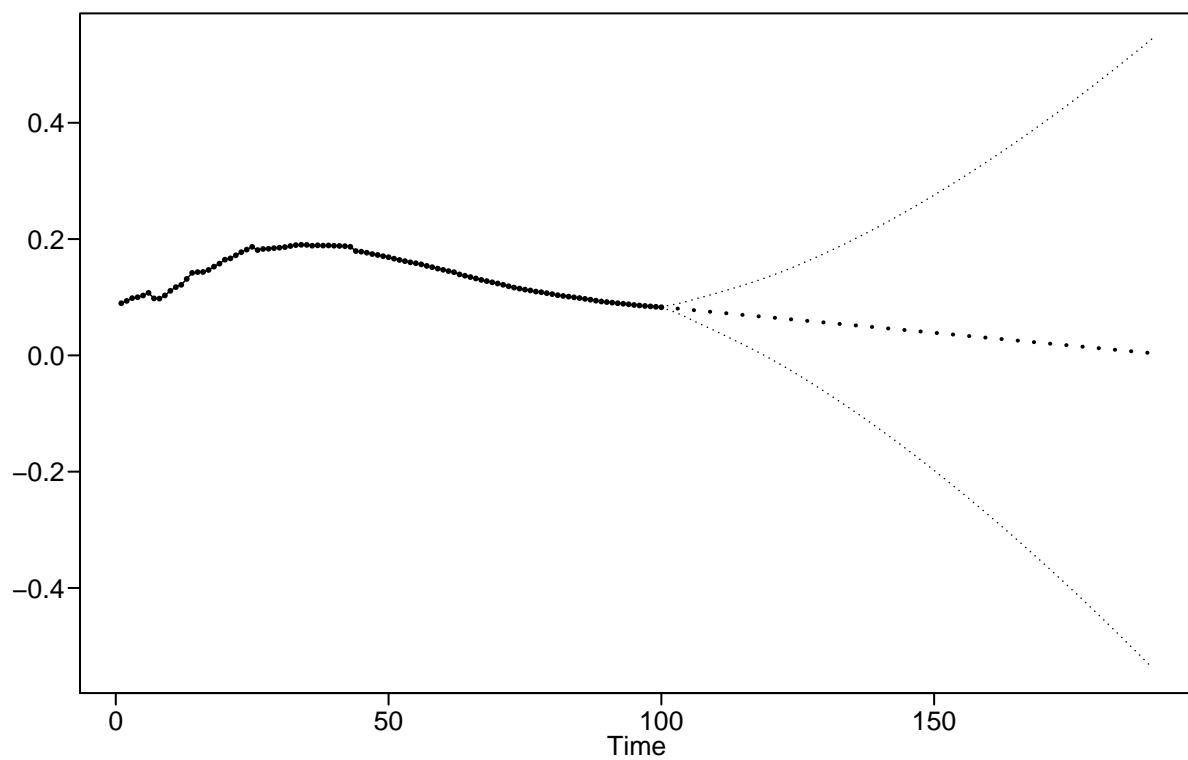


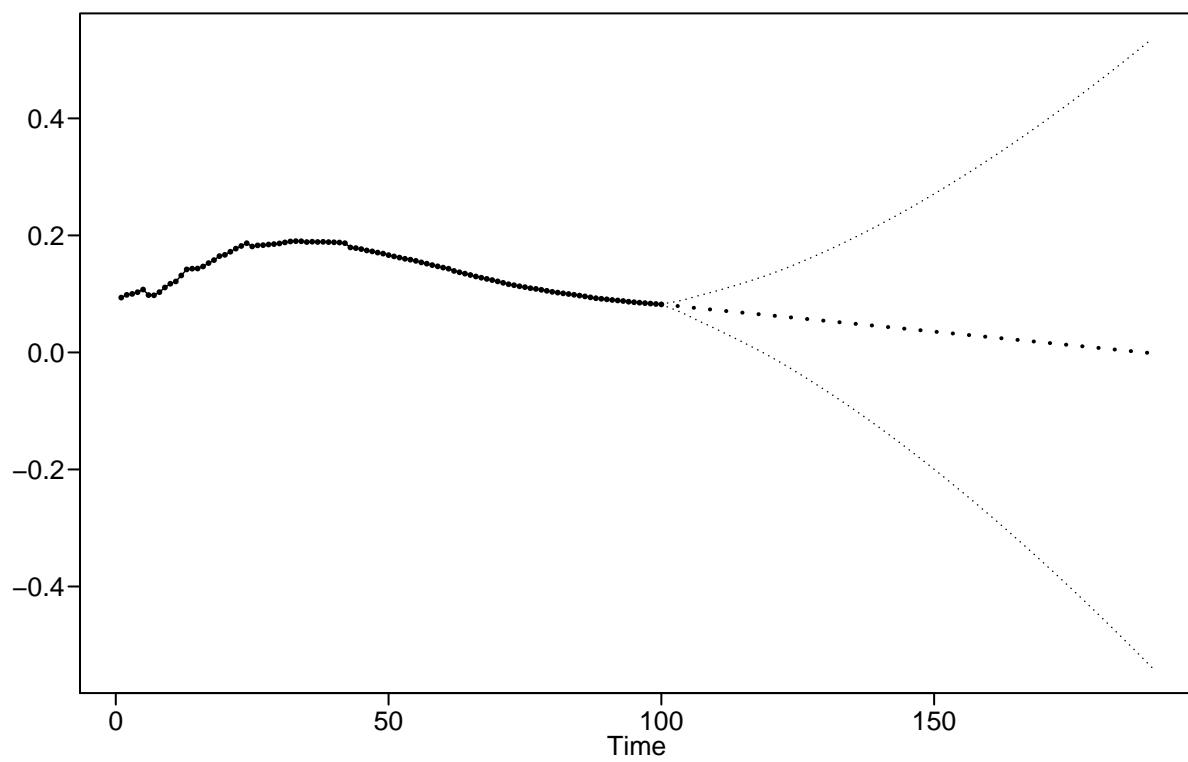


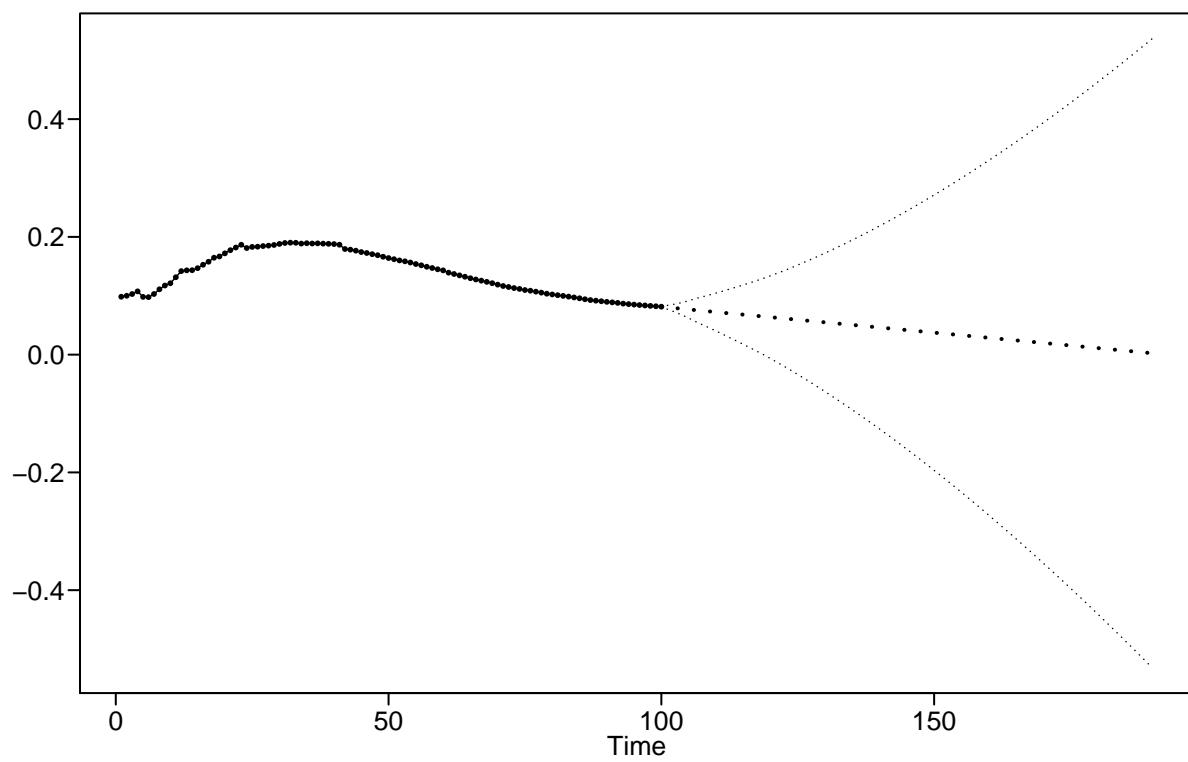


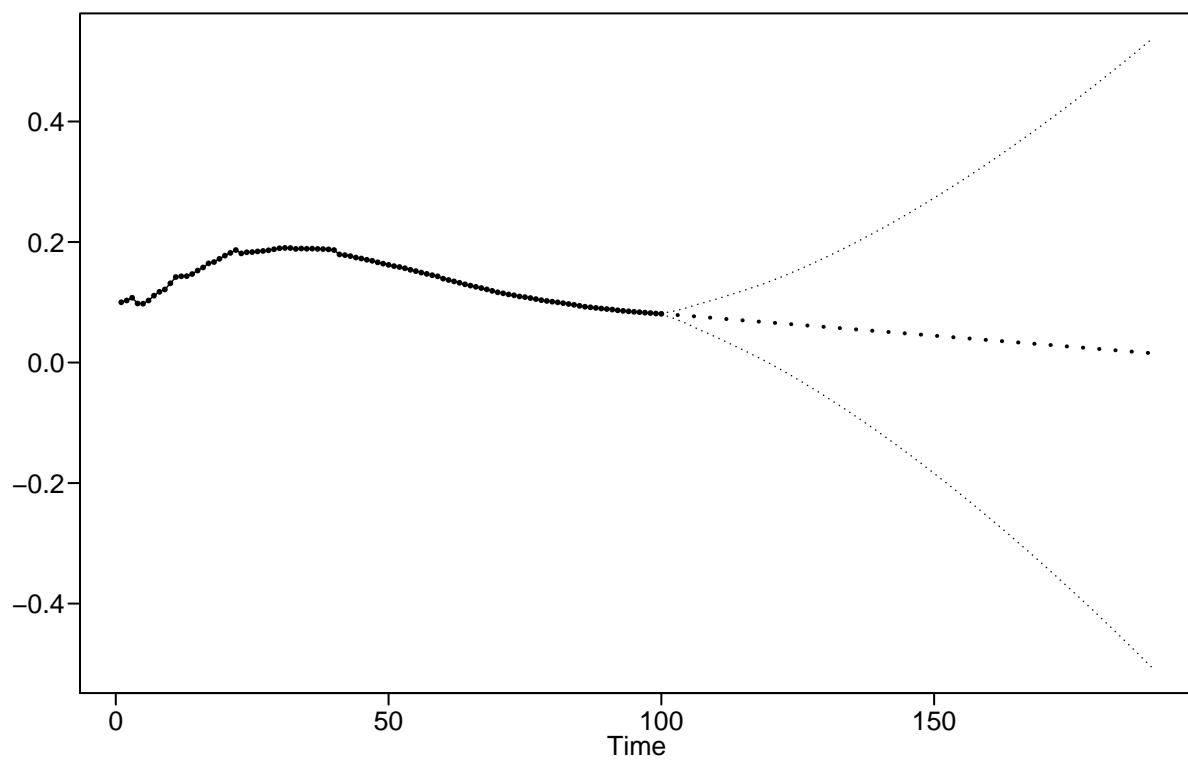


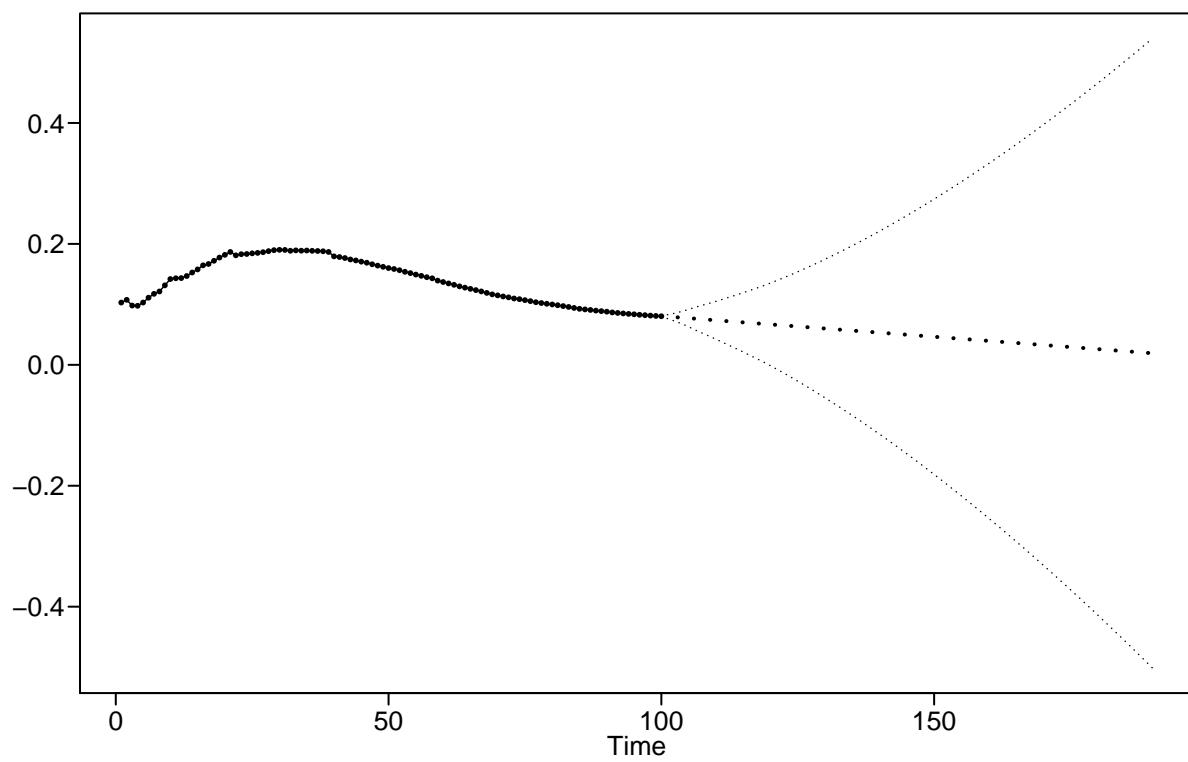


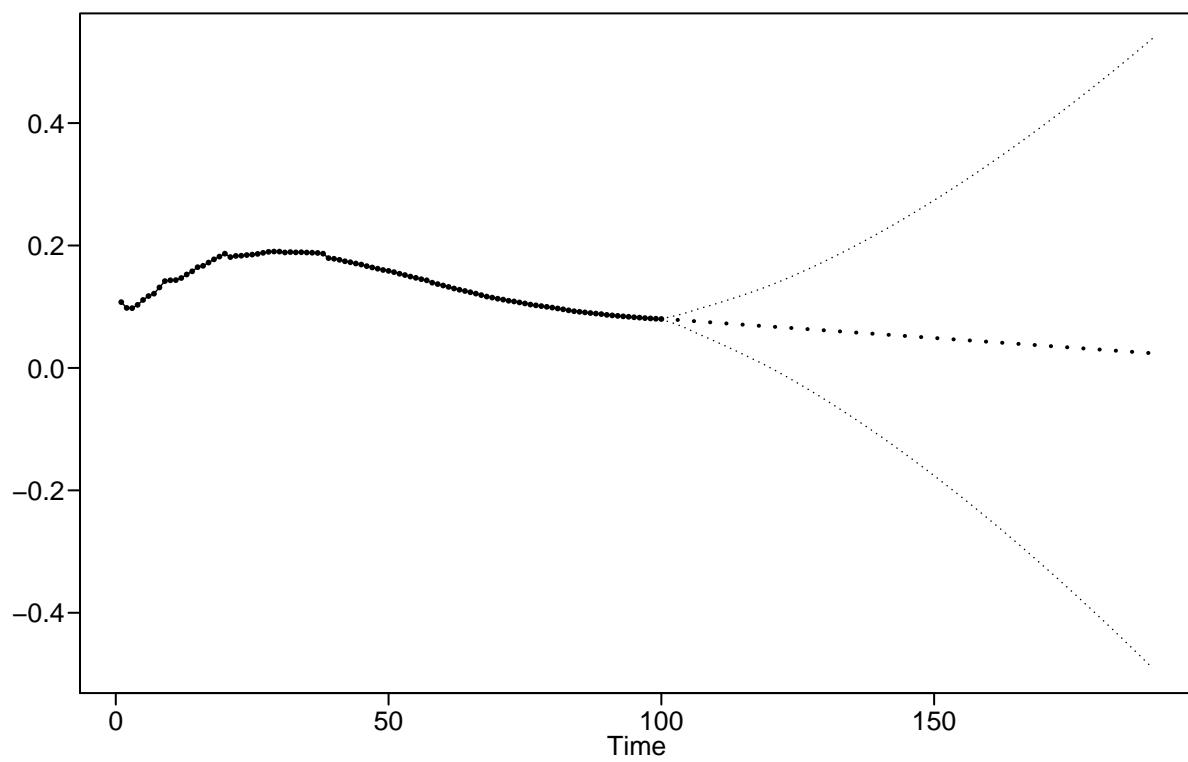


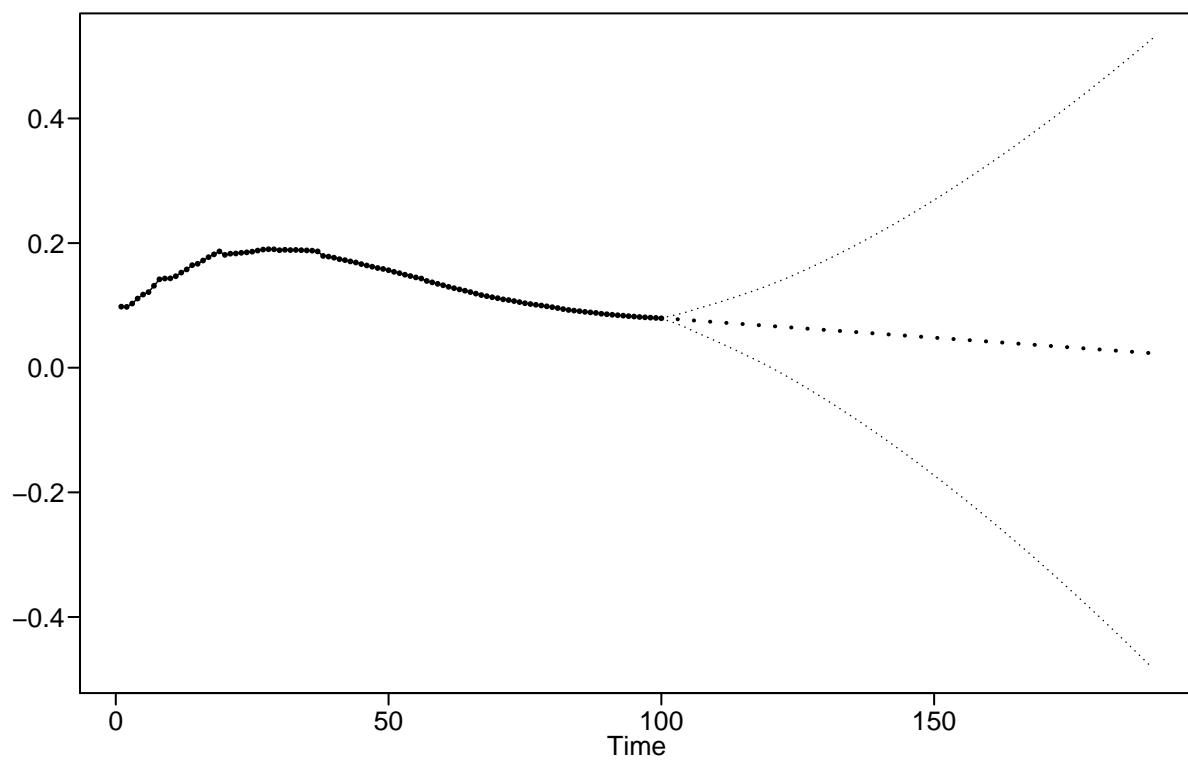


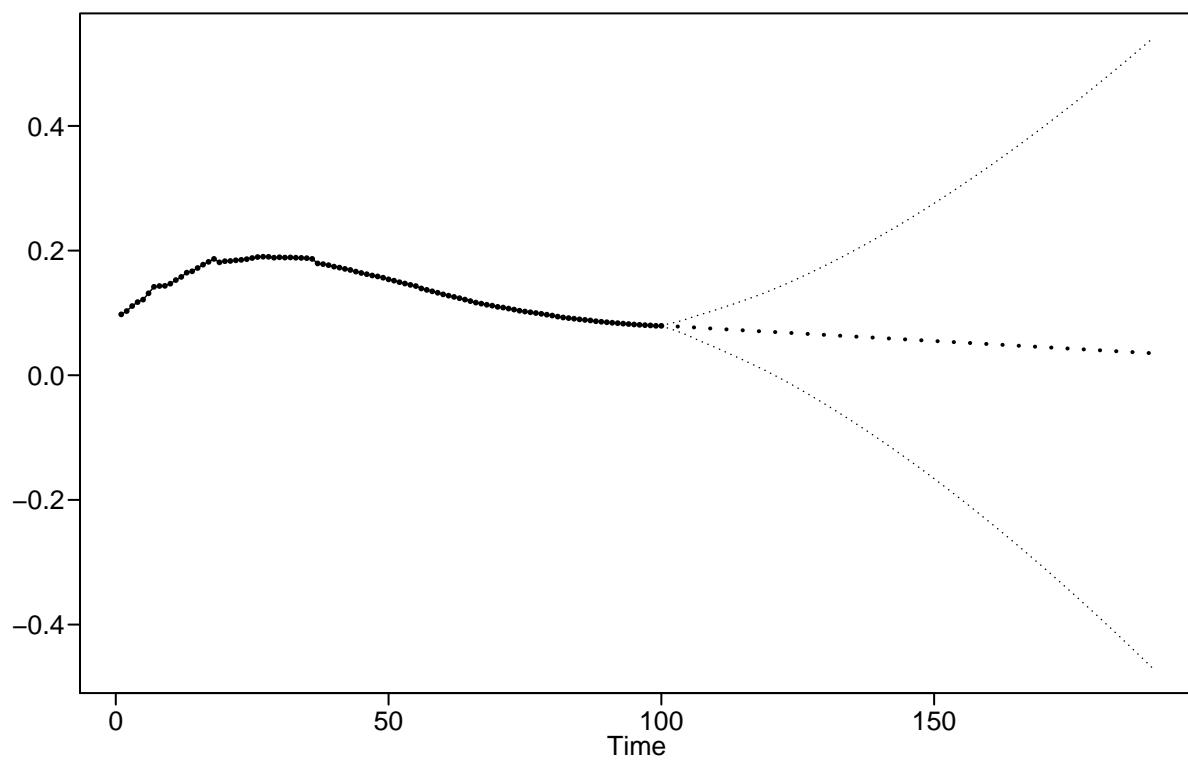


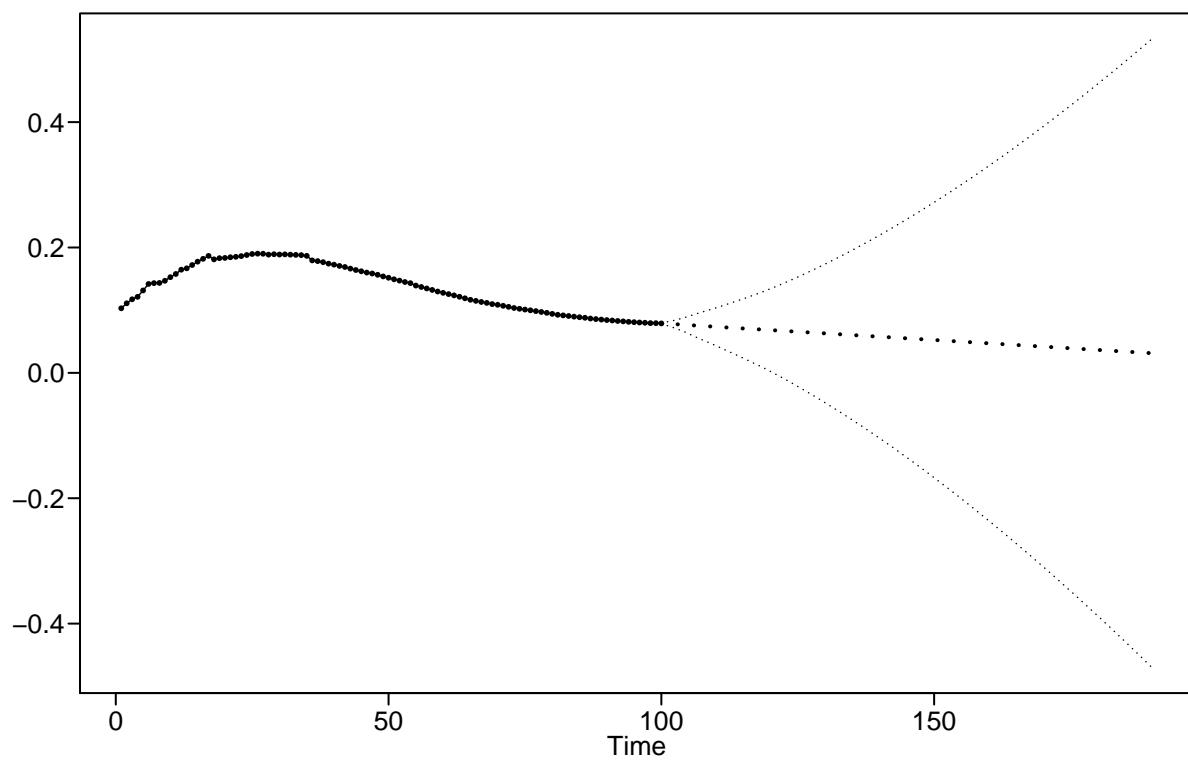


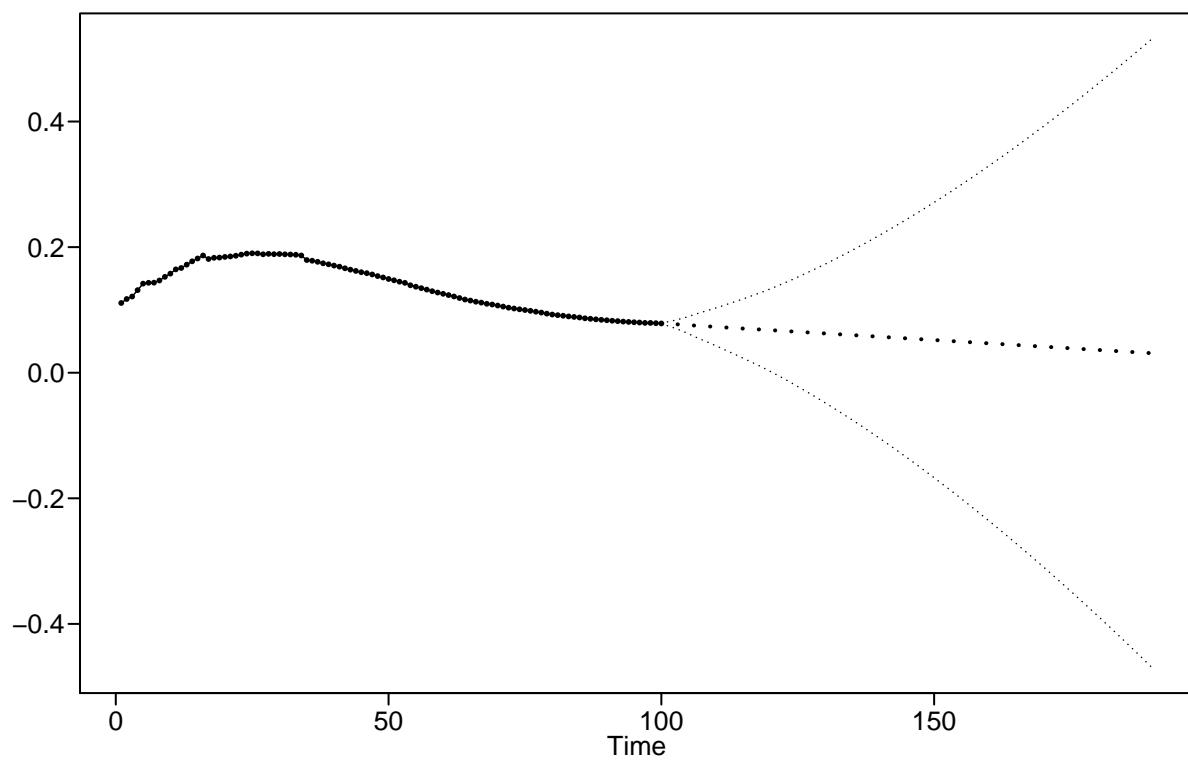


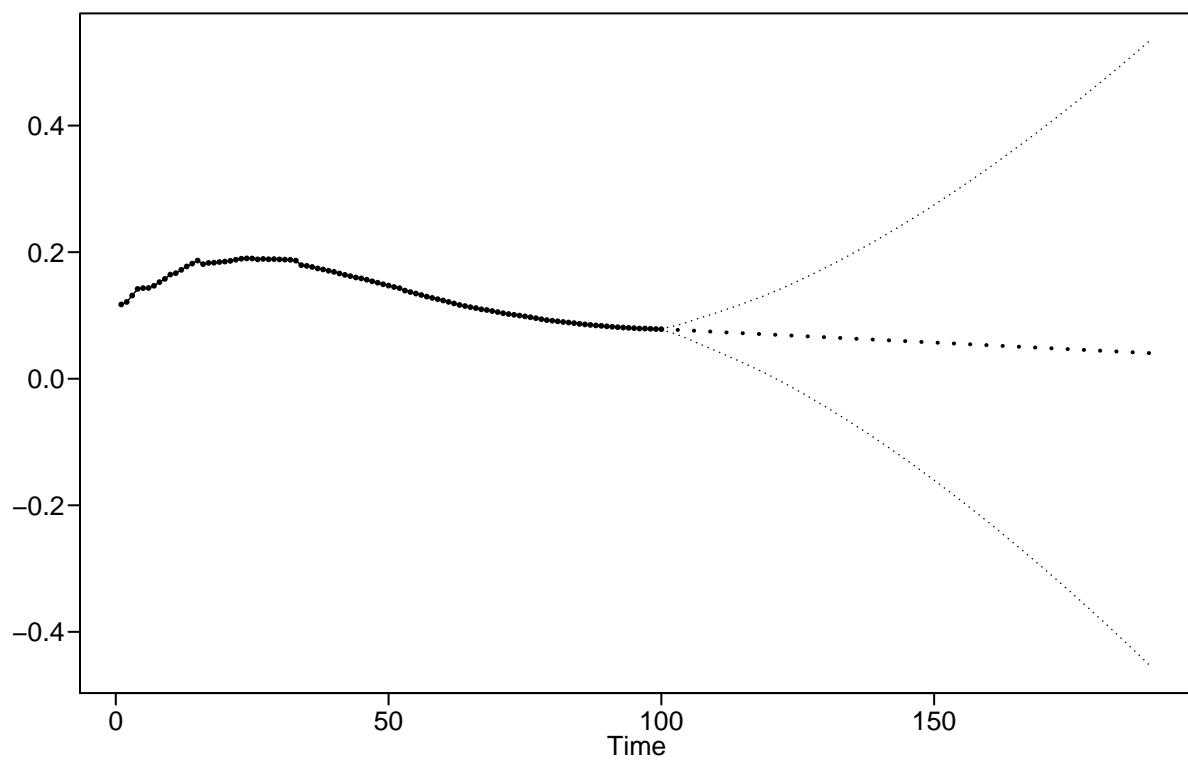


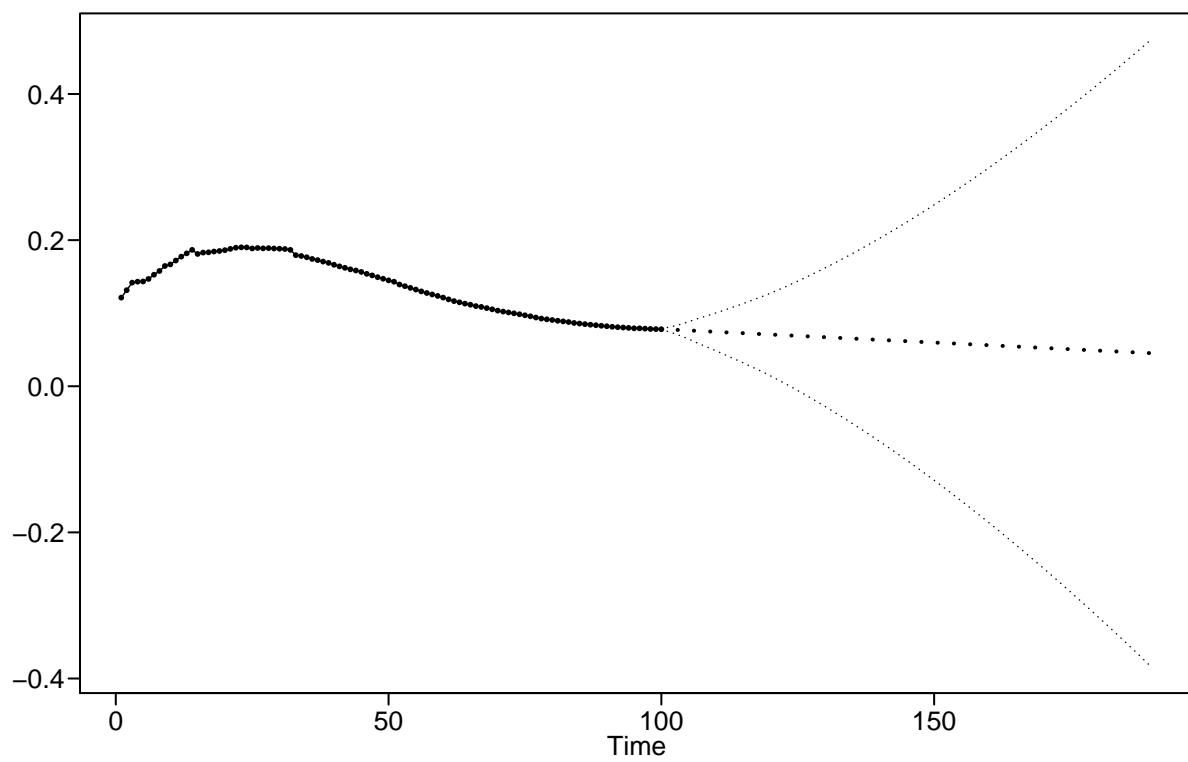


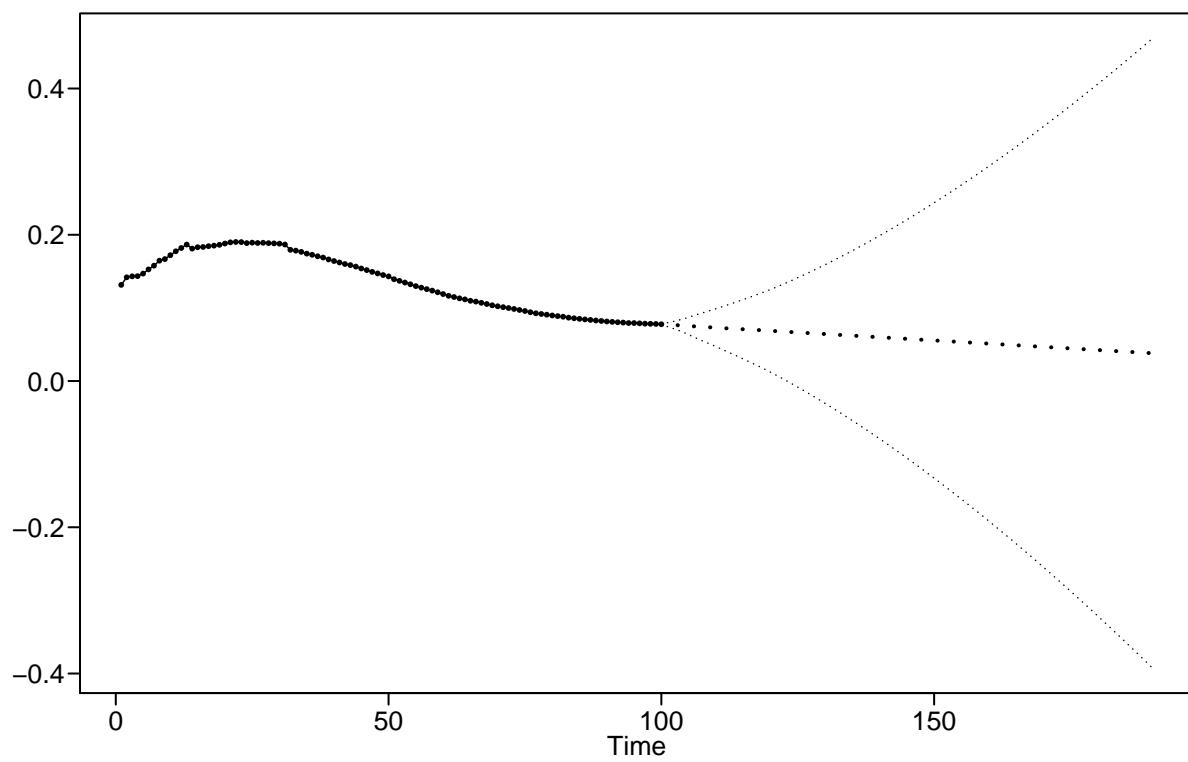


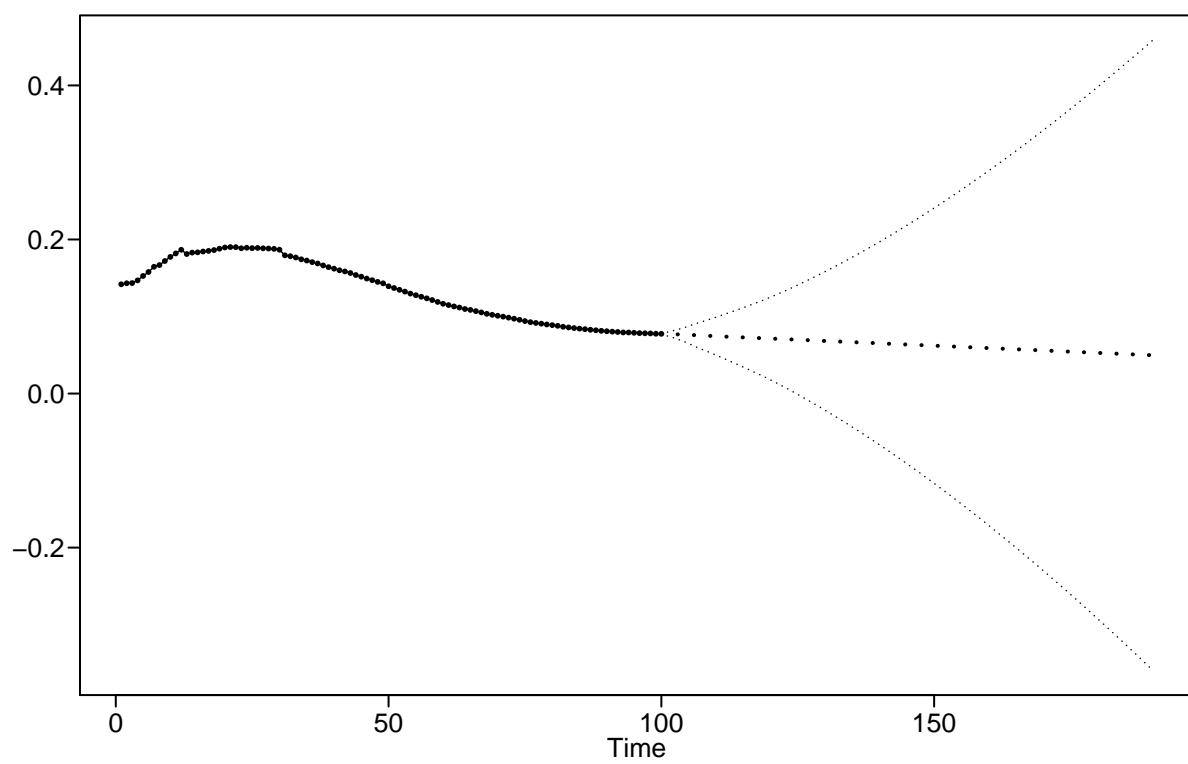


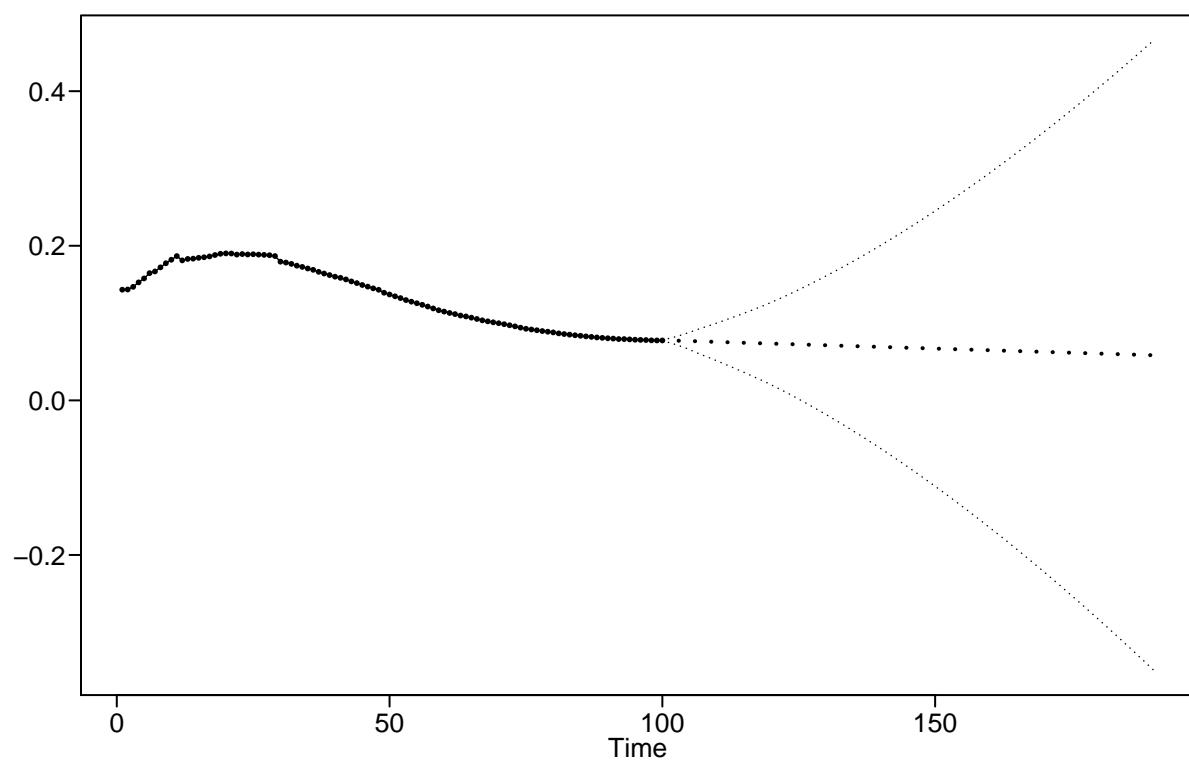


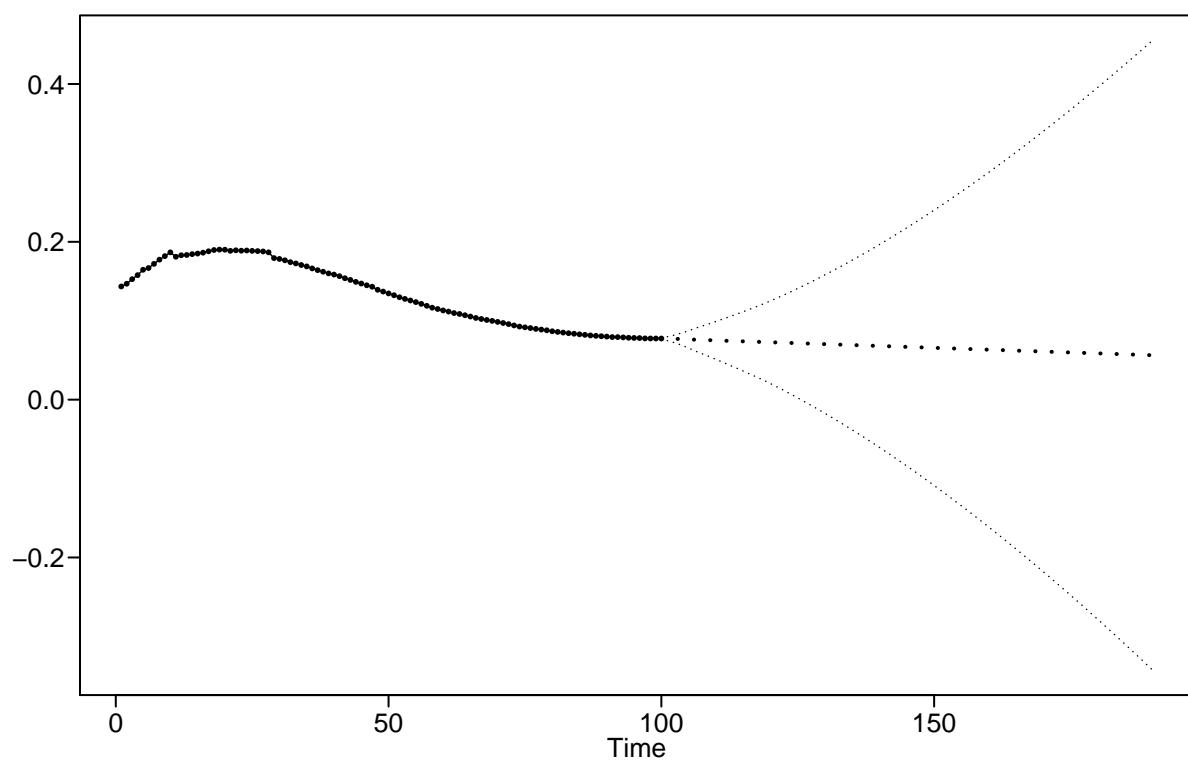


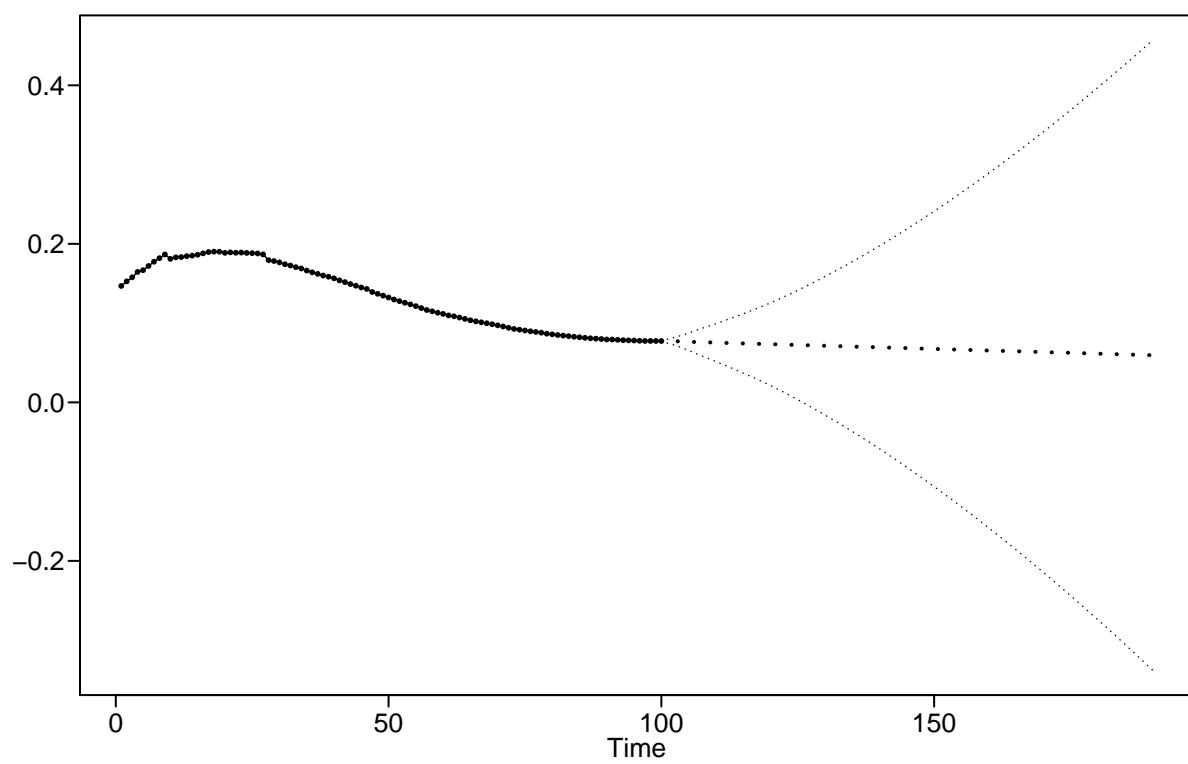


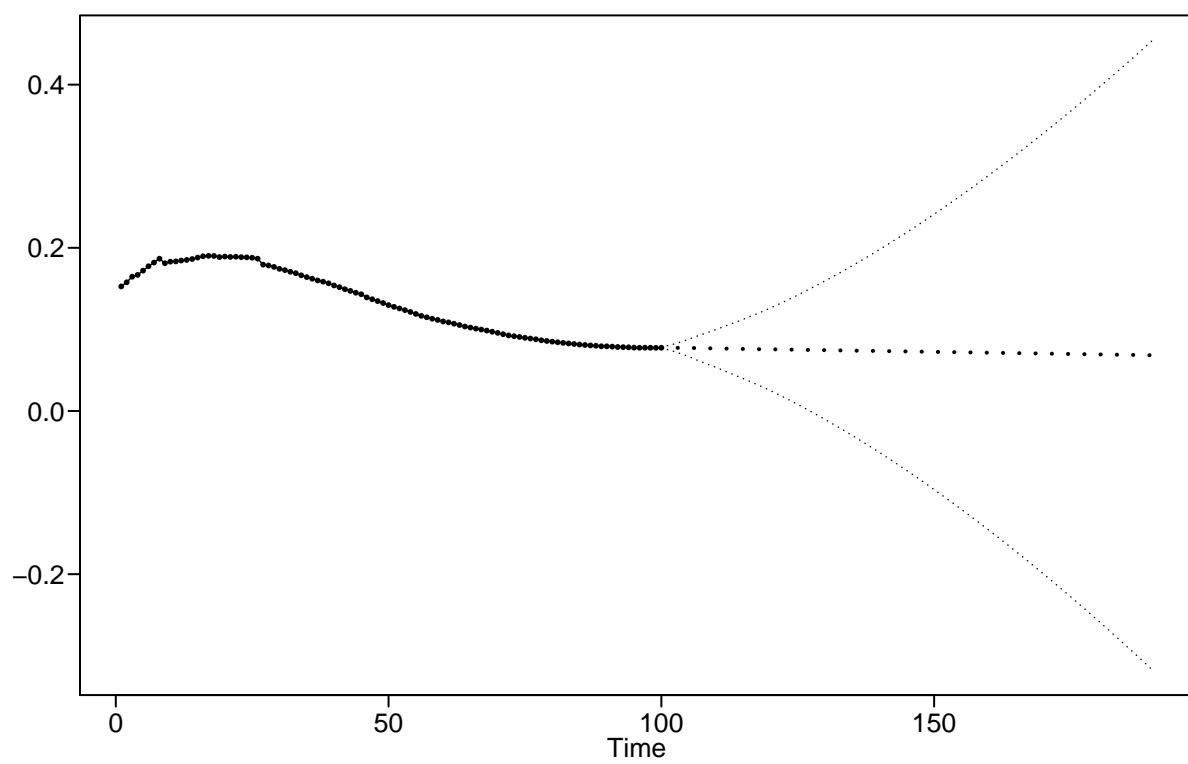


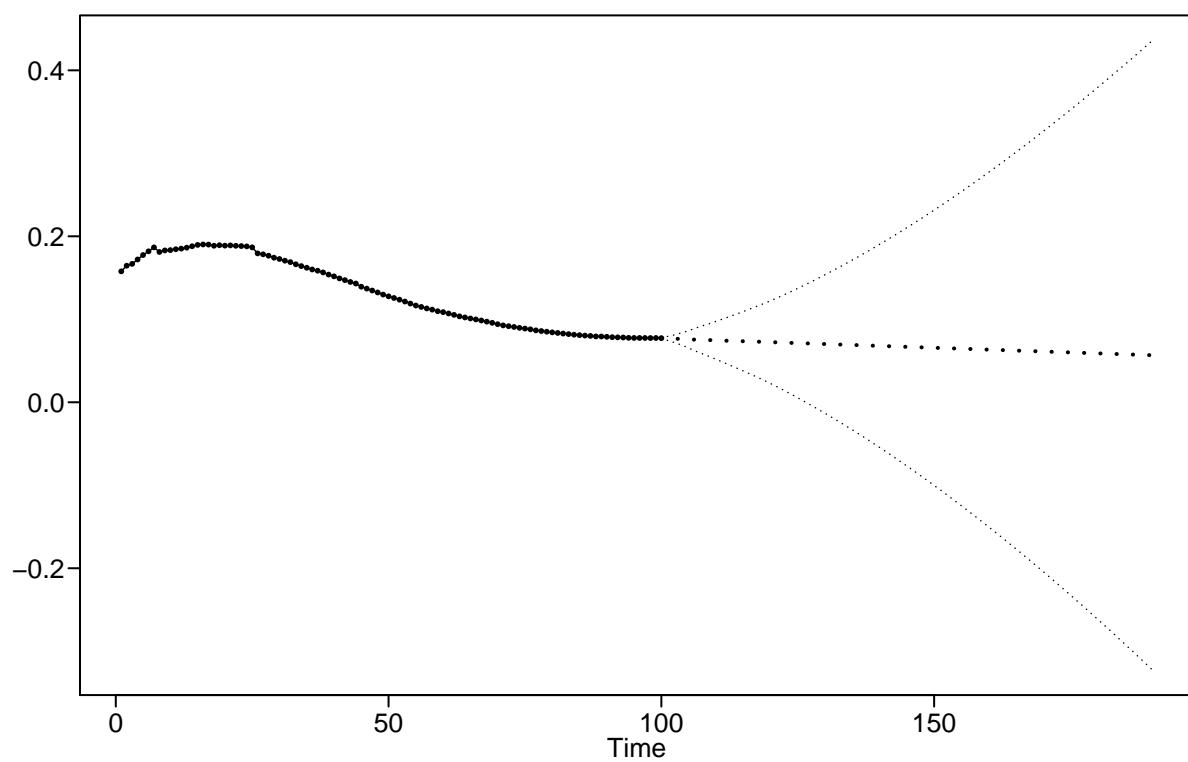


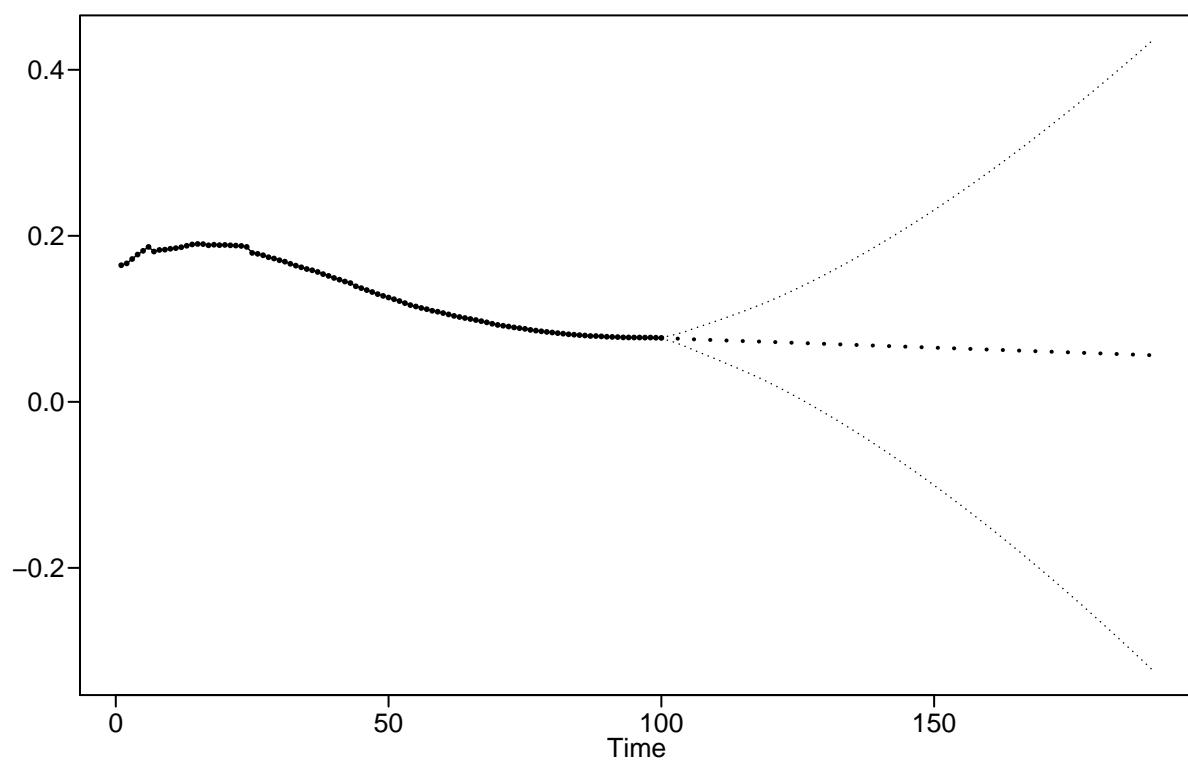


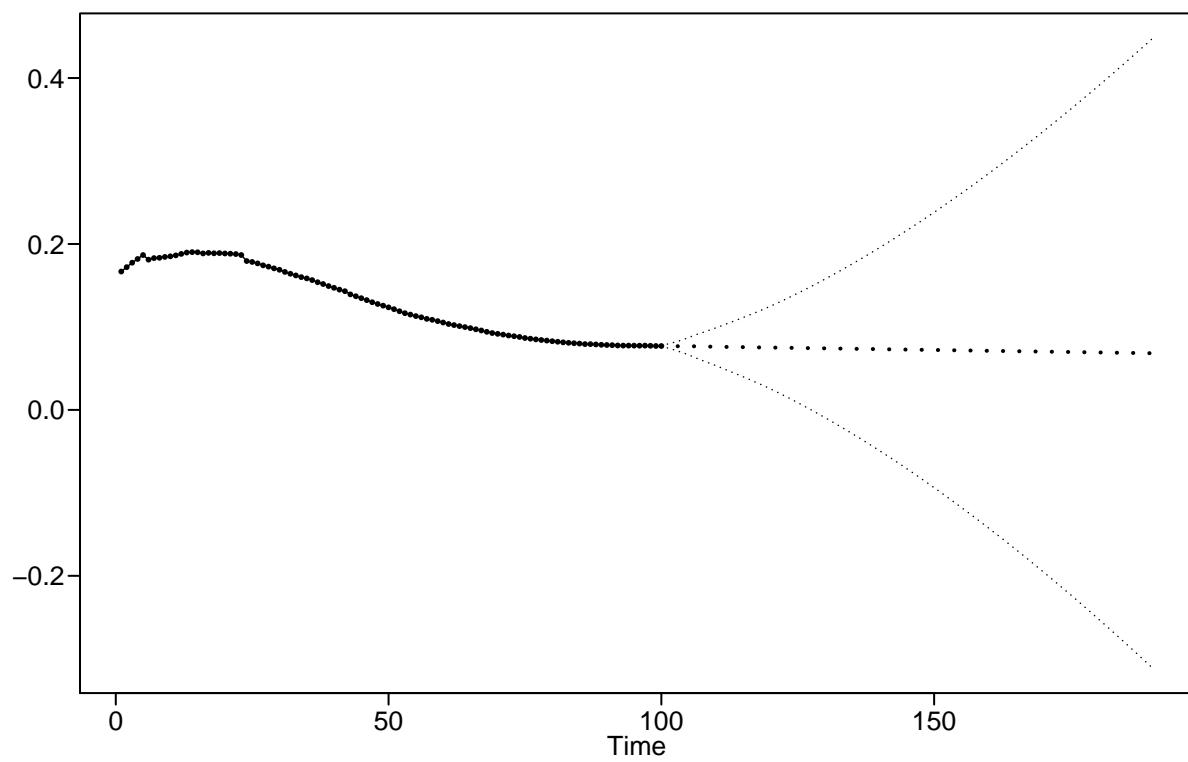


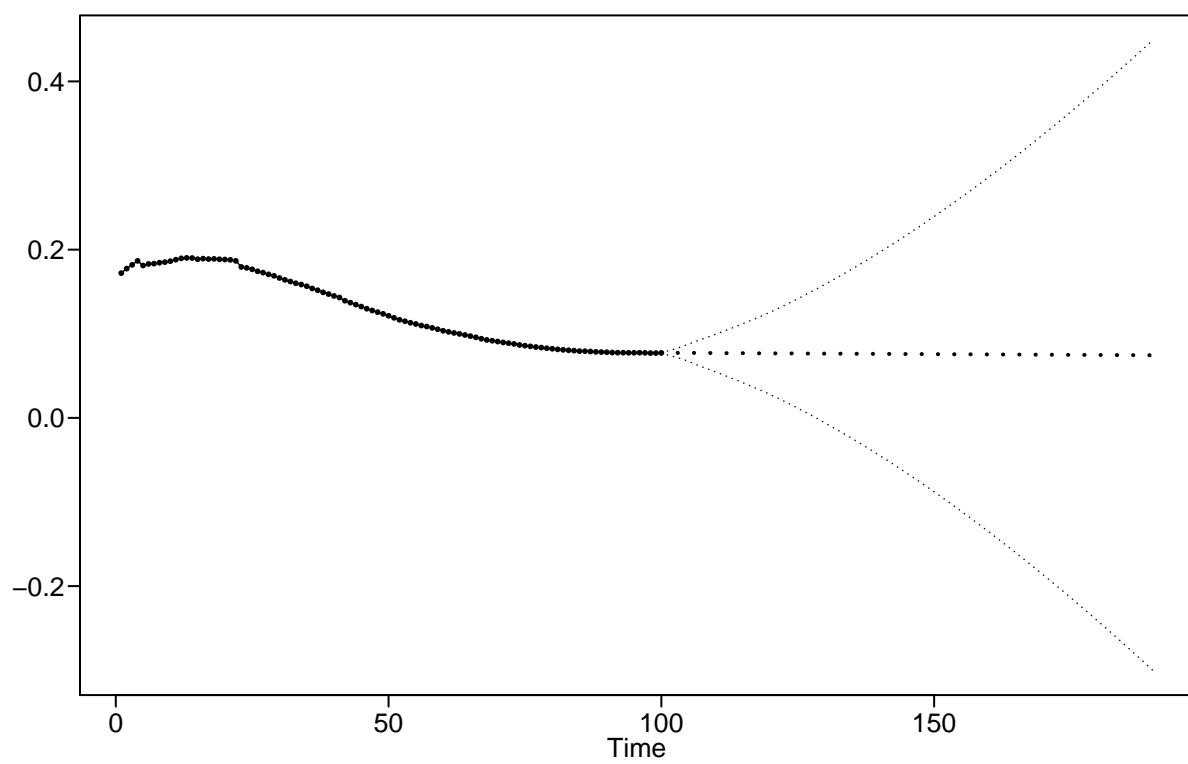


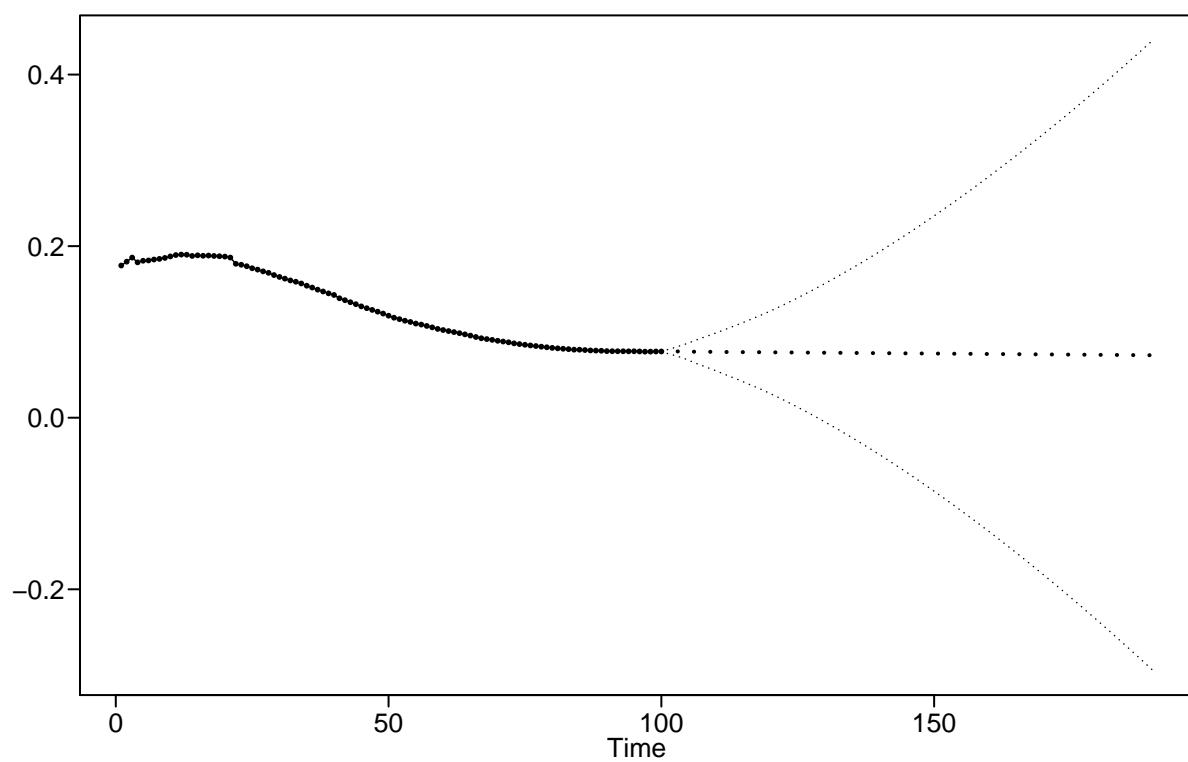


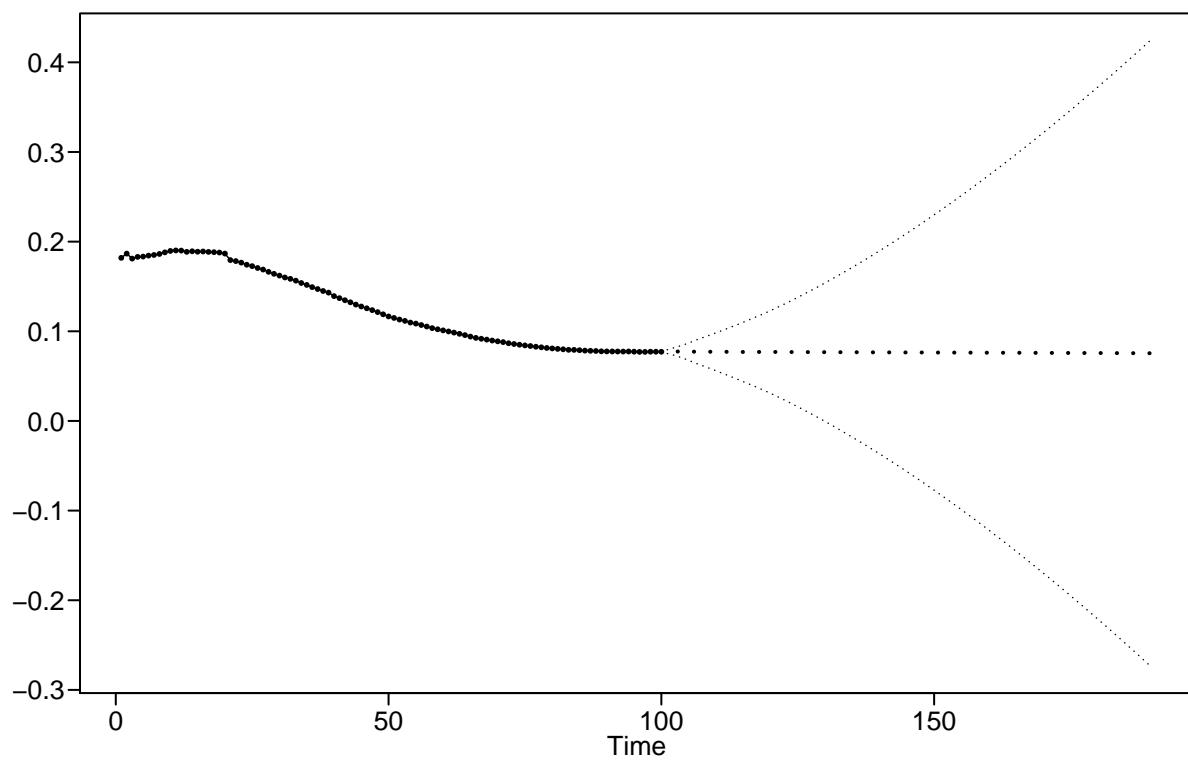


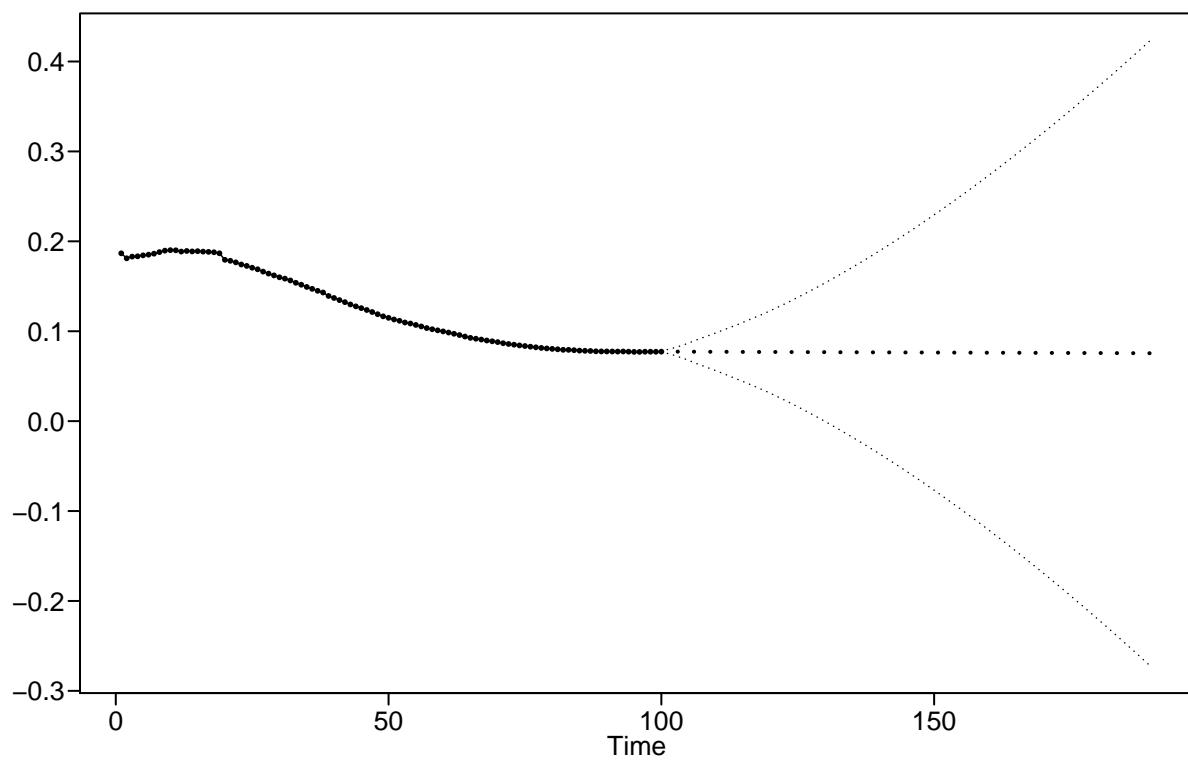


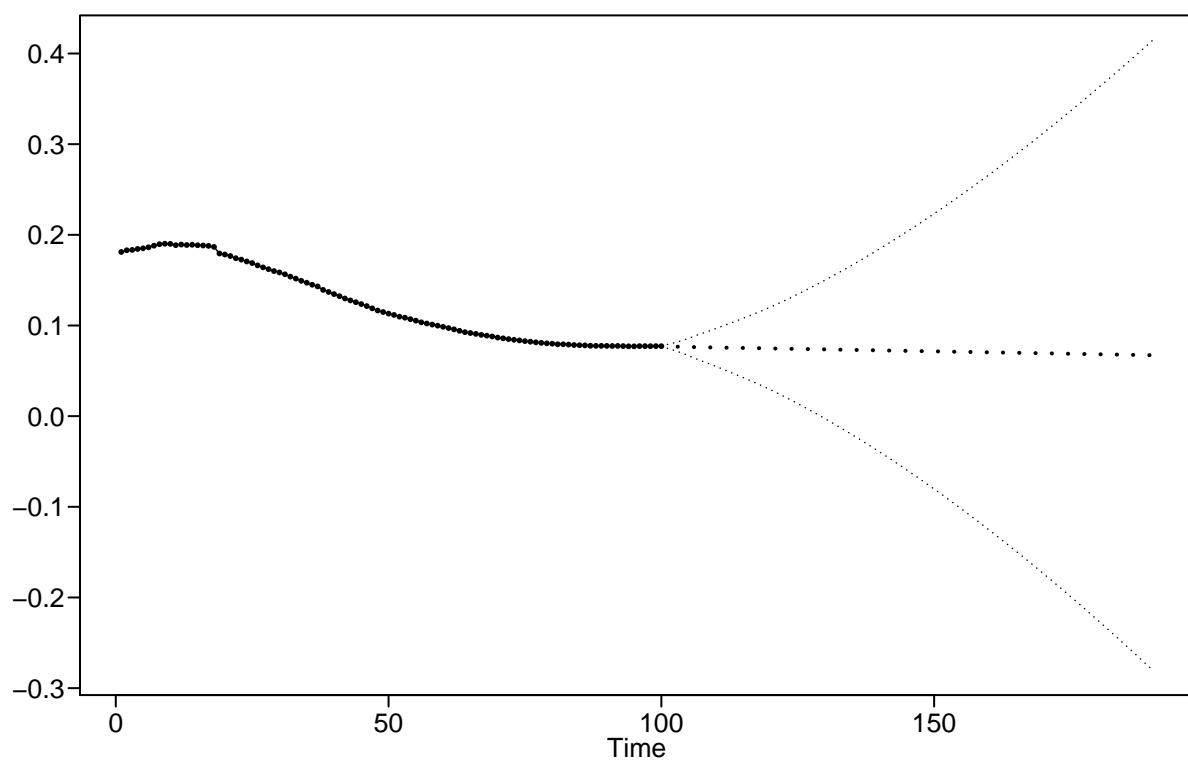


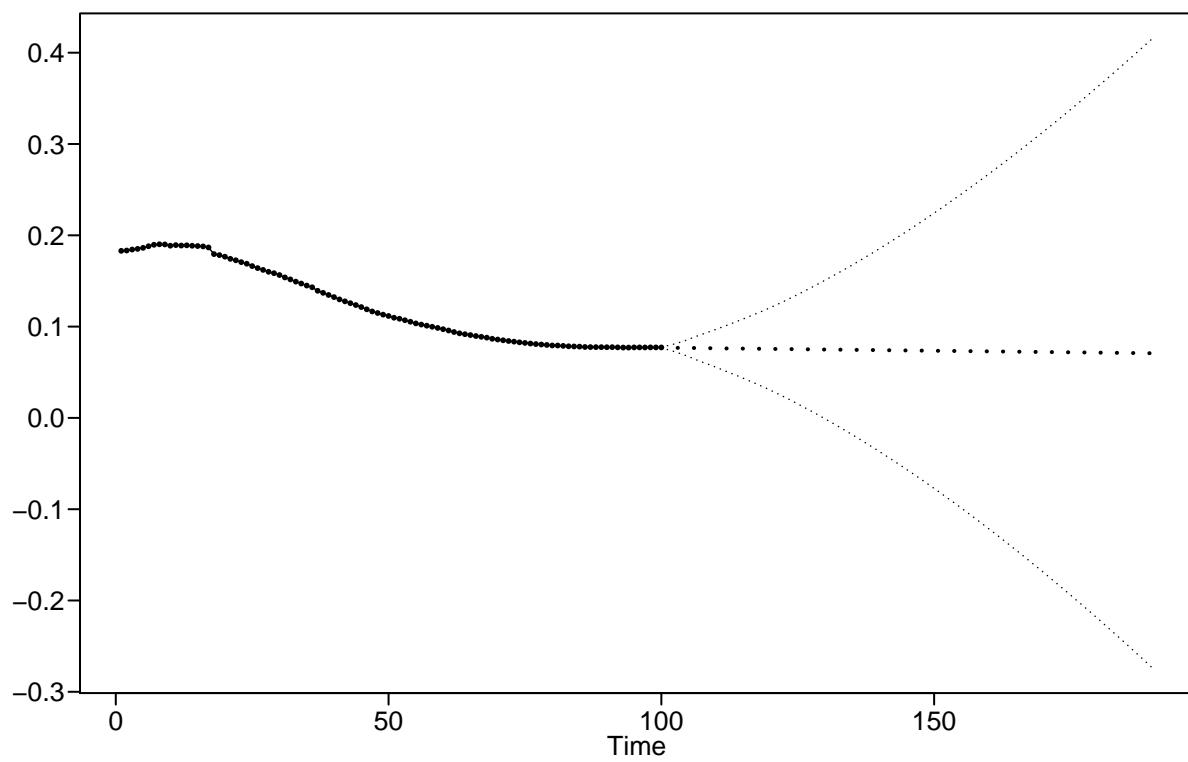


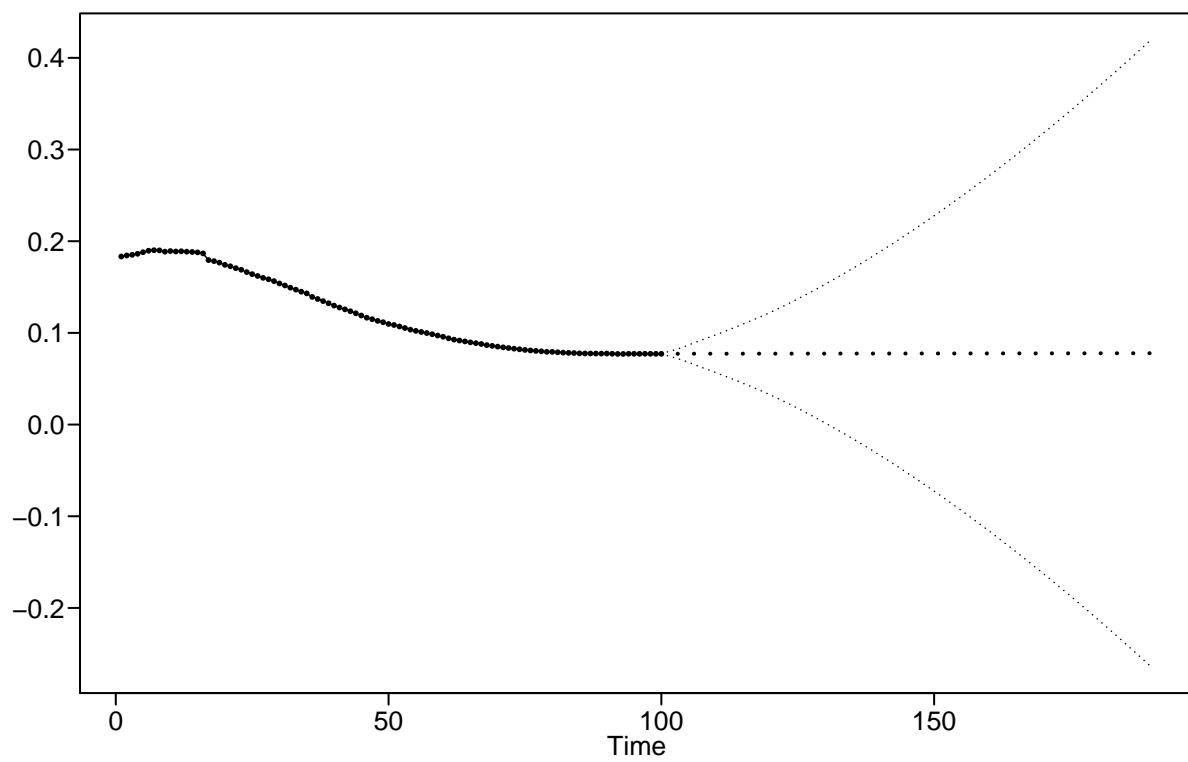


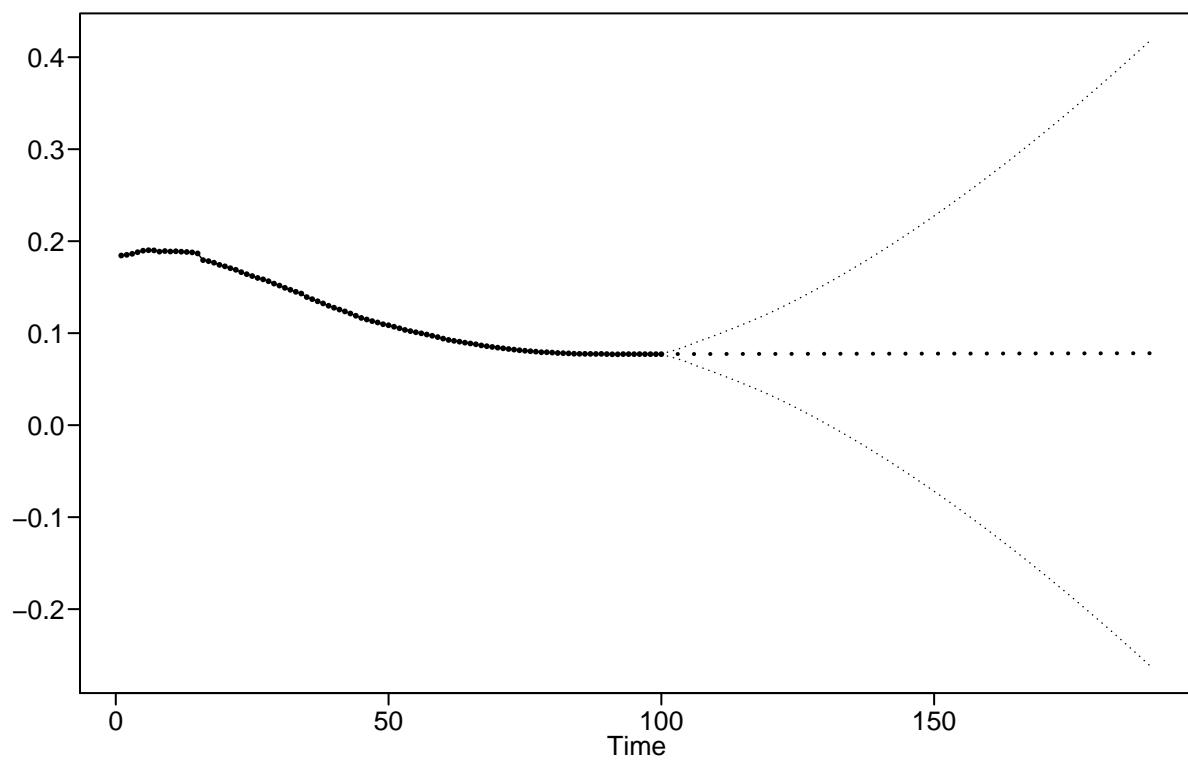


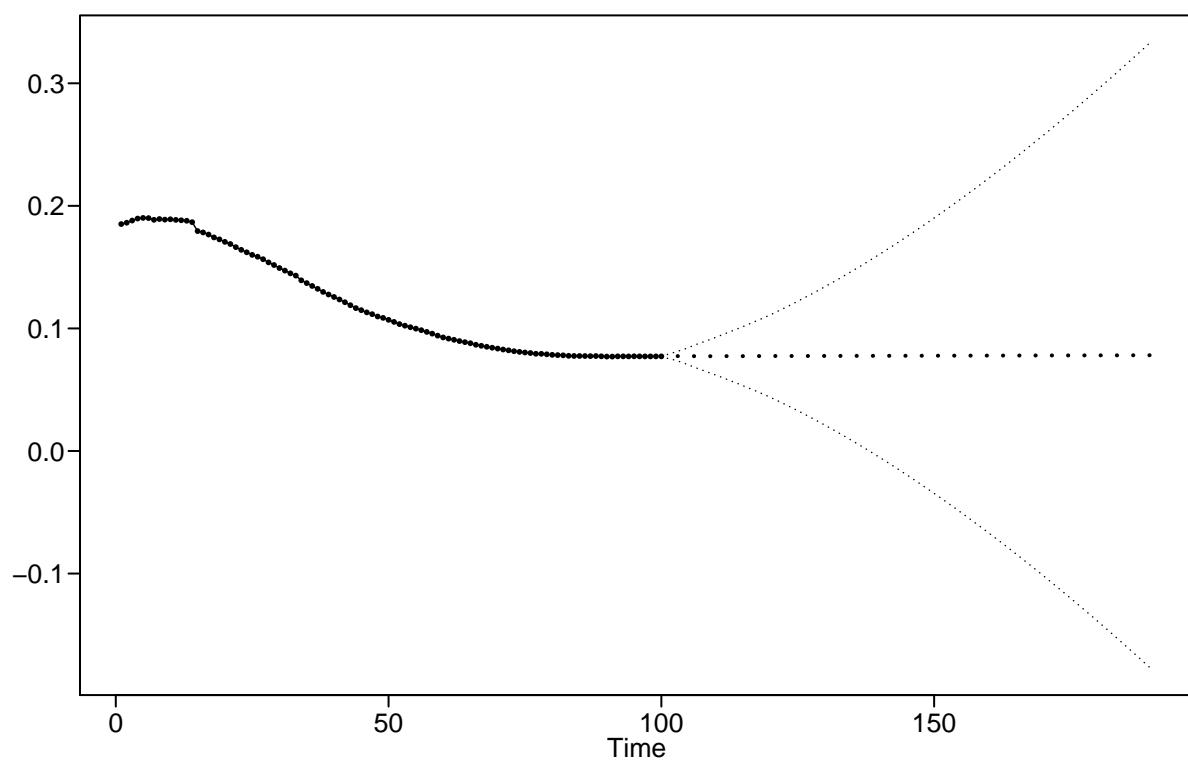


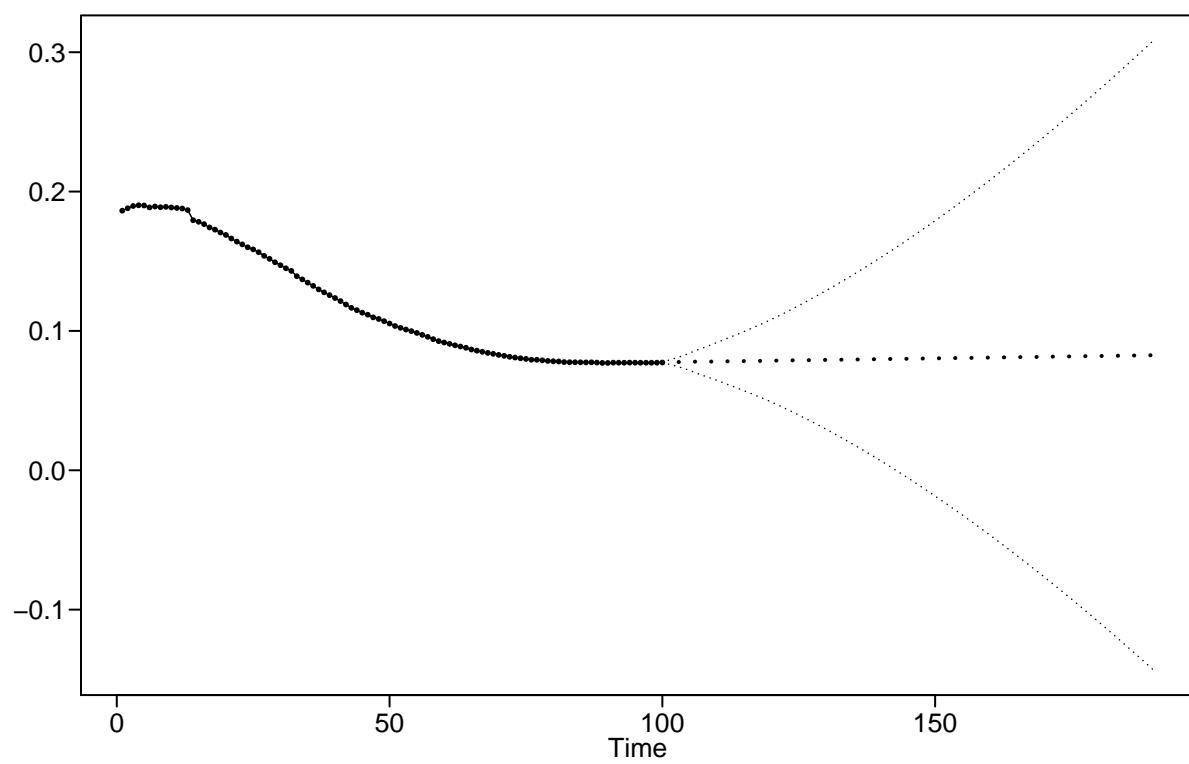


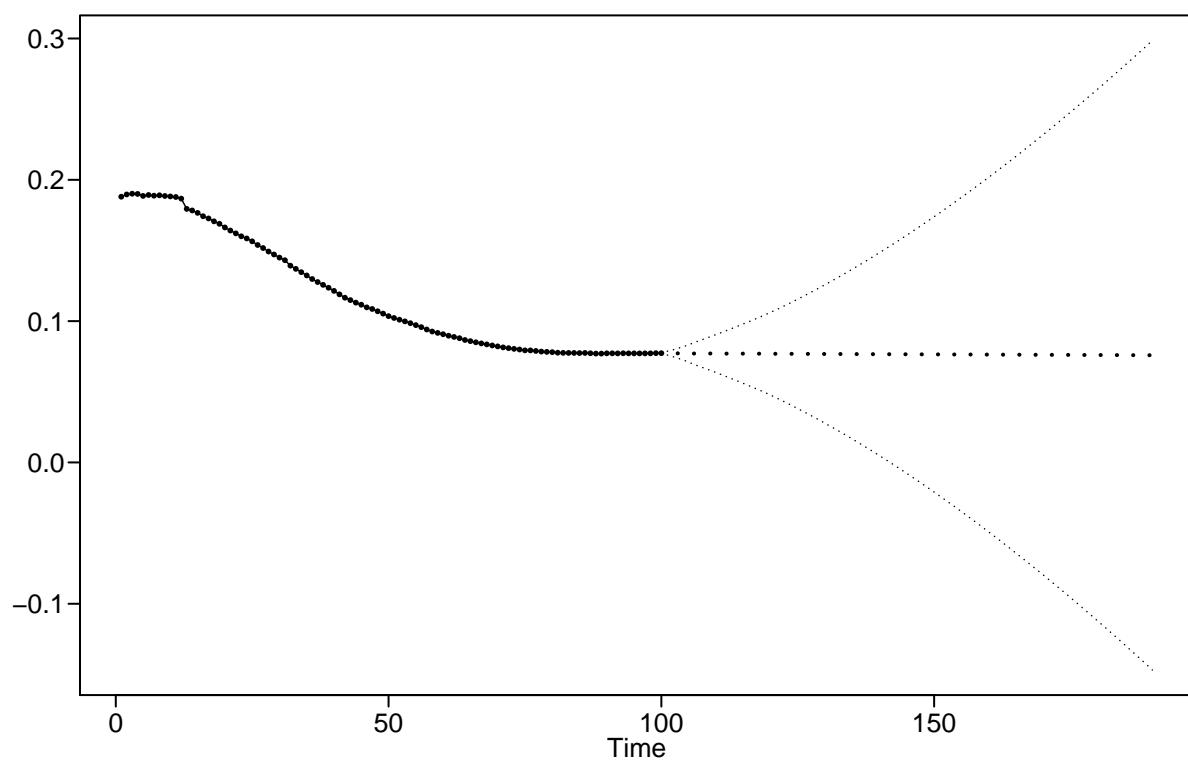


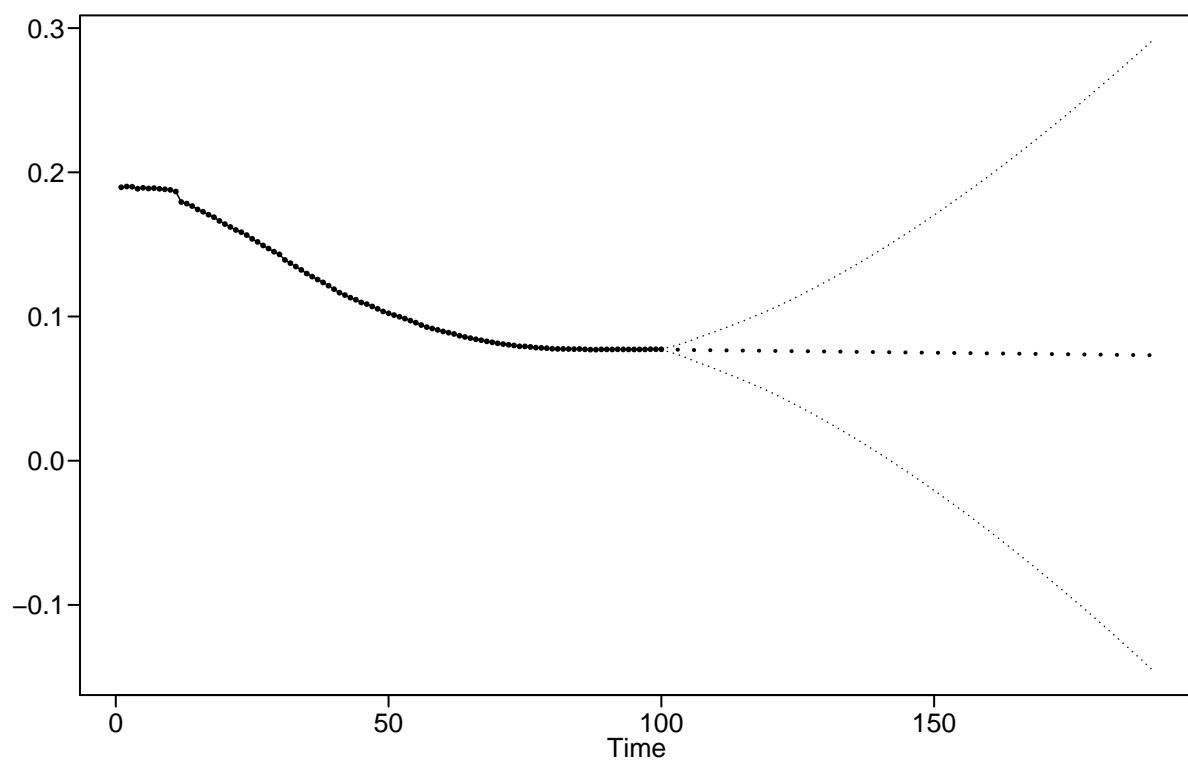


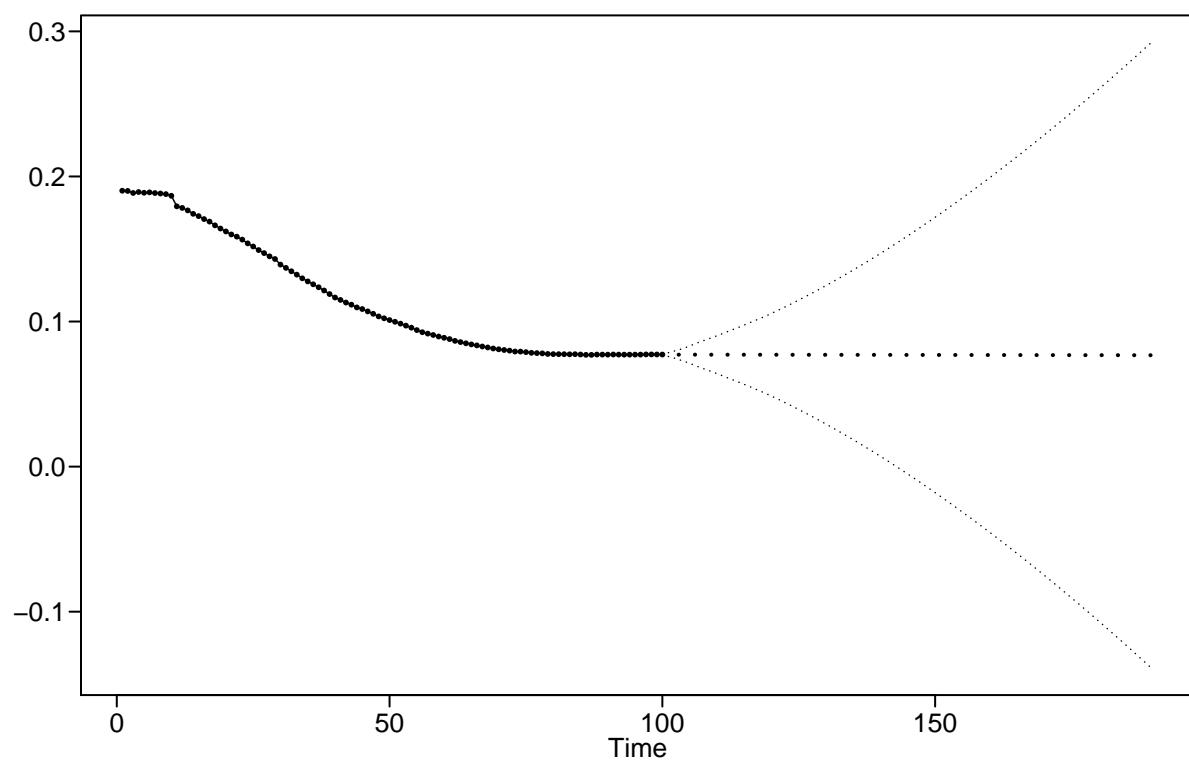


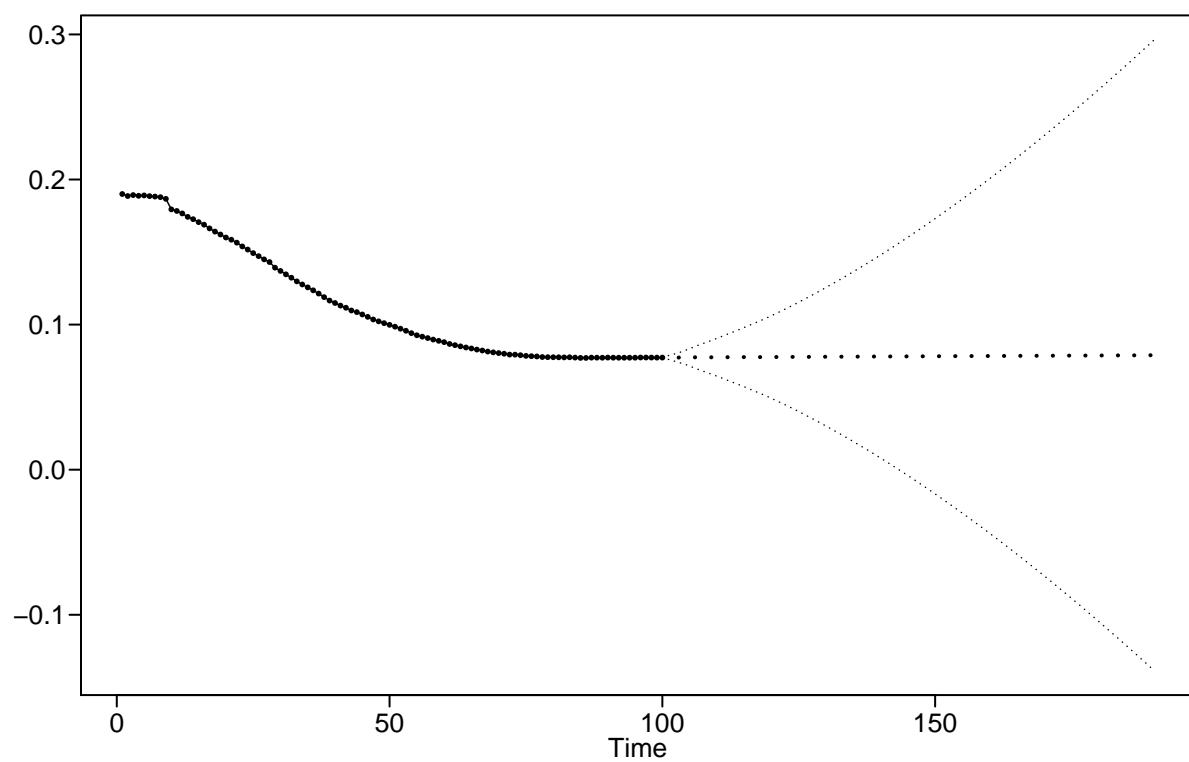


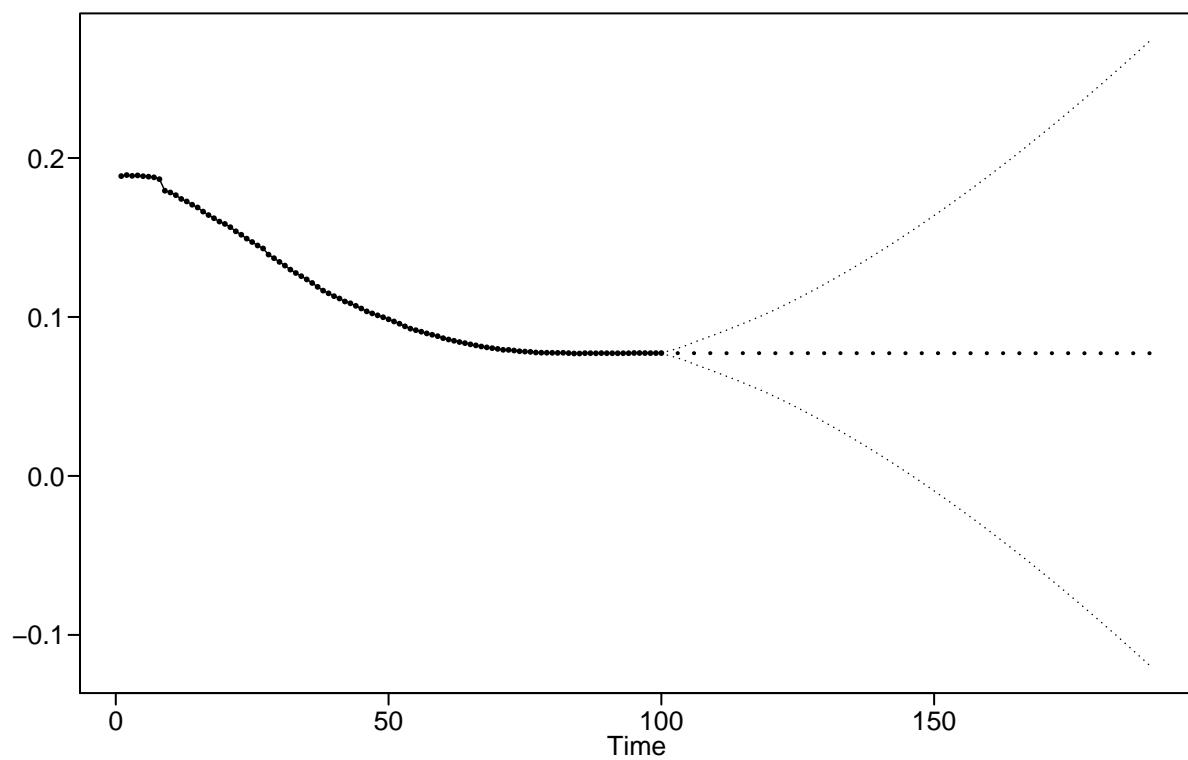


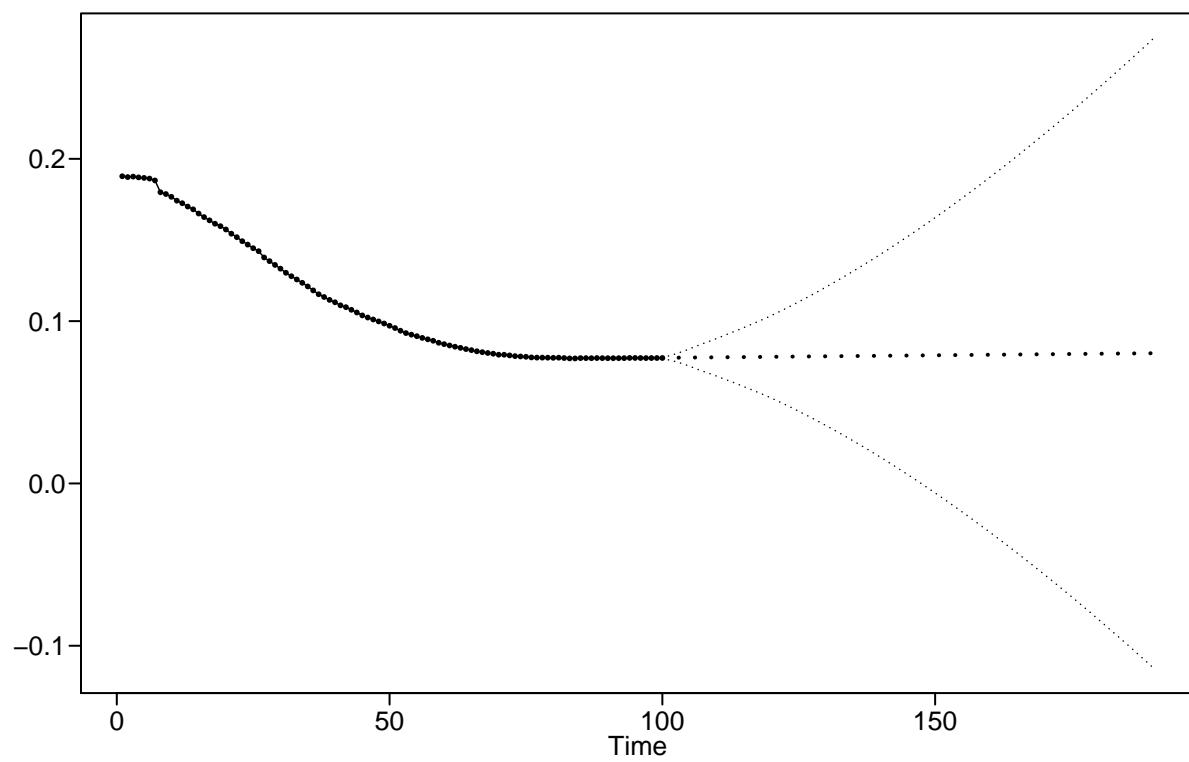


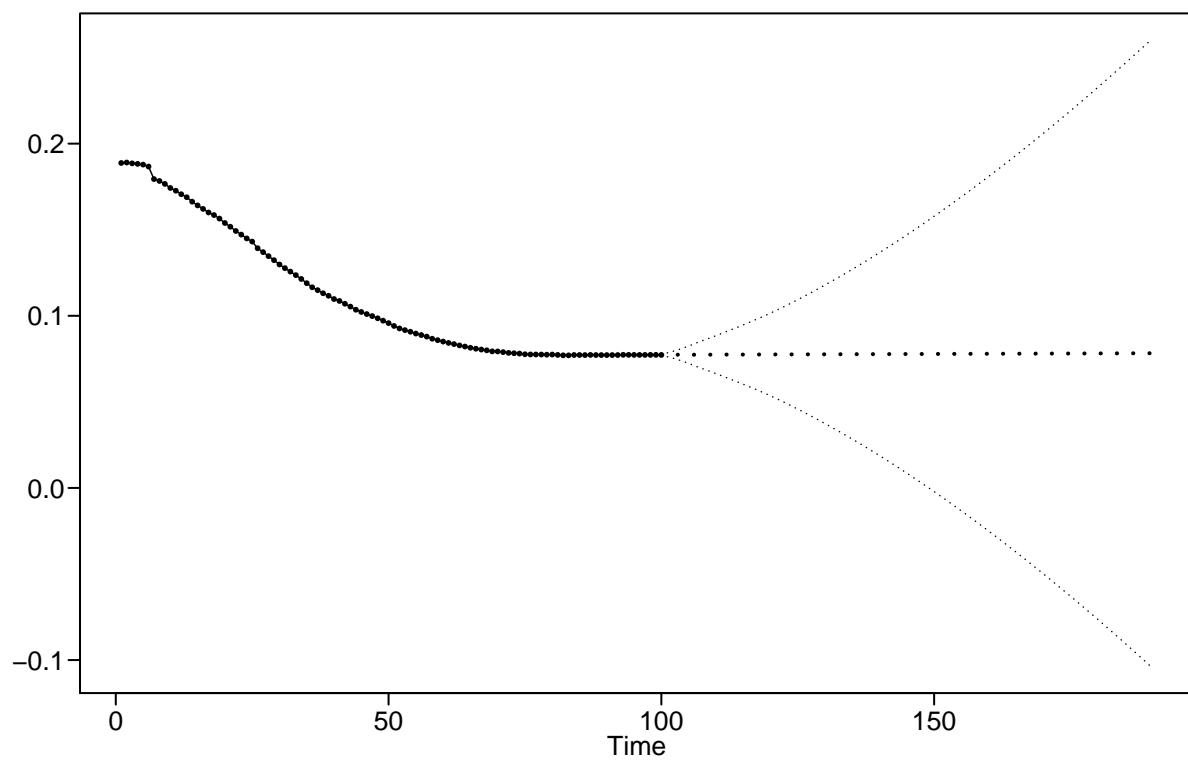


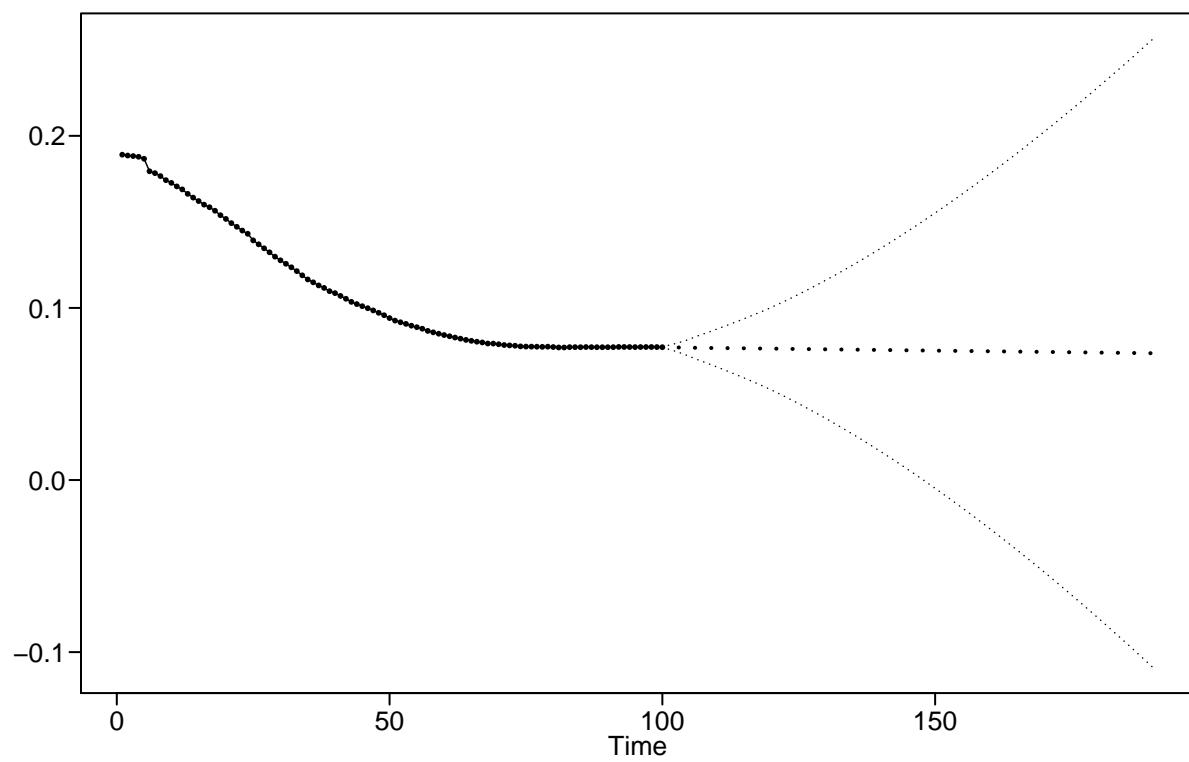


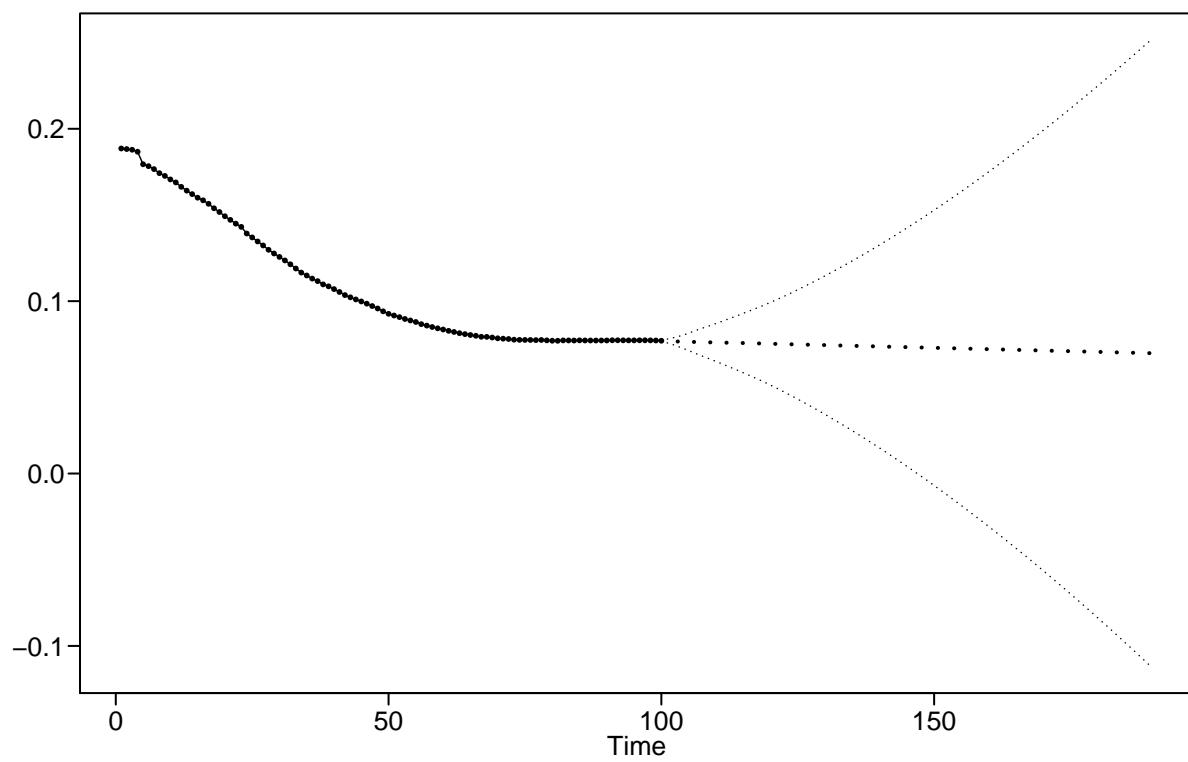


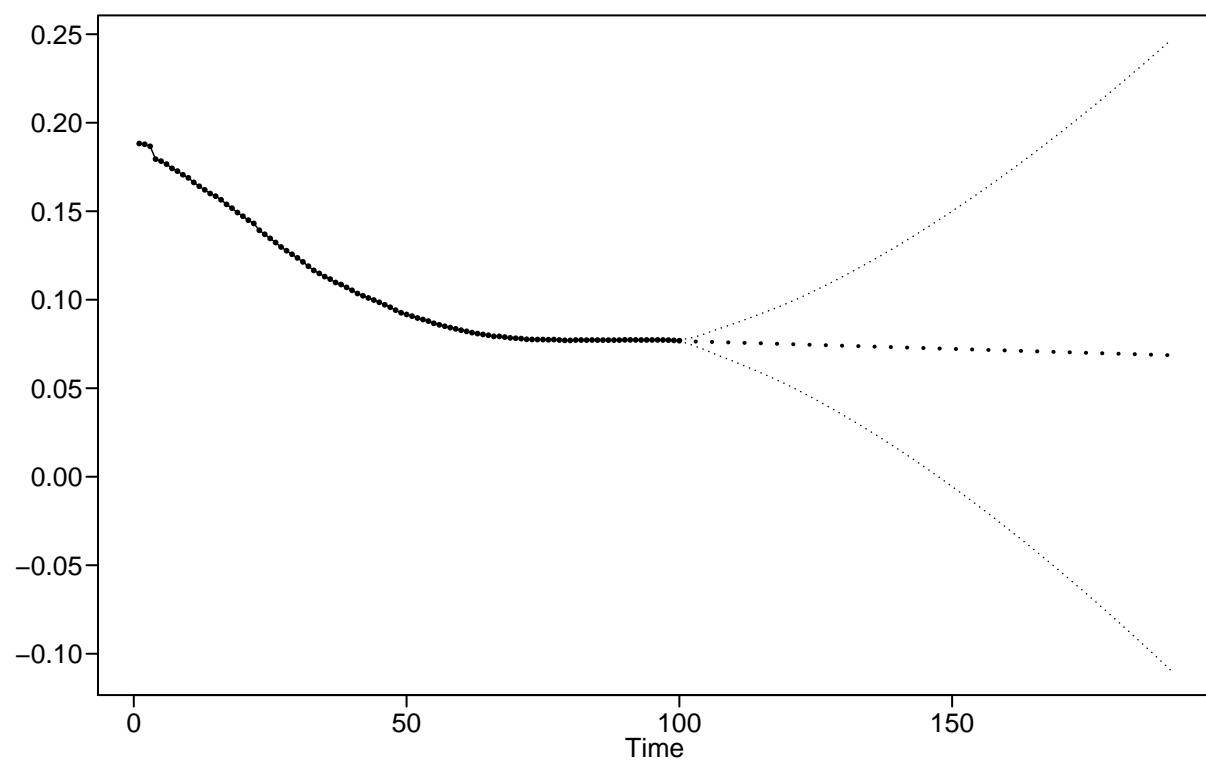


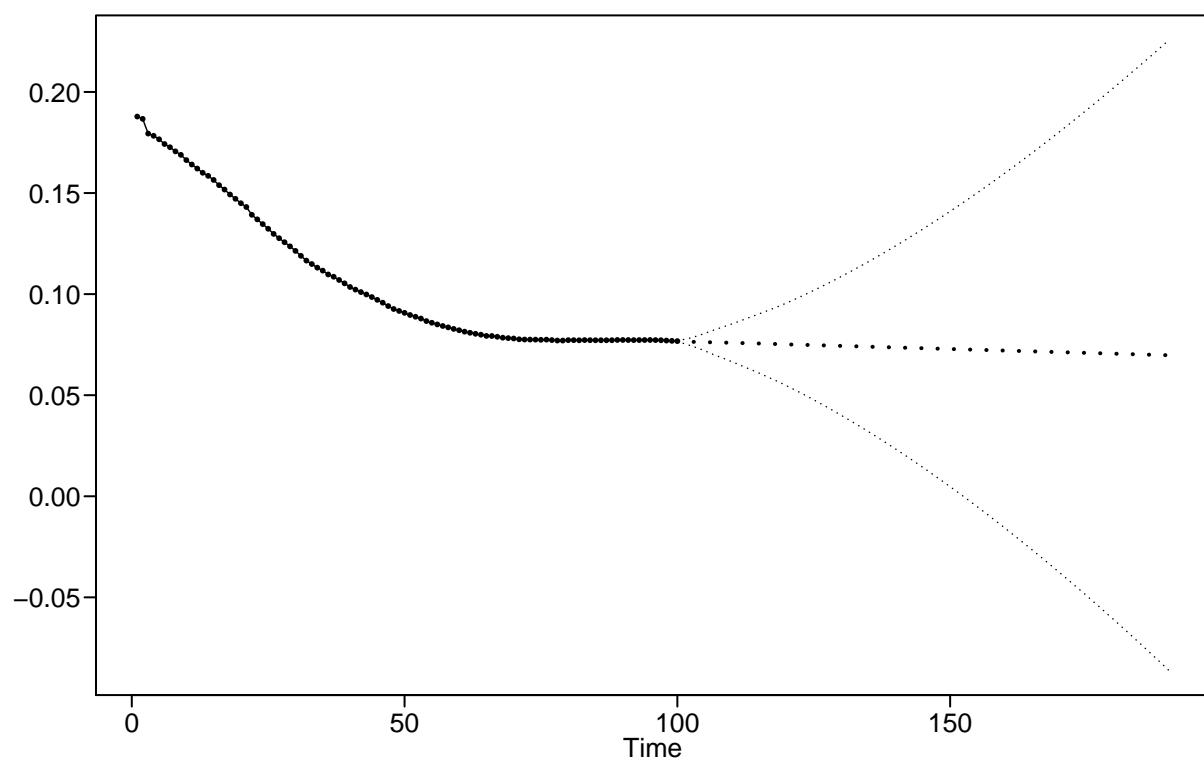


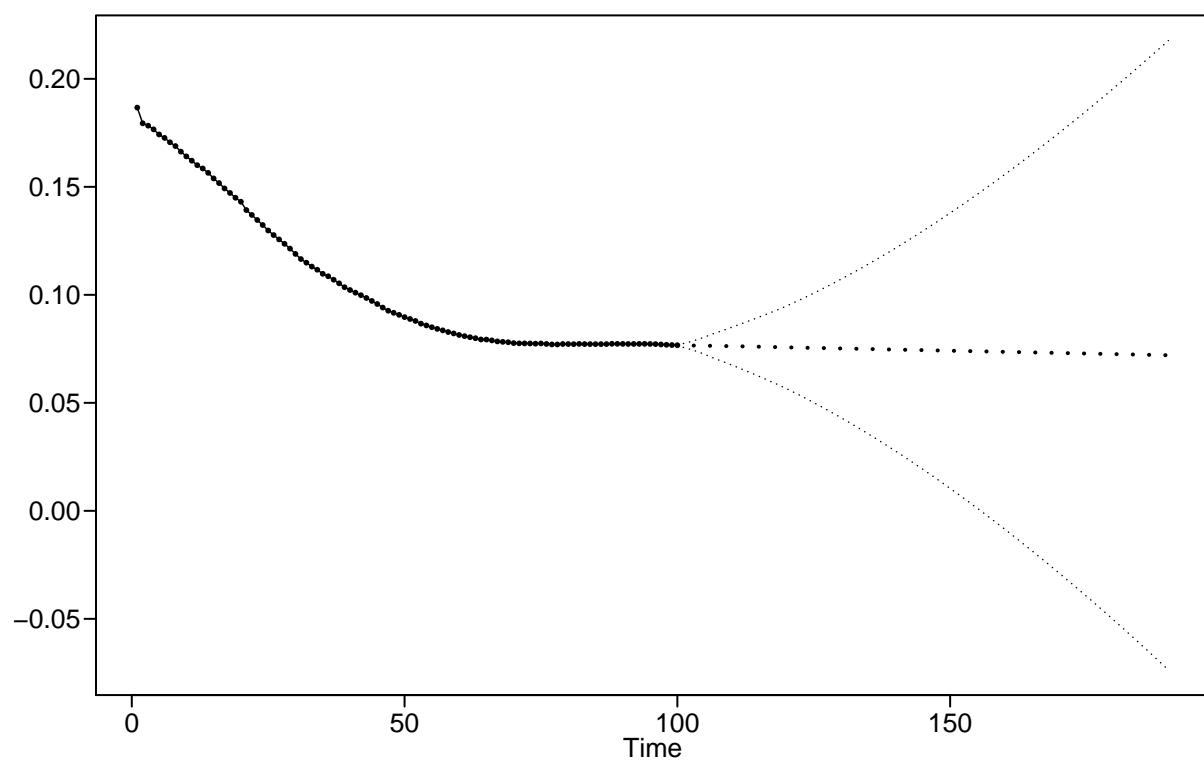


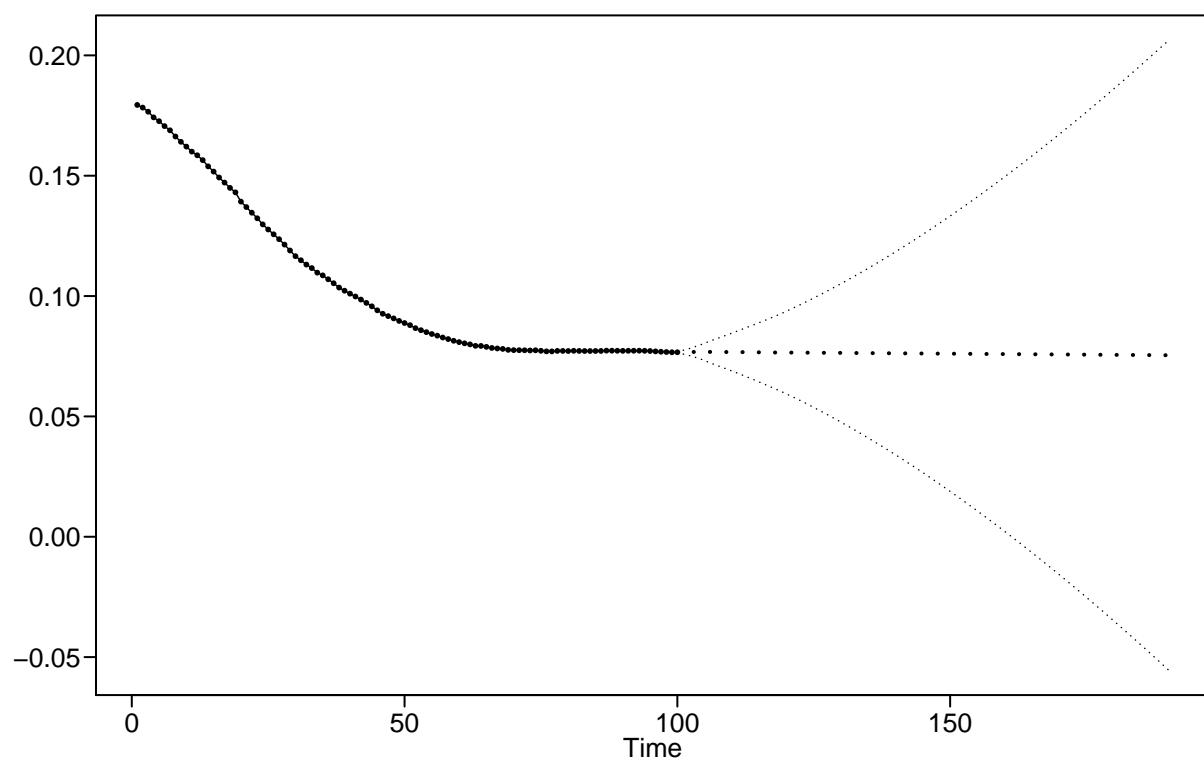


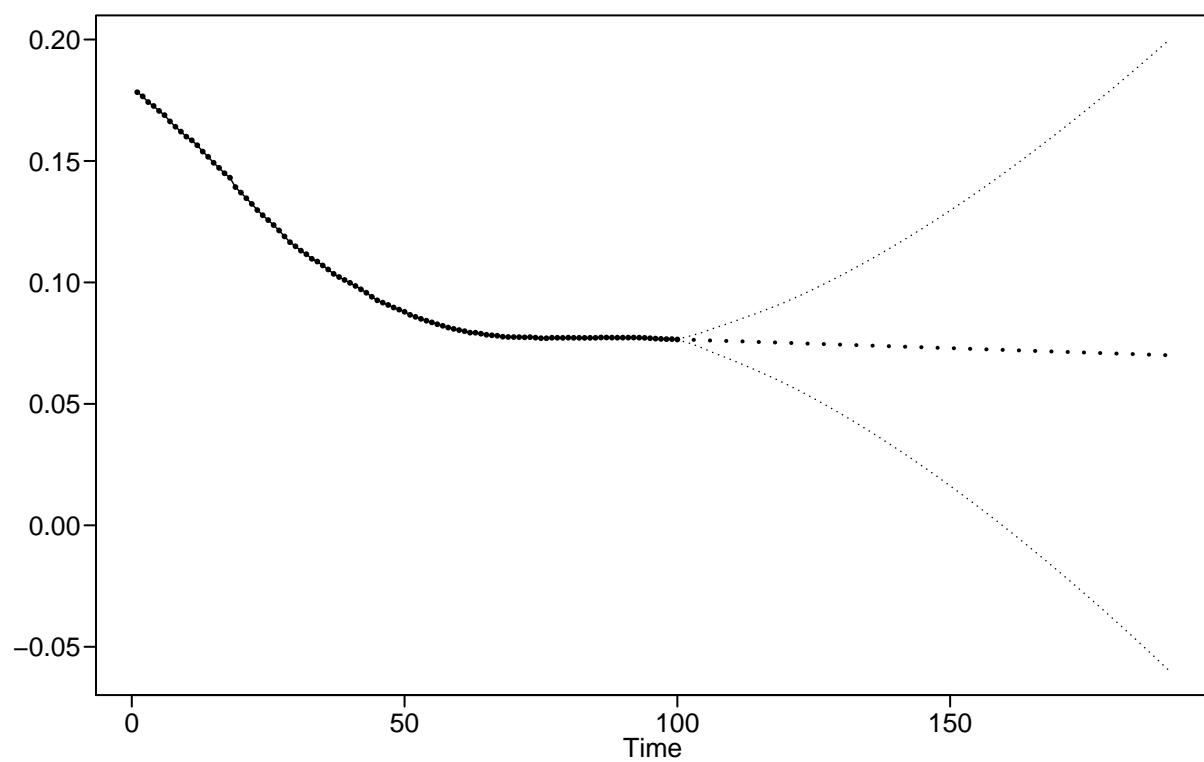


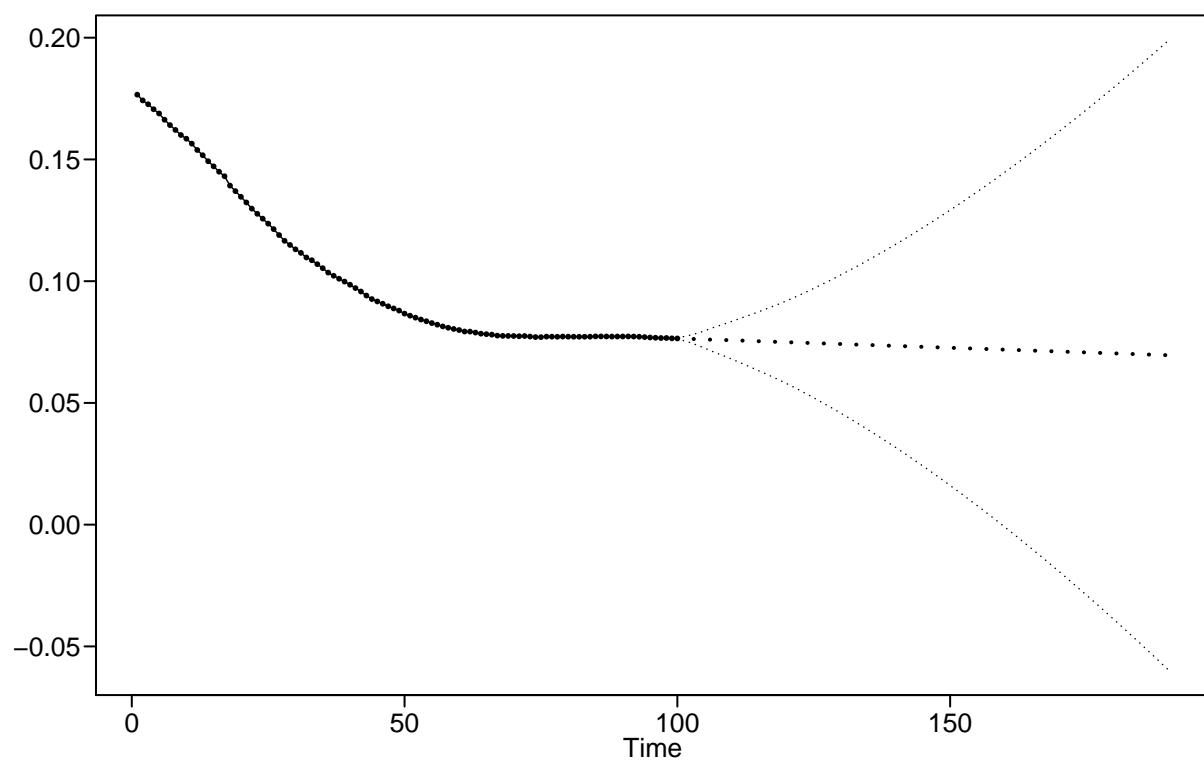


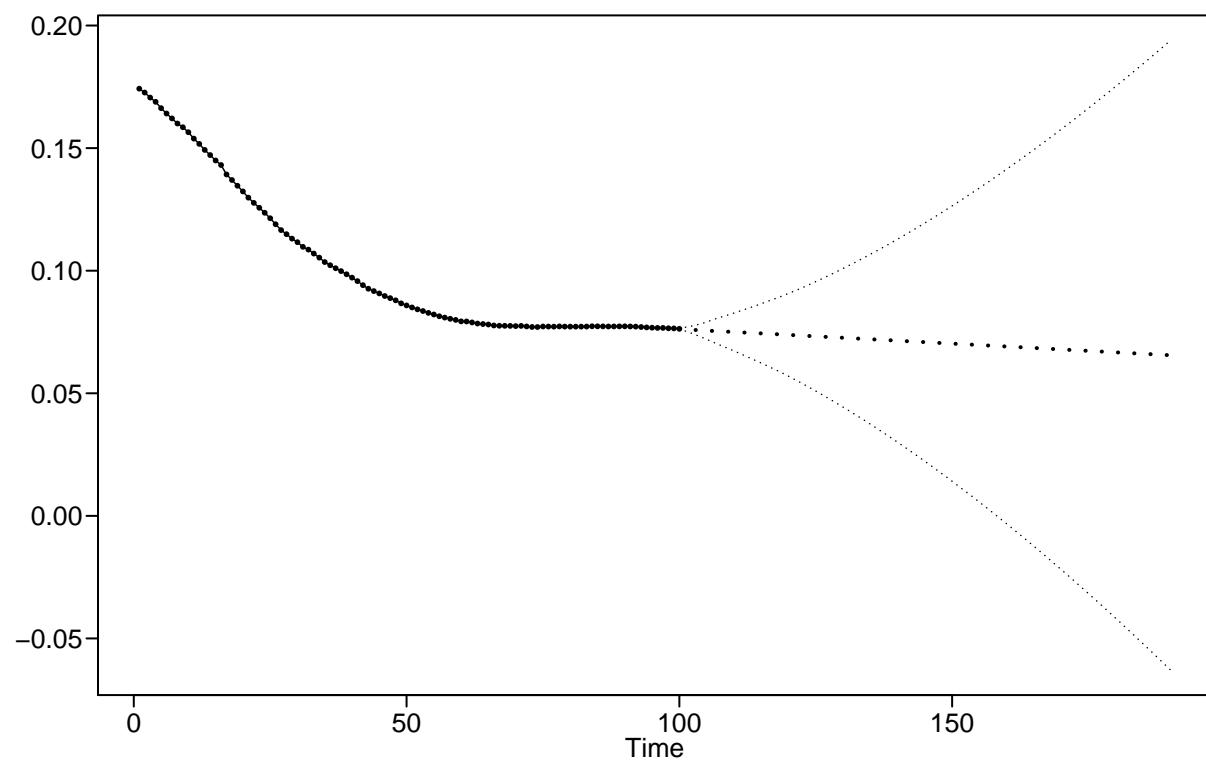


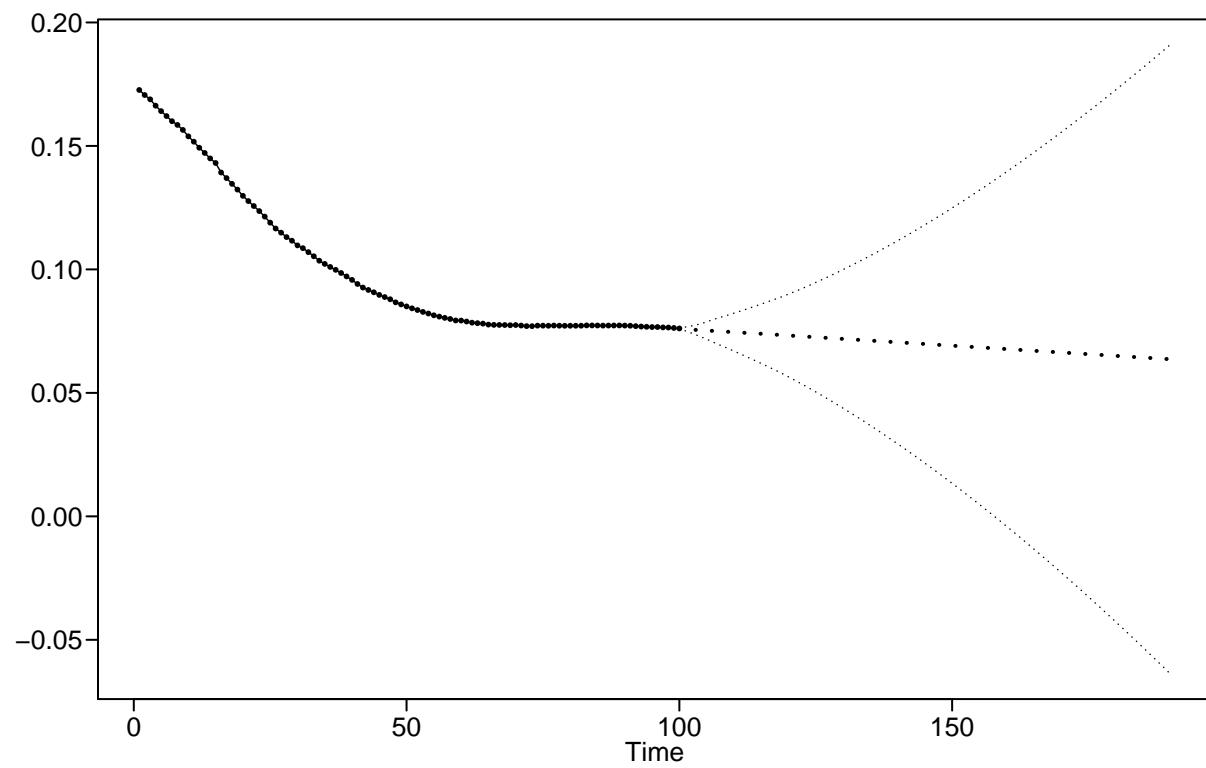


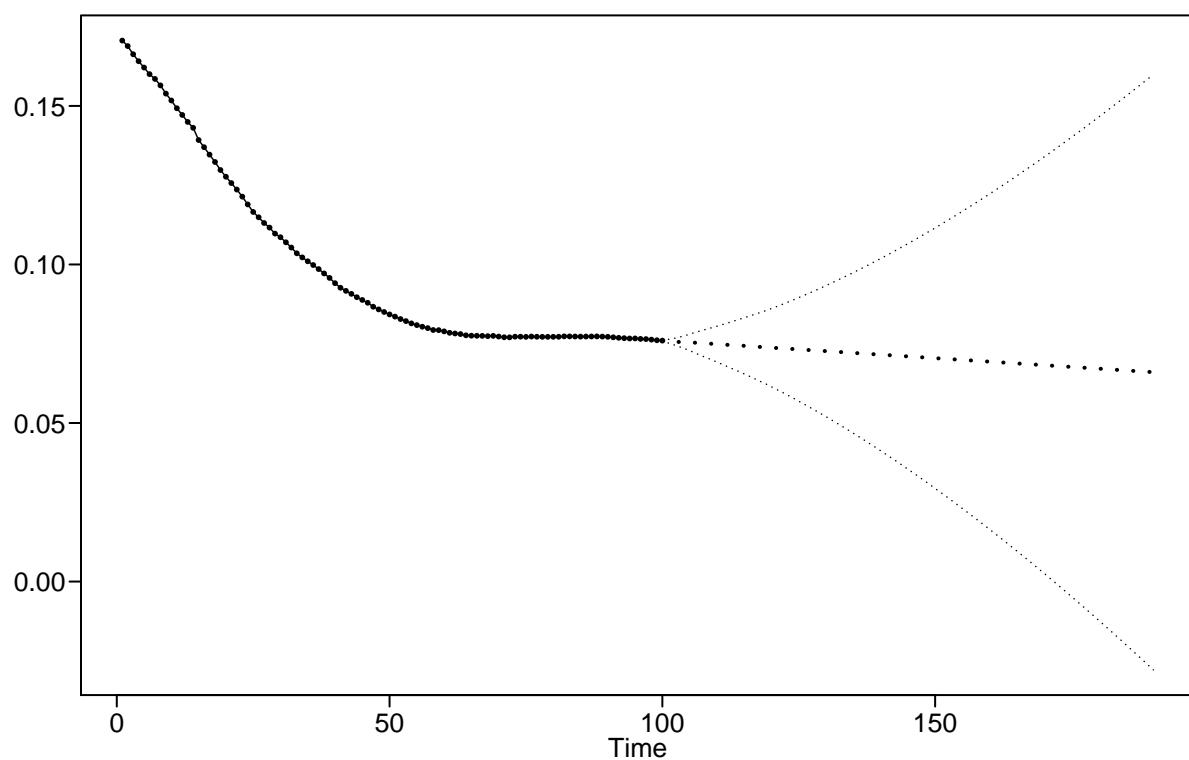


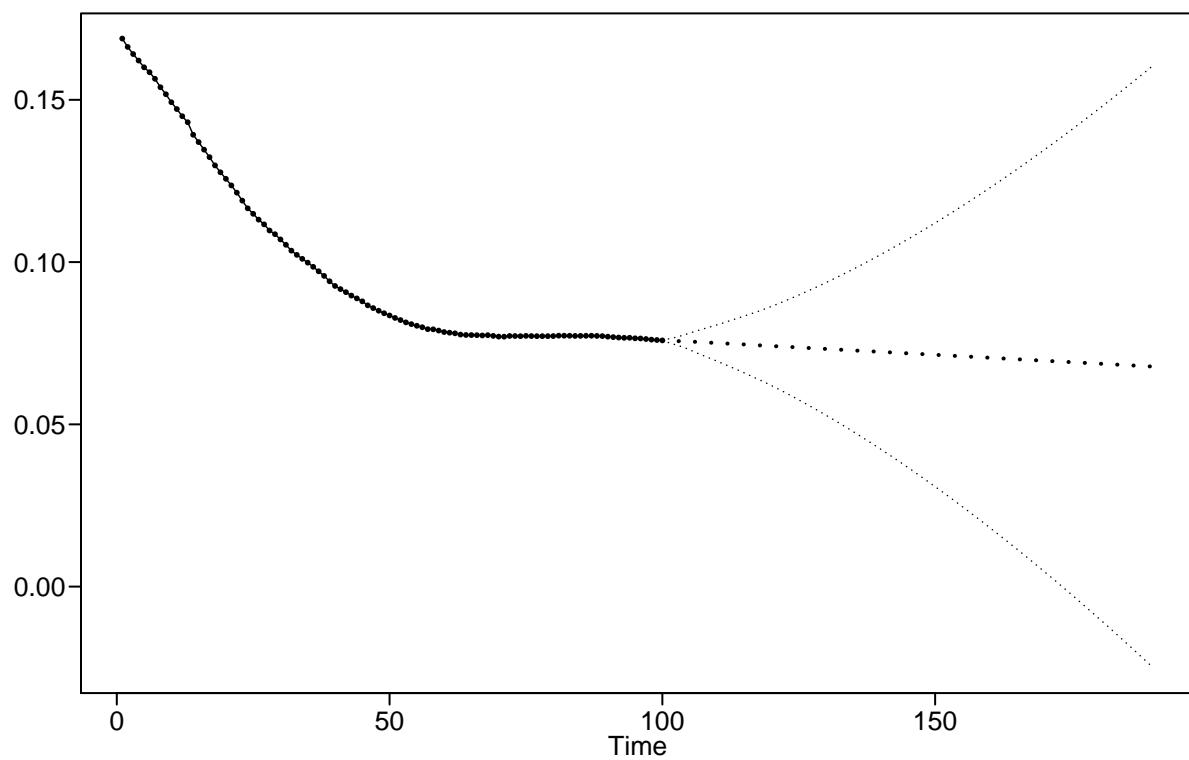


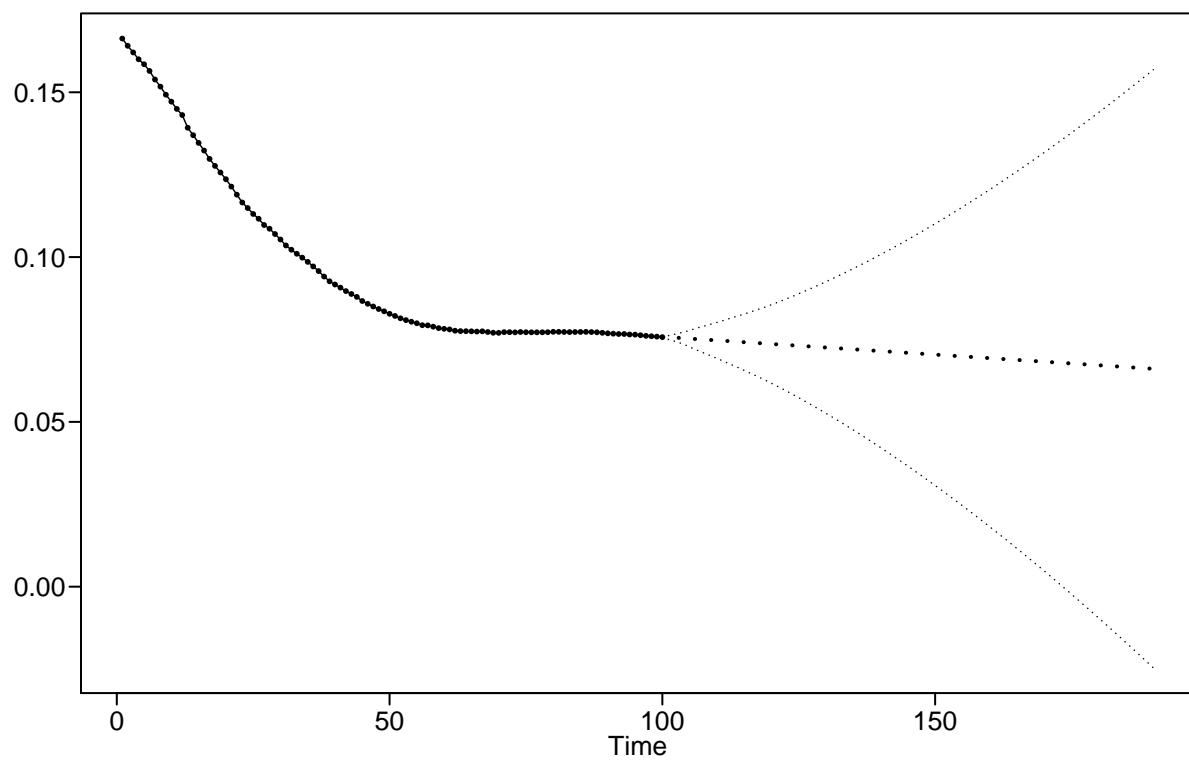


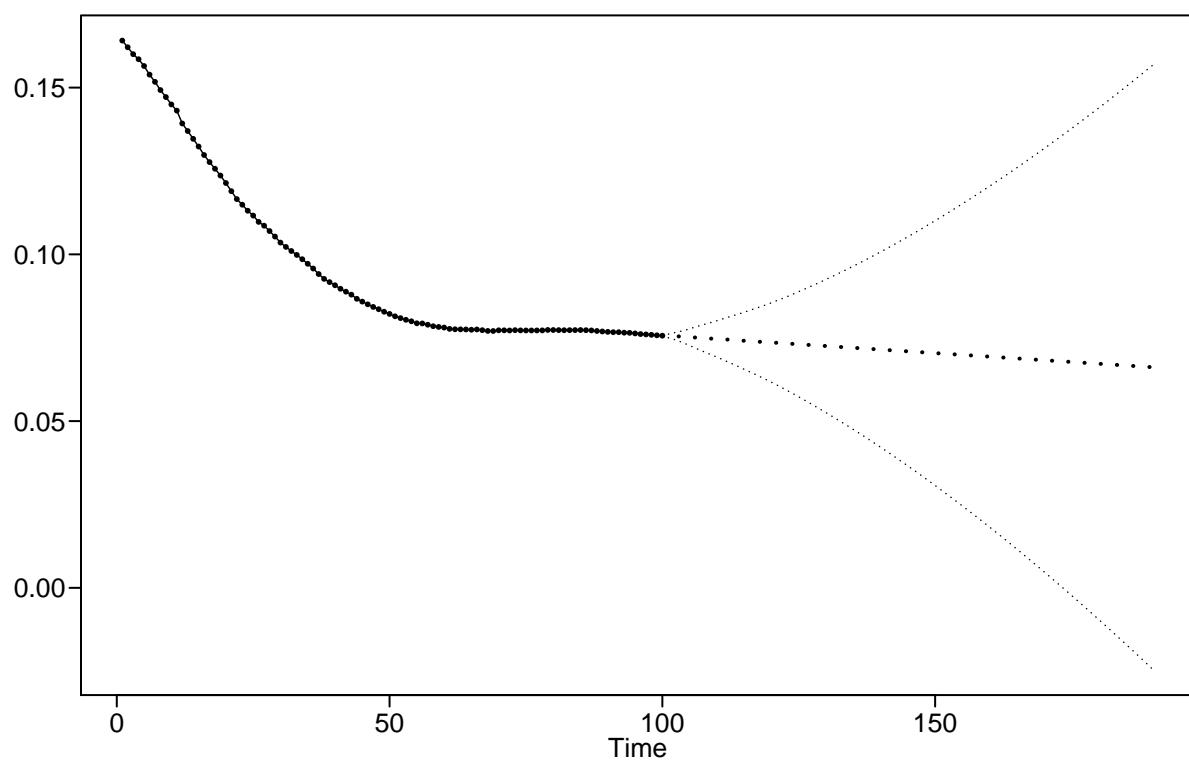


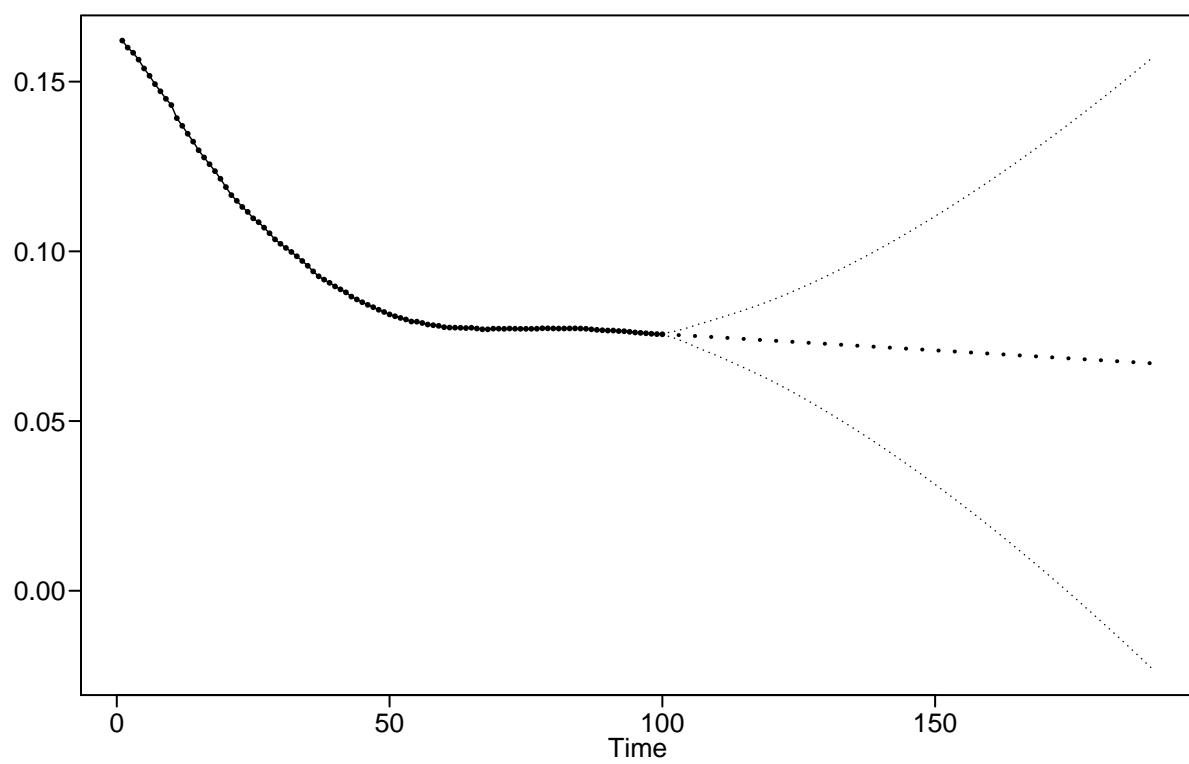


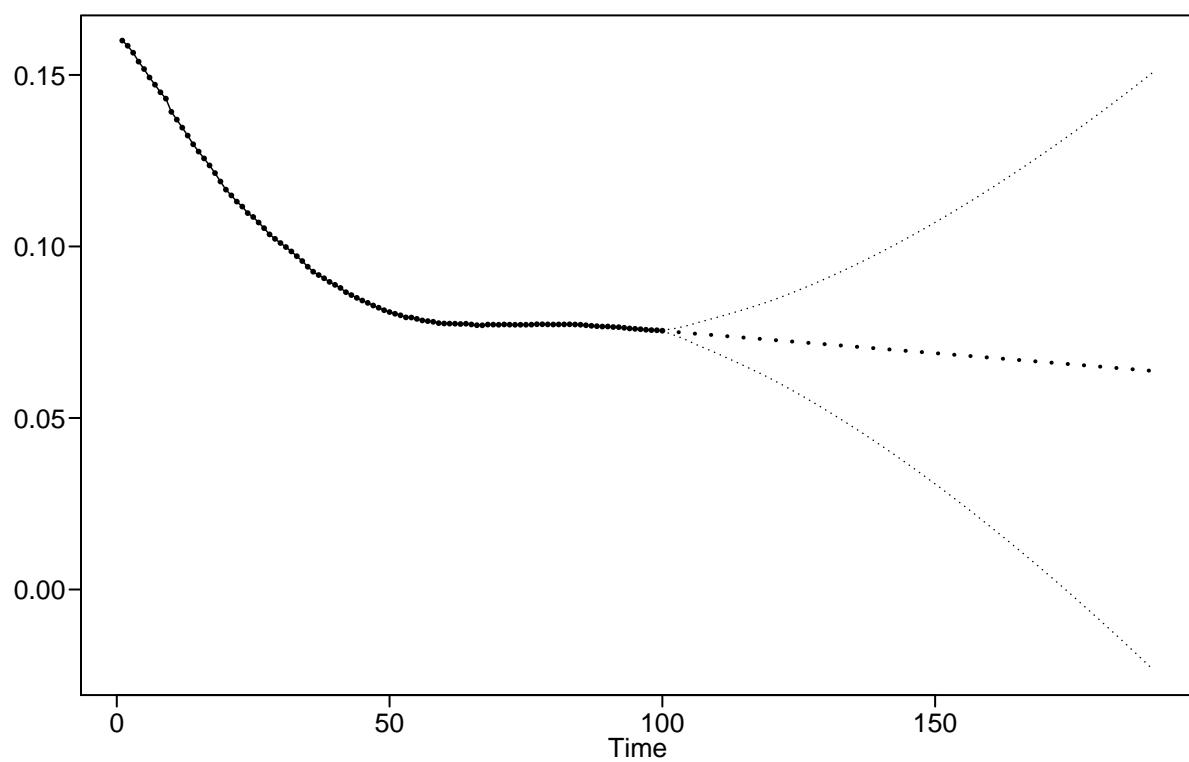


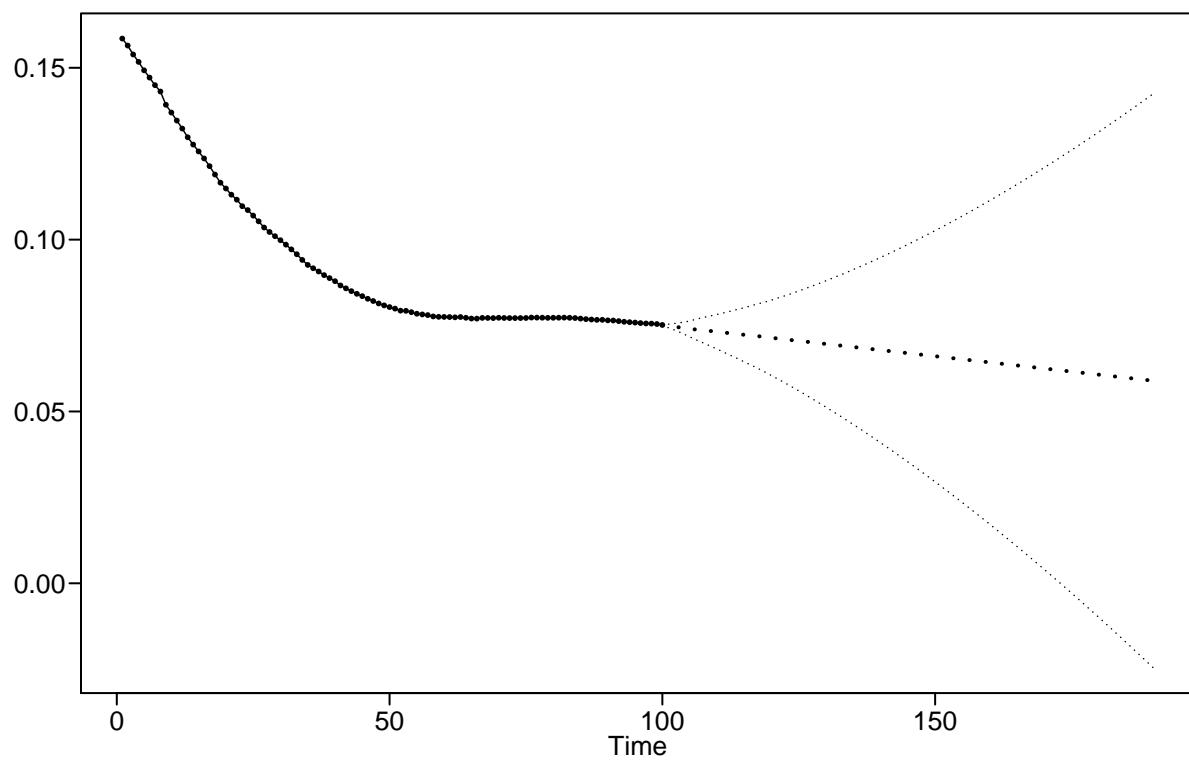


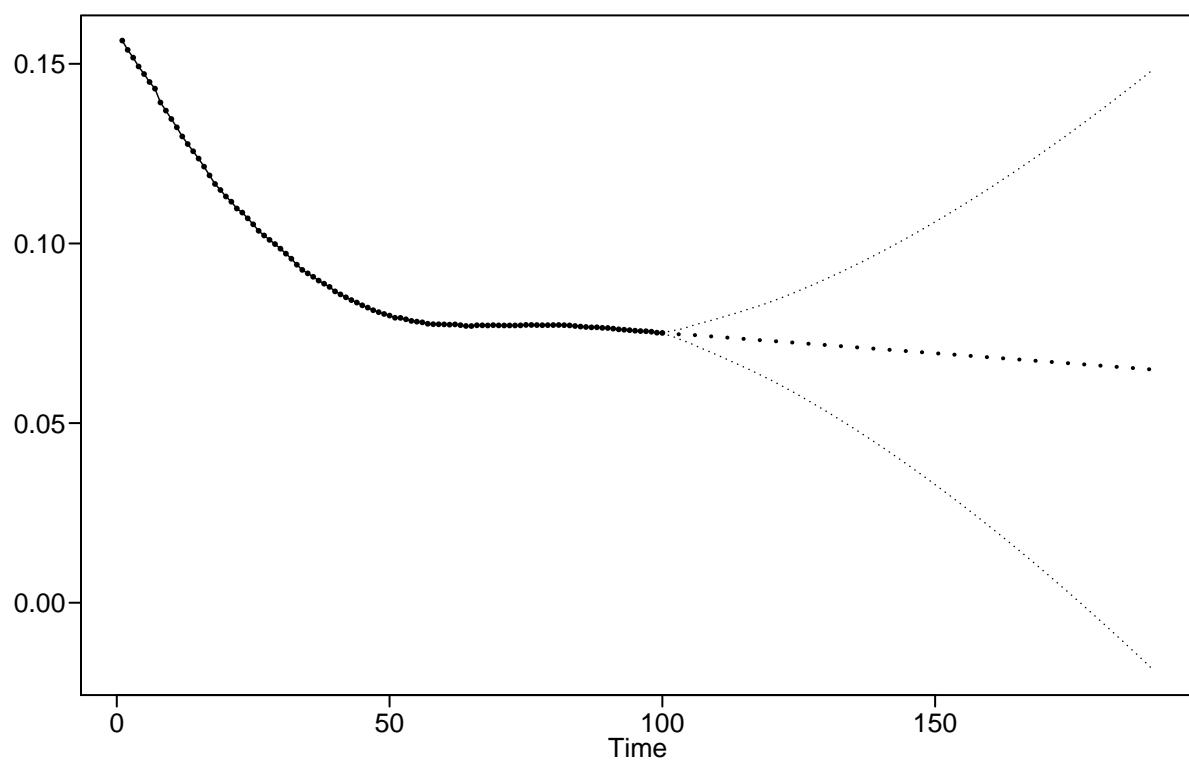


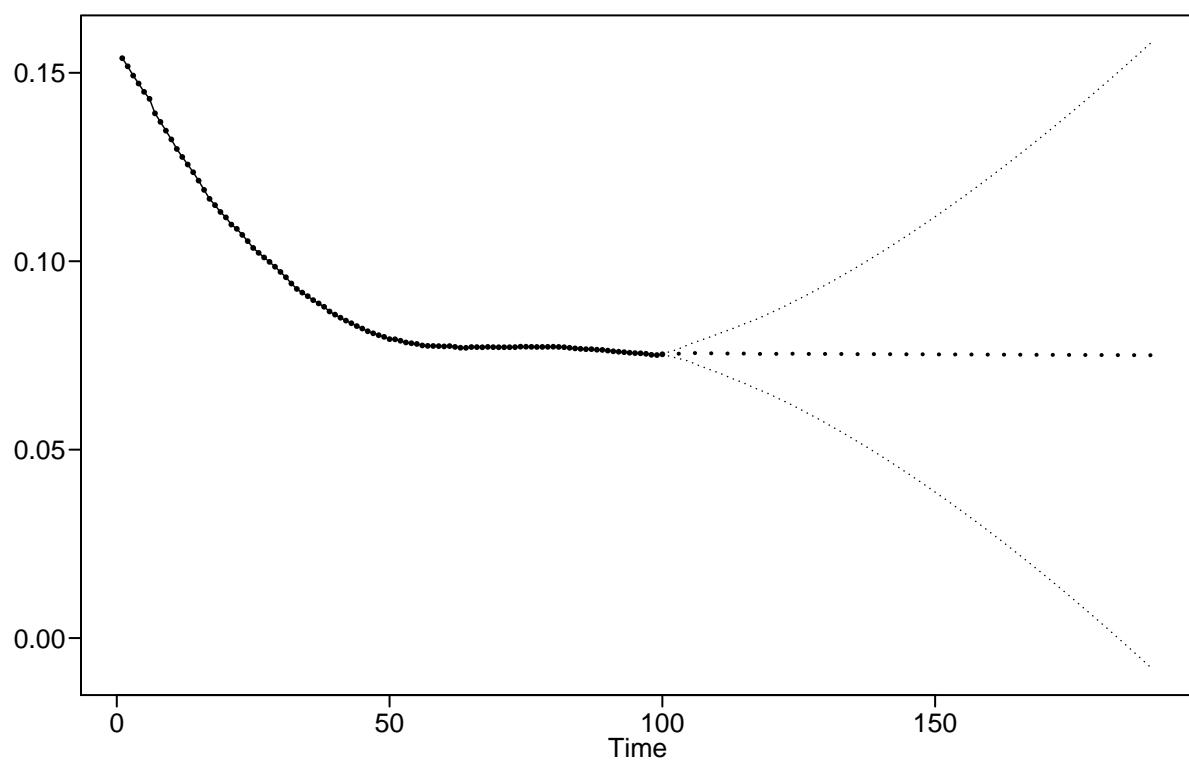


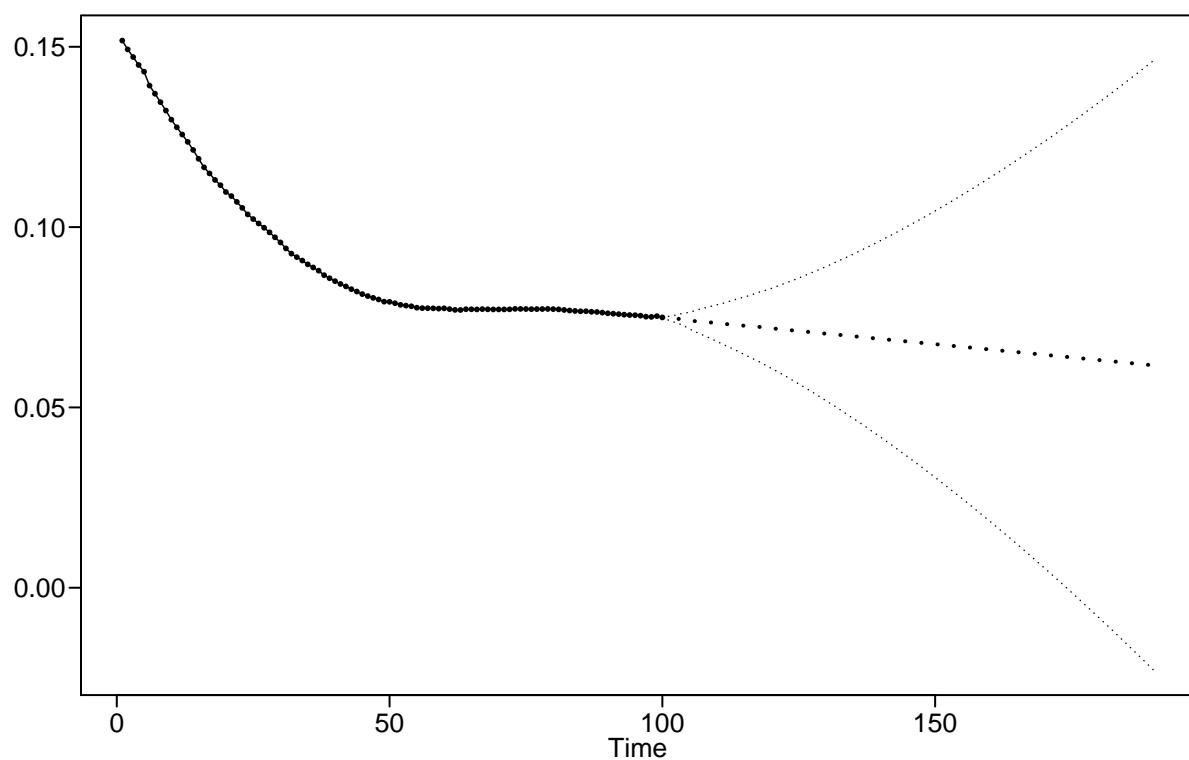


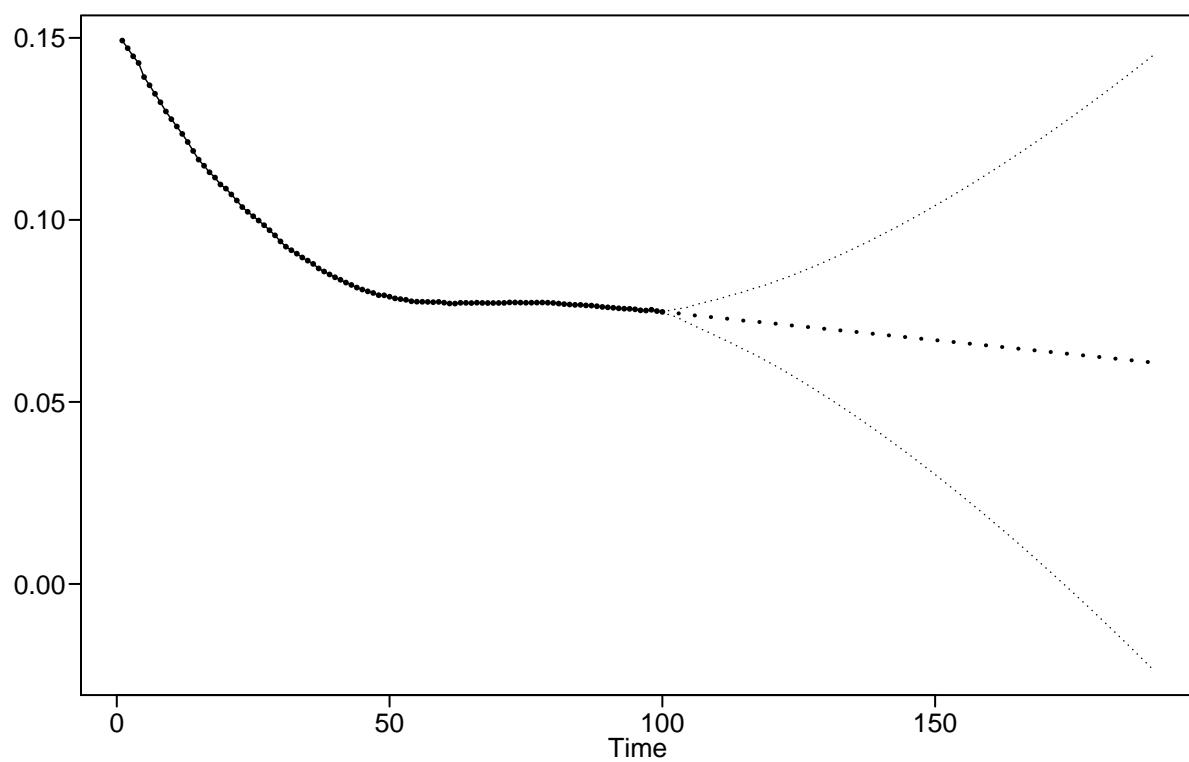


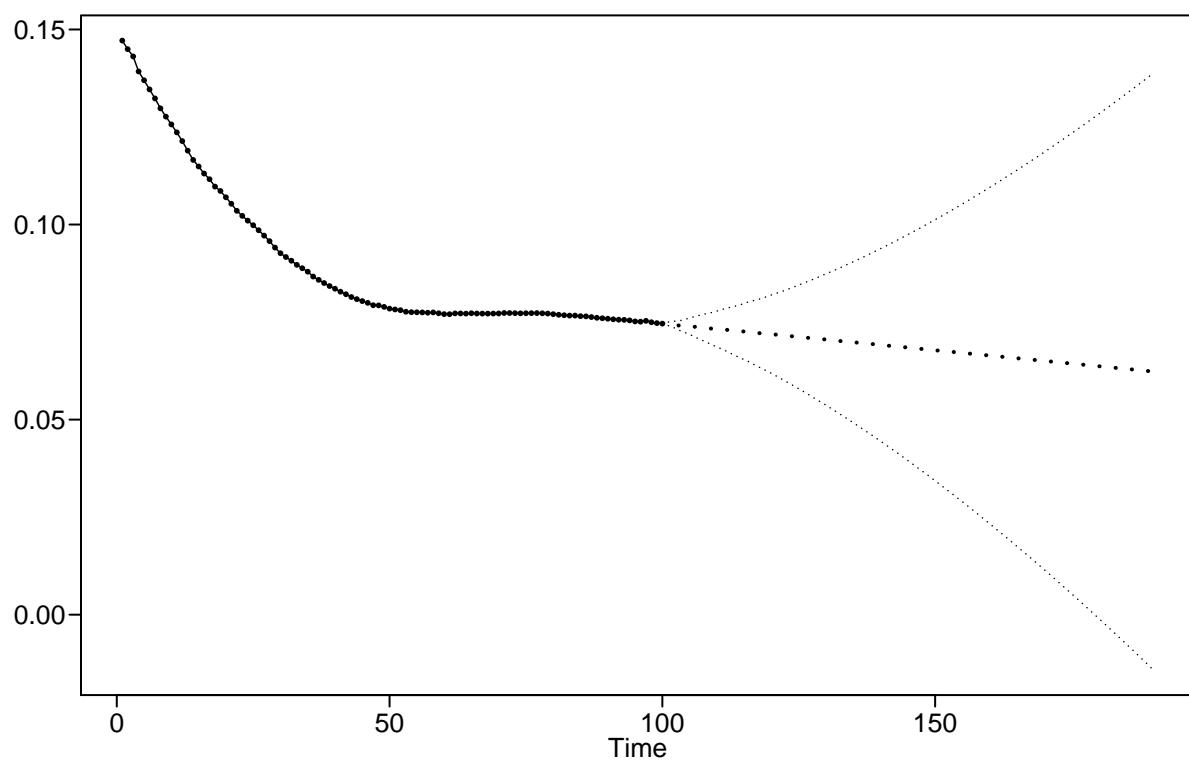


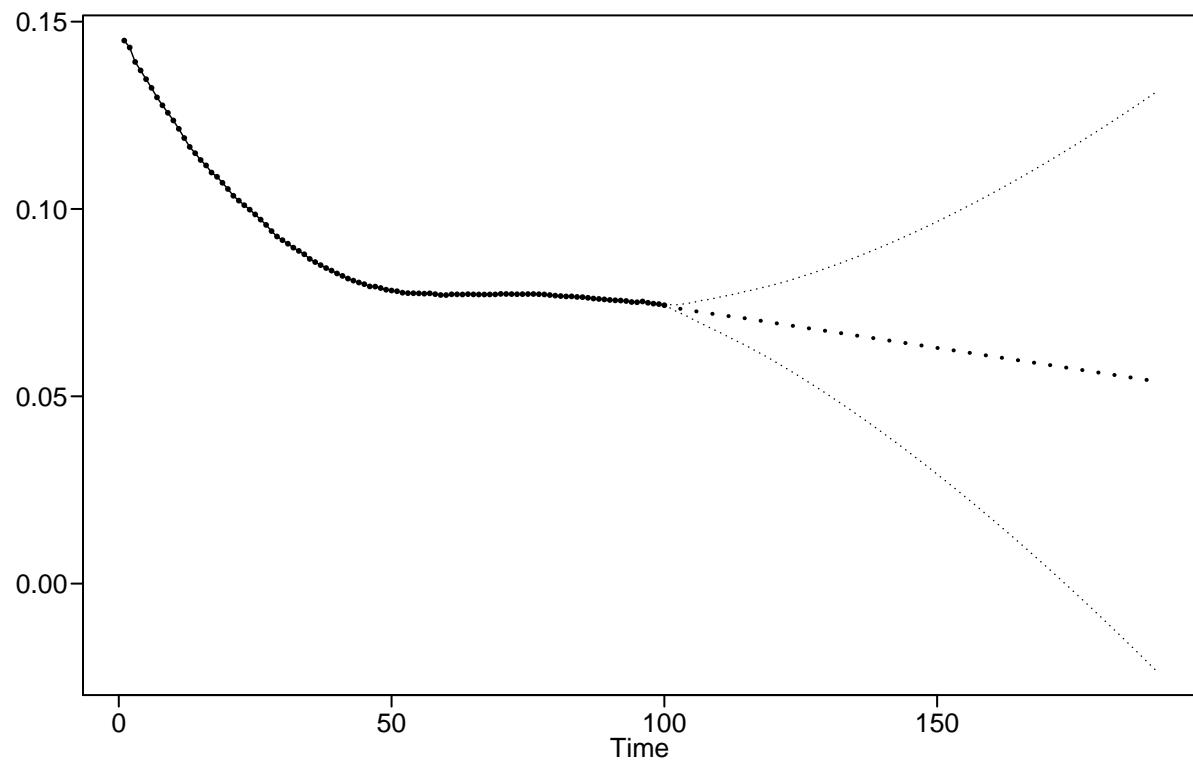


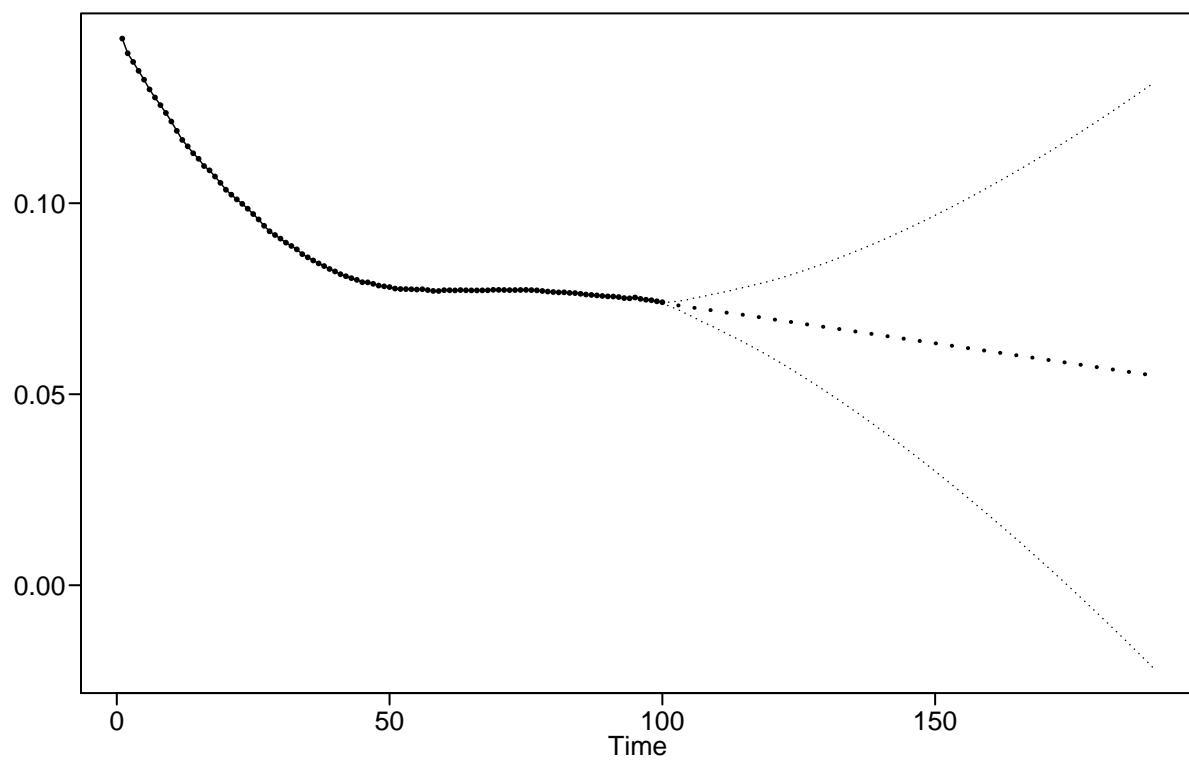


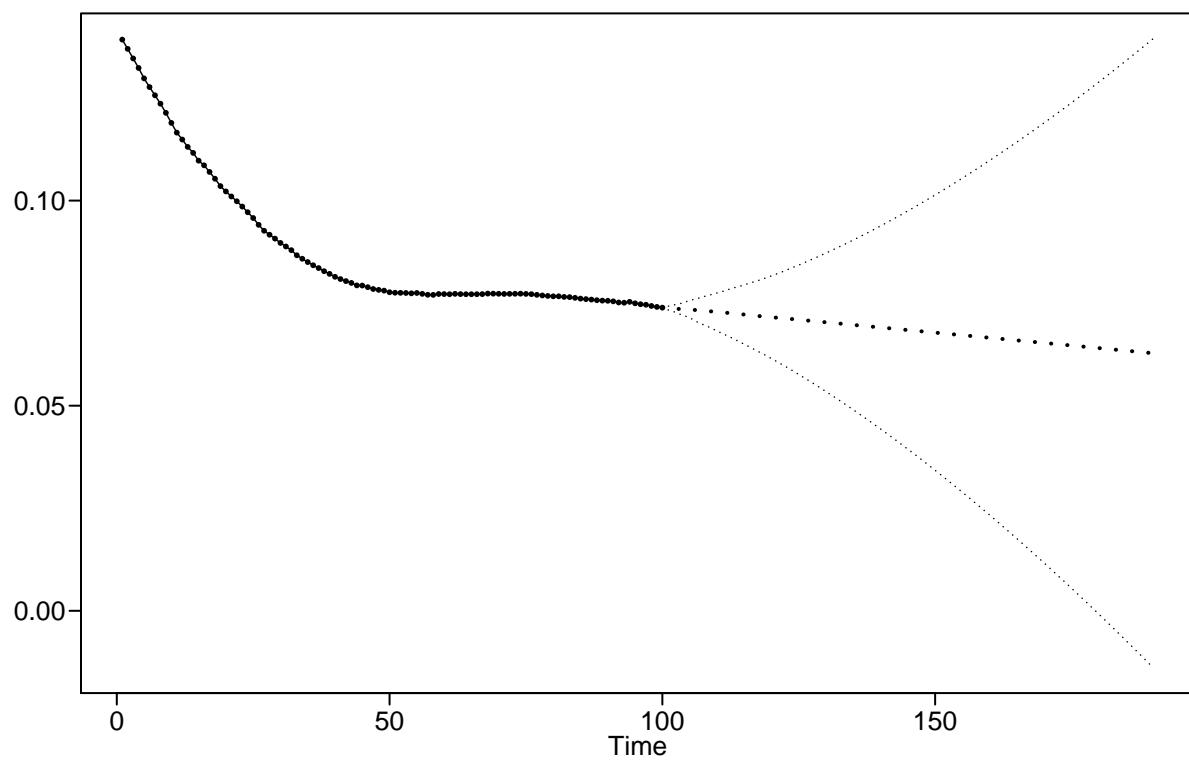


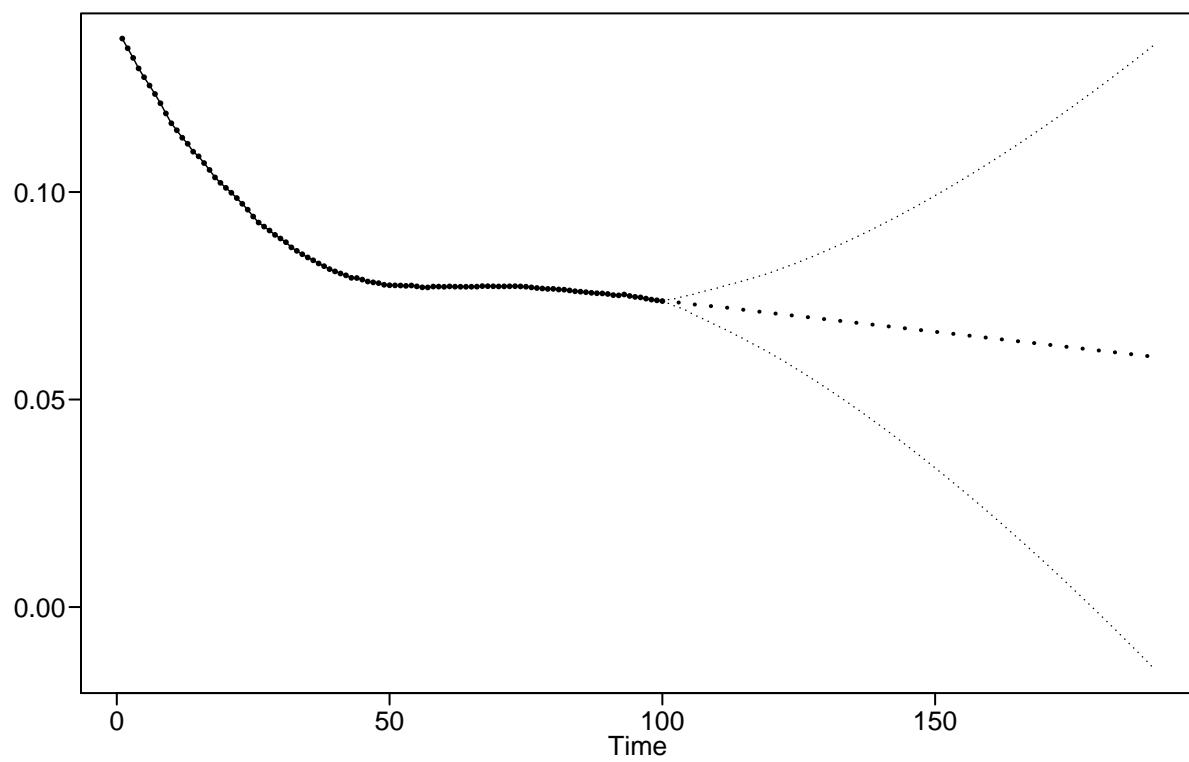


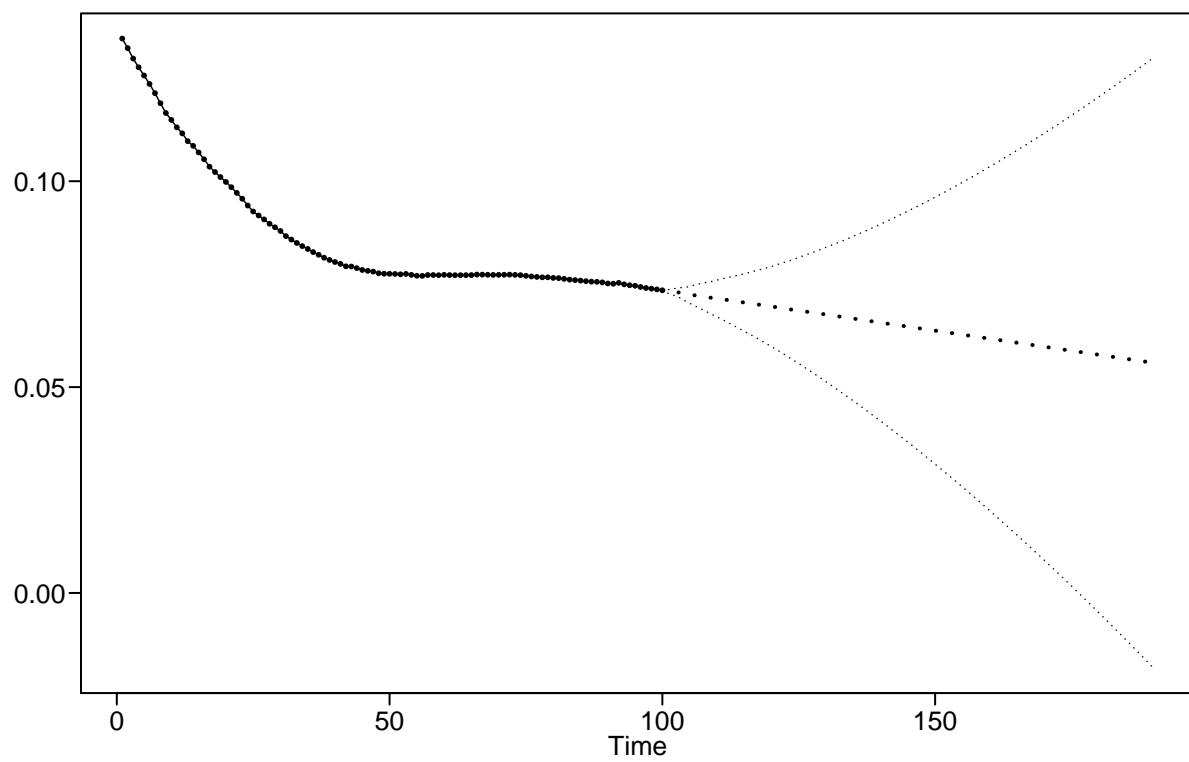


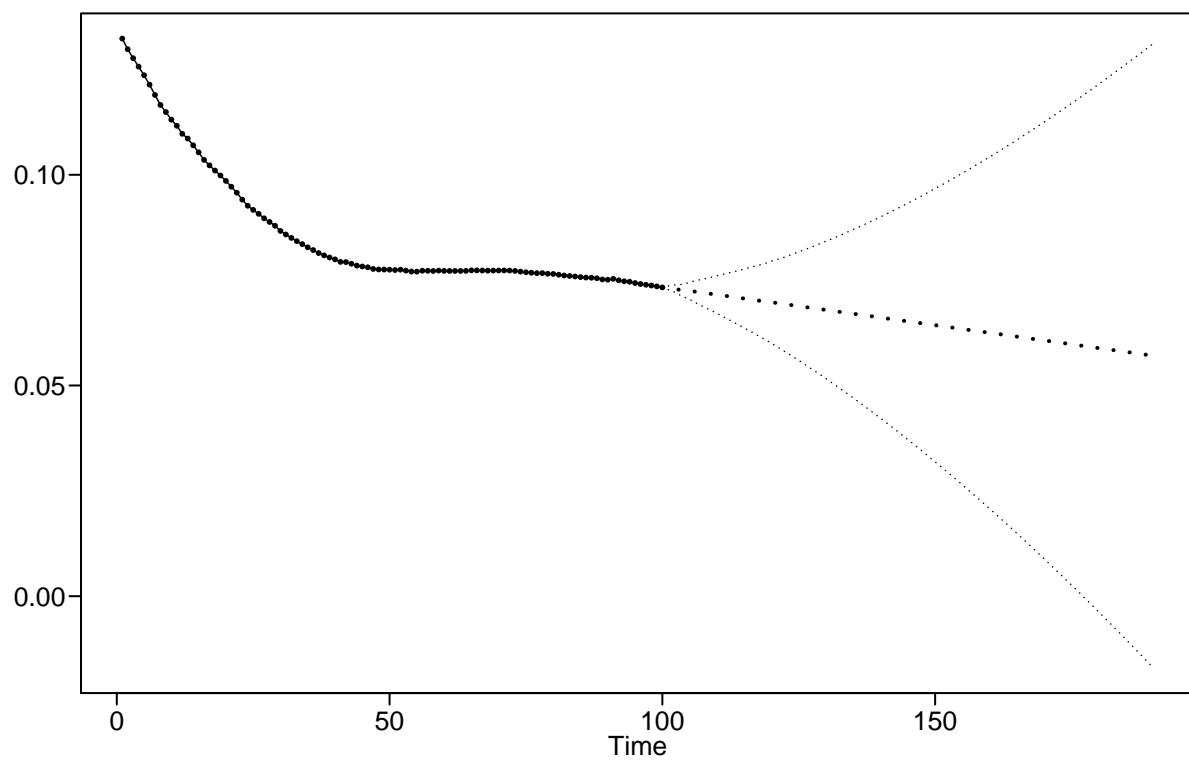


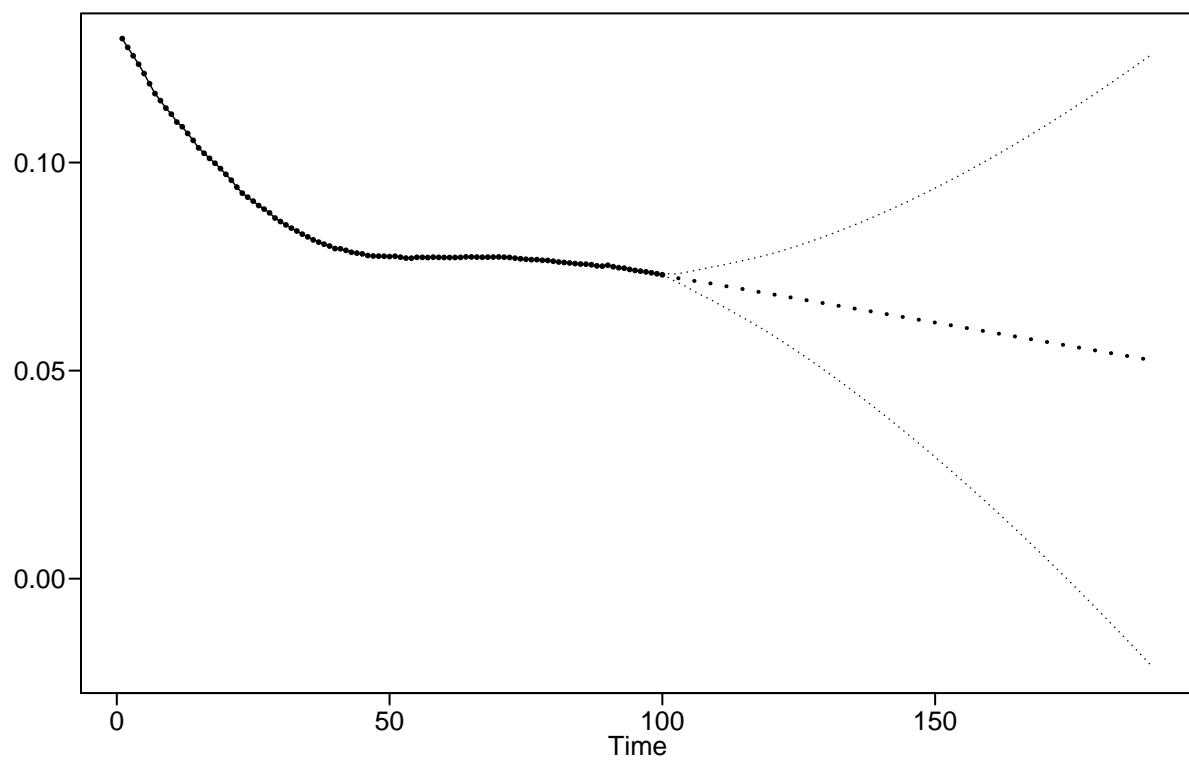


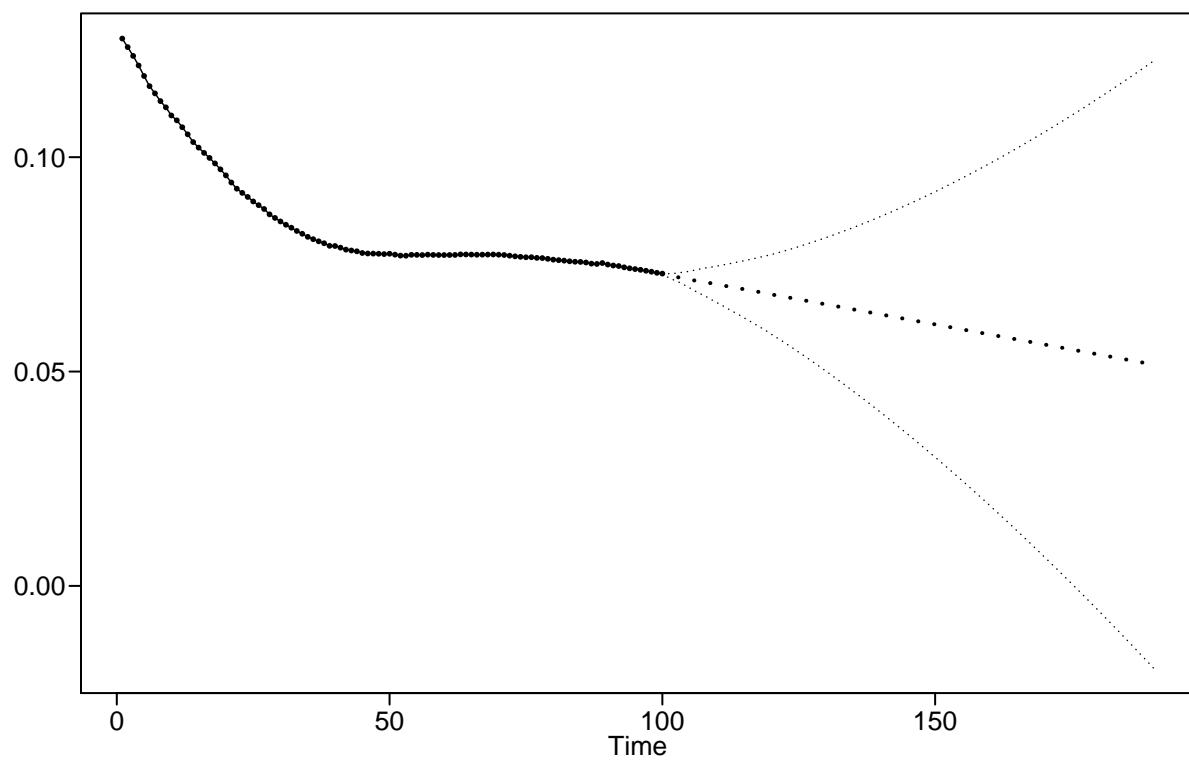


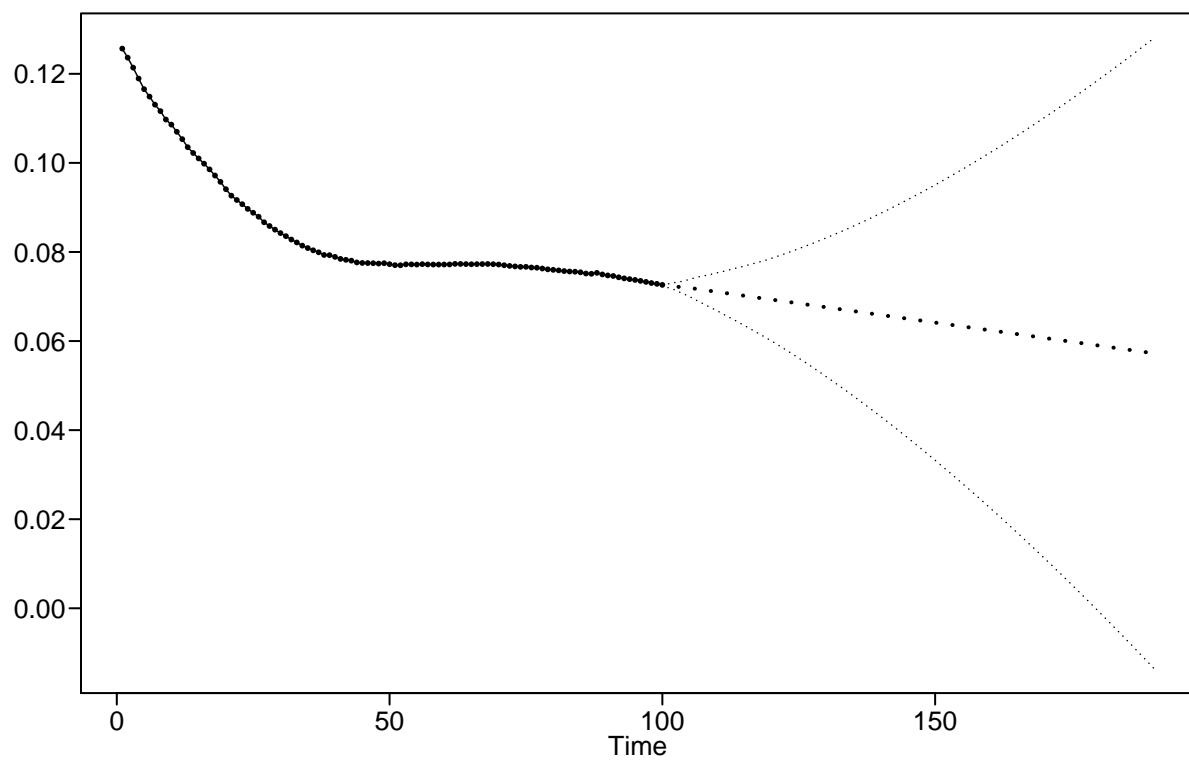


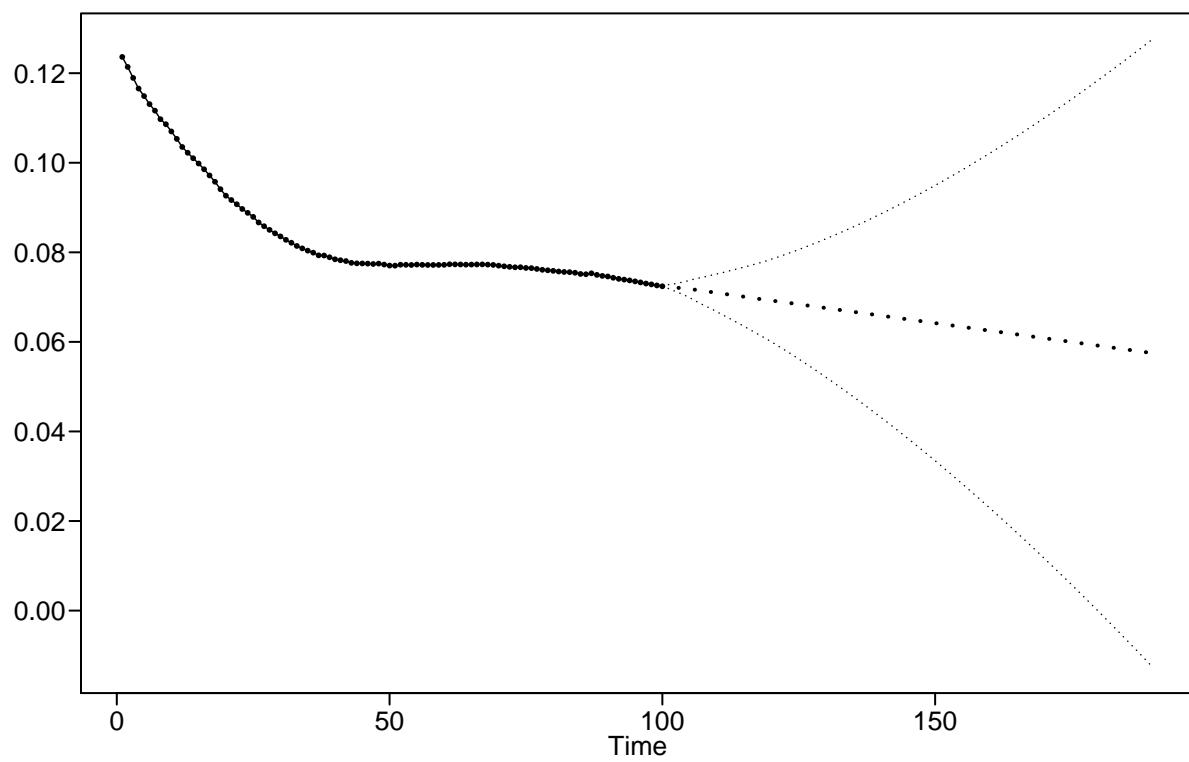


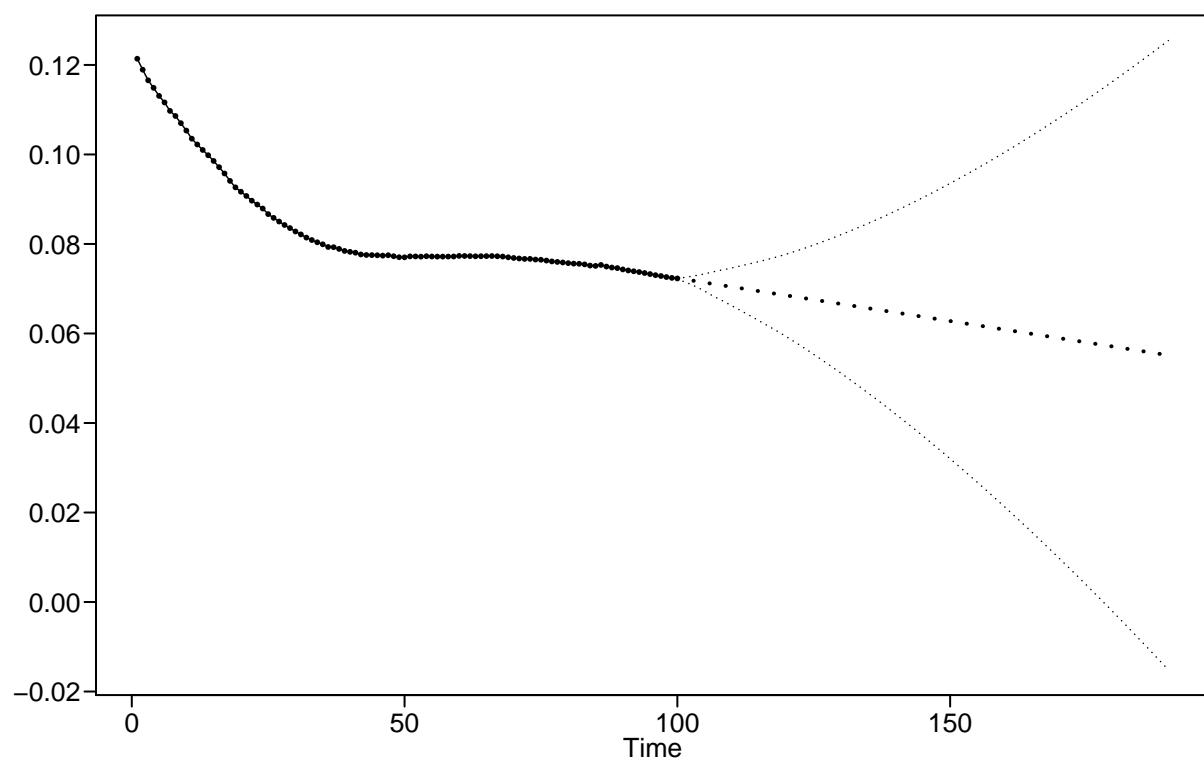


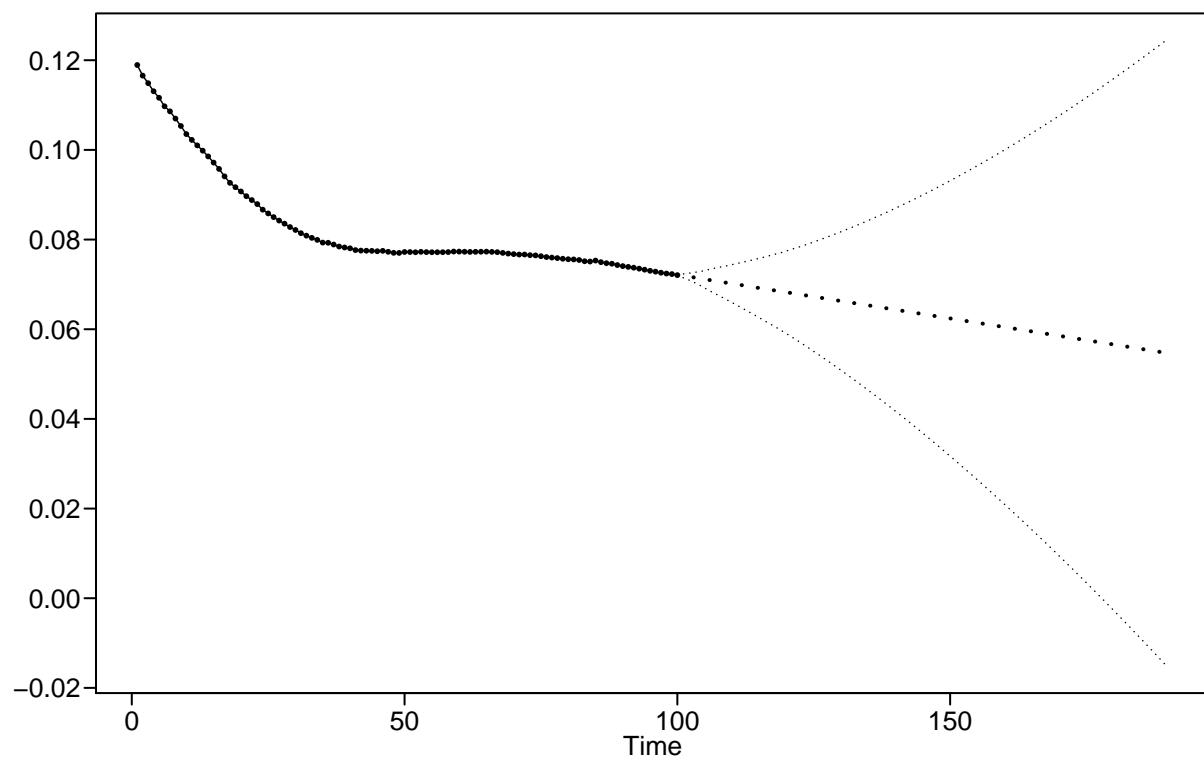






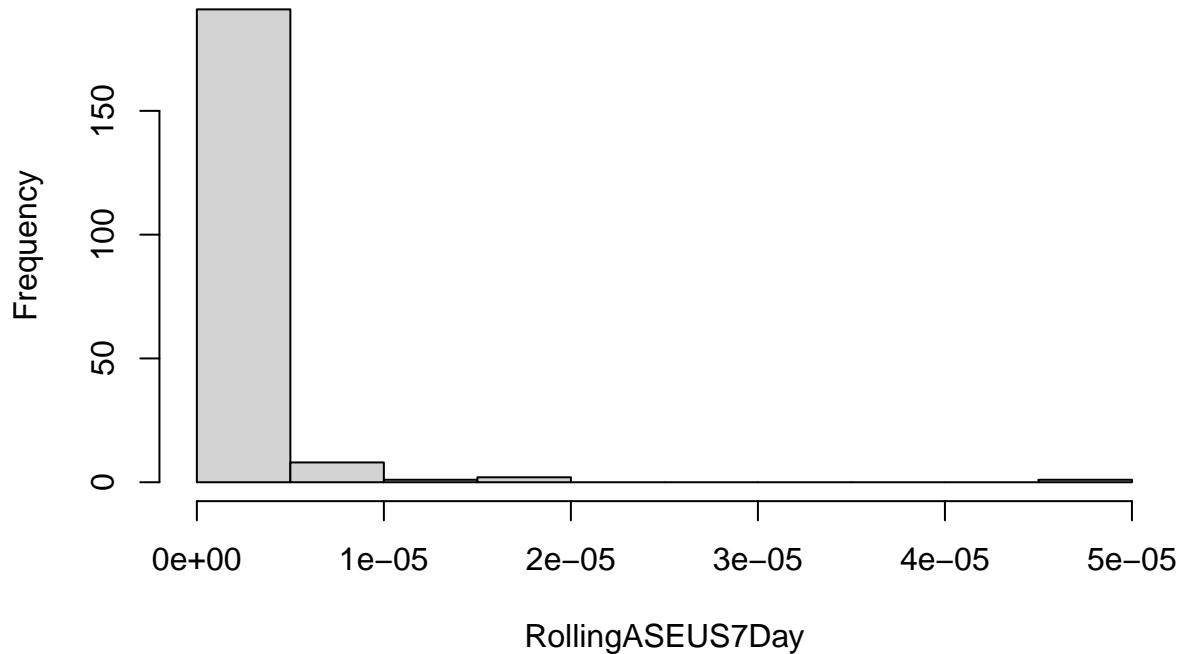






```
modelName = "US Model - 90 Days"
GetRollingWindowASEInfo(modelName, RollingASEUS90Day)
```

Histogram of US Model – 90 Days Windowed ASE



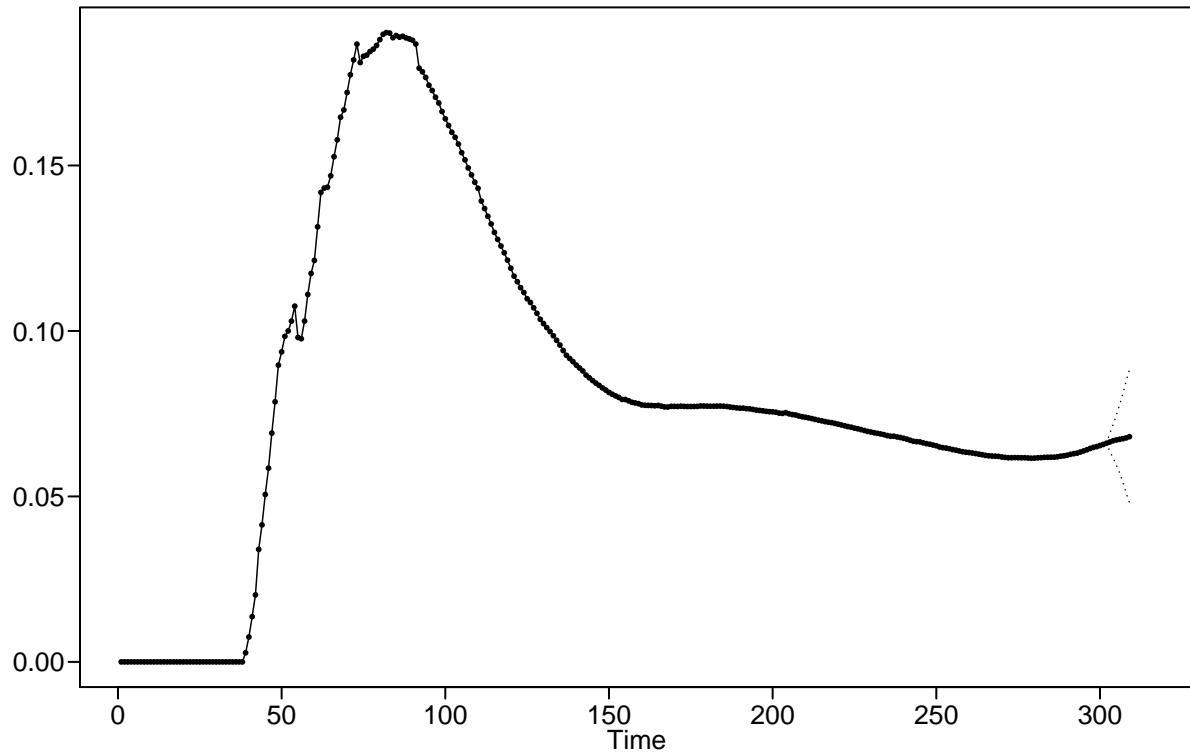
```
## [1] "US Model - 90 Days Windowed ASE Mean: 0.00173969287287246"  
## [1] "US Model - 90 Days Windowed ASE Median: 0.000224100192917218"
```

The rolling window ASEs are as follows:

- Rolling Window ASE - 7 Day: 1.26780e-06
- Rolling Window ASE - 90 Day: 0.0017

Now the individual forecasts for the short term and long term forecast horizons can be evaluated. The first evaluation will be the short term horizon and how well the forecast has been estimated.

```
#forecast 7 days ahead - zoom in  
fore.US7 = fore.aruma.wge(df_US_Final$Percent_Positive, phi=US.est$phi, theta=US.est$theta, d=US_Diffs, n.a.
```



In evaluating the forecasts above, it is difficult to tell from the full forecast just how well the forecast estimations are. The confidence intervals on the forecast are wide. In order to get a better view of the forecast compared to the original the plot below will provide better indications.

```

startPoint = length(df_US_Final$Percent_Positive) - 6
endPoint = length(df_US_Final$Percent_Positive)

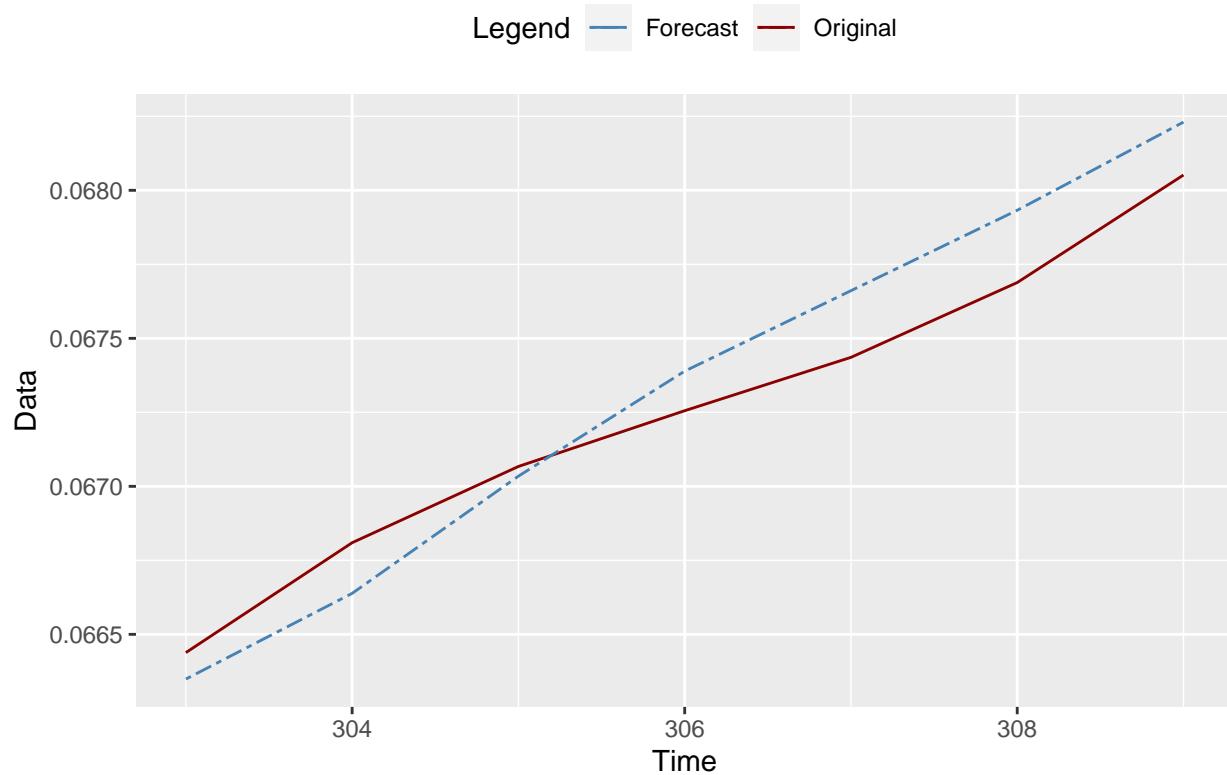
tl = seq(startPoint,endPoint)
orig = df_US_Final$Percent_Positive[startPoint:endPoint]
forecast = fore.US7$f
df_US7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Cases: Original vs. Forecast - 7 Day Horizon") +
  scale_color_manual(values = colors)

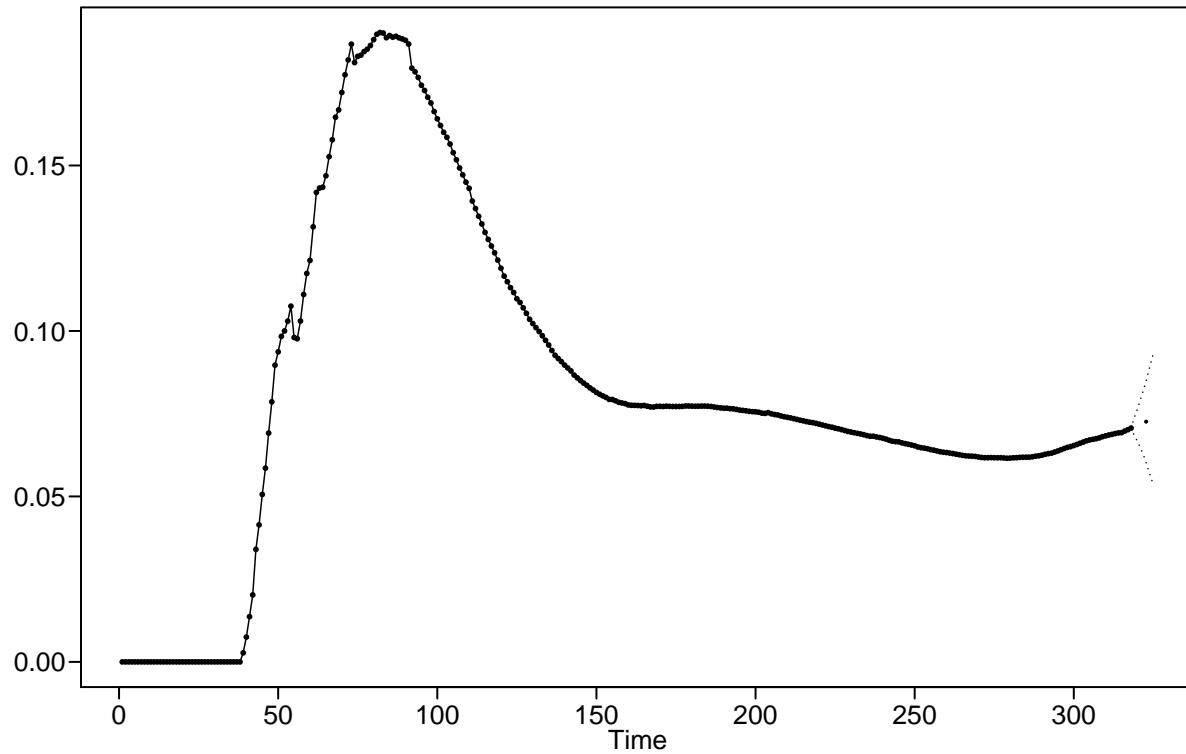
```

US Cases: Original vs. Forecast – 7 Day Horizon



In comparing the 7 day forecast to the original, the forecast trends well with the original data. Initially the forecast starts off lower but then progresses higher than the original data. Leveraging the forecast to project ahead on the short term horizon (7-days), the plot below shows the forecast with the large confidence interval bands that are produced.

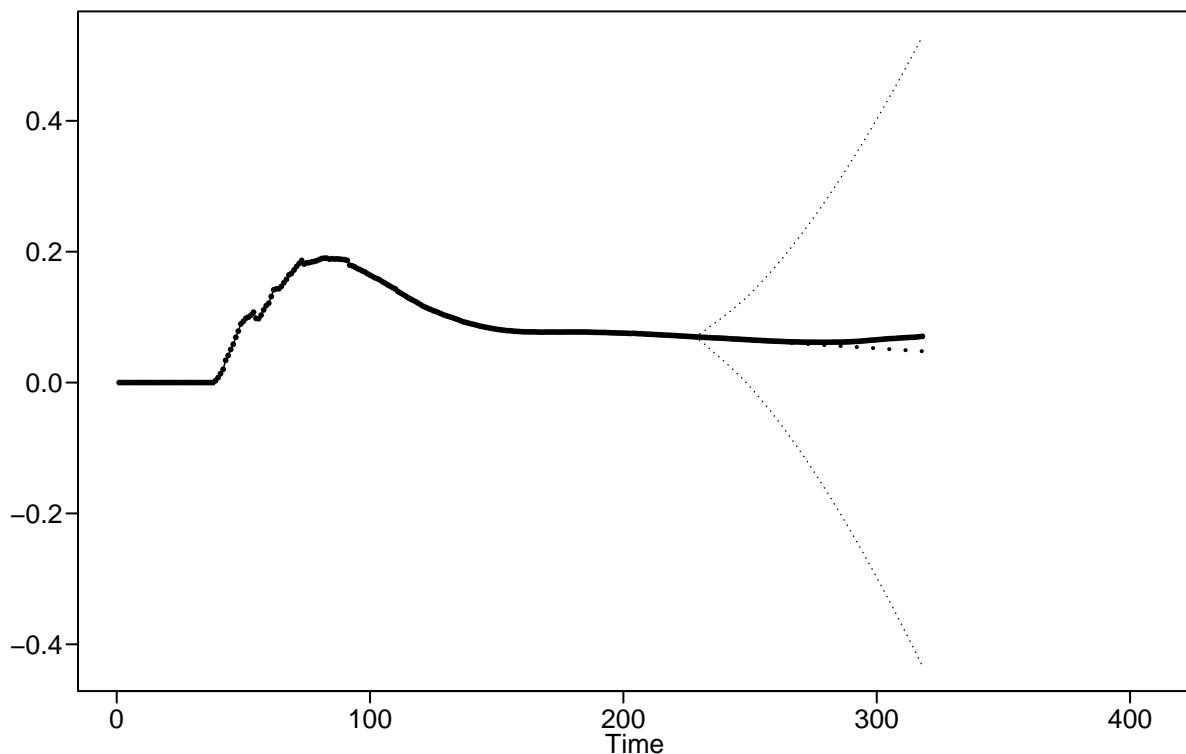
```
#forecast 7 days ahead
fore.US7 = fore.aruma.wge(df_US_Percent_Positive$Percent_Positive, phi=US.est$phi, theta=US.est$theta, d=U
```



Moving to the long term horizon (90-day) the same approach will be taken. First the comparison of the forecast to the original and then a projected forecast of the data. From the comparison of forecast data to original data the trends seem to be that the long term forecast is close to the original data. The confidence intervals are still wide which leaves some room for changes.

```
#forecast 90 days ahead - zoom in
```

```
fore.US90 = fore.aruma.wge(df_US_Percent_Positive$Percent_Positive,phi=US.est$phi,theta=US.est$theta,d=
```



Zooming into the forecast projection, below, the forecast trend is lower than the original data. The forecast maintains a downward linear trend which for the first part of the forecast maps well with the original data. At the end of the forecast the original and the forecast deviate with the forecast continuing the downward trend and the original data starting an upward trend.

```

startPoint = length(df_US_Final$Percent_Positive) - 89
endPoint = length(df_US_Final$Percent_Positive)

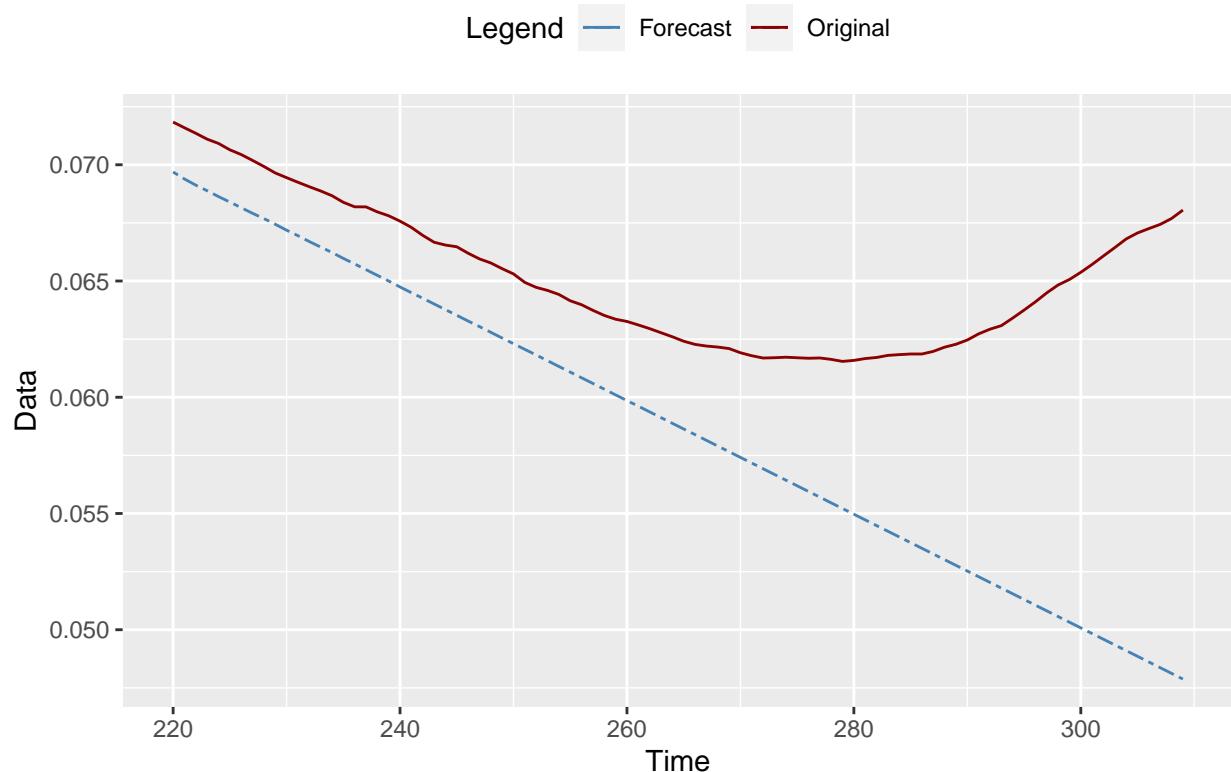
tl = seq(startPoint,endPoint)
orig = df_US_Final$Percent_Positive[startPoint:endPoint]
forecast = fore.US90$f
df_US90 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US90,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Cases: Original vs. Forecast - 90 Day Horizon") +
  scale_color_manual(values = colors)

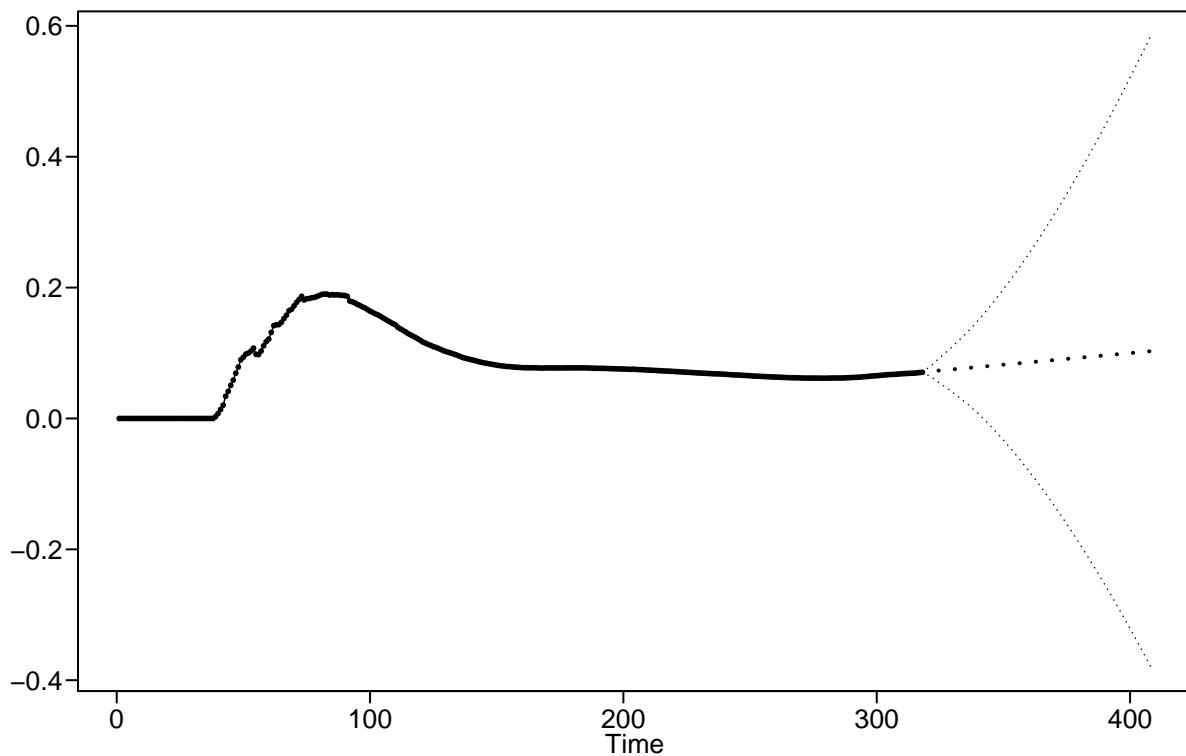
```

US Cases: Original vs. Forecast – 90 Day Horizon



As shown below, the projected forecast seems to have a few initial higher values and then trails off to a mean value as time progresses. As expected the confidence intervals for the forecast continue to get wider the further away from the data the forecast goes.

```
#forecast 90 days (3 months) ahead
fore.US90 = fore.aruma.wge(df_US_Percent_Positive$Percent_Positive,phi=US.est$phi,theta=US.est$theta,d=
```



Multi-Layer Perceptron (MLP)

Another time series modeling approach that can be taken is to leverage a neural network, also called a Multi-Layer Perceptron (MLP). Through this model an input layer is defined. Each input layer is connected to a “hidden layer”. The connection between the input layer and the “hidden layer” contains a weighting. The “hidden layer” is used to approximate any continuous function. Within the MLP model there can be one or more hidden layers. The “hidden layers” then all connect to an output layer. The output layer provides the final result of the model.

As was done for the ARIMA modeling, the MLP modeling will be very similar. First the model will be evaluated. The model will be set to perform 50 repetitions of the MLP. Each of these repetition results will be combined based on the mean calculation. In the case of the MLP, the models will be executed twice. Once to see evaluate the model on the base data and secondly to take into account the differencing that was seen in the initial ARIMA evaluation.

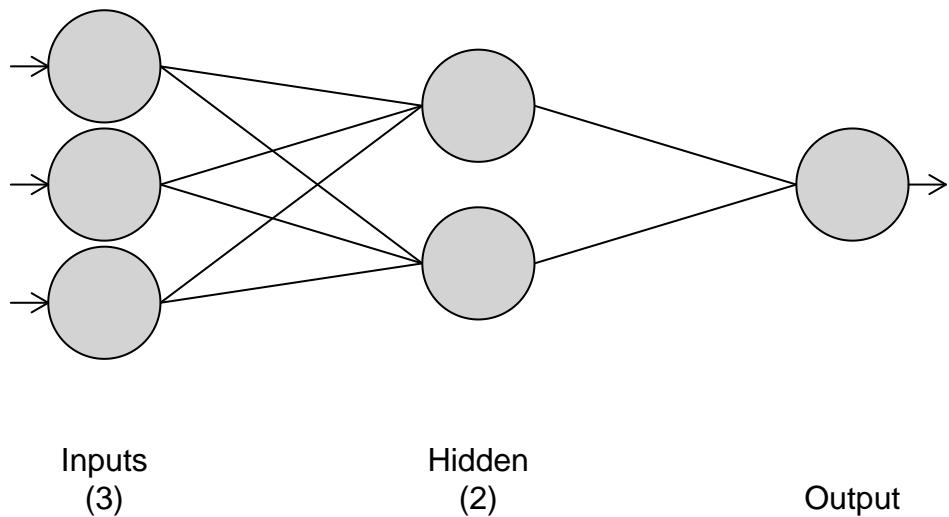
Evaluating the model with the base data results in the following:

```
set.seed(100)
fit.mlp.US = mlp(ts(df_US_Final$Percent_Positive), reps = NN_REPS, comb = "mean", hd.auto.type="cv")
fore.mlp.US = forecast(fit.mlp.US) # full forecast
fore.mlp.US7 = forecast(fit.mlp.US, h = SHORT_TERM_FORECAST_HORIZON)
fore.mlp.US90 = forecast(fit.mlp.US, h = LONG_TERM_FORECAST_HORIZON)
```

The visual below represents the MLP that was leveraged for the analysis. The MLP model has three (3) input layers, two (2) hidden layers and an output layer.

```
plot(fit.mlp.US)
```

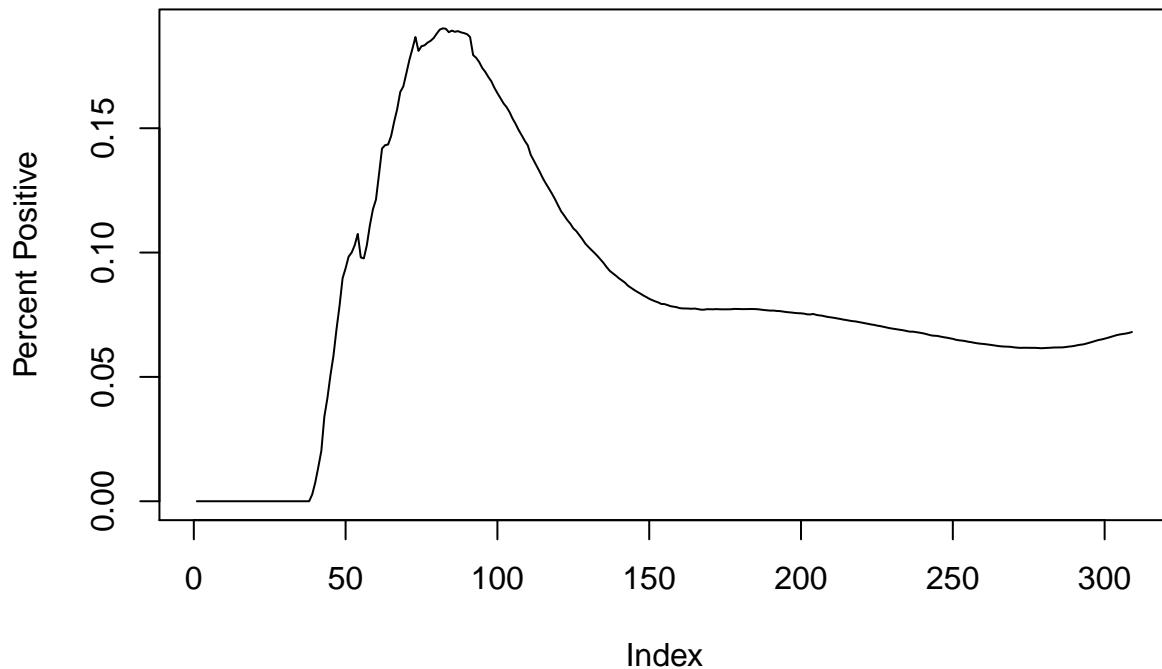
MLP



As a reminder, the plot below shows the current realization of the percent positive data for the US.

```
plot(df_US_Final$Percent_Positive, type = "l", ylab="Percent Positive", main="Percent Positive - Total US")
```

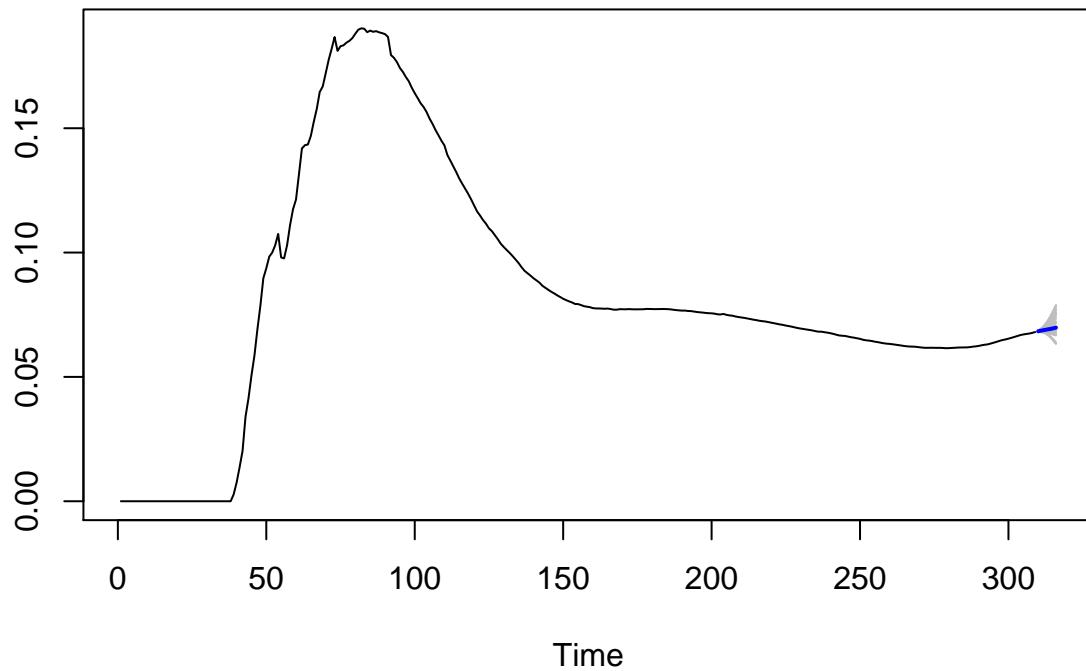
Percent Positive – Total US



The plot below shows the 7-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow the curve fairly well.

```
plot(fore.mlp.US7, ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total US")
```

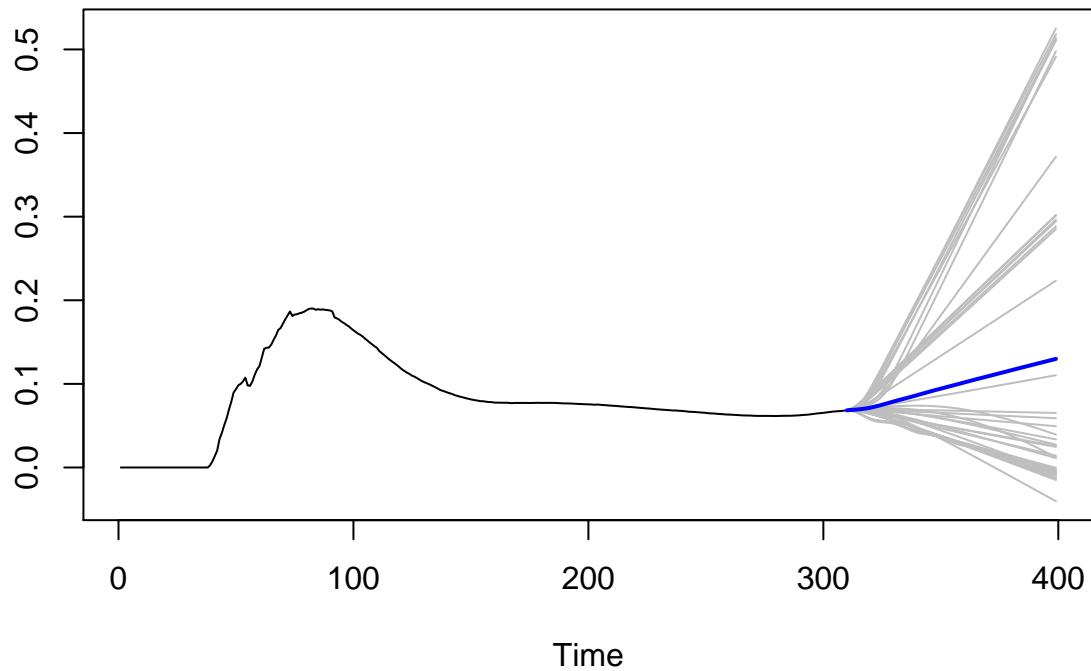
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow the curve fairly well.

```
plot(fore.mlp.US90,ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total US")
```

Forecasts from MLP

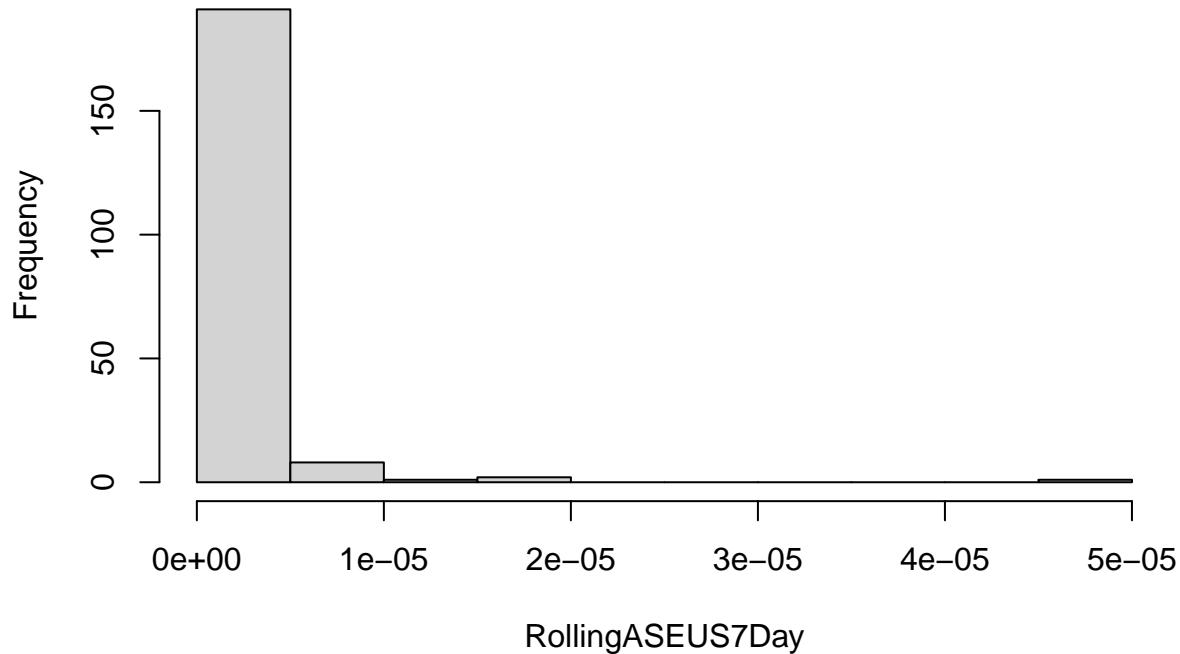


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcMLP7Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT_TERM)
RollingASECalcMLP90Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG_TERM)
```

```
modelName = "US Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcMLP7Day)
```

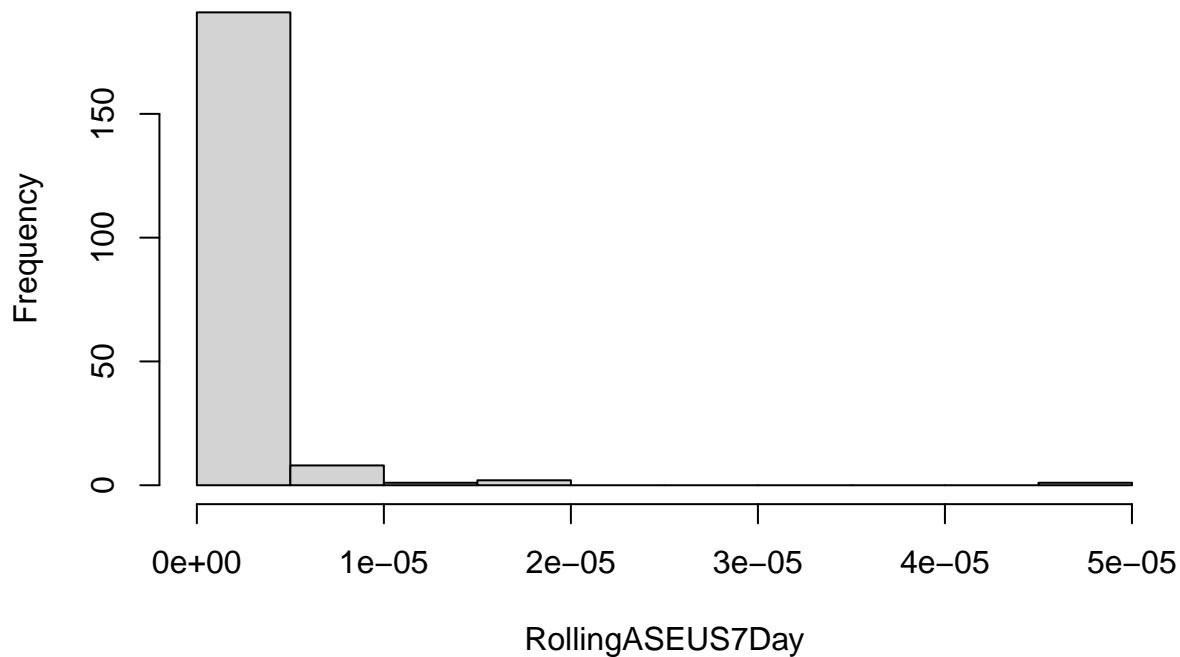
Histogram of US Model – 7 Days Windowed ASE



```
## [1] "US Model - 7 Days Windowed ASE Mean: 2.01956844442391e-06"  
## [1] "US Model - 7 Days Windowed ASE Median: 1.14642944922895e-07"
```

```
modelName = "US Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcMLP90Day)
```

Histogram of US Model – 90 Days Windowed ASE



```
## [1] "US Model - 90 Days Windowed ASE Mean: 0.000530849683581231"  
## [1] "US Model - 90 Days Windowed ASE Median: 5.60126383351628e-05"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 2.019568e-06
- Rolling Window ASE - 90 Day: 0.00053

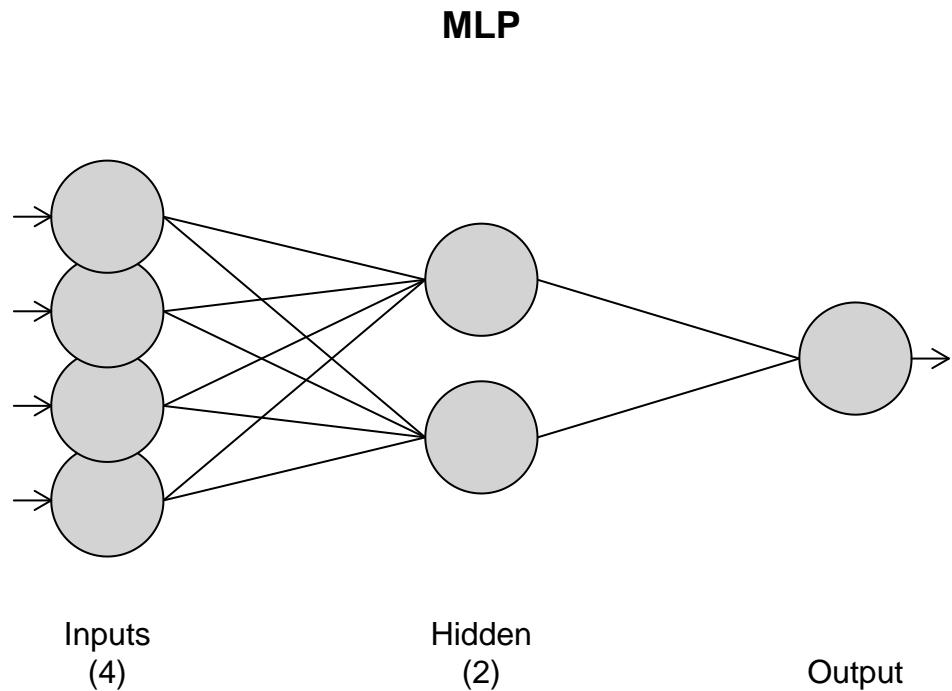
In comparison to the ARIMA(13,2,0) model that was developed these ASE values are lower for the 90 day horizon but slightly higher for the 7-day horizon.

MLP Analysis - Differenced Data The MLP model will now be executed taking into account the differencing that was seen through the ARIMA analysis.

```
set.seed(100)  
fit.mlp.USDiff = mlp(ts(df_US_Final$Percent_Positive), reps = NN_REPS, comb = "mean", hd.auto.type="cv",  
fore.mlp.USDiff = forecast(fit.mlp.USDiff)  
fore.mlp.US7Diff = forecast(fit.mlp.USDiff, h = SHORT_TERM_FORECAST_HORIZON)  
fore.mlp.US90Diff = forecast(fit.mlp.USDiff, h = LONG_TERM_FORECAST_HORIZON)
```

The visual below represents the MLP that was leveraged for the analysis. The MLP model has three (4) input layers, two (2) hidden layers and an output layer.

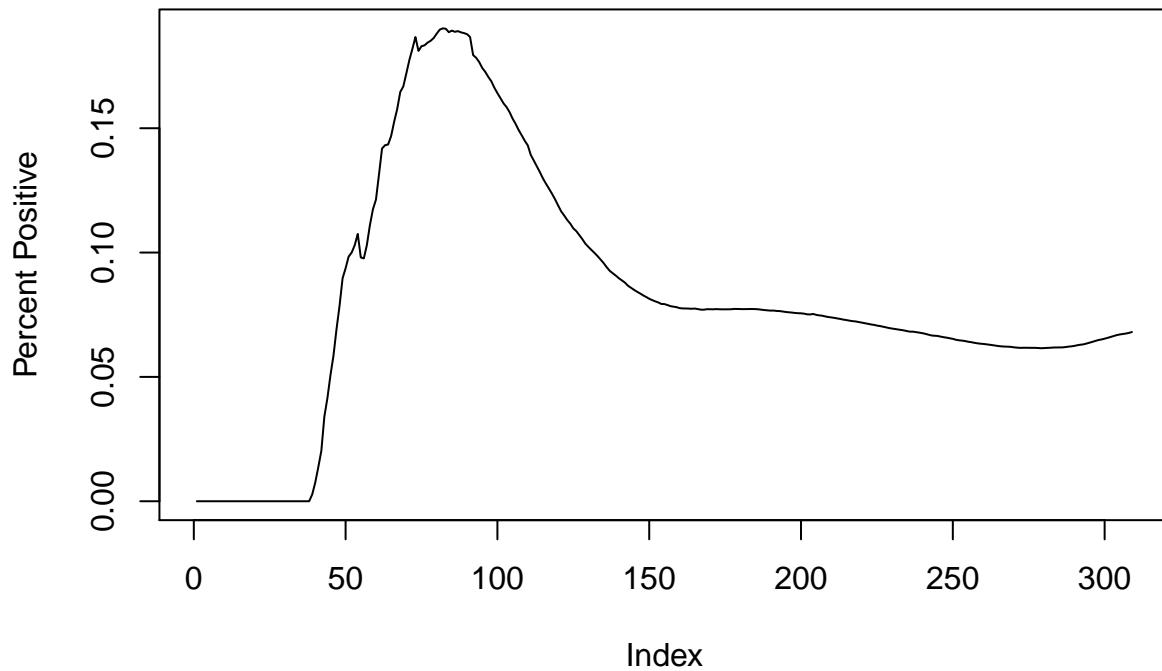
```
plot(fit.mlp.USDiff)
```



As a reminder, the plot below shows the current realization of the percent positive data for the US.

```
plot(df_US_Final$Percent_Positive, type = "l", ylab="Percent Positive", main="Percent Positive - Total US")
```

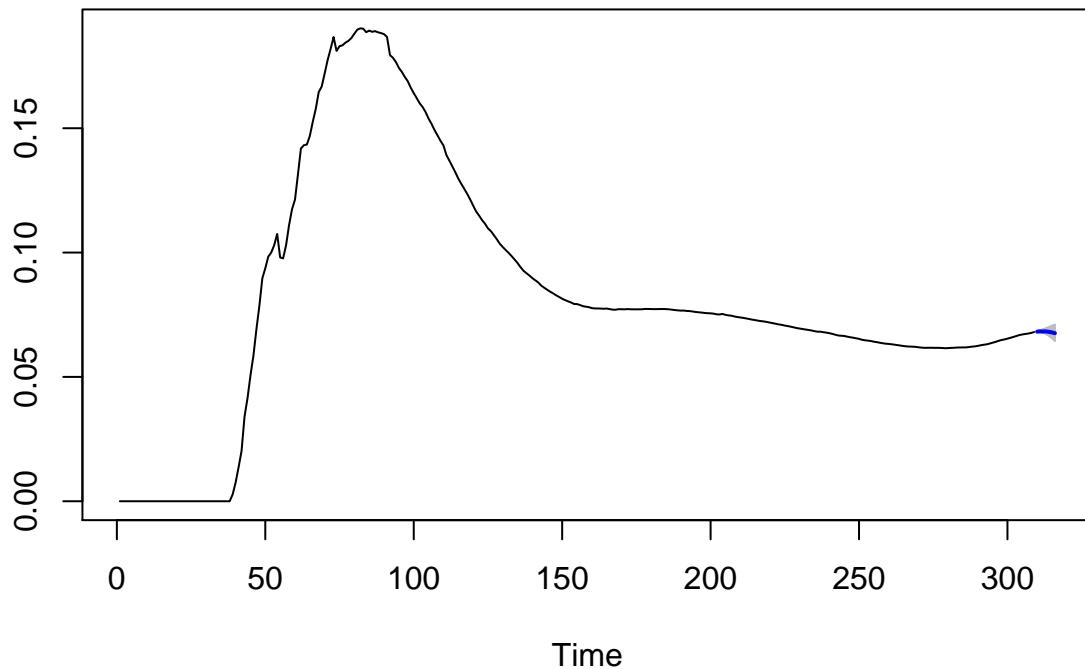
Percent Positive – Total US



The plot below shows the 7-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow the curve fairly well.

```
plot(fore.mlp.US7Diff, ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total US")
```

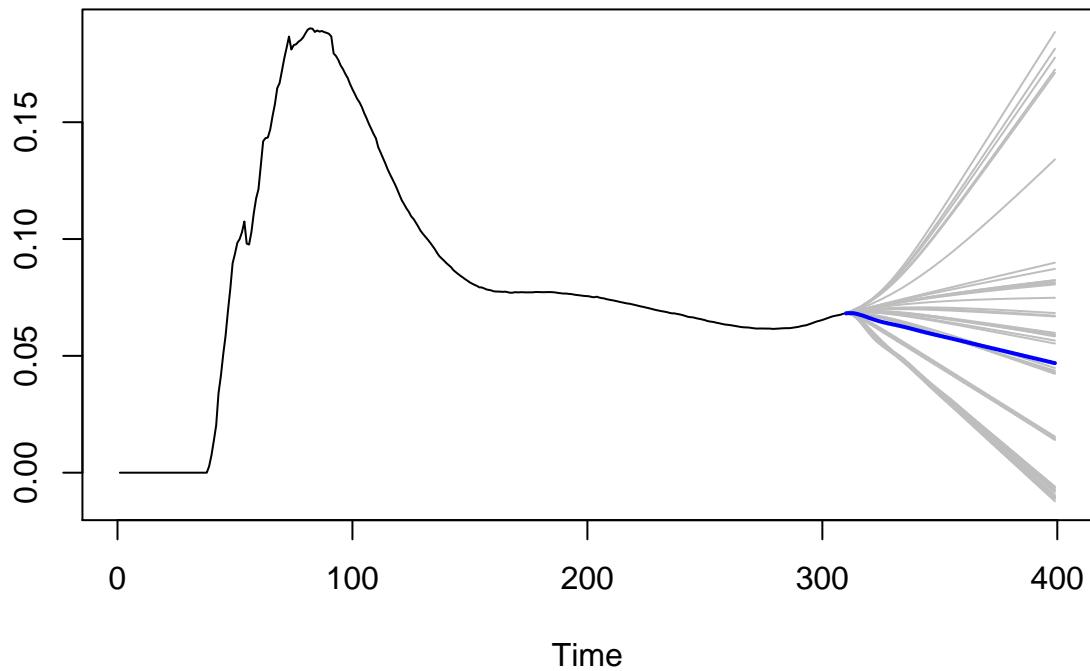
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow a more downward trajectory than the direction the original realization is moving toward.

```
plot(fore.mlp.US90Diff, ylab="Percent Positive", main="Percent Positive - 90 Day Forecast - Total US")
```

Forecasts from MLP

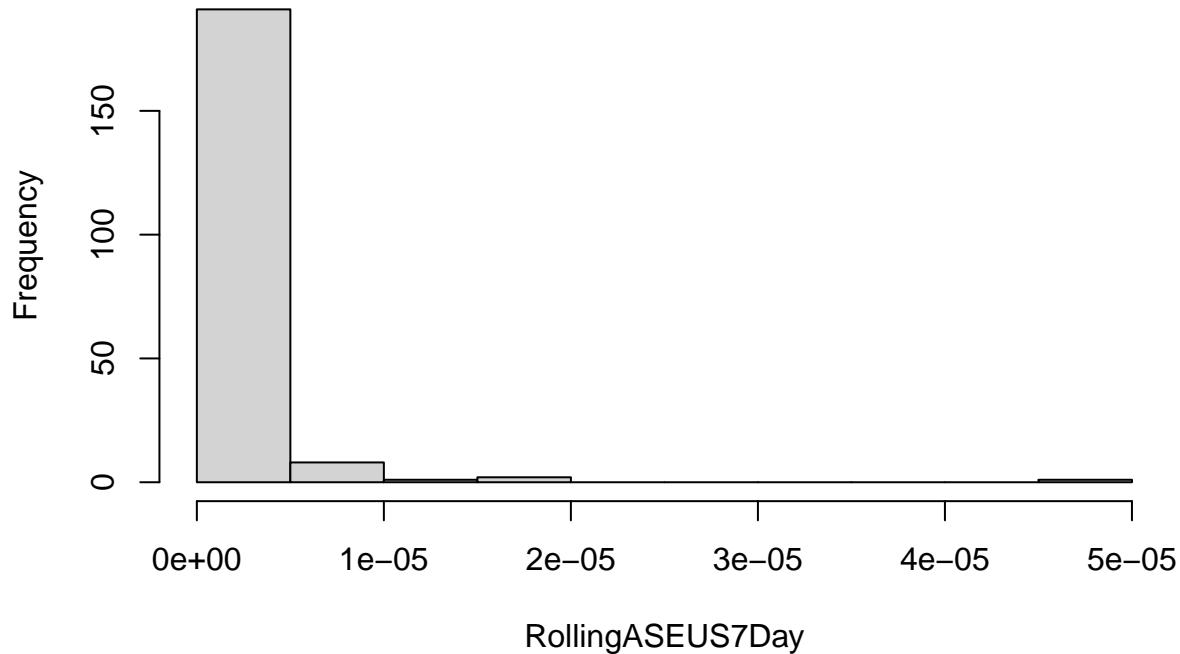


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcMLP7Day = RollingASECalcMLP(fit.mlp.USDiff$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT)
RollingASECalcMLP90Day = RollingASECalcMLP(fit.mlp.USDiff$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG)
```

```
modelName = "US Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcMLP7Day)
```

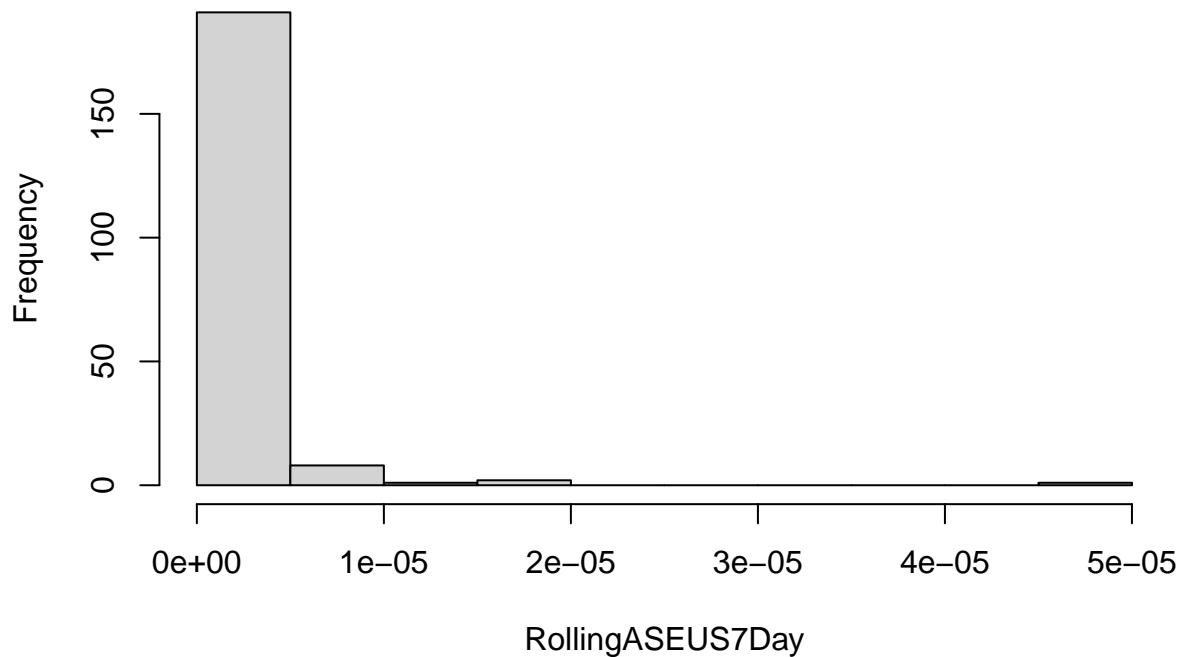
Histogram of US Model – 7 Days Windowed ASE



```
## [1] "US Model - 7 Days Windowed ASE Mean: 2.01956844442391e-06"  
## [1] "US Model - 7 Days Windowed ASE Median: 1.14642944922895e-07"
```

```
modelName = "US Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcMLP90Day)
```

Histogram of US Model – 90 Days Windowed ASE



```
## [1] "US Model - 90 Days Windowed ASE Mean: 0.000530849683581231"  
## [1] "US Model - 90 Days Windowed ASE Median: 5.60126383351628e-05"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 2.019568e-06
- Rolling Window ASE - 90 Day: 0.00053

In comparison to the ARIMA(13,2,0) model that was developed these ASE values are lower for the 90 day horizon but slightly higher for the 7-day horizon. The ASEs are approximately the same as the original MLP model. From a visual perspective the original MLP model seems to forecast the data trend better.

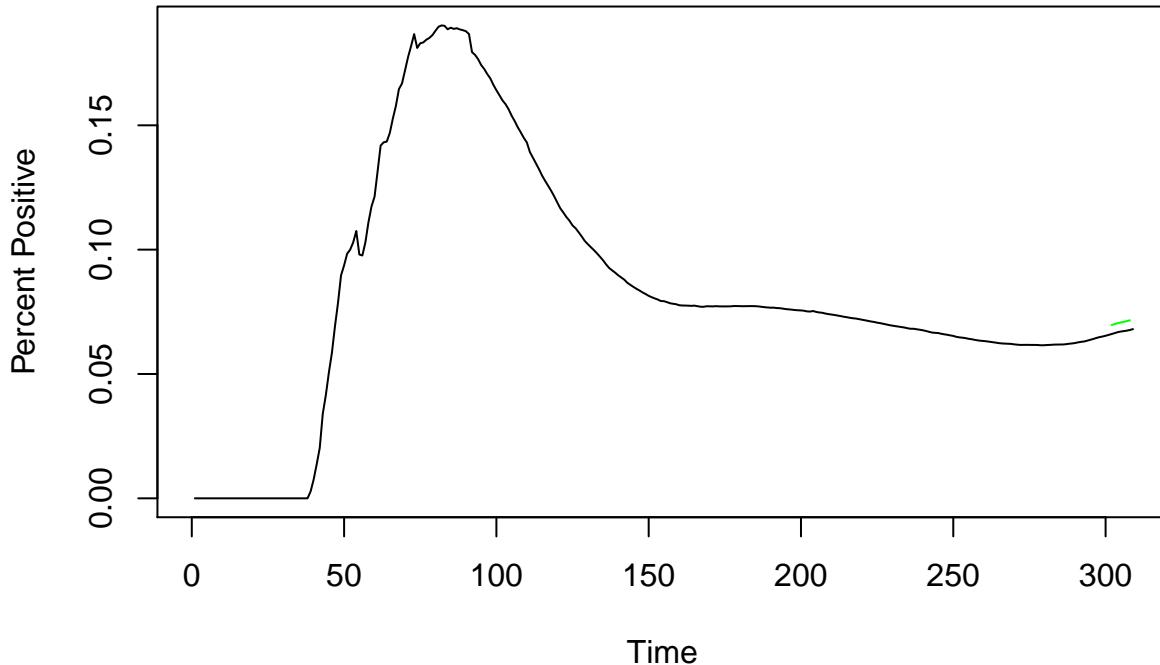
Univariate Ensemble Modeling Ensemble modeling is a way of combining models to see if the positives of both models will make the forecasts better. In the univariate modeling case the ARIMA(13,2,0) model and the non-differenced MLP model will be combined to see if there is better performance. Ensemble models will be executed for both the short-term and long-term forecasts.

Start with the short term forecast. The ensemble projection, shown in green, seems to track well for the short-term timeframe.

```
startPos = length(df_US_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON  
endPos = length(df_US_Final$Percent_Positive)-1  
  
USUV7Ensemble = (fore.US7$f + fore.mlp.US7$mean)/2
```

```
plot(seq(1,length(df_US_Final$Percent_Positive),1),df_US_Final$Percent_Positive, type = "l",xlim = c(1,300))
lines(seq(startPos,endPos,1), USUV7Ensemble, type = "l", col = "green")
```

Total US – Percent Positive – 7 Day Forecast – Ensemble Model



```
USUV7EnsembleASE = mean((df_US_Final$Percent_Positive[startPos:endPos] - USUV7Ensemble)^2)
```

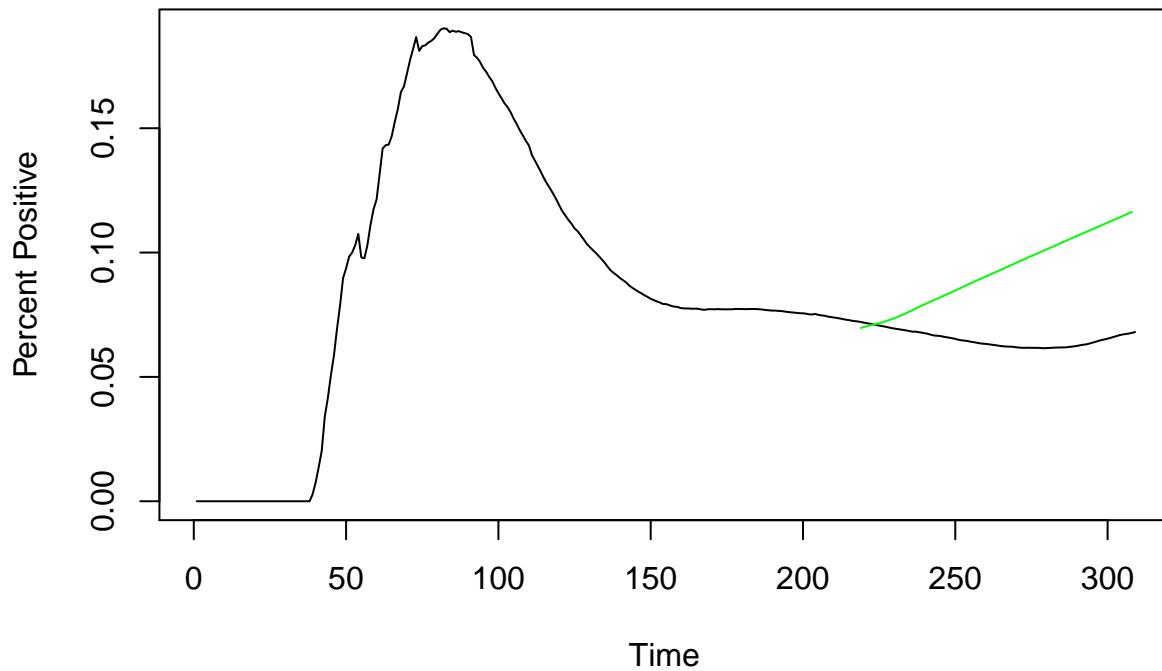
Now move to the 90 day forecast. The long-term forecast for the ensemble does not track well to the data. It has a sharp upward trend when compared to the original data plot.

```
startPos = length(df_US_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON
endPos = length(df_US_Final$Percent_Positive)-1

USUV90Ensemble = (fore.US90$f + fore.mlp.US90$mean)/2

plot(seq(1,length(df_US_Final$Percent_Positive),1),df_US_Final$Percent_Positive, type = "l",xlim = c(1,300))
lines(seq(startPos,endPos,1), USUV90Ensemble, type = "l", col = "green")
```

Total US – Percent Positive – 90 Day Forecast – Ensemble Model



```
USUV90EnsembleASE = mean((df_US_Final$Percent_Positive[startPos:endPos] - USUV90Ensemble)^2)
```

ASEs for the ensemble models are:

```
print(paste("ASE Ensemble 7-Day: ", USUV7EnsembleASE))
```

```
## [1] "ASE Ensemble 7-Day: 1.37443701770578e-05"
```

```
print(paste("ASE Ensemble 90-Day: ", USUV90EnsembleASE))
```

```
## [1] "ASE Ensemble 90-Day: 0.000994420614216203"
```

The ASEs for both the 7 day and 90 day ensemble models are larger than any other the other models developed.

US Univariate Modeling Summary In summary for US univariate modeling, the following models were developed with ASEs for each of the forecast periods. As shown, the MLP models seemed to perform the best on both forecasts. Individually the ARIMA(13,2,0) model performed best on the 7-day horizon and the MLP performed best on the 90-day horizon.

Method	ASE - 7 Day	ASE - 90 Day
ARIMA(13,2,0)	1.26780e-06	0.00173
MLP - No Difference	2.01956e-06	0.00053
MLP - Difference = 2	2.01956e-06	0.00053
Ensemble	1.37443e-05	0.00099

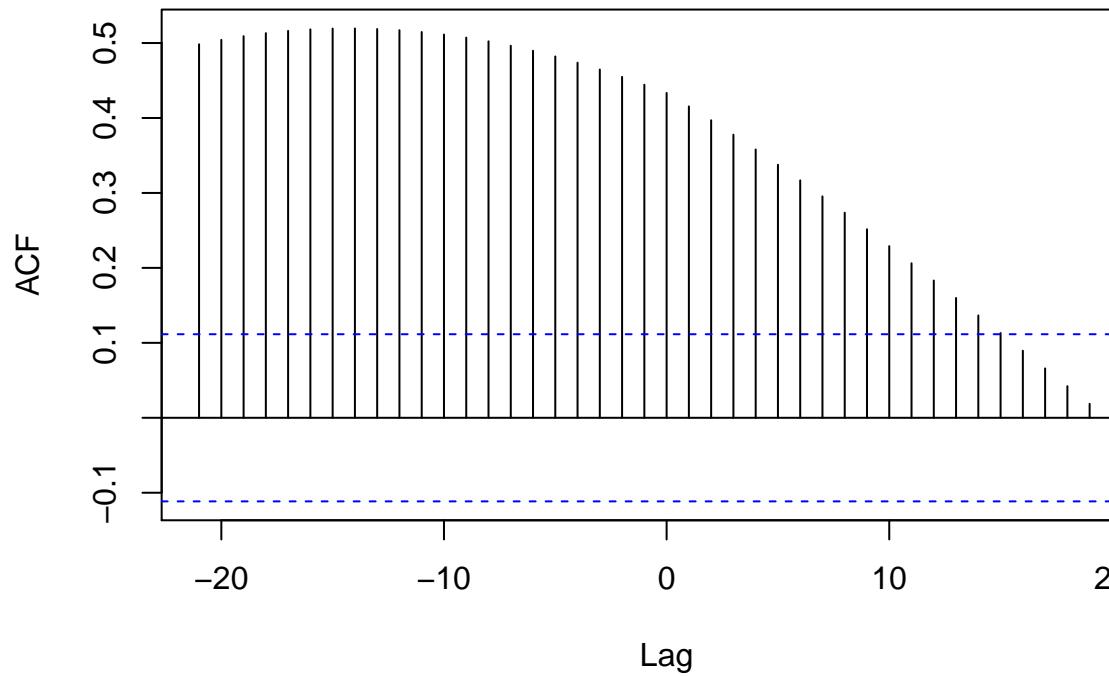
Multi-variate analysis

Progressing forward in the US Analysis, the time series modeling will now shift to multivariate analysis. In multivariate analysis additional variables are added to the analysis to continue to refine the model. For this analysis the number of patients hospitalized will be used as the external variable. The goal is to see if the number of hospitalizations is a leading or lagging indicator for the percent positive metric.

Initially a cross correlation plot will be built to see if there is any correlation between the two variables. As shown below, the cross correlation plot seems to indicate that there is a correlation around lag -15, -14, -13. All three of these lags have the exact same value.

```
#x1 = dependent var
#x2 = independent var
# call is ccf(x2,x1)
ccf(df_US_Final$Percent_Positive, df_US_Final$Hospitalized_Count) # looks like correlation may be at
```

df_US_Final\$Percent_Positive & df_US_Final\$Hospitalized_Count



Cross Correlation Plots

Now that it is understood that the hospitalization count and percent positive rate are correlated, multi-

ivariate analysis can begin. In executing the multivariate analysis two different types of model development will occur. Just as with the univariate model a MLP model will be build for the multivariate analysis. In addition to the MLP model a VAR model will be developed. A VAR model allows for understanding the relationships between all variable types, independent or dependent. These variables are all treated the same in a VAR model and thus could result in a better application of forecasting as all variable interactions are accounted for.

As performed in the univariate analysis forecasts for short term (7-days) and long term (90-day) horizons will be leveraged. Rolling window ASE will be used as the metric

VAR Analysis - 7 Day The VAR model for the 7-day horizon will be first evaluated. The default model settings will be used with a maximum lag of 15 for the model to select from.

```
# We know out to length of df_US_Final$Percent_Positive
backLen = length(df_US_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON

X = cbind(df_US_Final$Percent_Positive[1:backLen],df_US_Final$Hospitalized_Count[1:backLen])
names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut

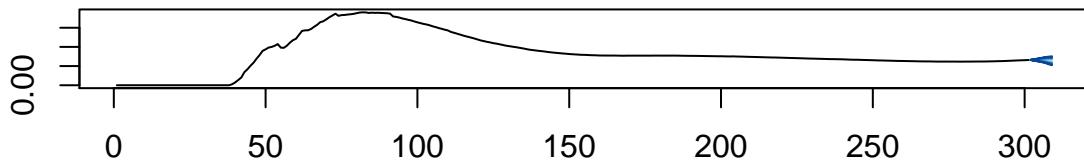
## $selection
## AIC(n)  HQ(n)  SC(n)  FPE(n)
##      15      10      3      15
##
## $criteria
##          1       2       3       4       5       6       7
## AIC(n) 2.031031 1.054713 0.9263808 0.9218343 0.8823216 0.8791423 0.7702137
## HQ(n)  2.061693 1.105816 0.9979255 1.0138202 0.9947489 1.0120109 0.9235236
## SC(n)  2.107536 1.182221 1.1048922 1.1513489 1.1628394 1.2106634 1.1527380
## FPE(n) 7.621951 2.871170 2.5254018 2.5140008 2.4166849 2.4091317 2.1606397
##          8       9      10      11      12      13      14
## AIC(n) 0.7336637 0.7042463 0.6475617 0.6403952 0.6601841 0.6275632 0.6489680
## HQ(n)  0.9074149 0.8984388 0.8621956 0.8754704 0.9157006 0.9035210 0.9453672
## SC(n)  1.1671912 1.1887770 1.1830957 1.2269324 1.2977245 1.3161068 1.3885149
## FPE(n) 2.0832755 2.0231064 1.9118775 1.8985379 1.9368625 1.8751324 1.9162154
##          15
## AIC(n) 0.5624719
## HQ(n)  0.8793123
## SC(n)  1.3530220
## FPE(n) 1.7579775
```

From the output above, the VAR model selects a VAR(15) model based on the AIC criteria. Fitting that model to the predictions for the individual variables produces the outputs below. In this instance y_1 = Percent Positive and y_2 = Hospital Count. From the plots, it is shown that the VAR(15) model seems to predict those values fairly well.

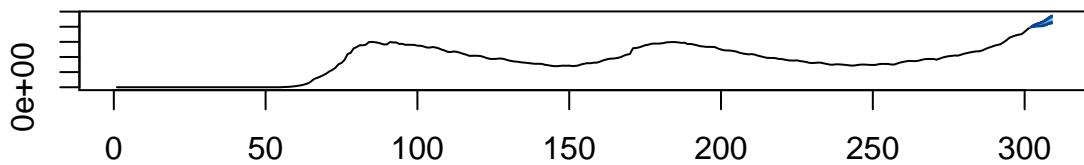
```
lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="const")
USVARPreds7Day=predict(lsfit,n.ahead=SHORT_TERM_FORECAST_HORIZON)

#Plot forecast predictions
fanchart(USVARPreds7Day, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make dist
```

Fanchart for variable y1



Fanchart for variable y2



Zooming in on the 7-day forecast for the percent positive from the VAR(15) model, the projected forecast initially tracks well but then deviates highly toward the end of the forecast.

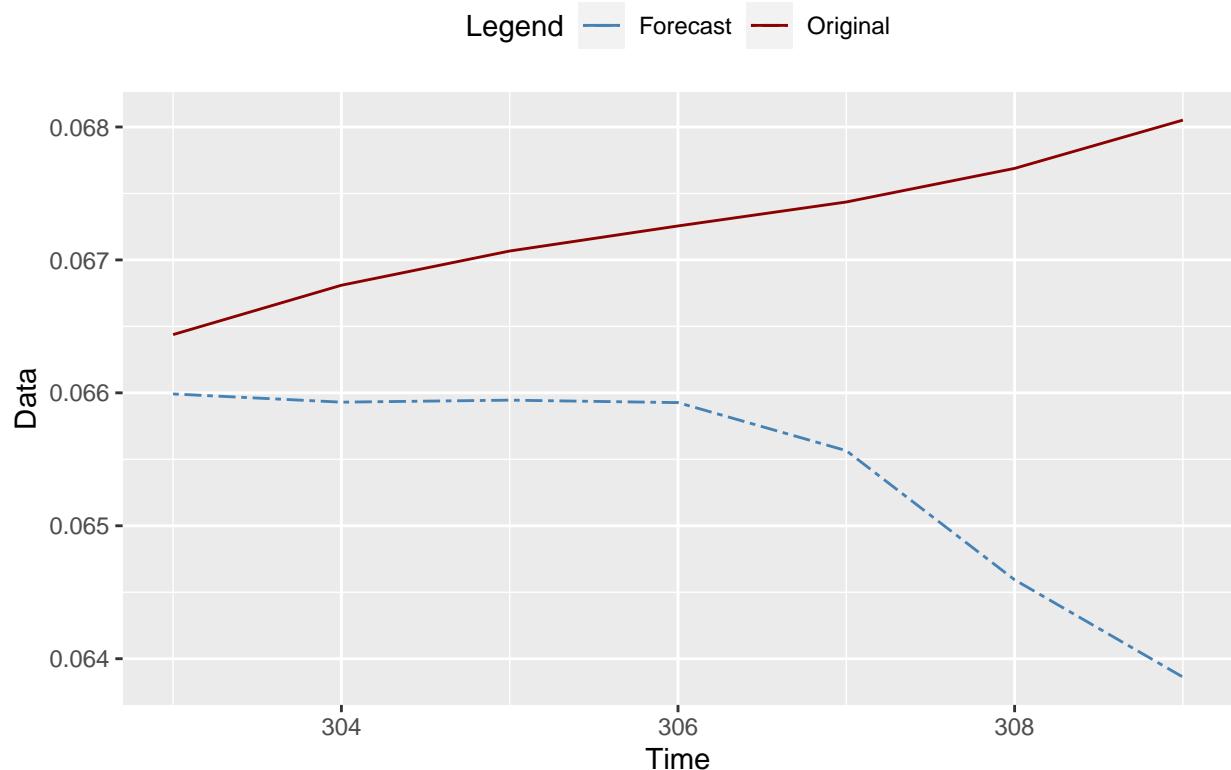
```
# Zoom In Percent Positive - 7day
startPoint = length(df_US_Final$Percent_Positive) - 6
endPoint = length(df_US_Final$Percent_Positive)

tl = seq(startPoint,endPoint)
orig = df_US_Final$Percent_Positive[startPoint:endPoint]
forecast = USVARPreds7Day$fcst$y1[,1]
df_US7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Percent Positive: Original vs. Forecast - 7 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

US Percent Positive: Original vs. Forecast – 7 Day Horizon – VAR Model



Zooming in on the 7-day forecast for the hospitalization count from the VAR(15) model, the projected forecast trends well with the original values.

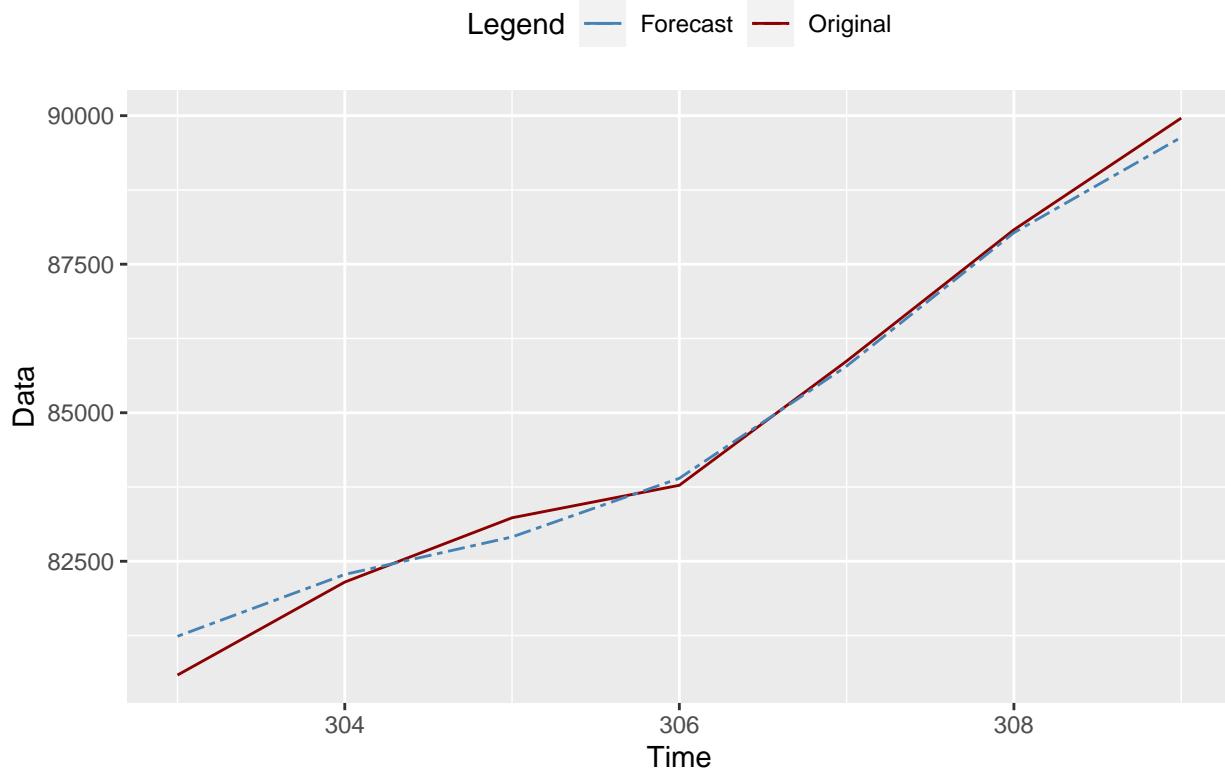
```
# Zoom In Hospital - 7day
startPoint = length(df_US_Final$Hospitalized_Count) - 6
endPoint = length(df_US_Final$Hospitalized_Count)

tl = seq(startPoint,endPoint)
orig = df_US_Final$Hospitalized_Count[startPoint:endPoint]
forecast = USVARPreds7Day$fcst$y2[,1]
df_US7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Hospitalization: Original vs. Forecast - 7 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

US Hospitalization: Original vs. Forecast – 7 Day Horizon – VAR Model



Forecasting out the full 7-day period, the following is shown. The VAR model still selects the VAR(15) for the full dataset. The predictions for the 7-day horizon still seem to trend well with the variables.

```
# Full Forward 7-day
X = cbind(df_US_Final$Percent_Positive, df_US_Final$Hospitalized_Count)
names(X) = c(COL_PERCENT_POSITIVE, COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut
```

```
## $selection
## AIC(n)  HQ(n)  SC(n)  FPE(n)
##      15      10      3      15
##
## $criteria
##          1       2       3       4       5       6       7
## AIC(n) 2.046835 1.025155 0.8975791 0.8922006 0.8498379 0.8464605 0.7339070
## HQ(n)  2.076941 1.075330 0.9678250 0.9825167 0.9602242 0.9769171 0.8844338
## SC(n)  2.122010 1.150447 1.0729877 1.1177259 1.1254799 1.1722193 1.1097825
## FPE(n) 7.743367 2.787545 2.4537001 2.4405877 2.3394311 2.3316495 2.0835733
##          8       9      10      11      12      13      14
## AIC(n) 0.6896901 0.6596453 0.6044475 0.5971893 0.616988 0.5838484 0.6050768
## HQ(n)  0.8602872 0.8503125 0.8151851 0.8279970 0.867866 0.8547966 0.8960953
## SC(n)  1.1156823 1.1357542 1.1306732 1.1735317 1.243447 1.2604242 1.3317694
## FPE(n) 1.9936127 1.9348041 1.8311331 1.8181691 1.854864 1.7947870 1.8337518
##          15
## AIC(n) 0.5153119
```

```

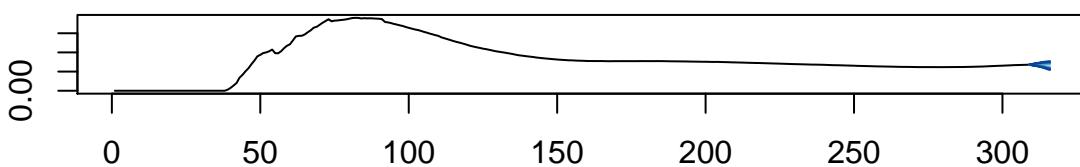
## HQ(n)  0.8264006
## SC(n)  1.2921212
## FPE(n) 1.6767971

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="const")
preds=predict(lsfit,n.ahead=SHORT_TERM_FORECAST_HORIZON)

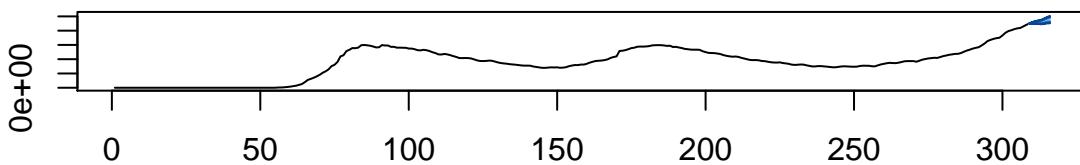
#Plot forecast predictions
fanchart(preds, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make distinguishable

```

Fanchart for variable y1



Fanchart for variable y2



VAR Analysis - 90 Day. The VAR model for the 90-day horizon will be first evaluated. The default model settings will be used with a maximum lag of 15 for the model to select from.

```

# We know out to length of df_US_Final$Percent_Positive
backLen = length(df_US_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON

X = cbind(df_US_Final$Percent_Positive[1:backLen],df_US_Final$Hospitalized_Count[1:backLen])
names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "both", season = NULL, exogen = NULL)
vsOut

## $selection
## AIC(n)  HQ(n)  SC(n)  FPE(n)
##      15      8      3     15

```

```

## 
## $criteria
##      1      2      3      4      5      6      7      8
## AIC(n) 2.343897 1.580059 1.430761 1.418232 1.391798 1.402325 1.315954 1.277307
## HQ(n)  2.396534 1.659014 1.536035 1.549825 1.549709 1.586554 1.526502 1.514173
## SC(n)  2.474019 1.775243 1.691006 1.743538 1.782165 1.857754 1.836444 1.862858
## FPE(n) 10.421877 4.855408 4.182216 4.130464 4.023170 4.066398 3.730716 3.590269
##      9     10     11     12     13     14     15
## AIC(n) 1.268777 1.263392 1.258883 1.284420 1.263057 1.288511 1.194564
## HQ(n)  1.531961 1.552894 1.574704 1.626560 1.631515 1.683288 1.615659
## SC(n)  1.919388 1.979065 2.039617 2.130215 2.173914 2.264429 2.235543
## FPE(n) 3.560996 3.543361 3.529205 3.622657 3.548569 3.643000 3.319413

```

From the output above, the VAR model selects a VAR(15) model based on the AIC criteria. Fitting that model to the predictions for the individual variables produces the outputs below. In this instance y_1 = Percent Positive and y_2 = Hospital Count. From the plots, it is shown that the VAR(15) model seems to predict those values fairly well.

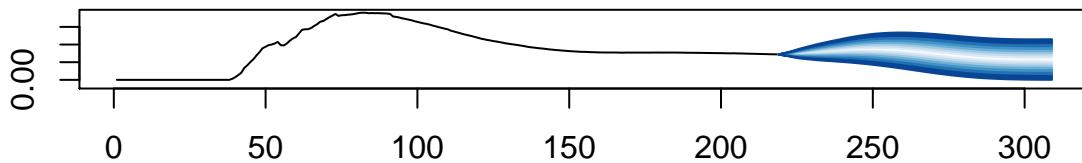
```

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="both")
USVARPreds90Day=predict(lsfit,n.ahead=LONG_TERM_FORECAST_HORIZON)

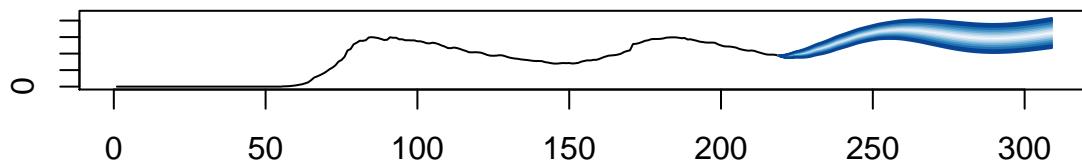
#Plot forecast predictions
fanchart(USVARPreds90Day, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make dis

```

Fanchart for variable y_1



Fanchart for variable y_2



Zooming in on the 90-day forecast for the percent positive from the VAR(15) model, the projected forecast initially over shoots the original value and then crosses under the original values. It does not follow the original trend line very well.

```

# Zoom In Percent Positive - 90day
startPoint = length(df_US_Final$Percent_Positive) - 89
endPoint = length(df_US_Final$Percent_Positive)

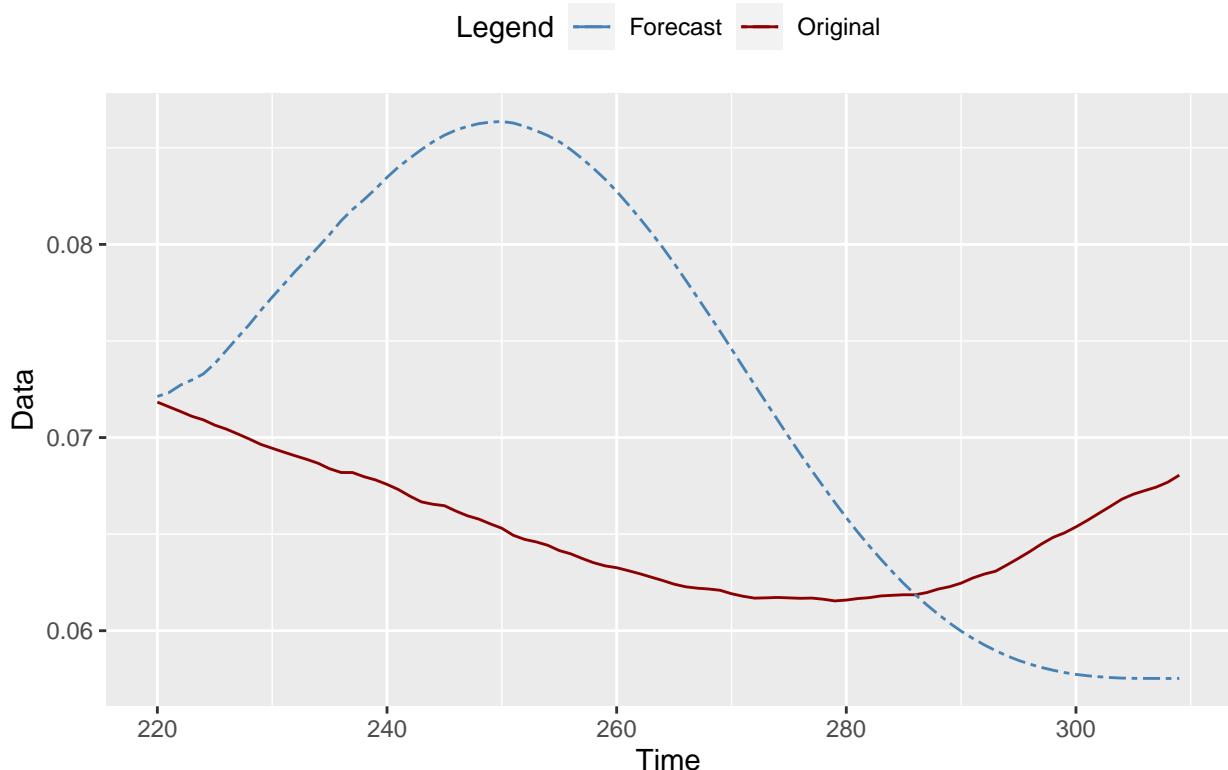
tl = seq(startPoint,endPoint)
orig = df_US_Final$Percent_Positive[startPoint:endPoint]
forecast = USVARPreds90Day$fcst$y1[,1]
df_US7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Percent Positive: Original vs. Forecast - 90 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)

```

US Percent Positive: Original vs. Forecast – 90 Day Horizon – VAR Model



Zooming in on the 90-day forecast for the hospitalization count from the VAR(15) model, the projected forecast does not closely follow the original data. The forecast line follows the original data early and then deviates significantly higher, resulting in an over forecast. The forecast then crosses under the original data and under forecasts. The forecast line is trending back upward following the original data in the end of the plot.

```

# Zoom In Percent Positive - 90day
startPoint = length(df_US_Final$Hospitalized_Count) - 89
endPoint = length(df_US_Final$Hospitalized_Count)

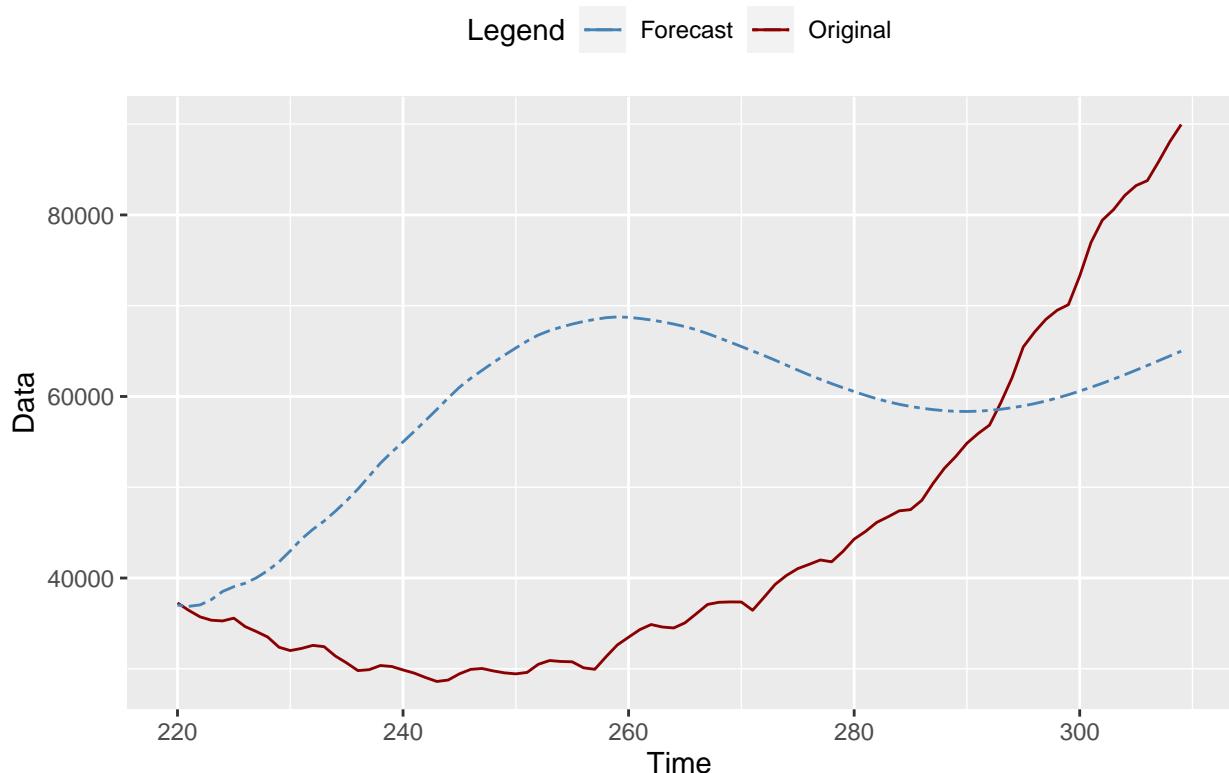
tl = seq(startPoint,endPoint)
orig = df_US_Final$Hospitalized_Count[startPoint:endPoint]
forecast = USVARPreds90Day$fcst$y2[,1]
df_US7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_US7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("US Percent Positive: Original vs. Forecast - 90 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)

```

US Percent Positive: Original vs. Forecast – 90 Day Horizon – VAR Mode



Forecasting out the full 7-day period, the following is shown. The VAR model still selects the VAR(15) for the full dataset. The predictions for the 7-day horizon still seem to trend well with the variables. As expected the confidence intervals continue to increase the further away from the original data plots the forecast moves.

```

# Full Forward 90-day
X = cbind(df_US_Final$Percent_Positive,df_US_Final$Hospitalized_Count)
names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut

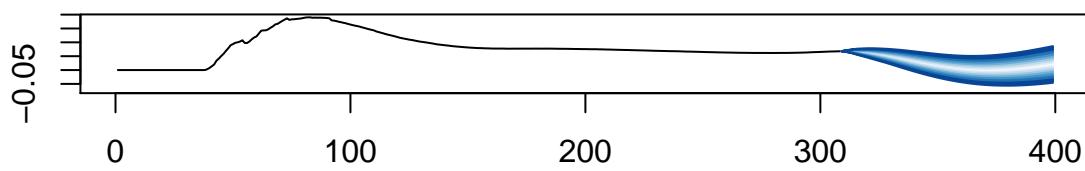
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      15     10      3    15
##
## $criteria
##          1       2       3       4       5       6       7
## AIC(n) 2.046835 1.025155 0.8975791 0.8922006 0.8498379 0.8464605 0.7339070
## HQ(n)  2.076941 1.075330 0.9678250 0.9825167 0.9602242 0.9769171 0.8844338
## SC(n)  2.122010 1.150447 1.0729877 1.1177259 1.1254799 1.1722193 1.1097825
## FPE(n) 7.743367 2.787545 2.4537001 2.4405877 2.3394311 2.3316495 2.0835733
##          8       9      10      11      12      13      14
## AIC(n) 0.6896901 0.6596453 0.6044475 0.5971893 0.616988 0.5838484 0.6050768
## HQ(n)  0.8602872 0.8503125 0.8151851 0.8279970 0.867866 0.8547966 0.8960953
## SC(n)  1.1156823 1.1357542 1.1306732 1.1735317 1.243447 1.2604242 1.3317694
## FPE(n) 1.9936127 1.9348041 1.8311331 1.8181691 1.854864 1.7947870 1.8337518
##          15
## AIC(n) 0.5153119
## HQ(n)  0.8264006
## SC(n)  1.2921212
## FPE(n) 1.6767971

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n")]],type="const")
preds=predict(lsfit,n.ahead=LONG_TERM_FORECAST_HORIZON)

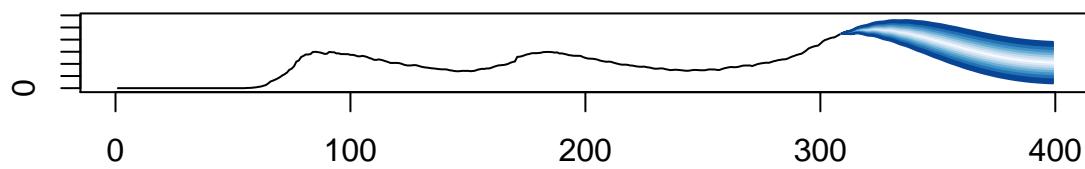
#Plot forecast predictions
fanchart(preds, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make distinguishable

```

Fanchart for variable y1



Fanchart for variable y2

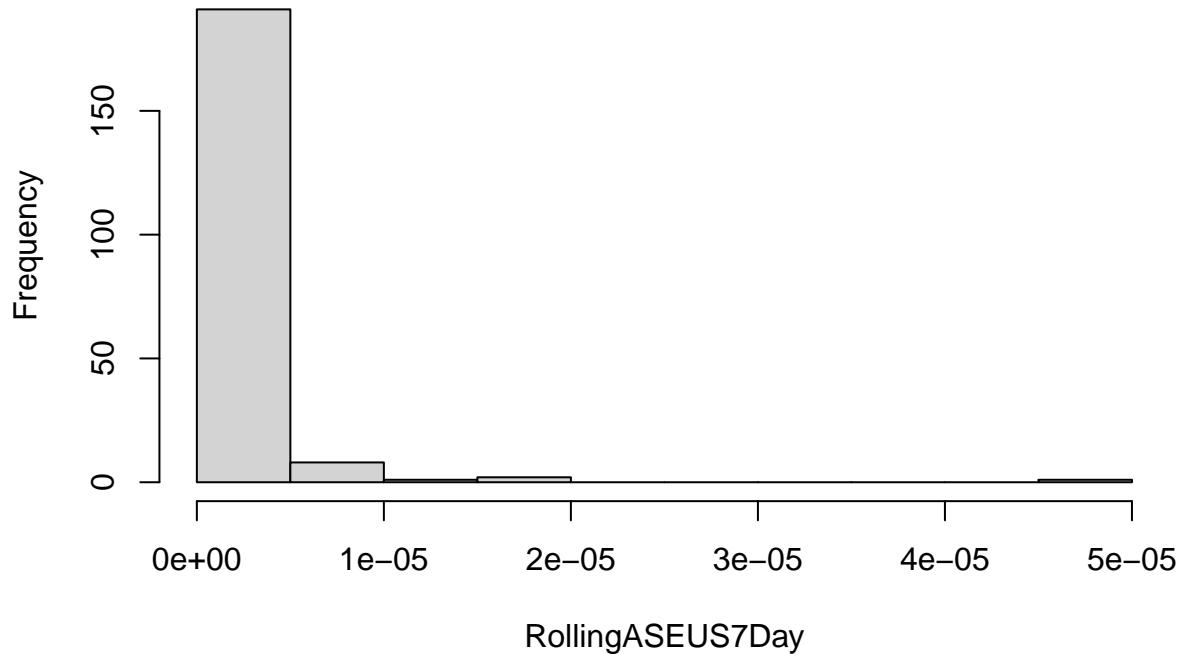


For comparison purposes calculate the rolling window ASEs for the VAR models.

```
RollingASEVARUS7Day = RollingASECalcVAR(df_US_Final$Percent_Positive,df_US_Final$Hospitalized_Count,pVa  
RollingASEVARUS90Day = RollingASECalcVAR(df_US_Final$Percent_Positive,df_US_Final$Hospitalized_Count,pV
```

```
GetRollingWindowASEInfo("7-Day VAR",RollingASEVARUS7Day)
```

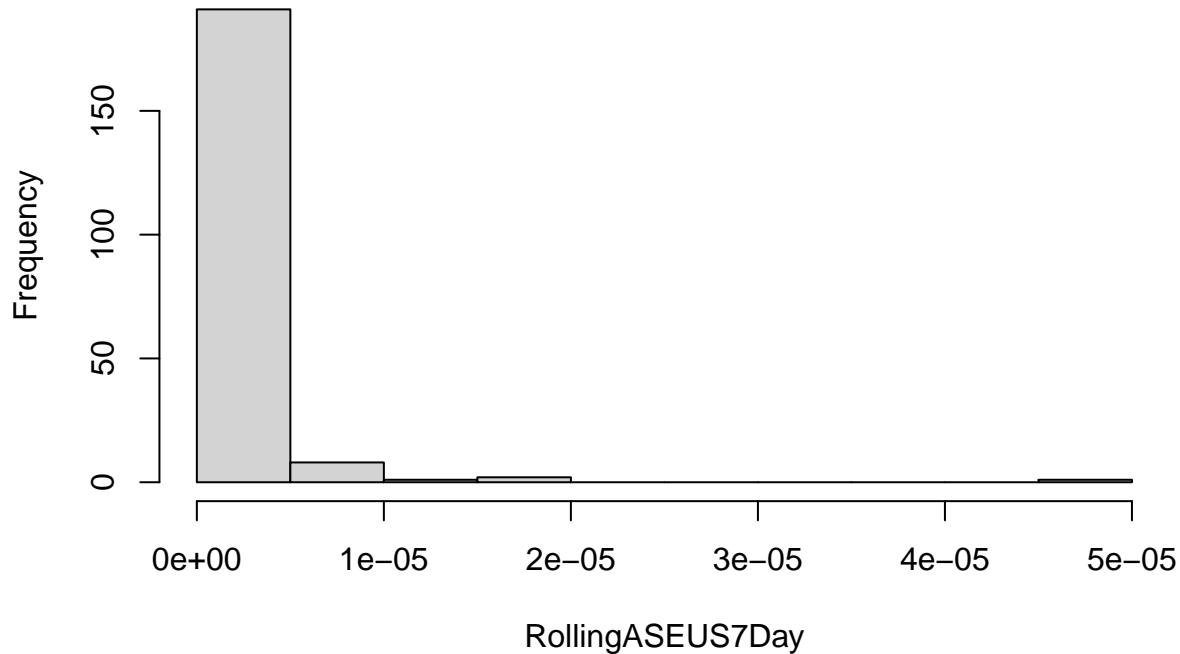
Histogram of 7-Day VAR Windowed ASE



```
## [1] "7-Day VAR Windowed ASE Mean: 0.000102732014044877"  
## [1] "7-Day VAR Windowed ASE Median: 5.70779788156812e-06"
```

```
GetRollingWindowASEInfo("90-Day VAR", RollingASEVARUS90Day)
```

Histogram of 90-Day VAR Windowed ASE



```
## [1] "90-Day VAR Windowed ASE Mean: 2.67343984602754e+35"
## [1] "90-Day VAR Windowed ASE Median: 0.00155424709971617"
```

The rolling window ASE plots for the VAR(15) model are:

- Rolling Window ASE - 7 Day: 0.00010
- Rolling Window ASE - 90 Day: 2.6734e+35

Multi-Layer Perceptron (MLP) Another time series modeling approach that can be taken is to leverage a neural network, also called a Multi-Layer Perceptron (MLP). Through this model an input layer is defined. Each input layer is connected to a “hidden layer”. The connection between the input layer and the “hidden layer” contains a weighting. The “hidden layer” is used to approximate any continuous function. Within the MLP model there can be one or more hidden layers. The “hidden layers” then all connect to an output layer. The output layer provides the final result of the model.

First the model will be evaluated. The model will be set to perform 50 repetitions of the MLP. Each of these repetition results will be combined based on the mean calculation. For this multivariate MLP model the, similar to the VAR model, the Hospitalization count will be added to the model.

Evaluating the model with the base data results in the following:

```
# FULL FORWARD FORECAST
endPoint90Day = length(df_US_Final$Hospitalized_Count) - LONG_TERM_FORECAST_HORIZON -1
endPoint7Day = length(df_US_Final$Hospitalized_Count) - SHORT_TERM_FORECAST_HORIZON-1
fullData = data.frame(hosp=ts(df_US_Final$Hospitalized_Count))
```

```

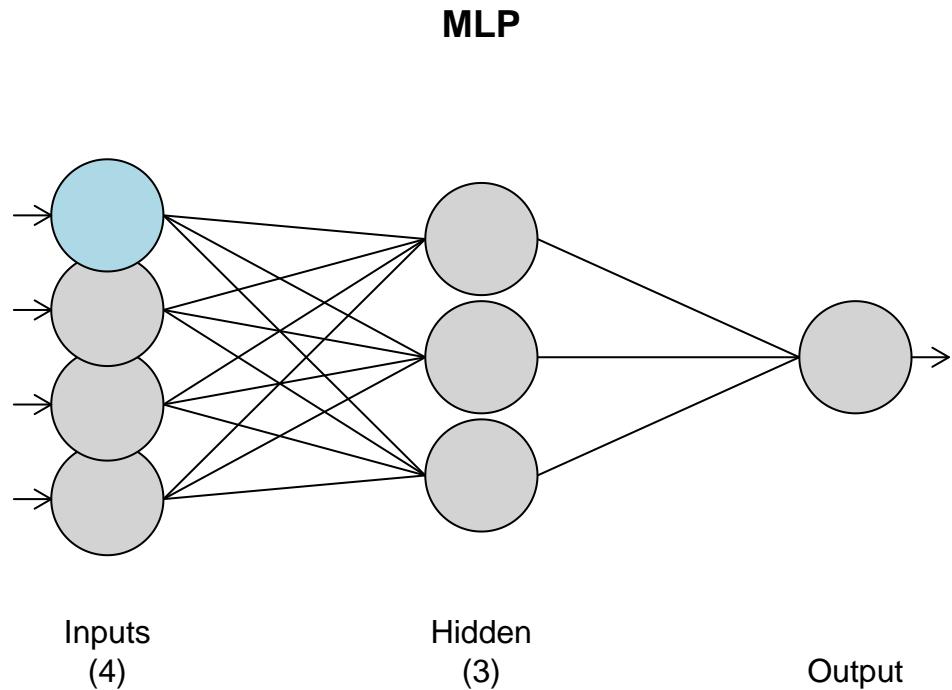
# Build for 7 Day Horizon
tl = ts(df_US_Final$Percent_Positive[1:endPoint7Day])
Sx = data.frame(hosp=ts(df_US_Final$Hospitalized_Count[1:endPoint7Day]))
fit.mlp.MVUS7 = mlp(tl,xreg=Sx,sel.lag = TRUE,reps=NN_REPS,hd.auto.type="cv")
fore.mlp.MVUS7 = forecast(fit.mlp.MVUS7, h = SHORT_TERM_FORECAST_HORIZON,xreg=fullData)

tl = ts(df_US_Final$Percent_Positive[1:endPoint90Day])
Sx = data.frame(hosp=ts(df_US_Final$Hospitalized_Count[1:endPoint90Day]))
fit.mlp.MVUS90 = mlp(tl,xreg=Sx,sel.lag = TRUE,reps=NN_REPS,hd.auto.type="cv")
fore.mlp.MVUS90 = forecast(fit.mlp.MVUS90, h = LONG_TERM_FORECAST_HORIZON,xreg=fullData)

```

The visual below represents the MLP configuration used for the short-term (7-day) forecast. The model contains four (4) input layers and one (1) hidden layer.

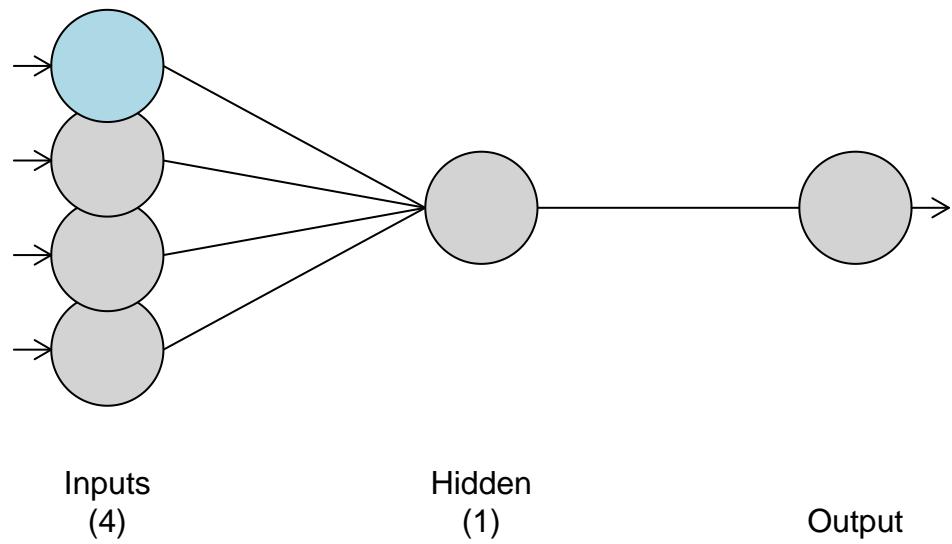
```
plot(fit.mlp.MVUS7)
```



The visual below represents the MLP configuration used for the long-term (90-day) forecast. The model contains four (4) input layers and three (3) hidden layer.

```
plot(fit.mlp.MVUS90)
```

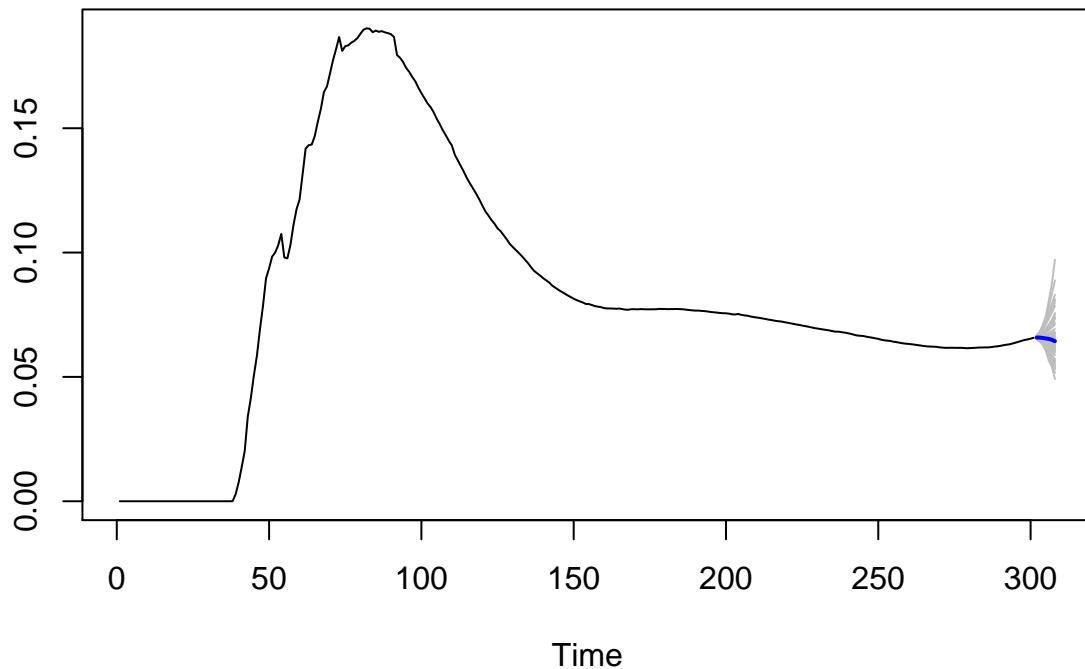
MLP



The plot below shows the 7-day forecasts that were calculated from the MLP model. From the visual it seems that all of the projections for the short-term forecast were trending in the same direction. The blue color on the plot represents the mean of the projections. It does seem that this forecast begins a downward trend.

```
plot(fore.mlp.MVUS7)
```

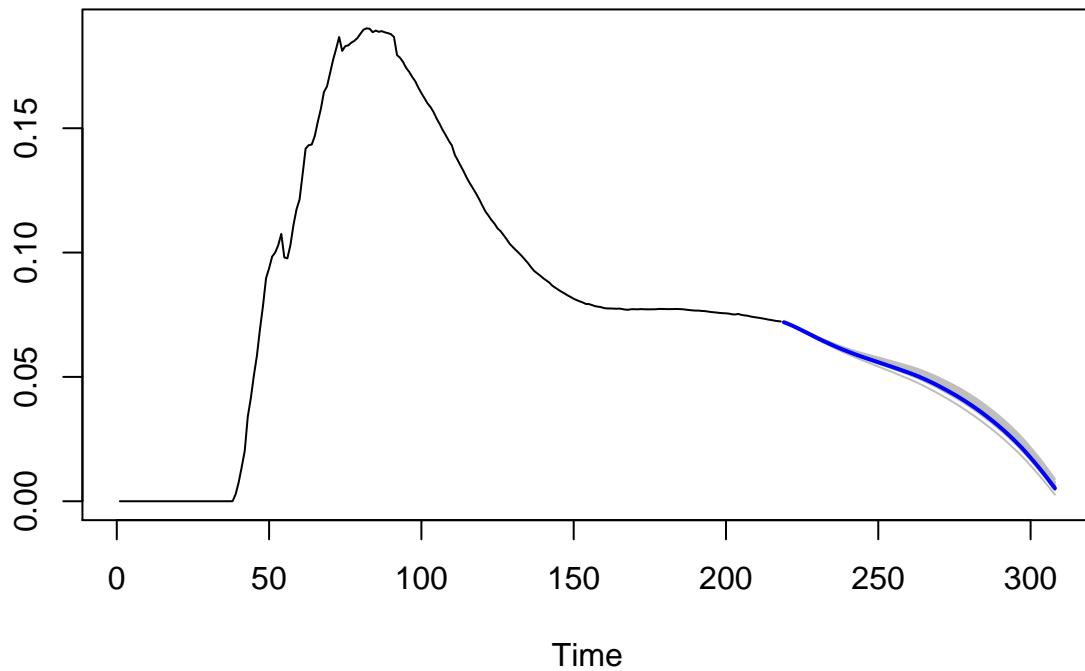
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to deviate downward sharply.

```
plot(fore.mlp.MVUS90)
```

Forecasts from MLP

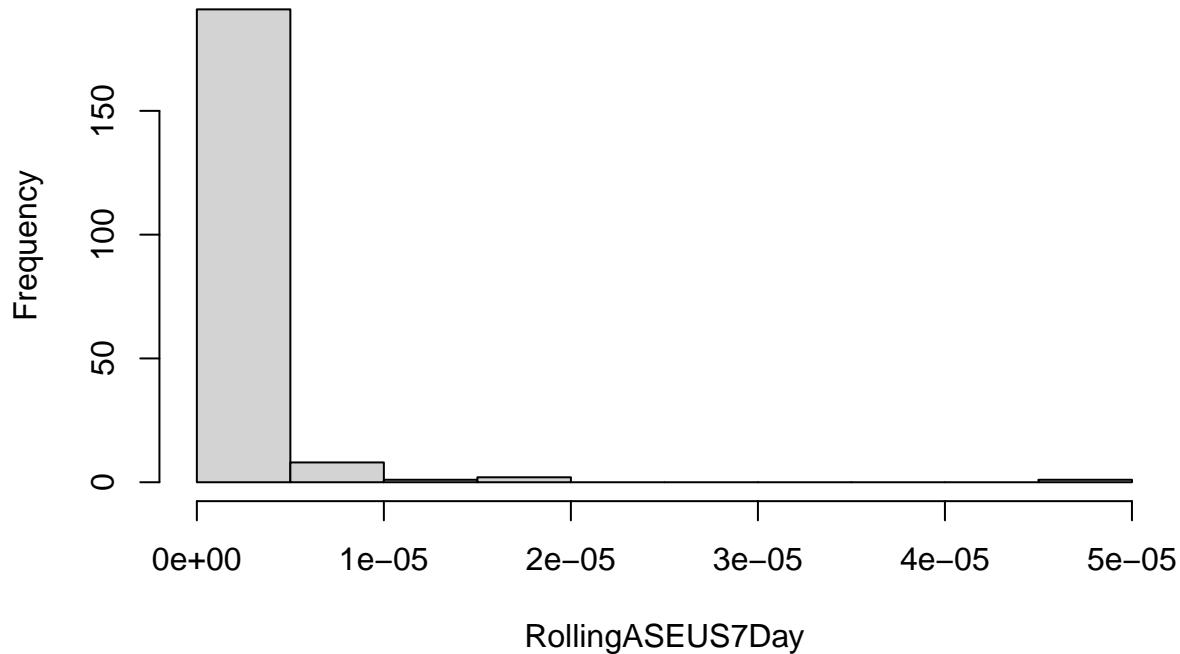


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcMVMLP7Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT_7_DAY)
RollingASECalcMVMLP90Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG_90_DAY)
```

```
modelName = "US Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcMVMLP7Day)
```

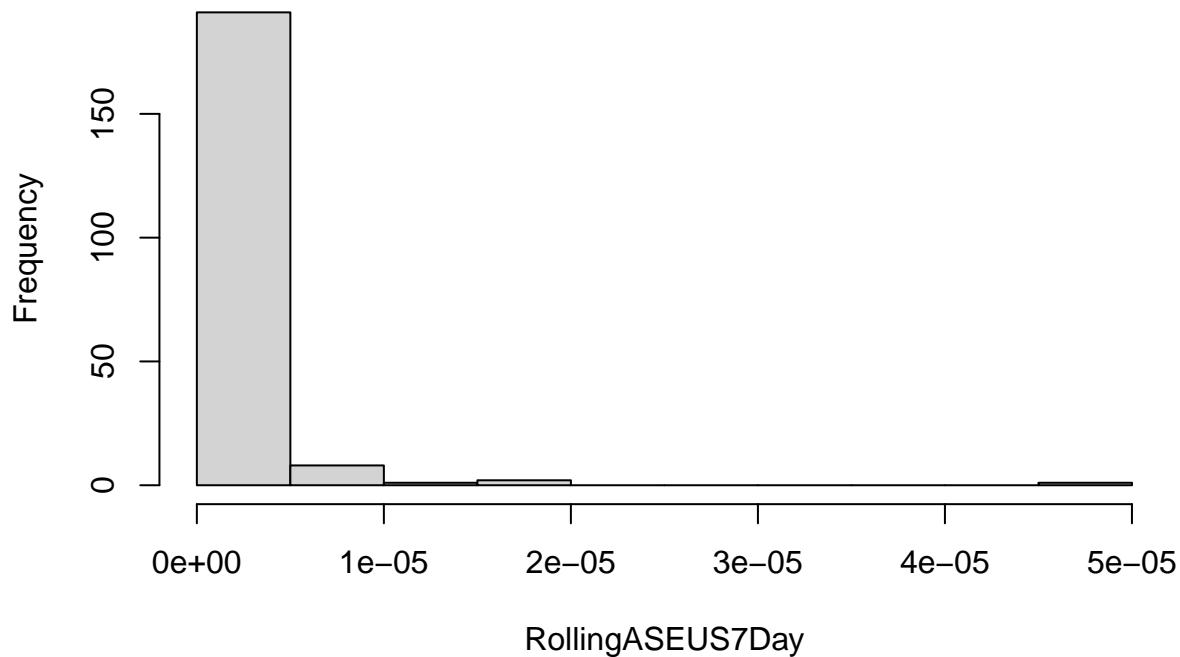
Histogram of US Model – 7 Days Windowed ASE



```
## [1] "US Model - 7 Days Windowed ASE Mean: 2.01956844442391e-06"  
## [1] "US Model - 7 Days Windowed ASE Median: 1.14642944922895e-07"
```

```
modelName = "US Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcMVMLP90Day)
```

Histogram of US Model – 90 Days Windowed ASE



```
## [1] "US Model - 90 Days Windowed ASE Mean: 0.000530849683581231"  
## [1] "US Model - 90 Days Windowed ASE Median: 5.60126383351628e-05"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 2.019568e-06
- Rolling Window ASE - 90 Day: 0.000530

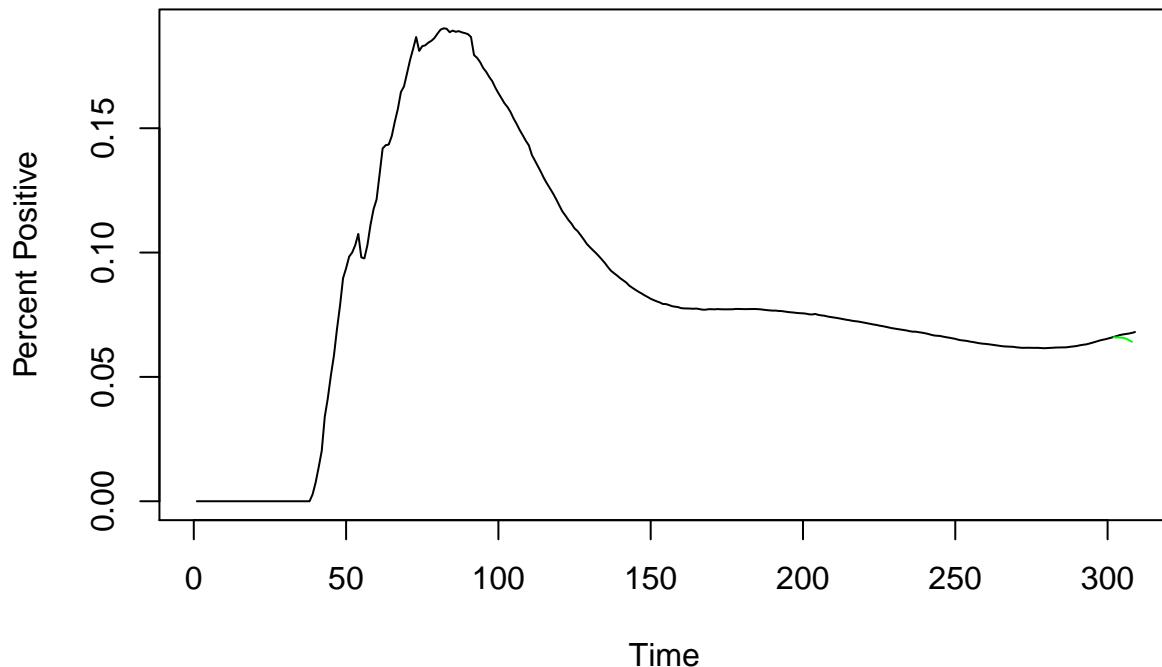
In comparison to the VAR(15) model that was developed these ASE values are much lower for the MLP model.

Multivariate Ensemble Modeling Ensemble modeling is a way of combining models to see if the positives of both models will make the forecasts better. As was done in the univariate case, models used in the multivariate analysis will be combined to see if there is better performance. Ensemble models will be executed for both the short-term and long-term forecasts.

Start with the short term forecast. The ensemble projection, shown in green, seems to deviate downward versus the original realization trending upward.

```
startPos = length(df_US_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON  
endPos = length(df_US_Final$Percent_Positive)-1  
  
USMV7Ensemble = (USVARPreds7Day$fcst$y1[,1] + fore.mlp.MVUS7$mean)/2  
  
plot(seq(1,length(df_US_Final$Percent_Positive),1),df_US_Final$Percent_Positive, type = "l",xlim = c(1,  
lines(seq(startPos,endPos,1), USMV7Ensemble, type = "l", col = "green")
```

Total US – Percent Positive – 7 Day Forecast – Ensemble Model



```
USMV7EnsembleASE = mean((df_US_Final$Percent_Positive[startPos:endPos] - USMV7Ensemble)^2)
```

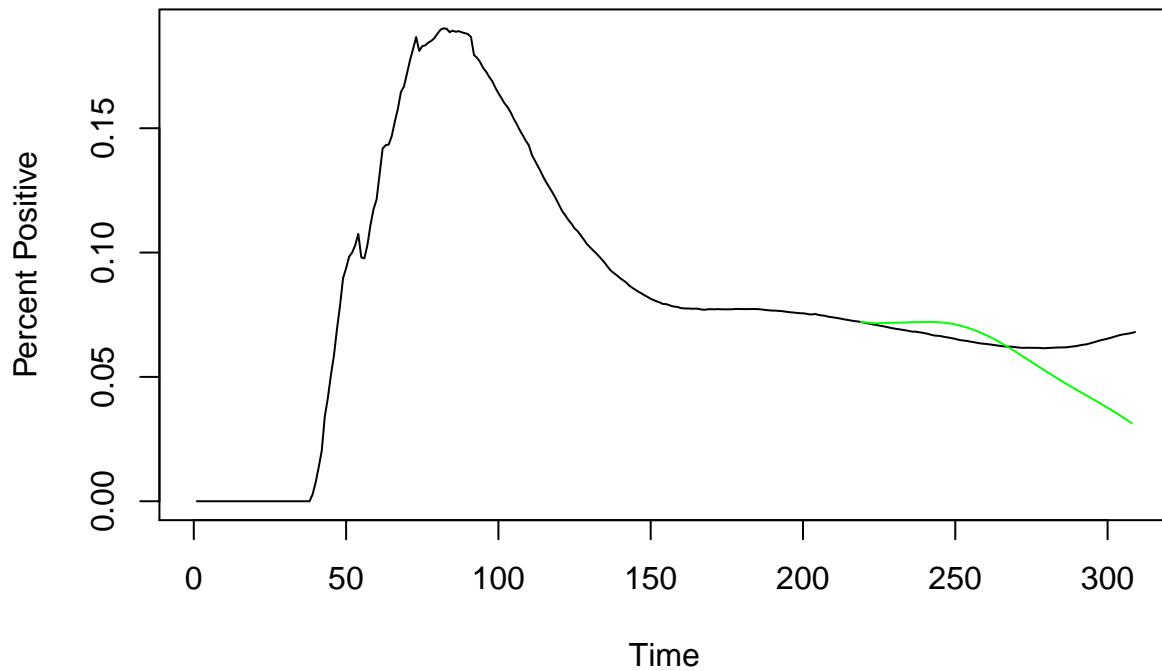
Now move to the 90 day forecast. The long-term forecast for the ensemble does not track well to the data. It initially over forecasts the original values and then takes a sharp downward trend as it progresses further.

```
startPos = length(df_US_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON  
endPos = length(df_US_Final$Percent_Positive)-1
```

```
USMV90Ensemble = (USVARPreds90Day$fcst$y1[,1] + fore.mlp.MVUS90$mean)/2
```

```
plot(seq(1,length(df_US_Final$Percent_Positive),1),df_US_Final$Percent_Positive, type = "l",xlim = c(1,1),  
lines(seq(startPos,endPos,1), USMV90Ensemble, type = "l", col = "green")
```

Total US – Percent Positive – 90 Day Forecast – Ensemble Model



```
USMV90EnsembleASE = mean((df_US_Final$Percent_Positive[startPos:endPos] - USMV90Ensemble)^2)
```

ASEs for the ensemble models are:

```
print(paste("ASE Ensemble 7-Day: ", USMV7EnsembleASE))
```

```
## [1] "ASE Ensemble 7-Day: 3.79753987024044e-06"
```

```
print(paste("ASE Ensemble 90-Day: ", USMV90EnsembleASE))
```

```
## [1] "ASE Ensemble 90-Day: 0.000191071652295514"
```

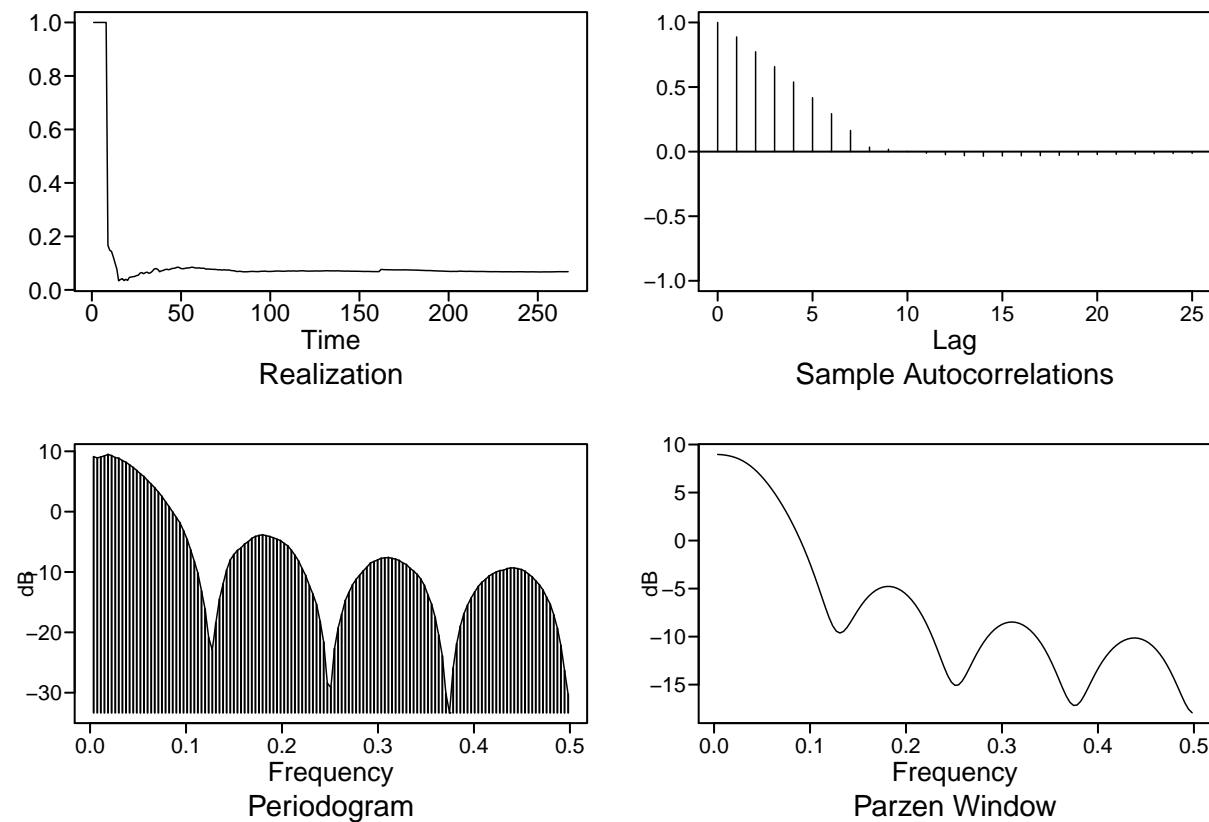
US Multivariate Modeling Summary In summary for US multivariate modeling, the following models were developed with ASEs for each of the forecast periods. As shown, the MLP models seemed to perform the best on both forecasts. Individually the MLP model performed best on the 7-day horizon and the Ensemble performed slightly better than the MLP model on the 90-day horizon.

Method	ASE - 7 Day	ASE - 90 Day
VAR(15)	0.00010	2.6734e+35
MLP	2.019568e-06	0.000530
Ensemble	1.135600e-05	0.000512

North Carolina Analysis

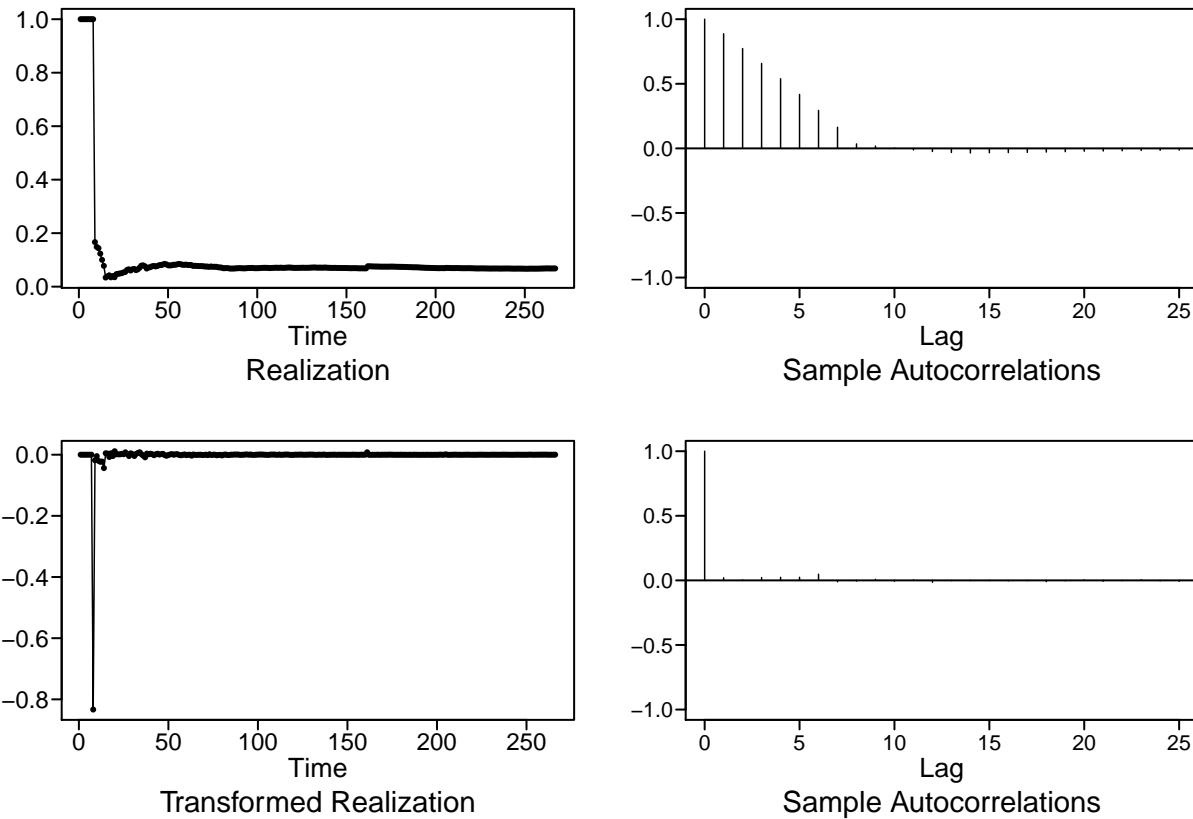
Moving on to an analysis of the North Carolina (NC) specific data, a very similar modeling approach that was taken for the total US will be taken for North Carolina. The initial time series model that will be created for North Carolina (NC) will be an ARIMA-type model. These models are relatively easy to build and provide good insight into the data characteristics. The first step in the ARIMA modeling process is to simply plot the realization of the data along with ACF and the spectral density. The ACF plot will provide an indication of model characteristics that may need to be taken into account. The spectral density plot will also provide an indication of model characteristics and also allows for confirming seasonality within the data, if there is any.

```
# Look at percent positive in NC
NCPPlotTSOut = plotts.sample.wge(df_NC_Final$Percent_Positive)
```



From the realization (top-left plot), there is a large peak at the initial outset and then the data drops off drastically over time. This is due to the data reporting for percent positive. For the first eight reports the the percent positive was 100% as both the number of positive tests and total tests were the same. This could be a reporting error however it could also be real data. For analysis purposes, it will be assumed this is real data. The realization does not indicate any wandering behavior, where the data is increasing or decreasing over time, nor do is there any indication of cyclic behavior. Moving to the ACF plot (top-right plot), the data looks to be quickly damping. The spectral density plot (bottom-right plot) indicates a large peak at zero and then three other smaller peaks. It's difficult to tell from these indicators if the data is stationary or not. As with the US data a differencing approach will be undertaken.

```
dfNCCasesDiff = df_NC_Final$Percent_Positive
dfNCCasesDiff = artrans.wge(dfNCCasesDiff, phi.tr=1)
```



After differencing the data one time, the realization of the differenced data (bottom-left plot) is starting to look a bit more stationary. In addition, the differenced ACF plot (bottom-right plot) indicates an immediate drop off. From here it will be assumed the data is stationary. Leveraging the differenced data, the AR(p) and MA(q) component values can be determined. In order to determine the best fit for these components both the AIC and BIC metrics will be used. The AR(p) component will be searched from zero (0) to fifteen (15) and the MA(q) component will be searched from zero (0) to fifteen (15).

```
#Get estimates for AIC
aic5.wge(dfNCCasesDiff,p=0:15,q=0:5)
```

```
## -----WORKING... PLEASE WAIT...
##
##
## Error in aic calculation at 4 4
## Error in aic calculation at 4 5
## Error in aic calculation at 6 3
## Error in aic calculation at 7 3
## Error in aic calculation at 7 4
## Error in aic calculation at 14 3
## Five Smallest Values of aic

##      p      q      aic
## 57     9      2   -6.038530
## 56     9      1   -6.005433
## 68    11      1   -5.997533
## 73    12      0   -5.975458
## 79    13      0   -5.971445
```

```

#Get estimates for BIC
aic5.wge(dfNCCasesDiff,p=0:15,q=0:5,type="bic")

## -----WORKING... PLEASE WAIT...
##
##
## Error in aic calculation at 4 4
## Error in aic calculation at 4 5
## Error in aic calculation at 6 3
## Error in aic calculation at 7 3
## Error in aic calculation at 7 4
## Error in aic calculation at 14 3
## Five Smallest Values of bic

##      p     q      bic
## 1    0     0 -5.925356
## 7    1     0 -5.904759
## 2    0     1 -5.904757
## 13   2     0 -5.883777
## 3    0     2 -5.883772

```

From the AR(p) and MA(q) selection process above, there is no commonality between the two selection criteria of AIC and BIC. Given this the the p=9, q=2 model from the AIC selection will be the chosen model to start as it had the lowest AIC.

```

NCSelPVal = 9
NCSelQVal = 2
NC_Diffs = 1

```

Now that the p and q values are defined for the data the actual coefficients of the ARIMA model will be estimated. These coefficients will assist in providing a true understanding of the model.

```

NC.est = est.arma.wge(dfNCCasesDiff,p=NCSelPVal,q=NCSelQVal)

##
## Coefficients of Original polynomial:
## -0.8219 0.1223 0.0326 0.0493 0.0478 0.0654 0.0285 -0.2875 -0.1141
##
## Factor          Roots          Abs Recip   System Freq
## 1+1.8375B+0.8899B^2 -1.0325+-0.2403i 0.9433 0.4636
## 1-0.6138B+0.6971B^2 0.4403+-1.1139i 0.8349 0.1901
## 1+0.7428B+0.6910B^2 -0.5375+-1.0763i 0.8312 0.3237
## 1-1.5385B+0.6756B^2 1.1387+-0.4286i 0.8219 0.0573
## 1+0.3939B           -2.5384            0.3939 0.5000
##
##
## White Noise Variance: ",NC.est$avar)

## [1] "White Noise Variance: 0.00217929023223711"

```

```
paste("Data Mean: ", mean(df_NC_Final$Percent_Positive))
```

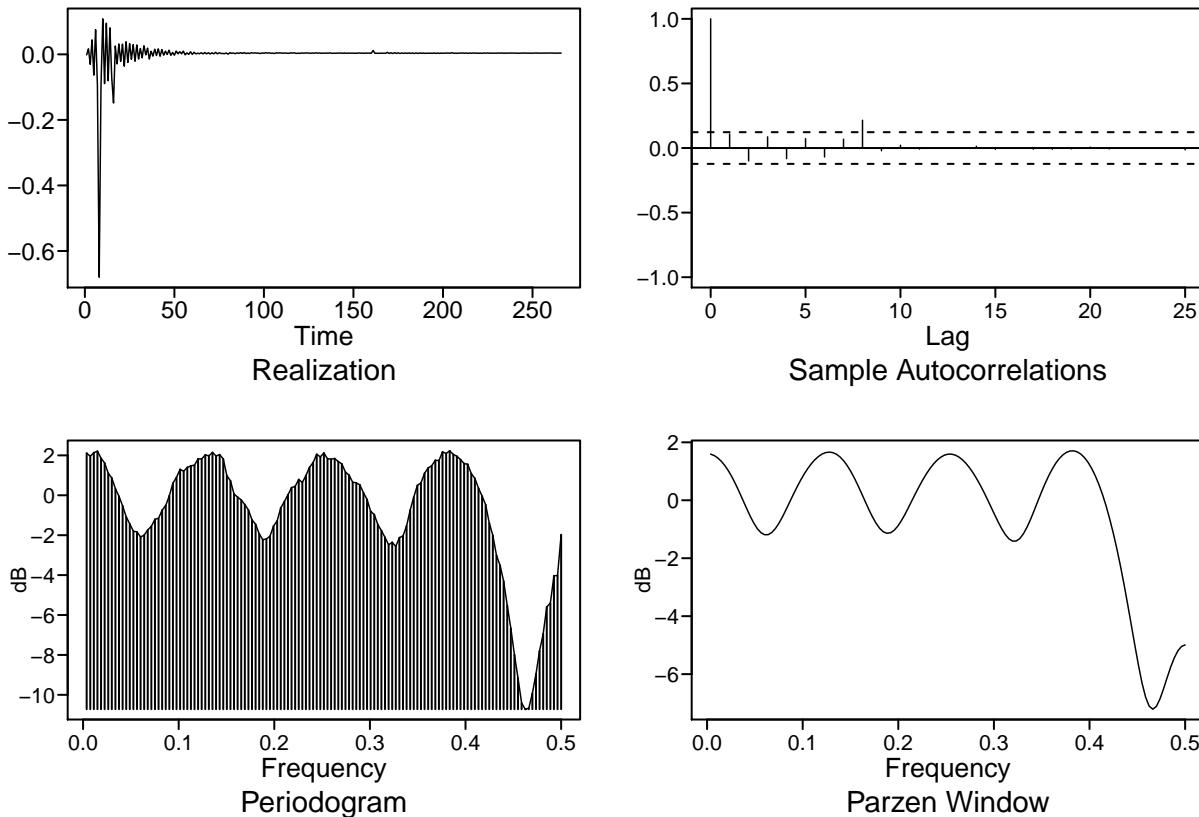
```
## [1] "Data Mean: 0.0991740237463197"
```

From the output above the coefficients of the model are displayed as well as the factor table of the model. The factor table indicates the influence each individual factor has on the model and whether the root of that factor is real or complex. From the factor table, there are four (4) complex roots. Of the five (5) total factors the first factor has high influence as they are close to 1.

Continuing the model development, the residuals from the model parameter estimates need to be evaluated. These will be evaluated in two ways: visually and through the Ljung-Box test. Each of these methods will help to determine if the residuals in the model are white noise are not. Modeling down to white noise would indicate that there is minimal to no “information” left in the remaining data and thus the model would contain everything. However, even with the residuals not being white noise the model can still be leveraged.

In evaluating the residuals for white noise the first step will be a visual inspection of the residual ACF plot. The residual ACF plot, shown below (top-right), indicates that there is only white noise left in the model. Only one of the autocorrelations is outside the limits which is roughly expected at the shown timeframe.

```
#Test the residuals for white noise using visual test  
NCResPlots = plotts.sample.wge(NC.est$res, arlimits=TRUE)
```



The Ljung-Box test is another test that can be leveraged to test for white noise in the residuals. The Ljung-Box test evaluates the autocorrelations as a group with the null hypothesis indicating that all the autocorrelations together equal zero. In order to determine if the null hypothesis is accepted or rejected, p-values from the Ljung-Box test will be evaluated. Each p-value will be tested against an alpha of 0.05.

This would indicate a 95% confidence level. It is best practice to execute the Ljung-Box test twice with differing values of the K parameter. In this case K will be set to 24 and 48 respectively.

```
#Test the residuals for white noise using Ljung-Box test
ljung.wge(NC.est$res,p=NCSelPVal,q=NCSelQVal)
```

```
## Obs 0.1087982 -0.09988667 0.08631986 -0.08191231 0.07228247 -0.06897433 0.06833139 0.21493 -0.021298
```

```
## $test
## [1] "Ljung-Box test"
##
## $K
## [1] 24
##
## $chi.square
## [1] 26.94256
##
## $df
## [1] 13
##
## $pval
## [1] 0.01266834
```

```
ljung.wge(NC.est$res,p=NCSelPVal,q=NCSelQVal,K=48)
```

```
## Obs 0.1087982 -0.09988667 0.08631986 -0.08191231 0.07228247 -0.06897433 0.06833139 0.21493 -0.021298
```

```
## $test
## [1] "Ljung-Box test"
##
## $K
## [1] 48
##
## $chi.square
## [1] 27.15478
##
## $df
## [1] 37
##
## $pval
## [1] 0.8821793
```

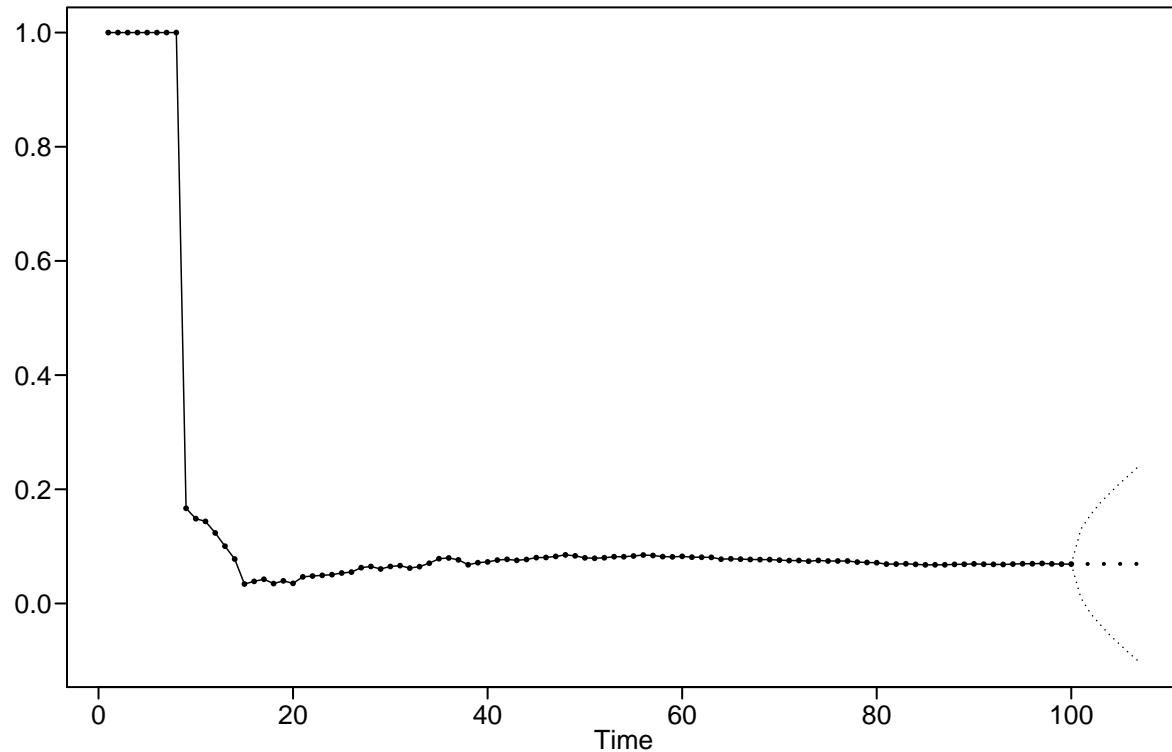
The results of the Ljung-Box test executions confirm that for K=24 the p-value is 0.01266 which at alpha = 0.05 would indicate a failure to reject the null hypothesis. At K=48, the p-value is 0.8821 which at alpha=0.05 would indicate a failure to reject the null hypothesis. From these results the model residuals seem to be white noise

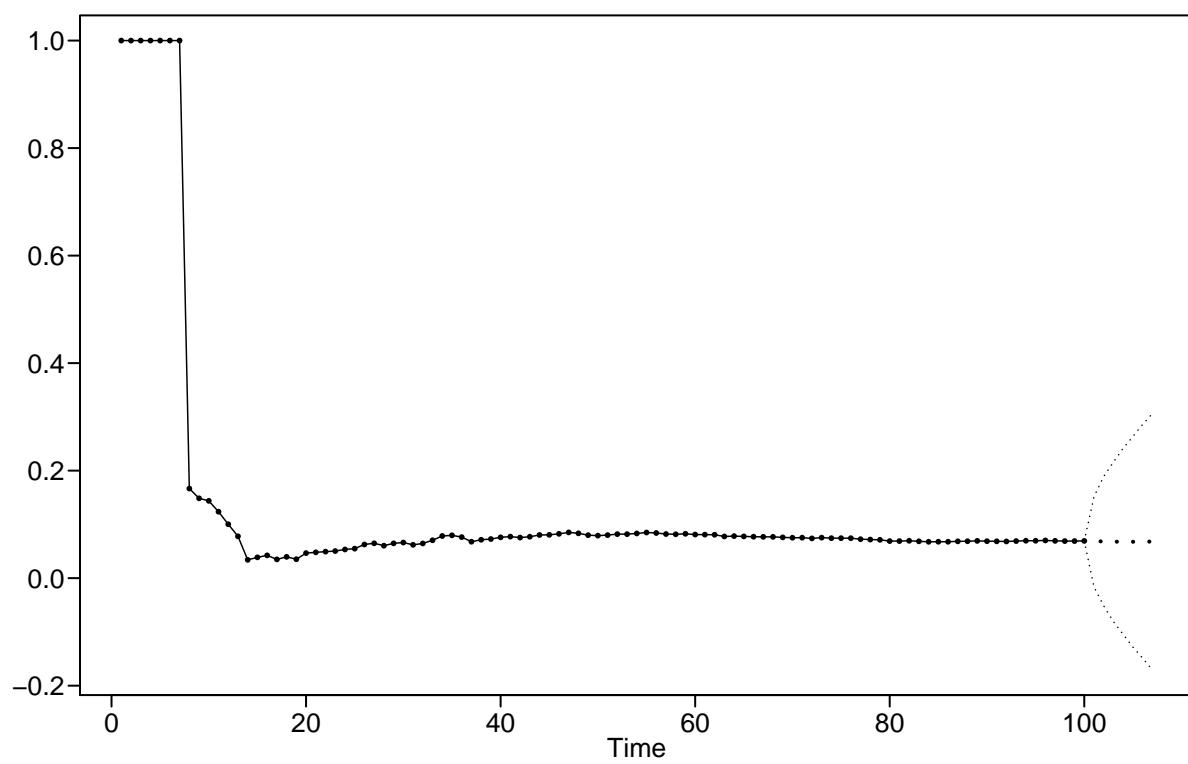
Thus the final model that will be leveraged will be an ARIMA(9,1,2).

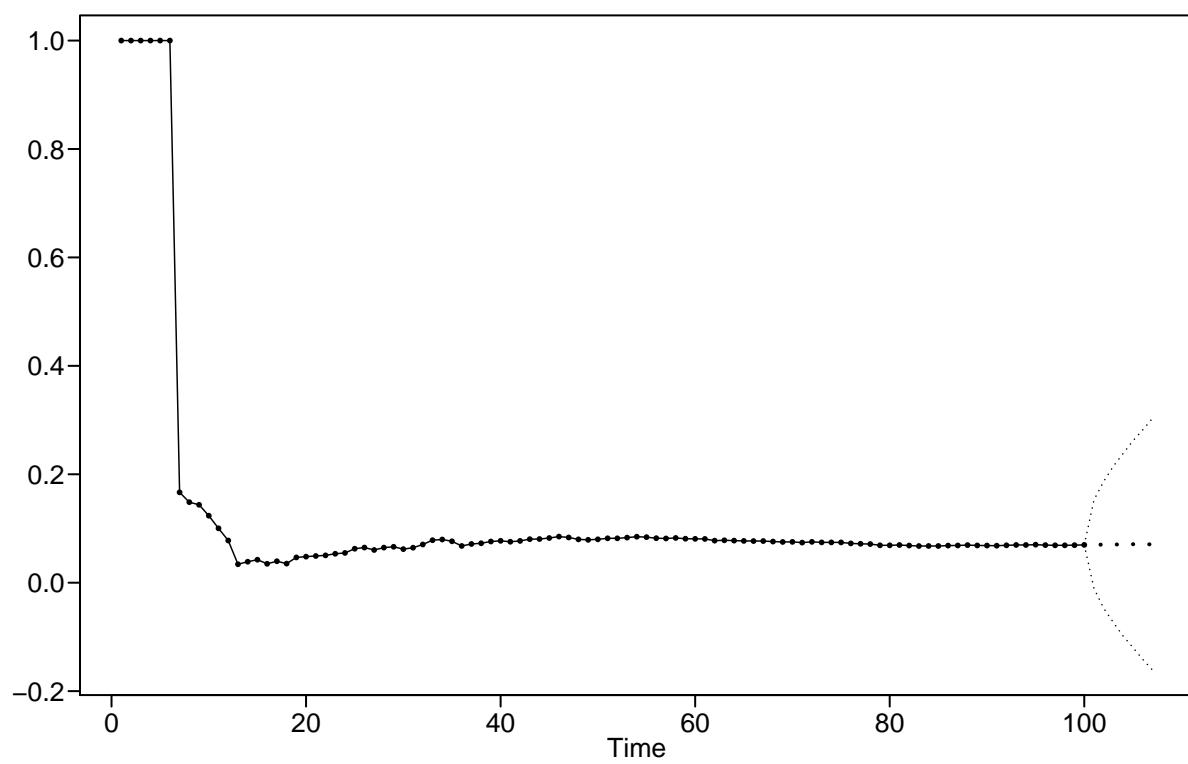
In order to evaluate this model, the data will be forecasted for a short term horizon of 7 days and a long term horizon of 90 days (3 months), The Rolling Window ASE will be leveraged as the evaluation metric for both of these horizons.

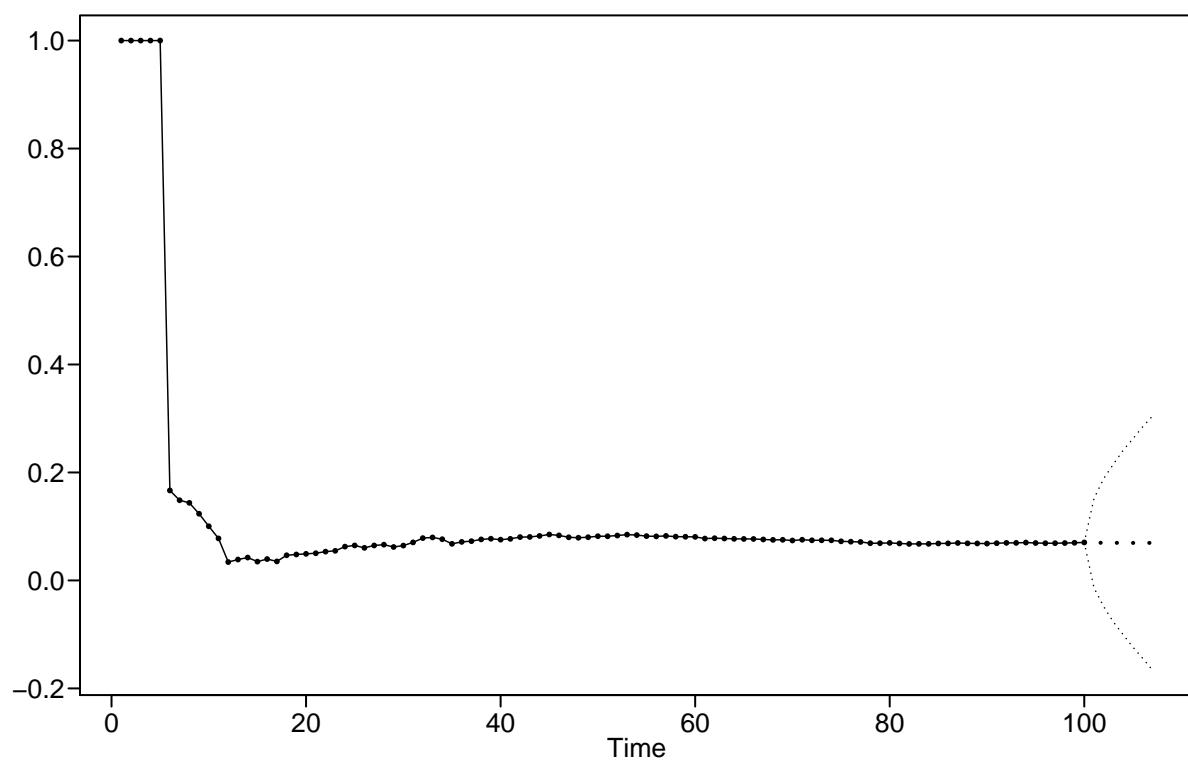
```
#Rolling Window ASE Calc - 7 Days
```

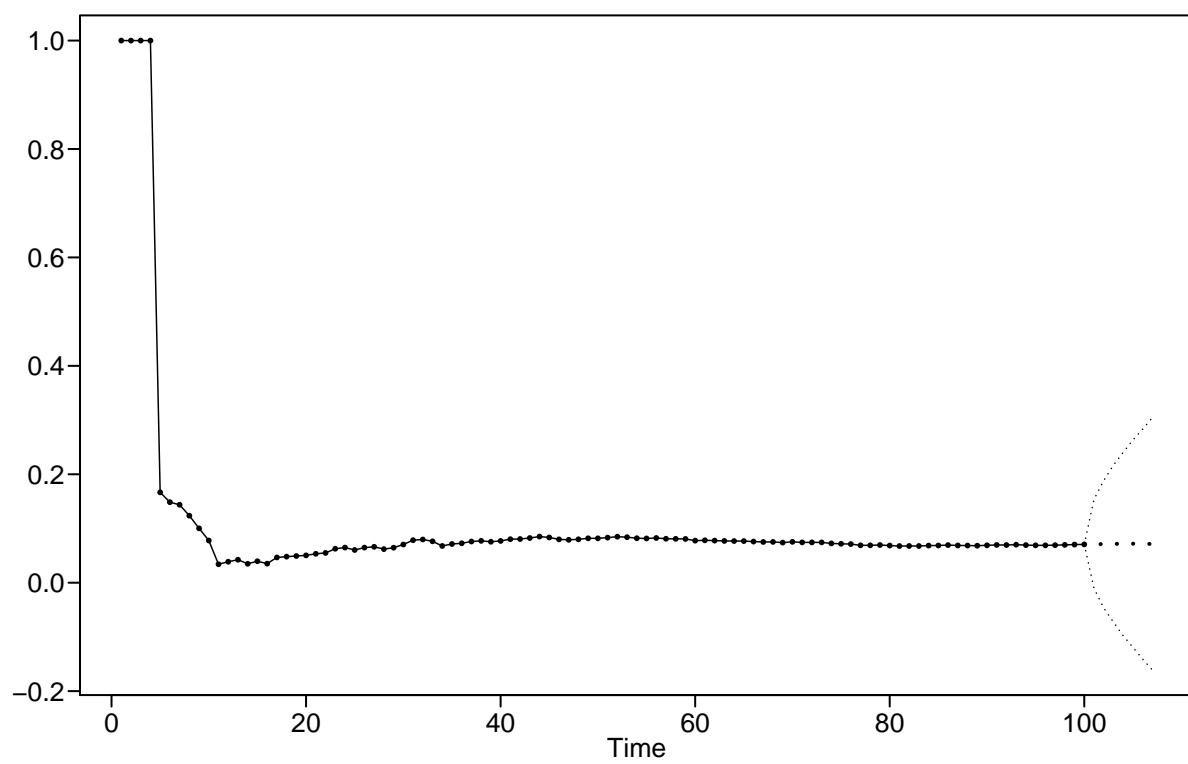
```
RollingASENC7Day = RollingASECalc(df_NC_Final$Percent_Positive,phis=NC.est$phi, thetas = NC.est$theta, s
```

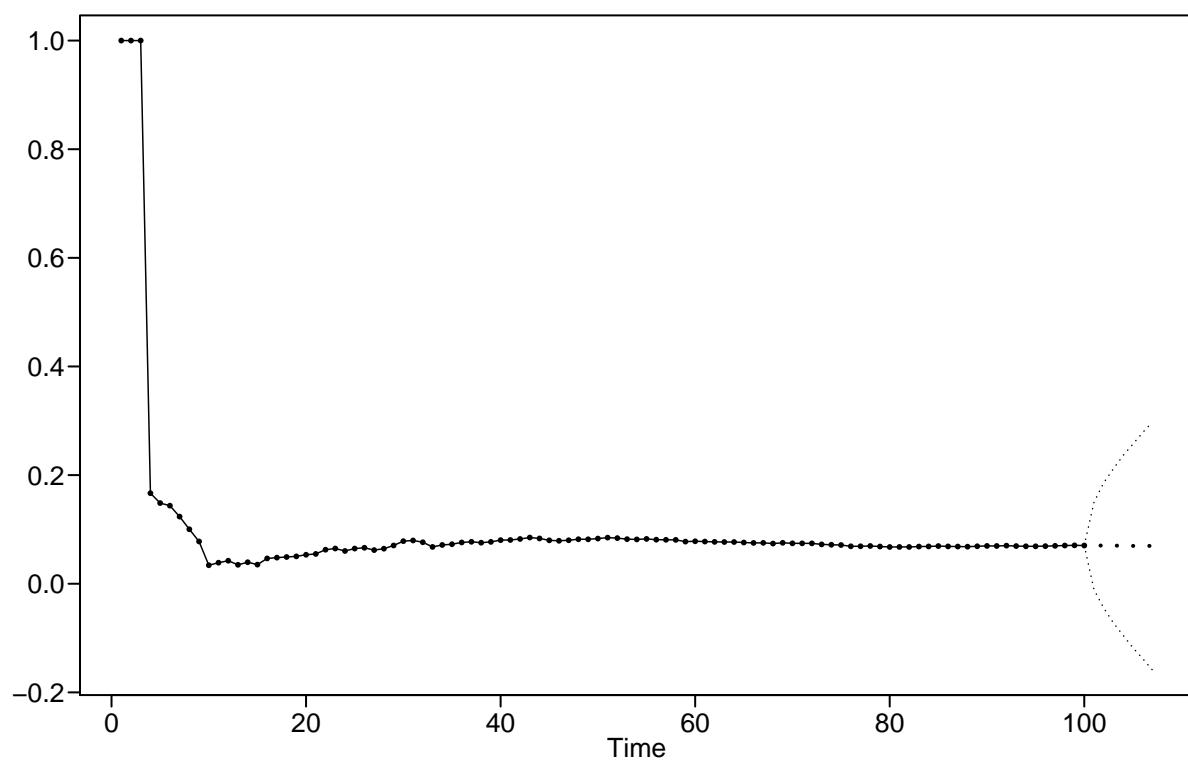


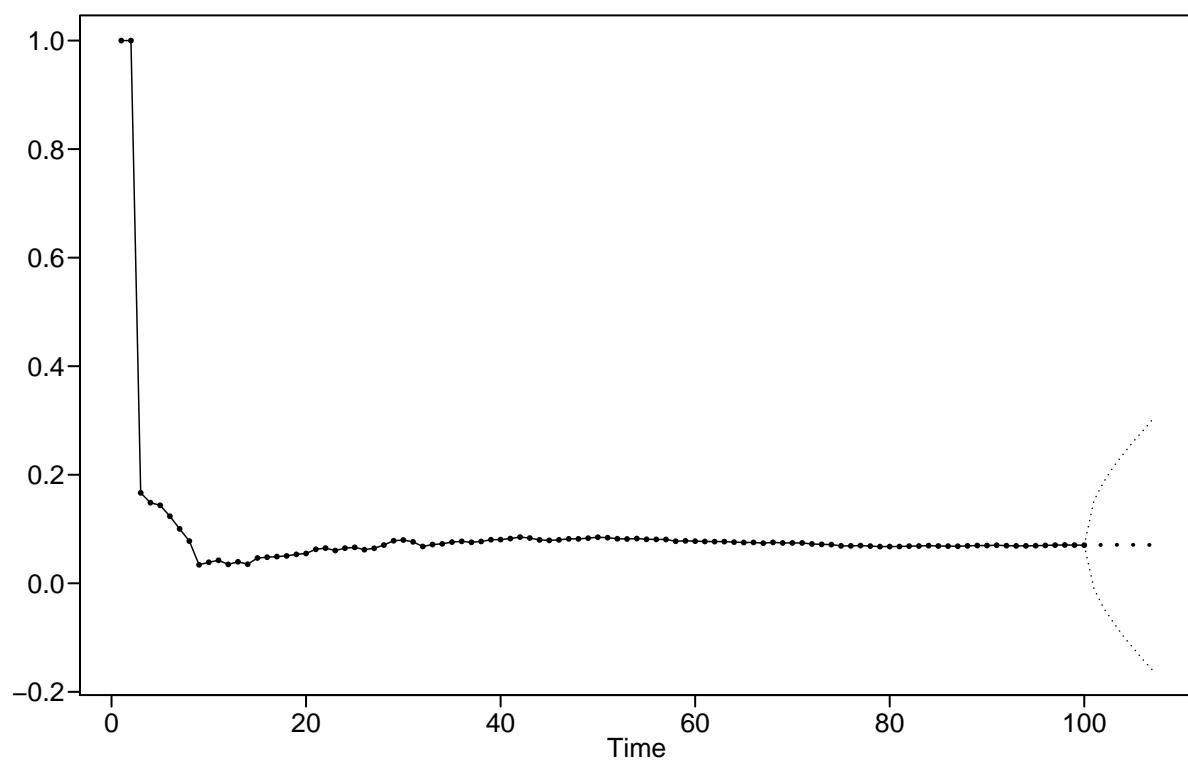


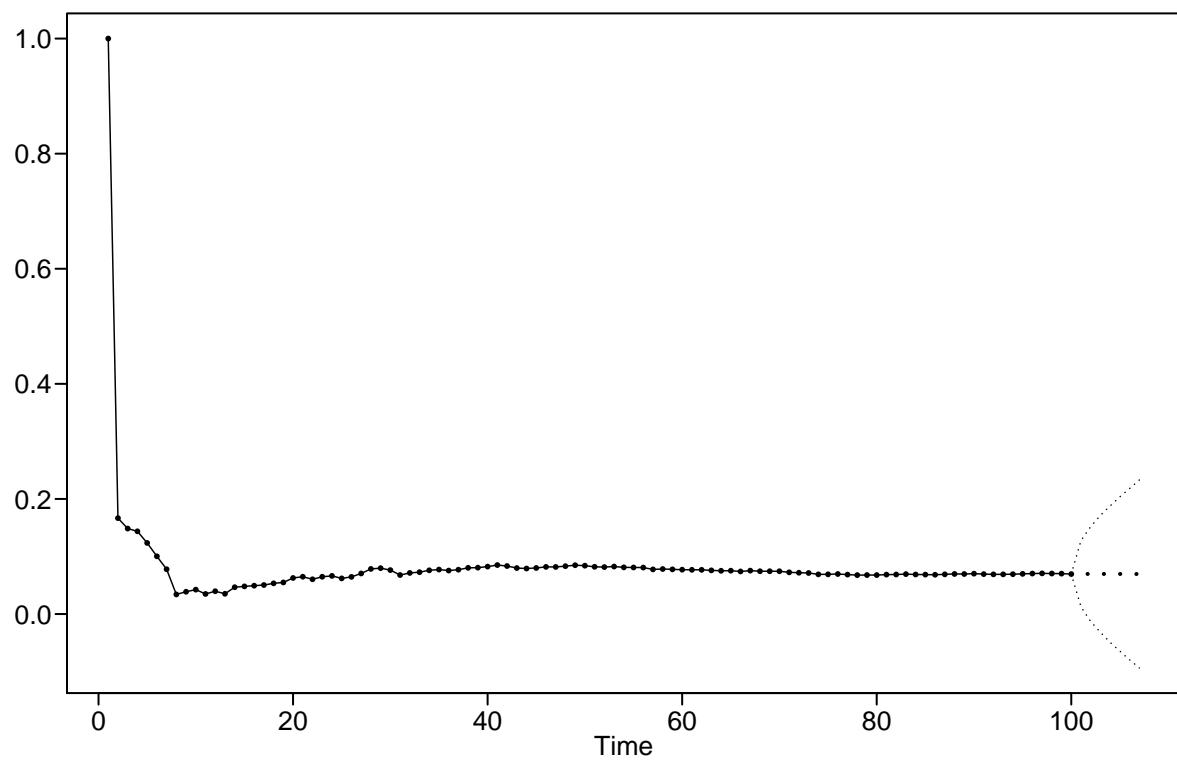


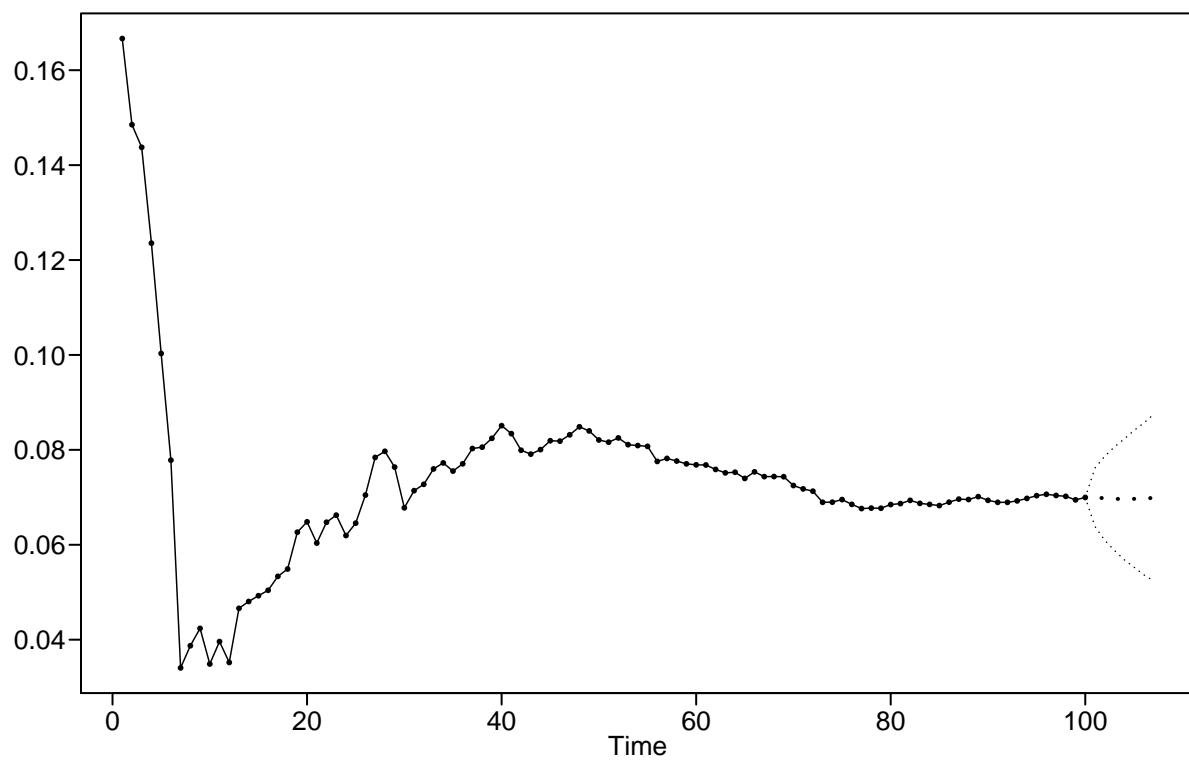


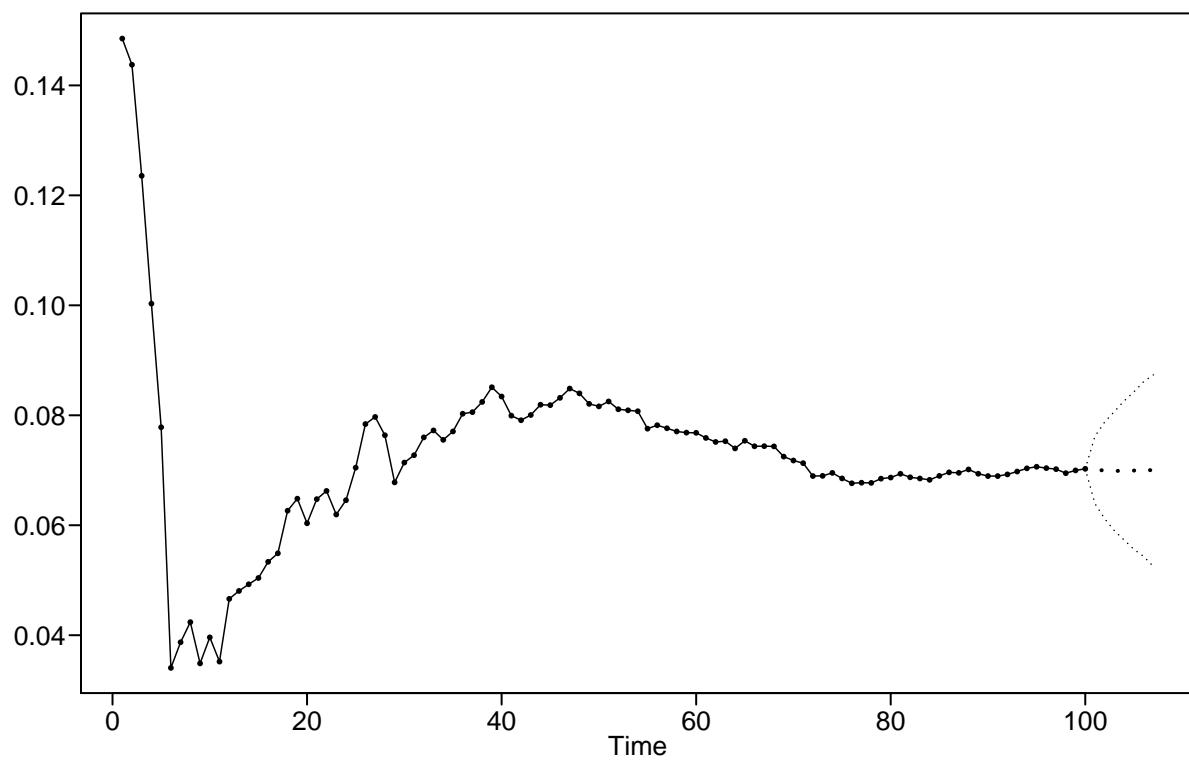


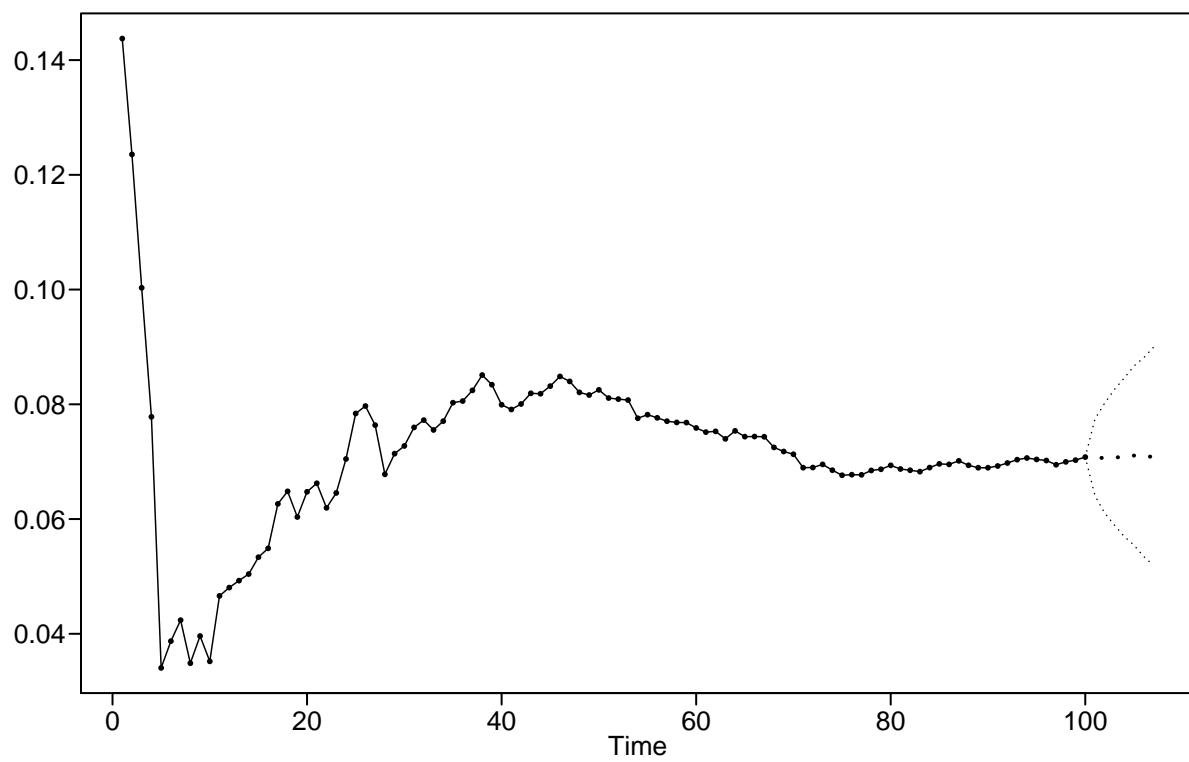


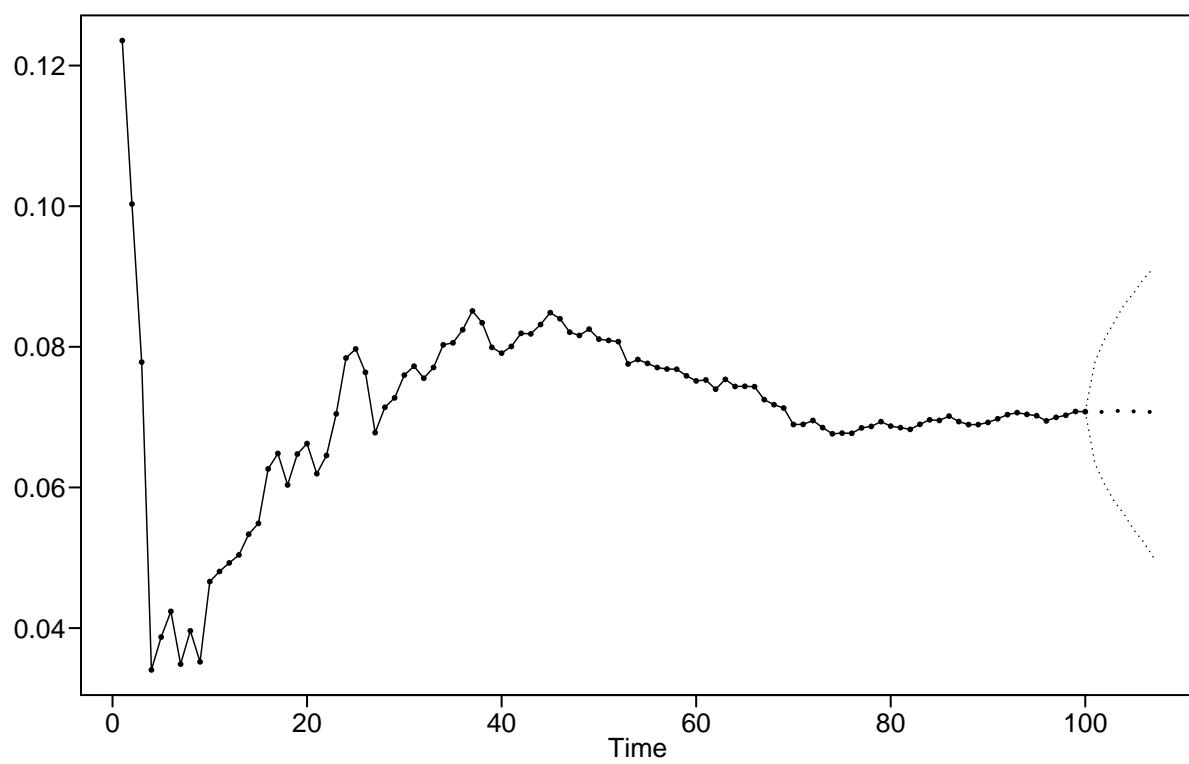


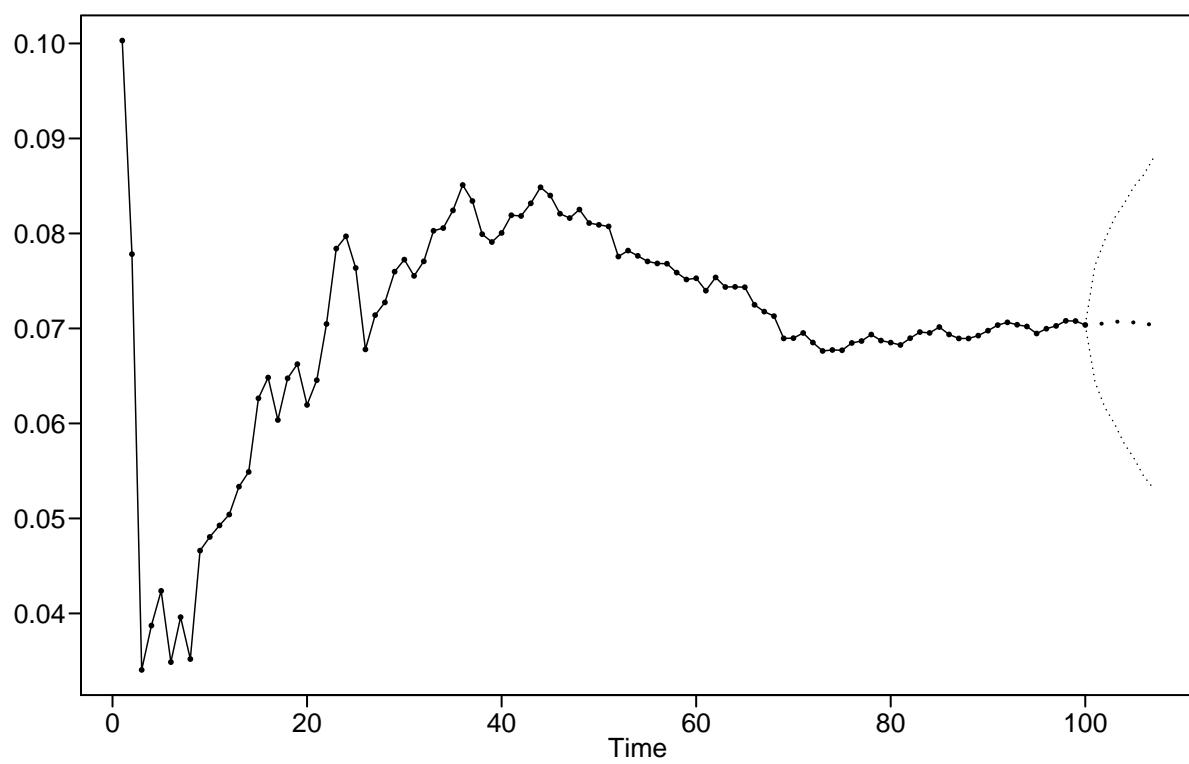


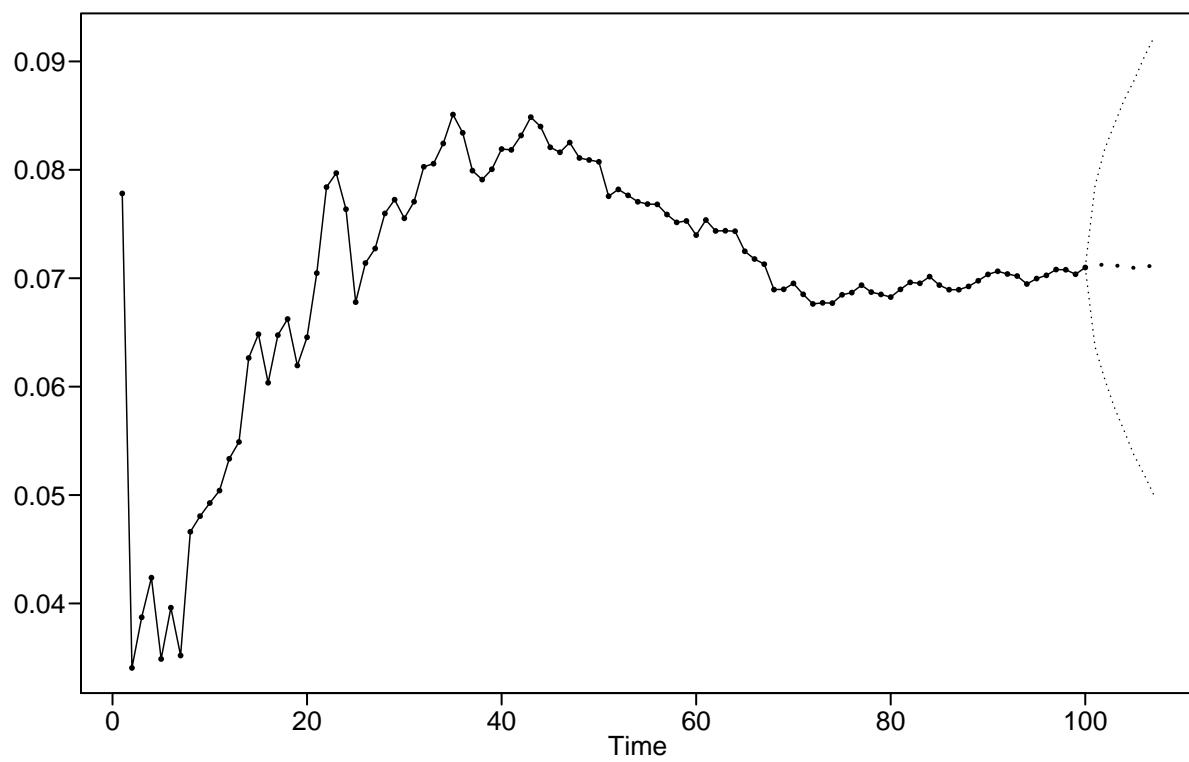


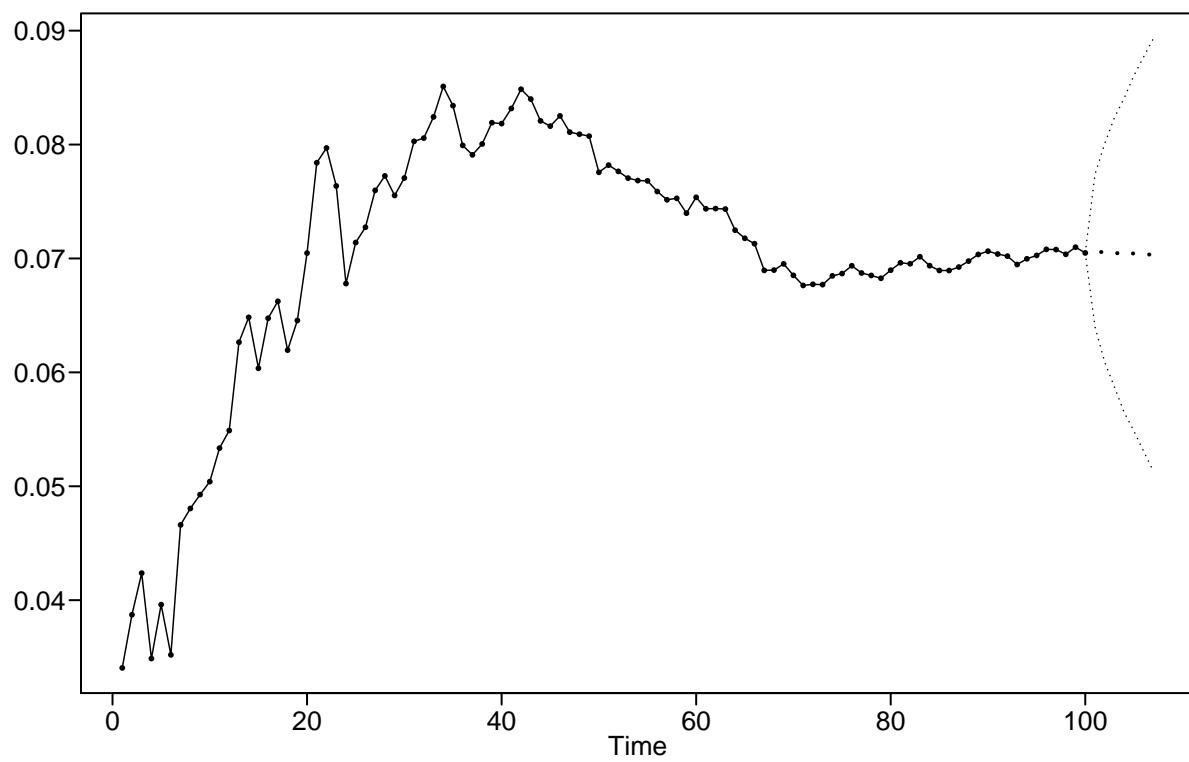


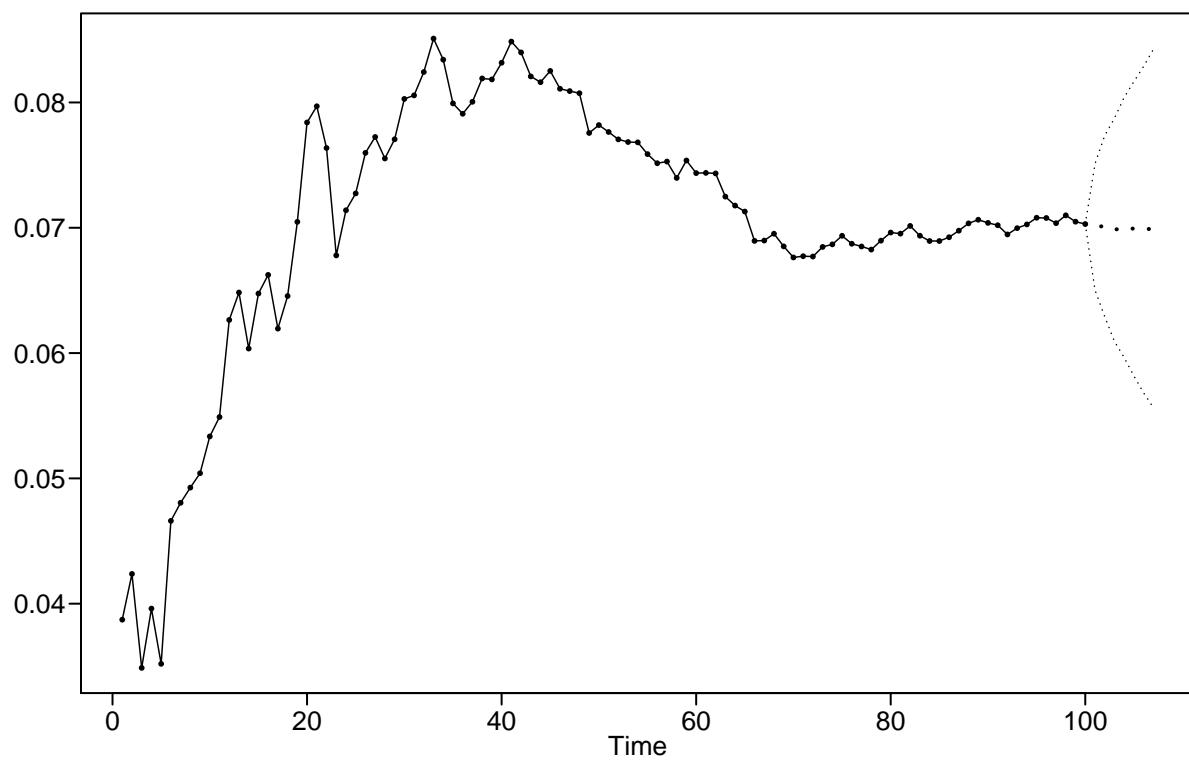


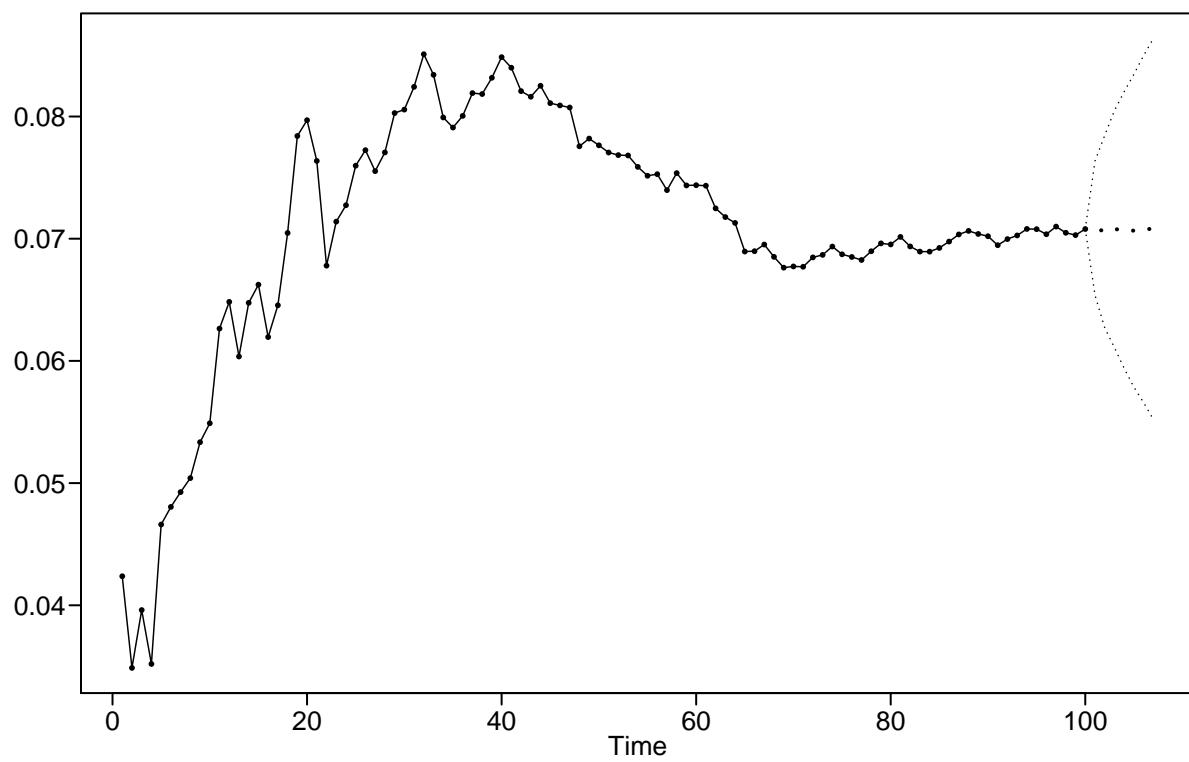


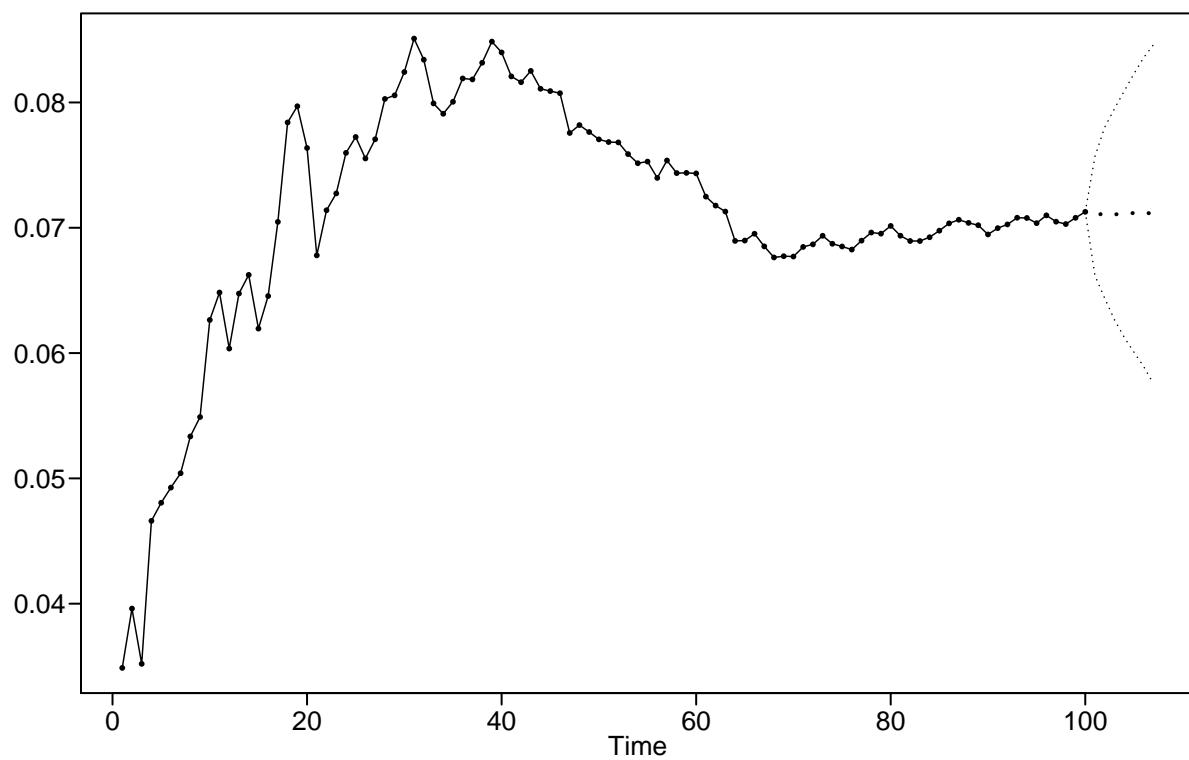


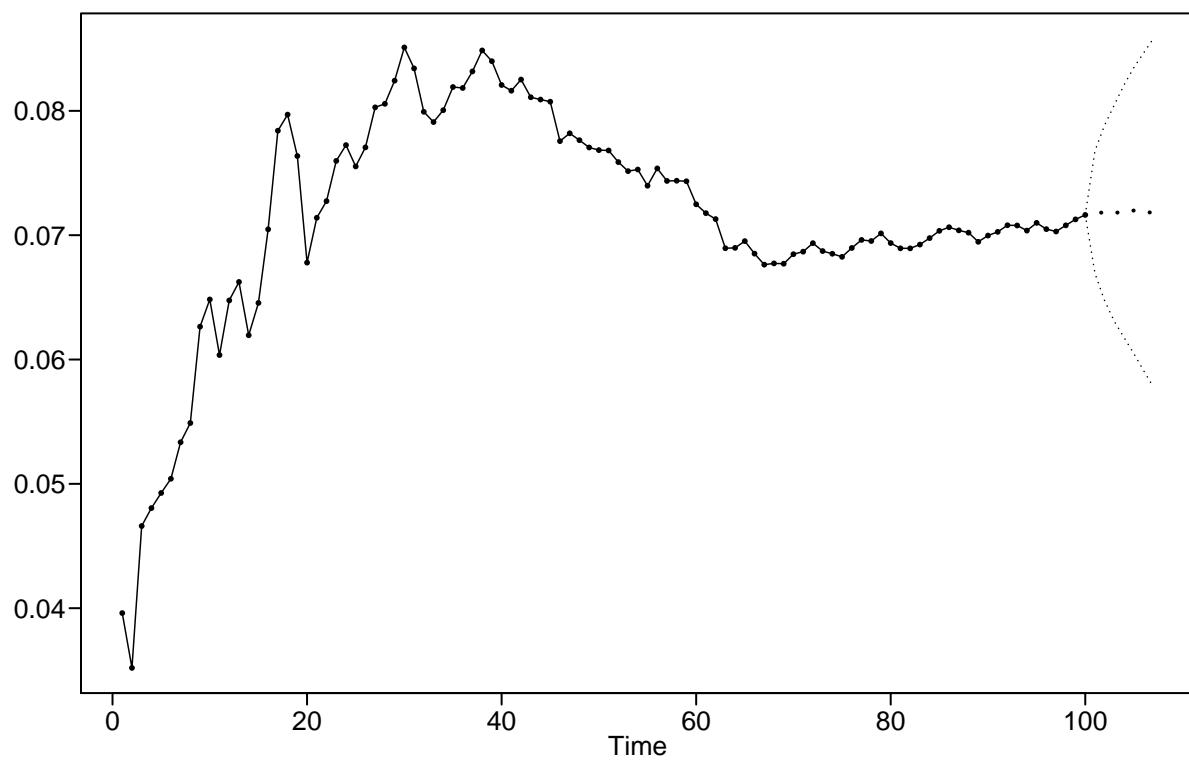


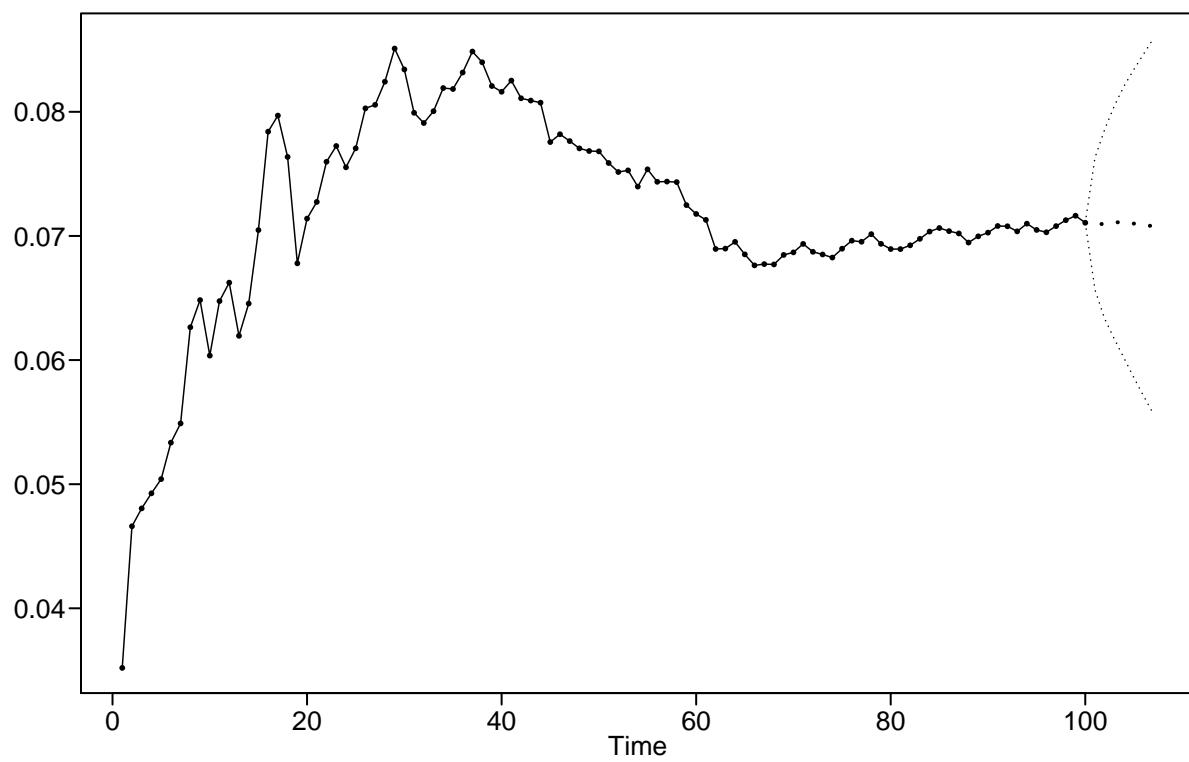


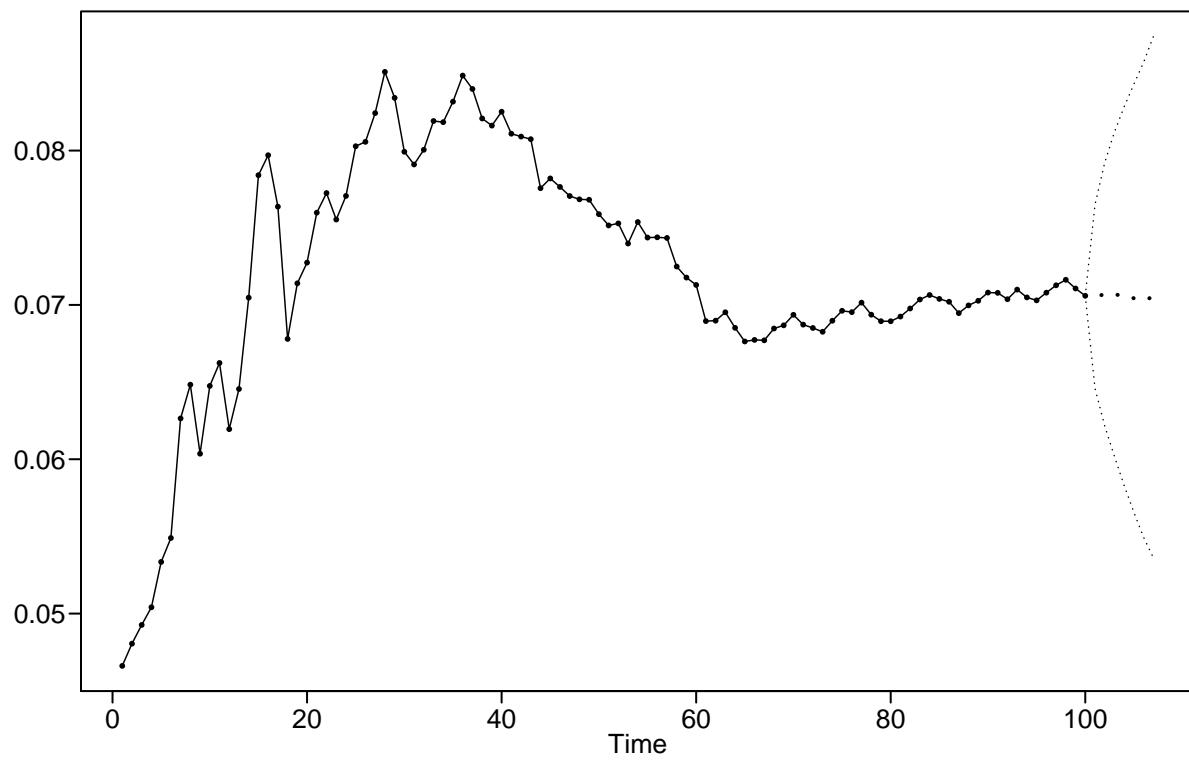


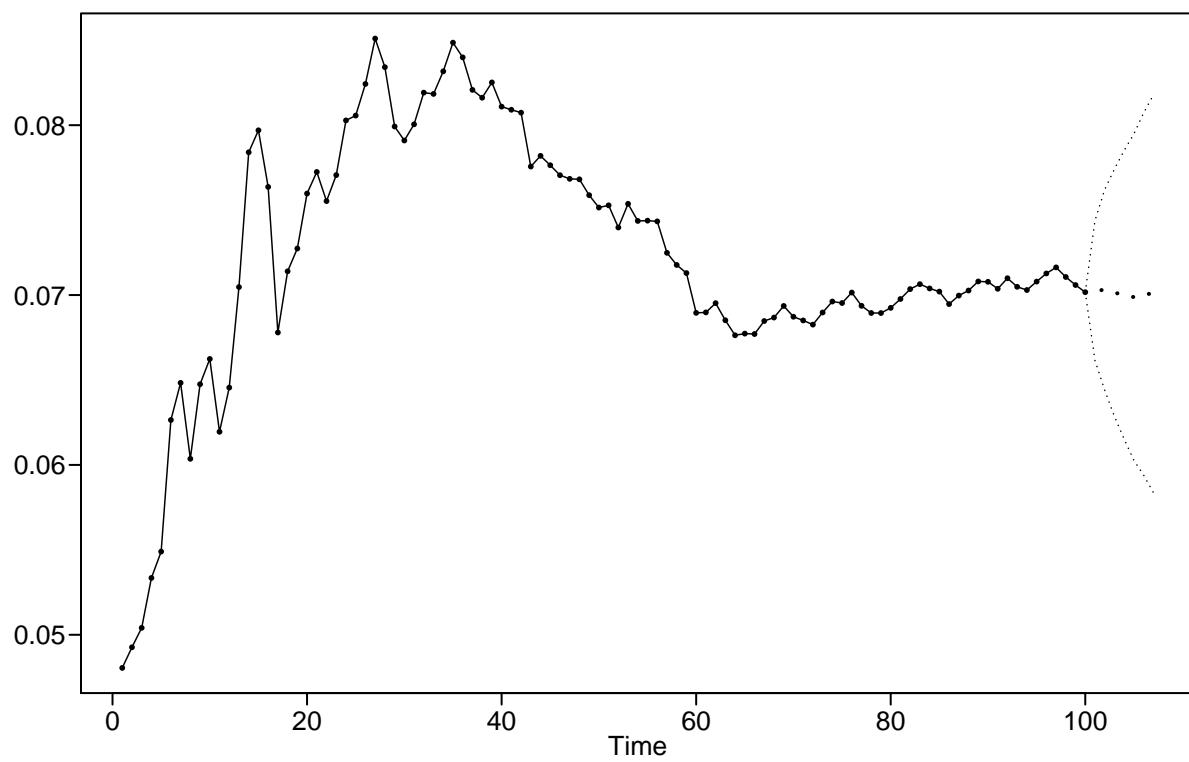


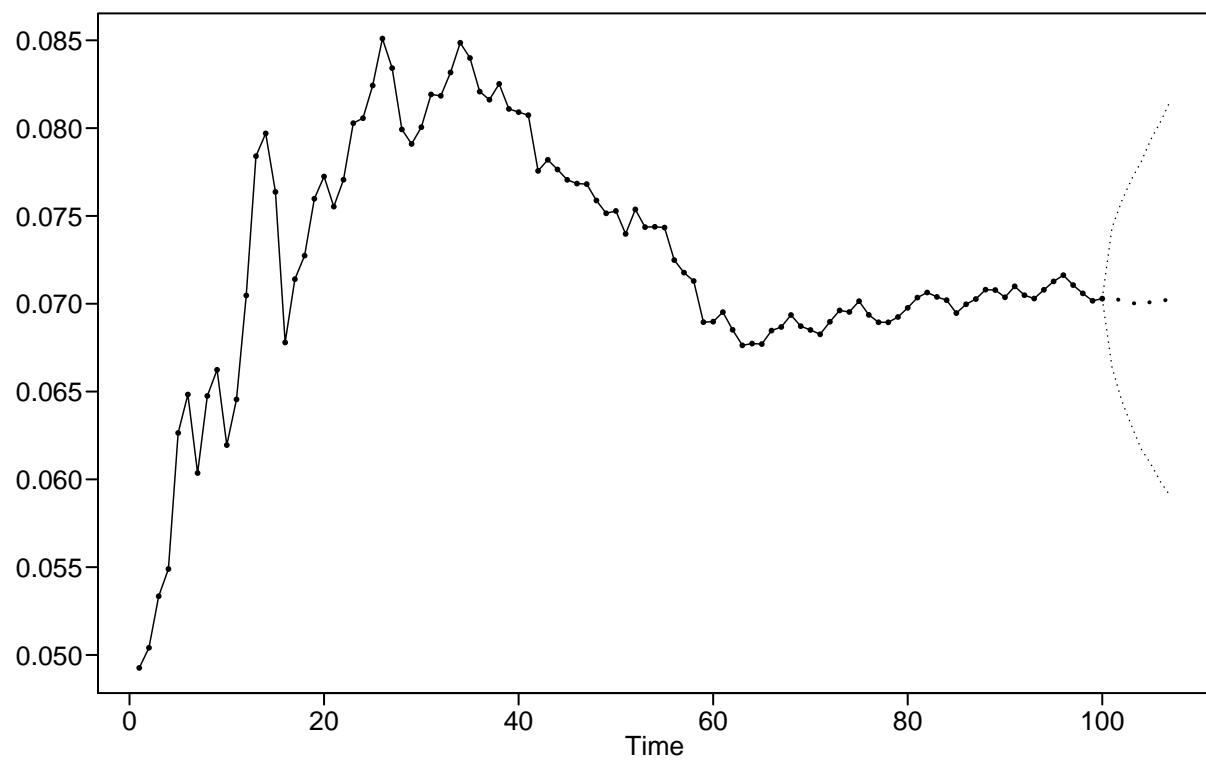


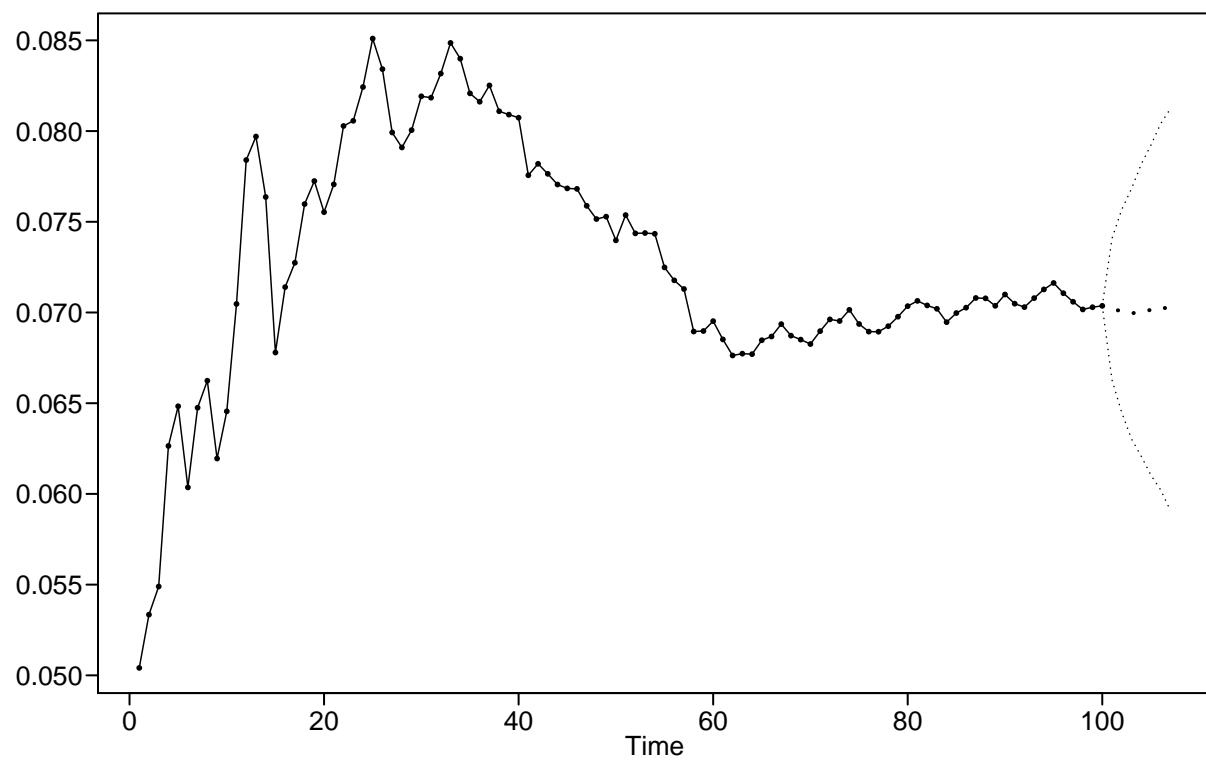


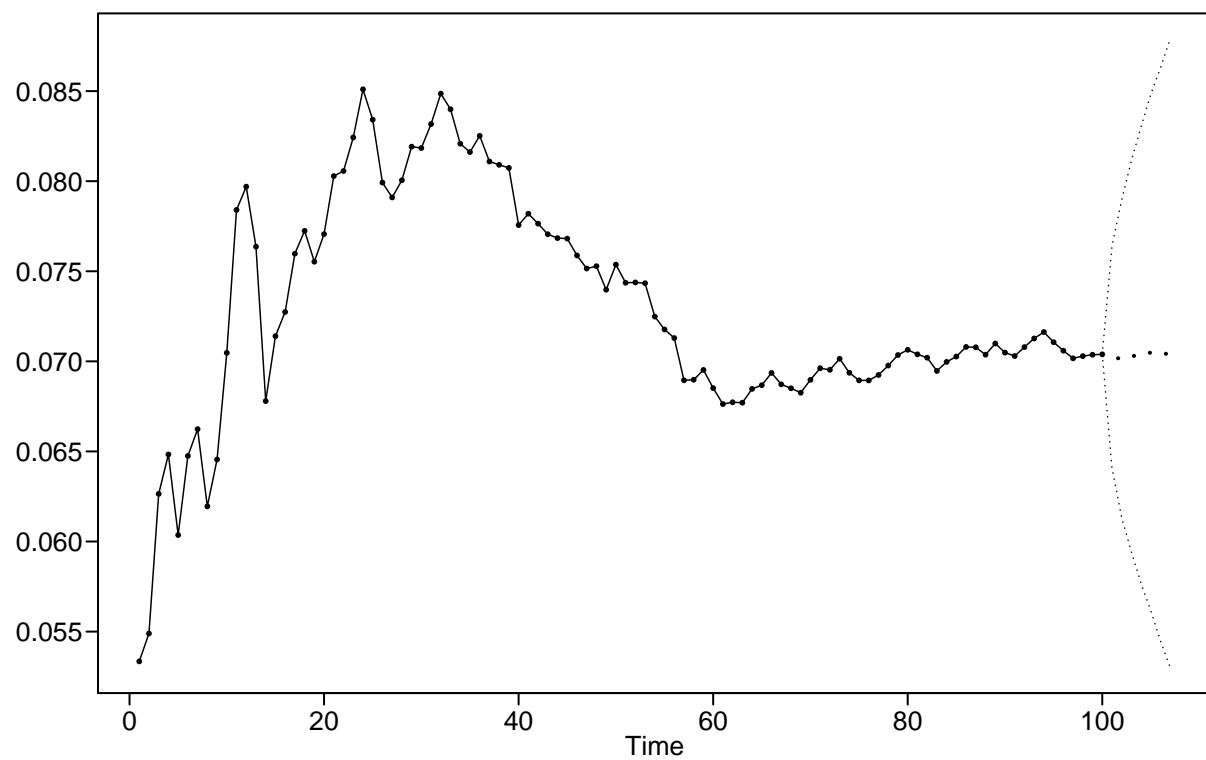


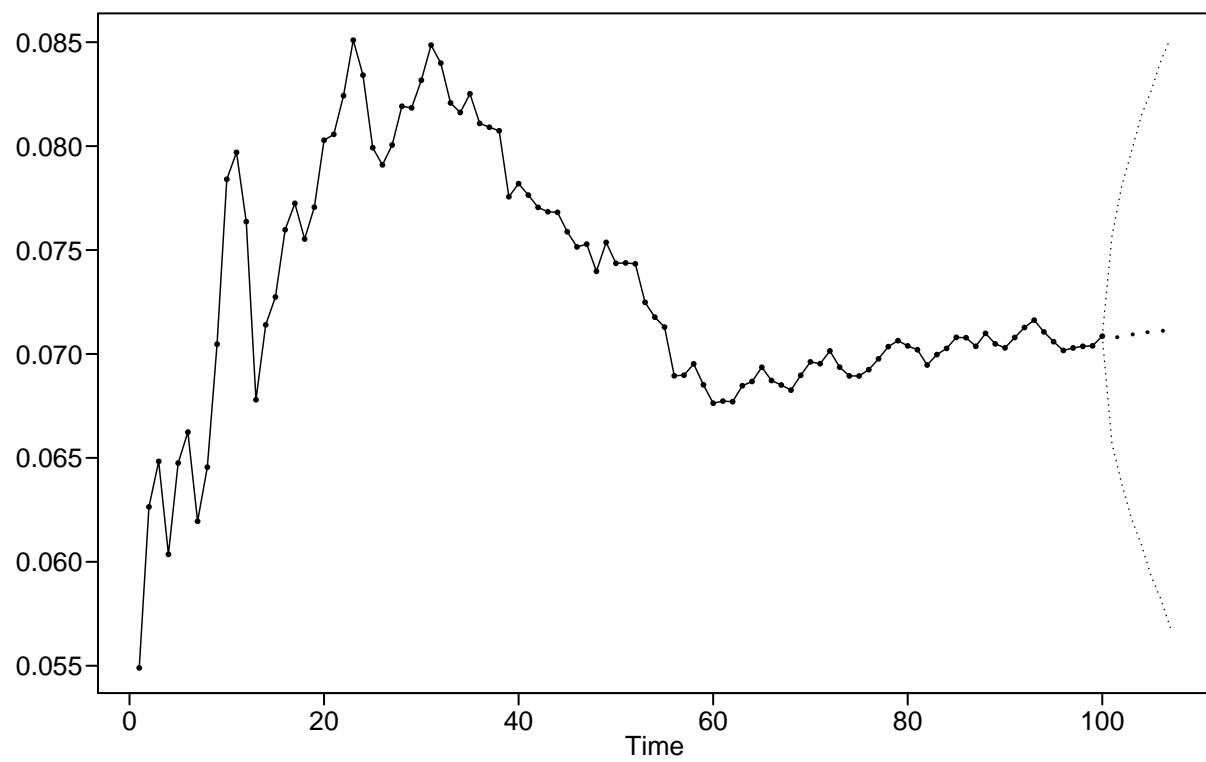


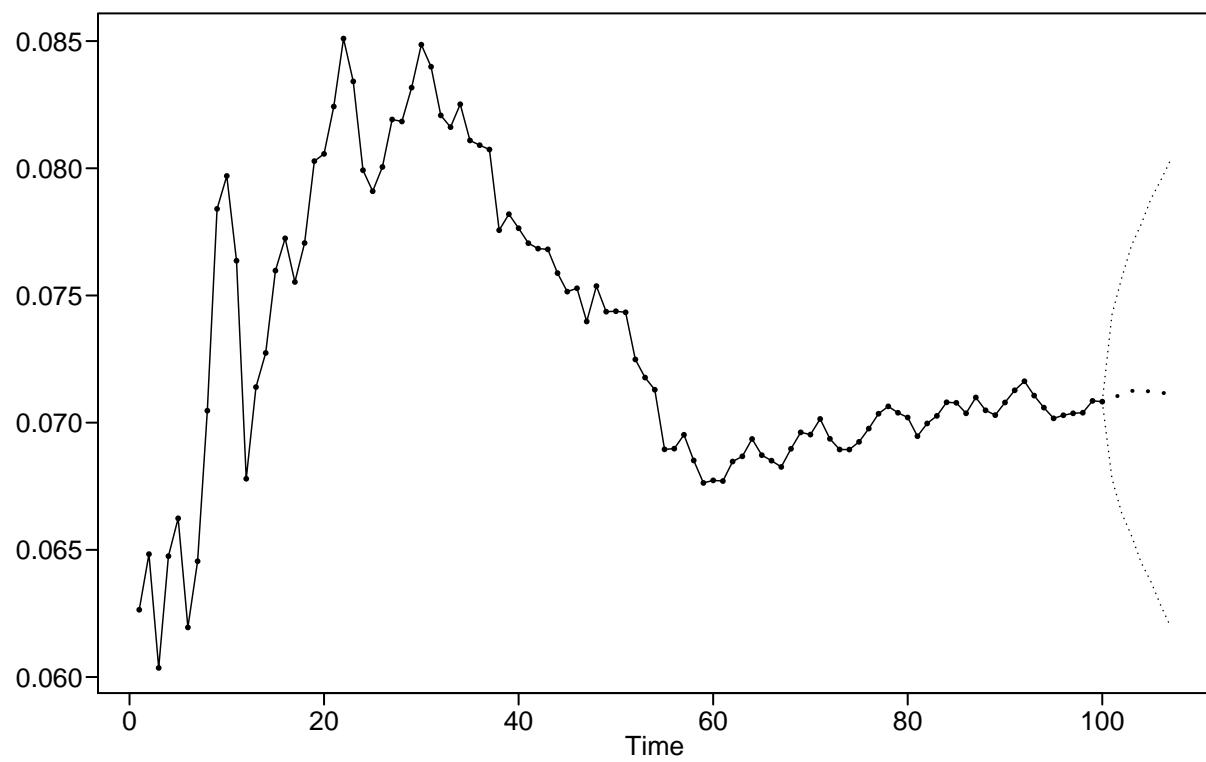


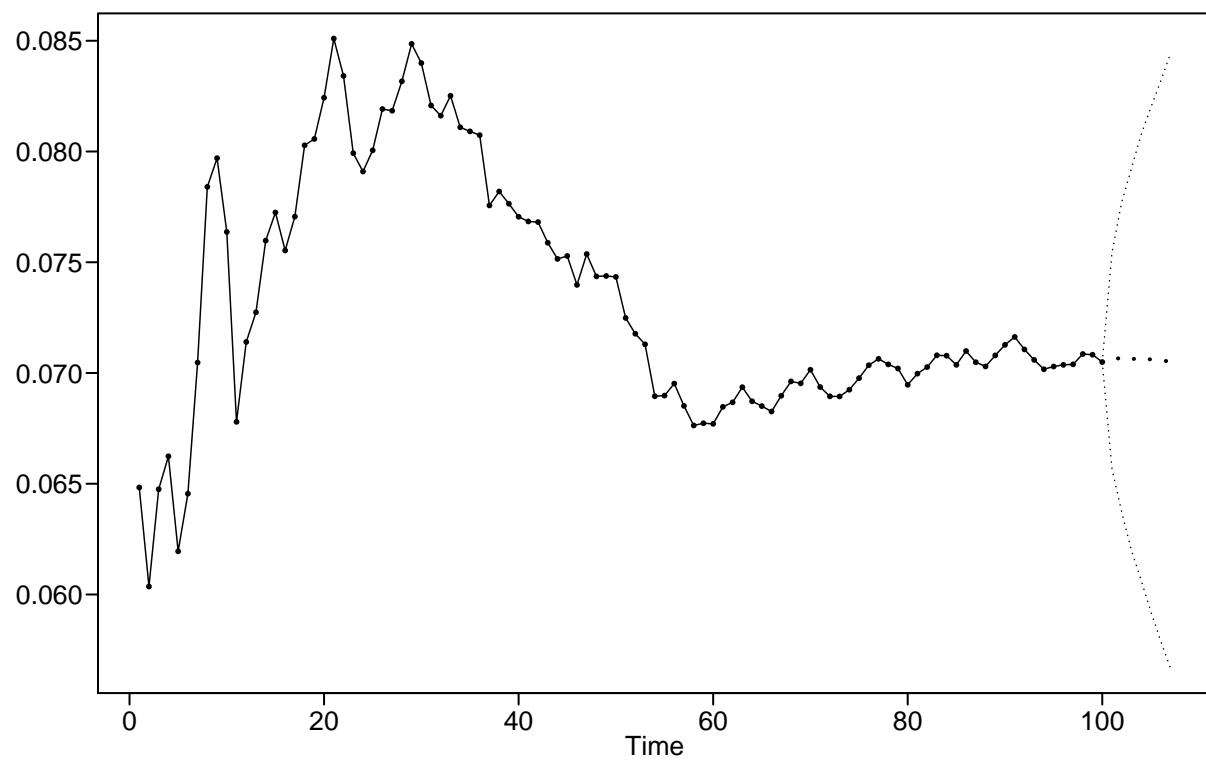


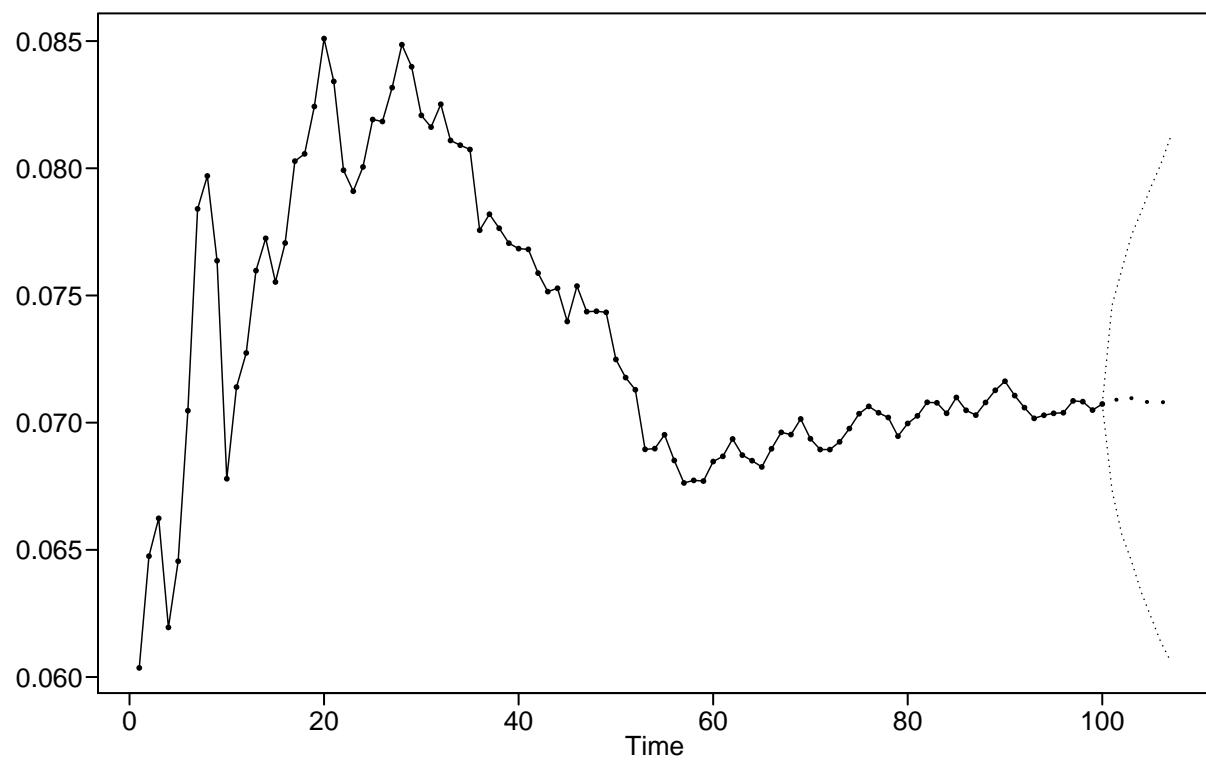


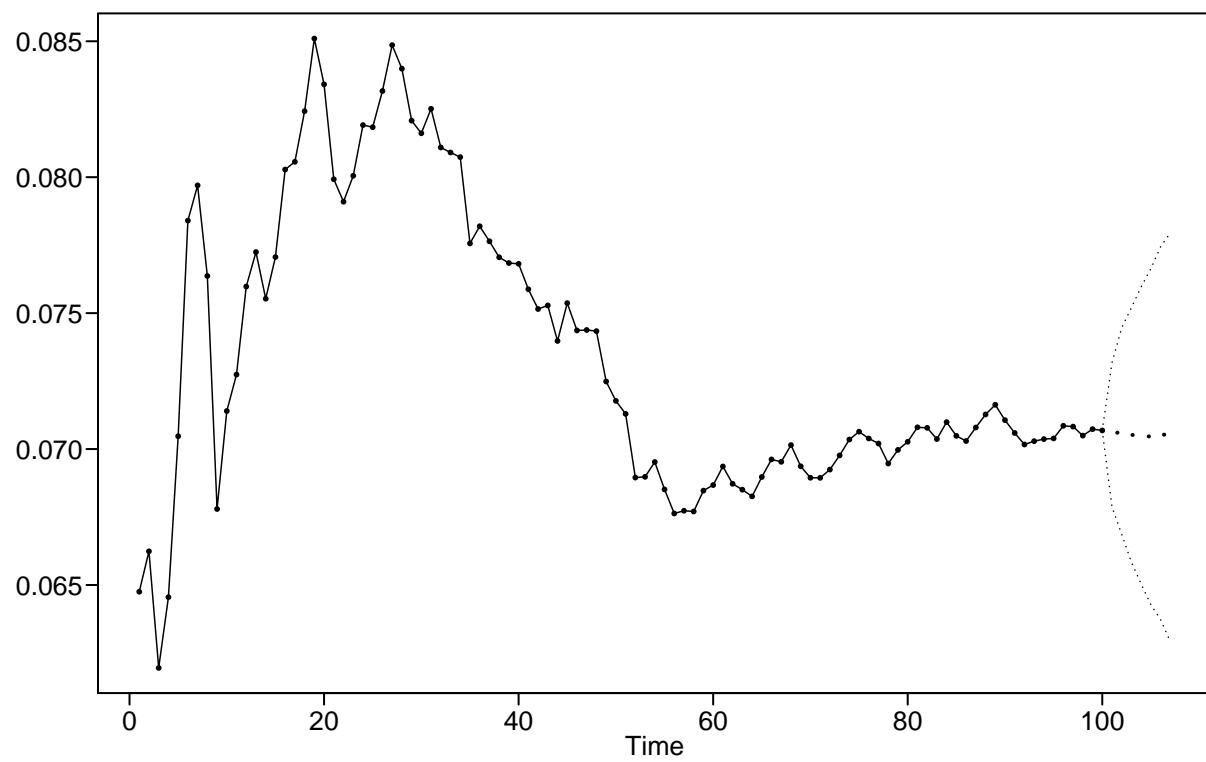


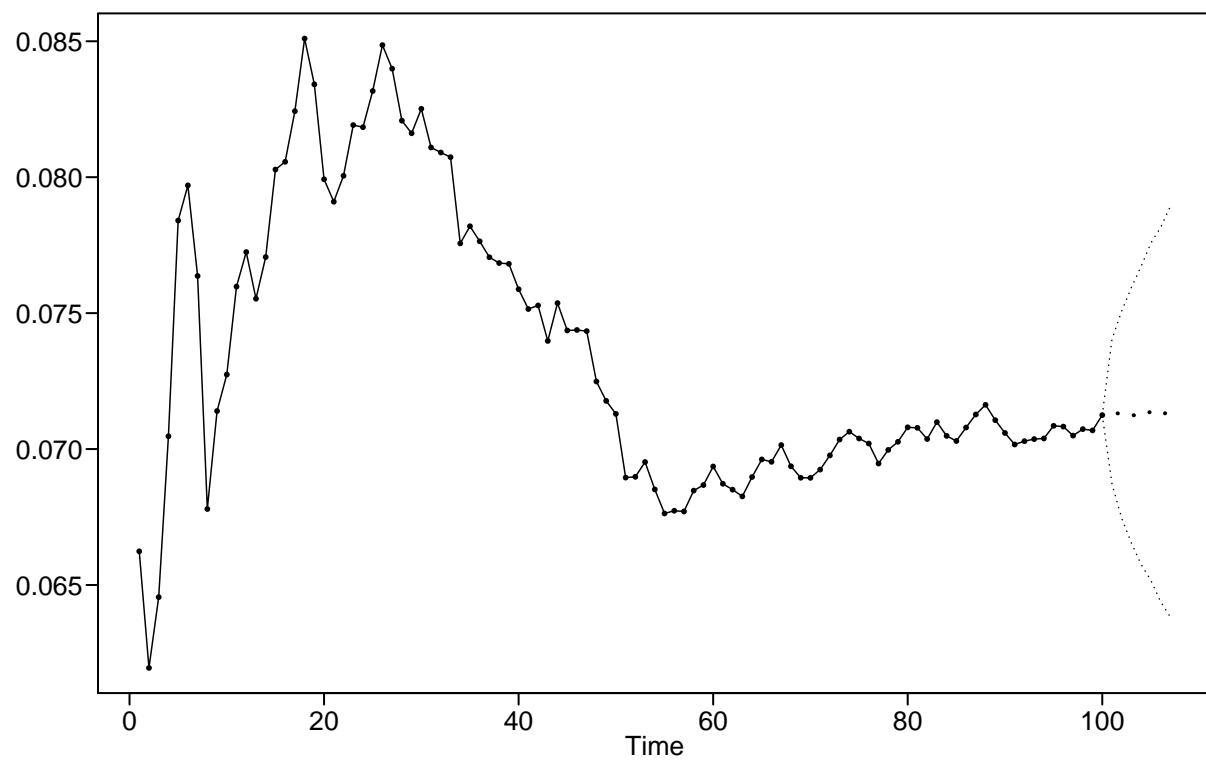


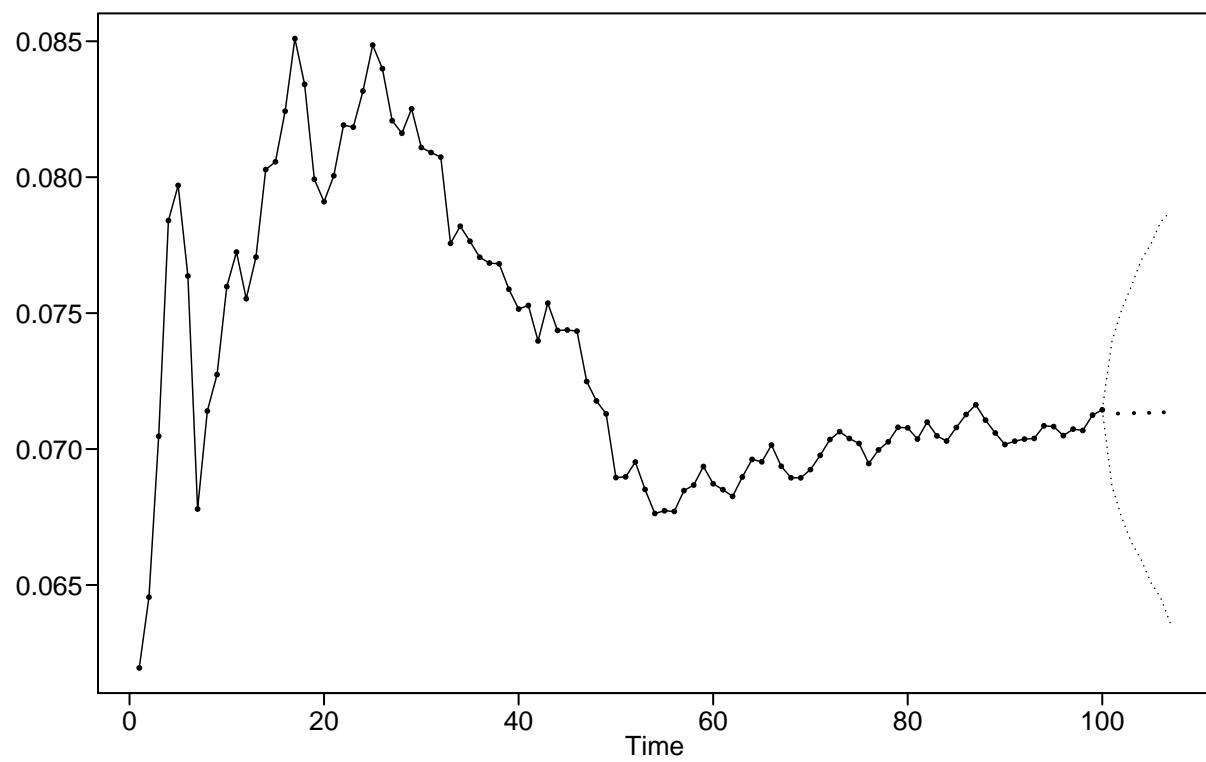


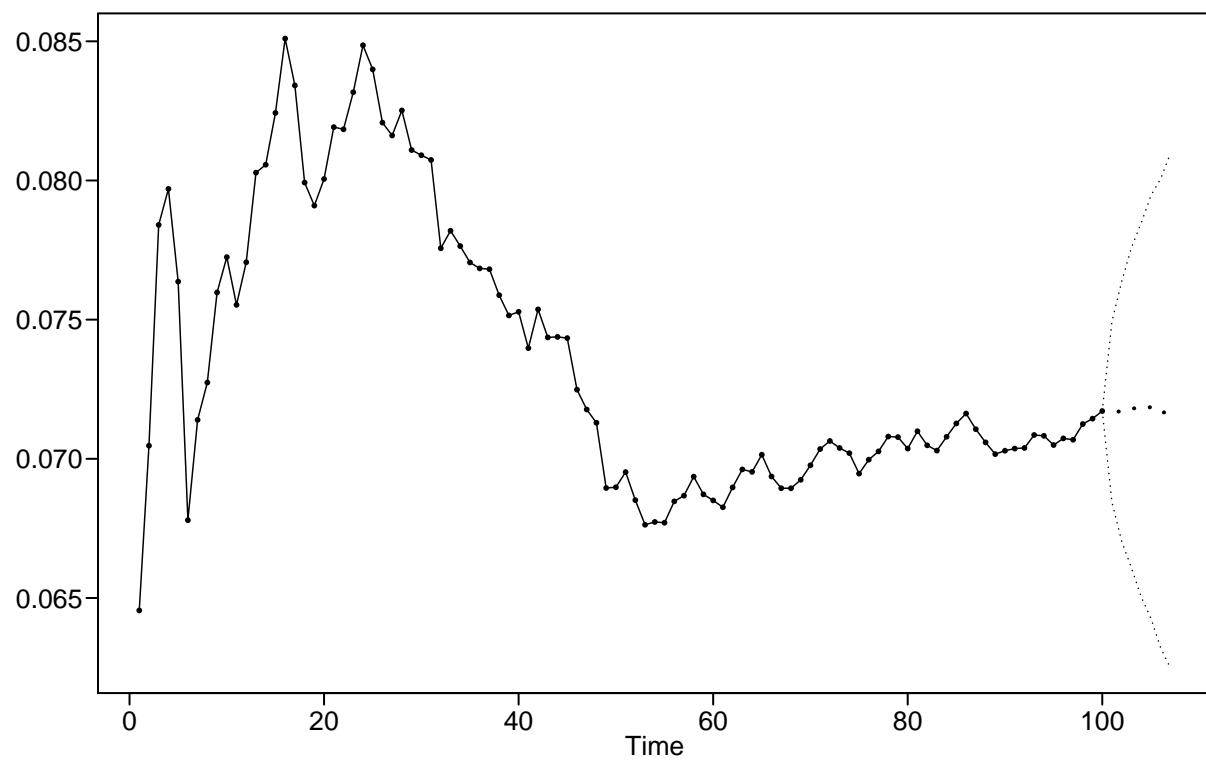


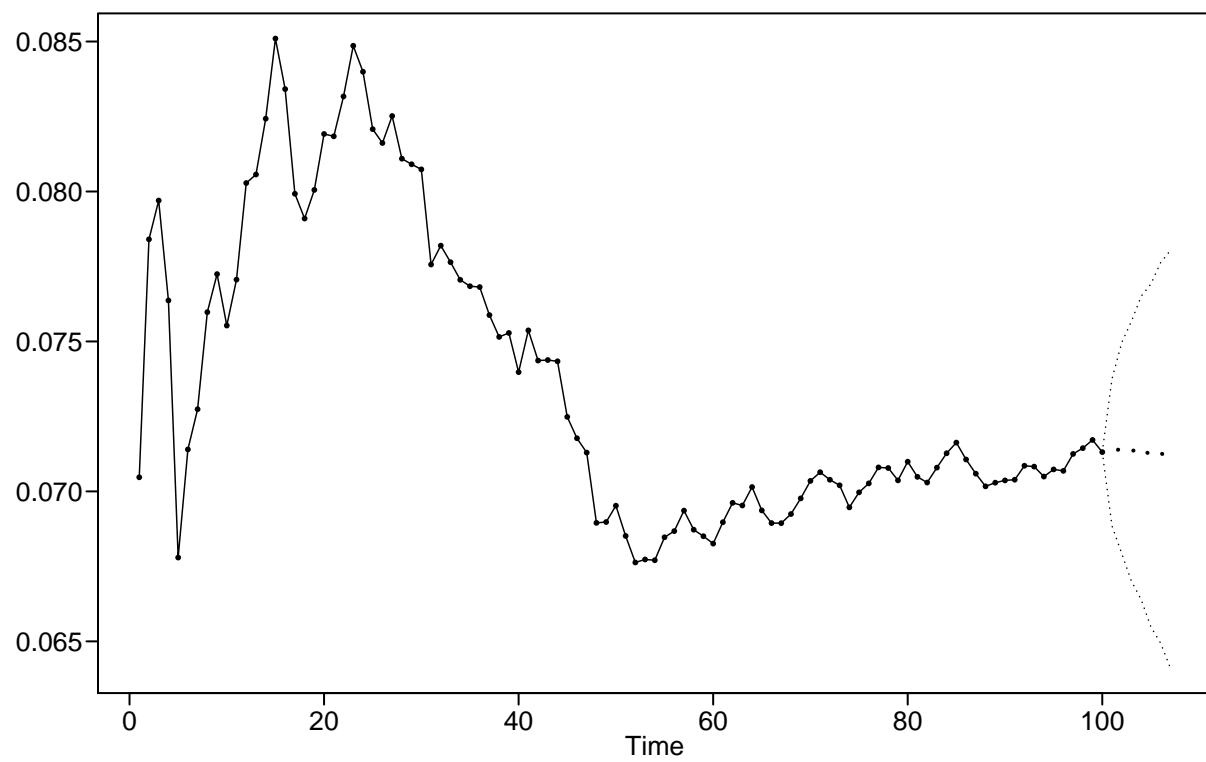


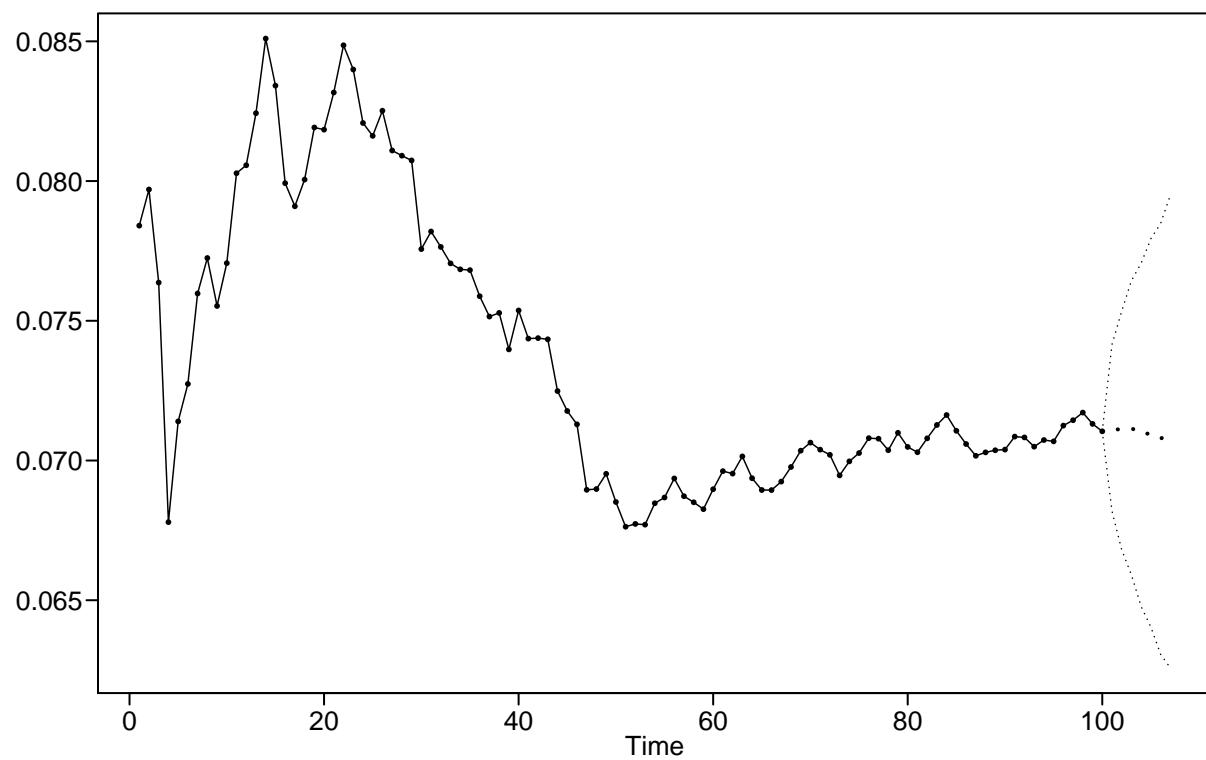


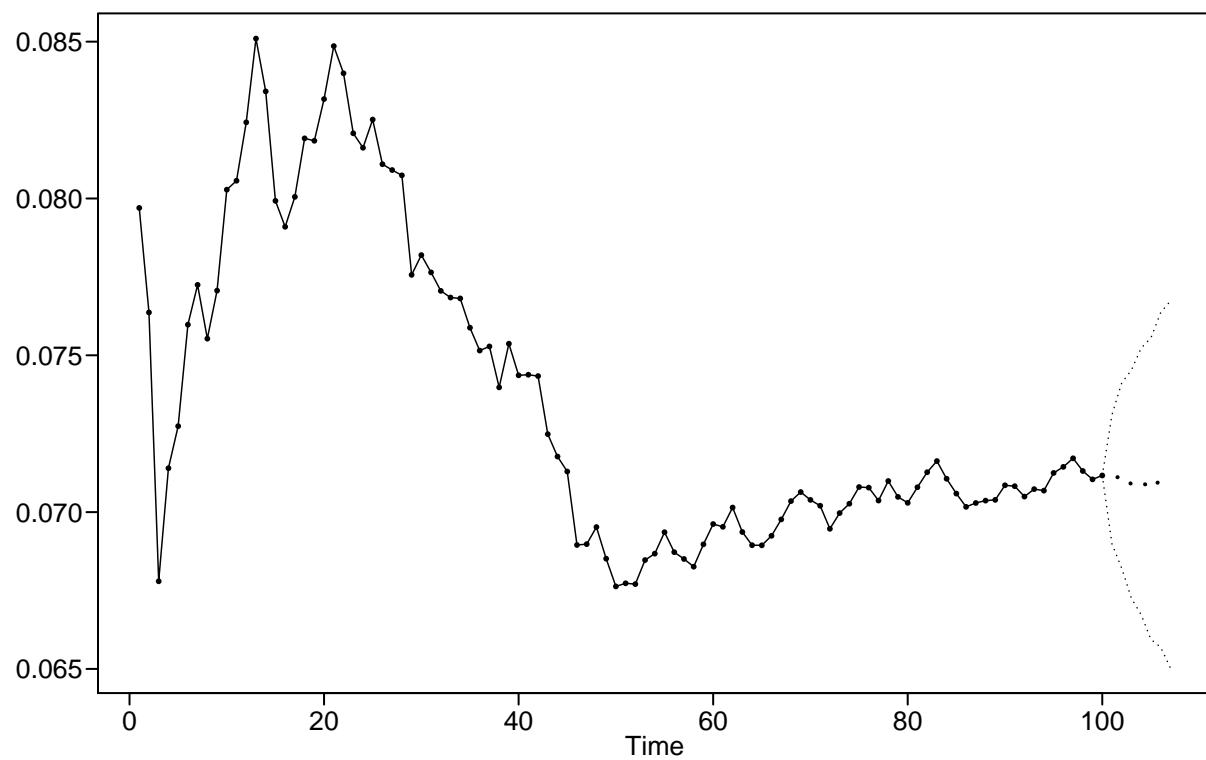


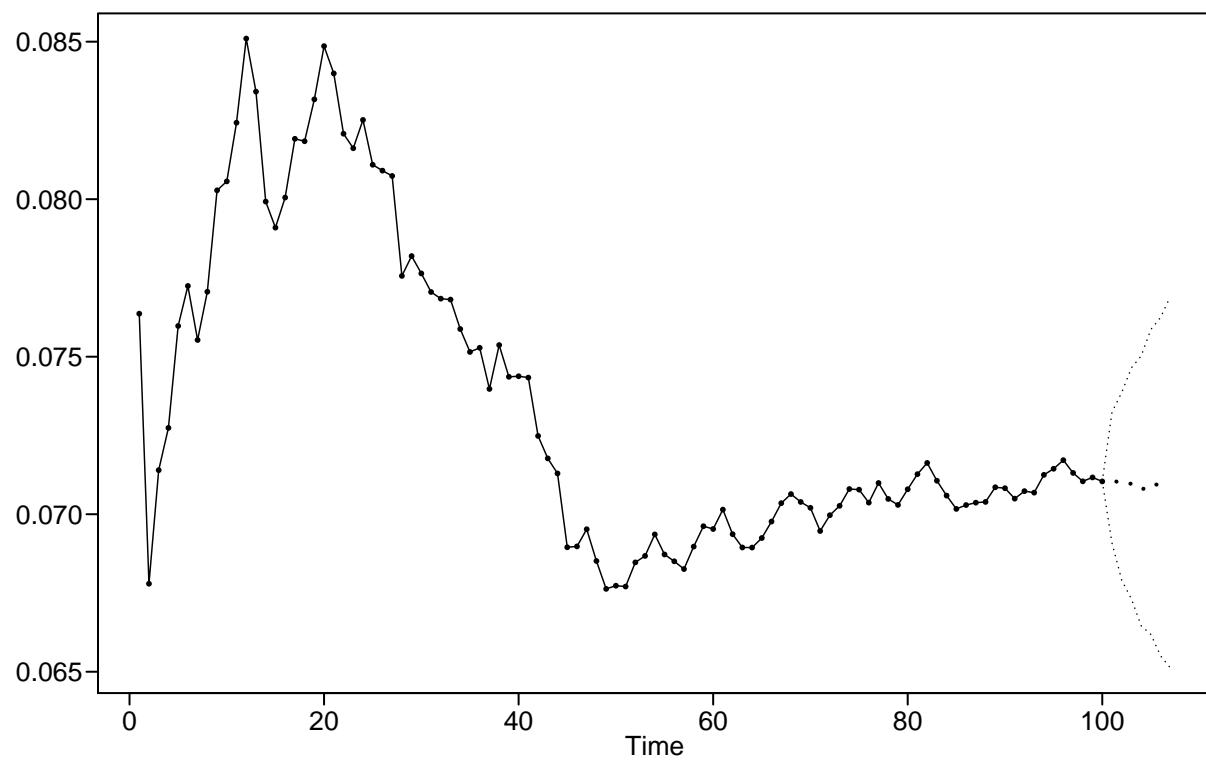


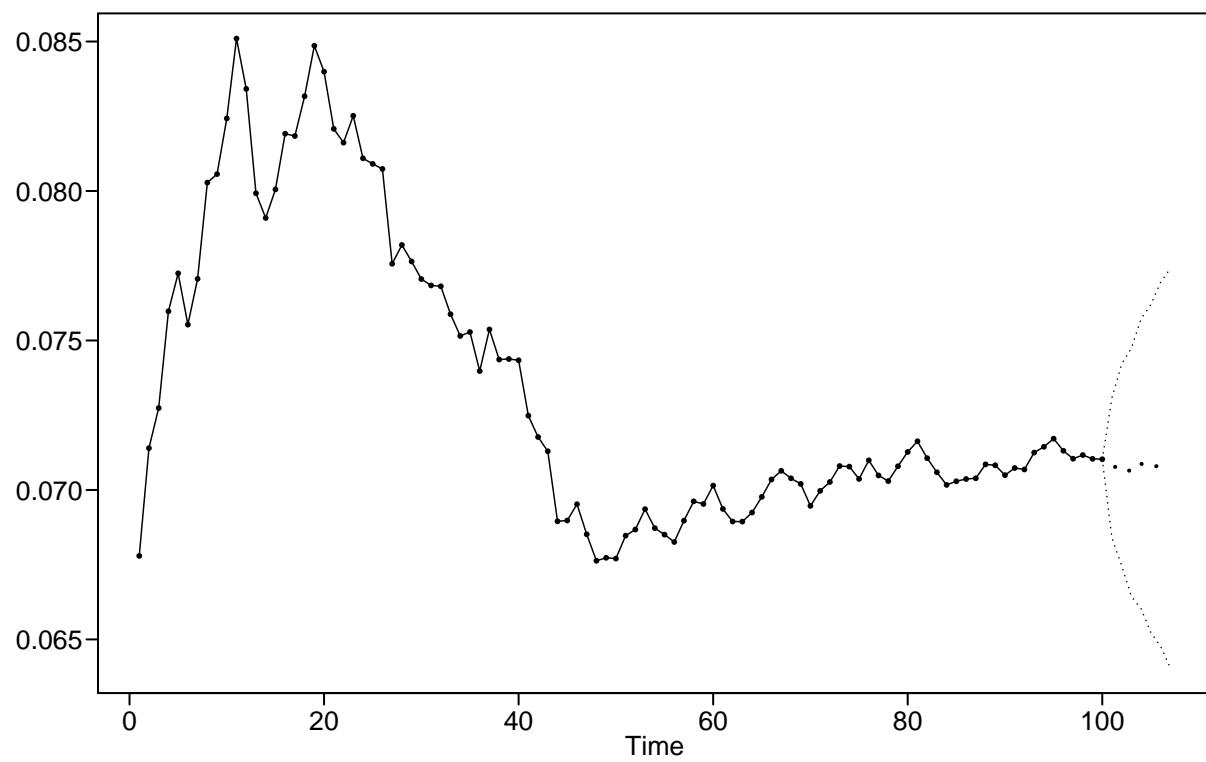


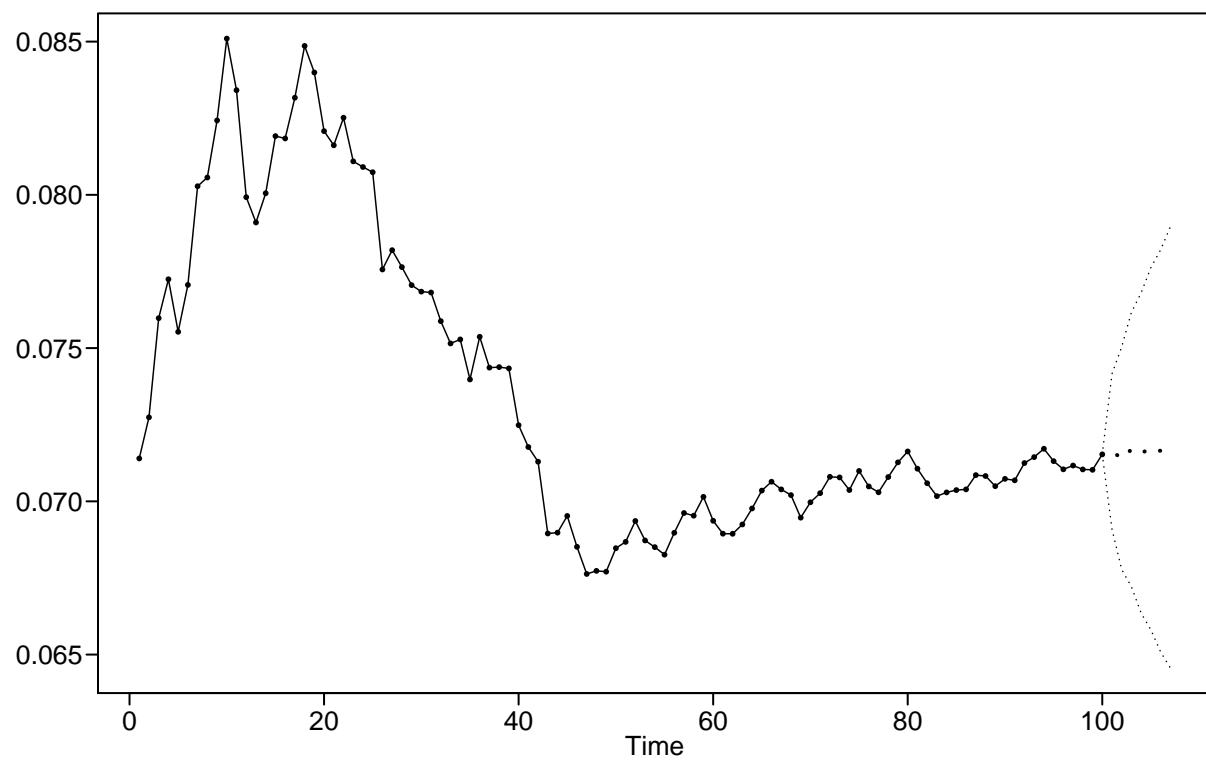


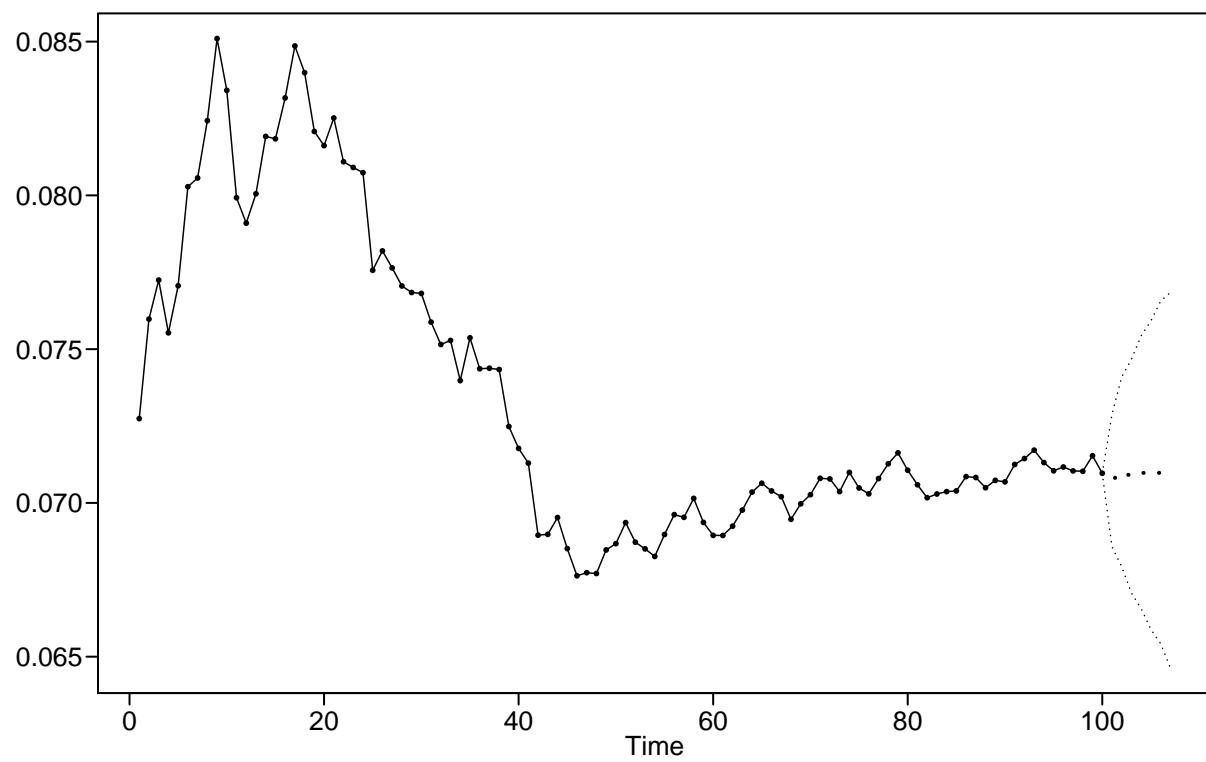


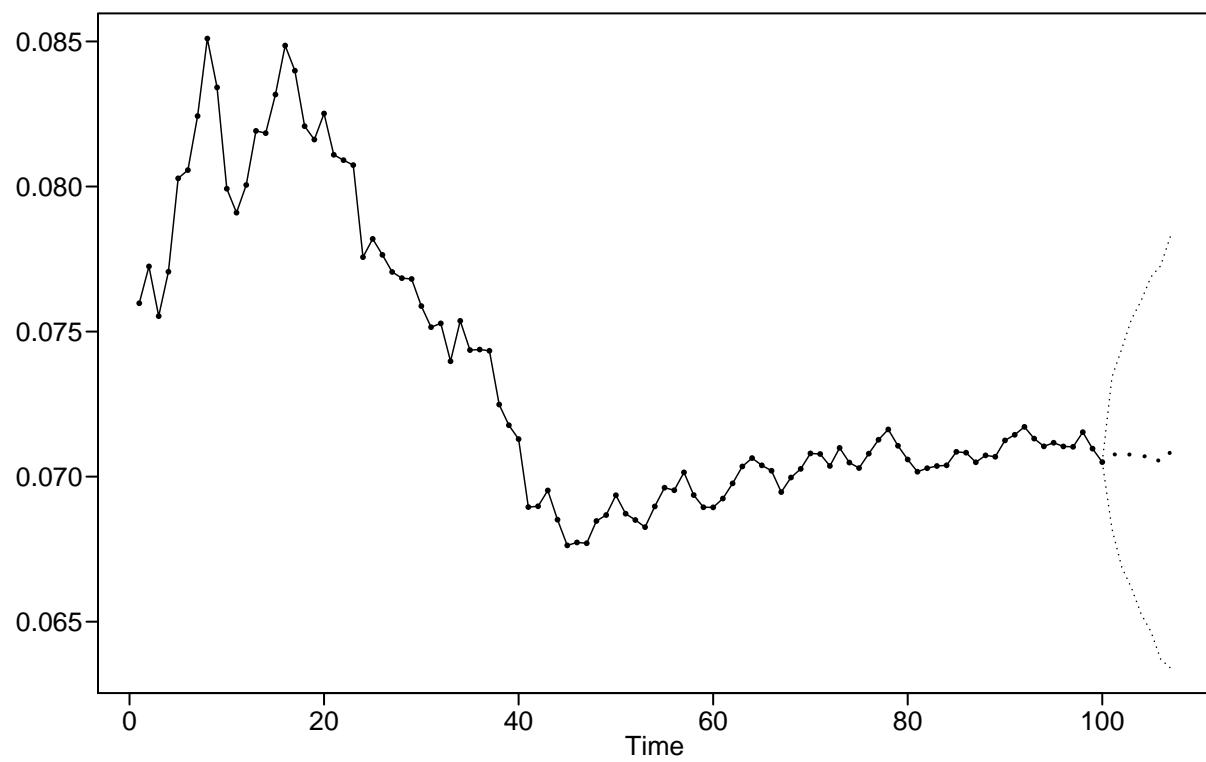


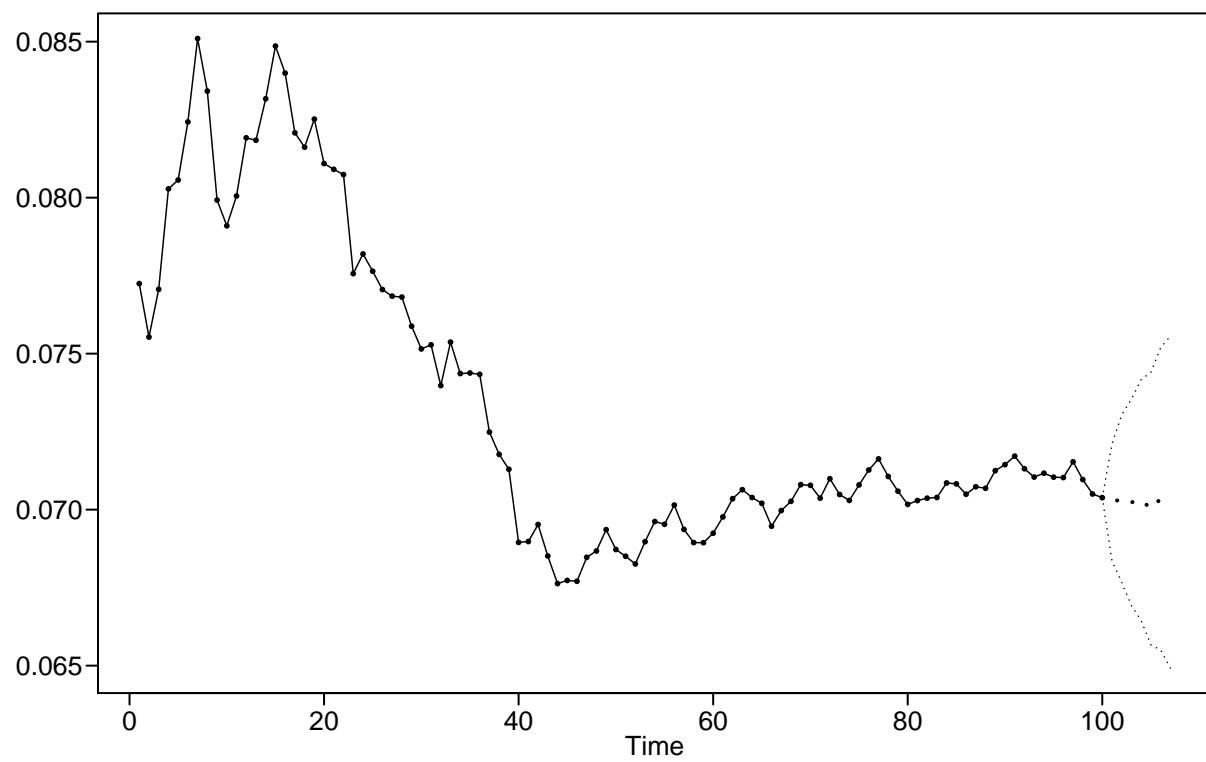


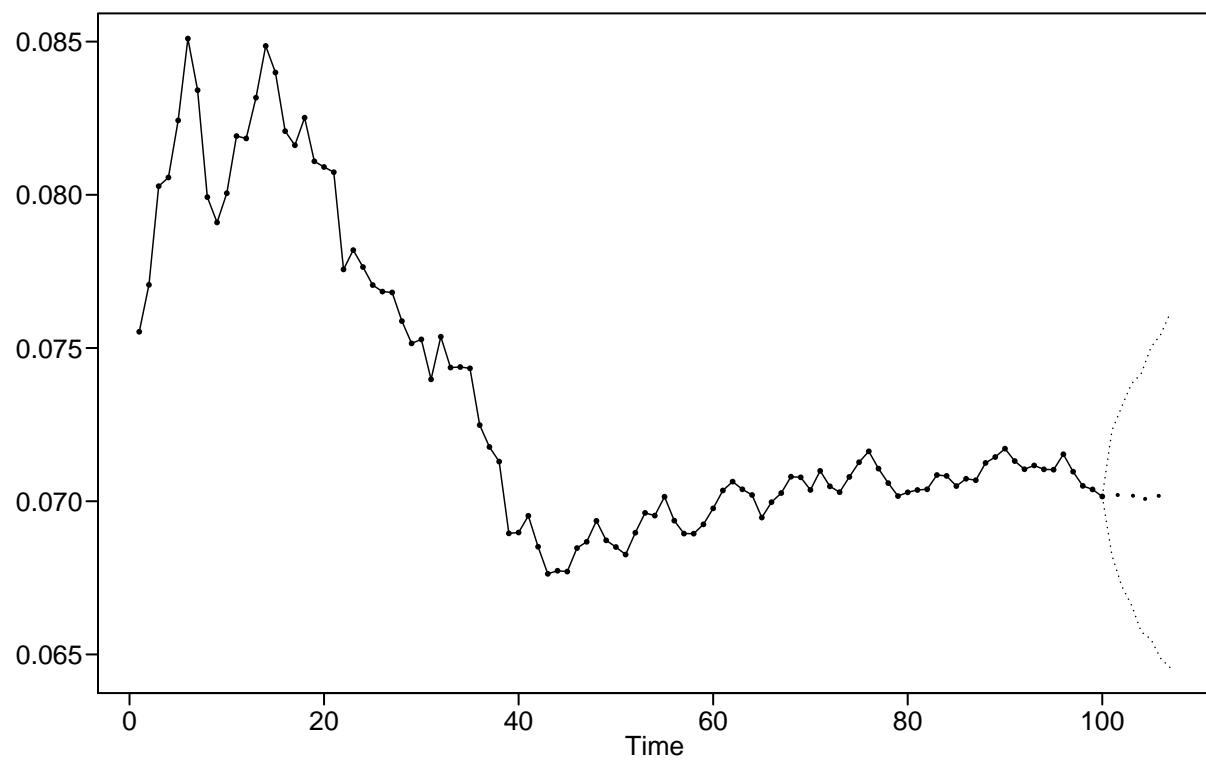


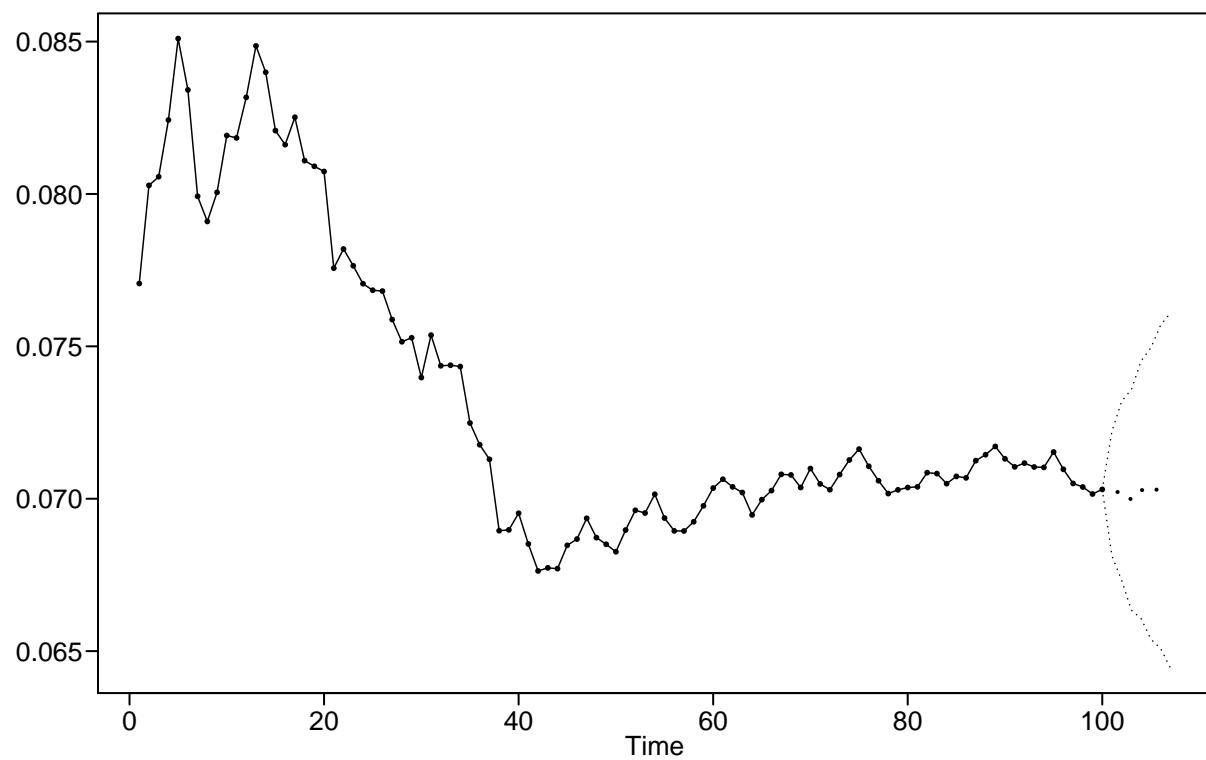


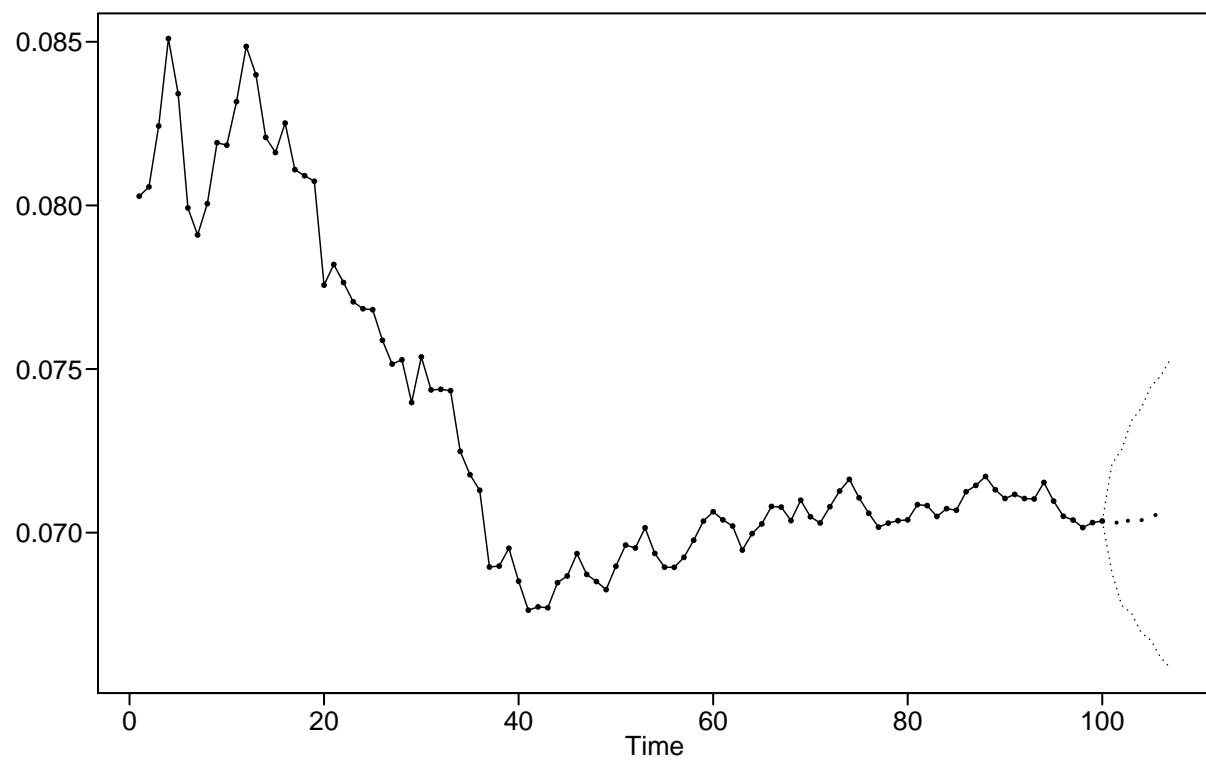


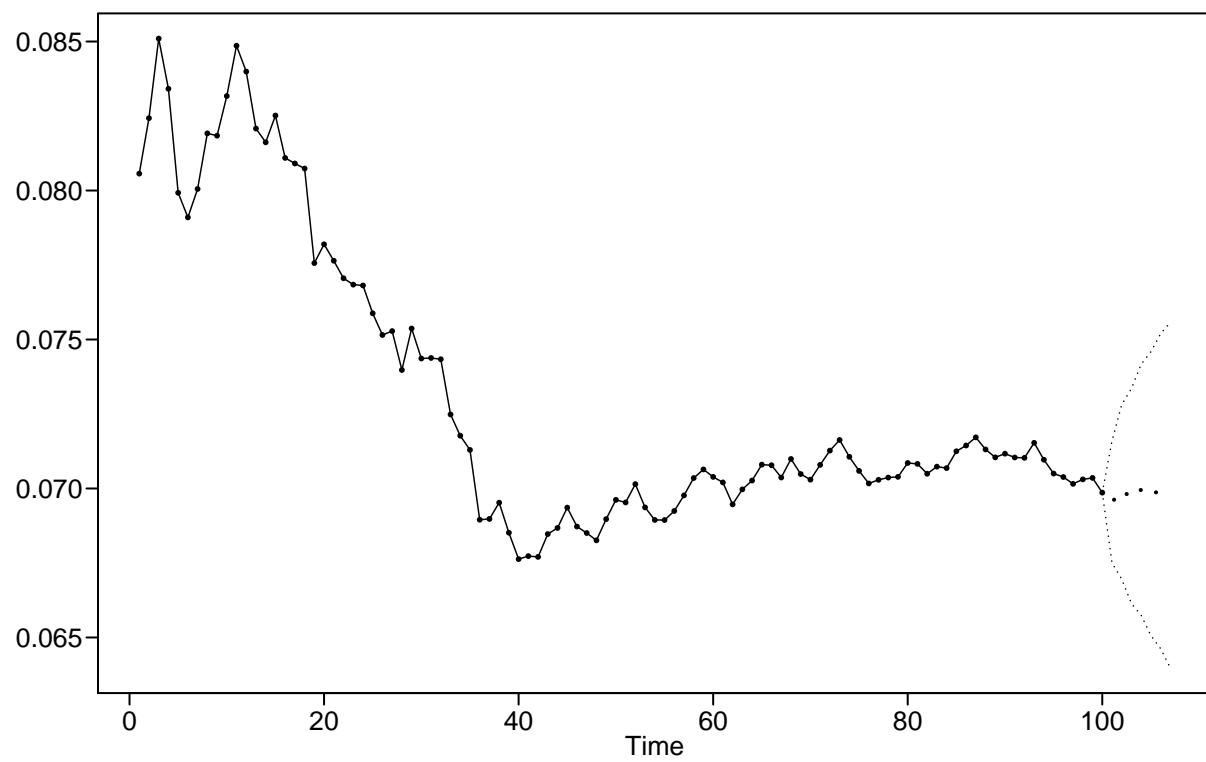


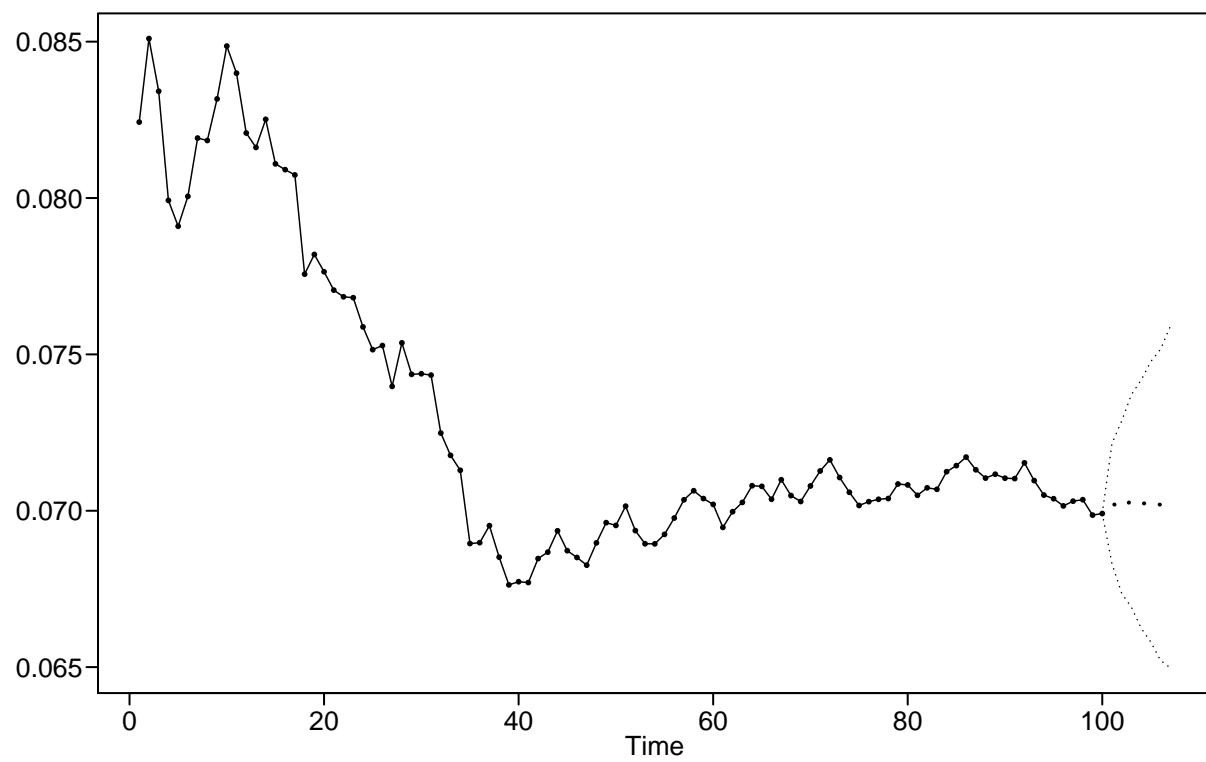


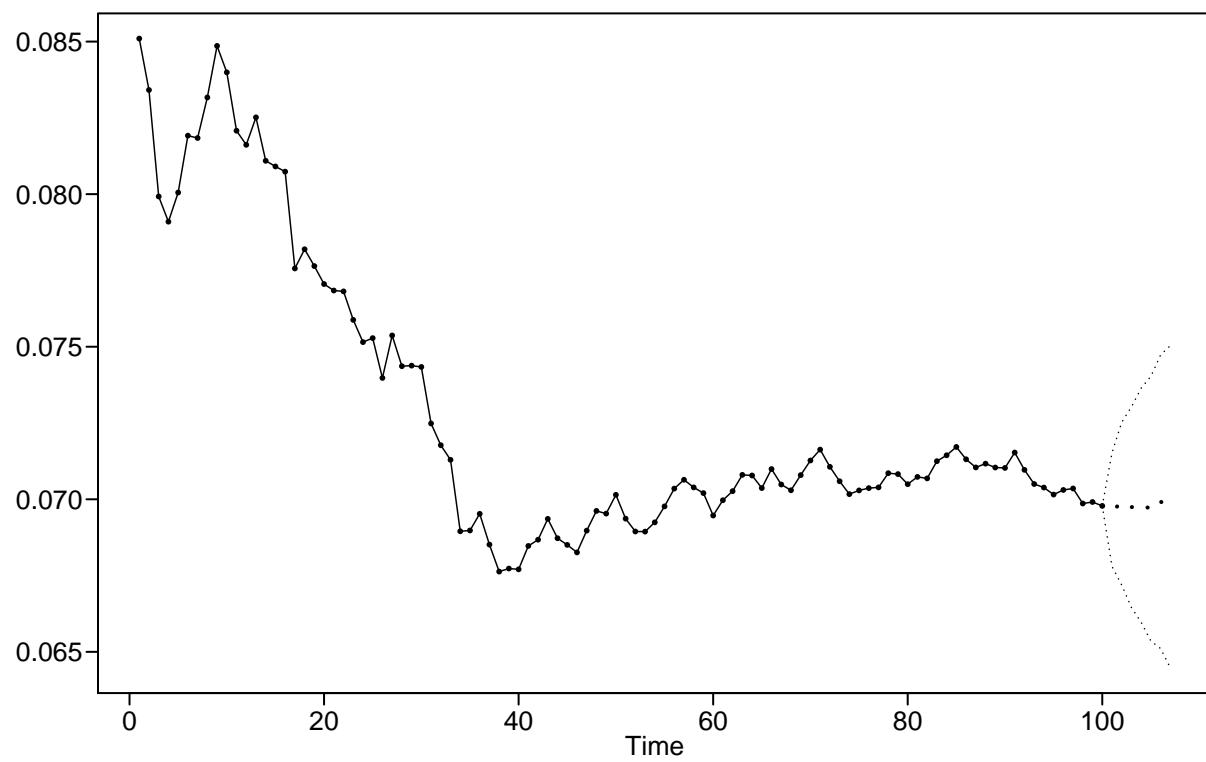


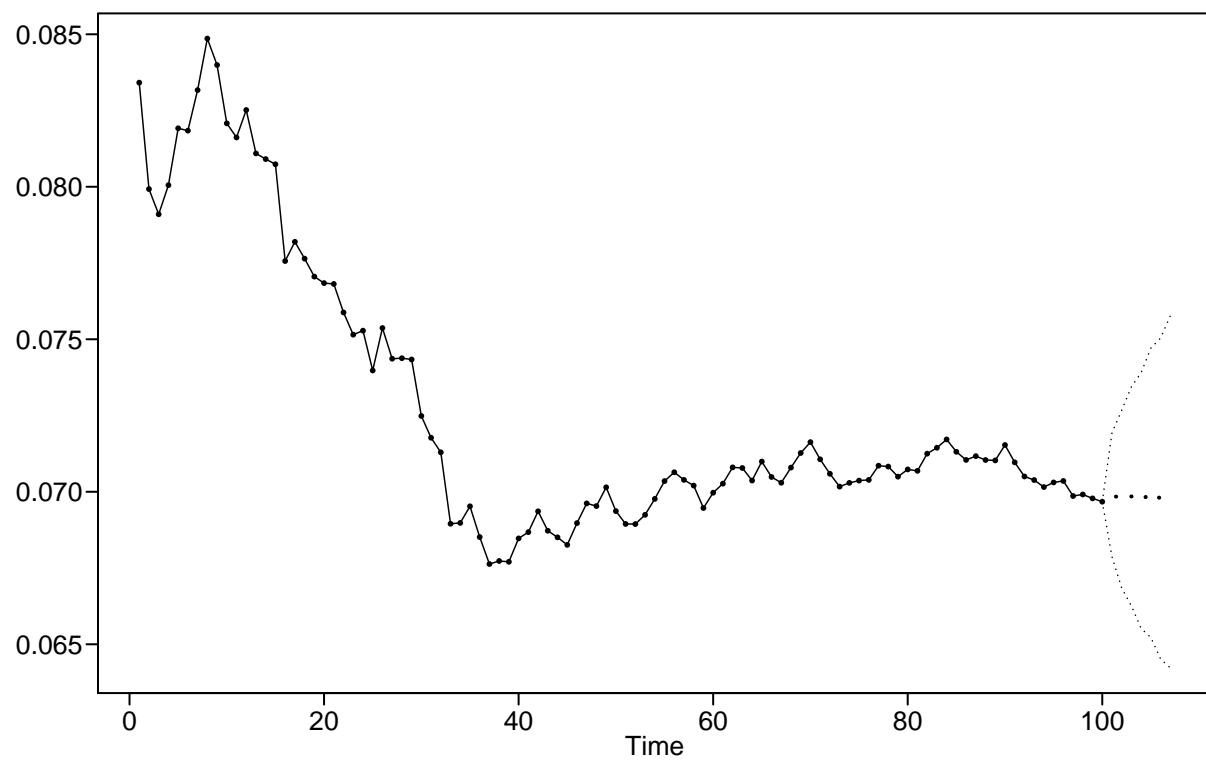


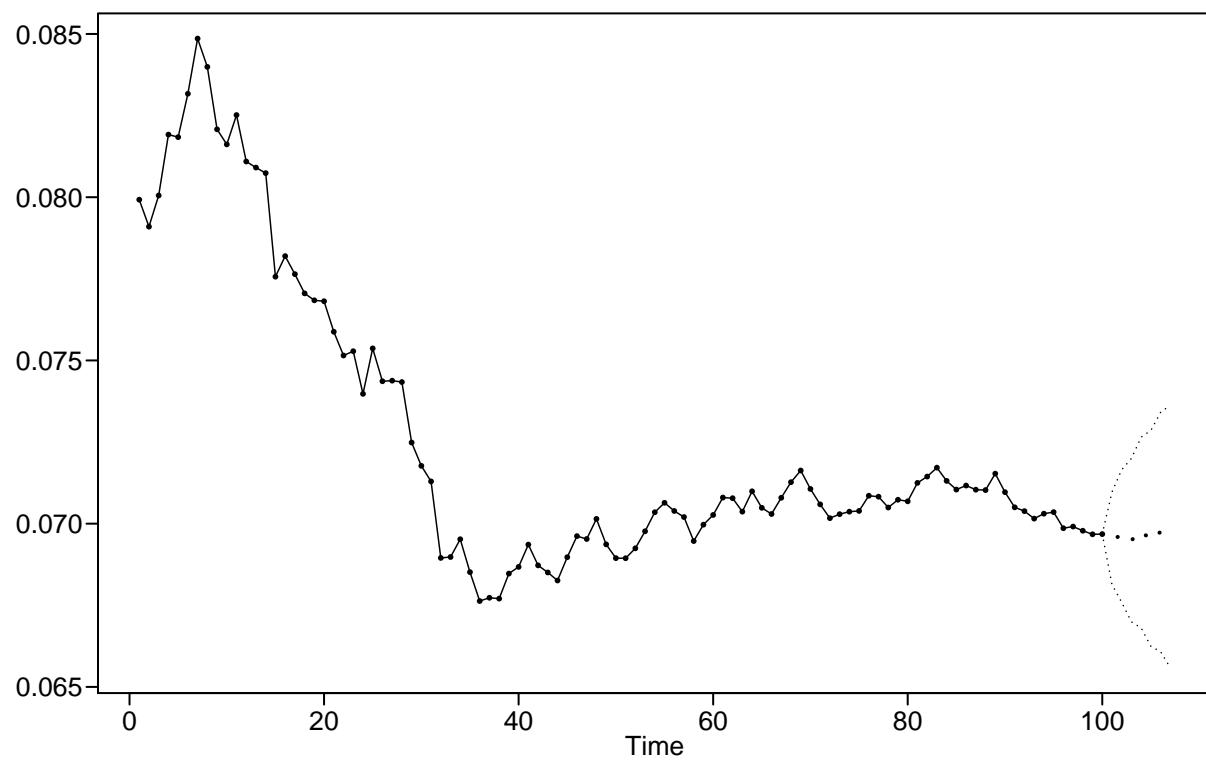


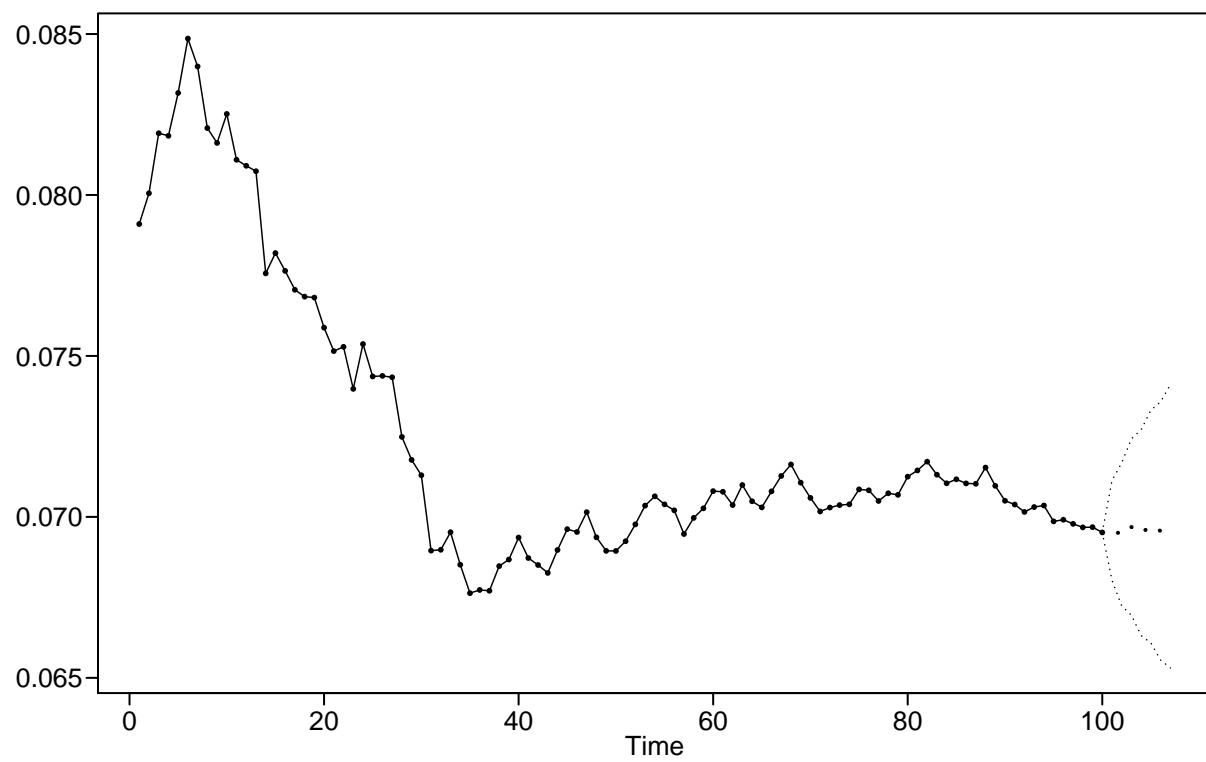


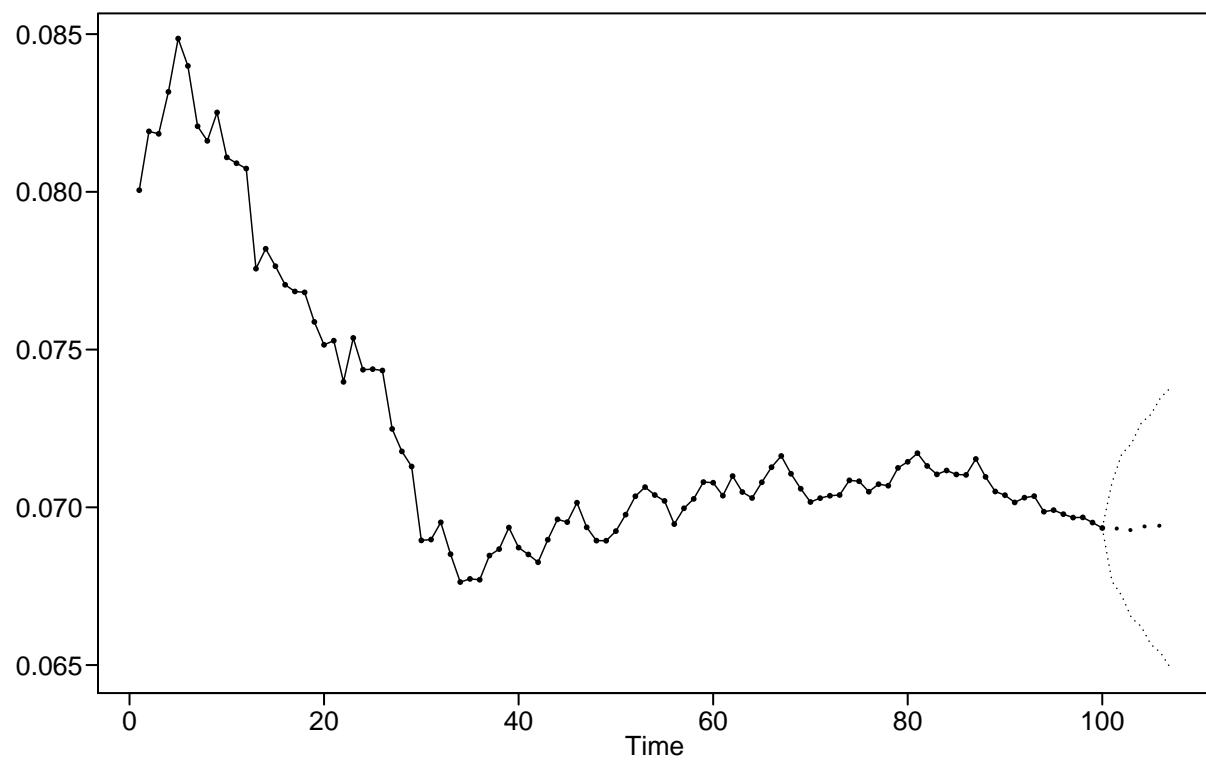


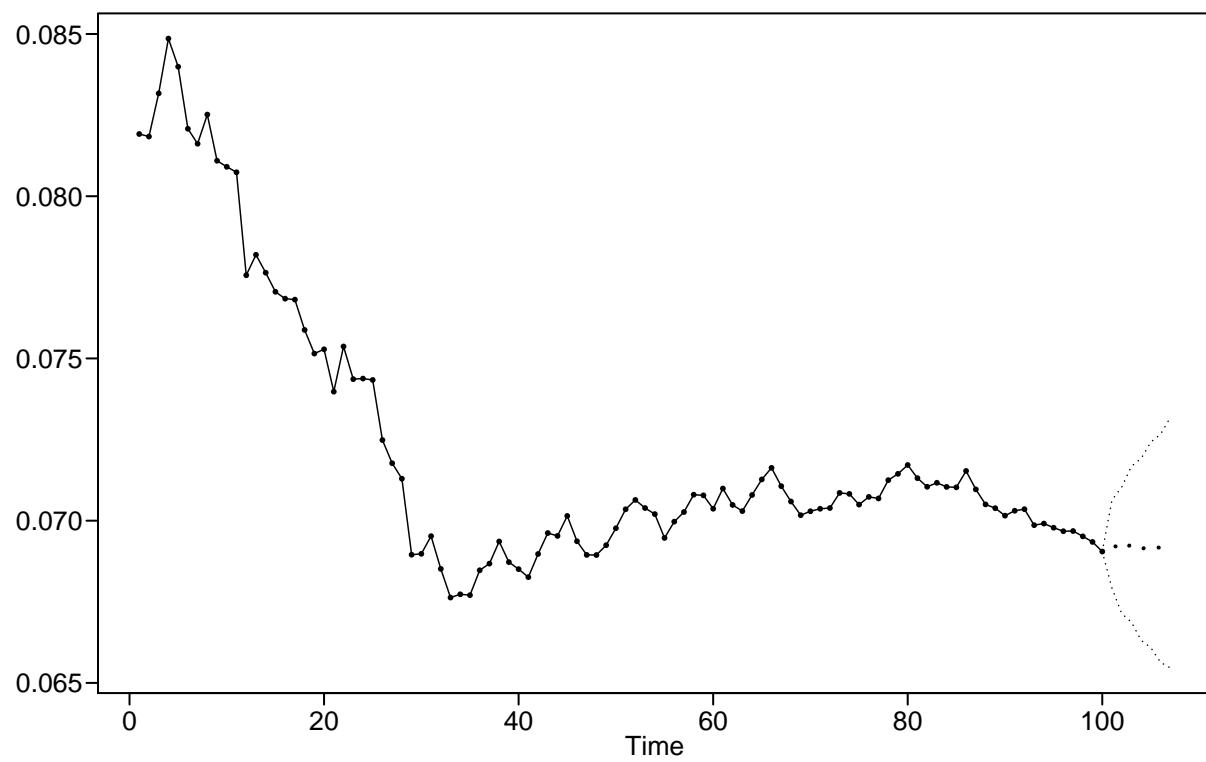


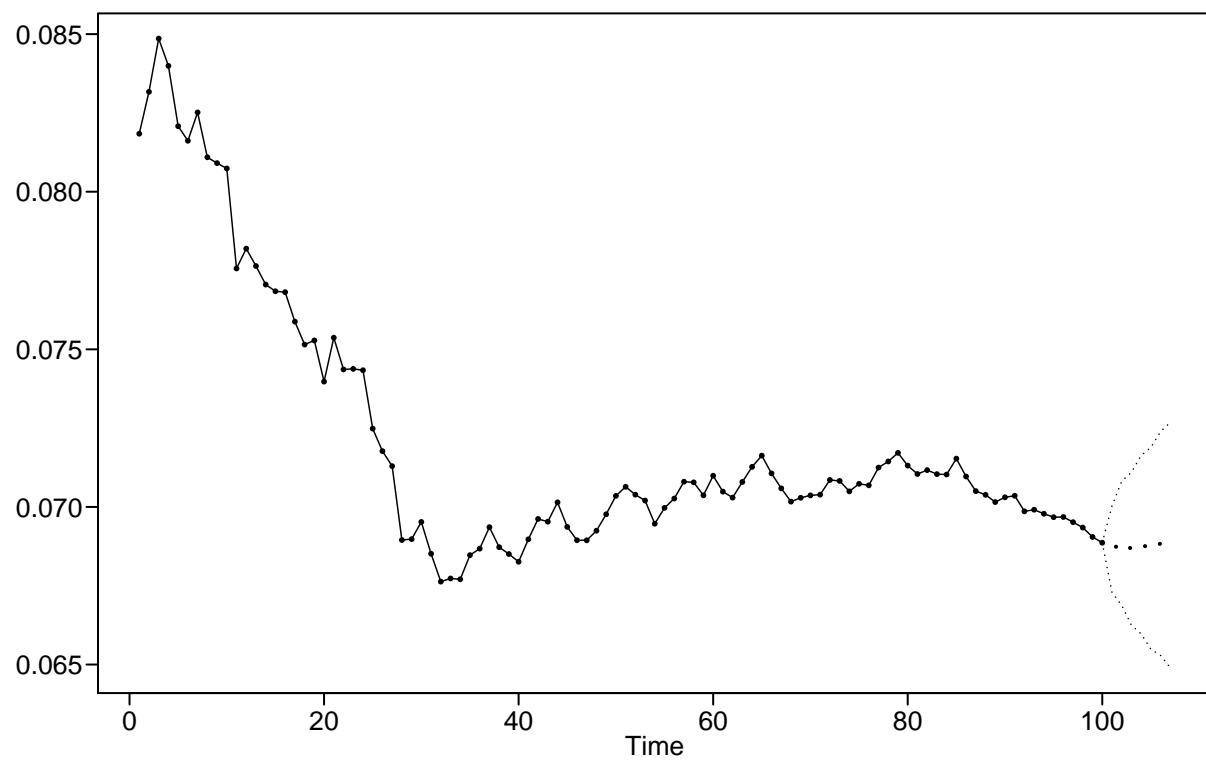


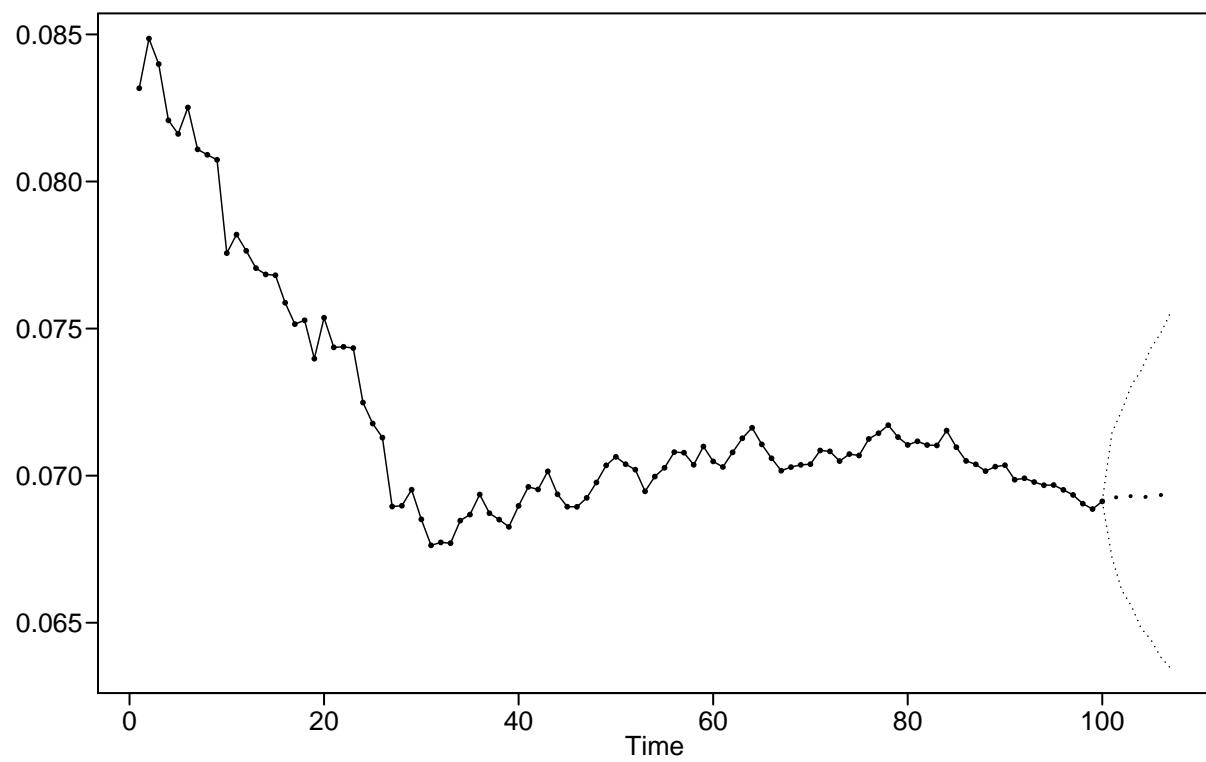


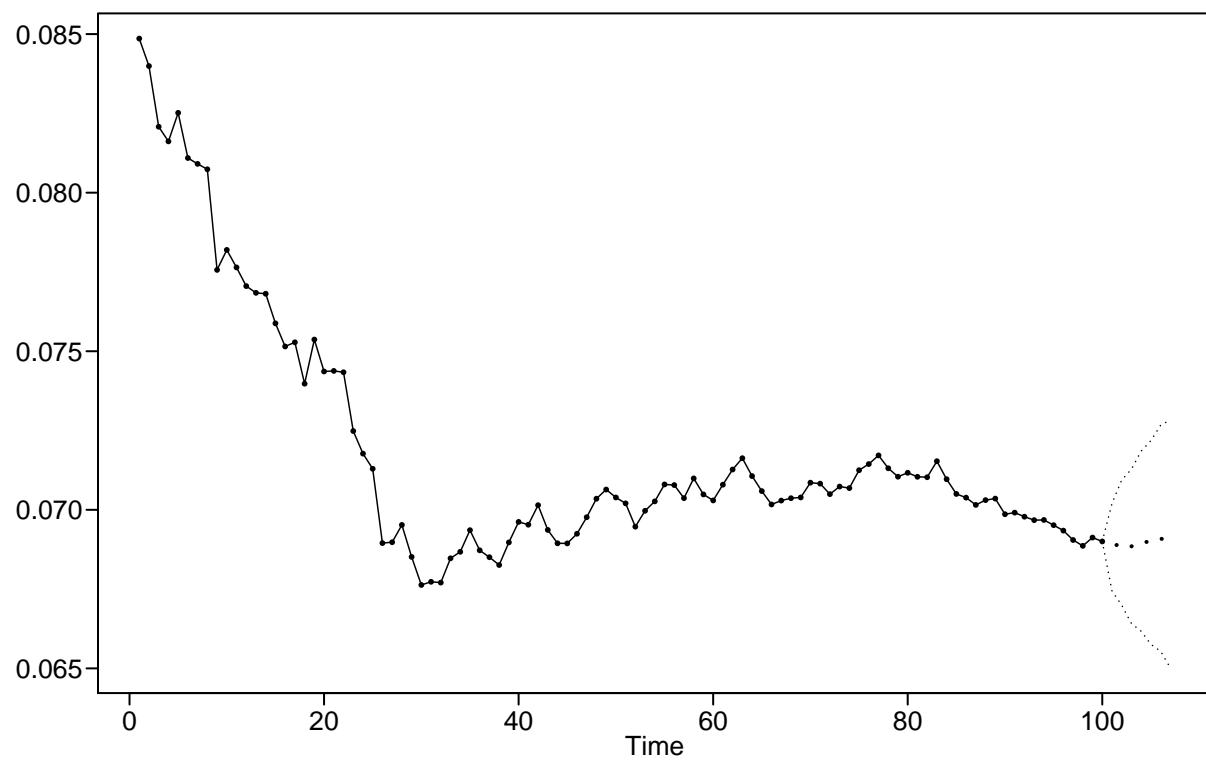


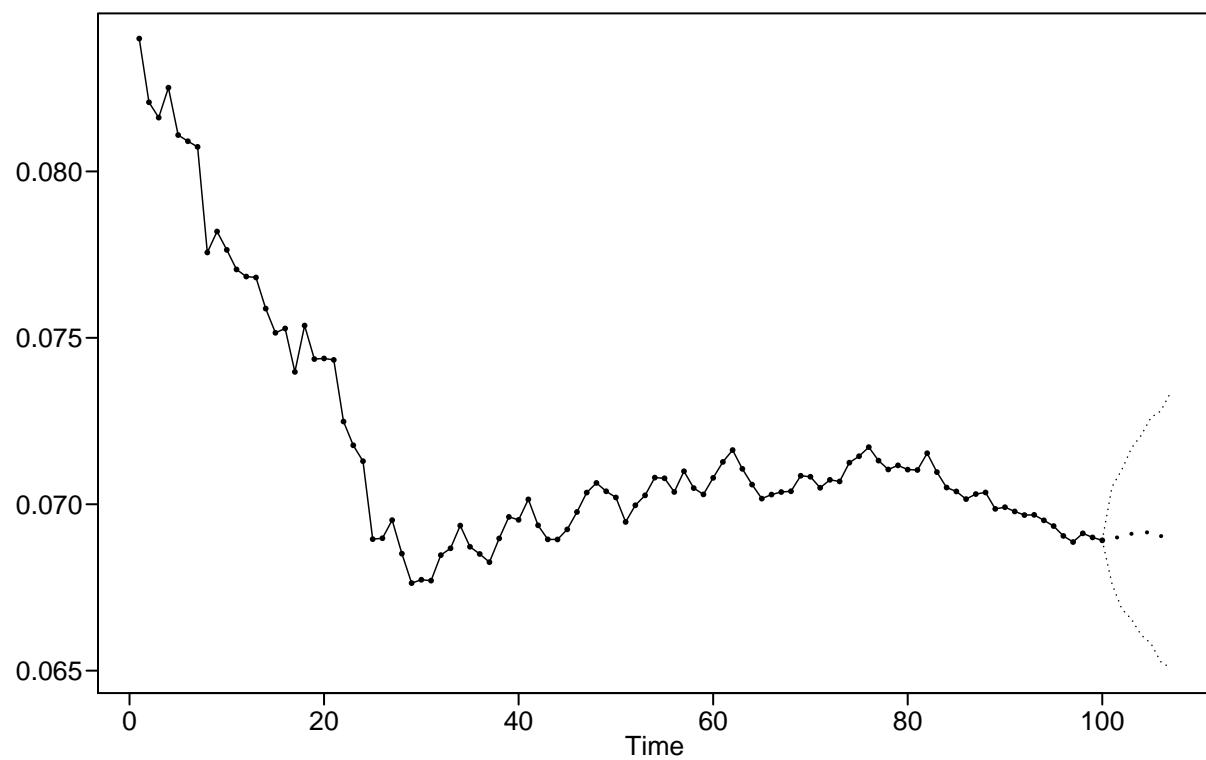


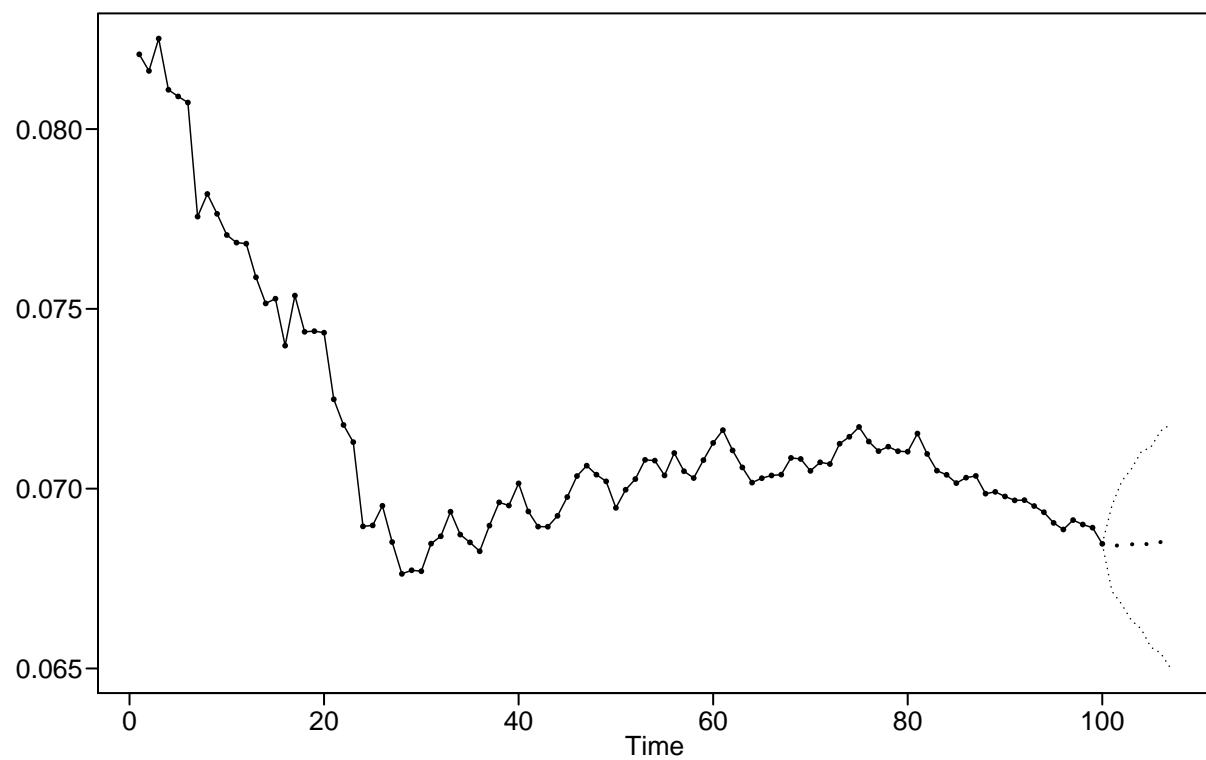


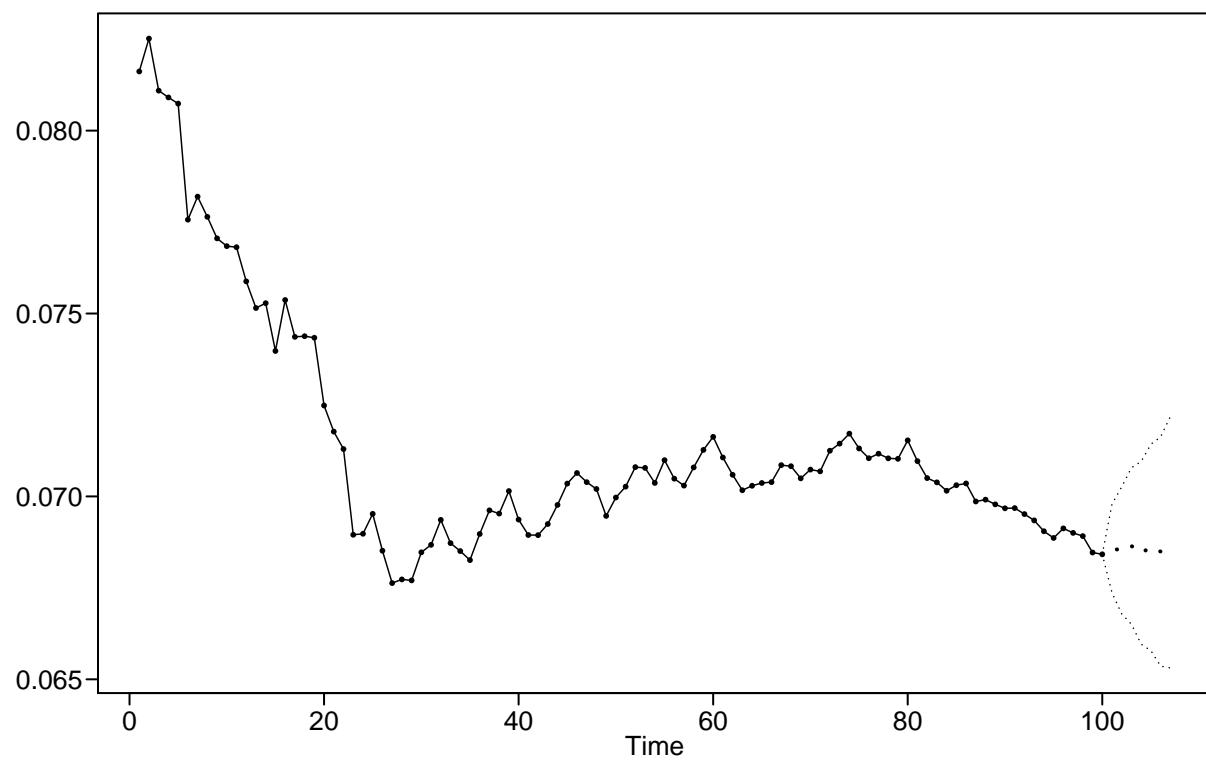


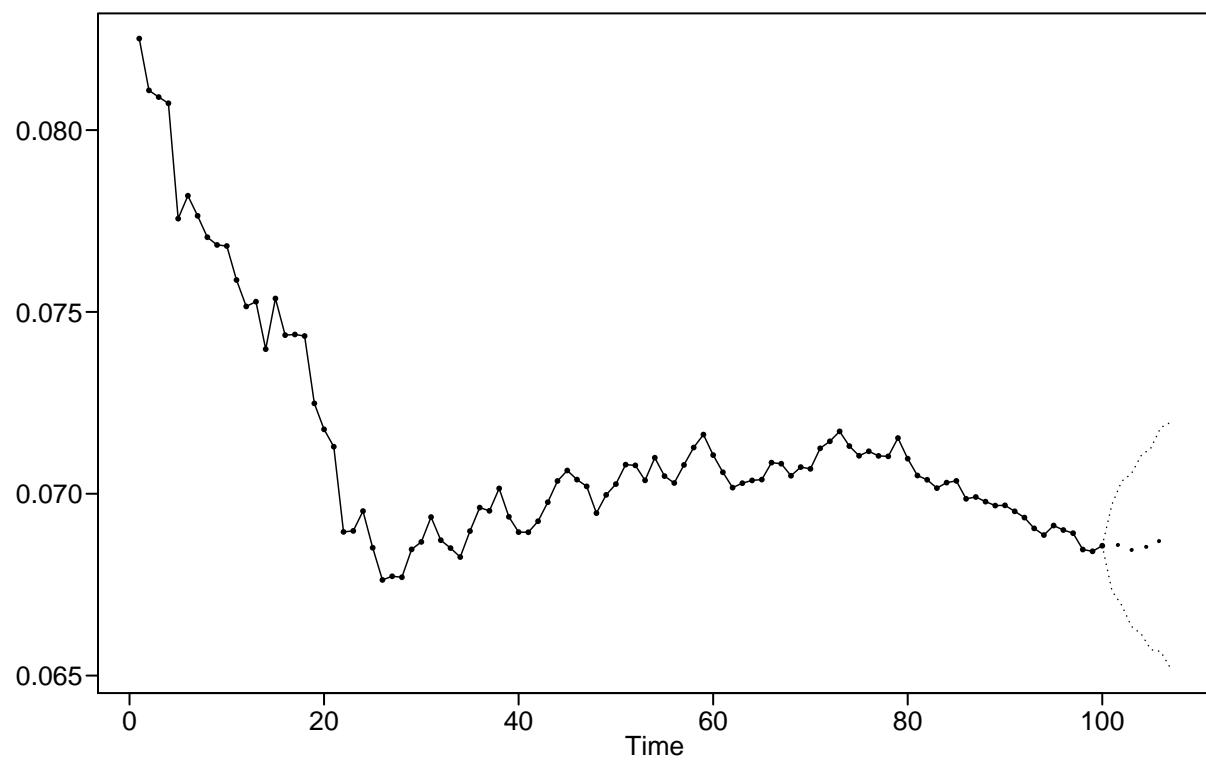


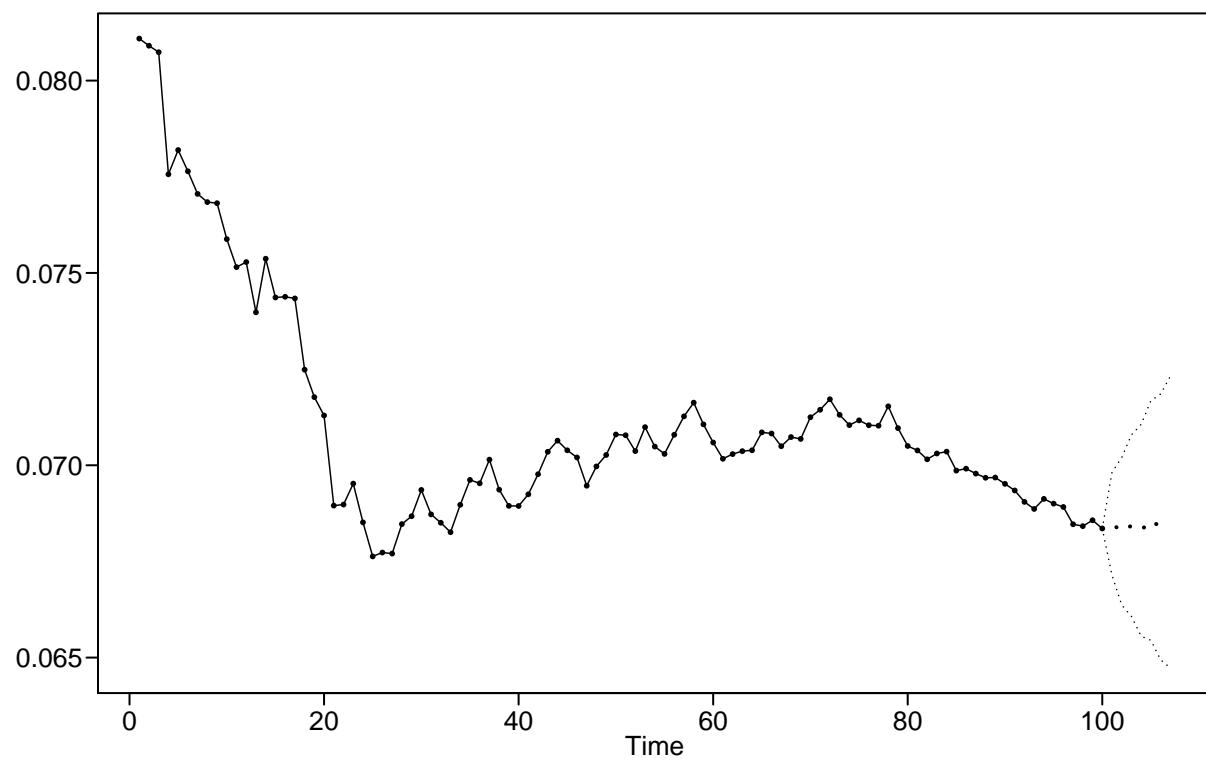


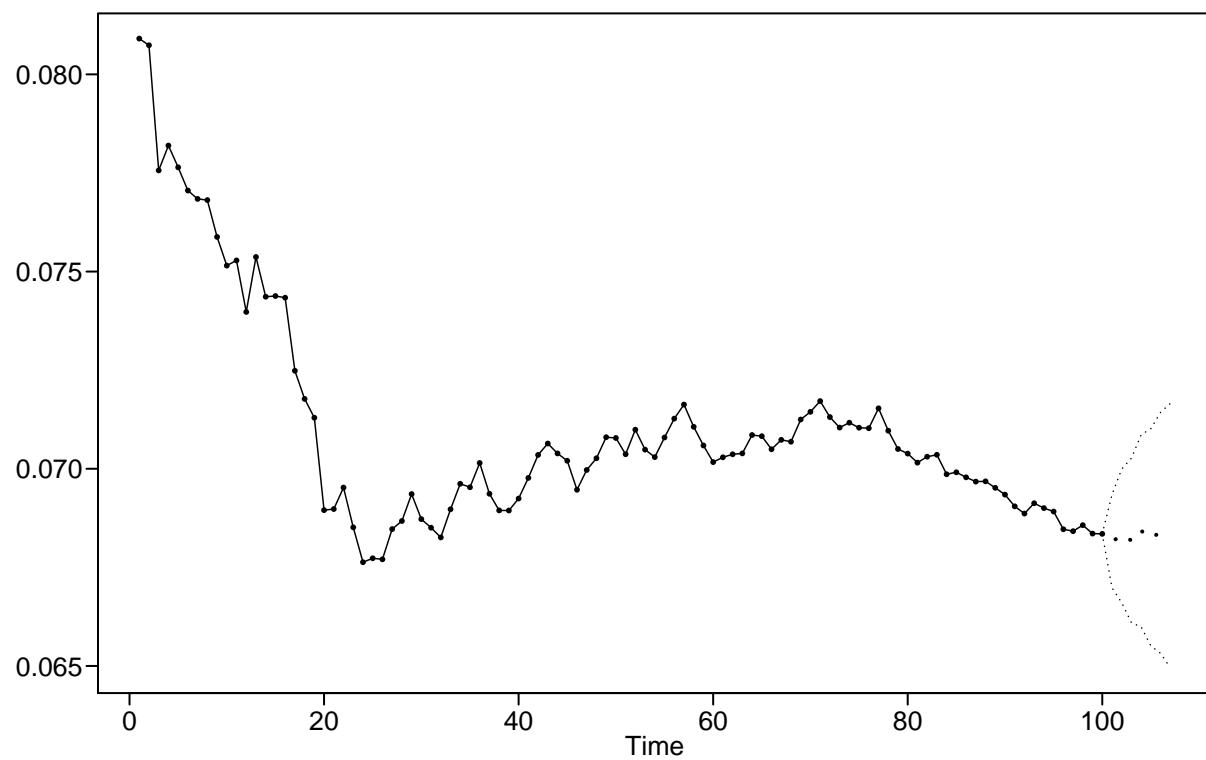


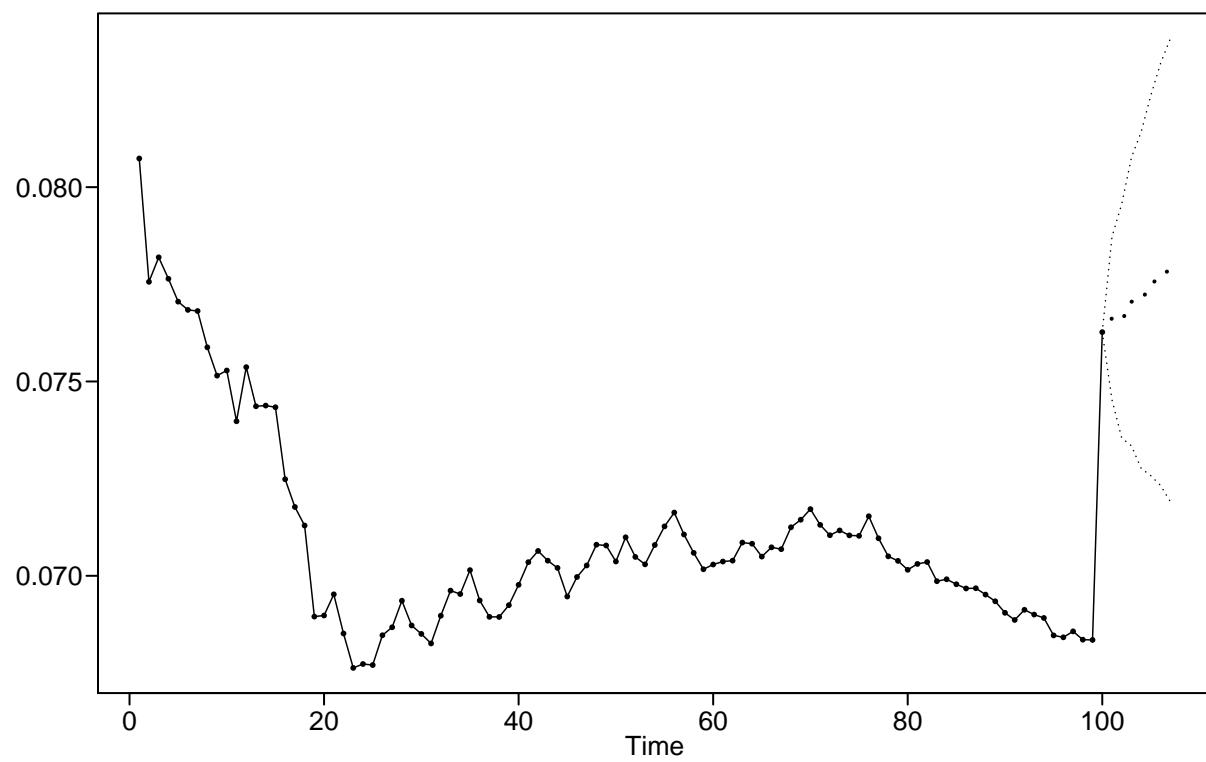


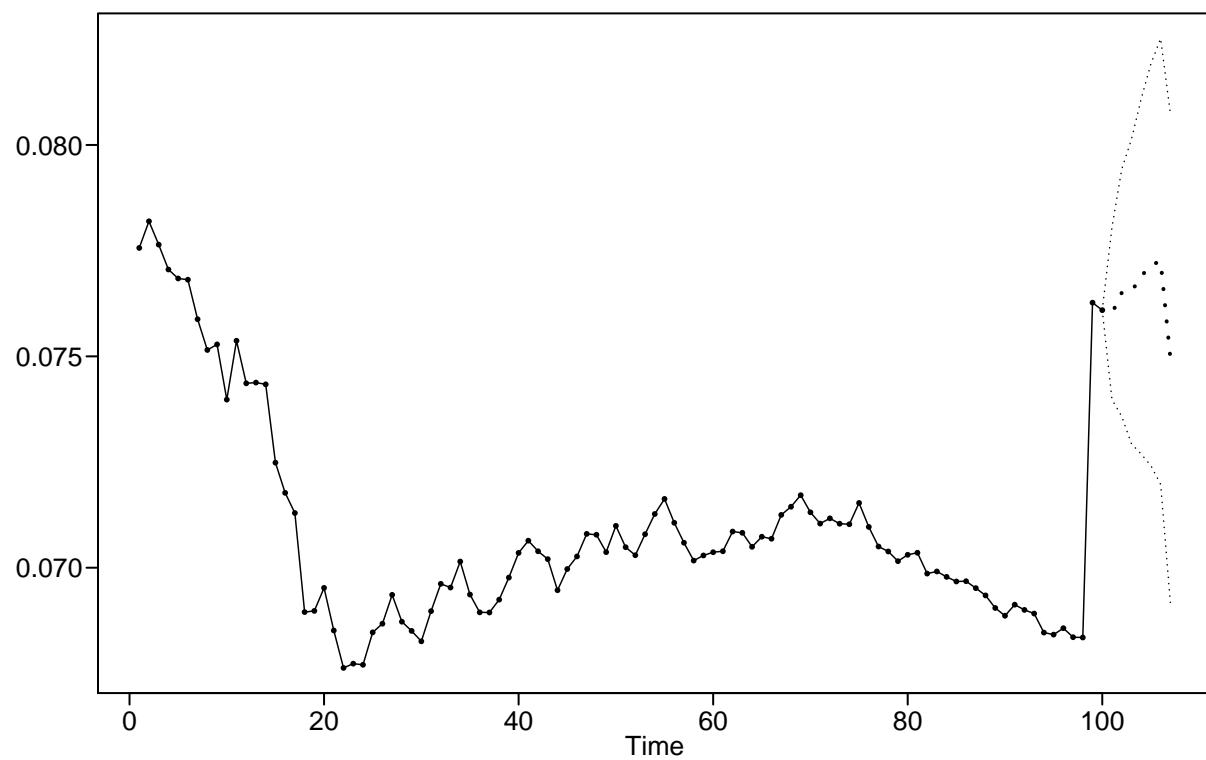


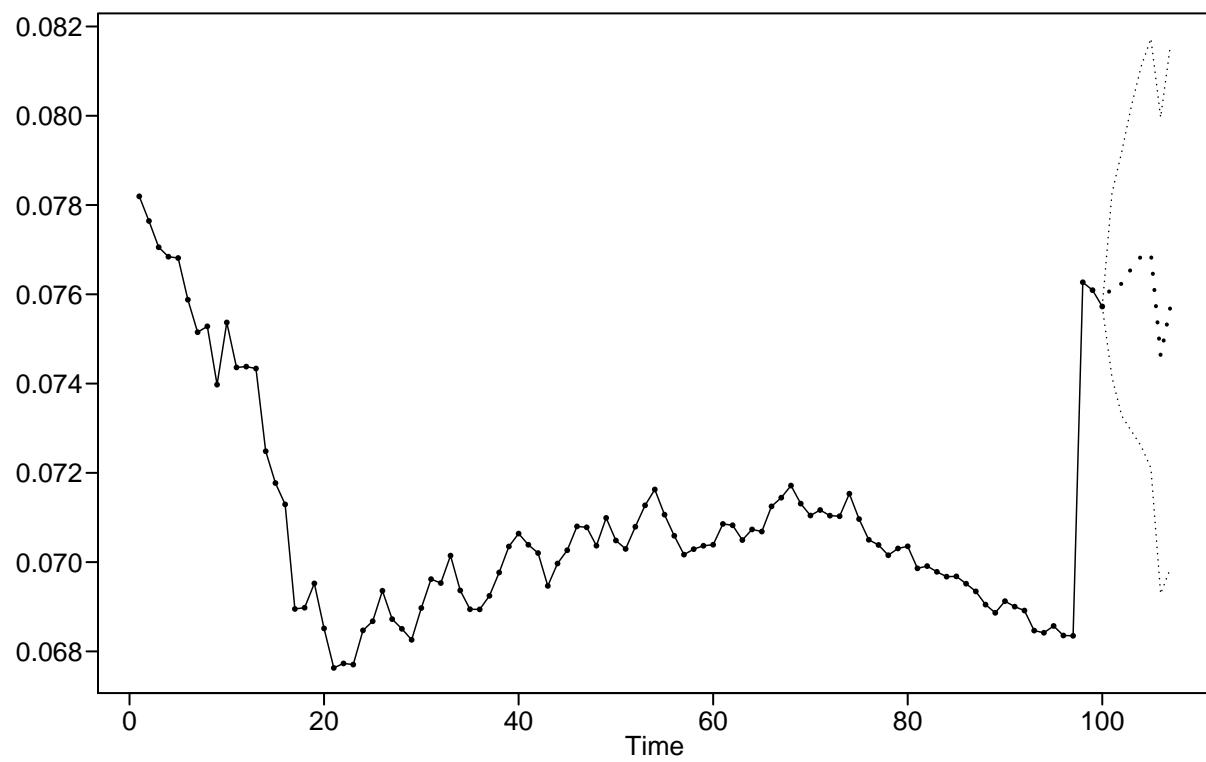


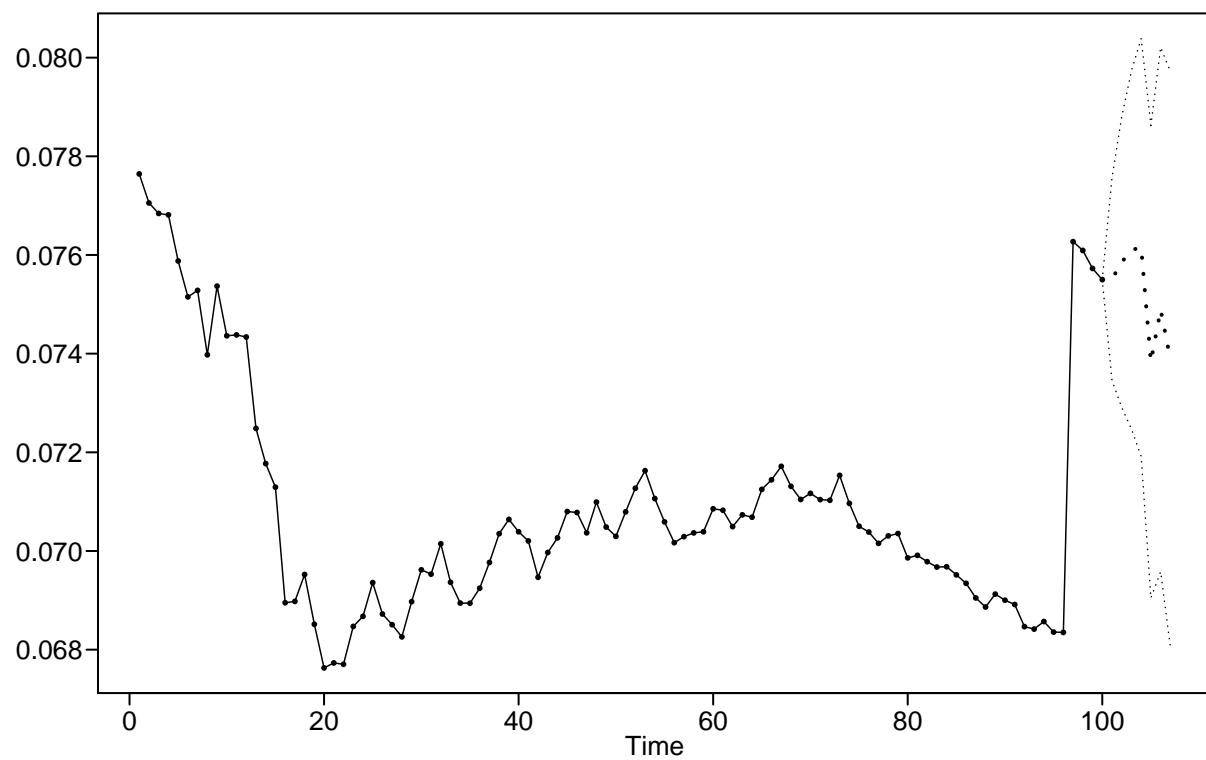


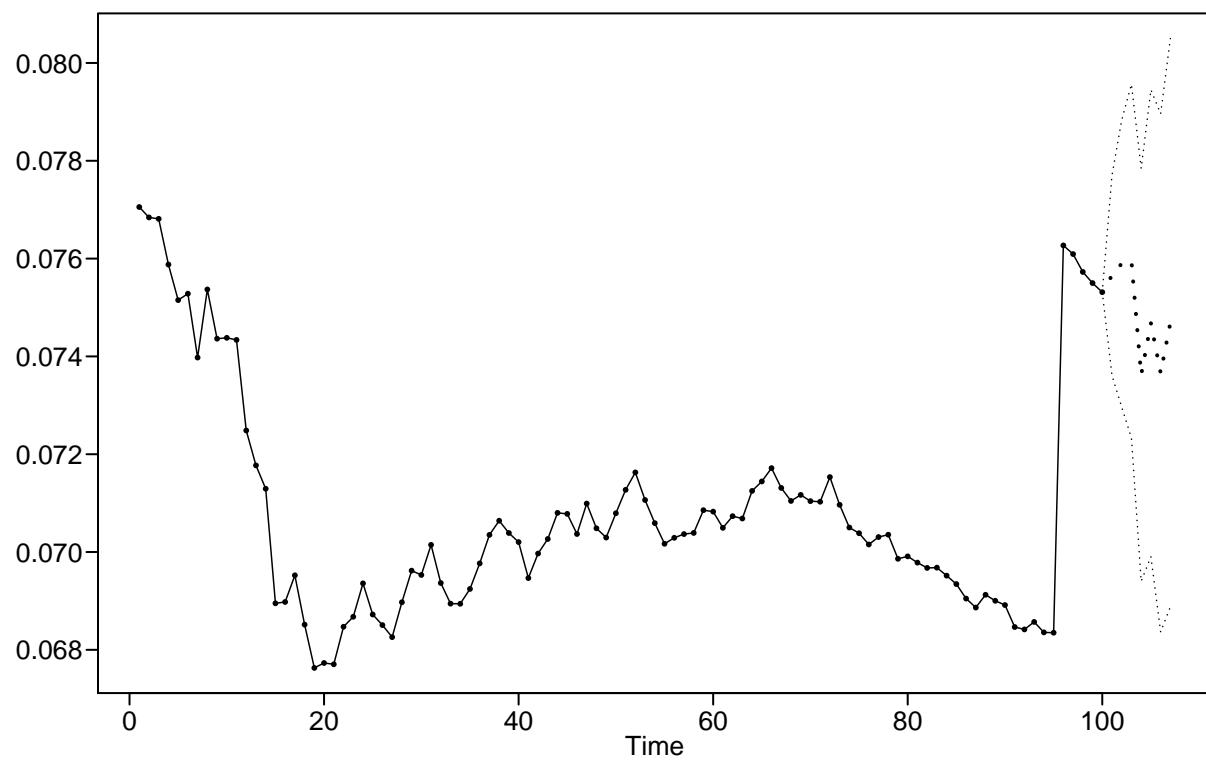


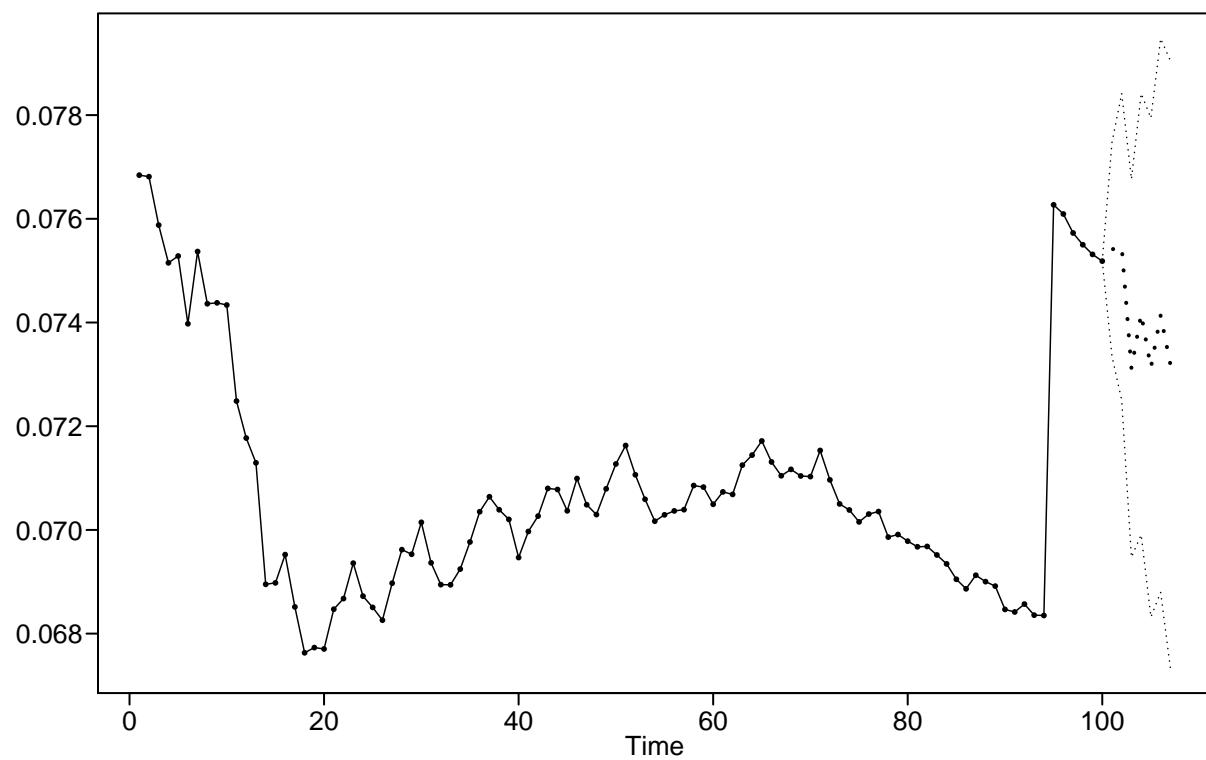


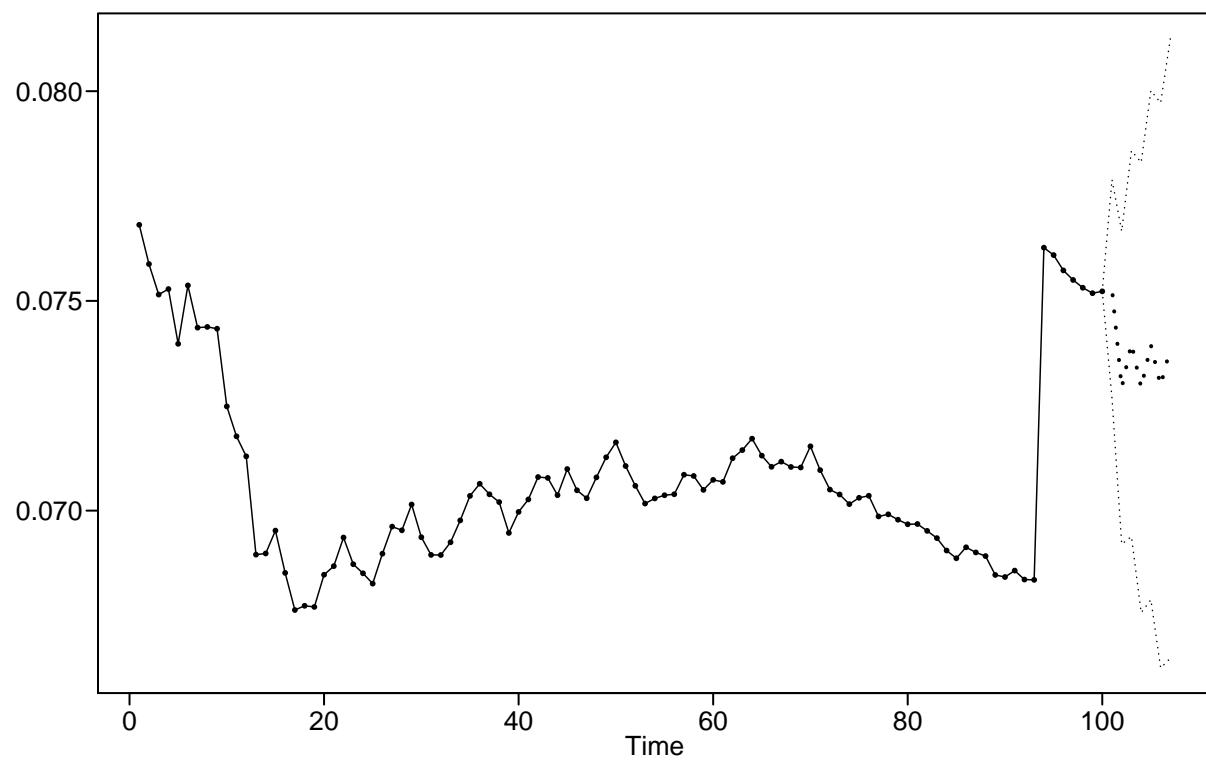


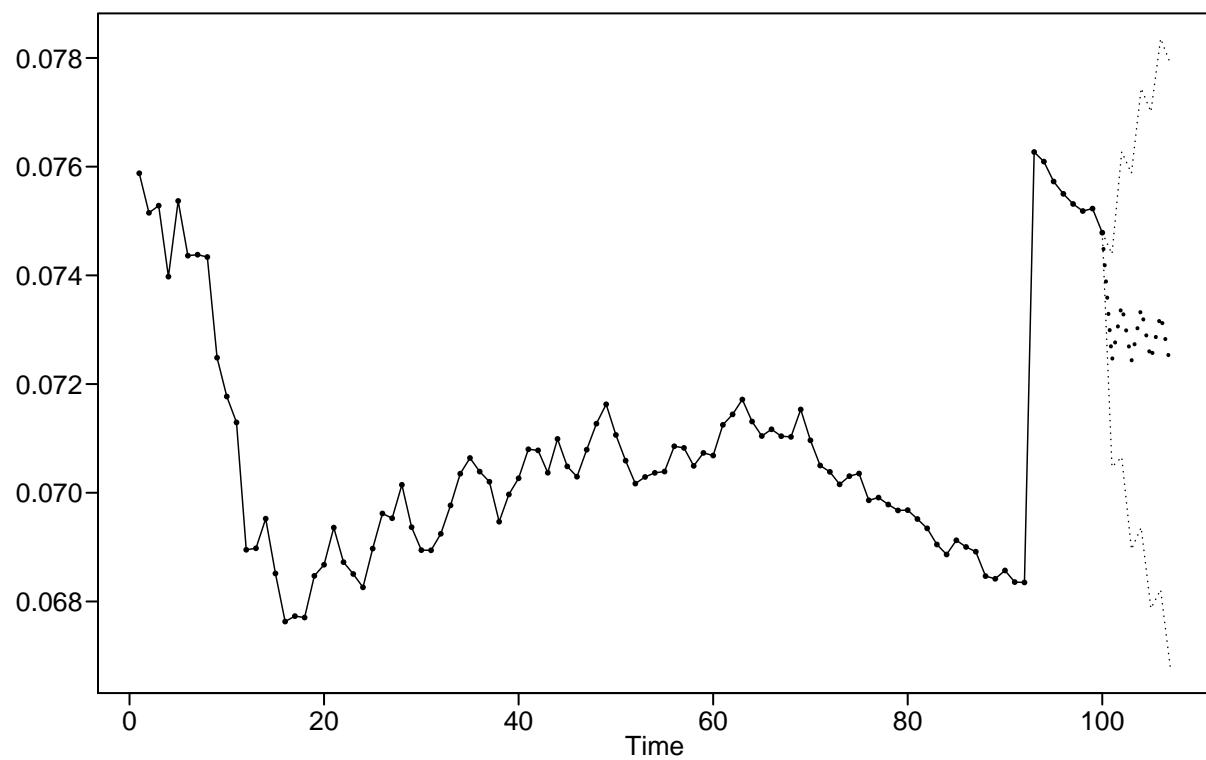


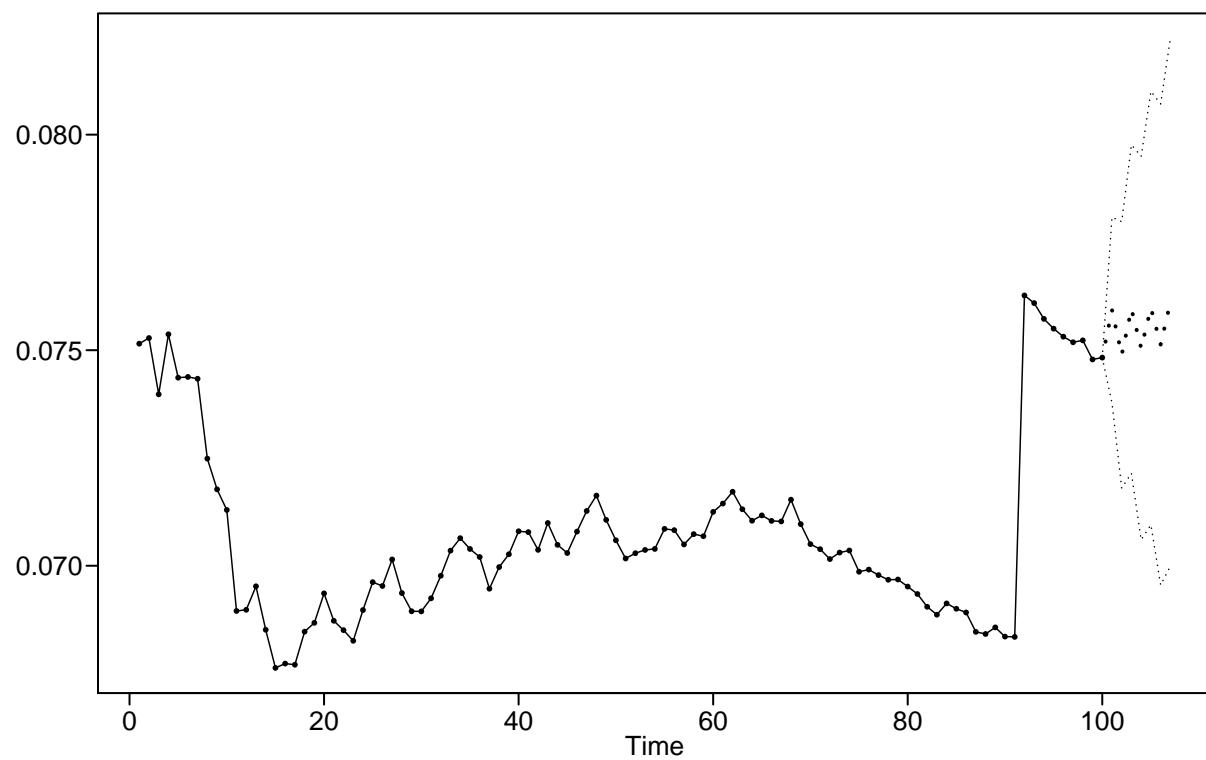


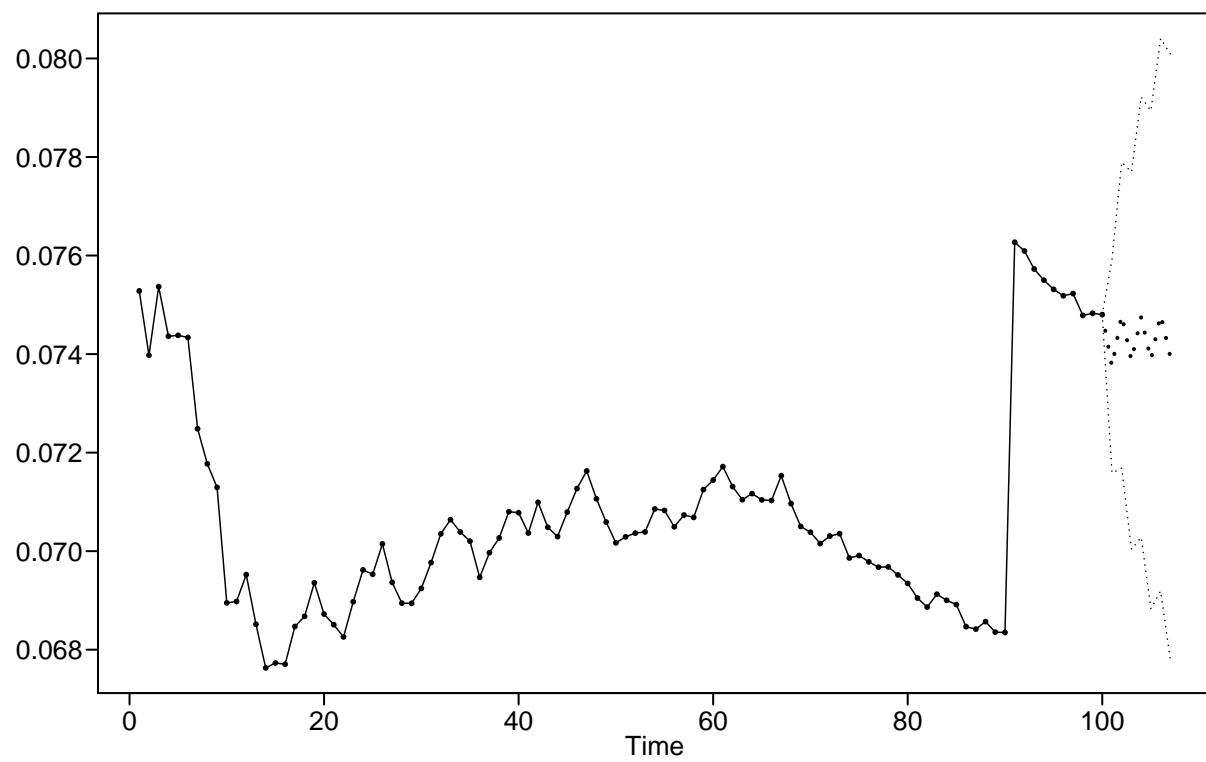


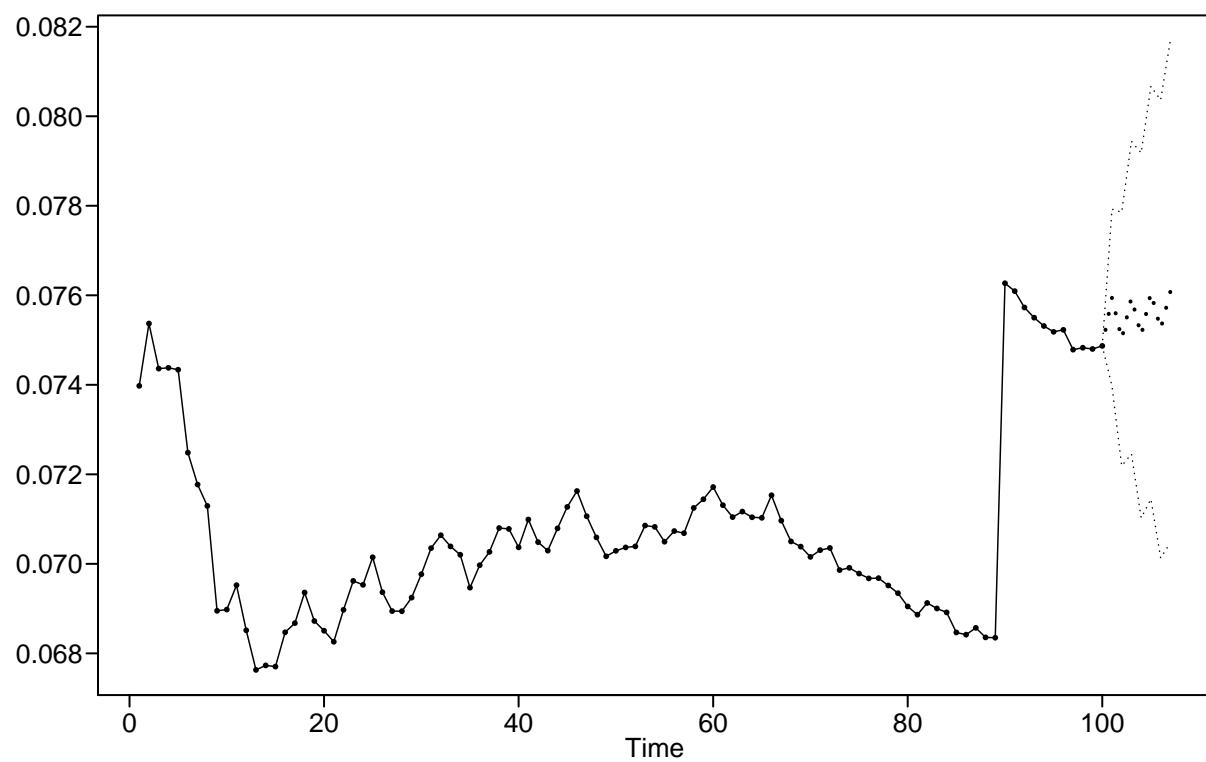


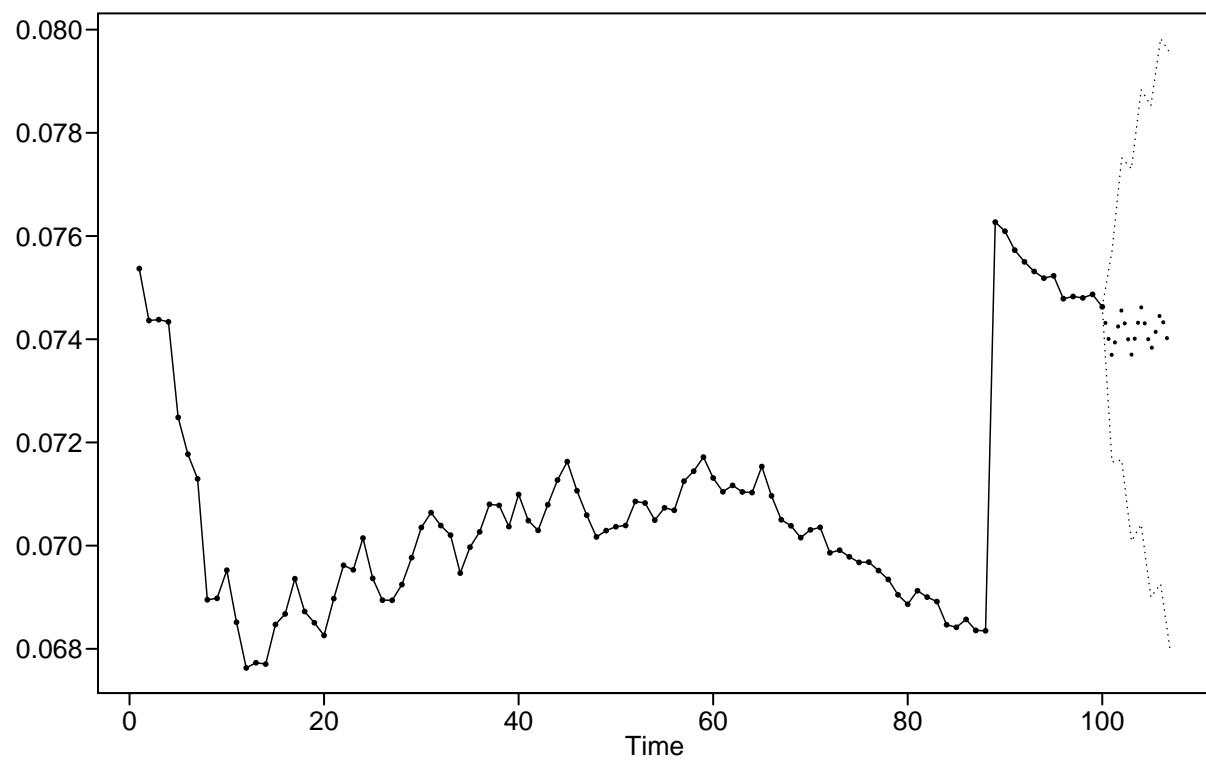


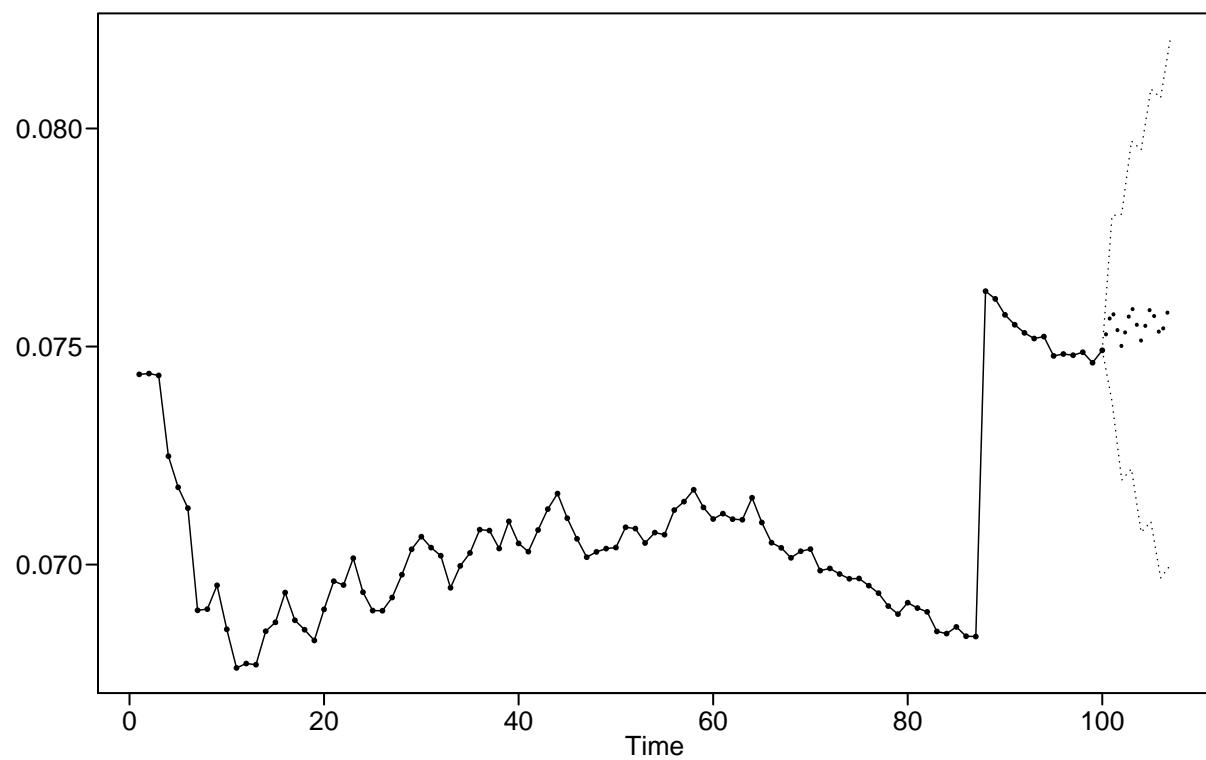


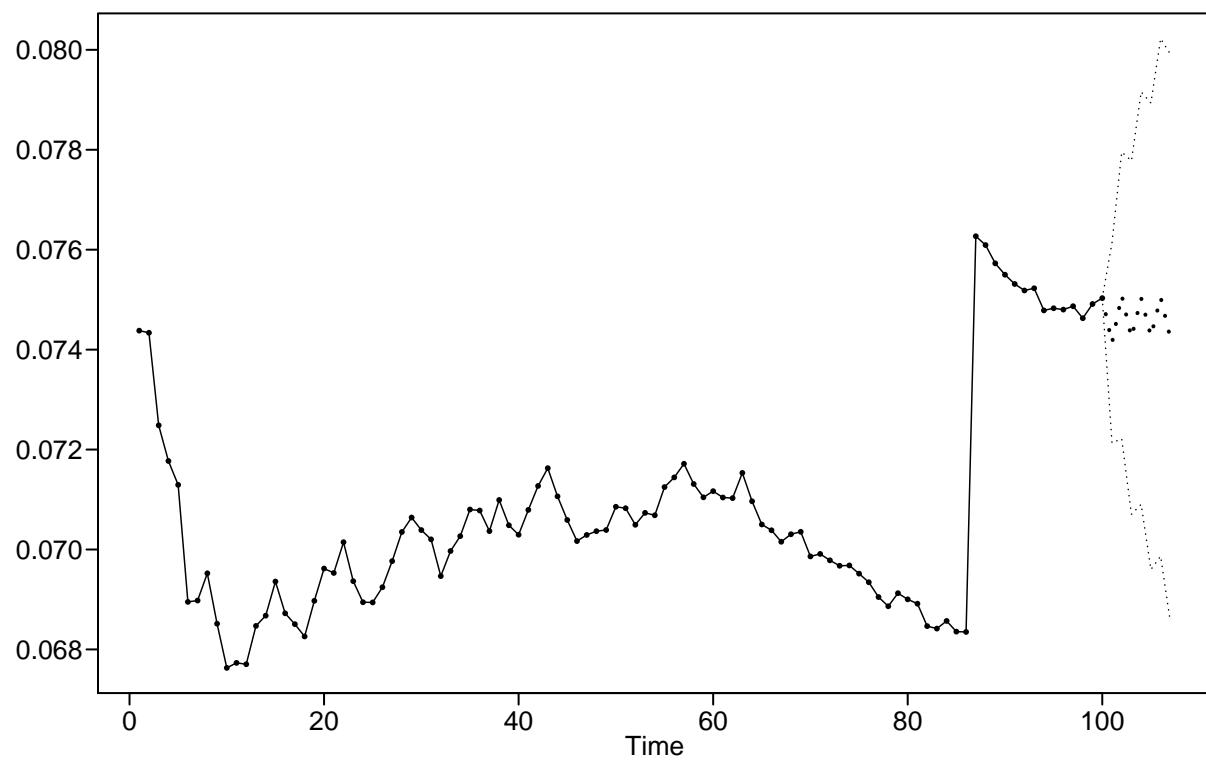


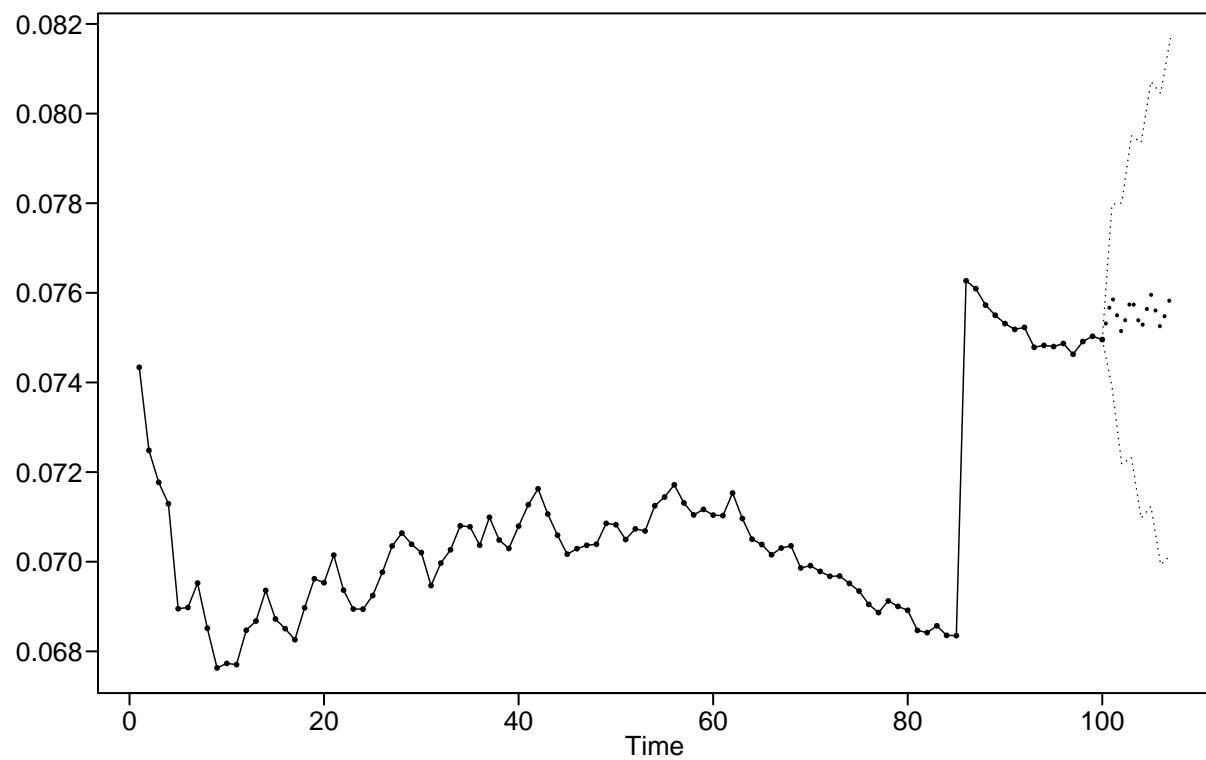


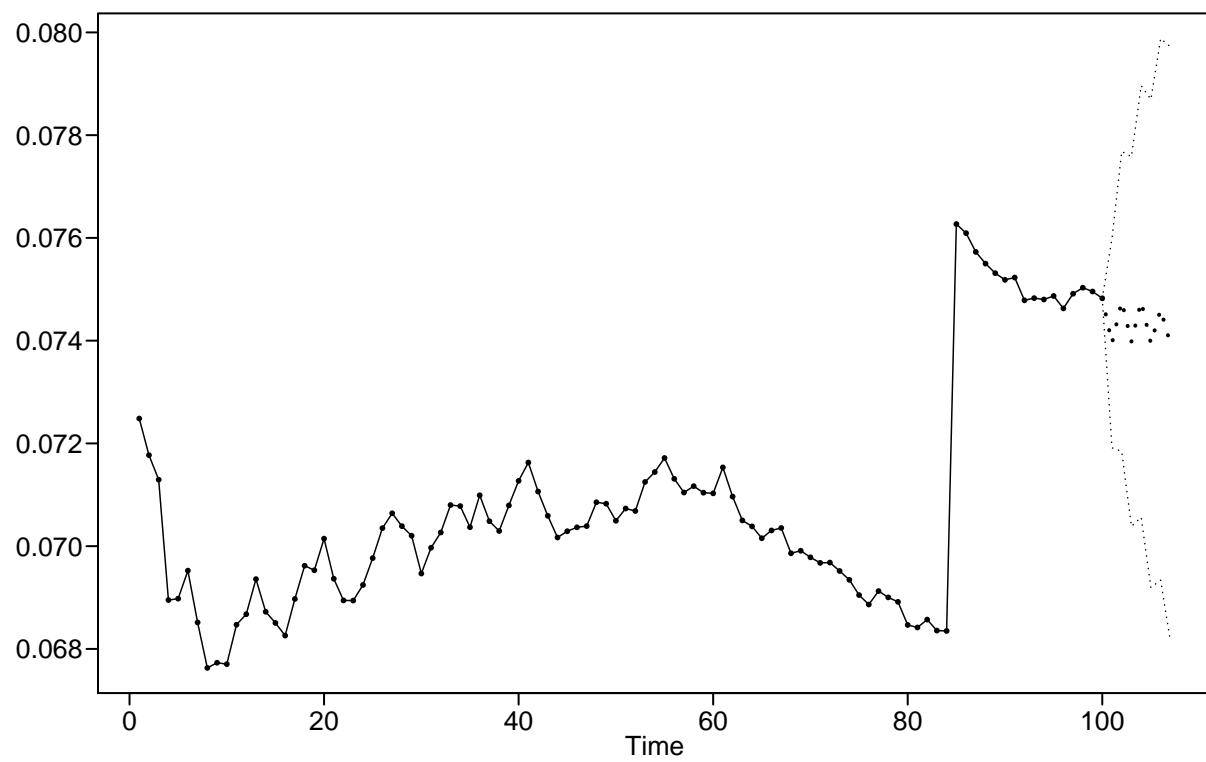


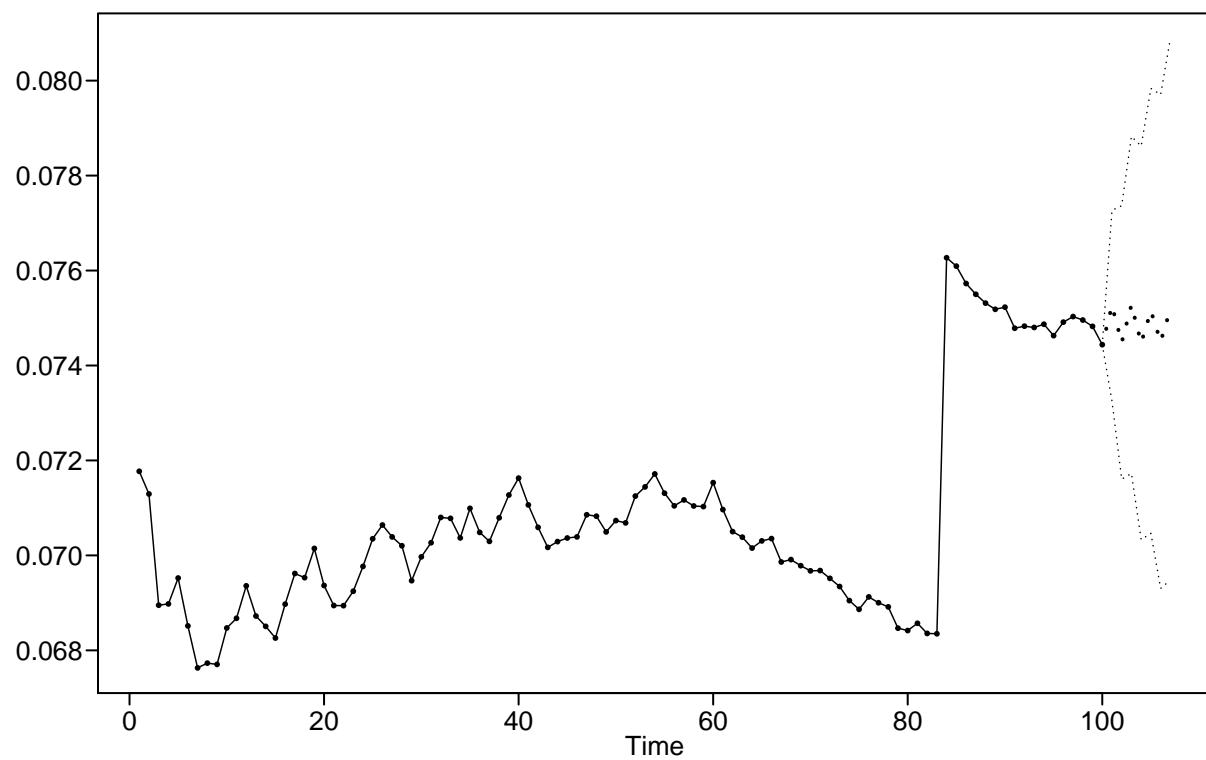


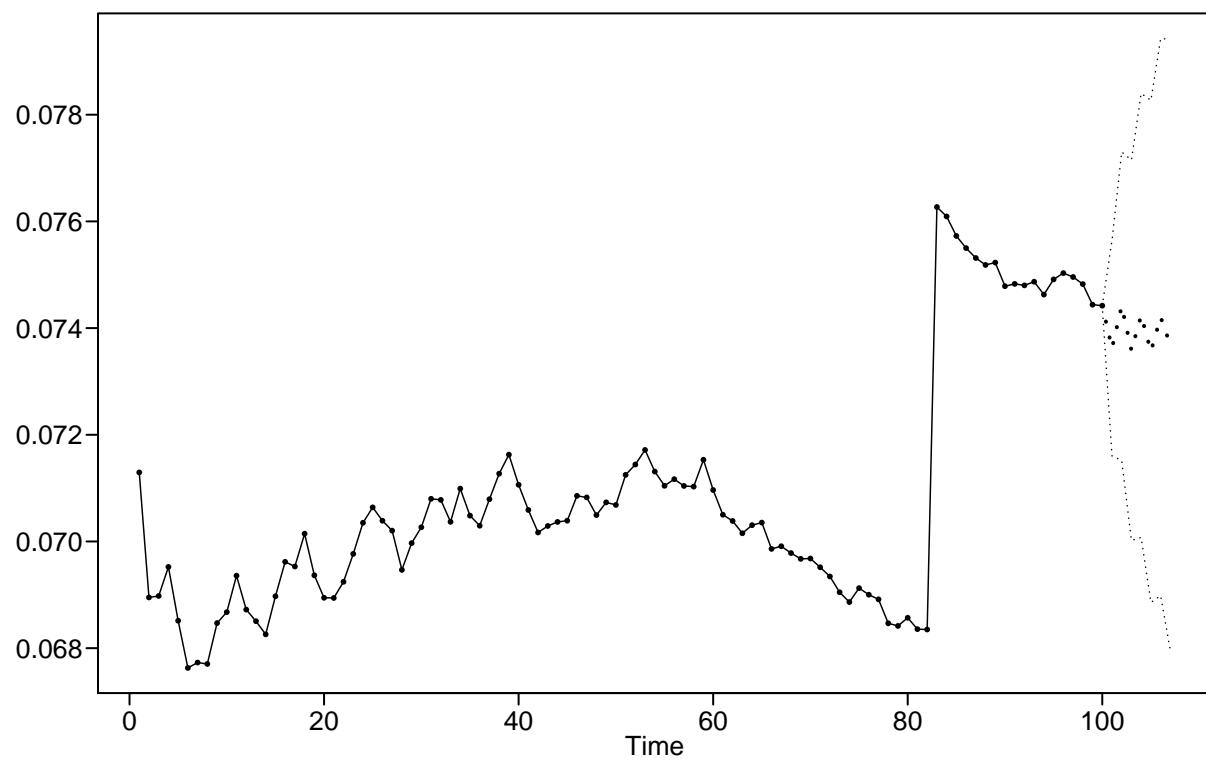


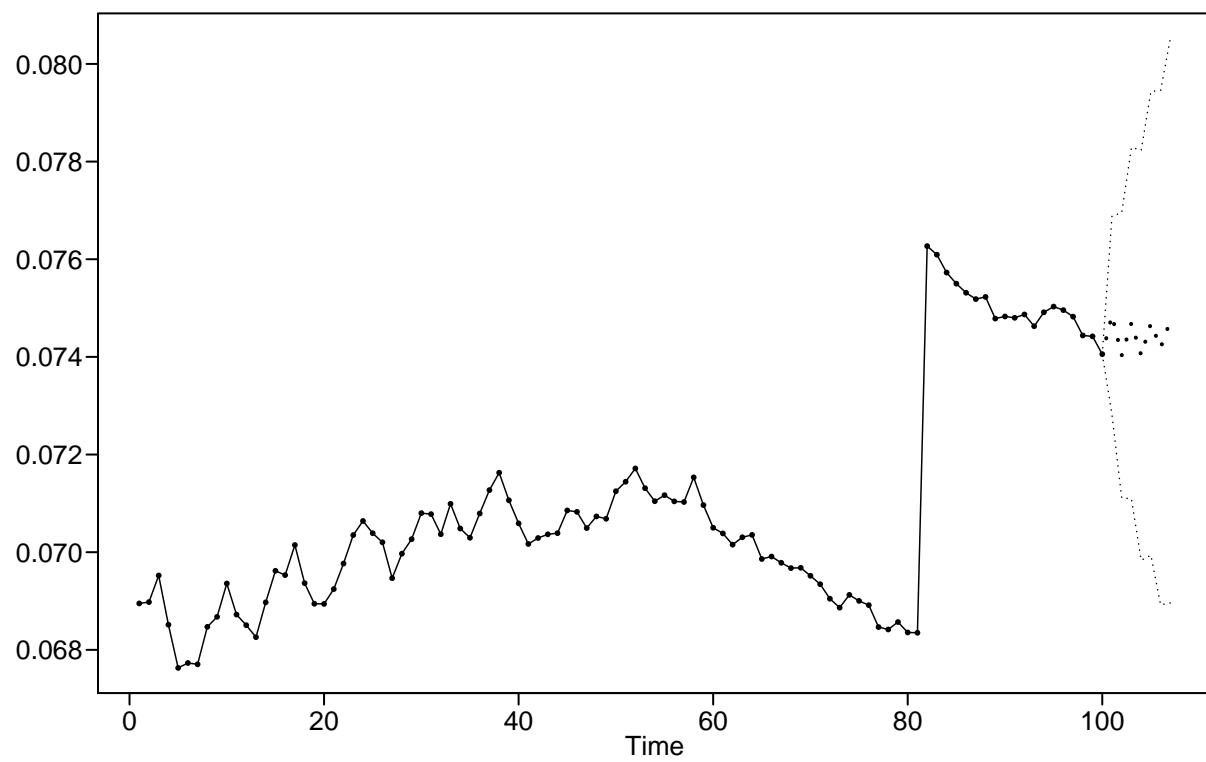


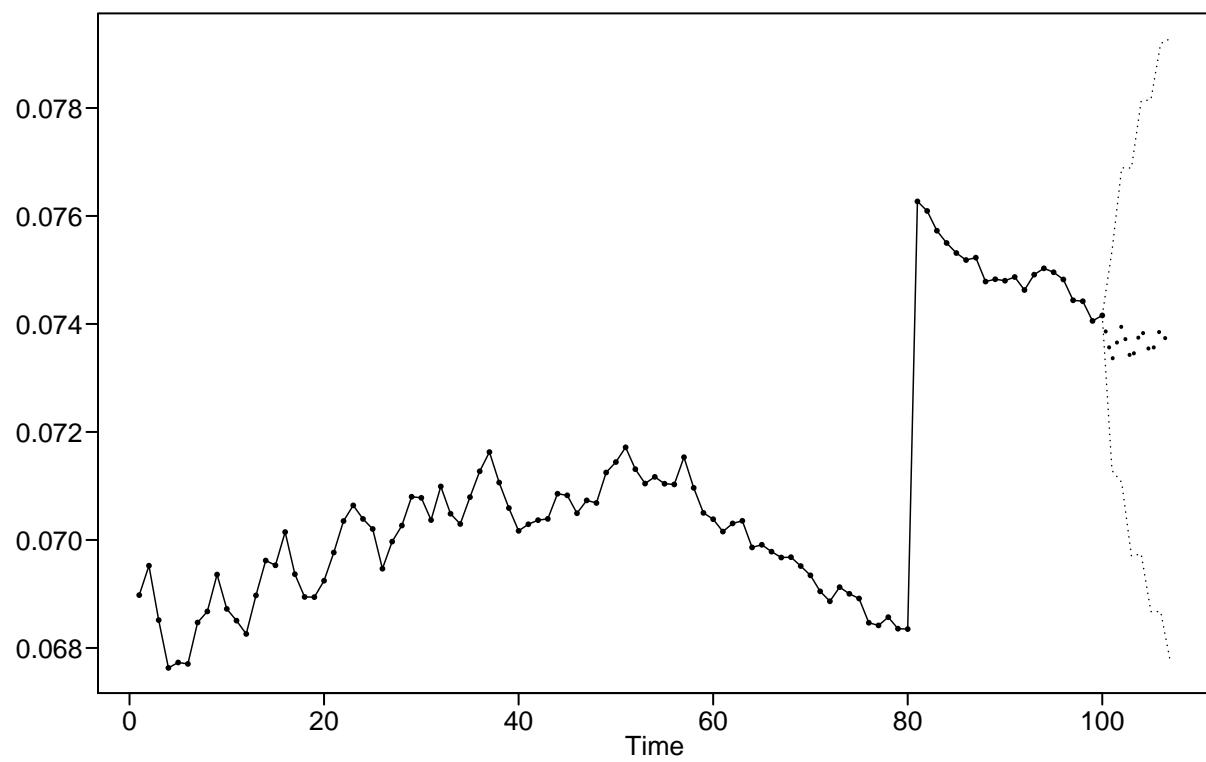


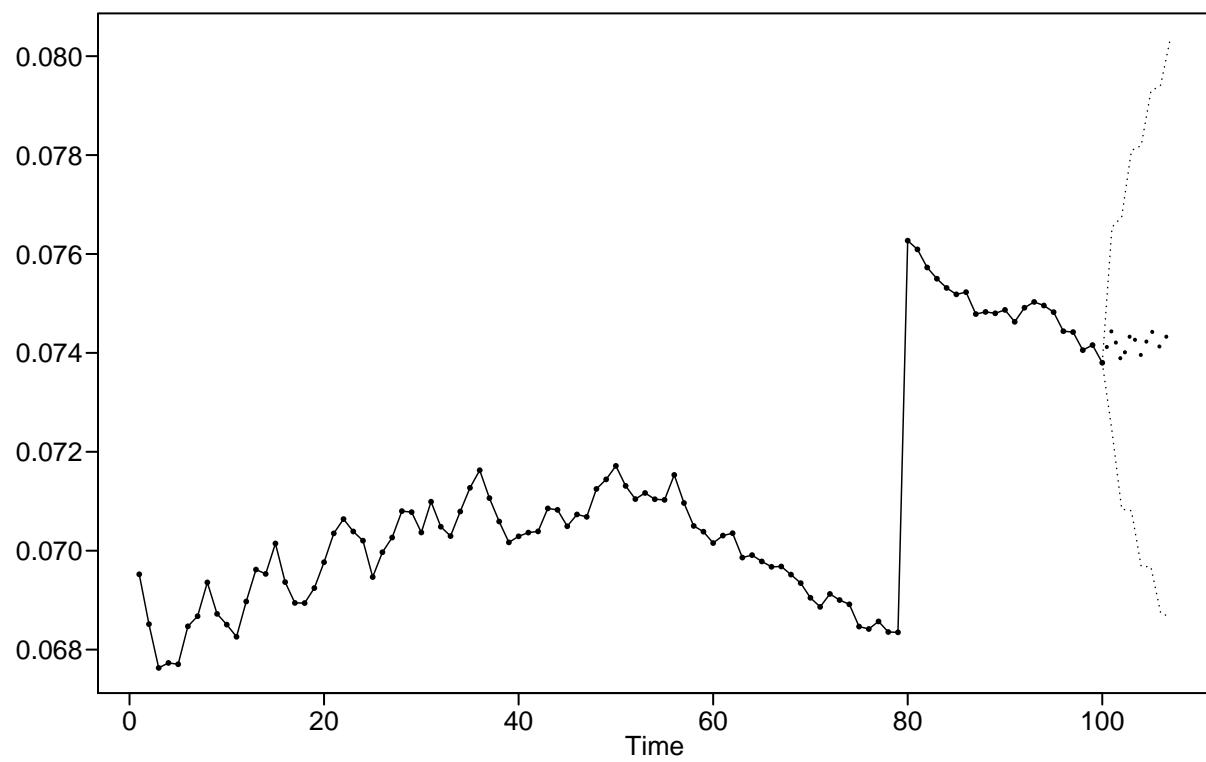


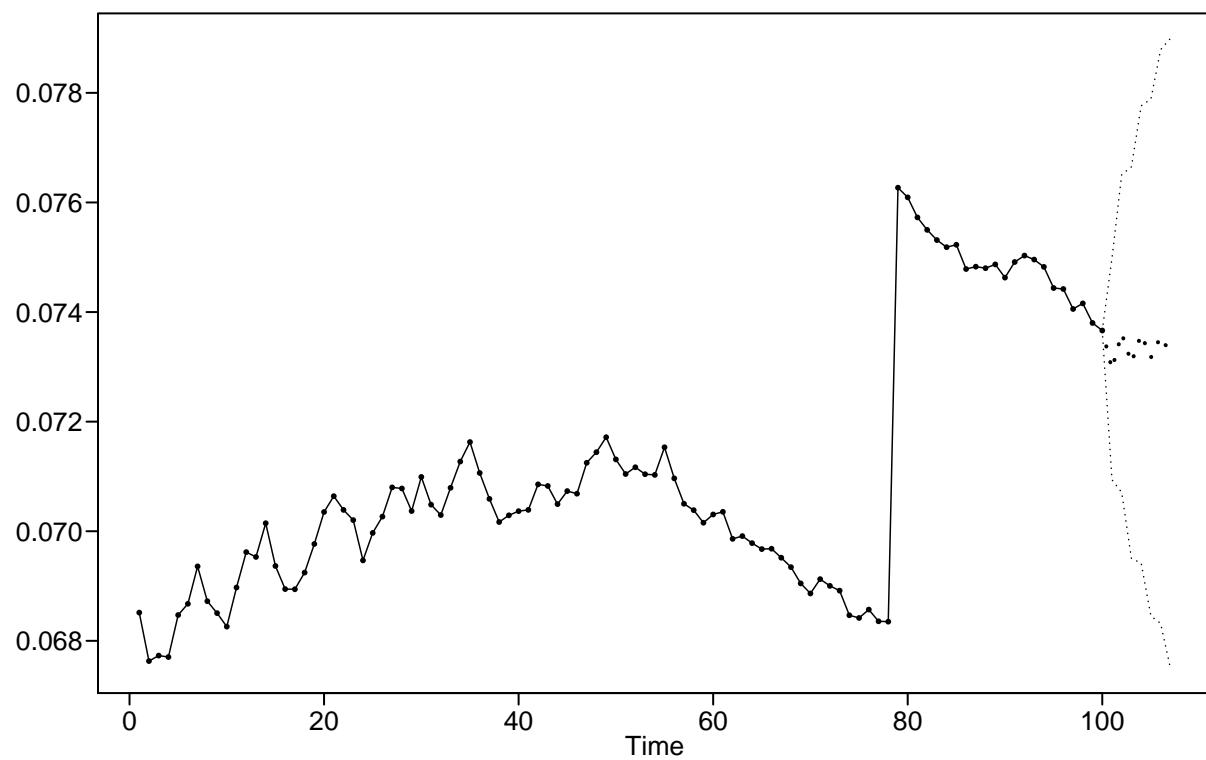


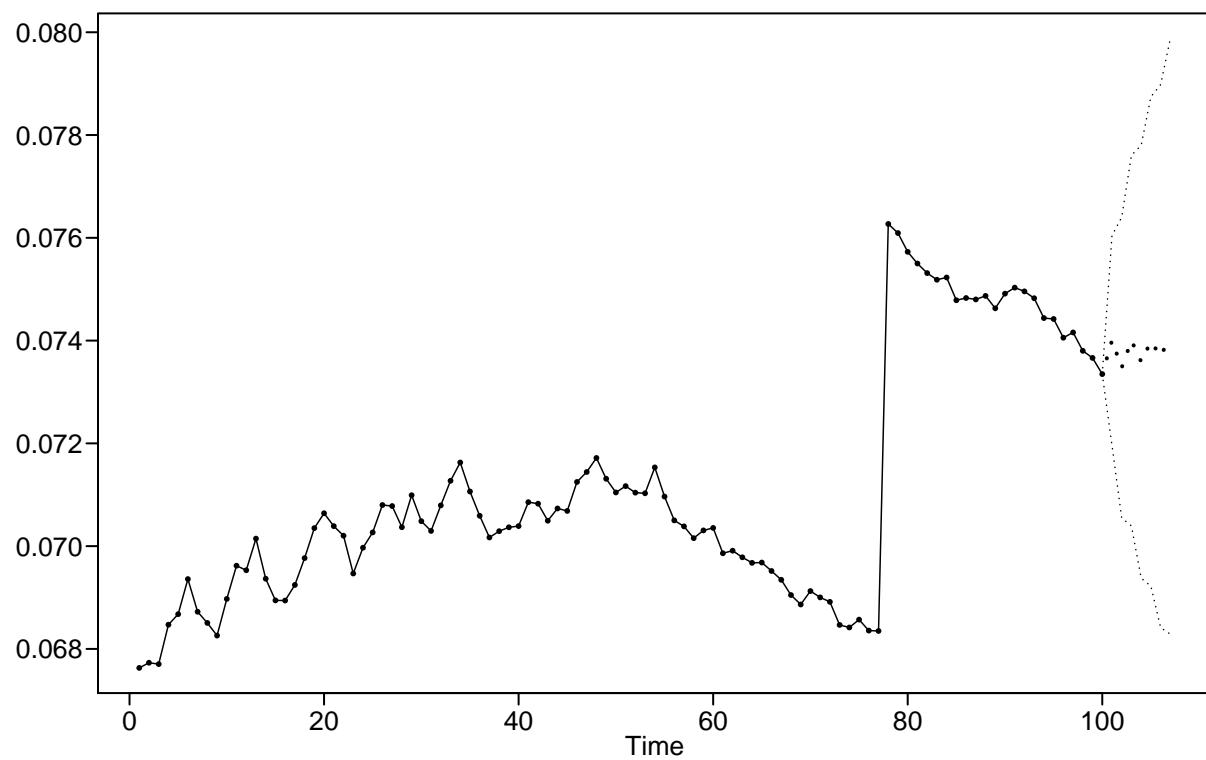


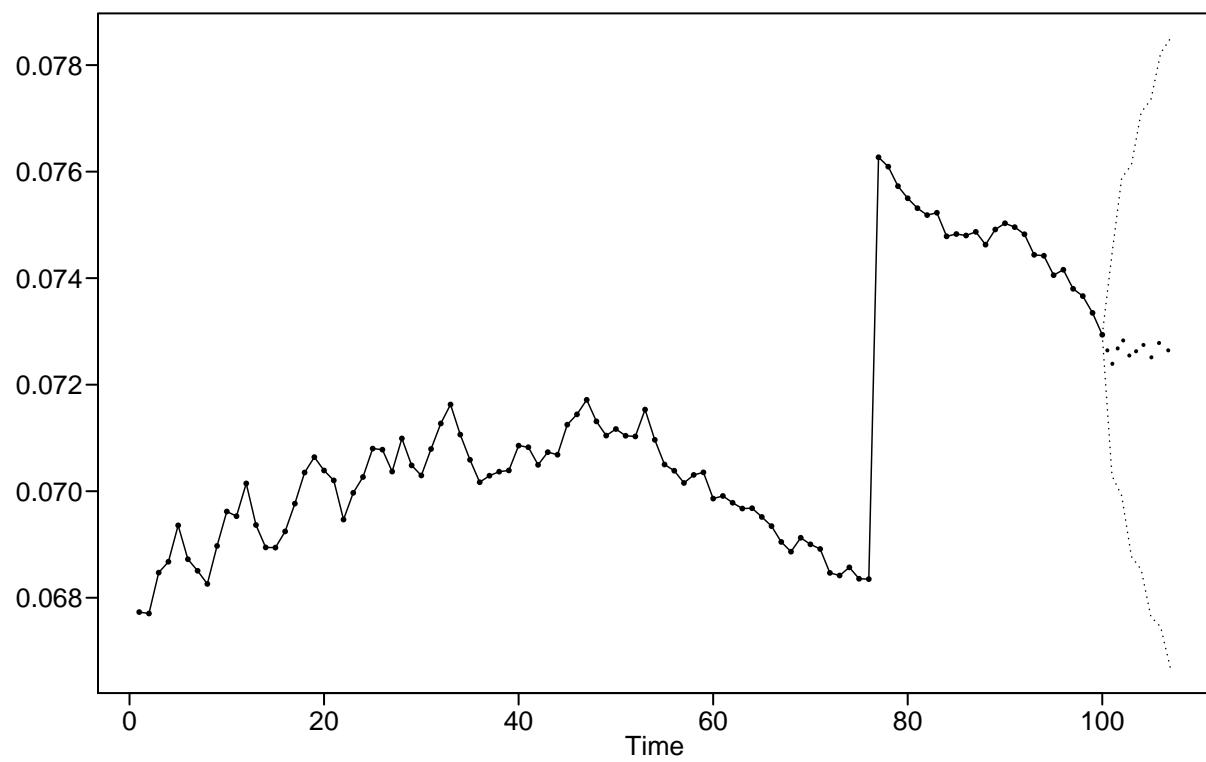


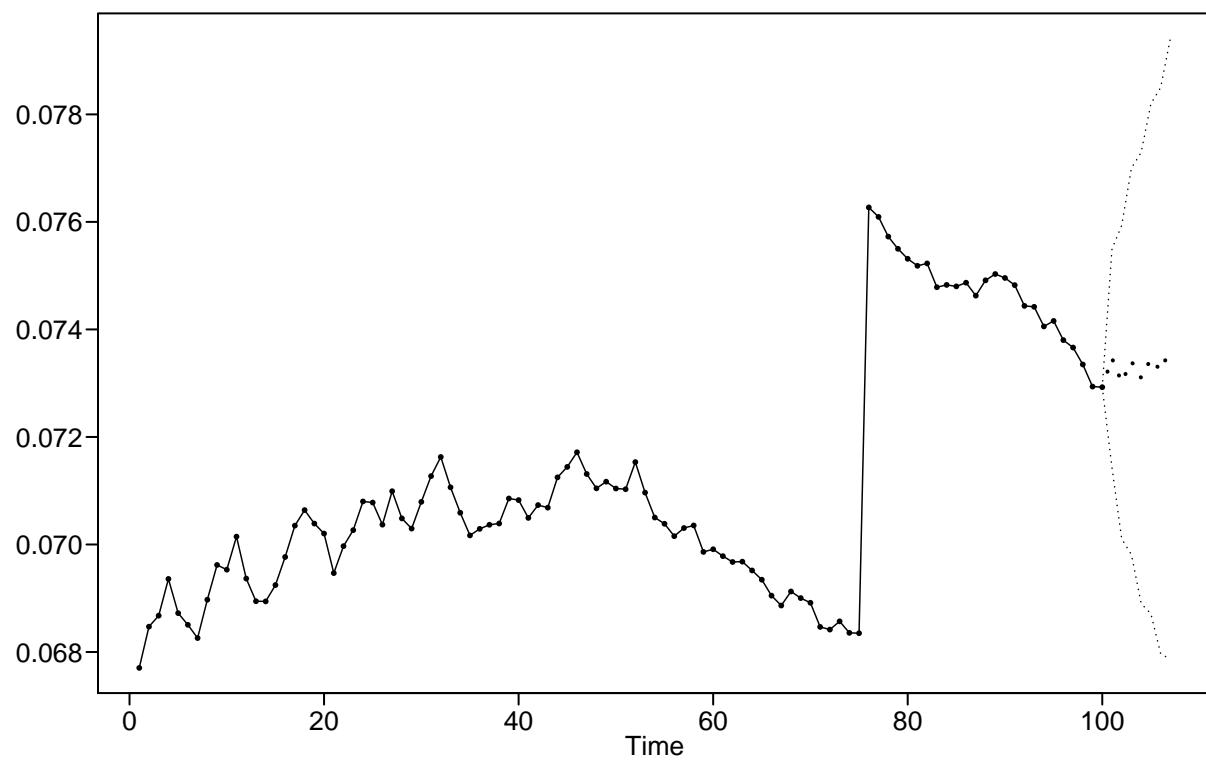


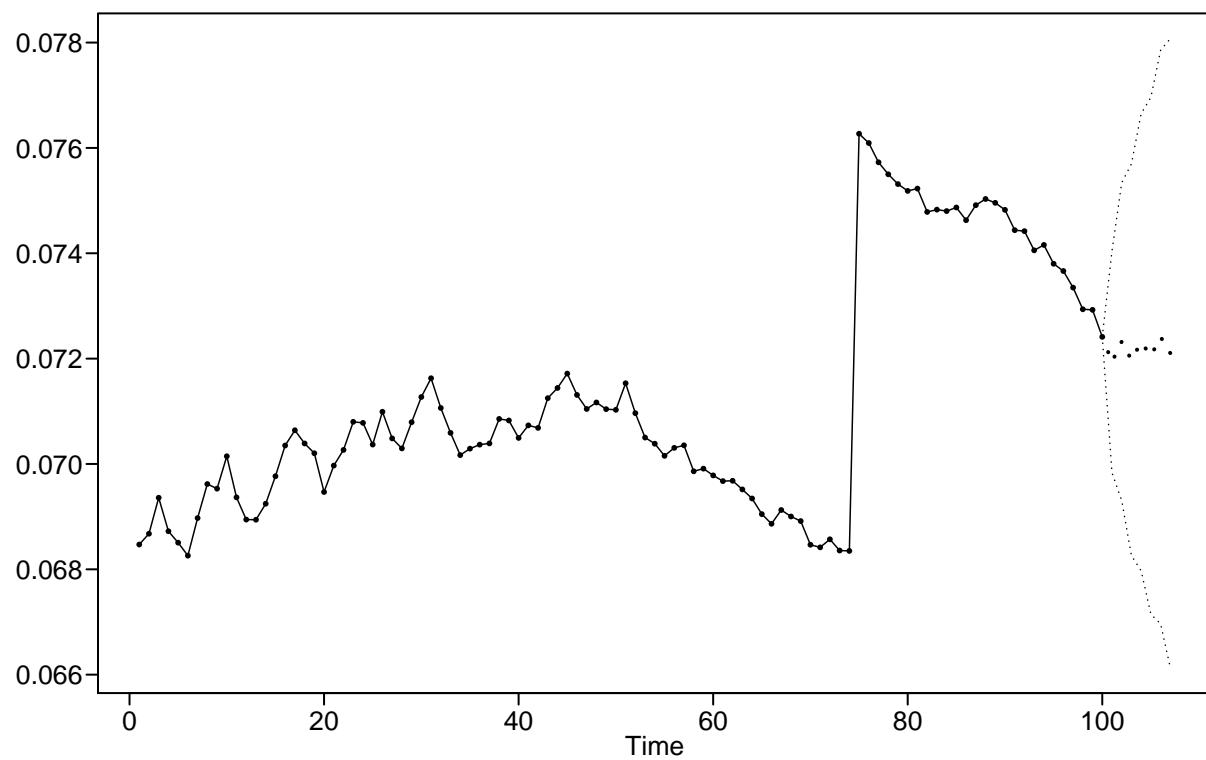


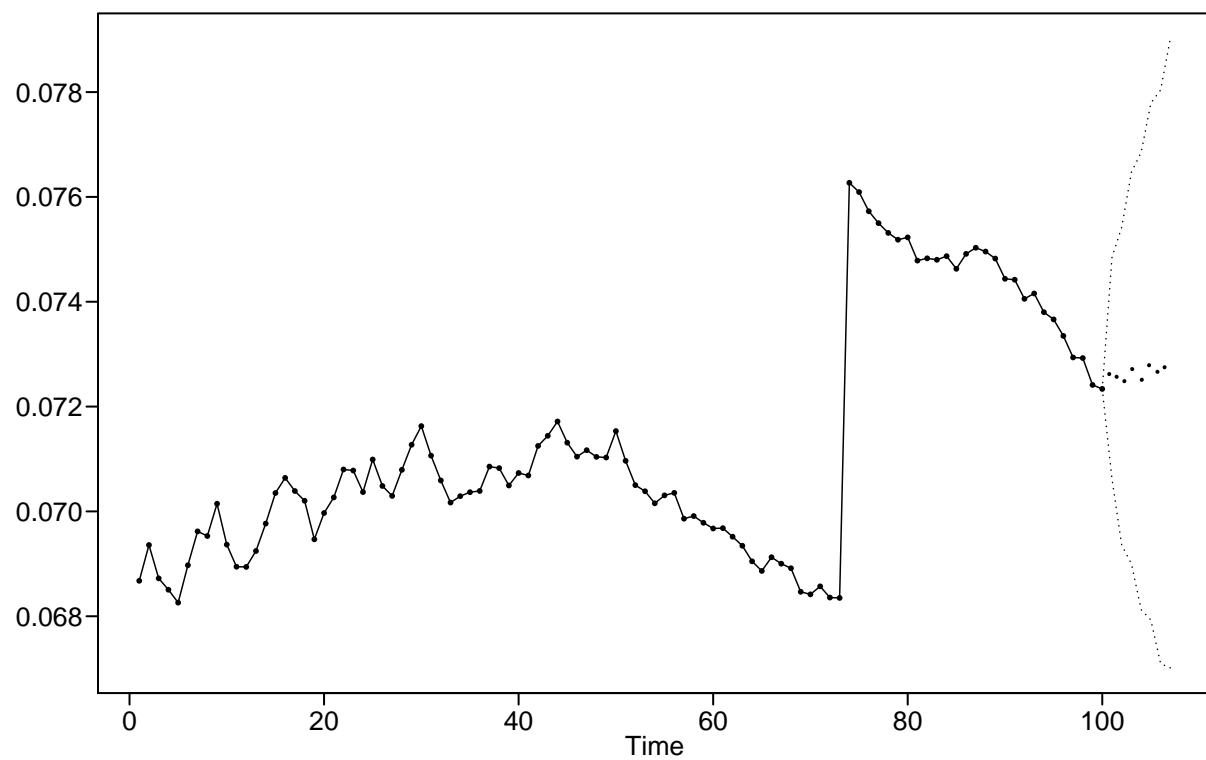


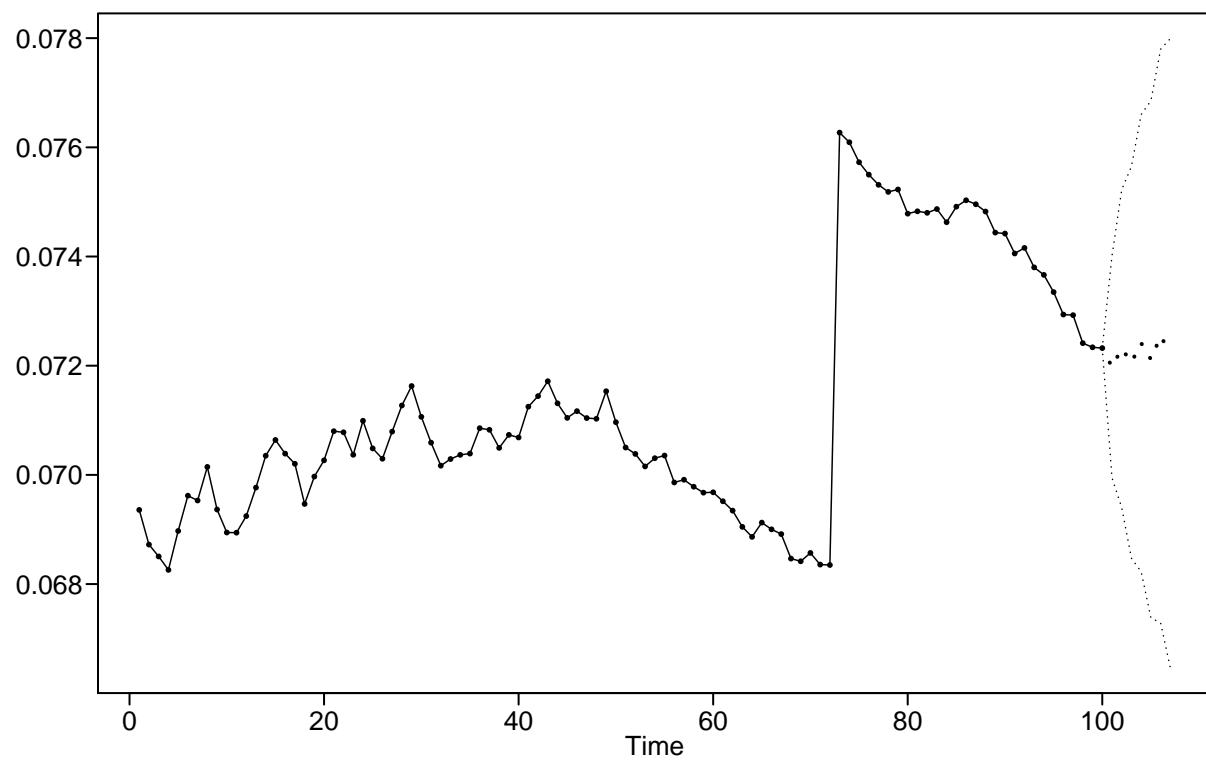


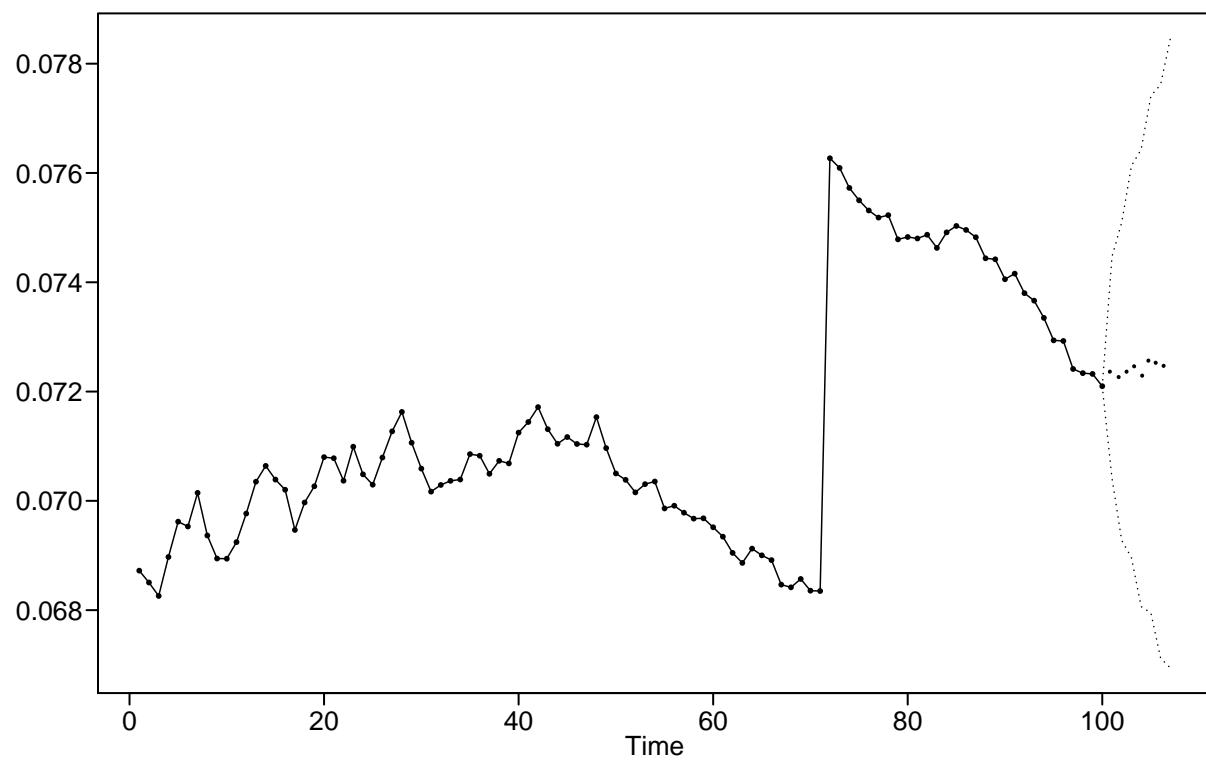


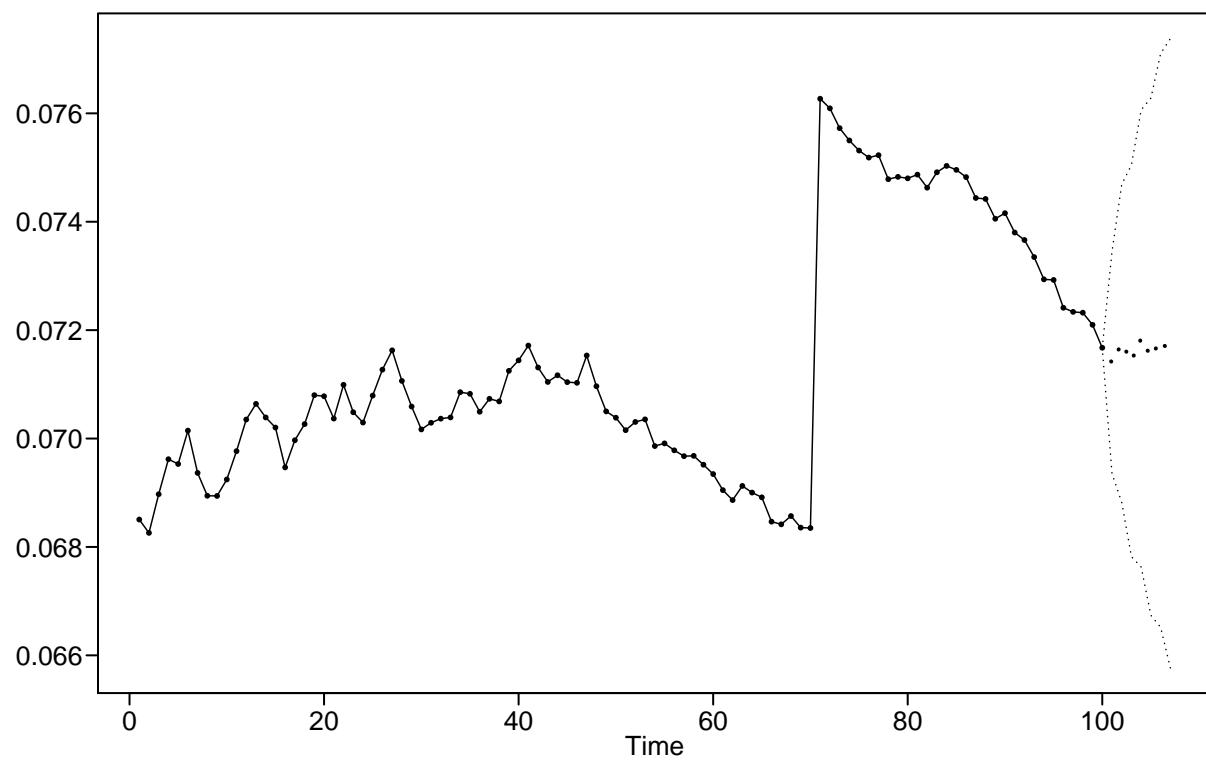


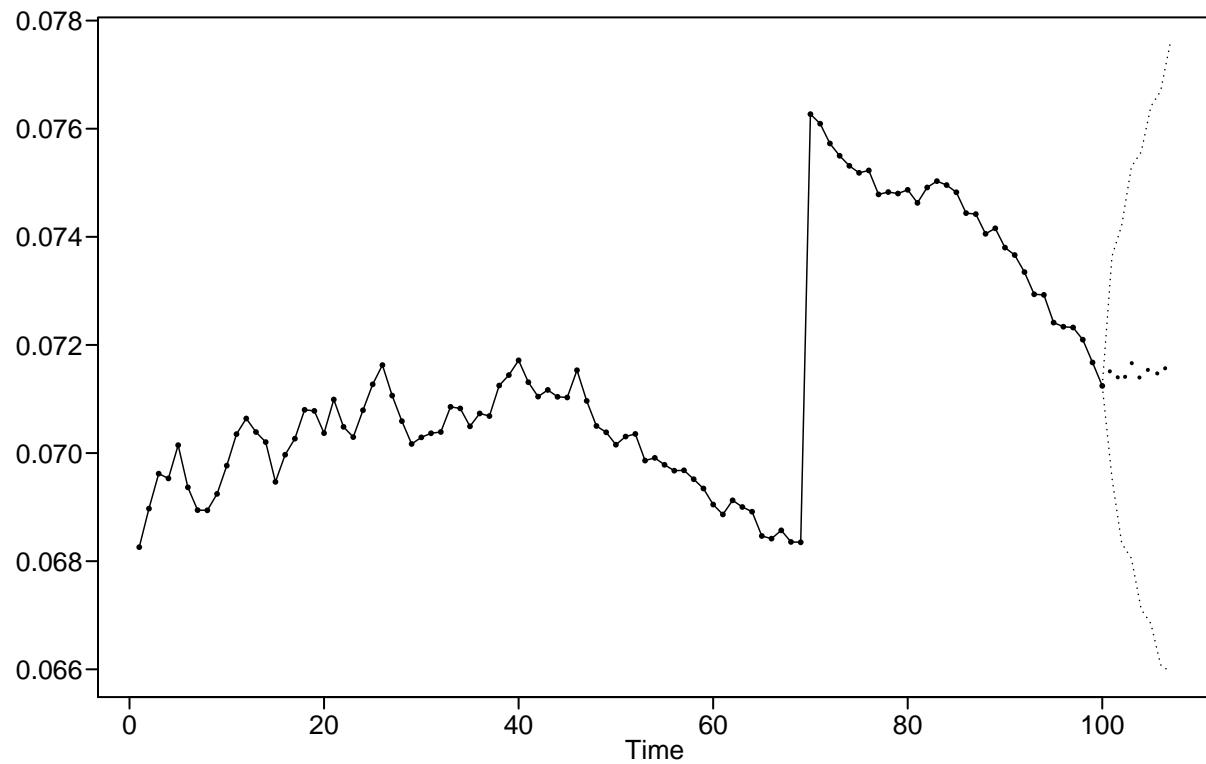


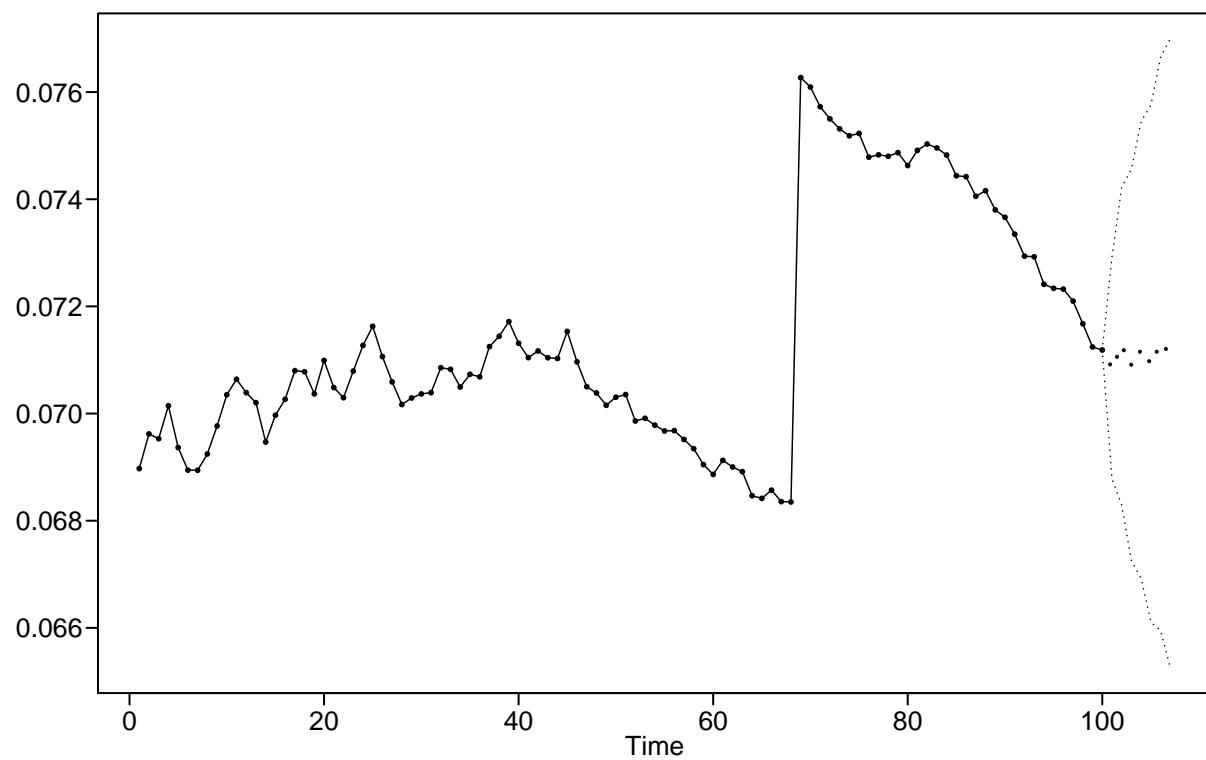


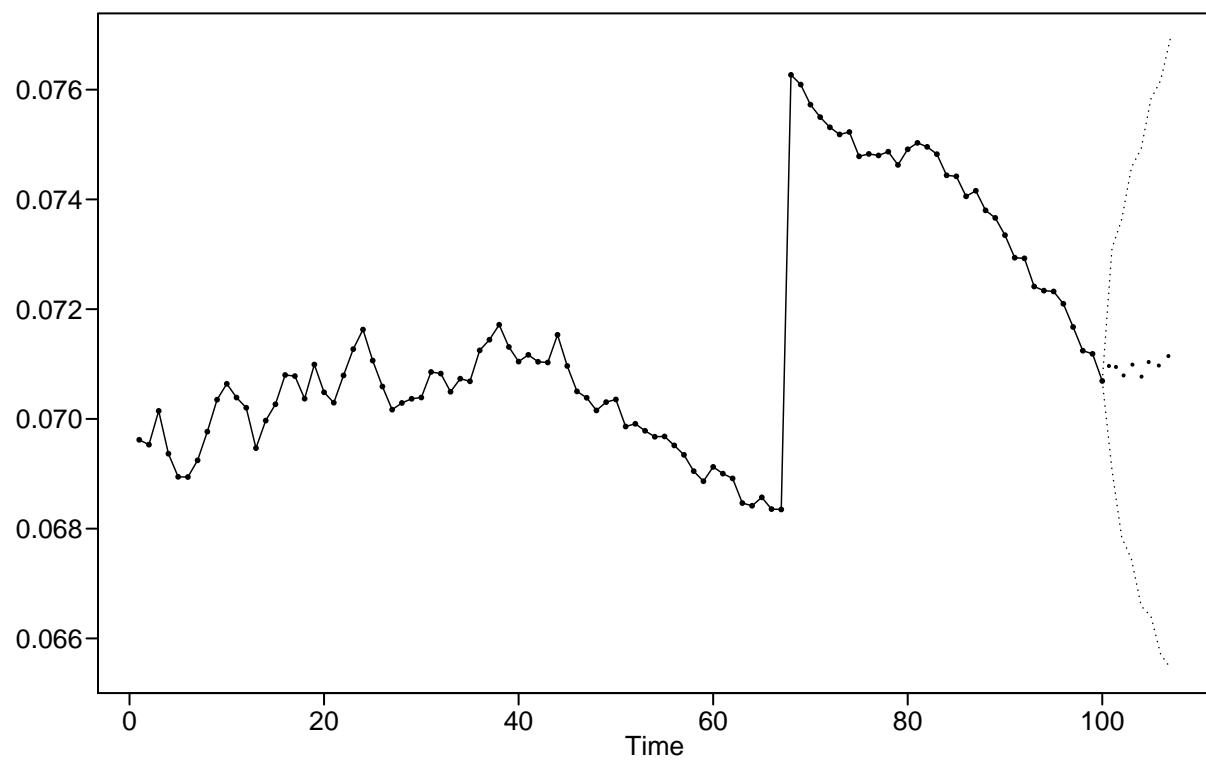


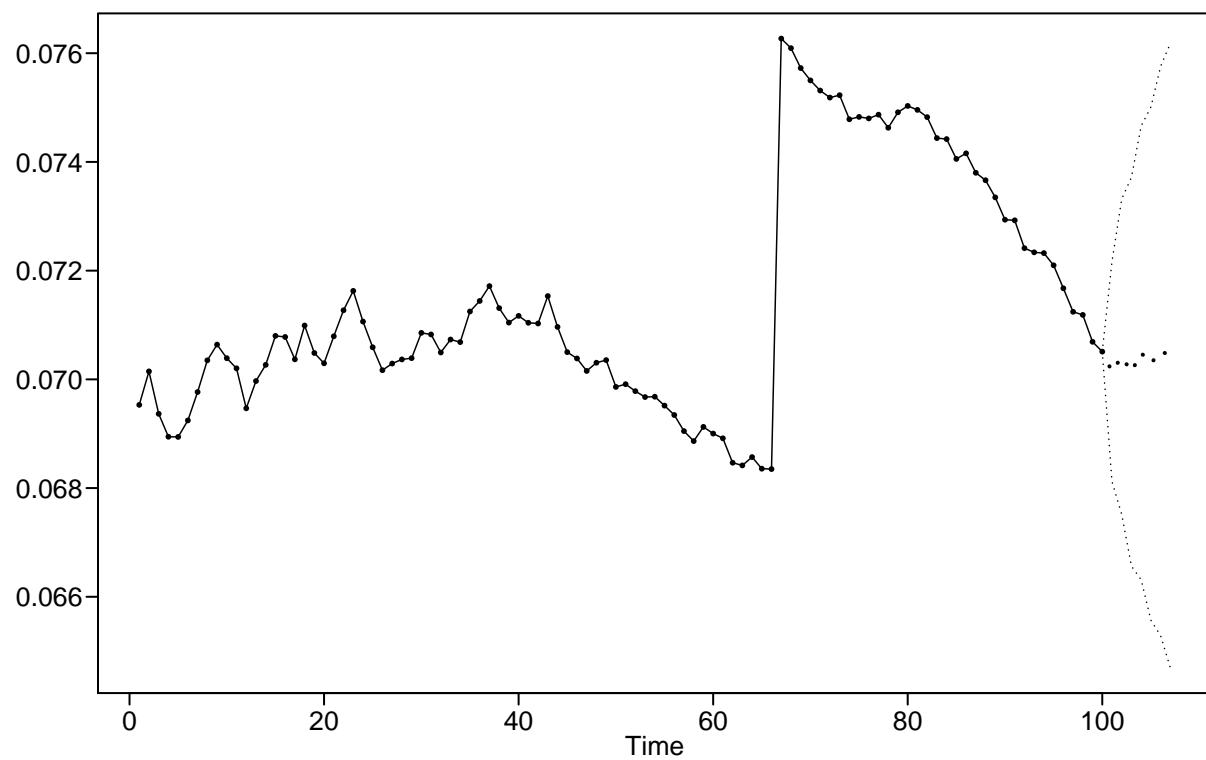


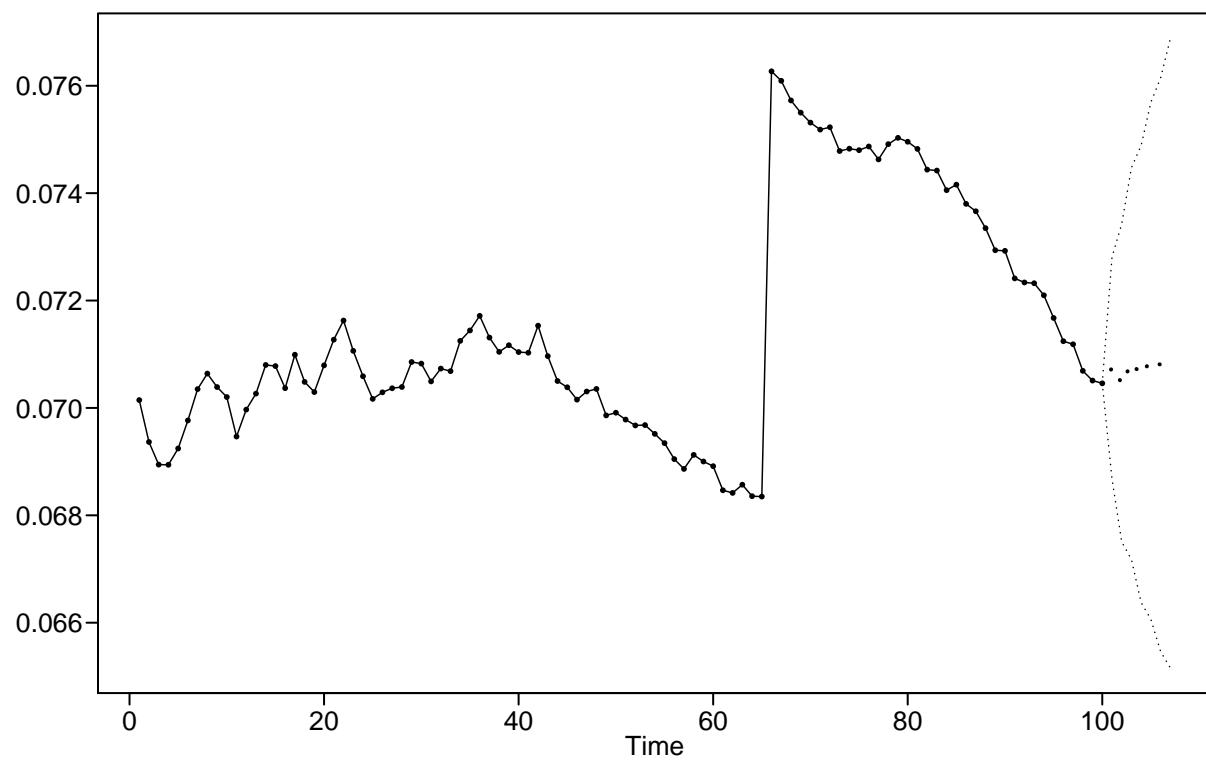


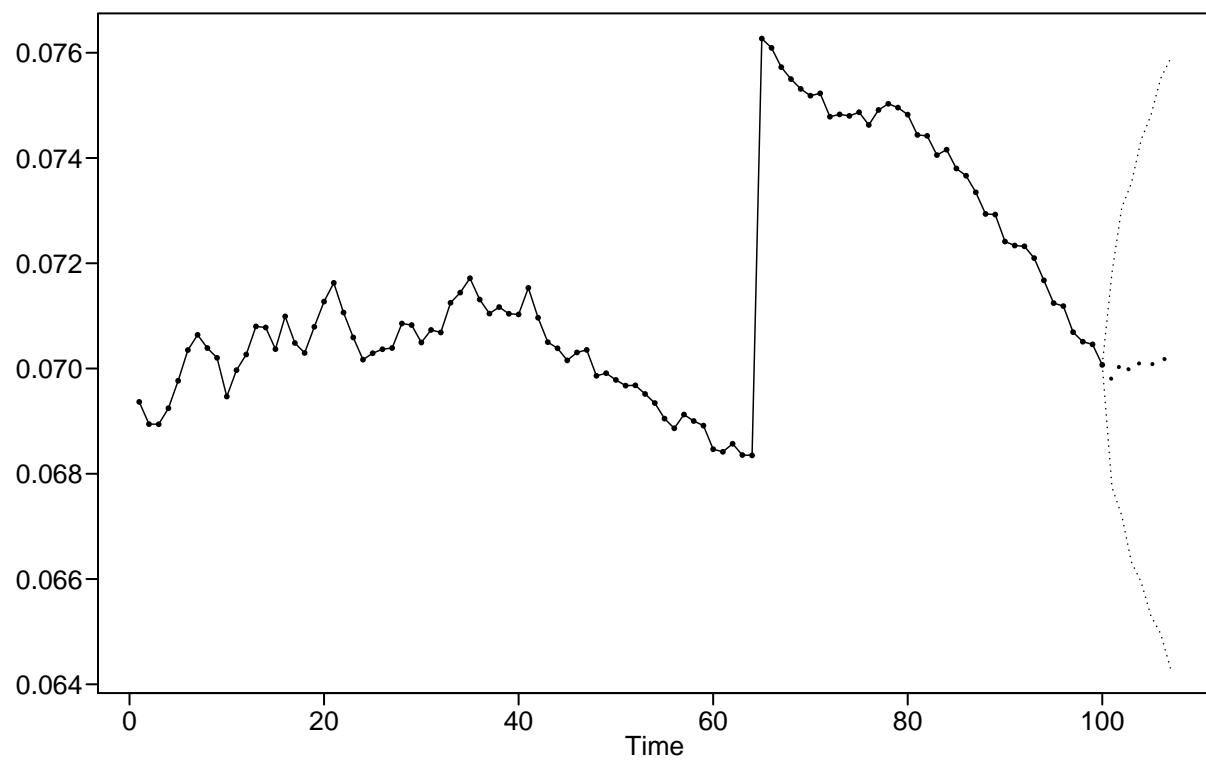


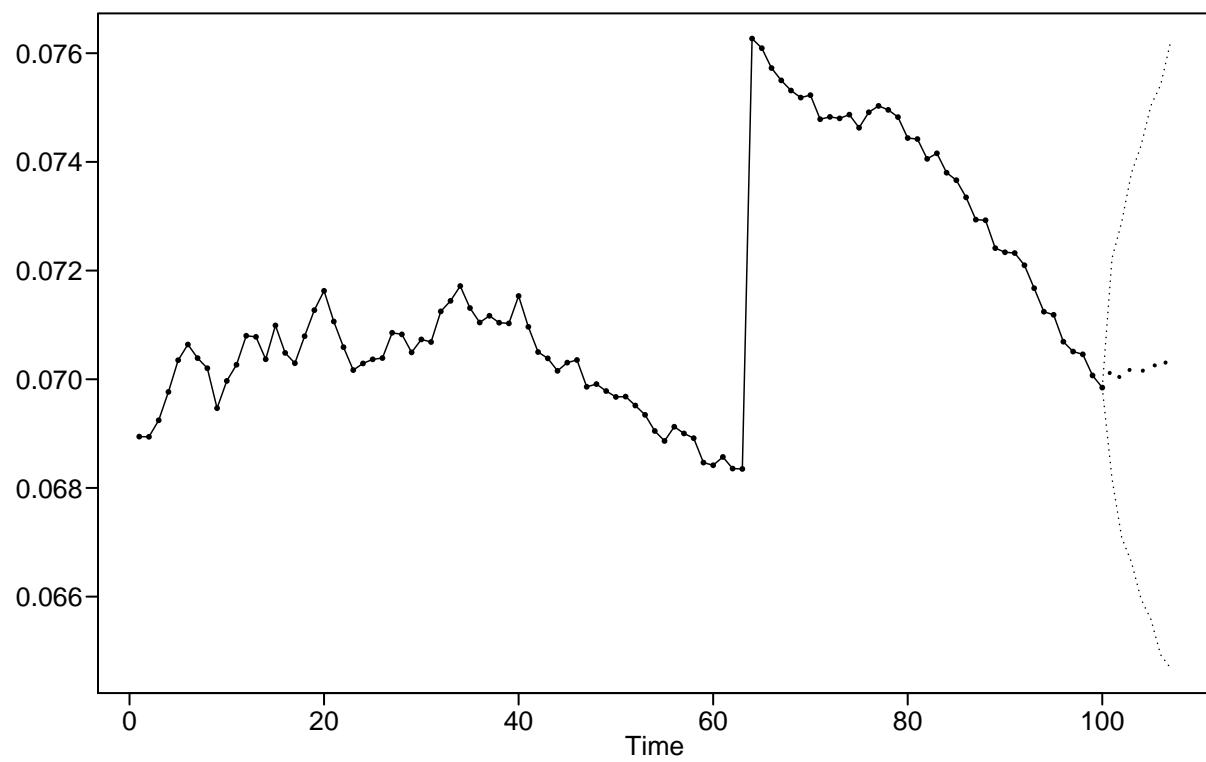


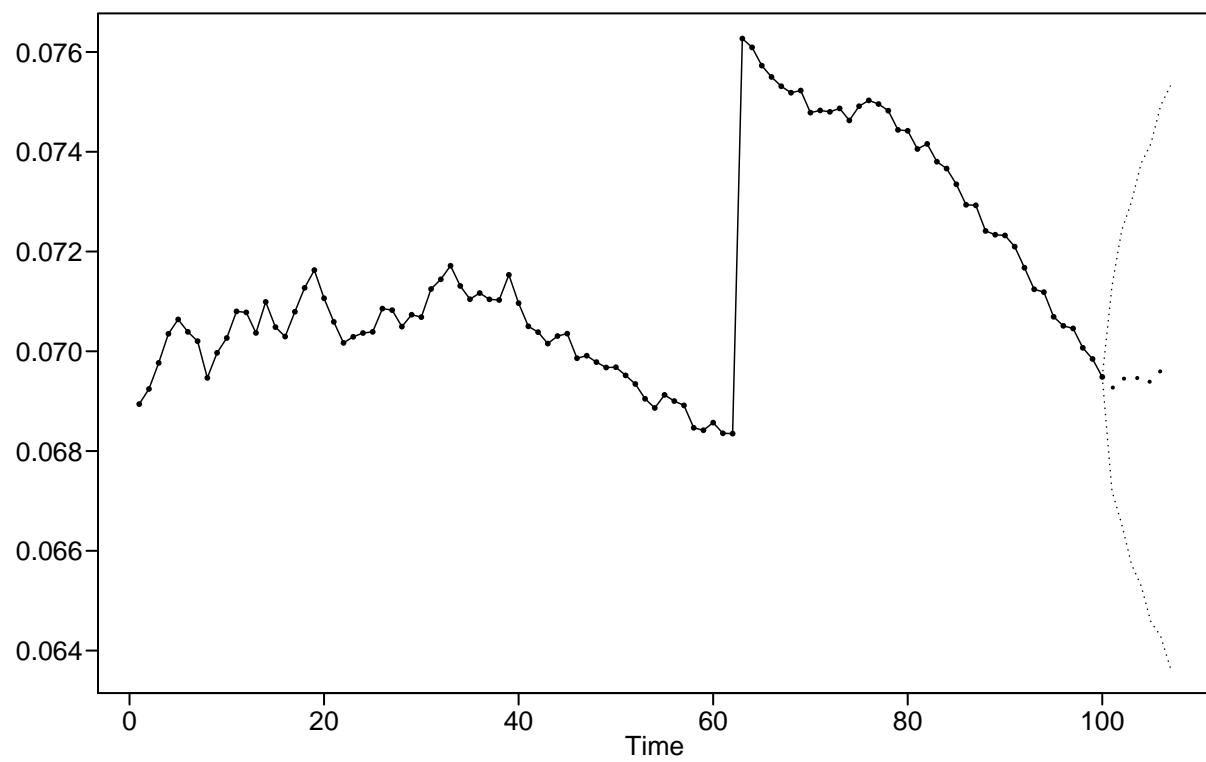


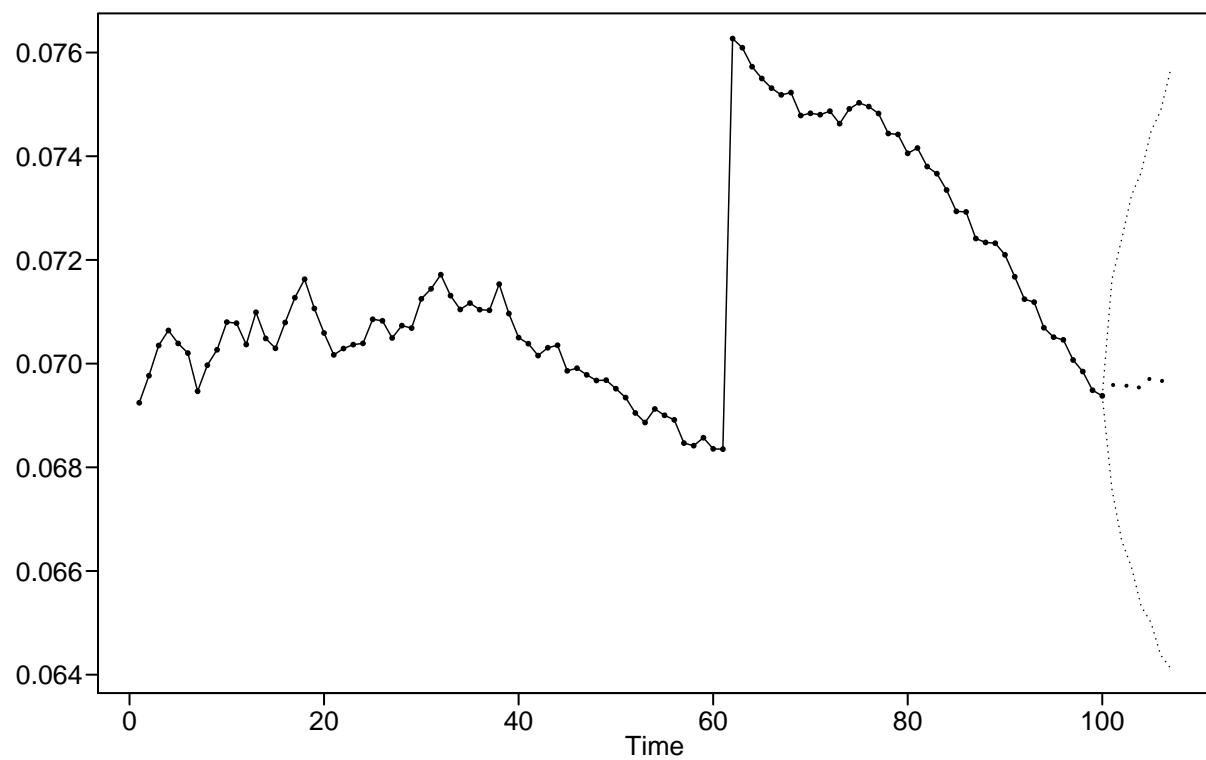


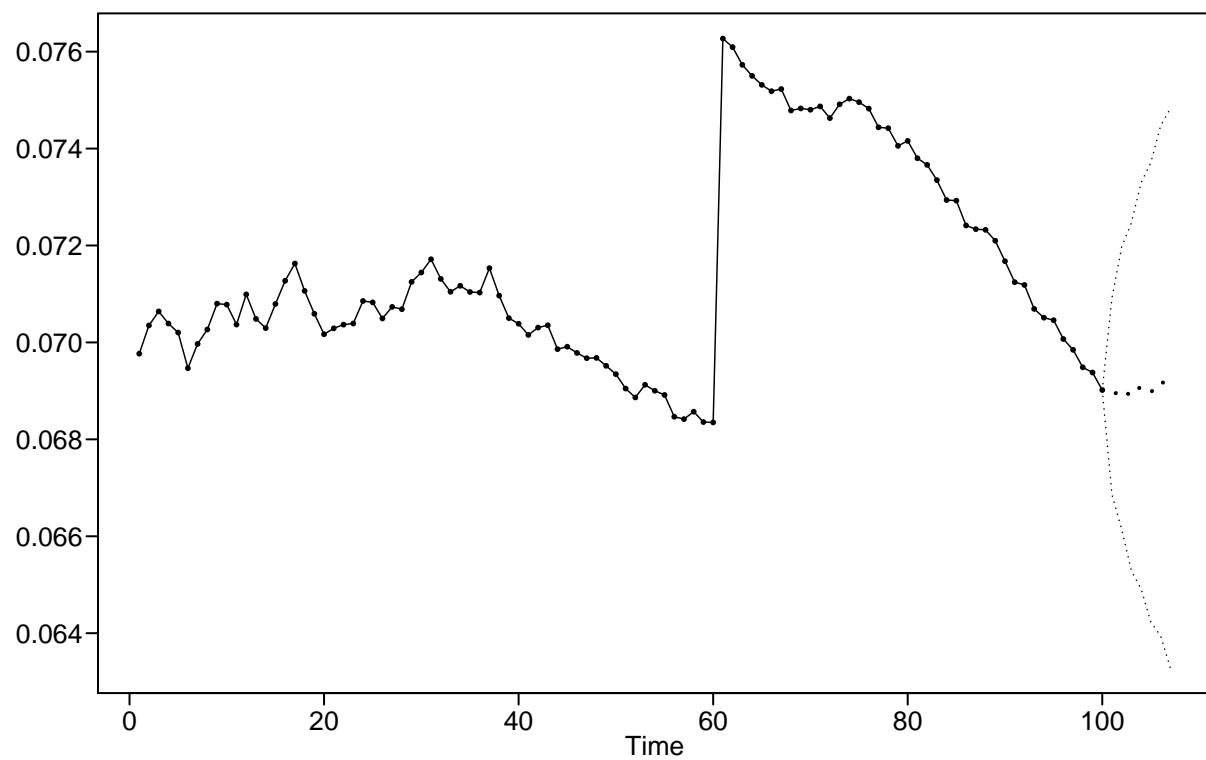


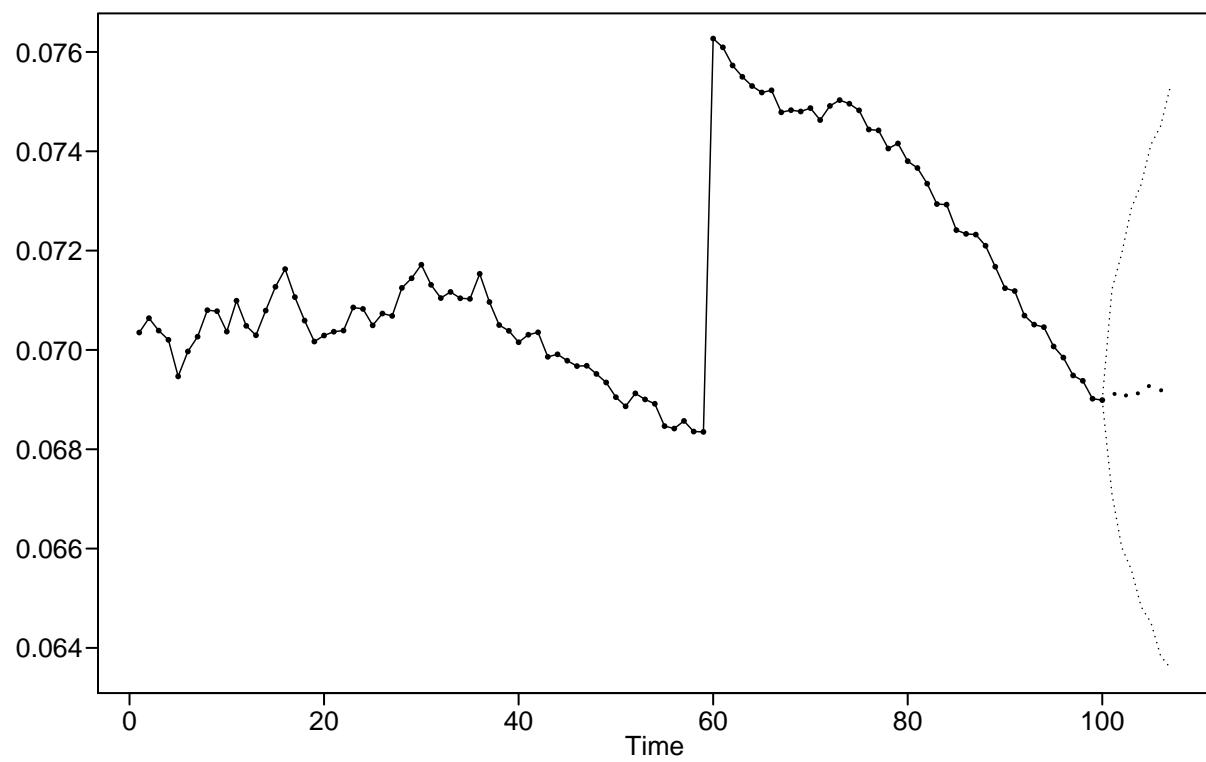


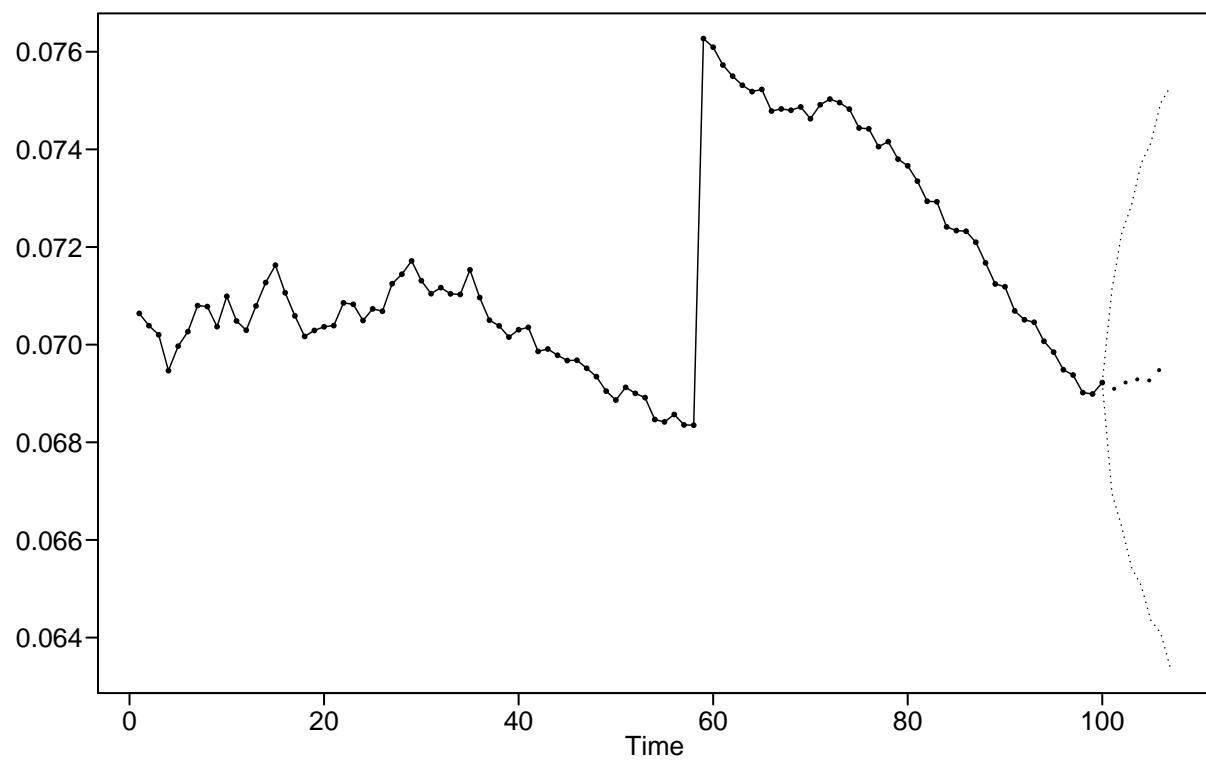


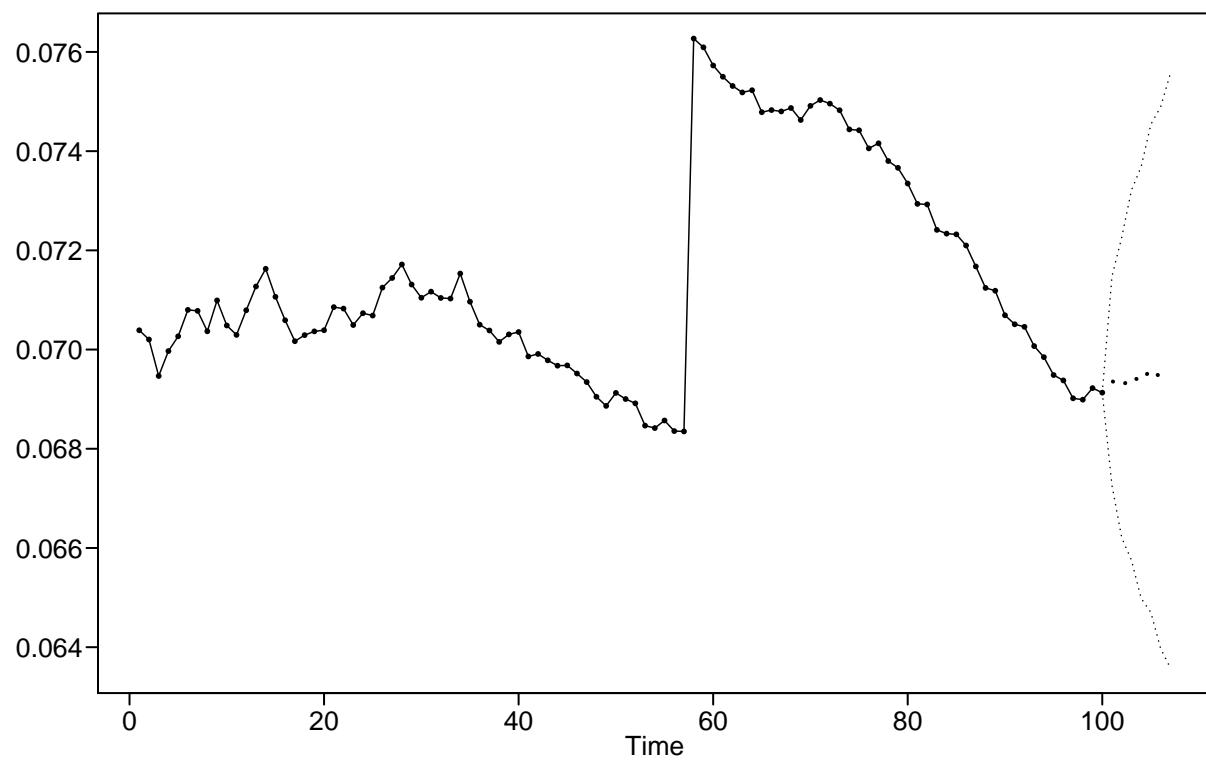


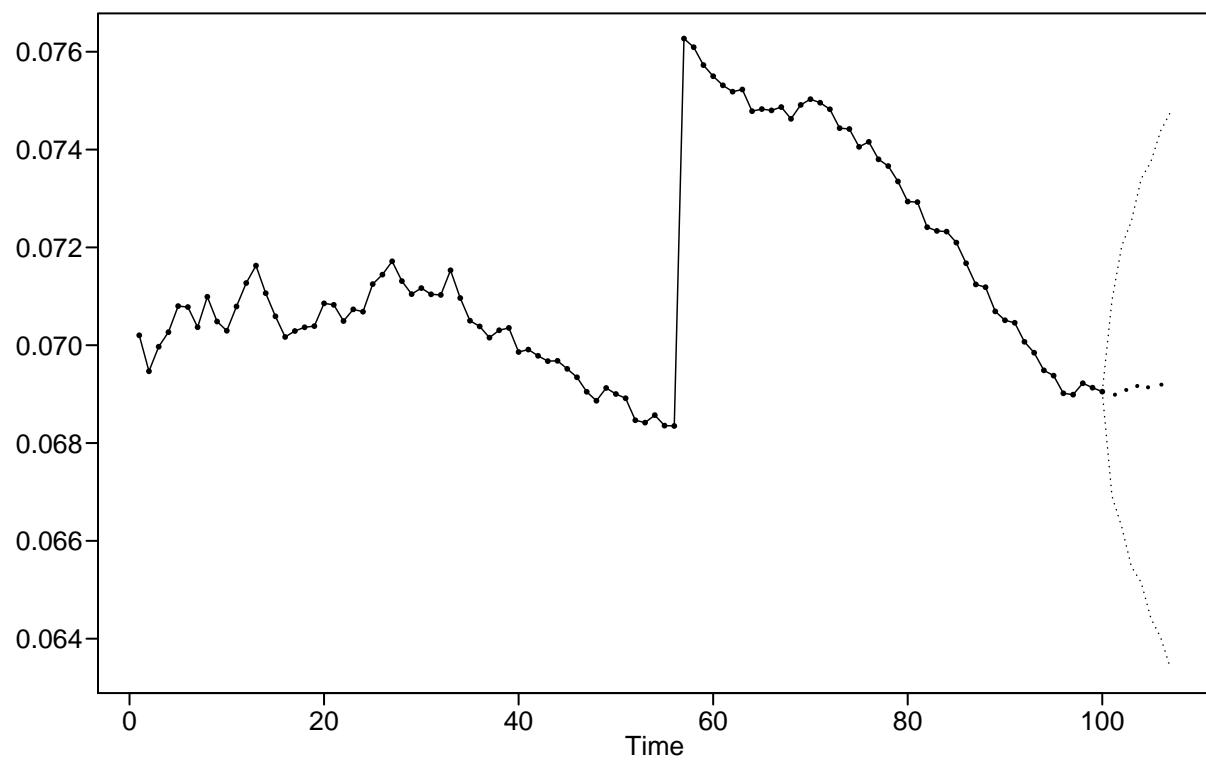


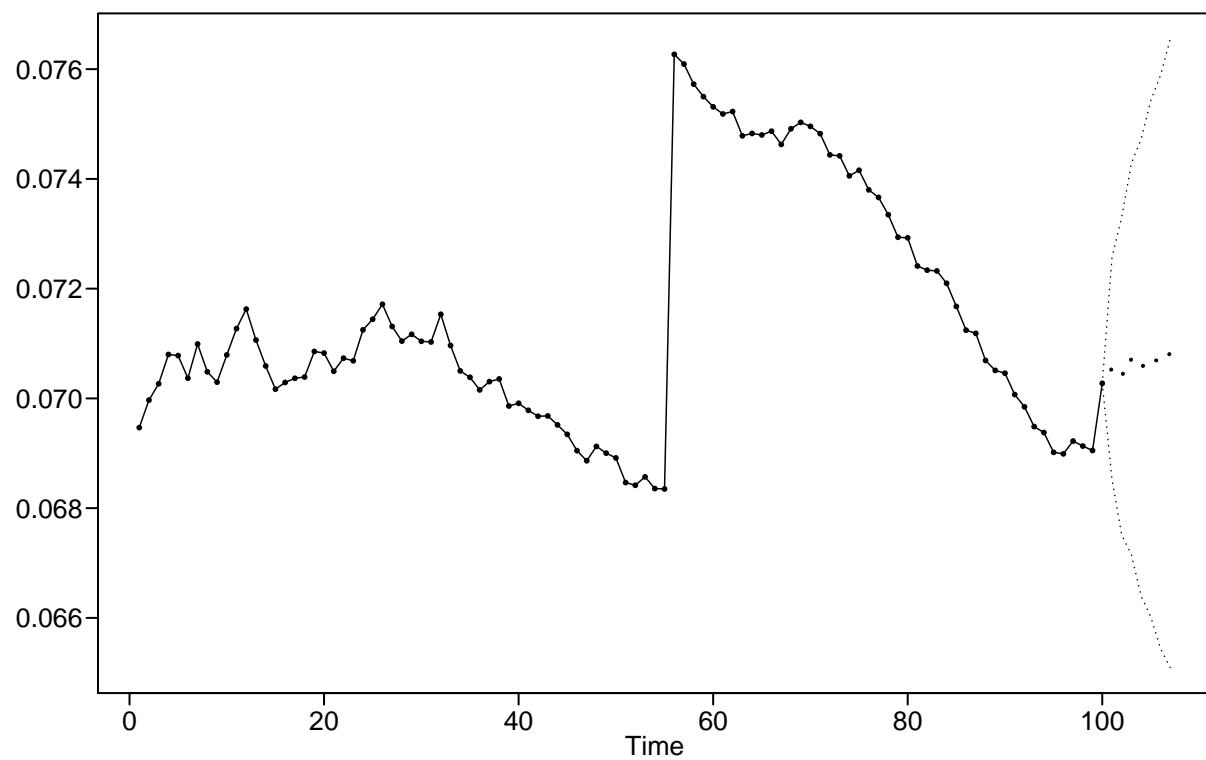


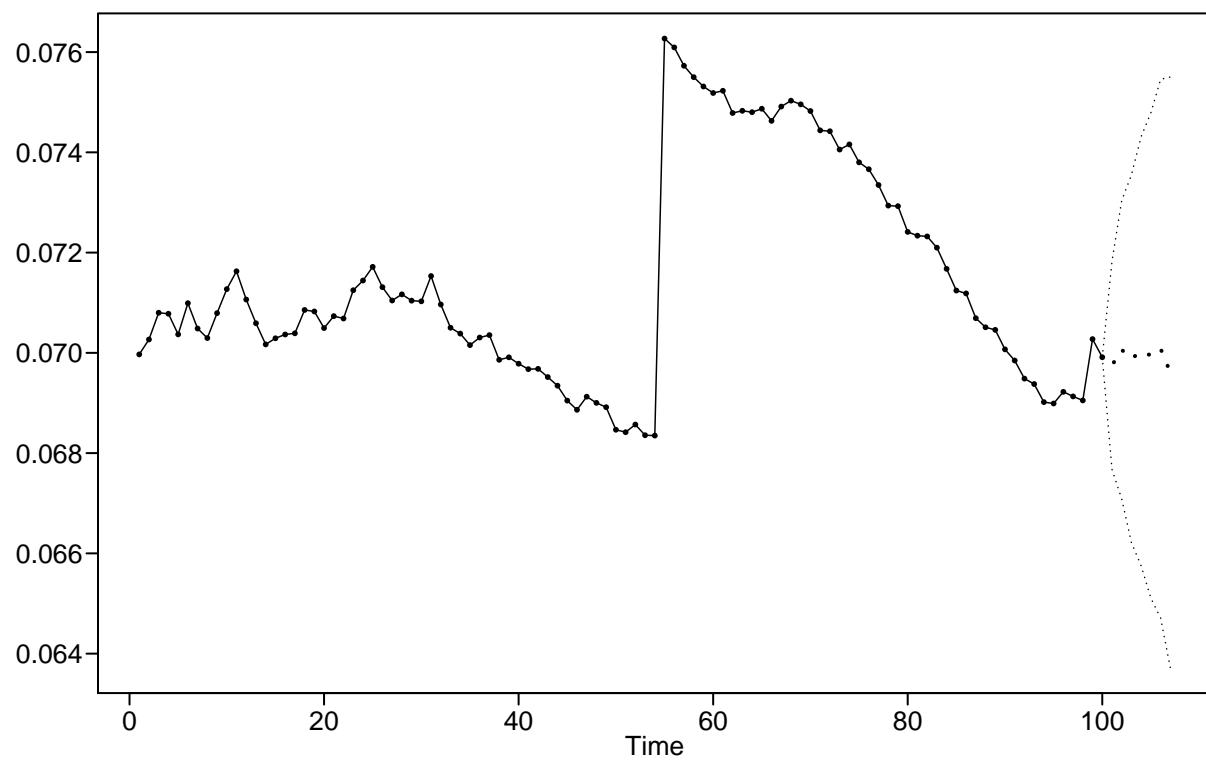


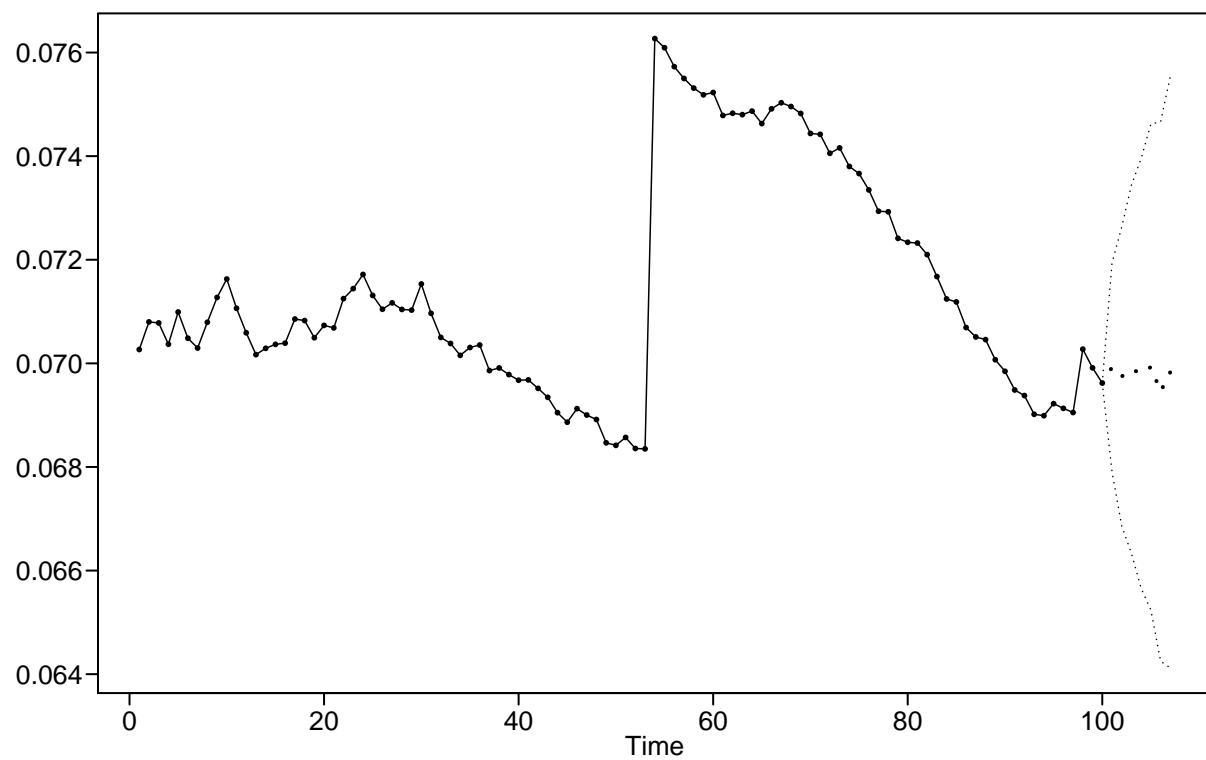


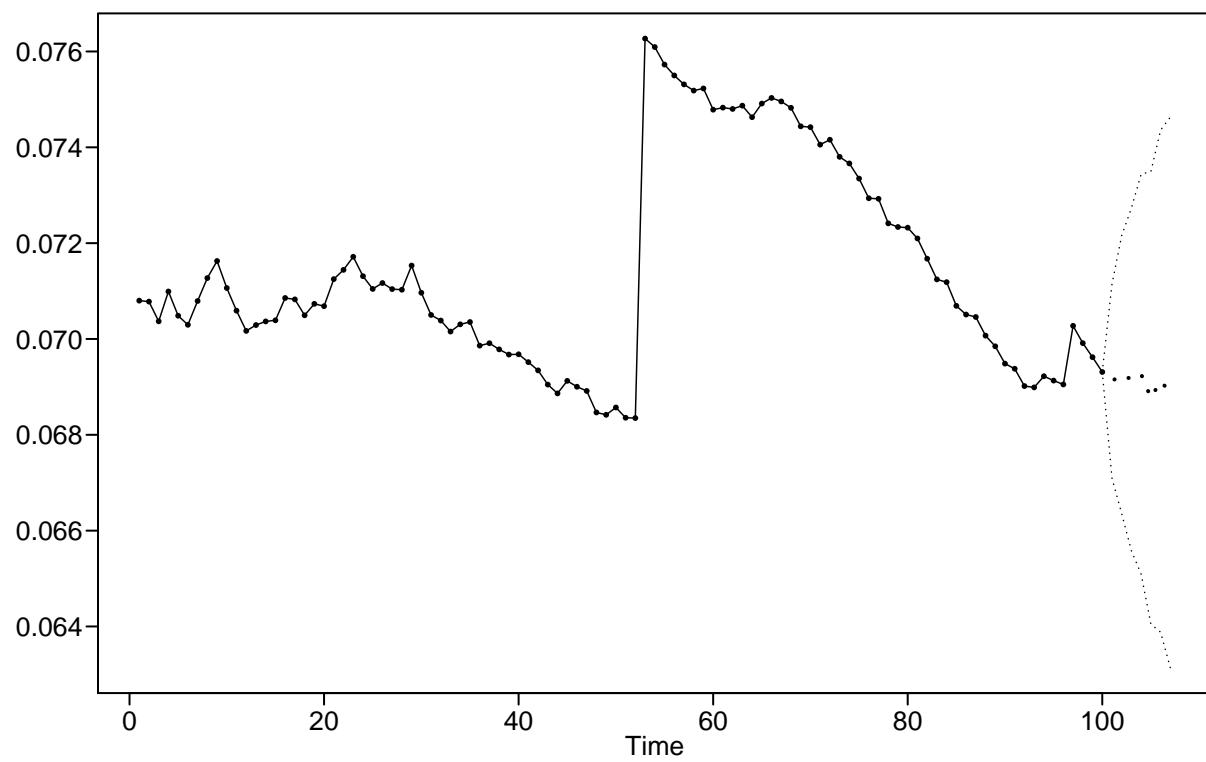


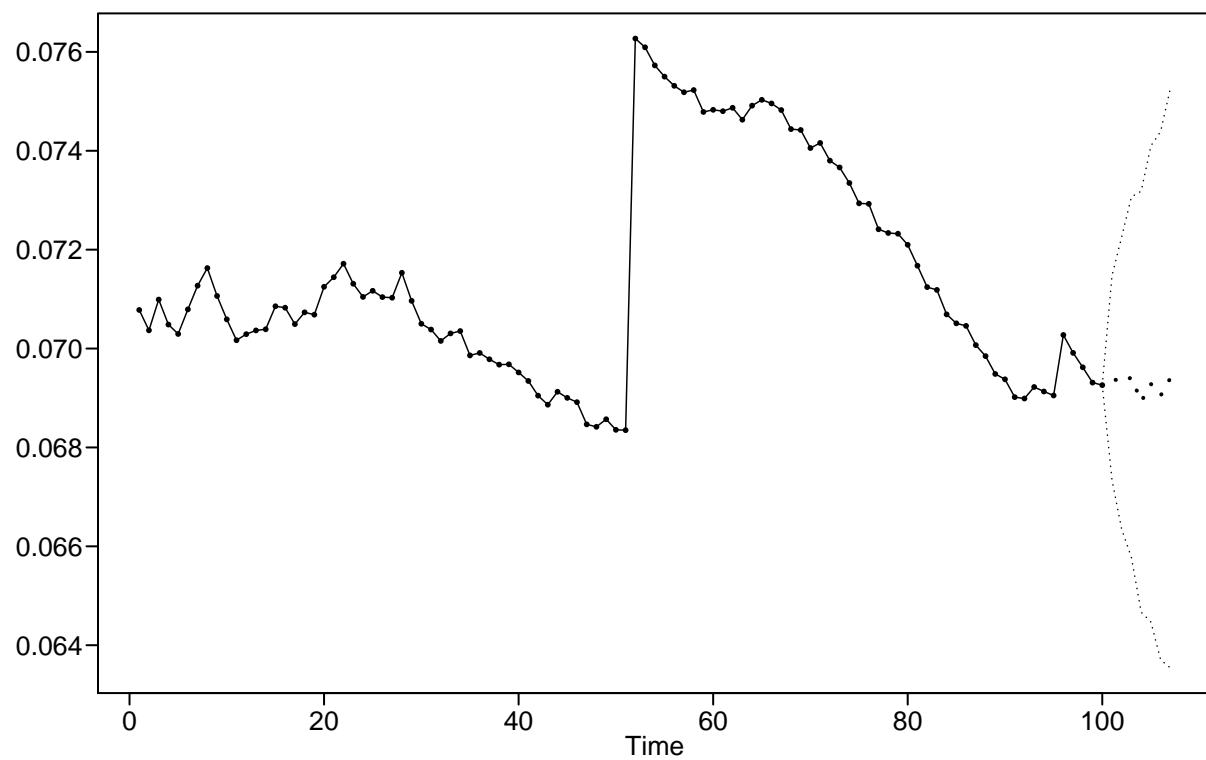


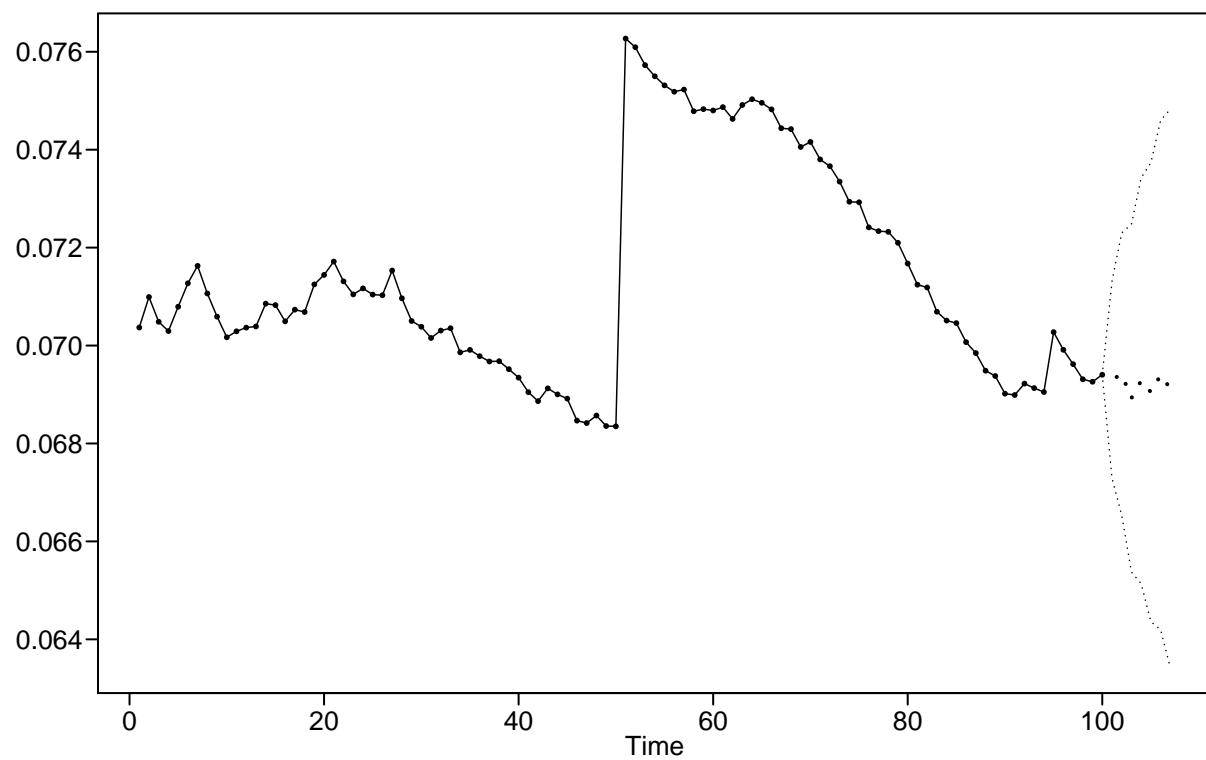


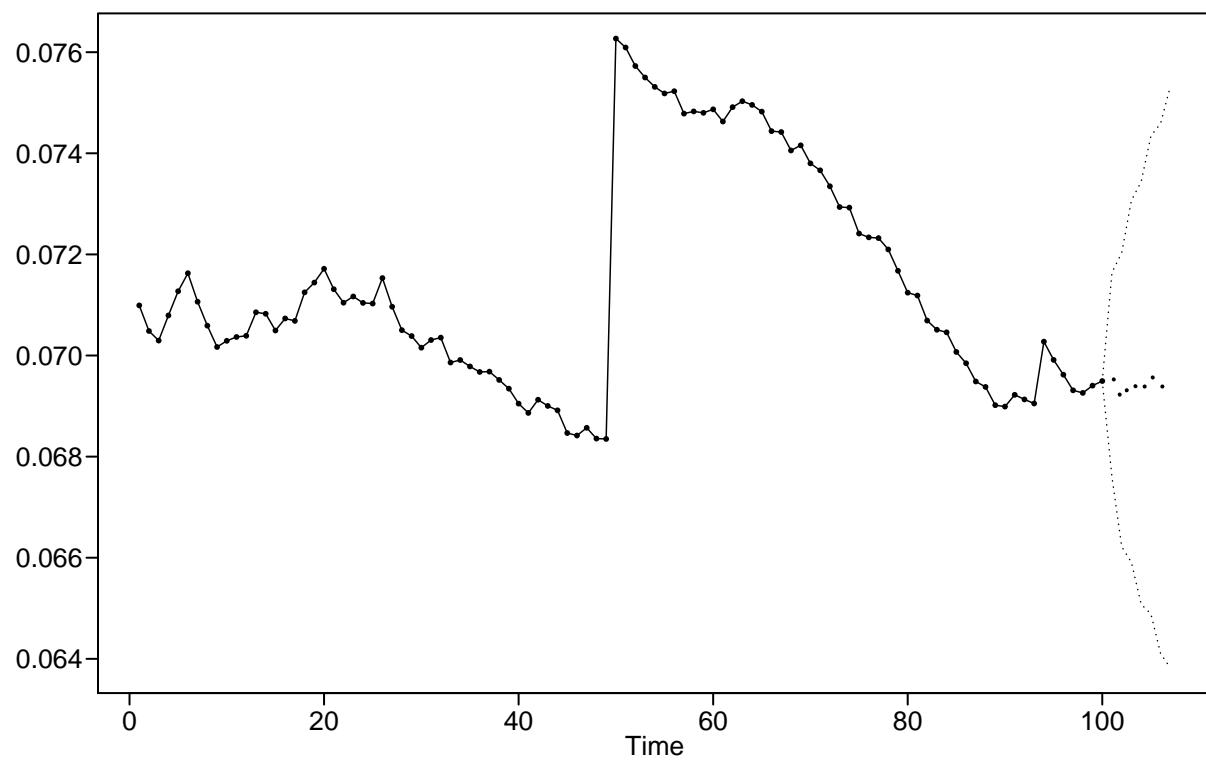


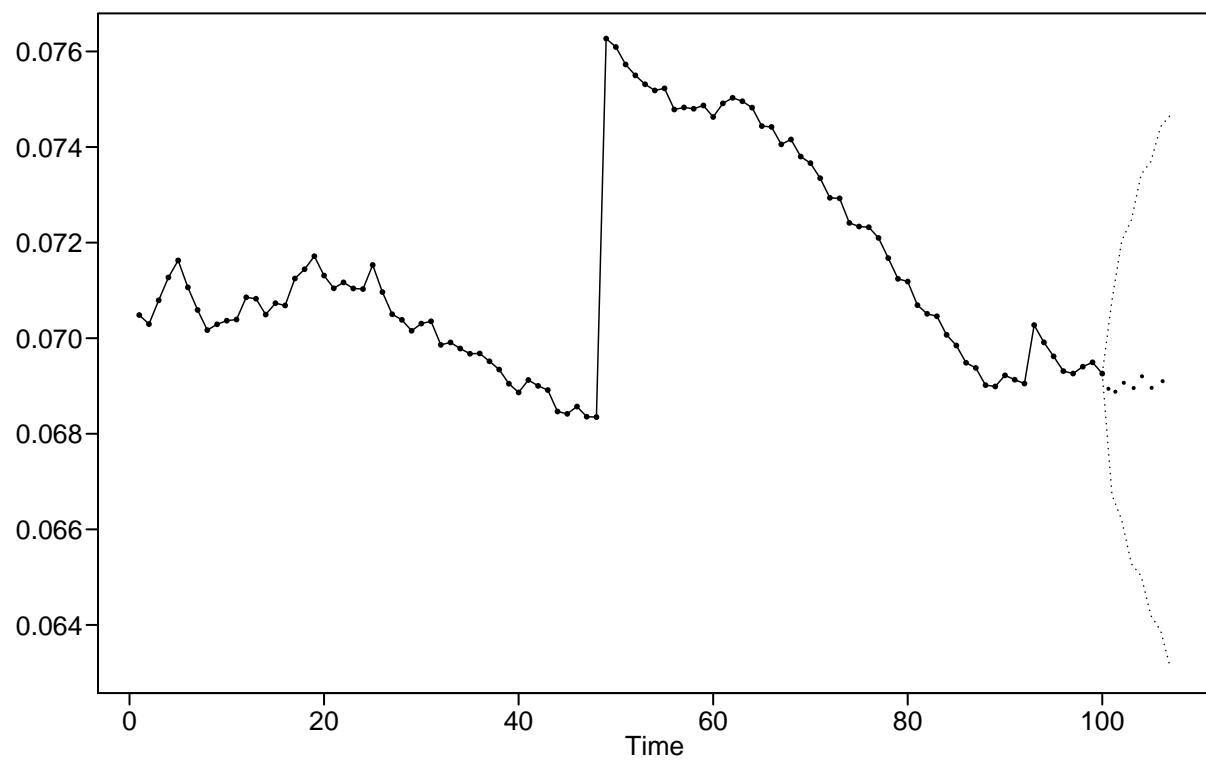


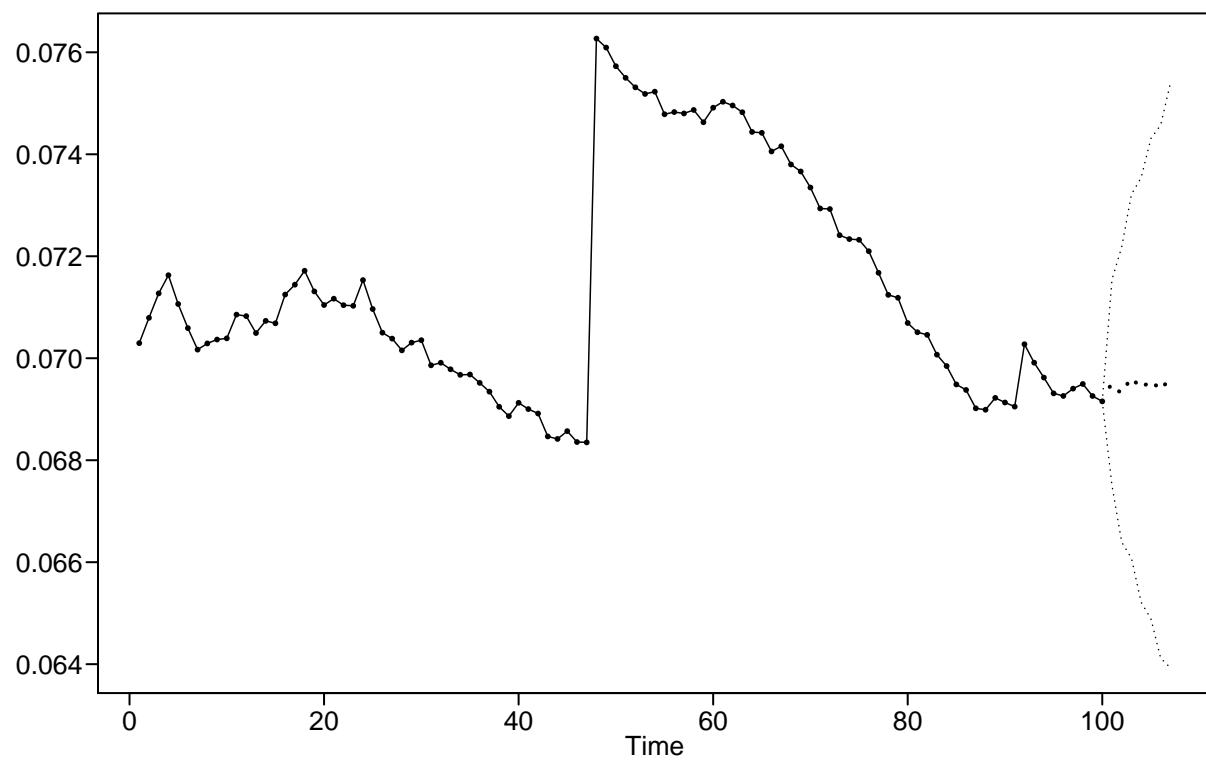


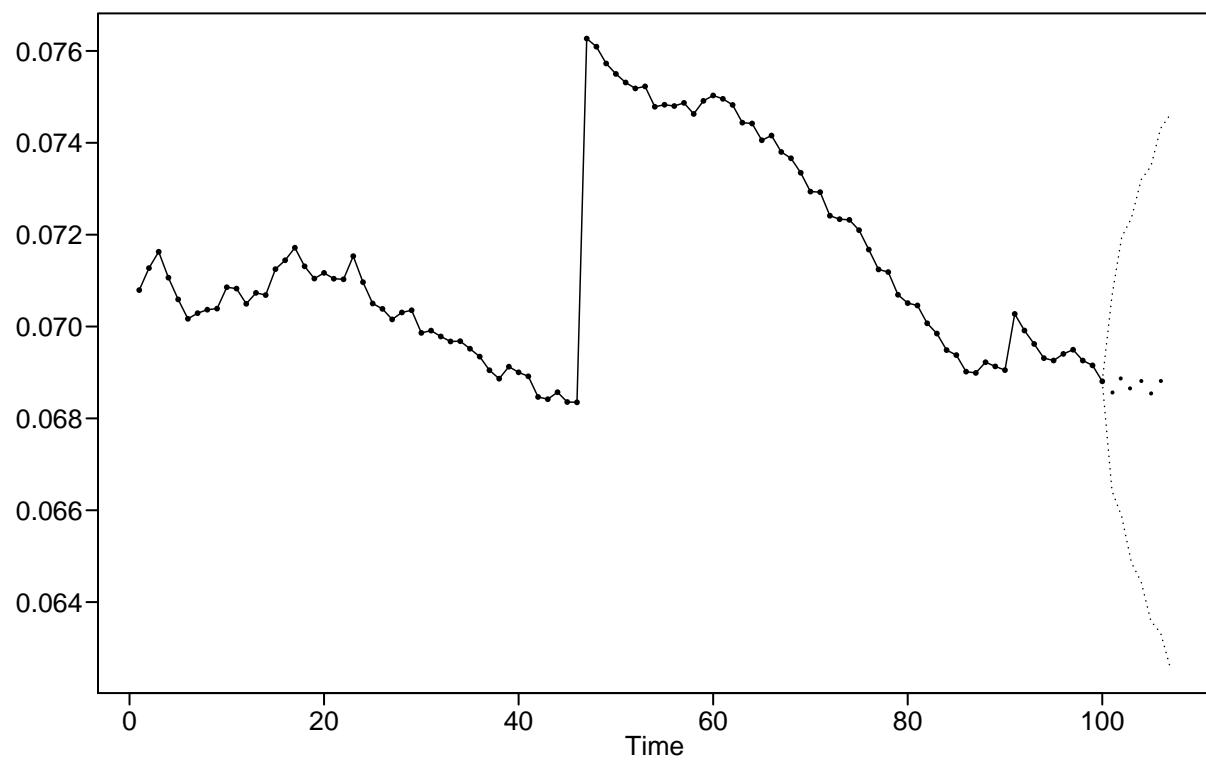


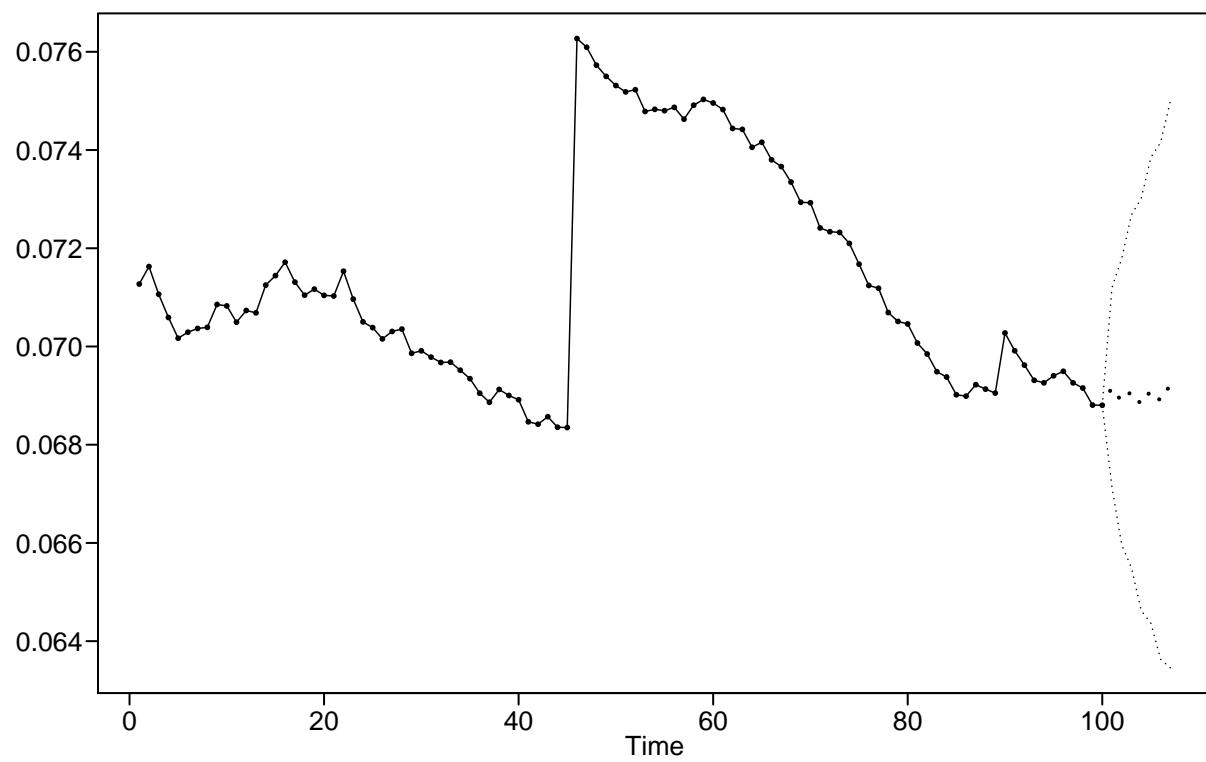


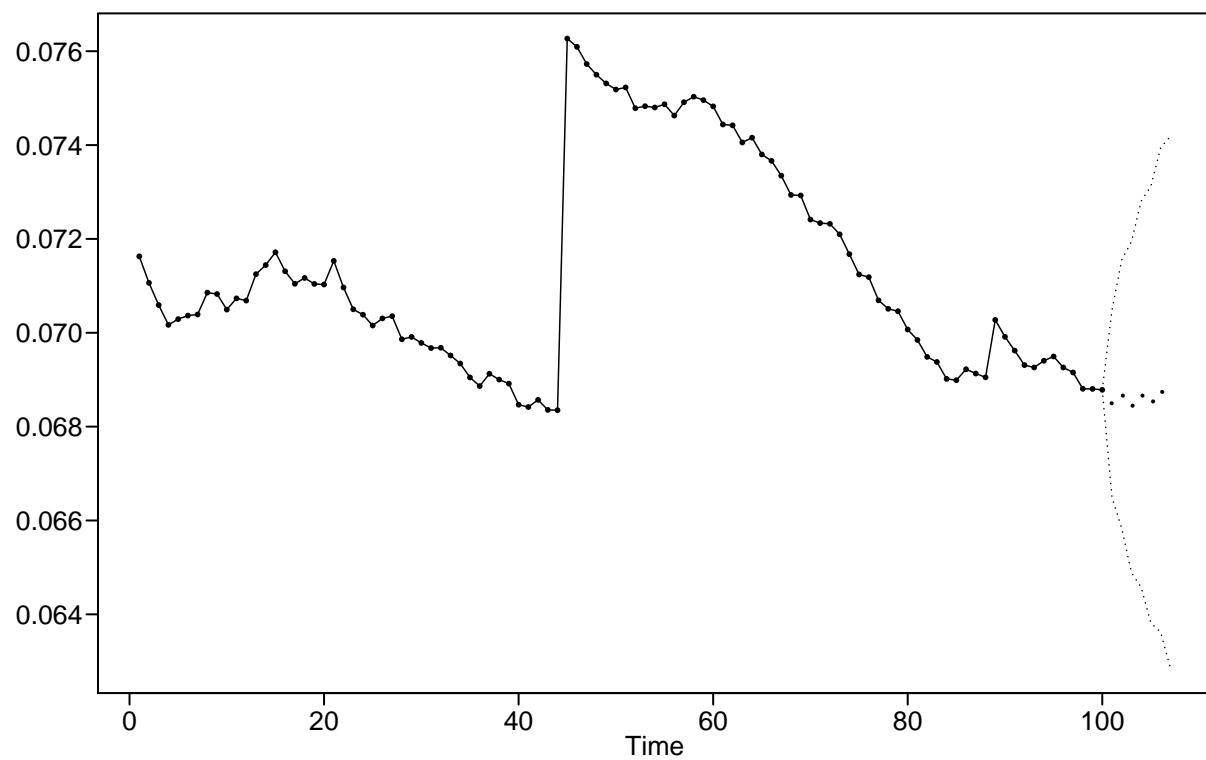


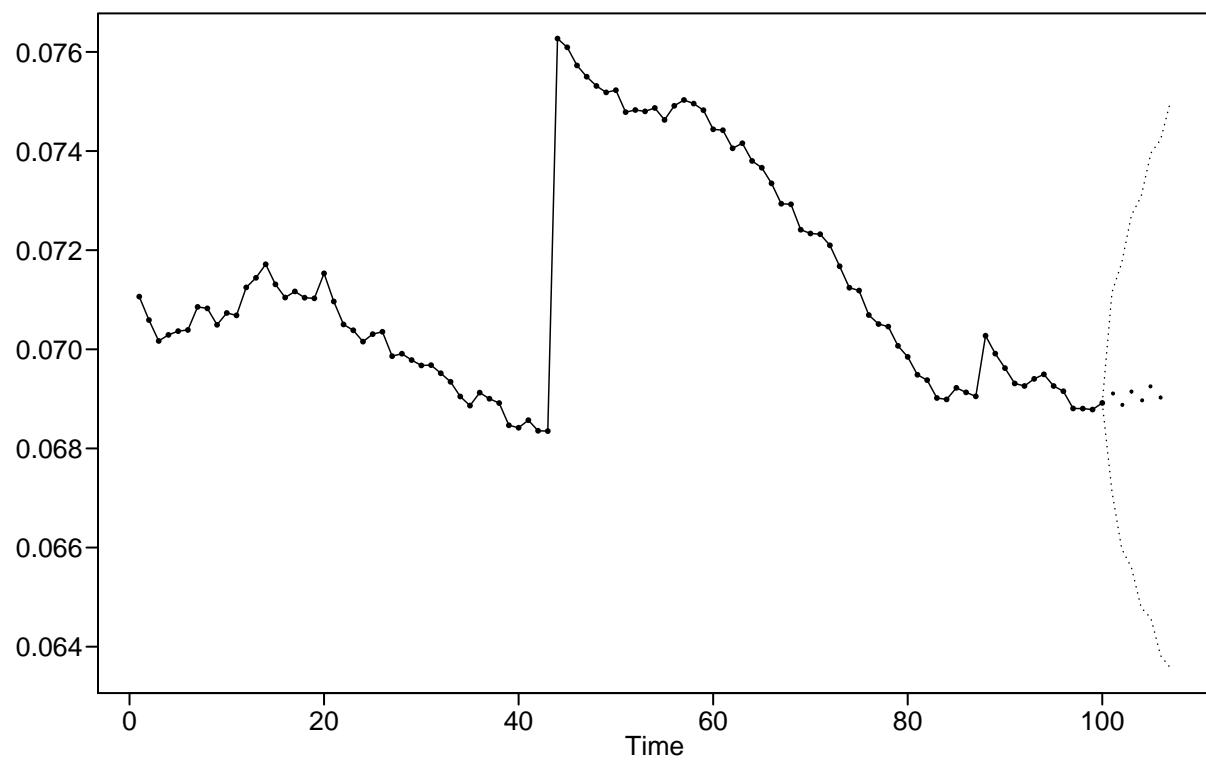


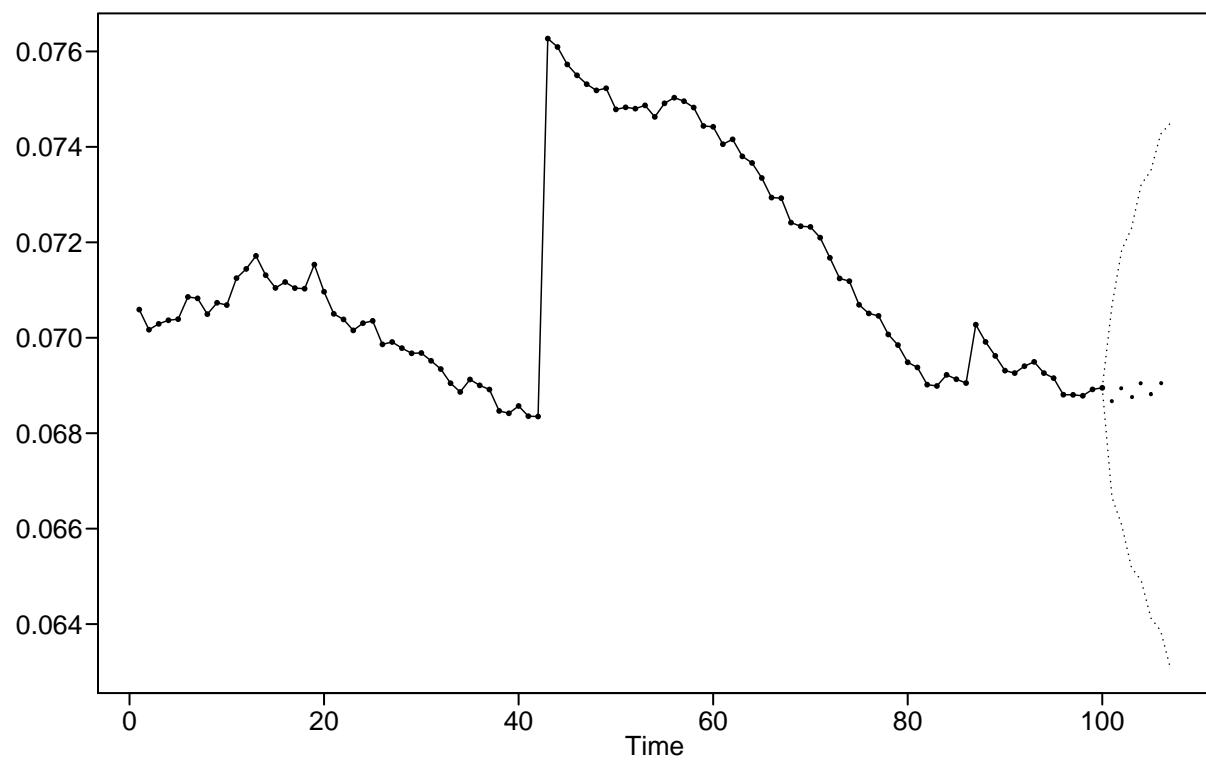


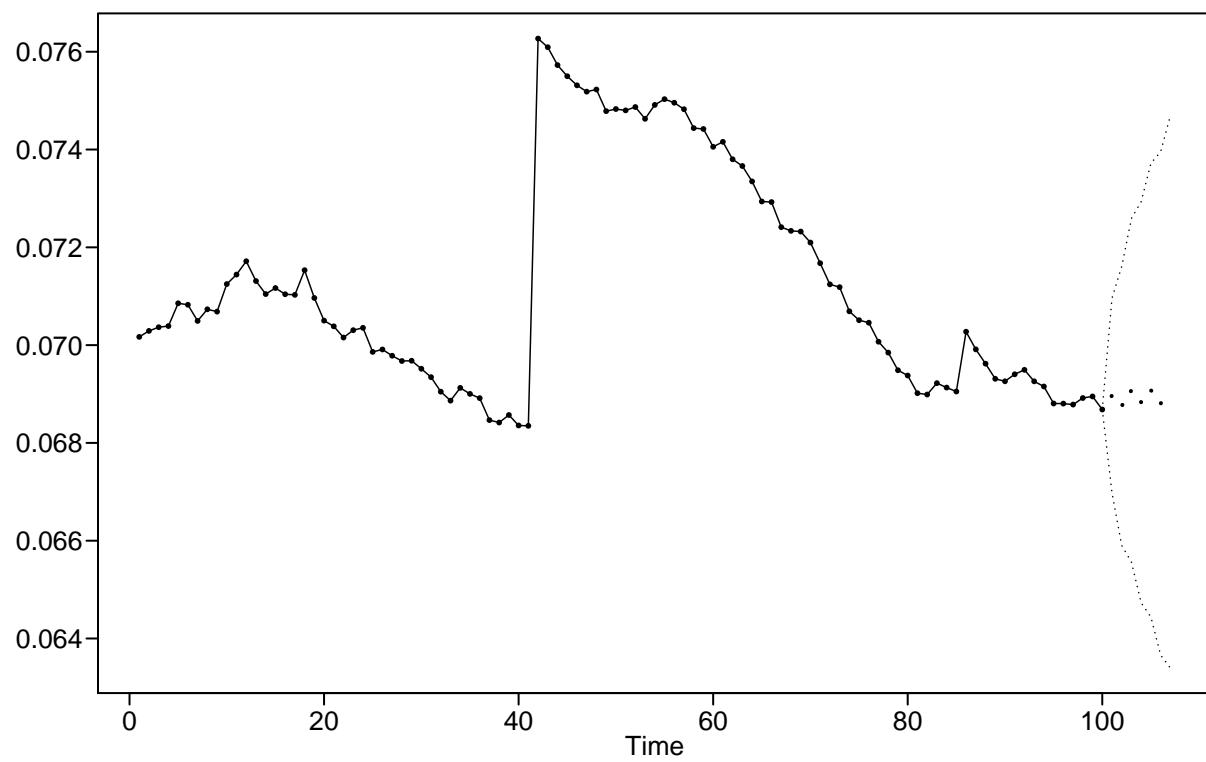


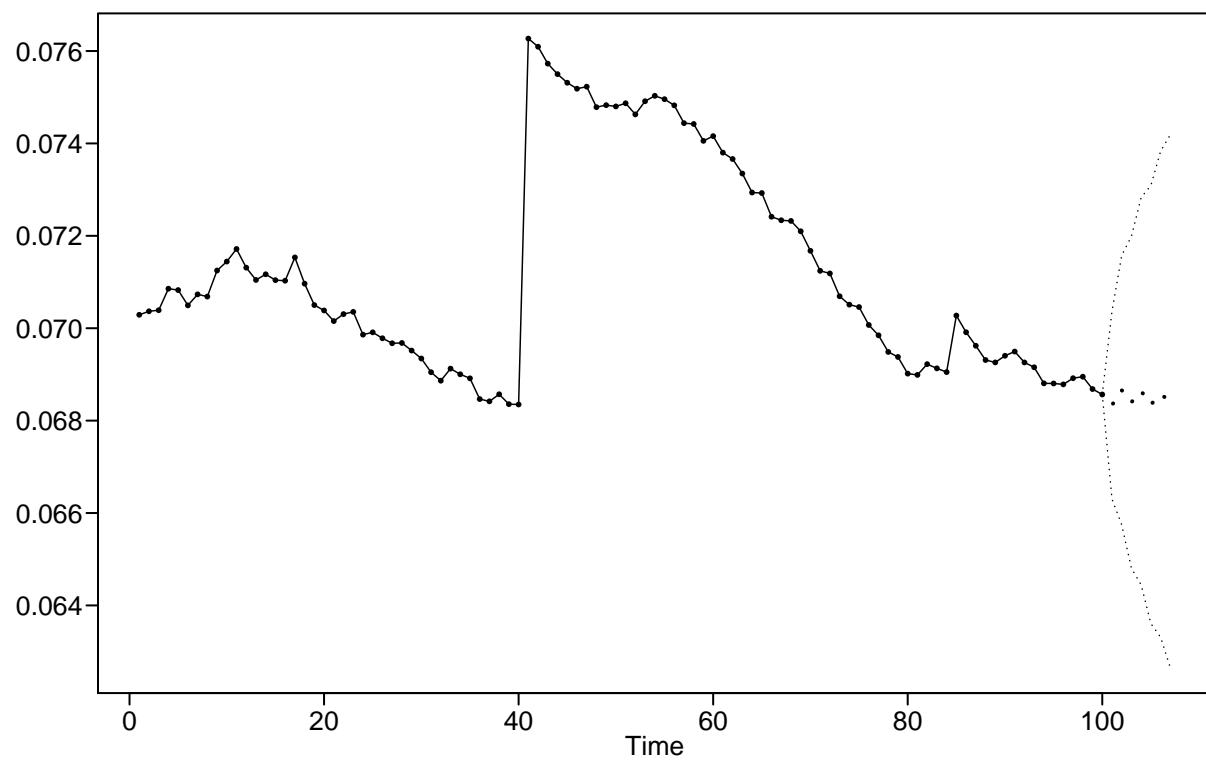


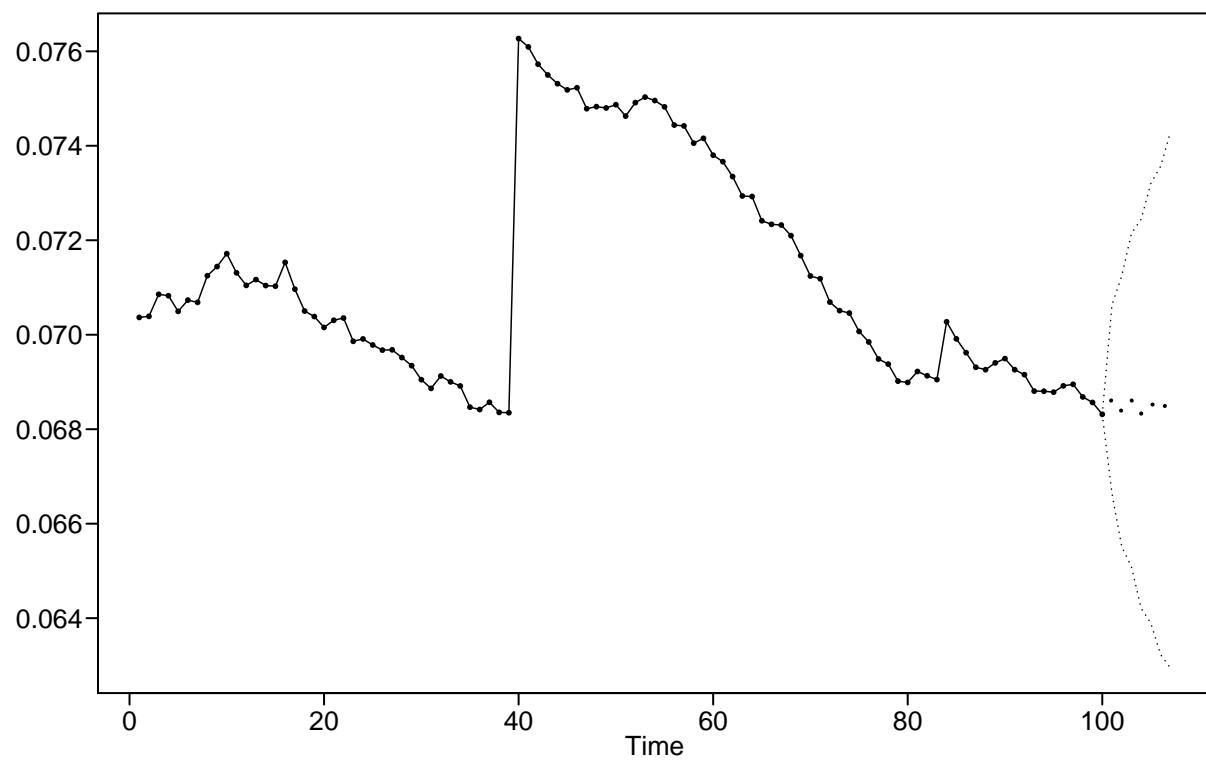


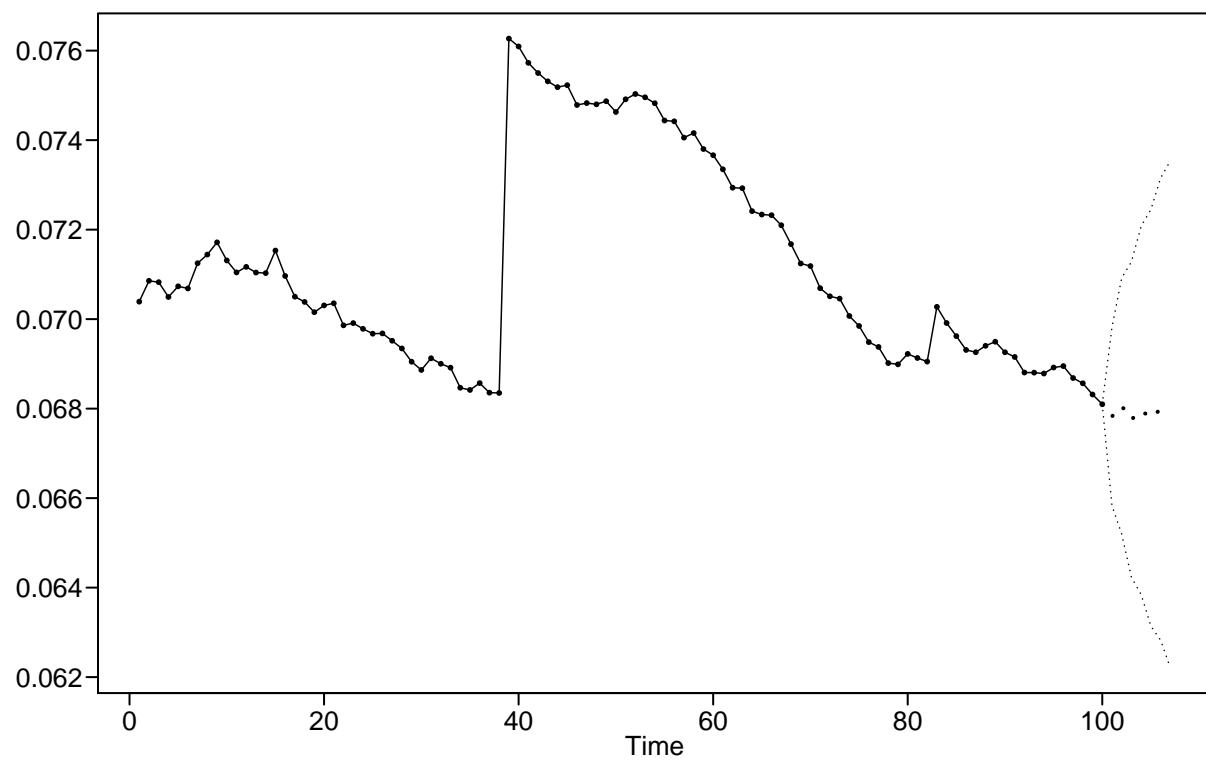


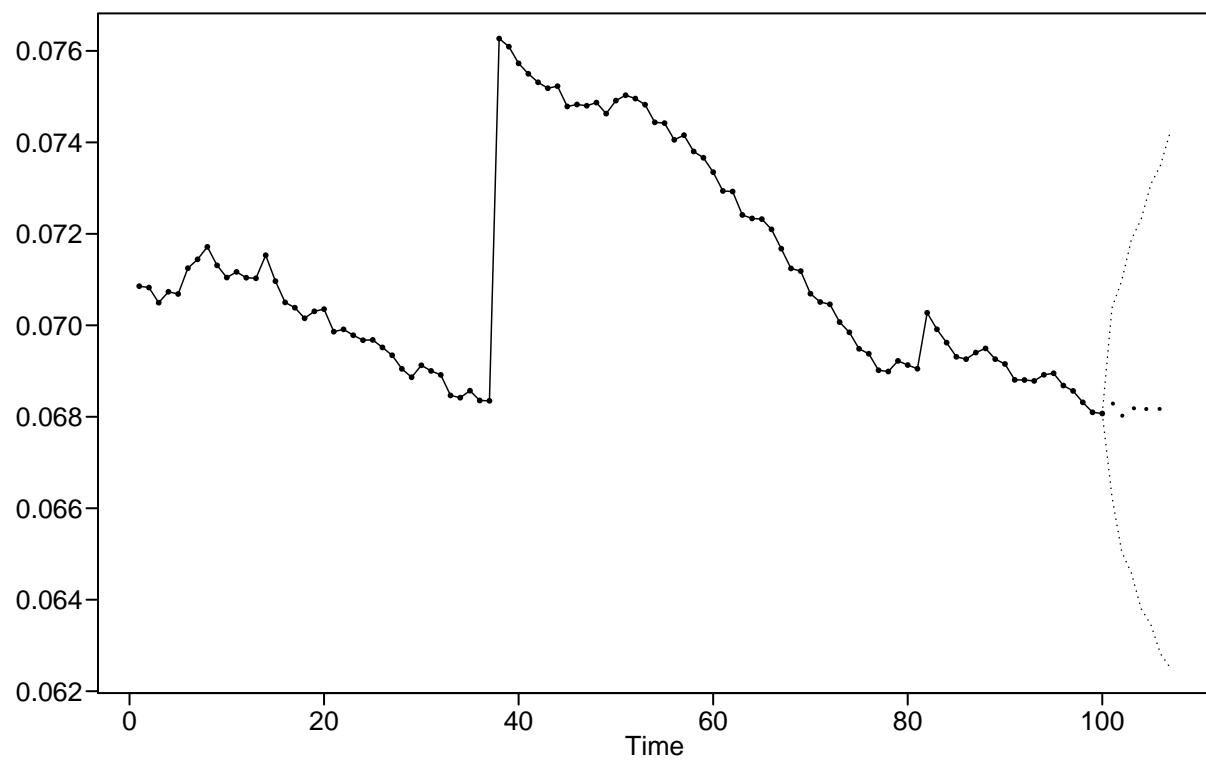


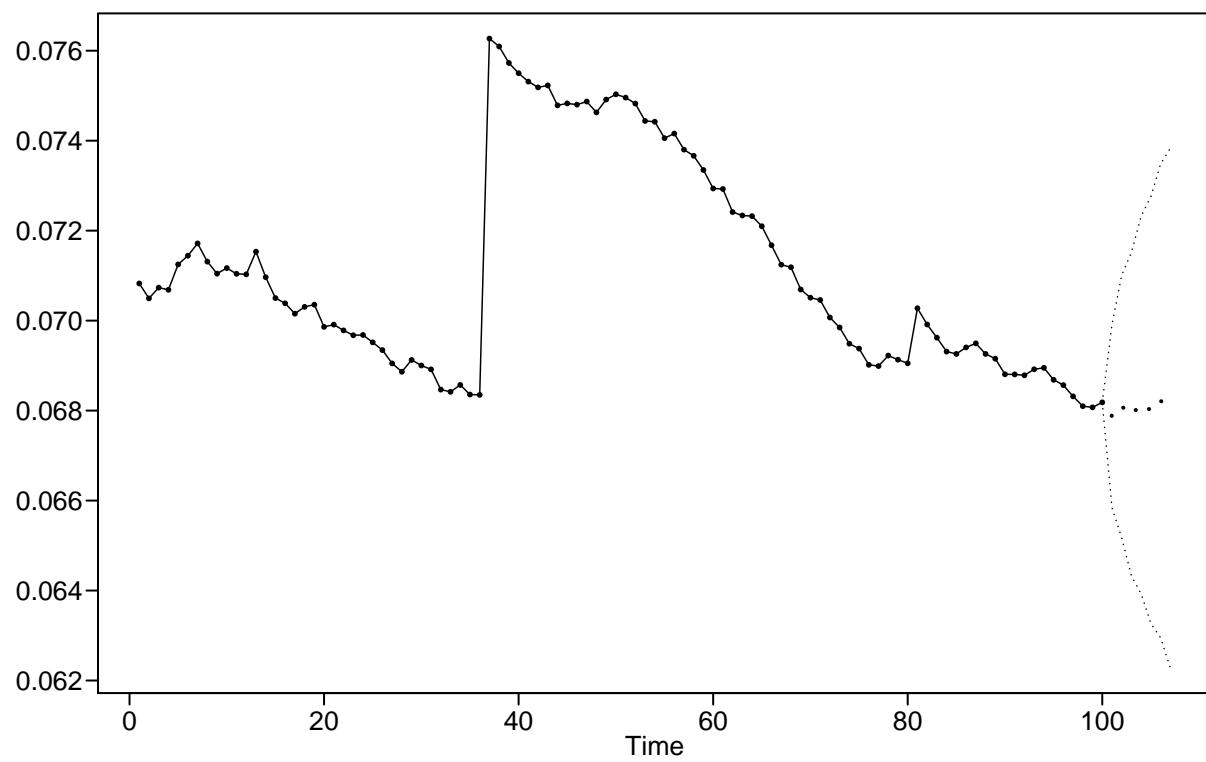


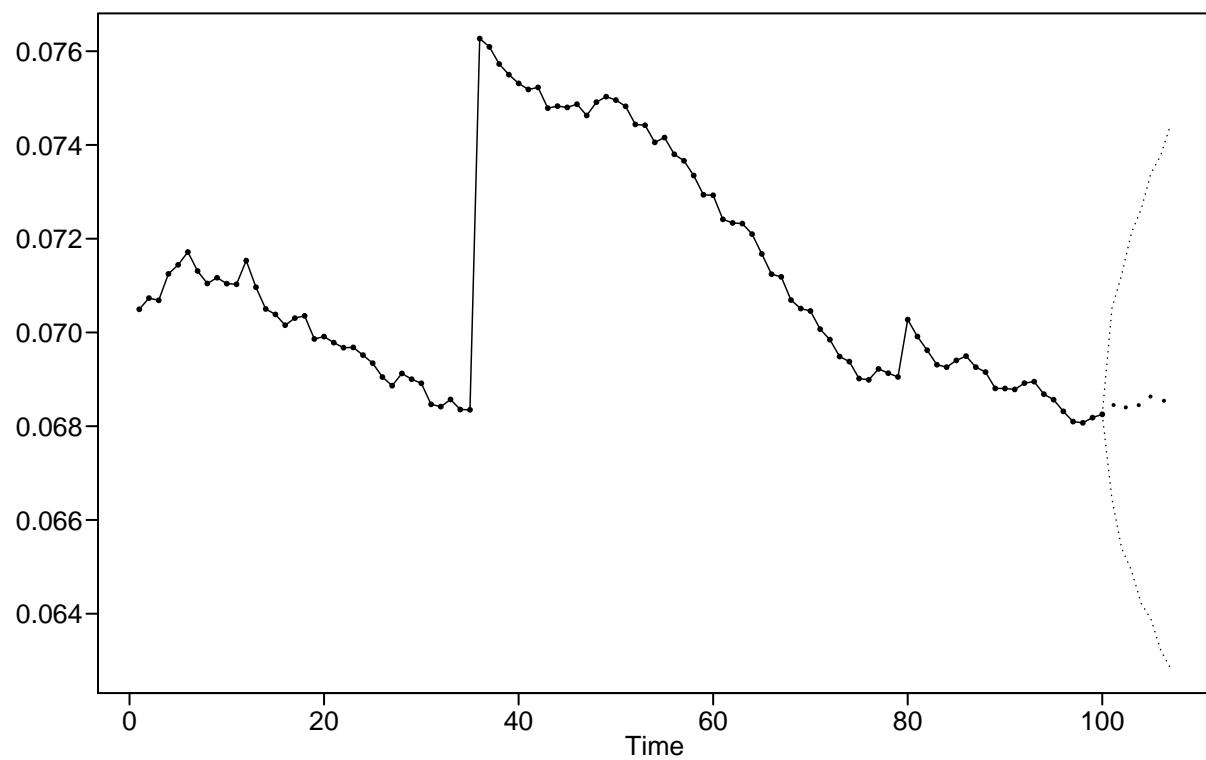


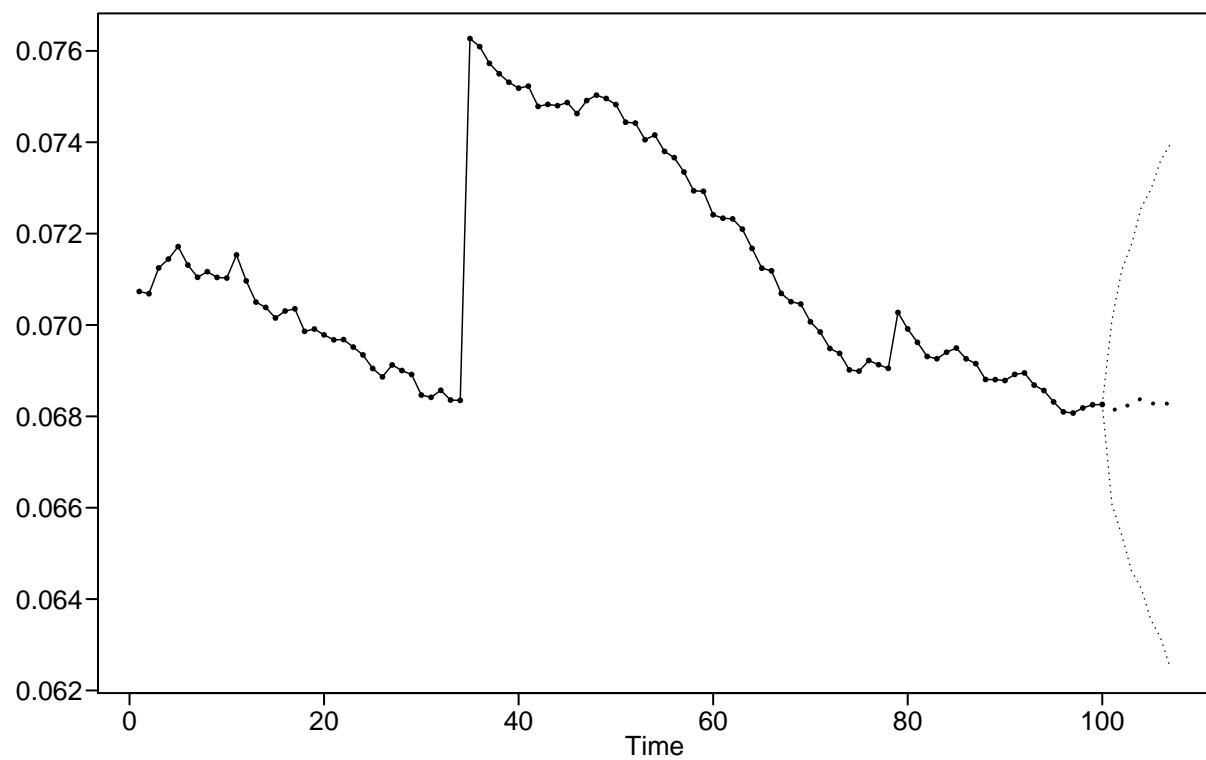


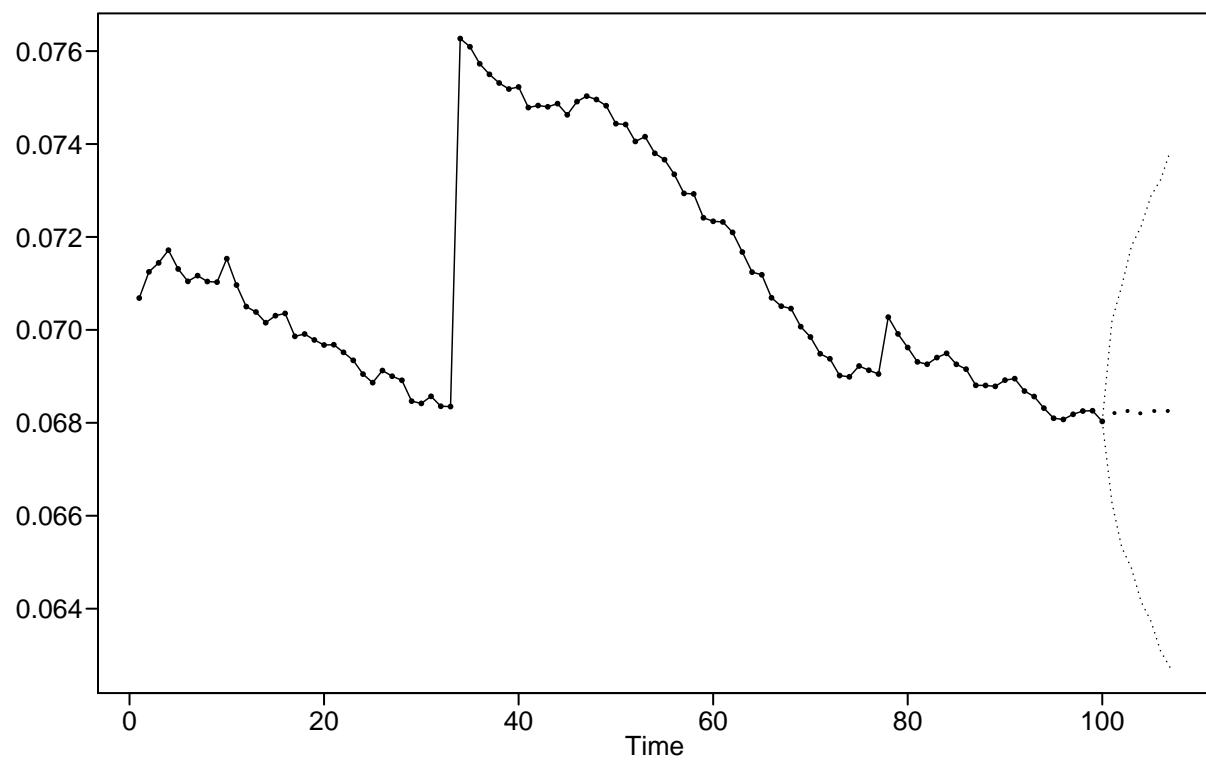


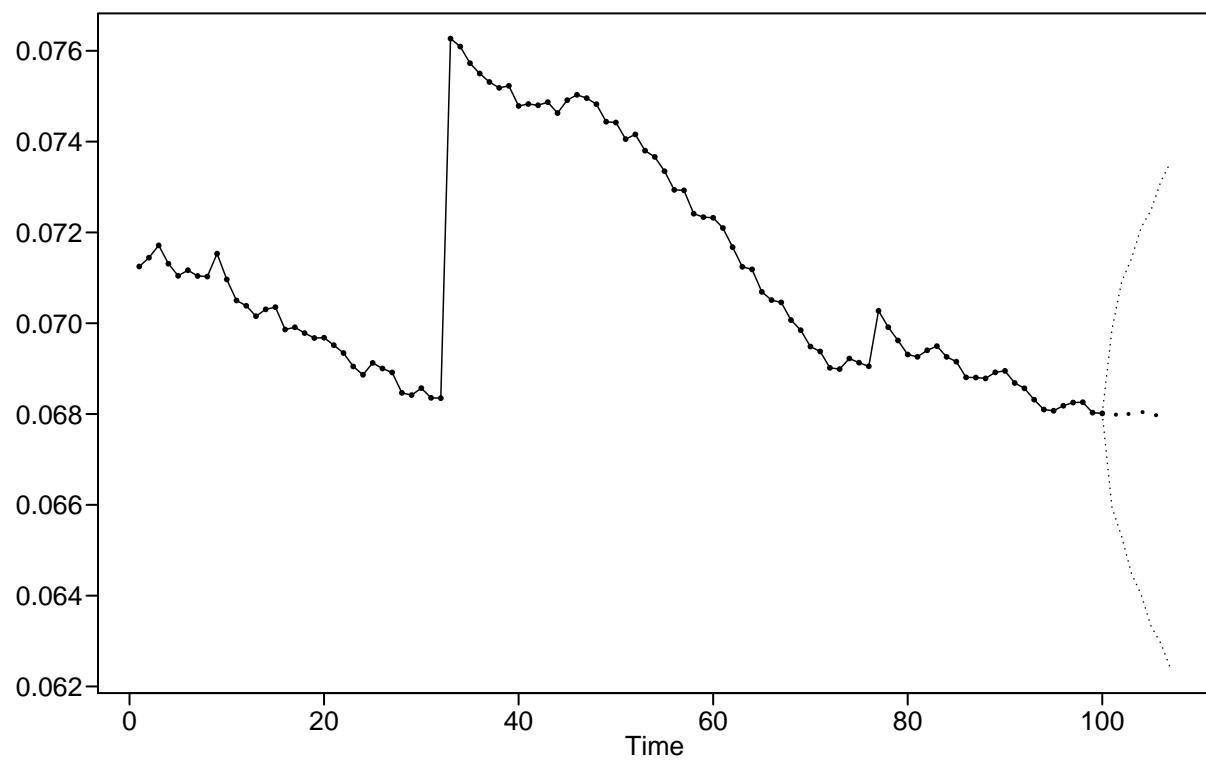


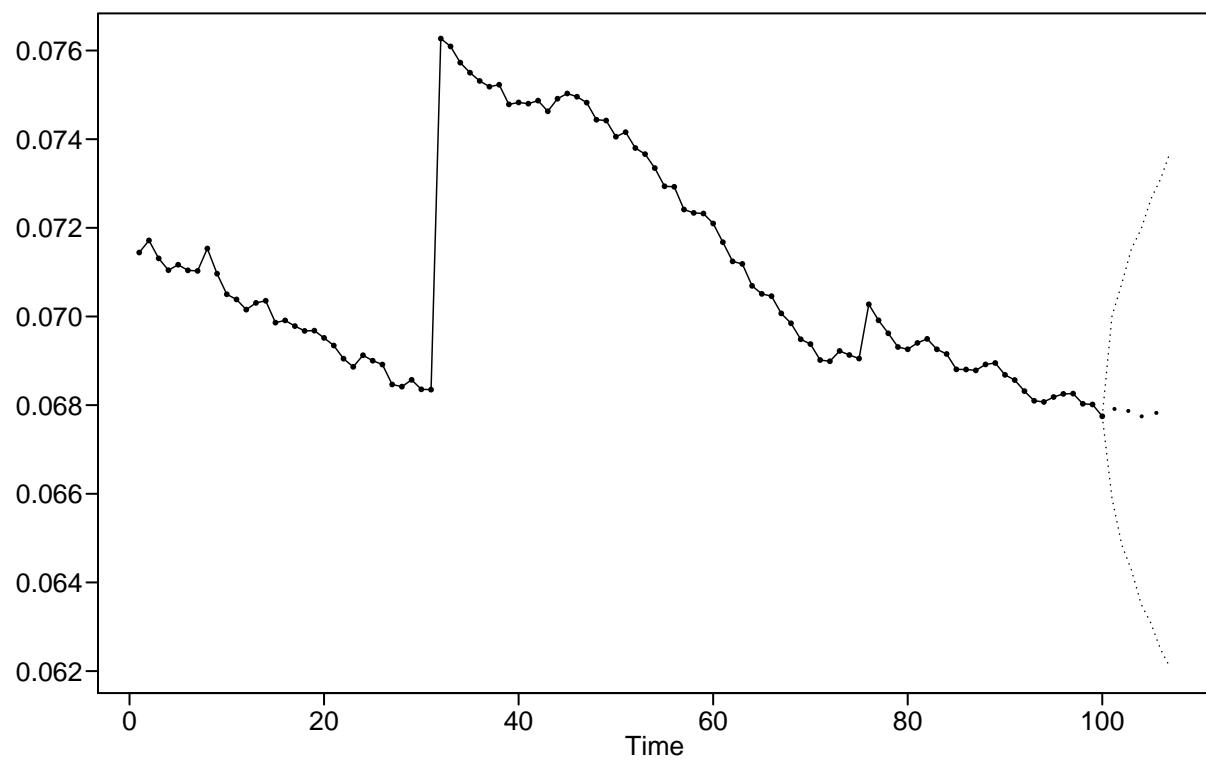


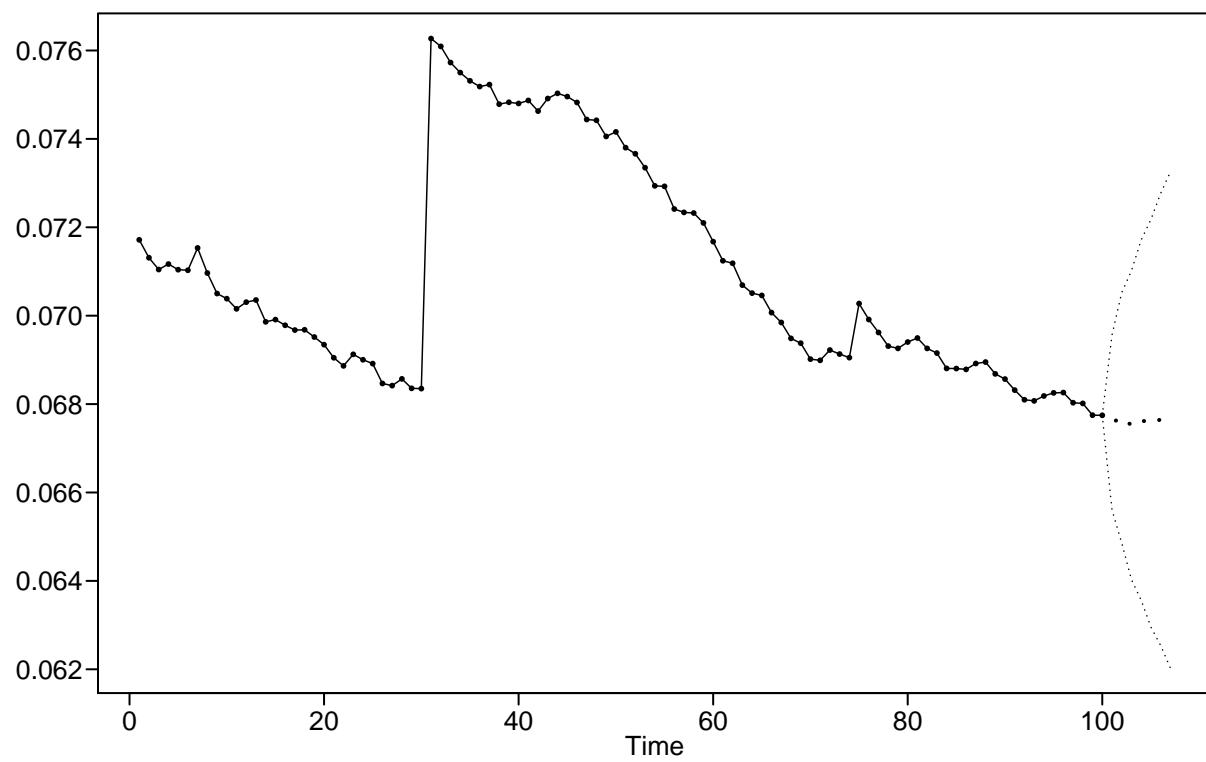


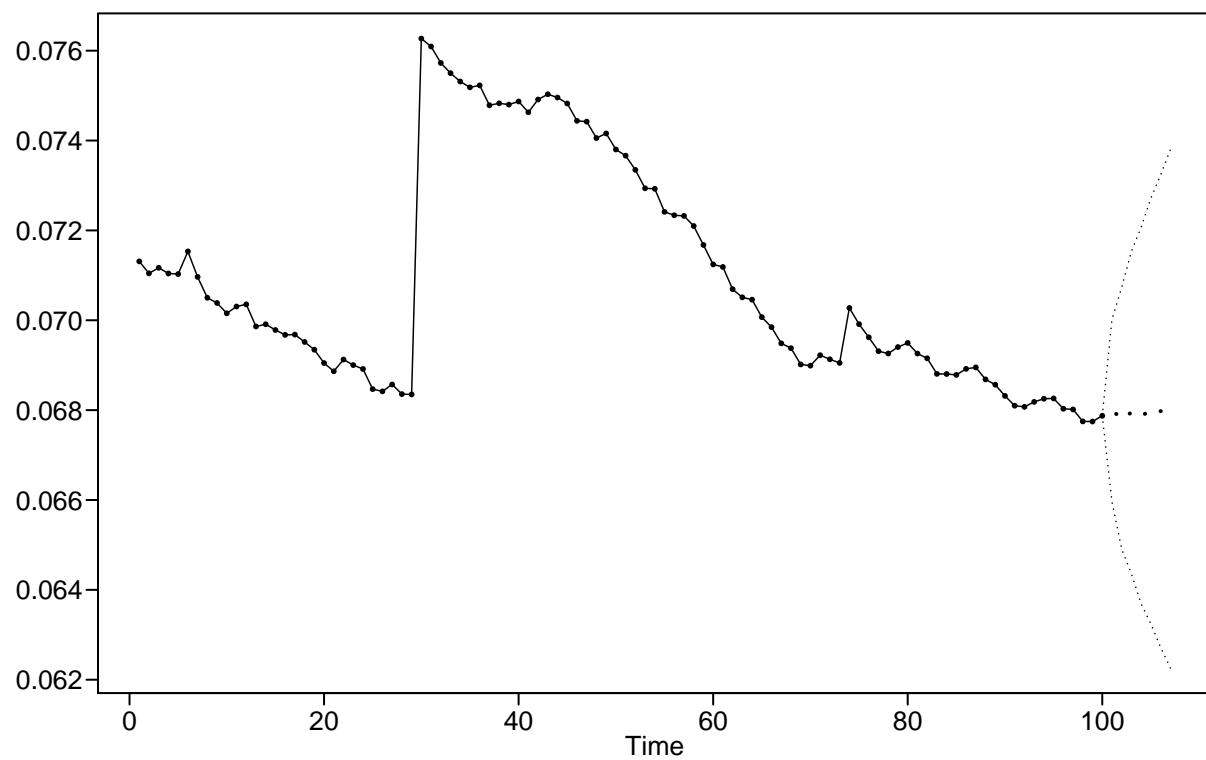


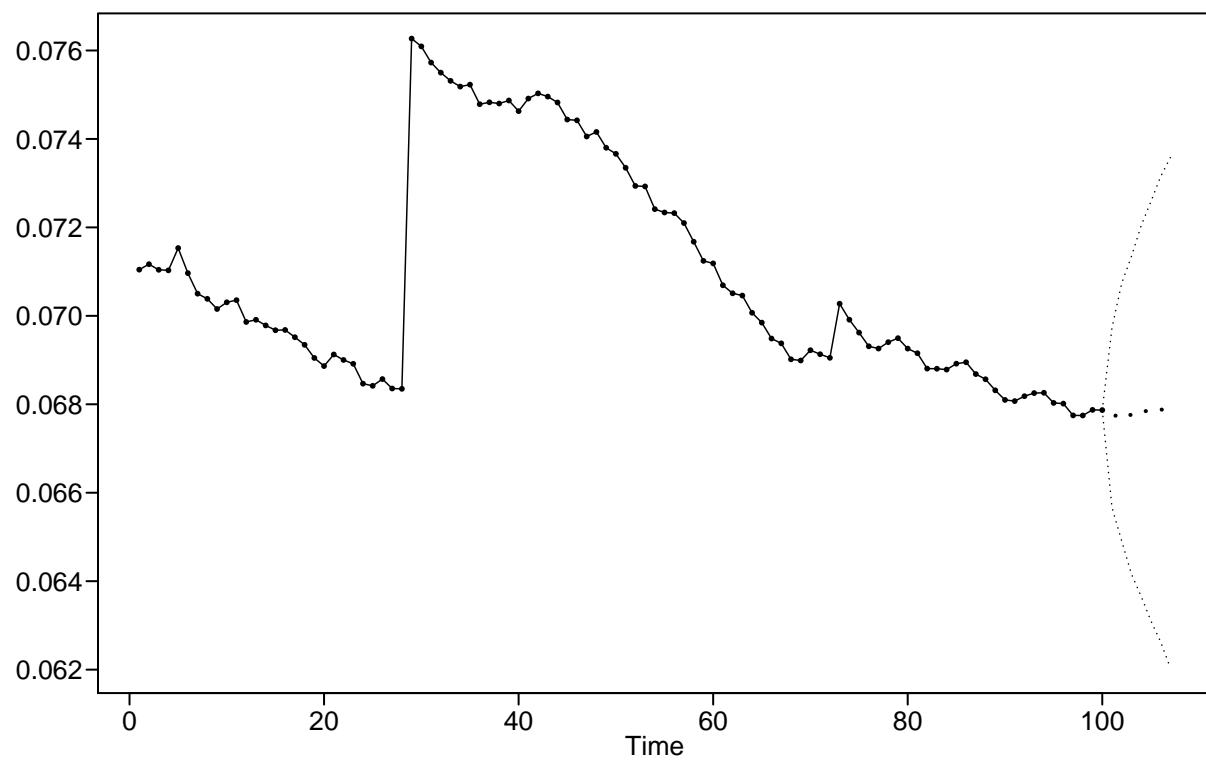


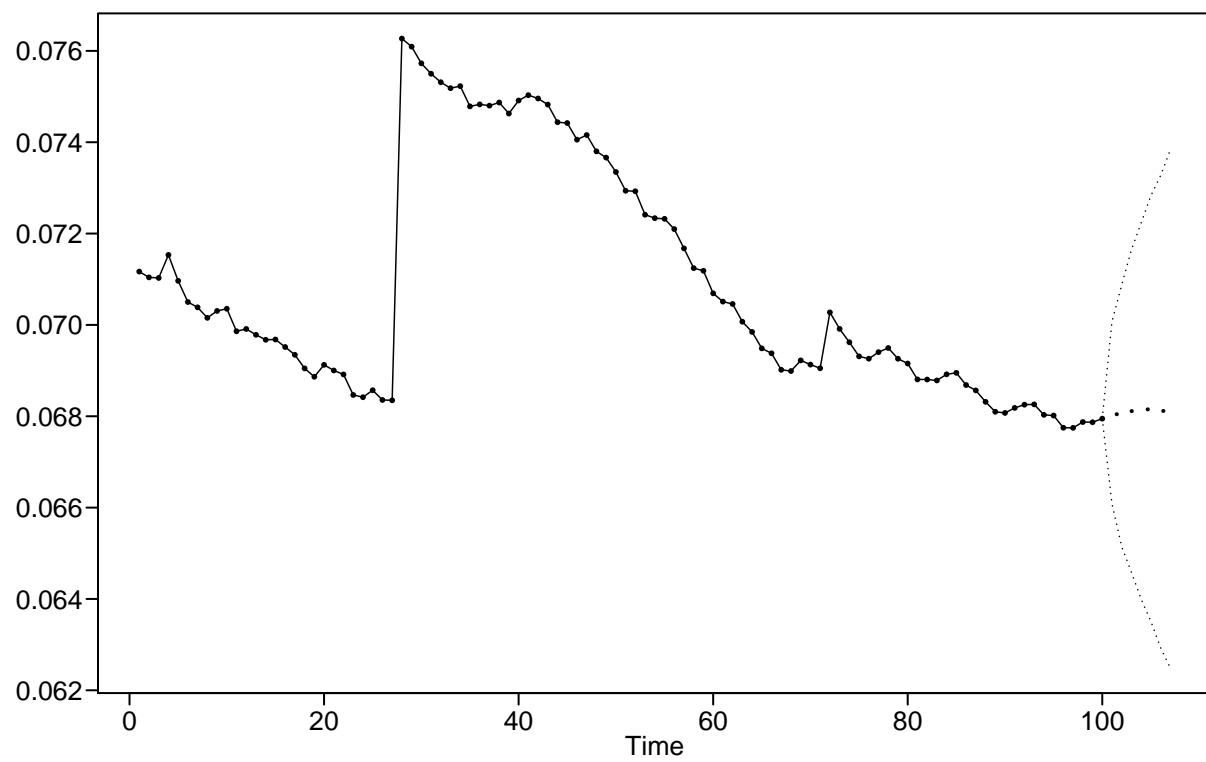


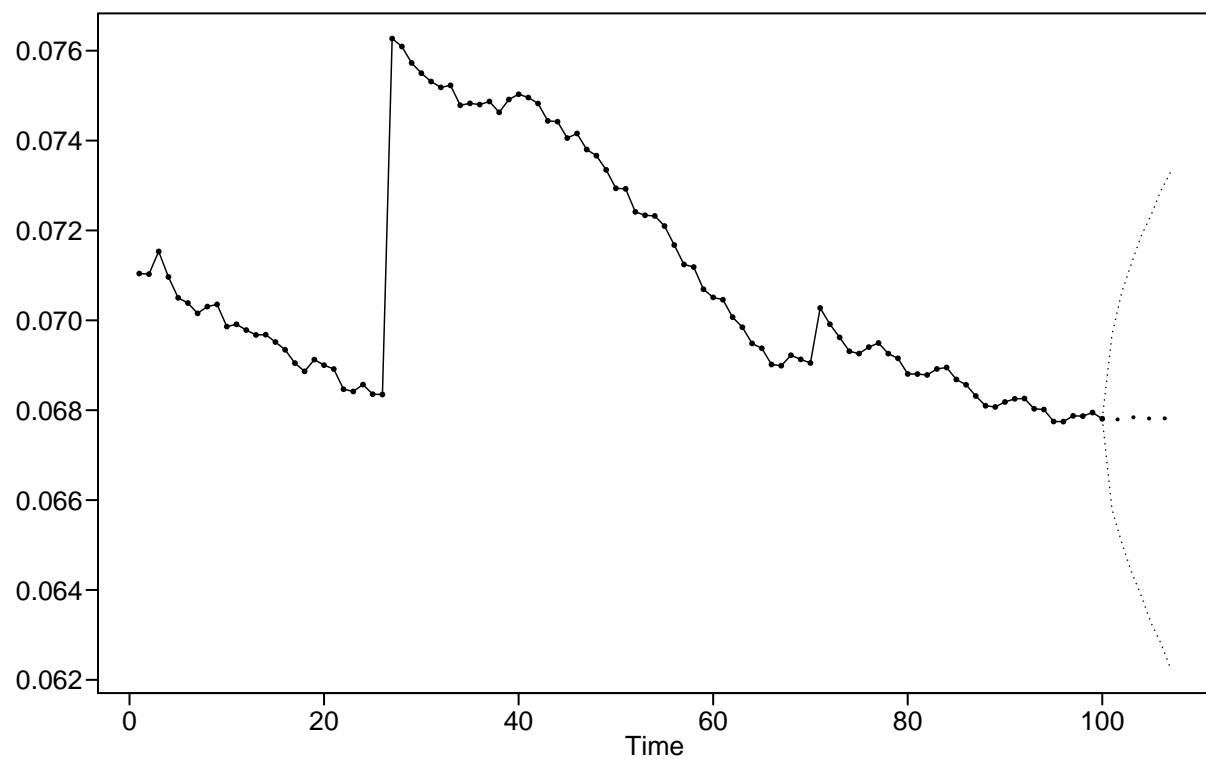


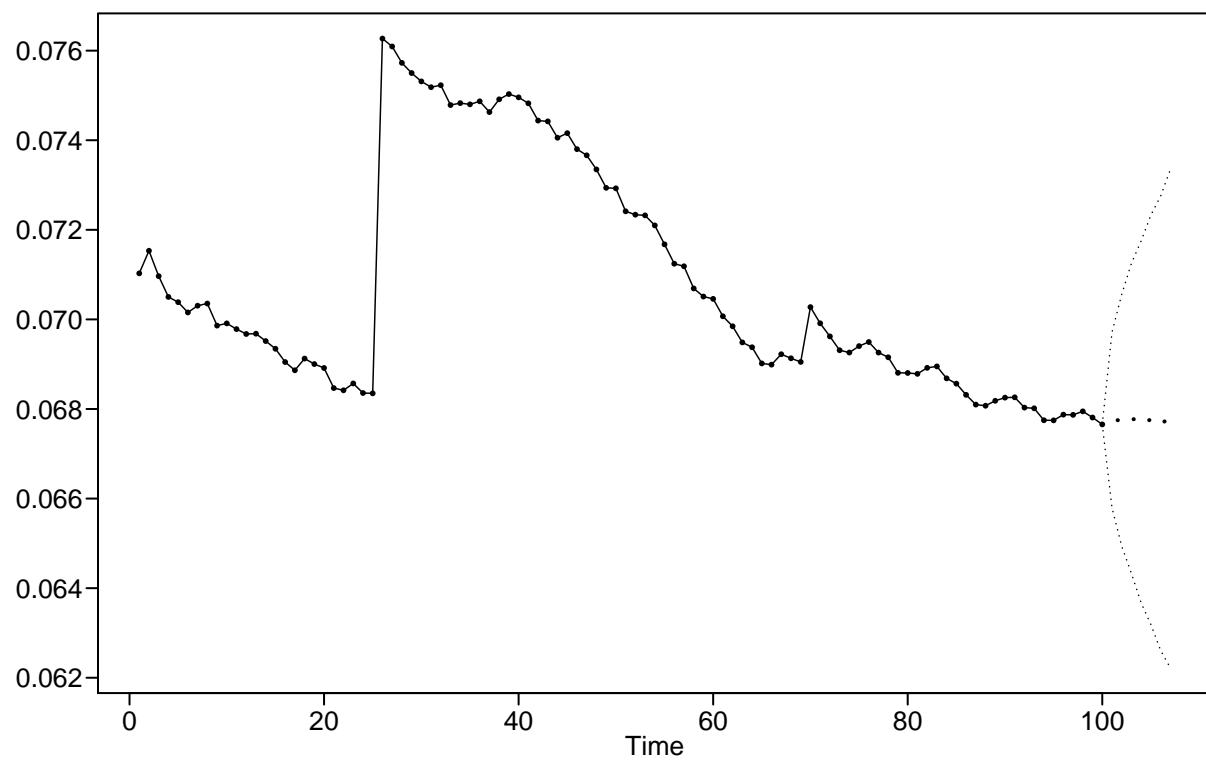


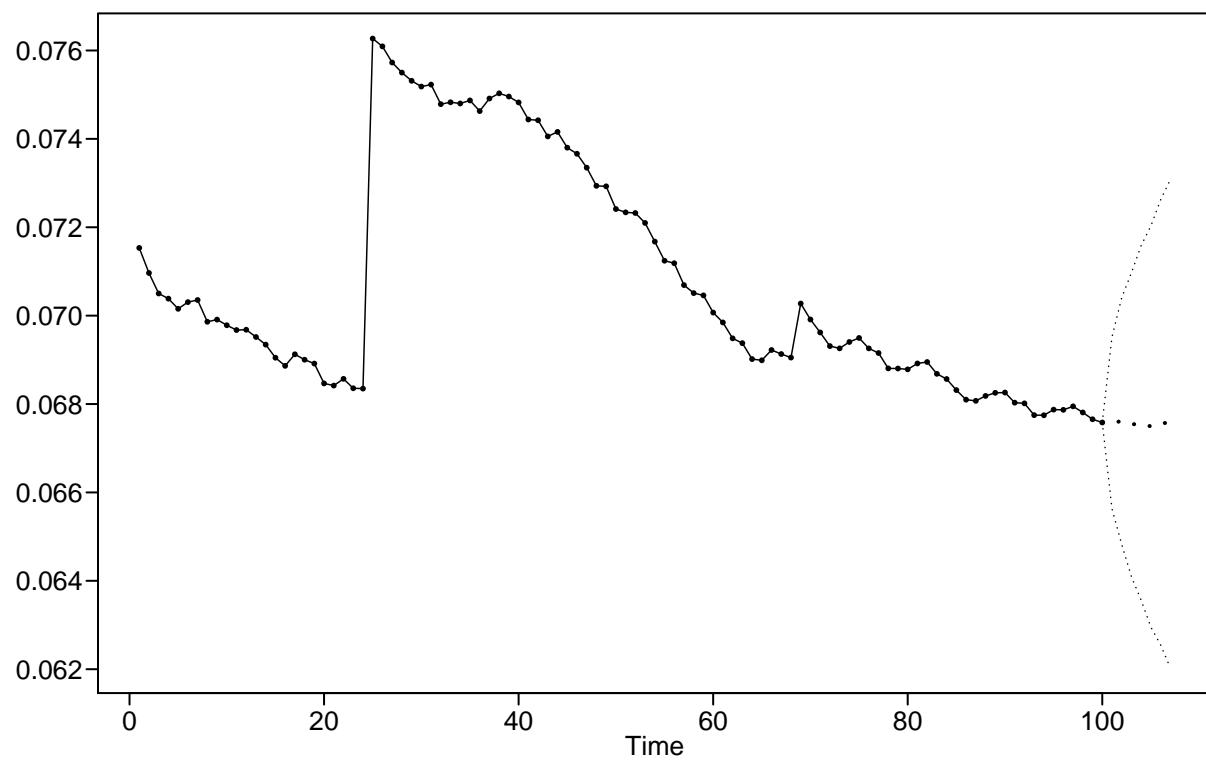


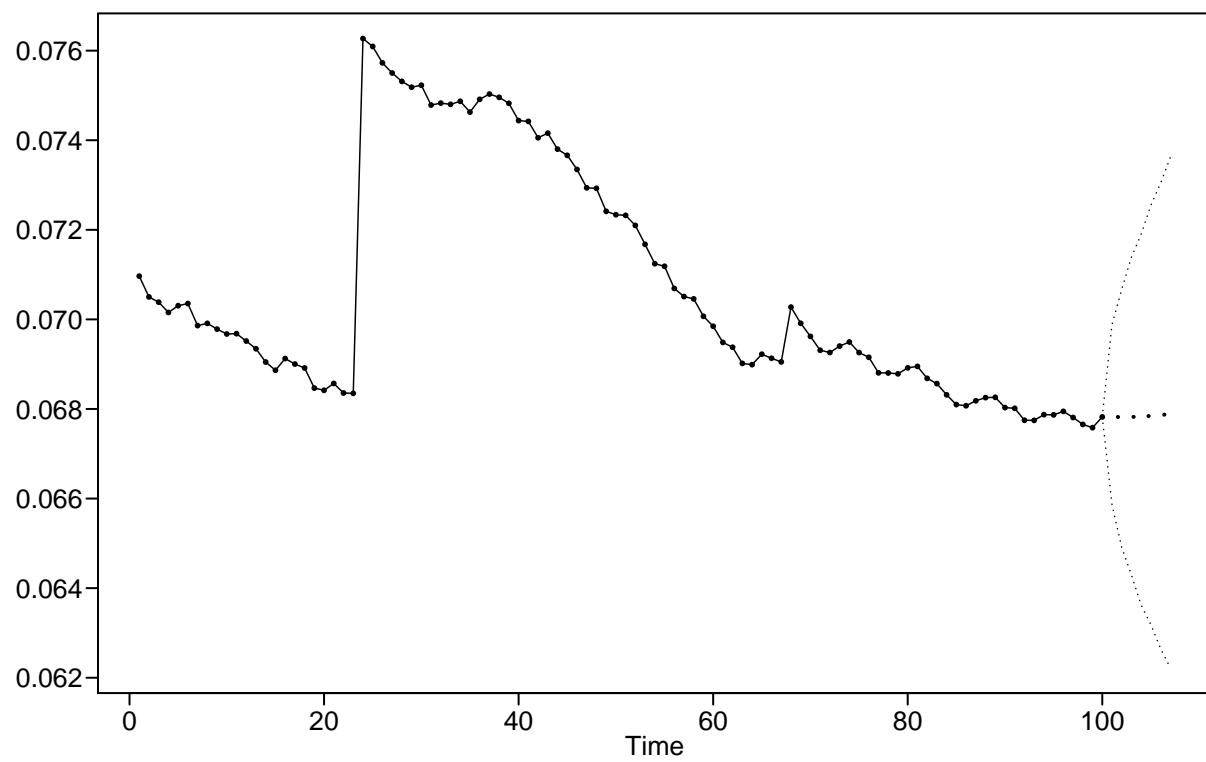


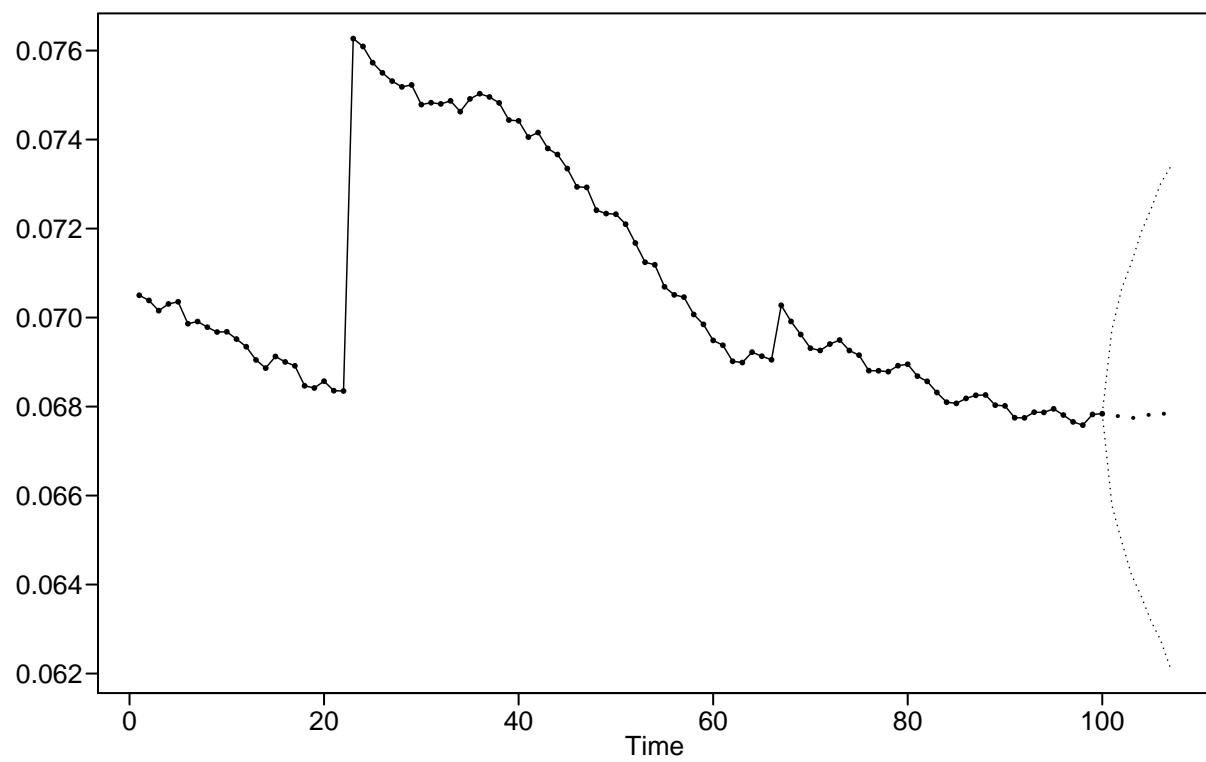


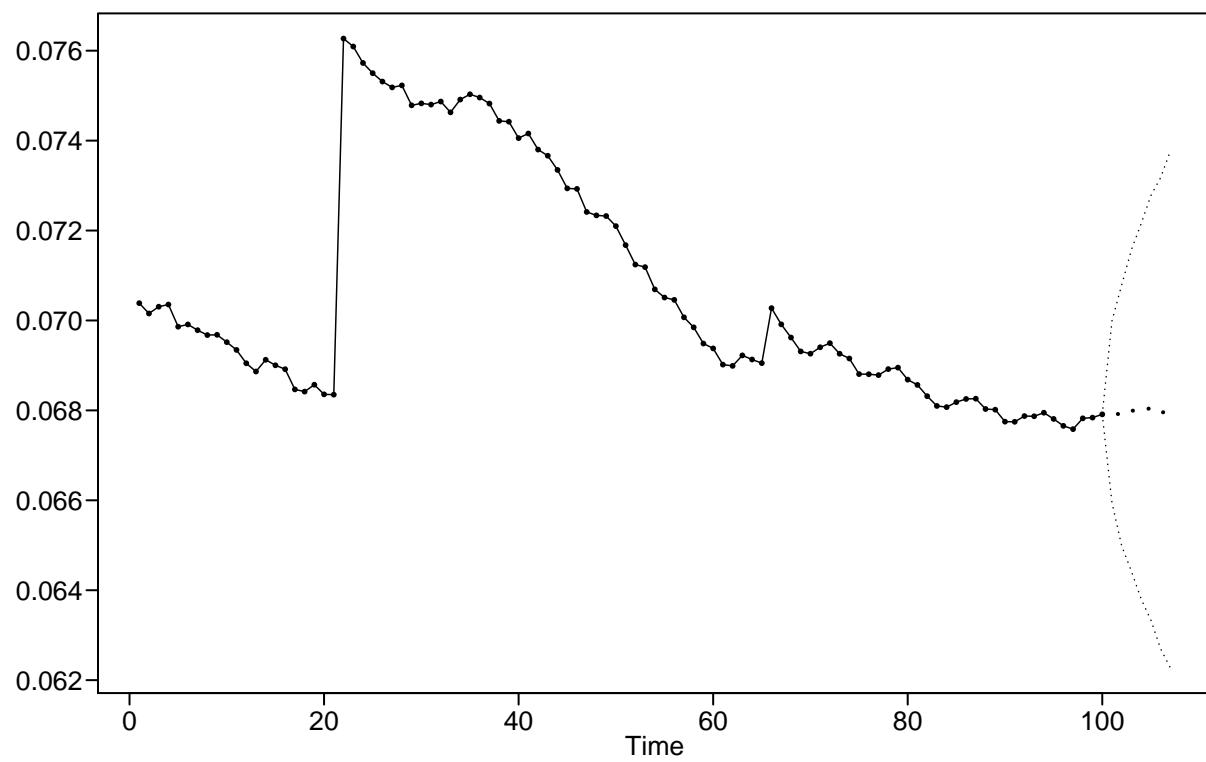


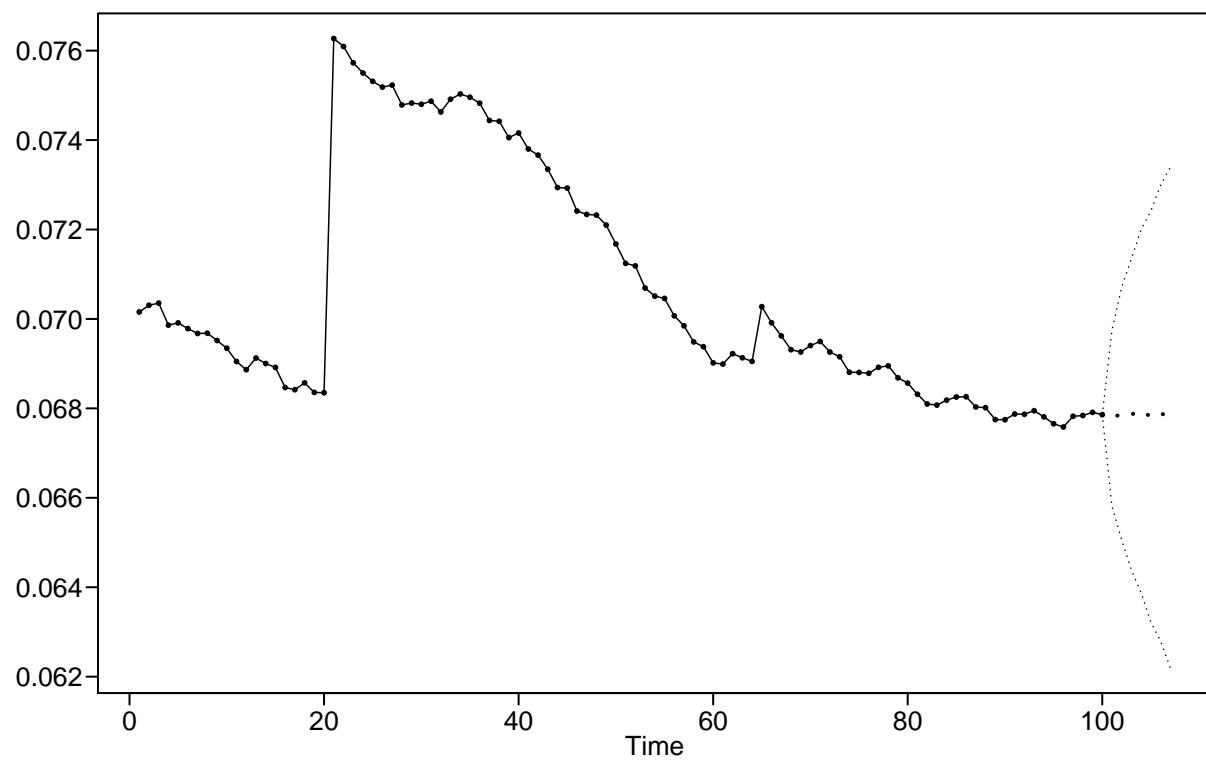


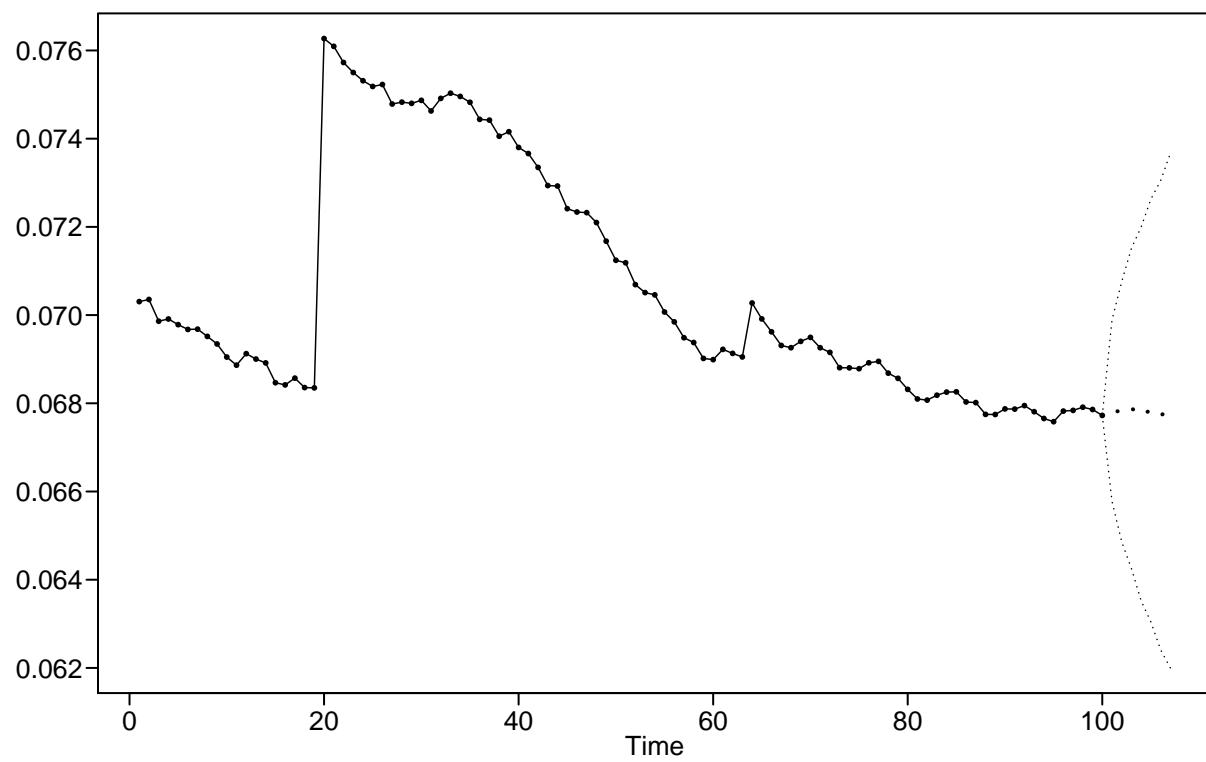


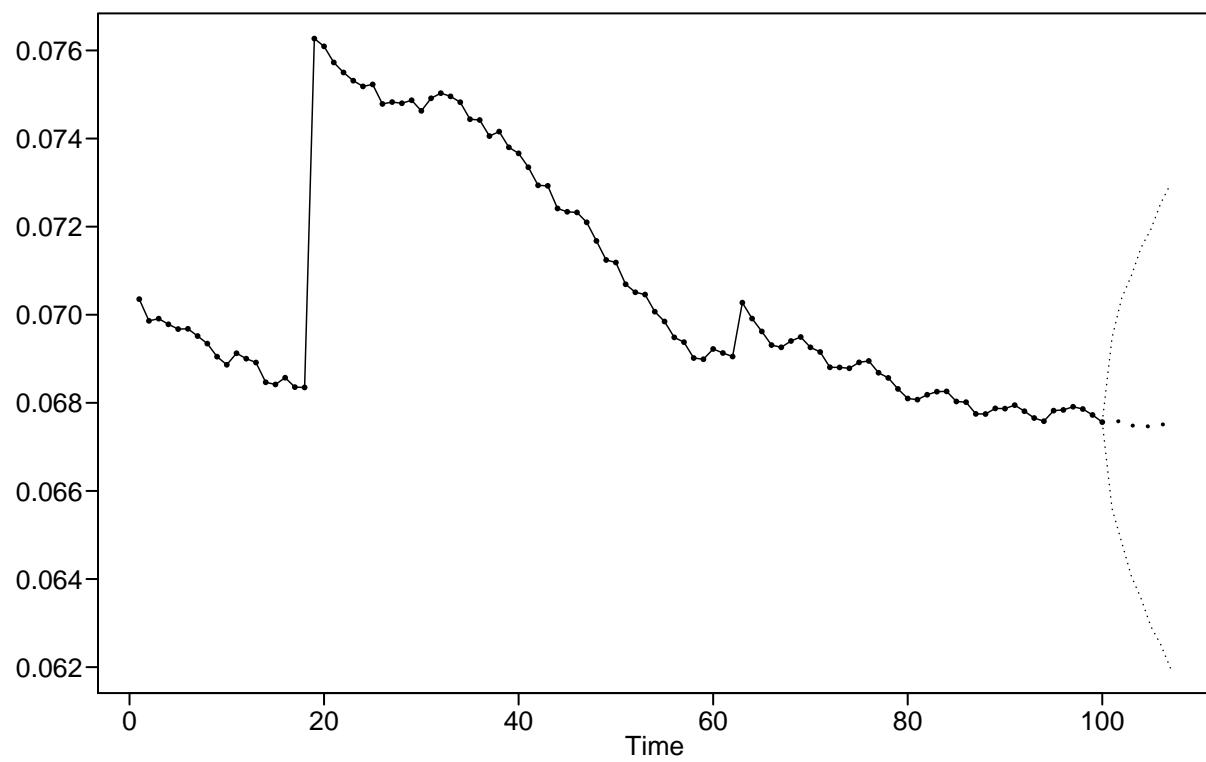


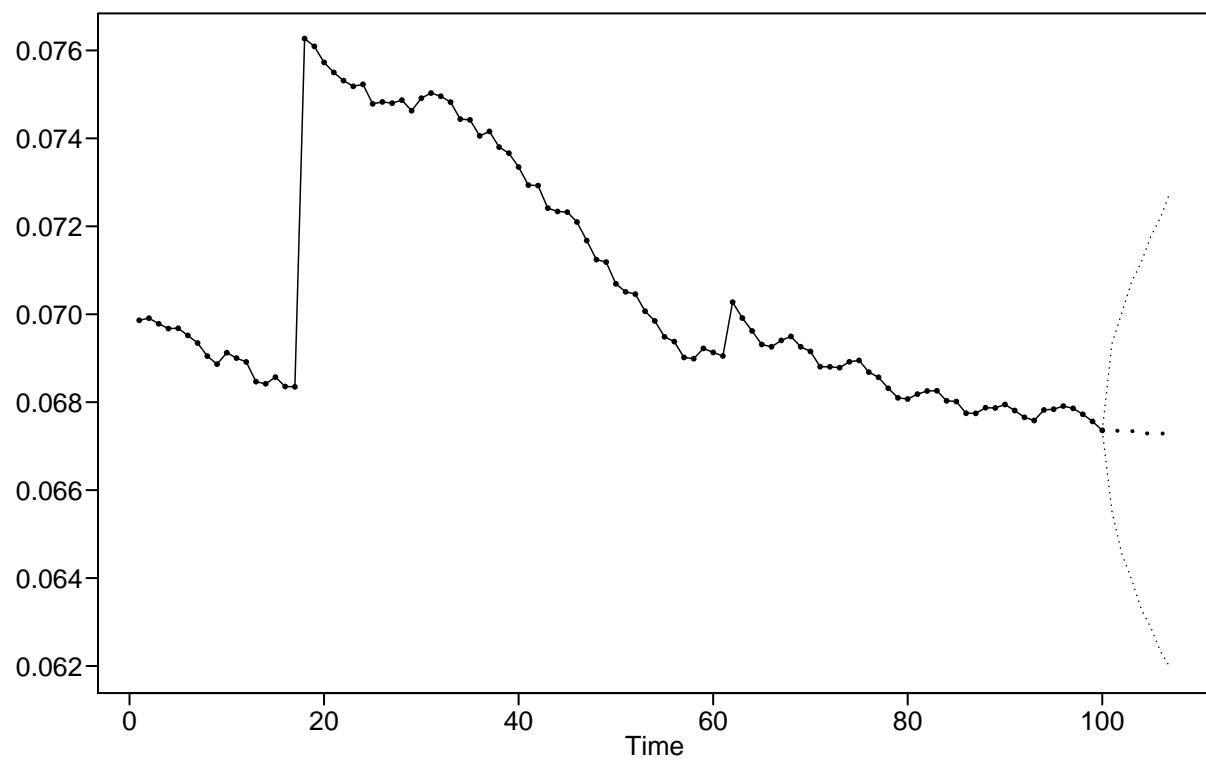


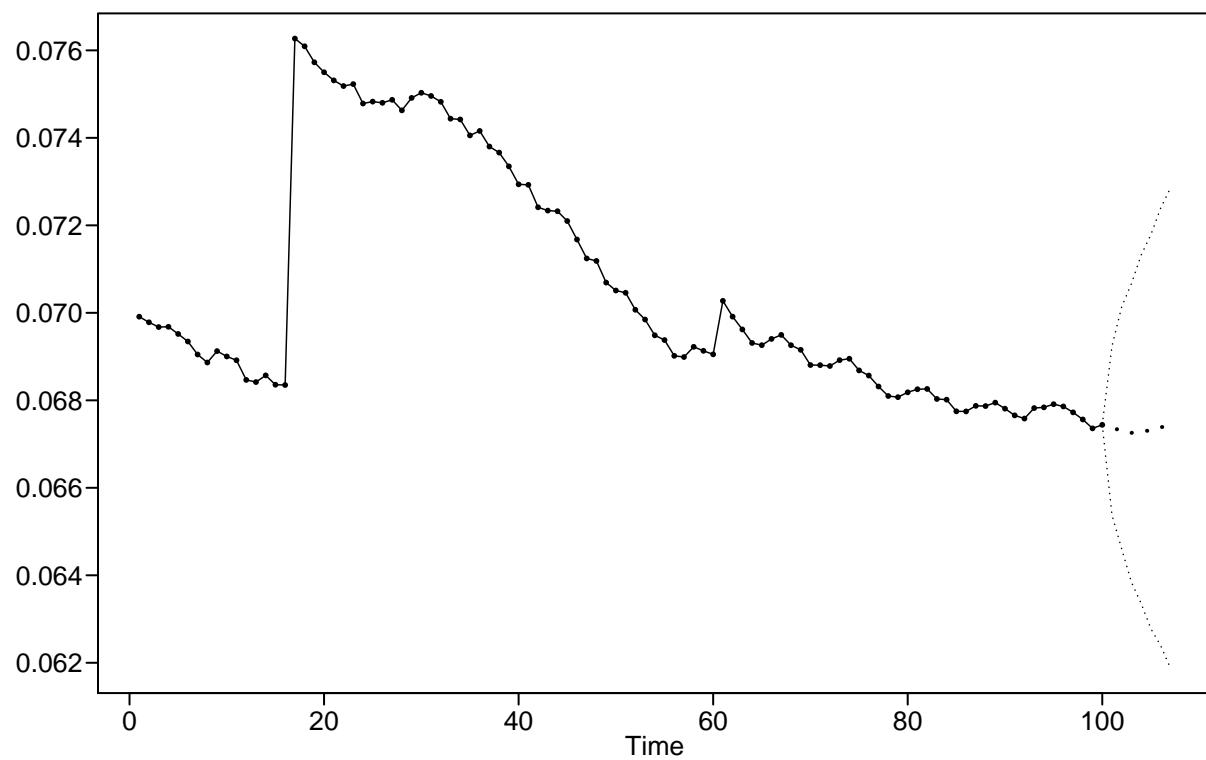


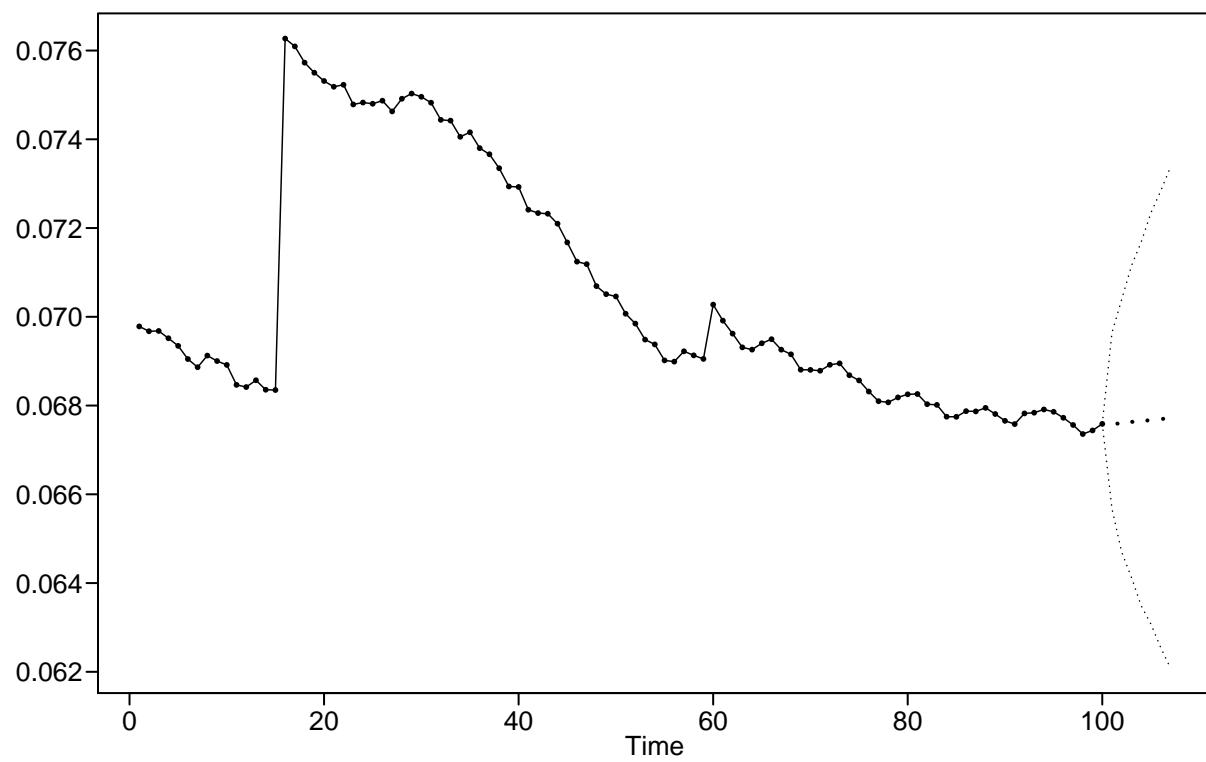


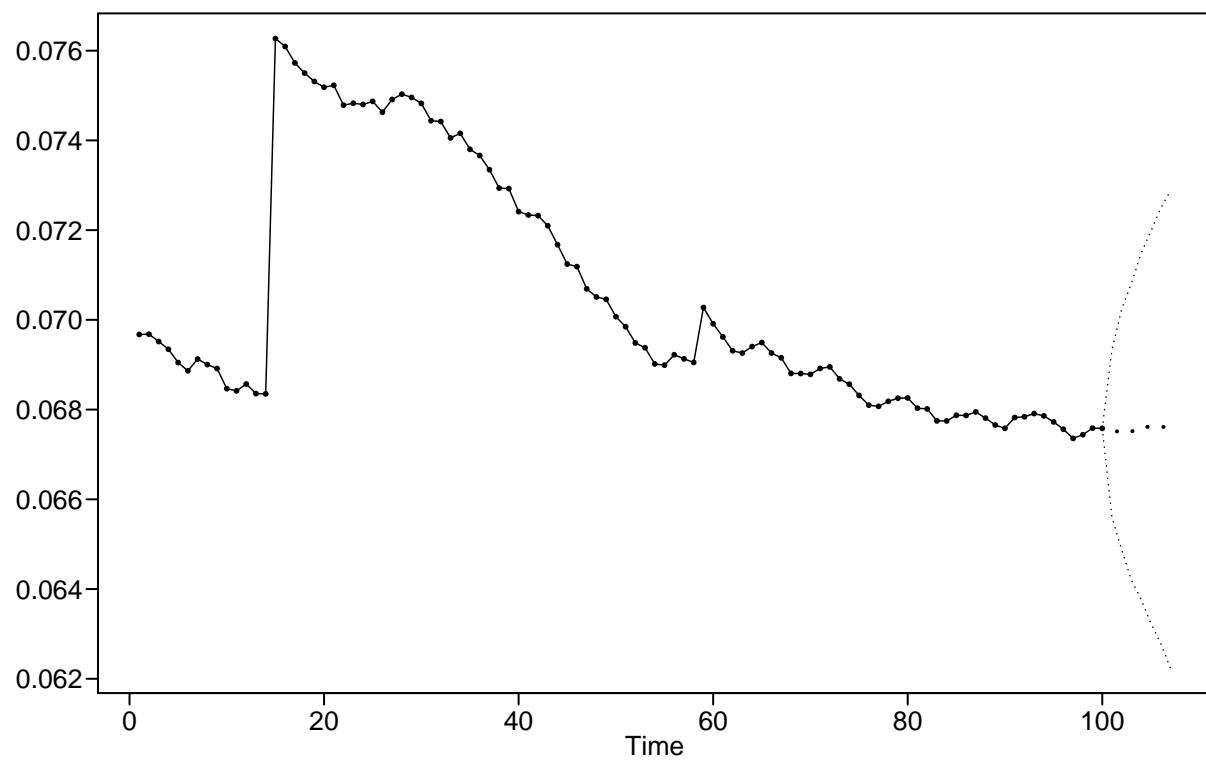


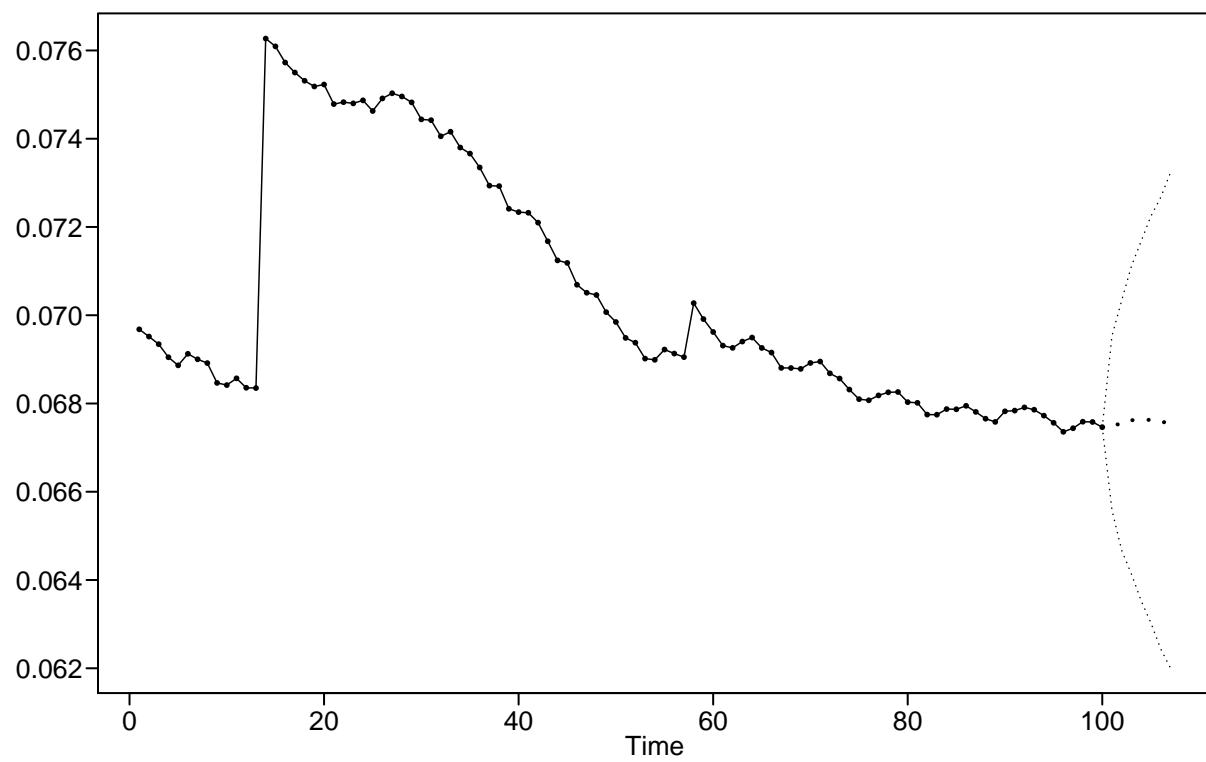


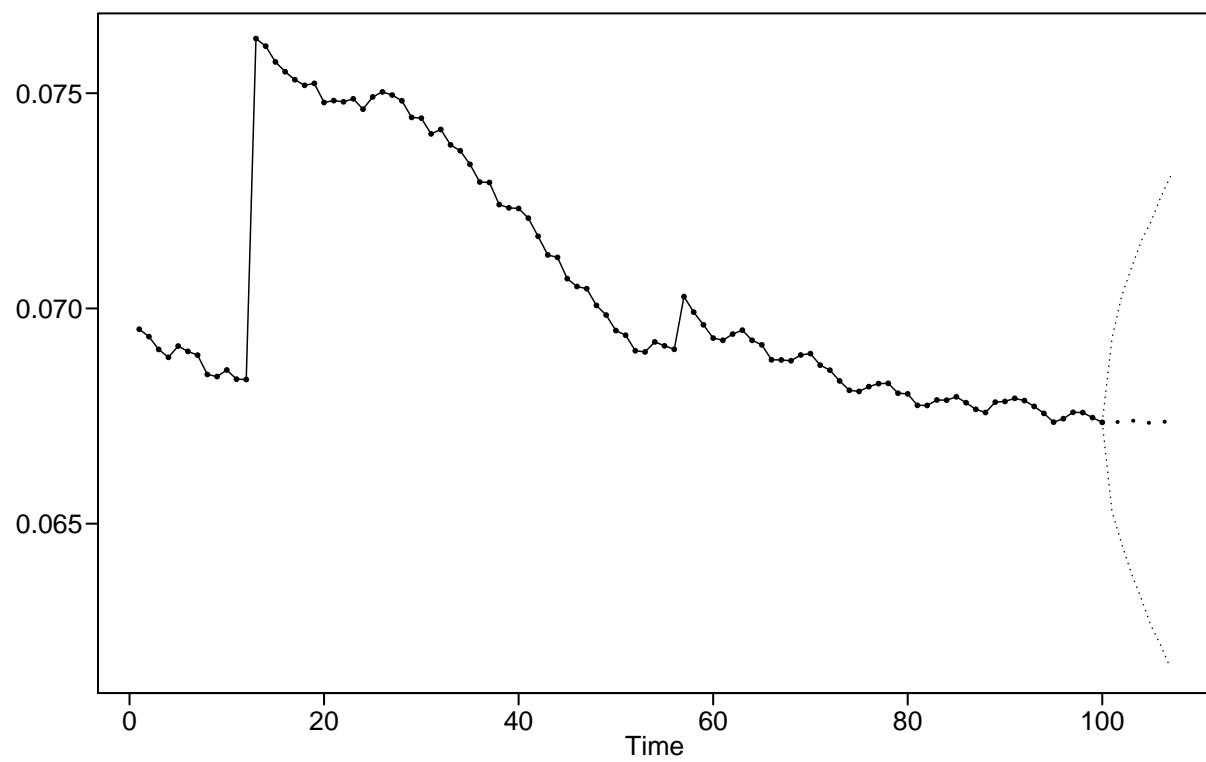


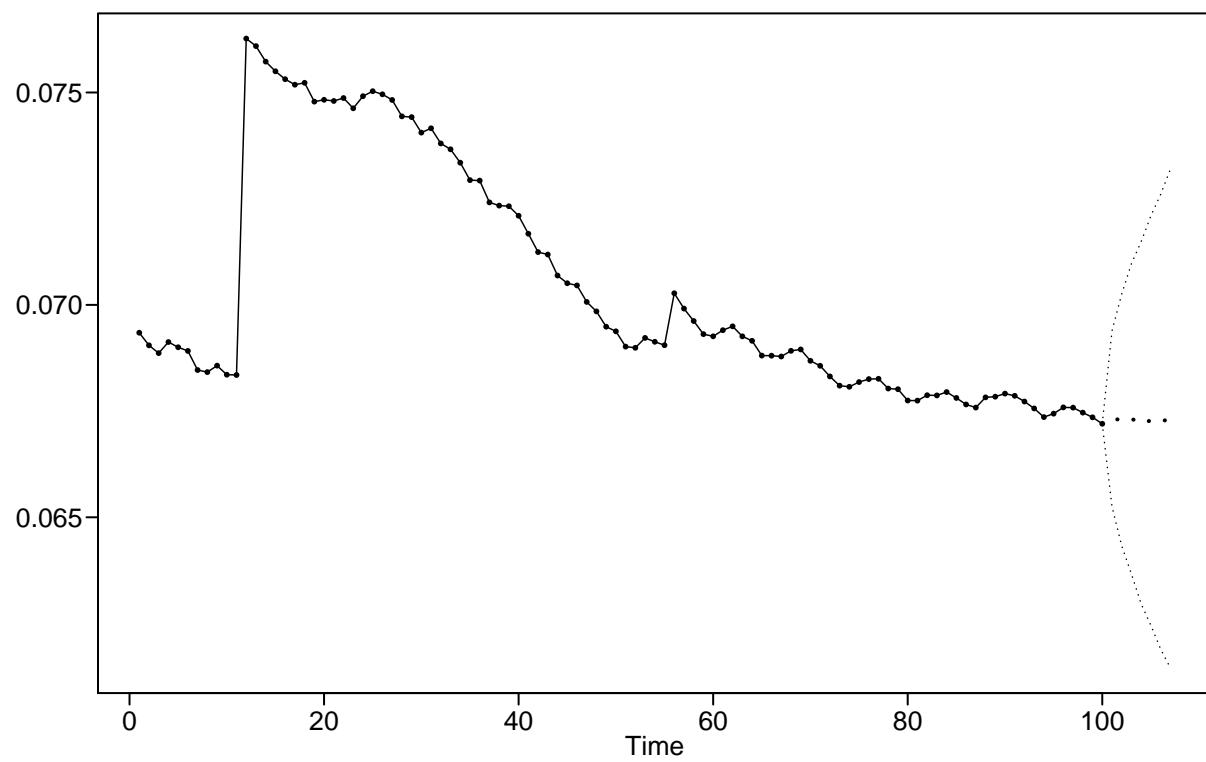


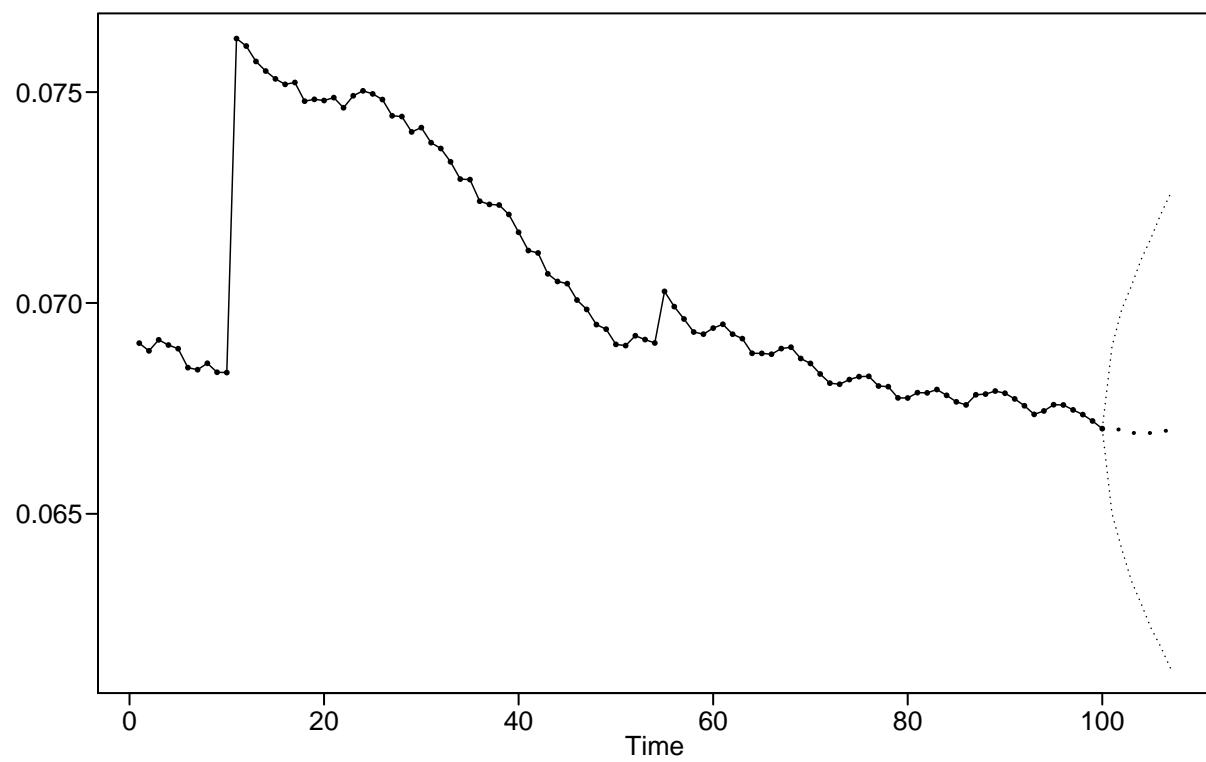


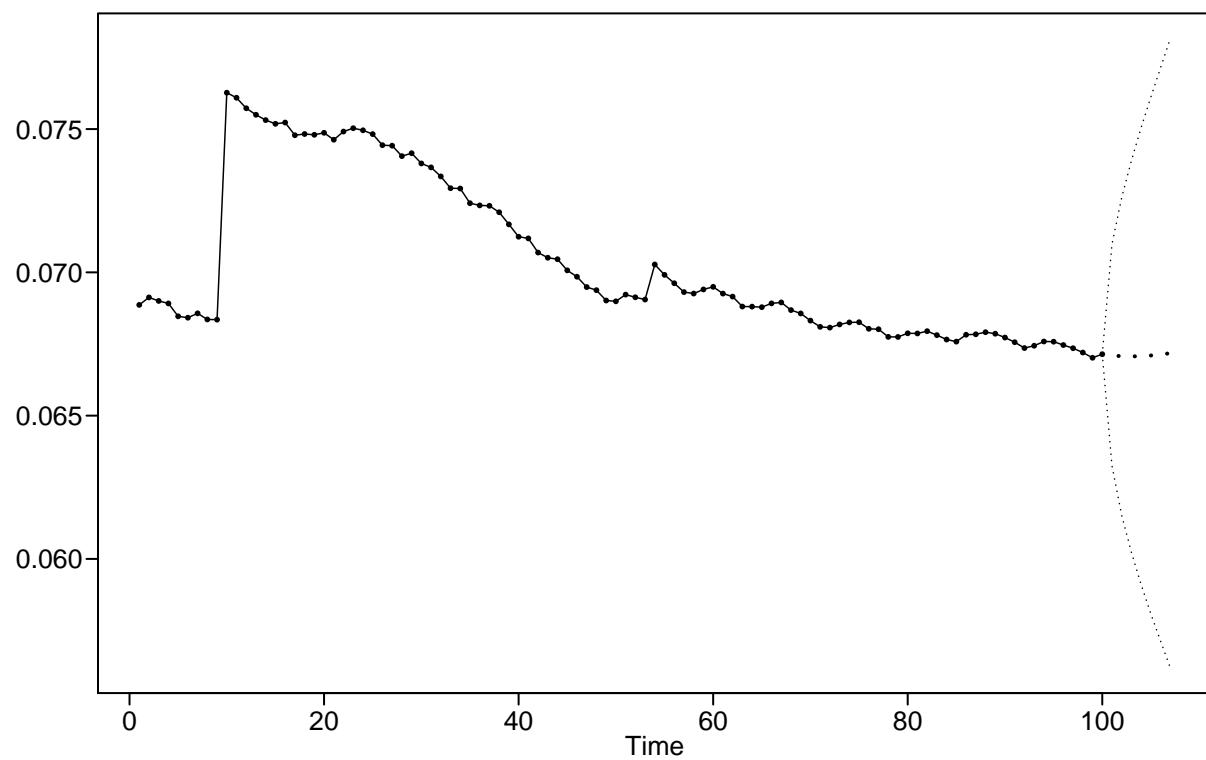


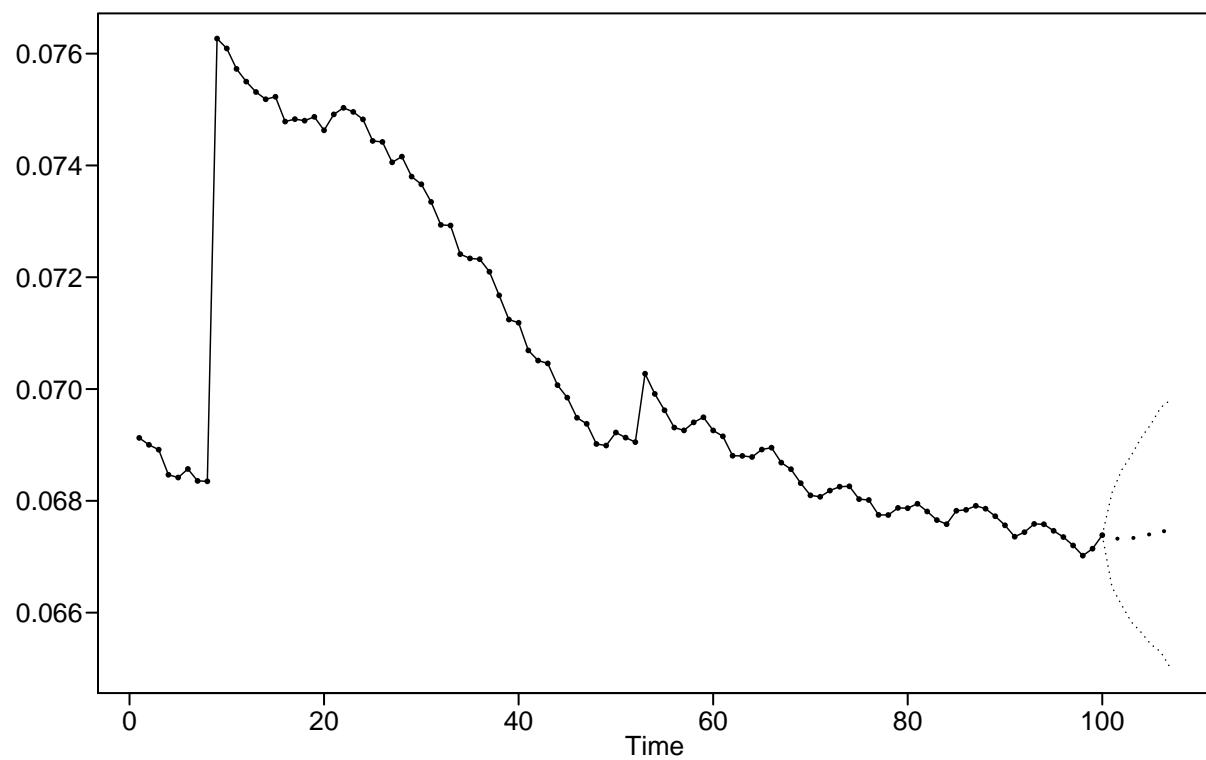


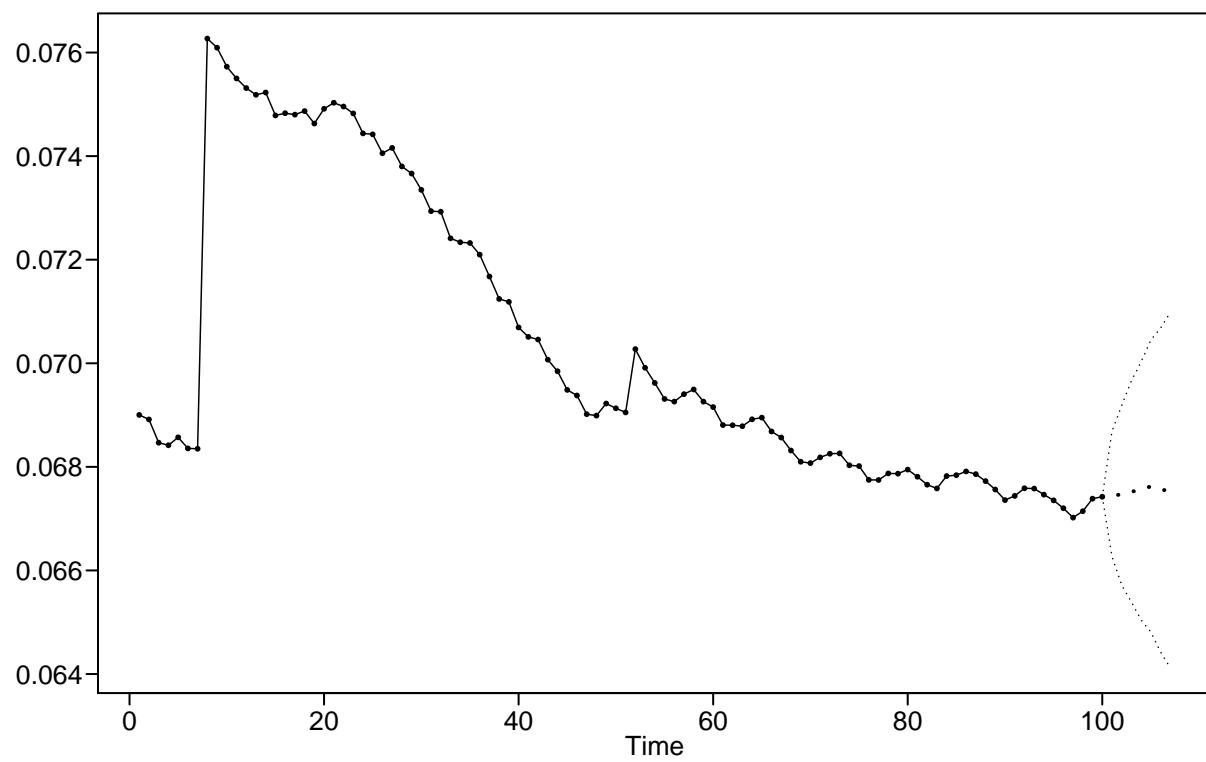


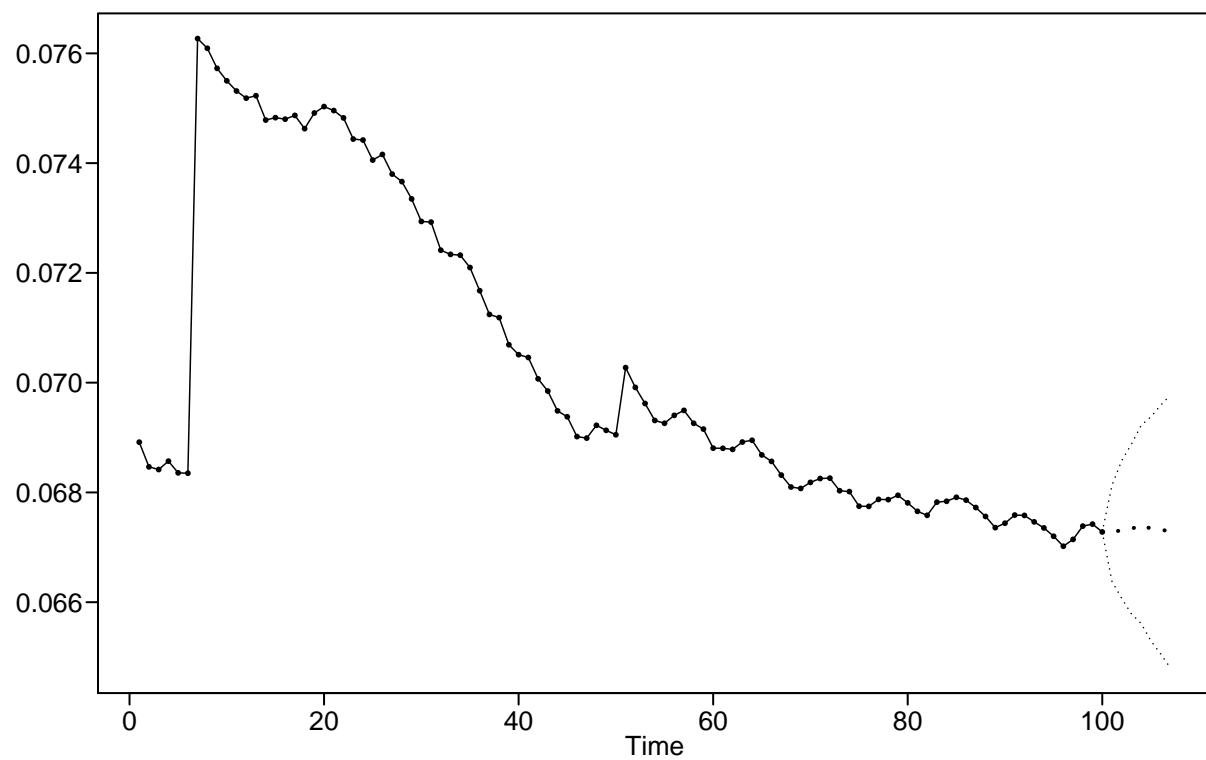


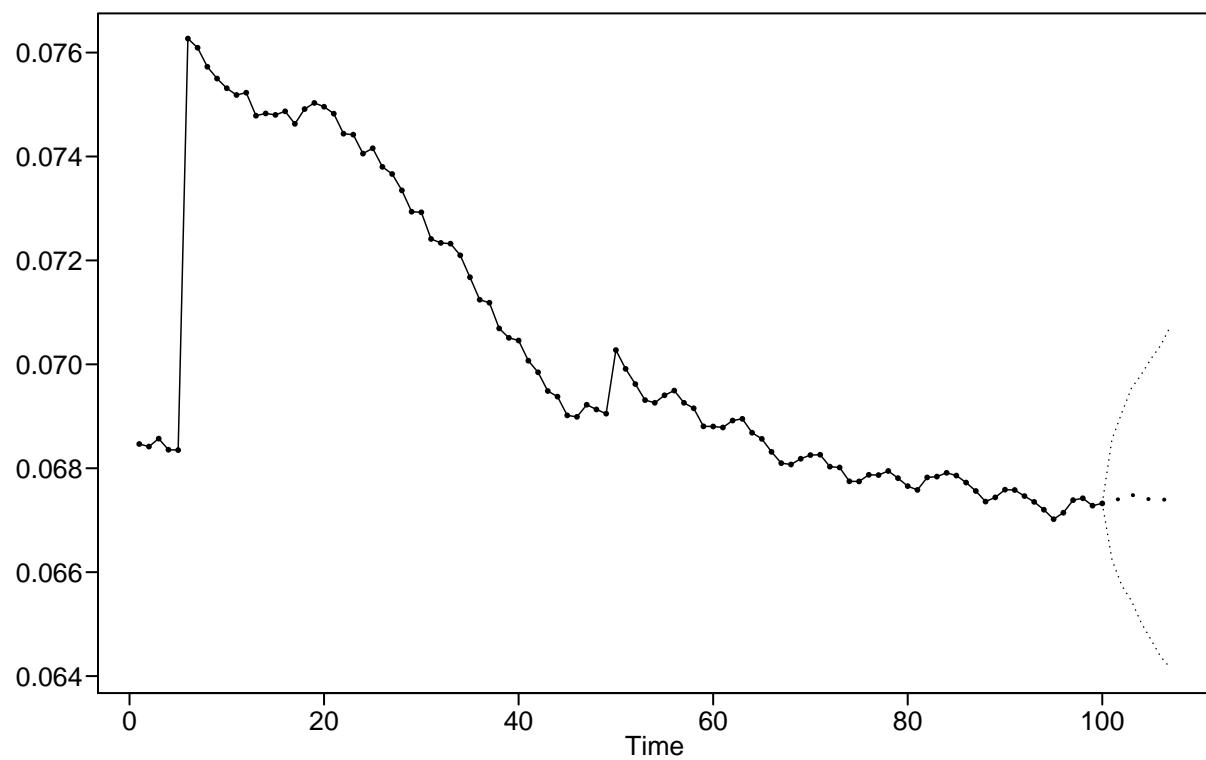


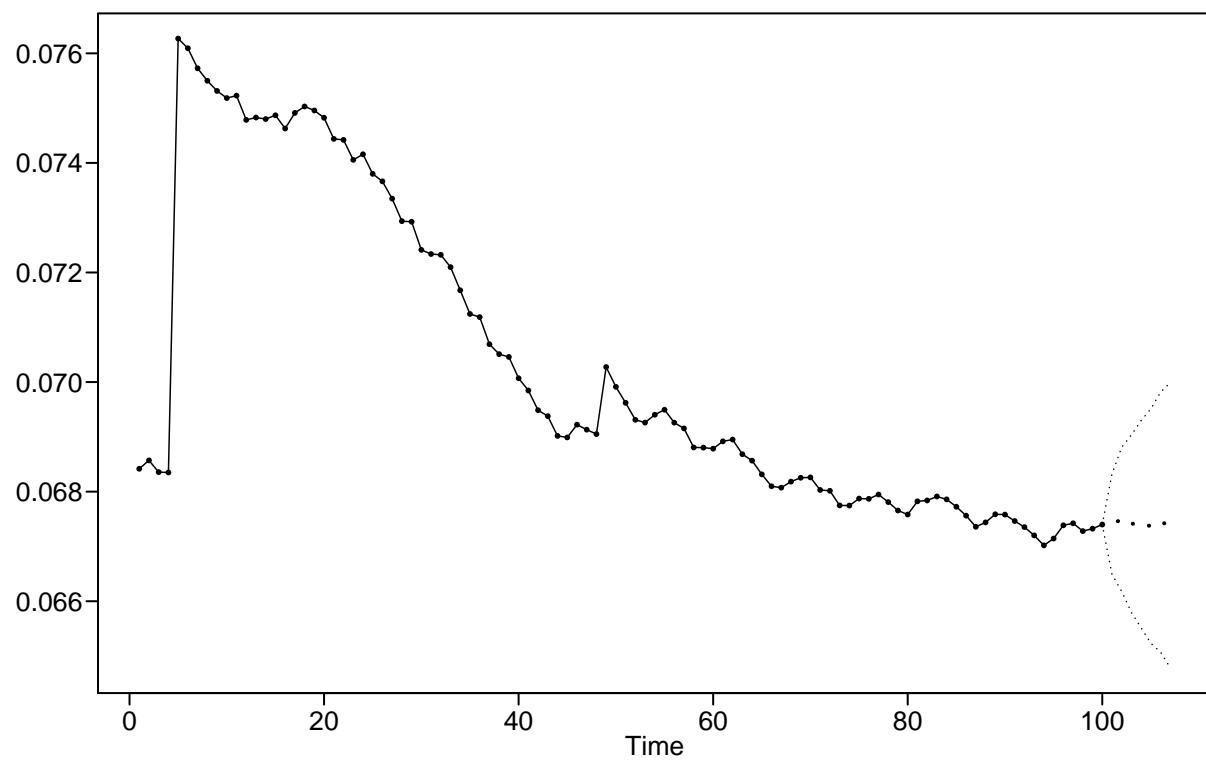


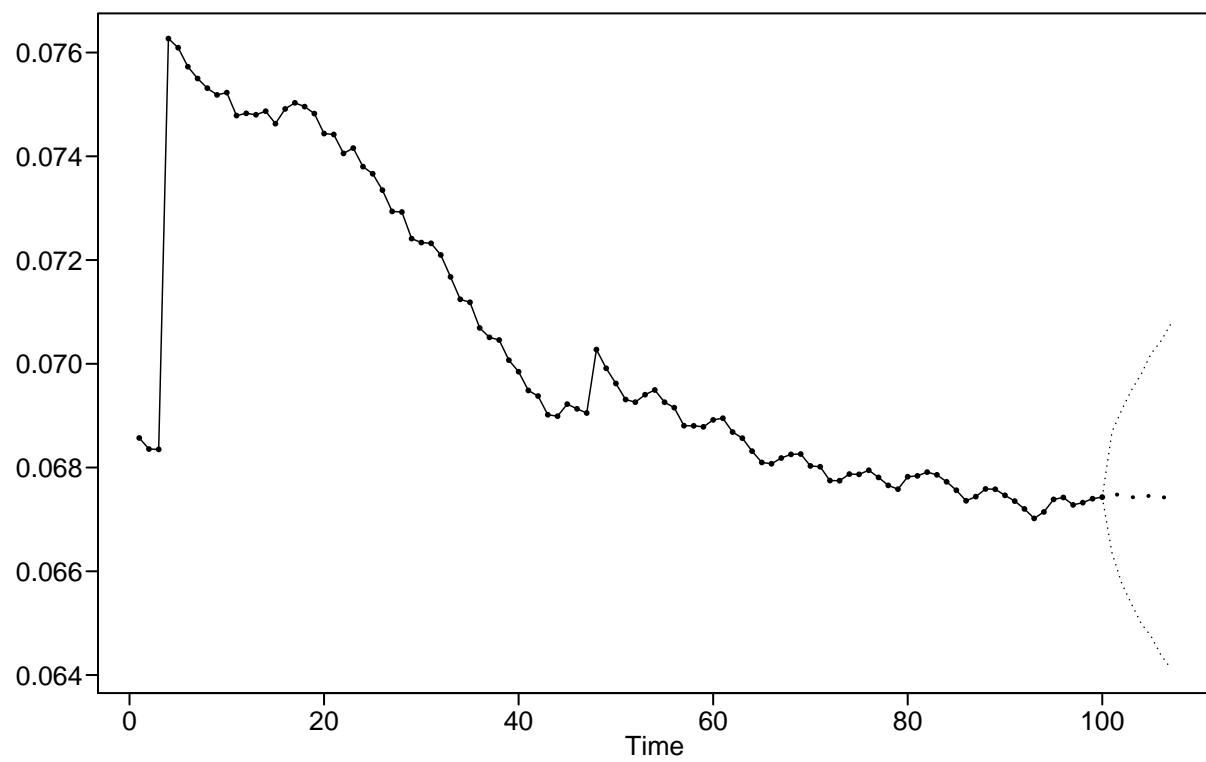


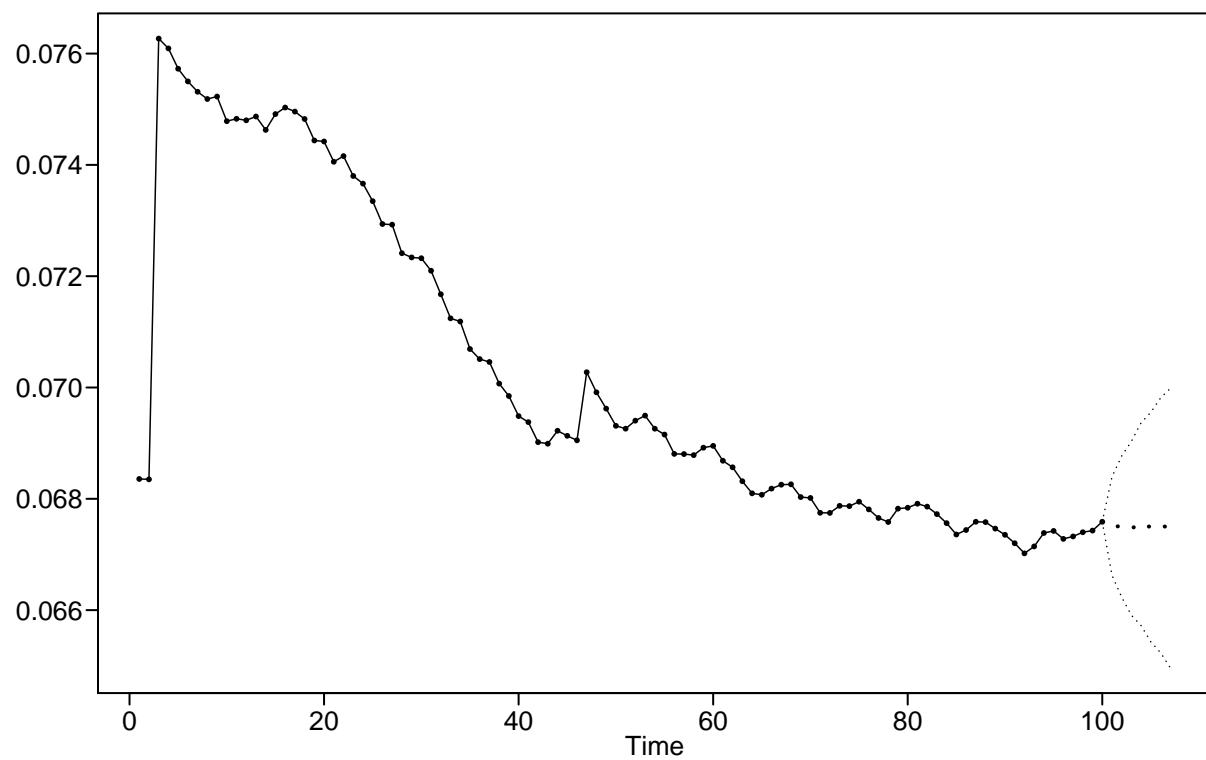


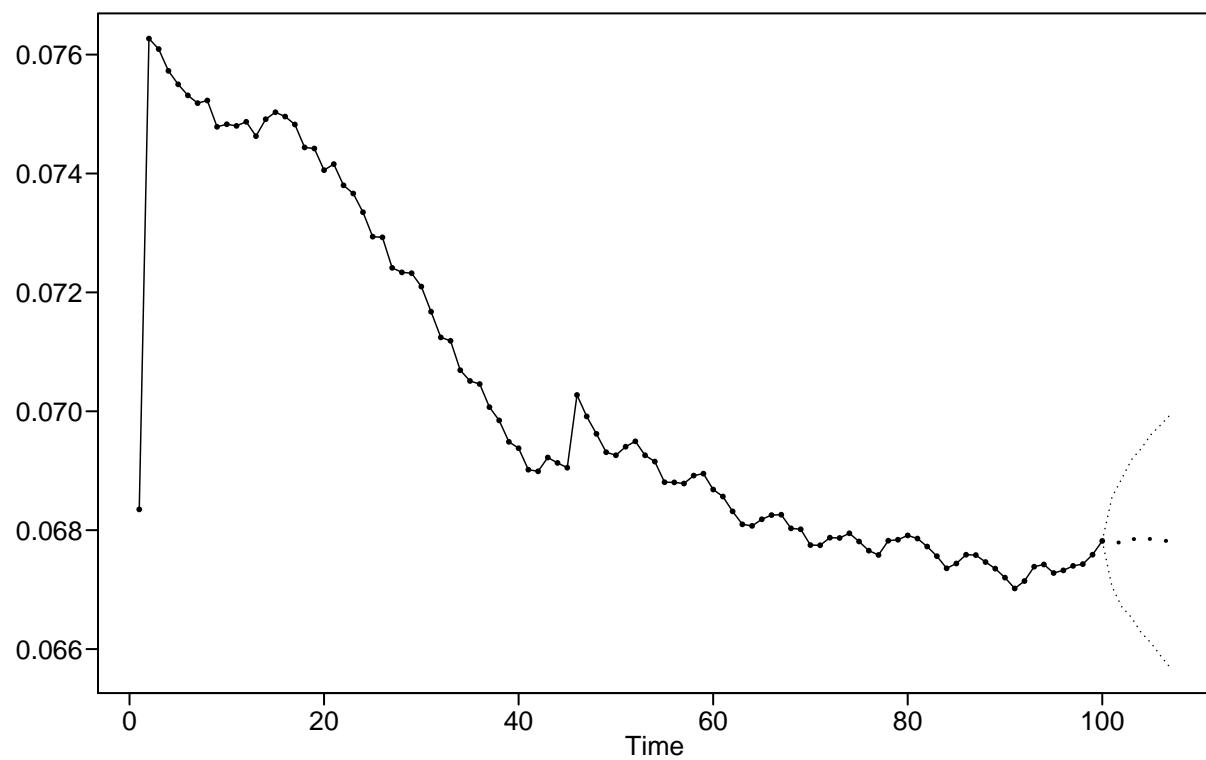






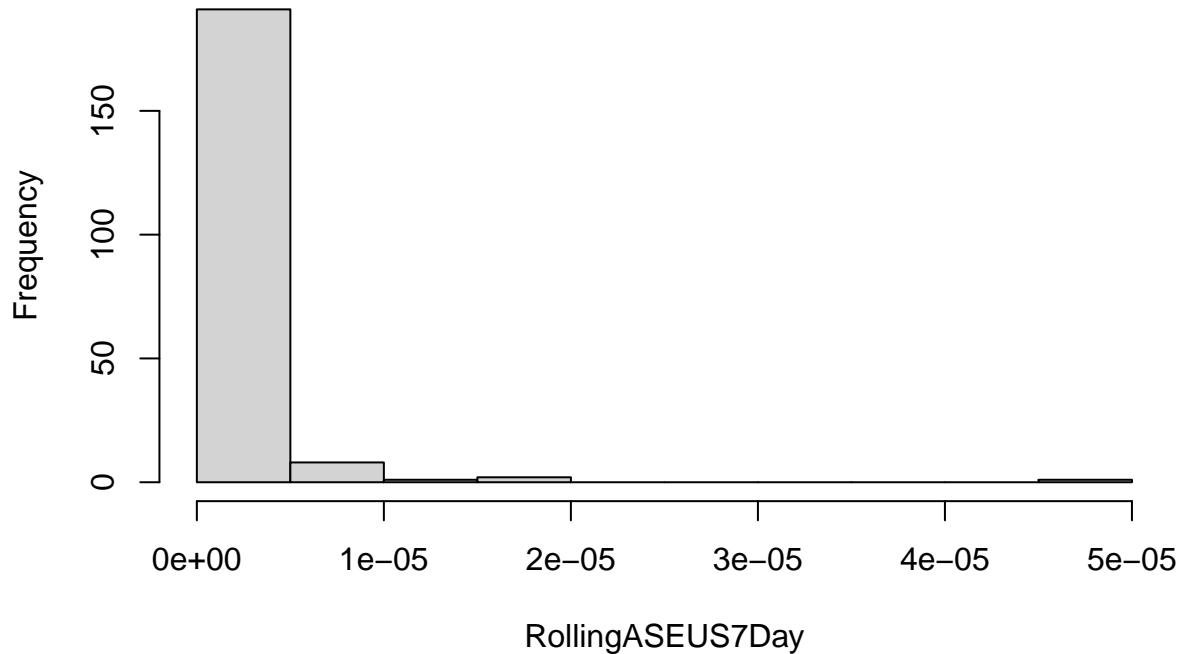






```
modelName = "NC Model - 7 Days"  
GetRollingWindowASEInfo(modelName, RollingASENC7Day)
```

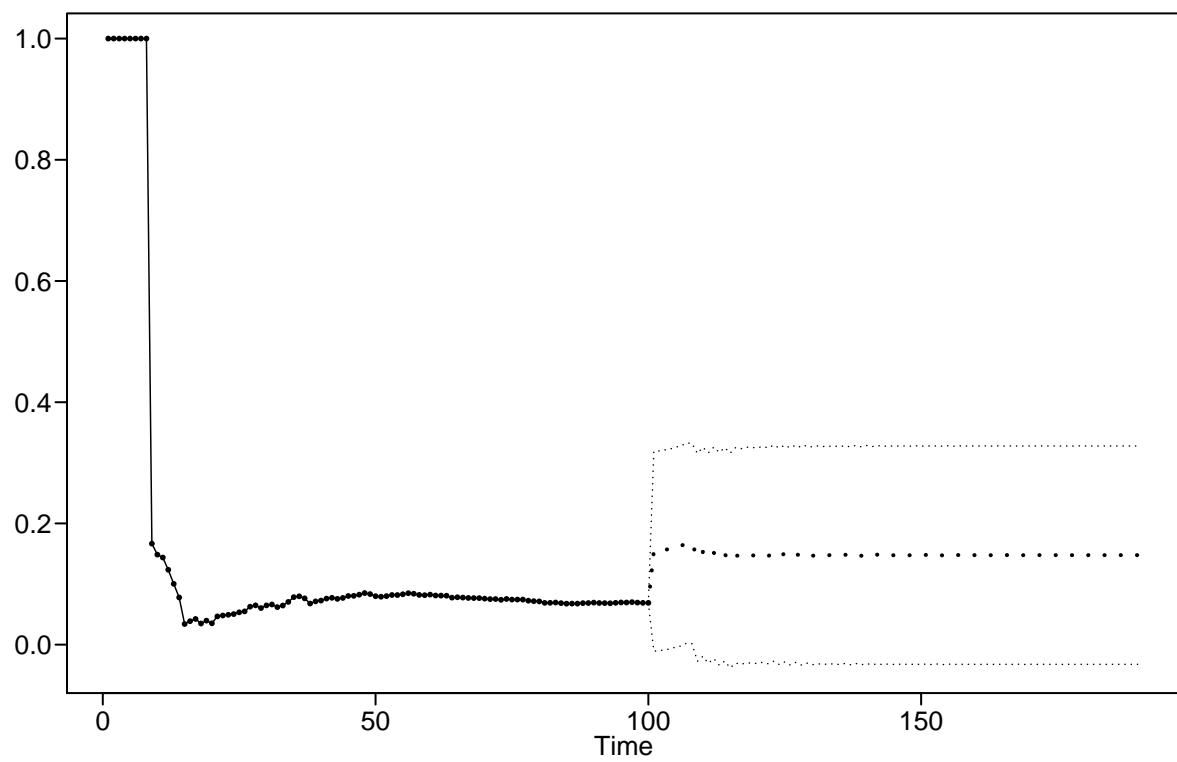
Histogram of NC Model – 7 Days Windowed ASE

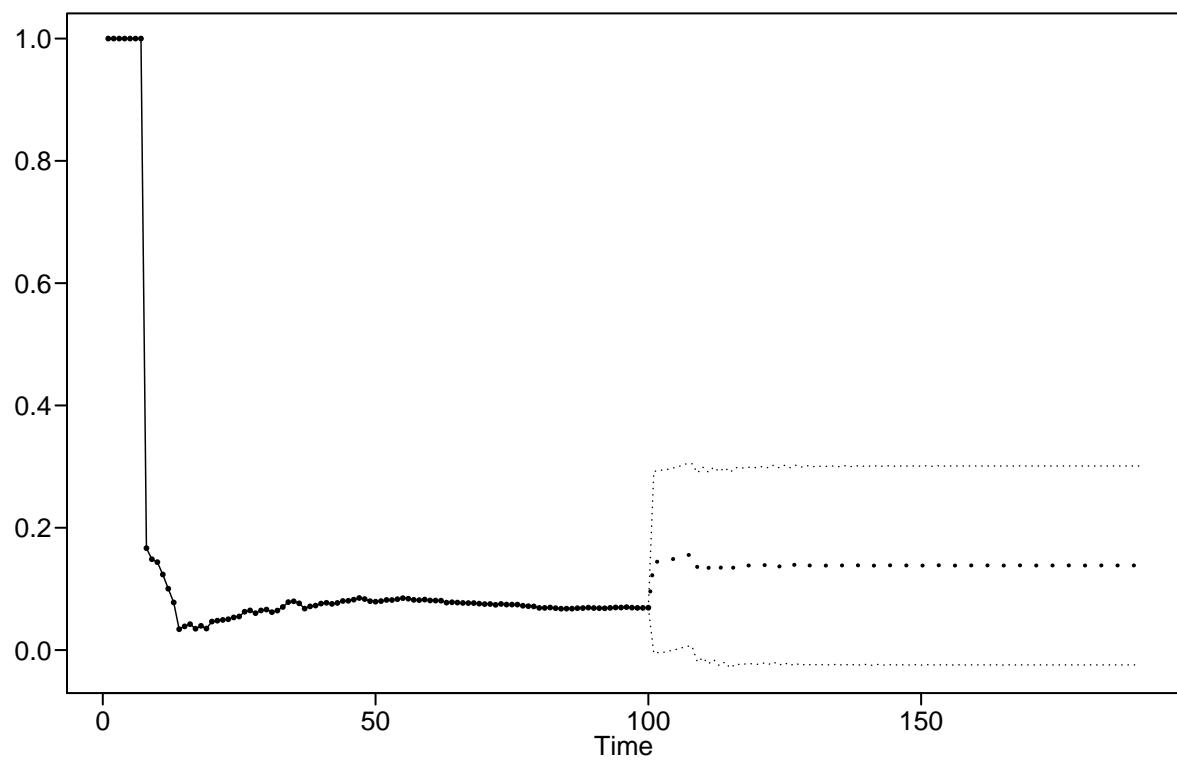


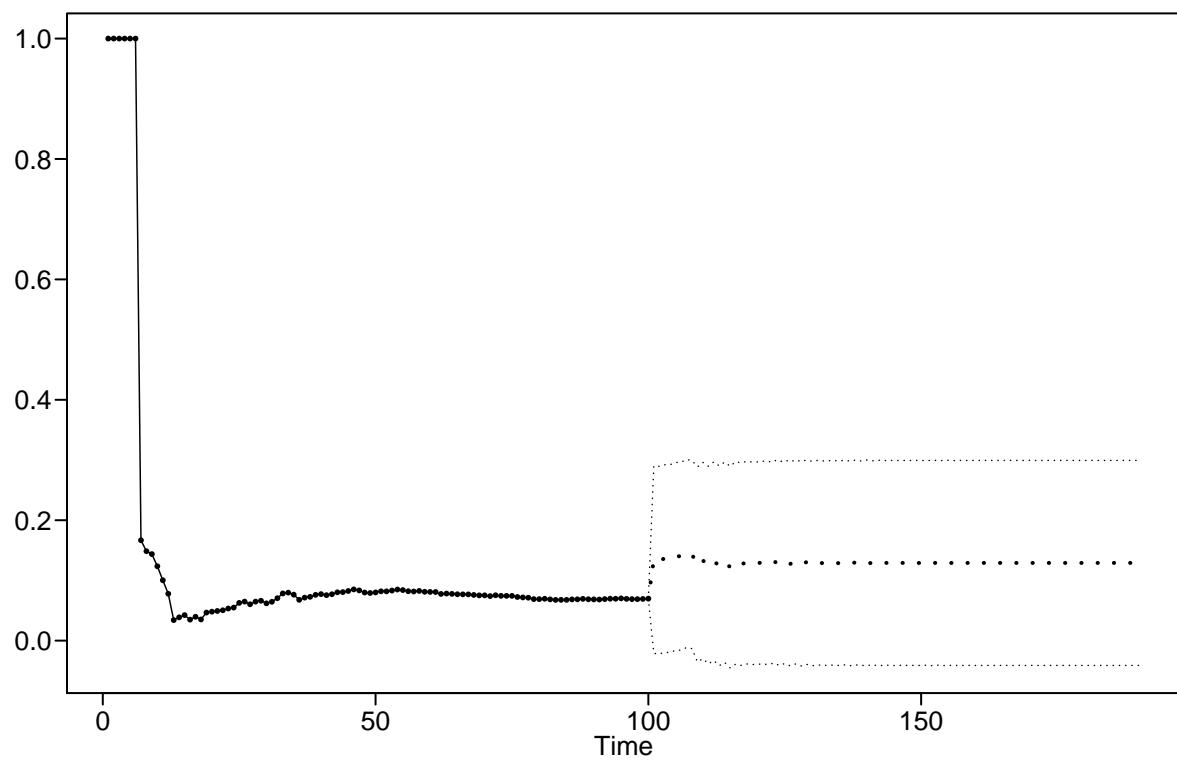
```
## [1] "NC Model - 7 Days Windowed ASE Mean: 1.95119012725954e-06"  
## [1] "NC Model - 7 Days Windowed ASE Median: 3.49974835297101e-07"
```

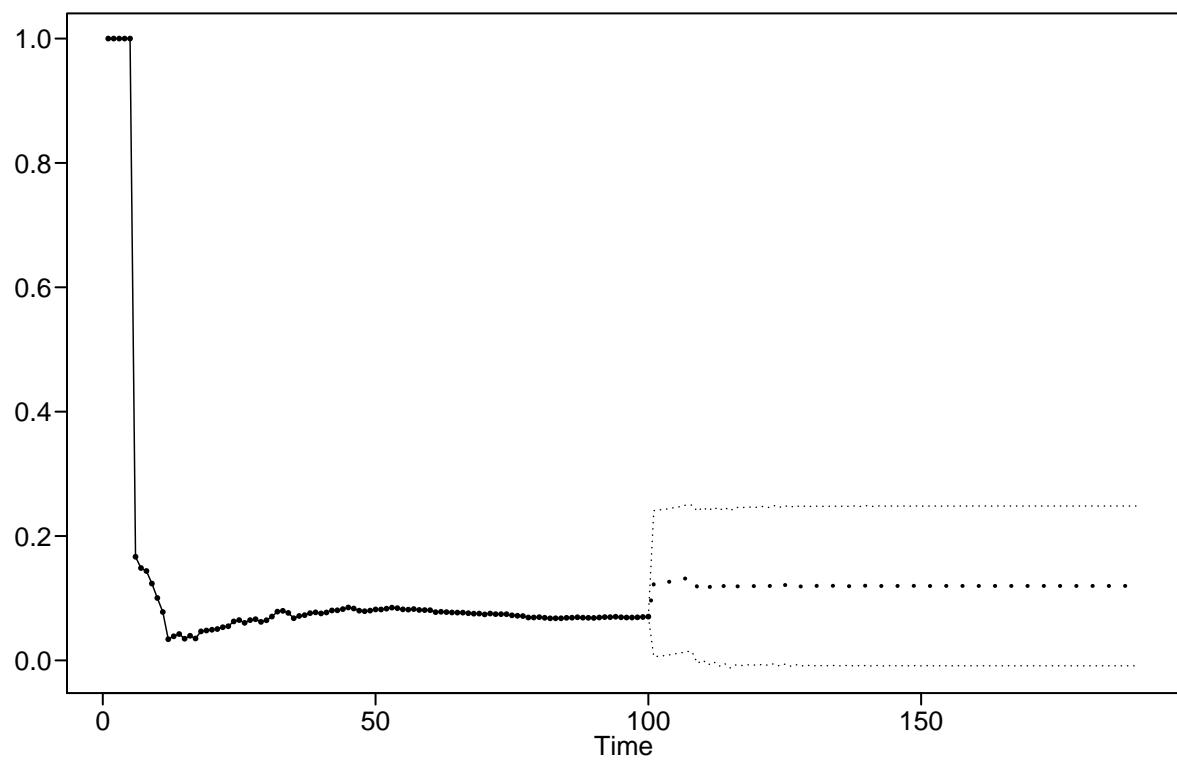
```
#Rolling Window ASE Calc - 90 Days
```

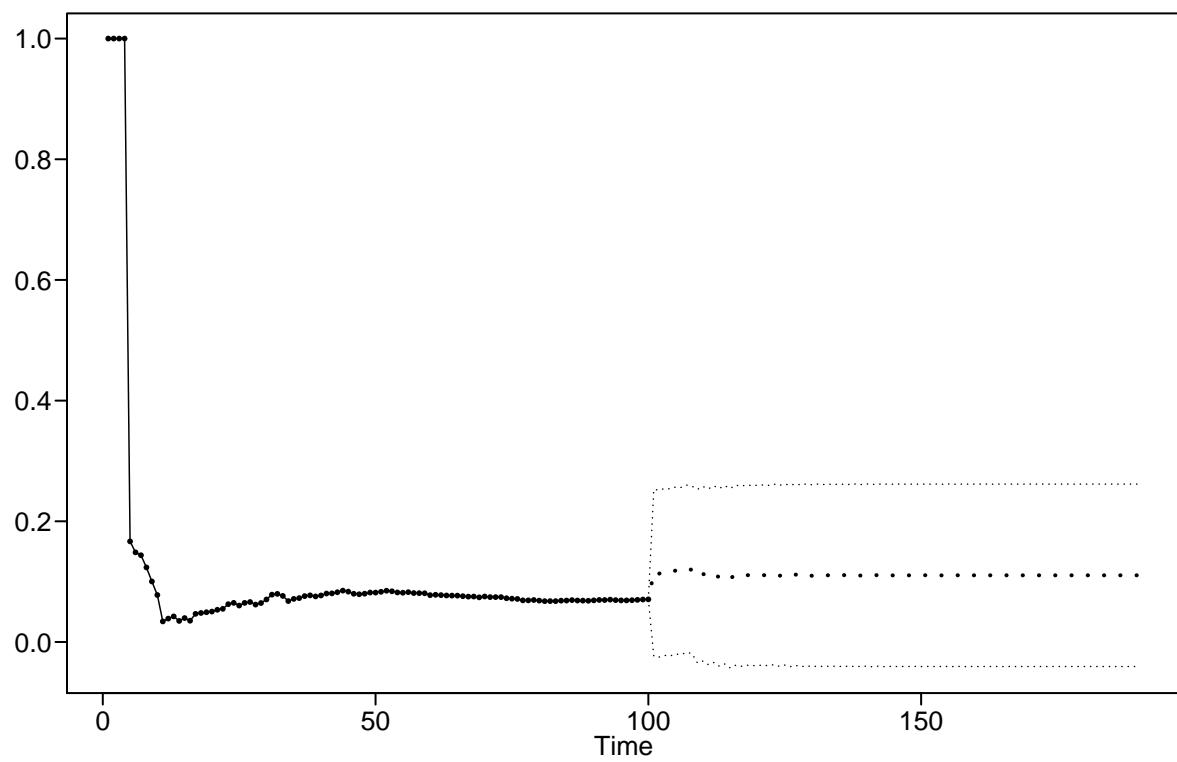
```
RollingASENC90Day = RollingASECalc(df_NC_Final$Percent_Positive, phis=NC.est$phi, thetas = NC.est$theta,
```

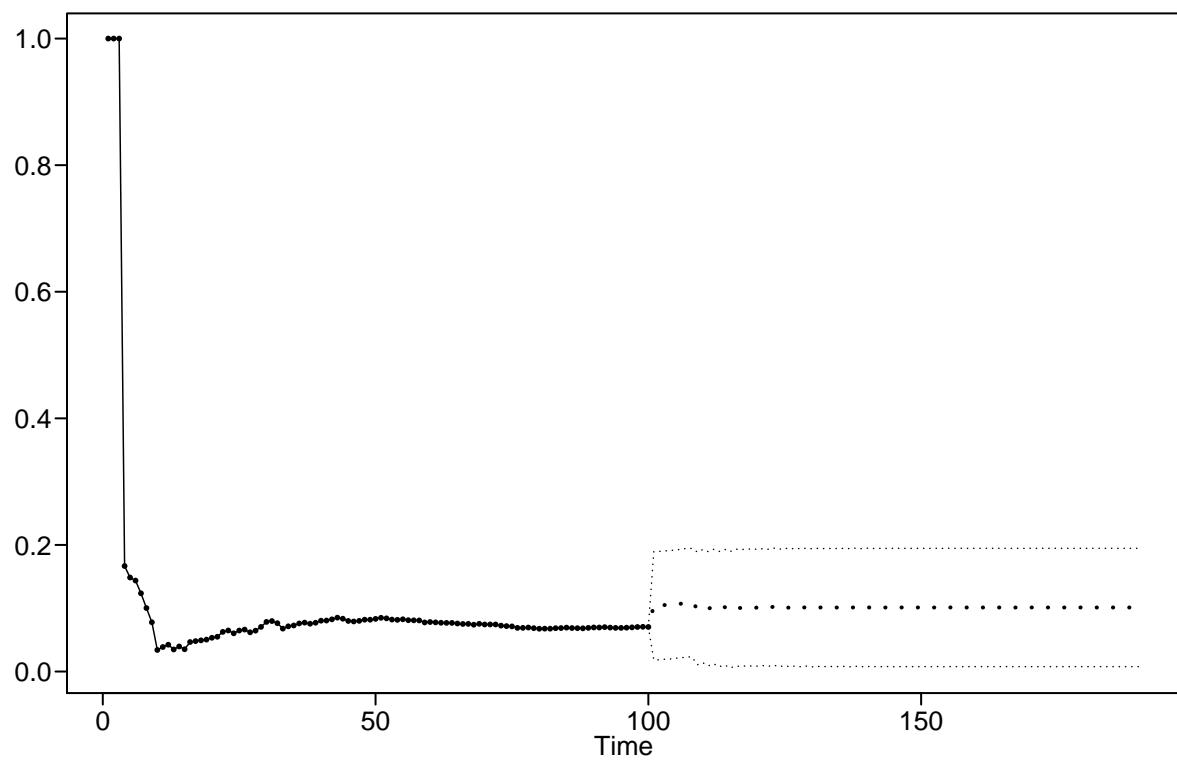


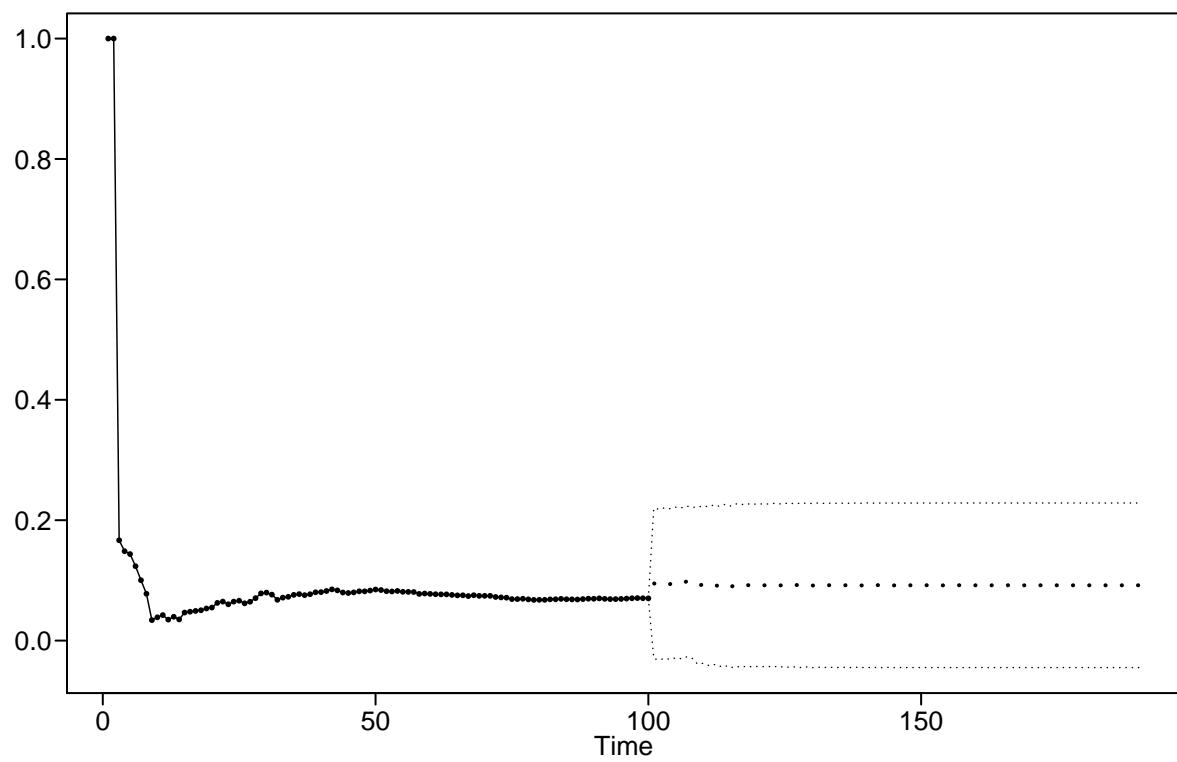


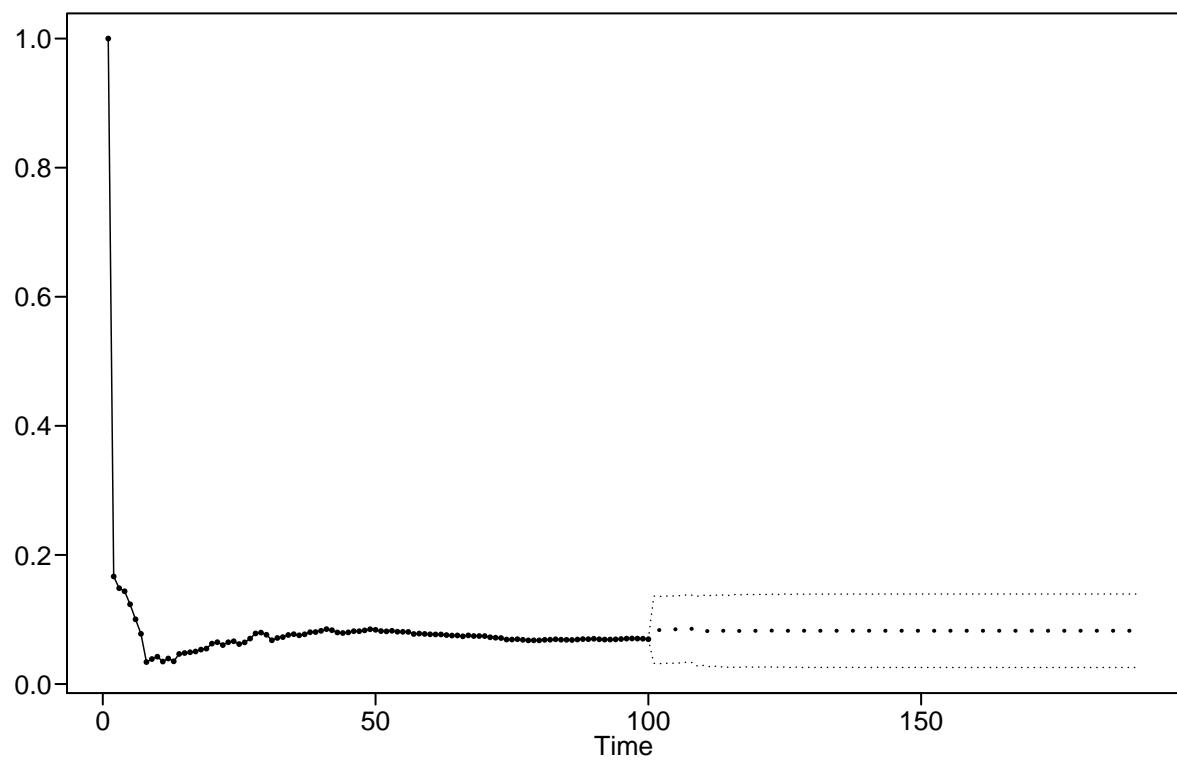


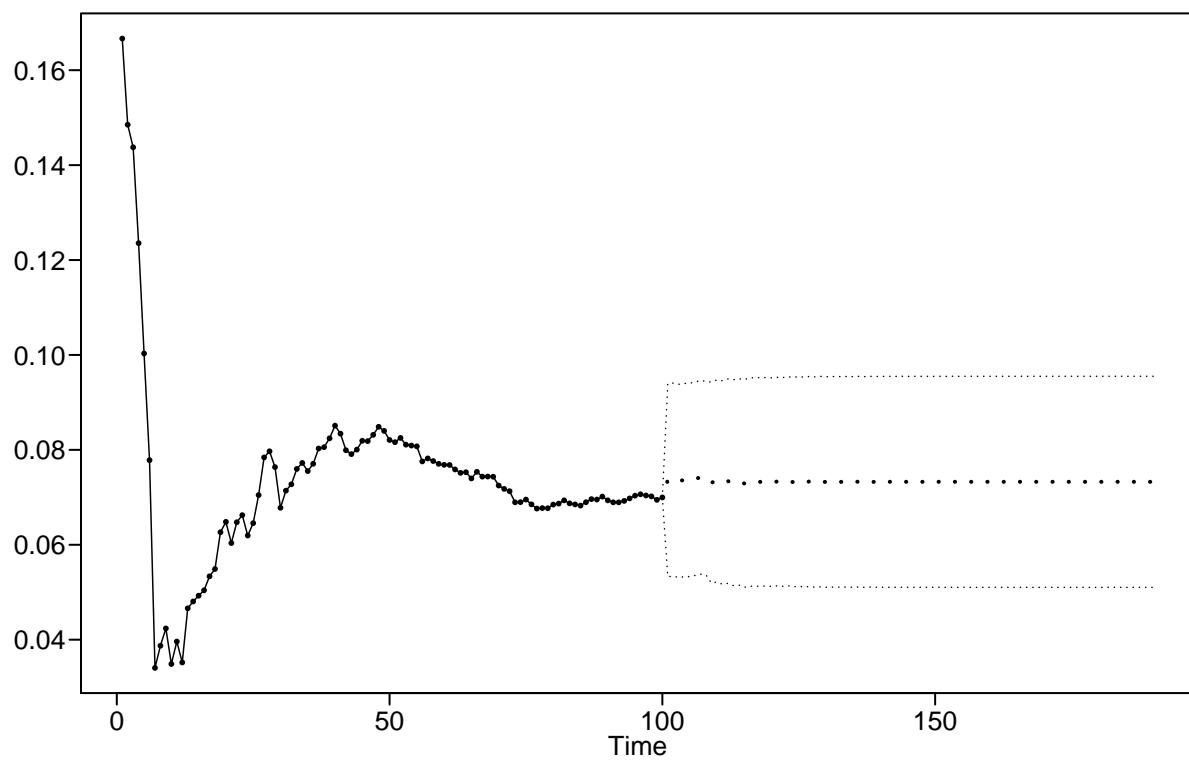


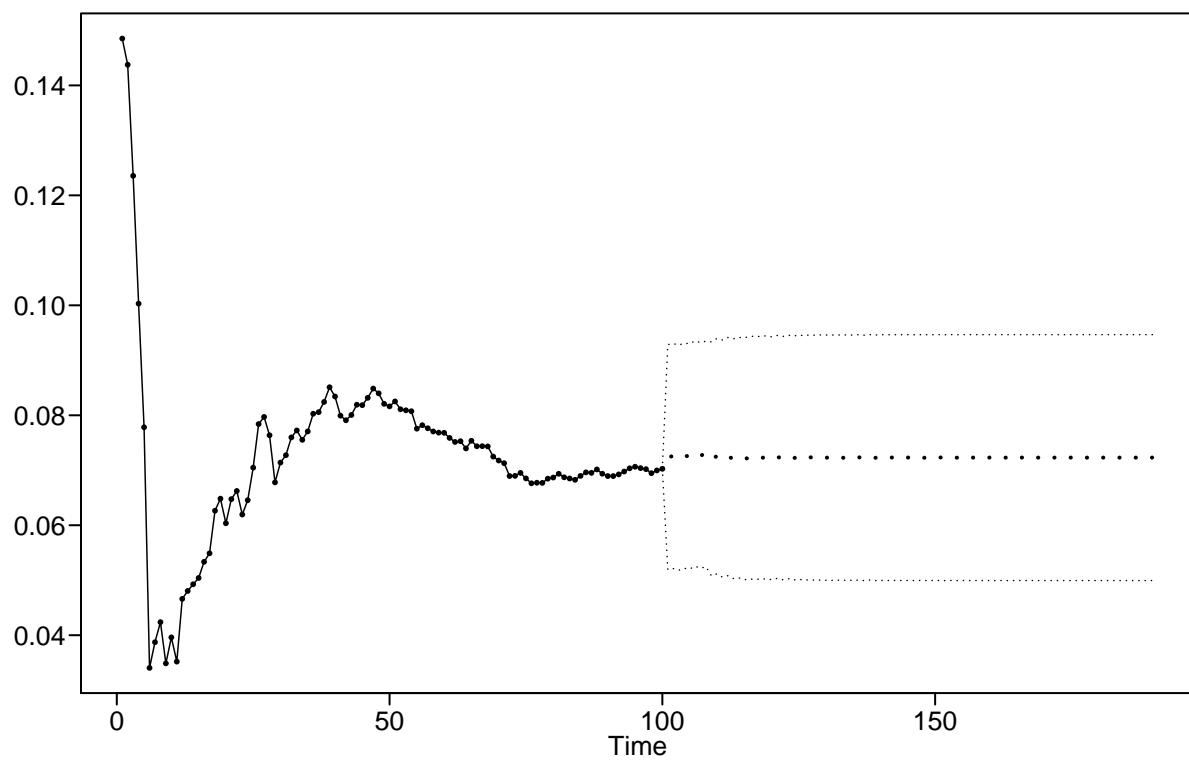


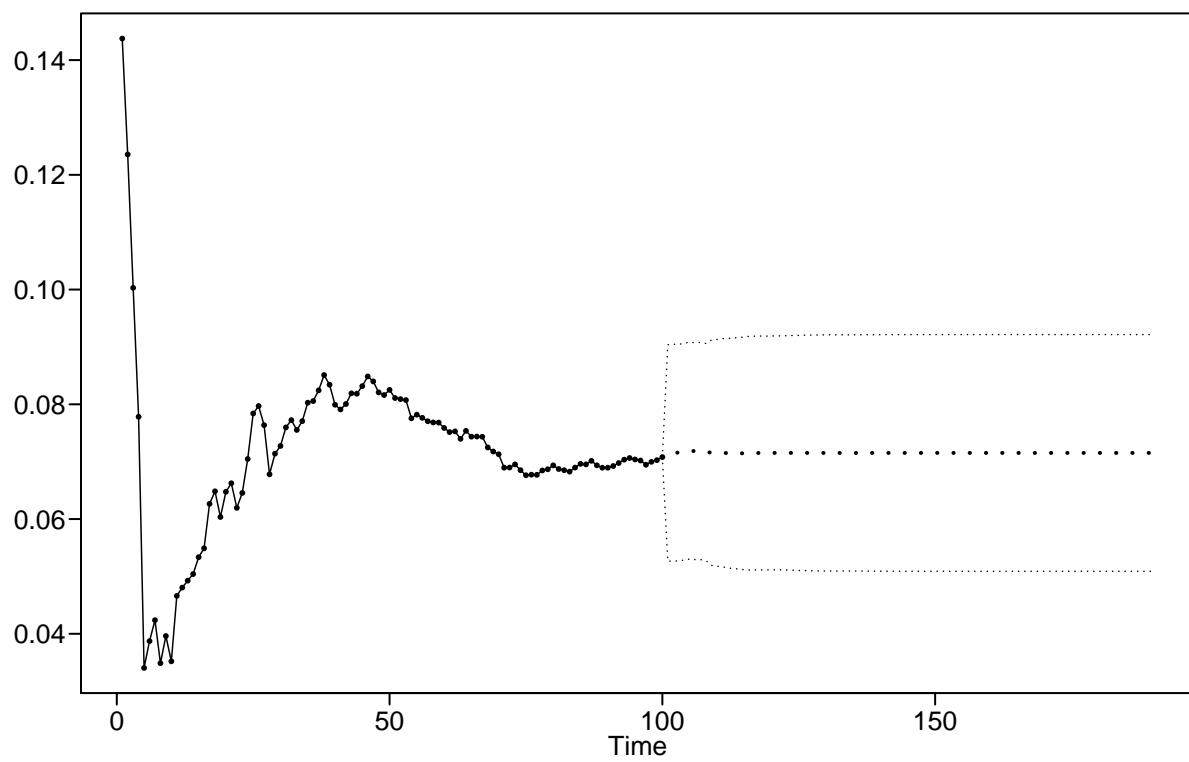


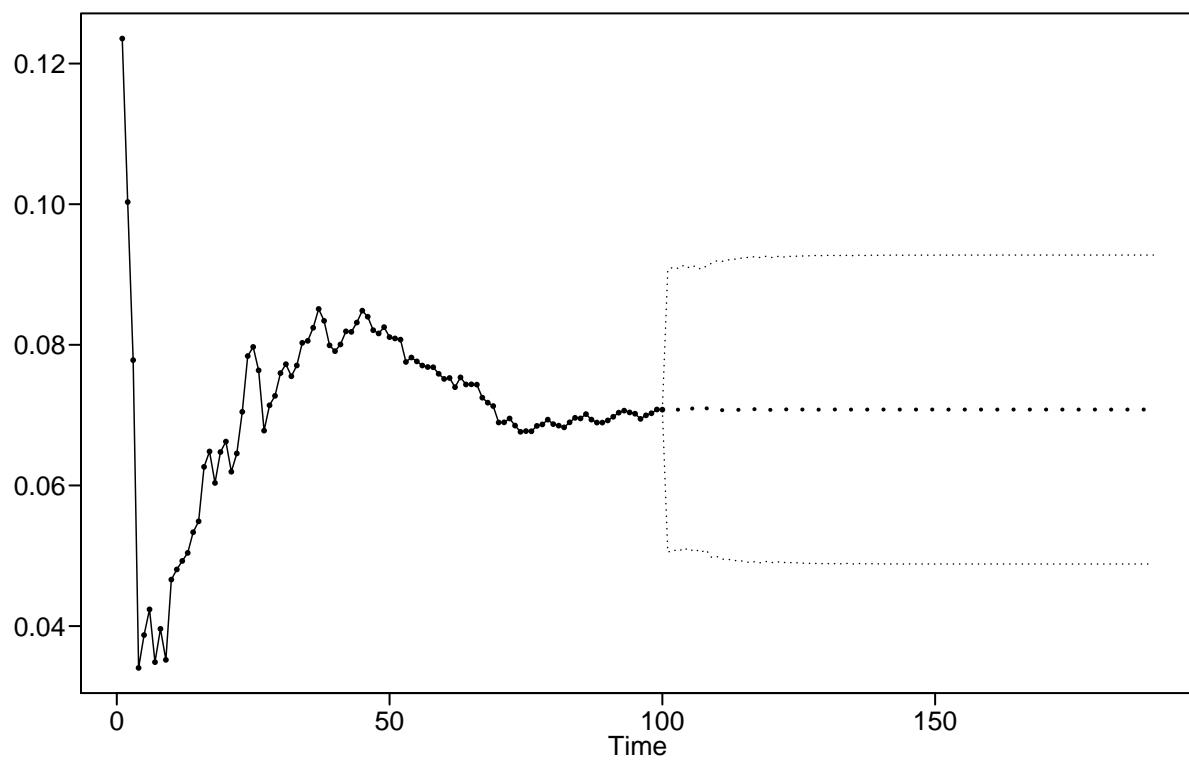


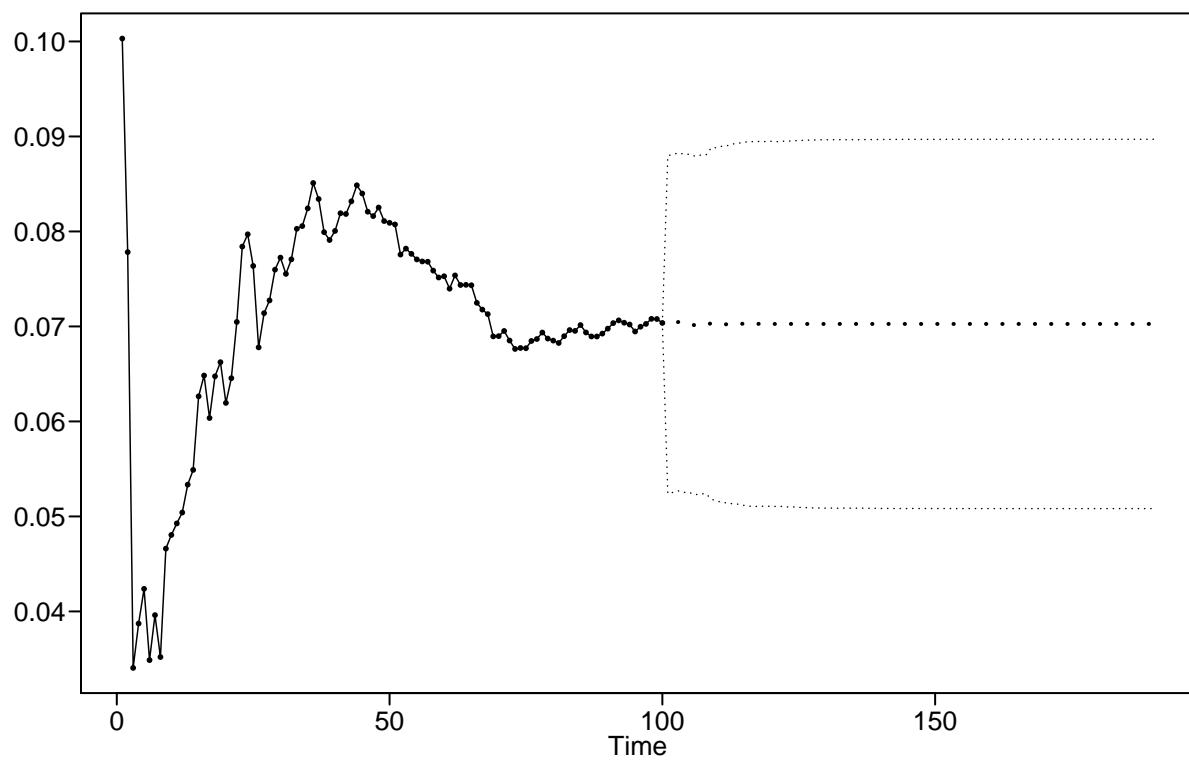


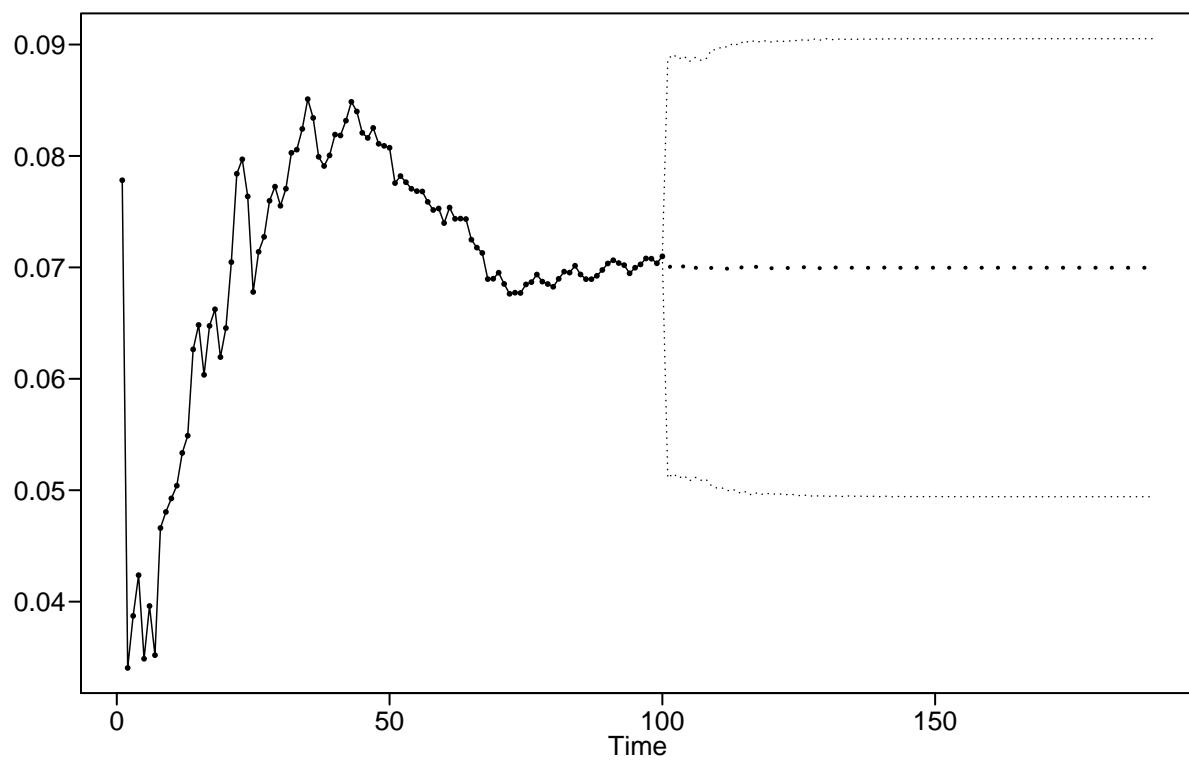


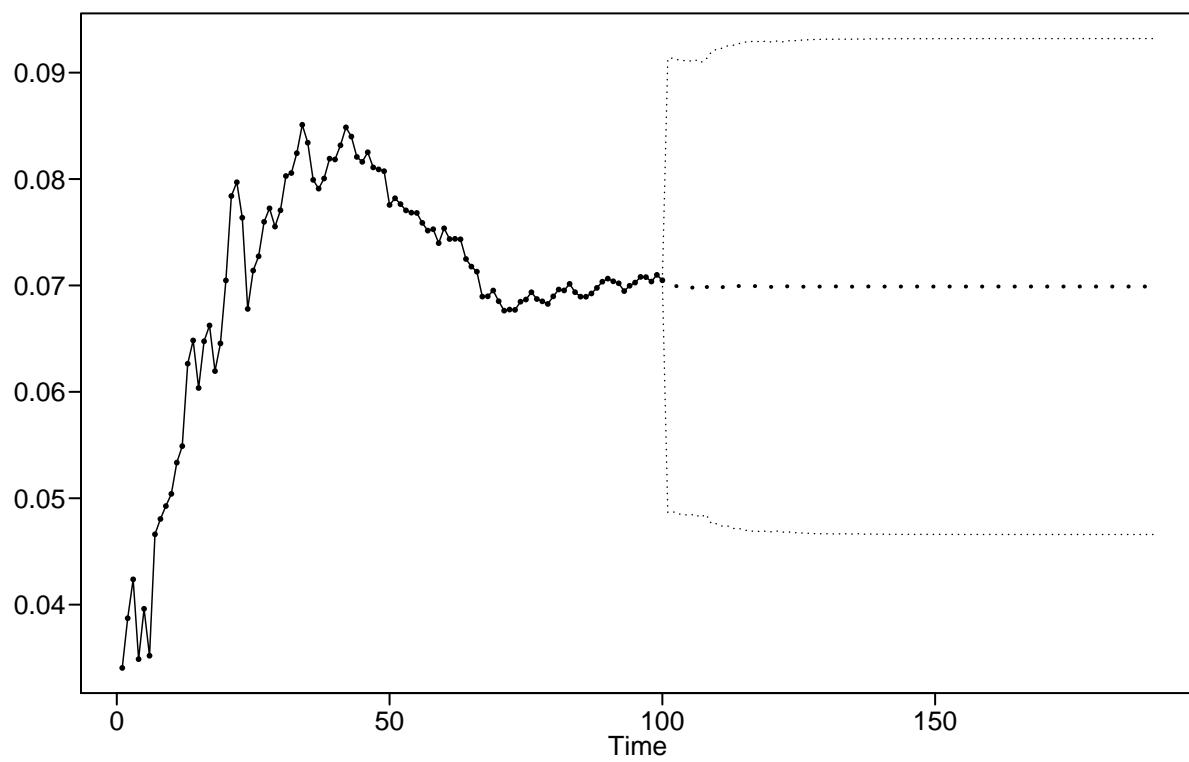


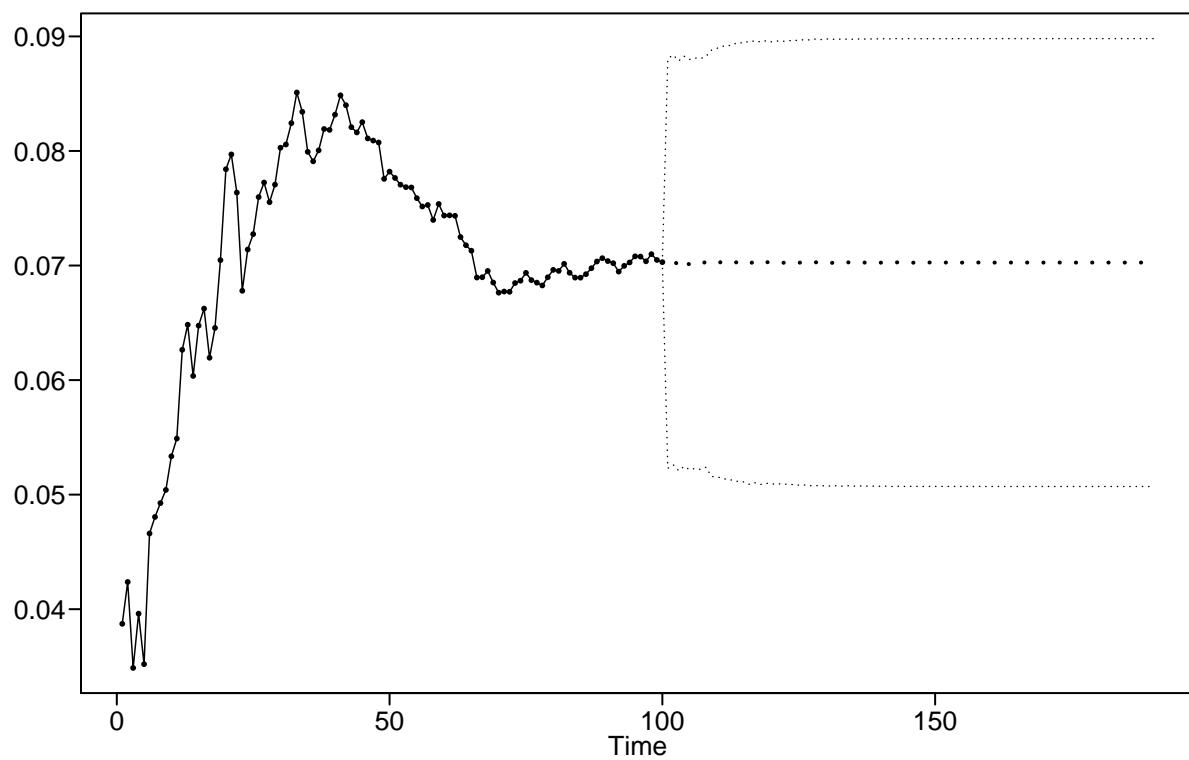


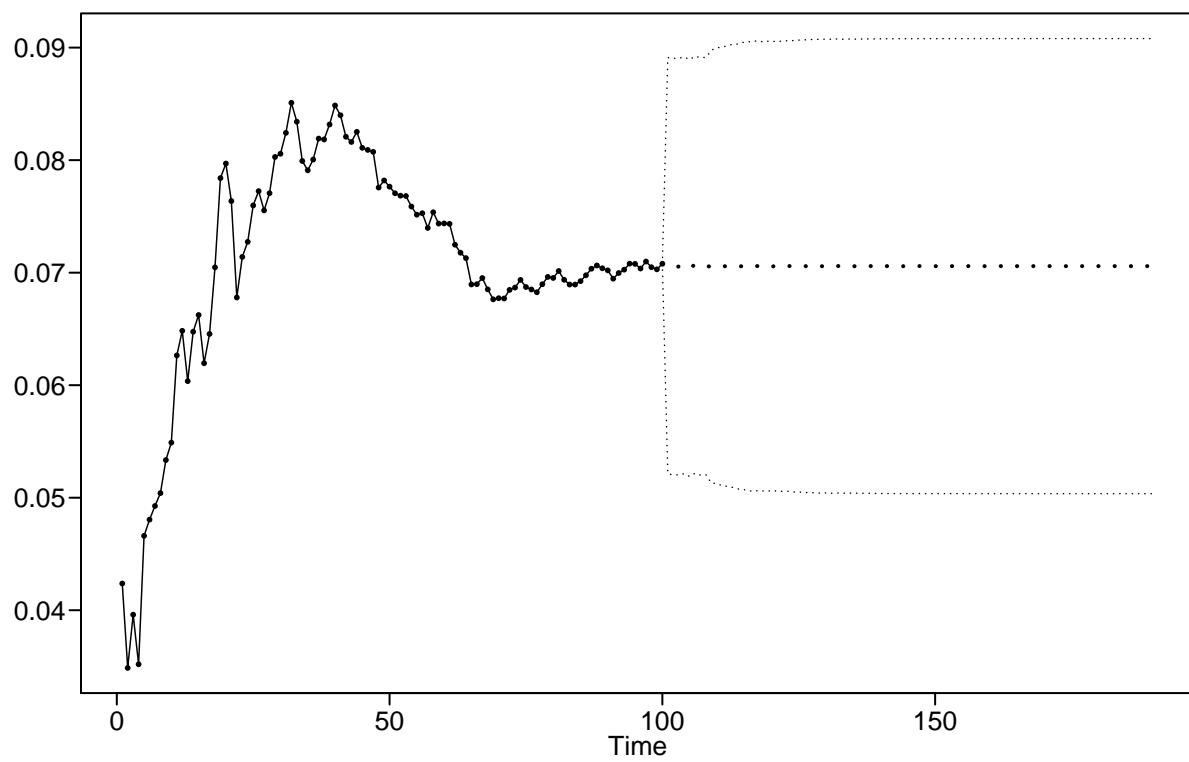


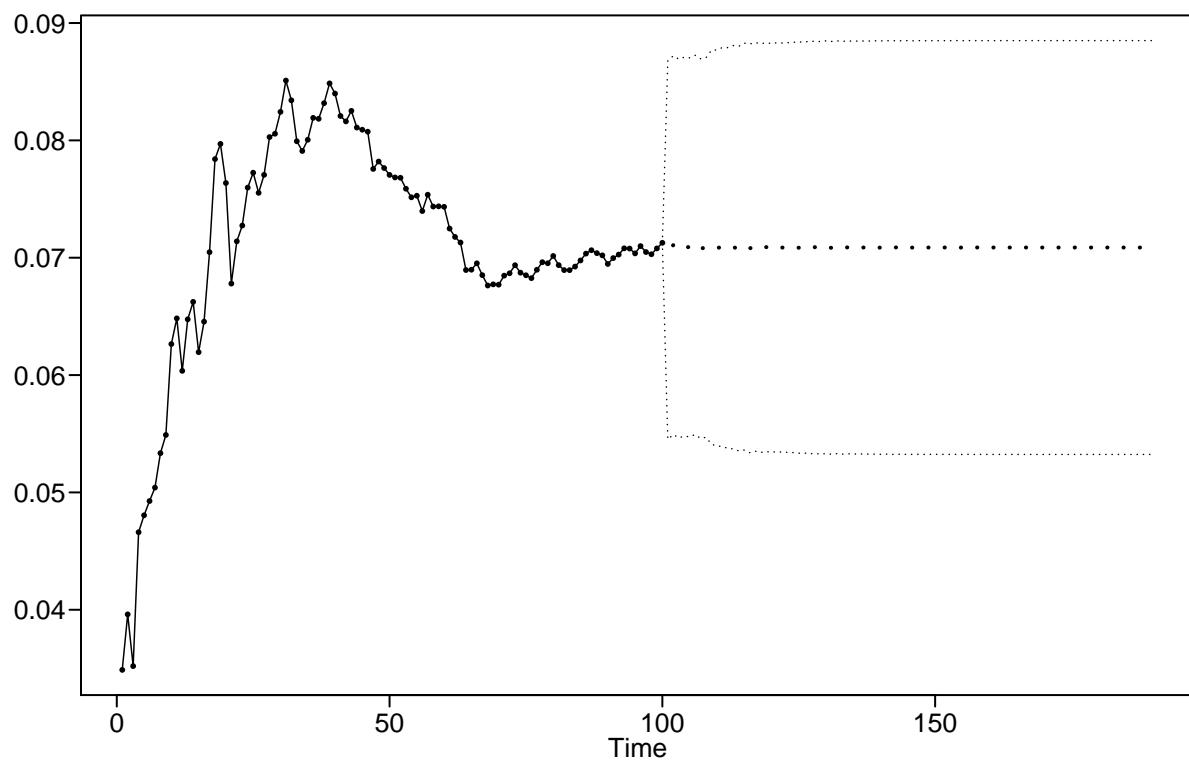


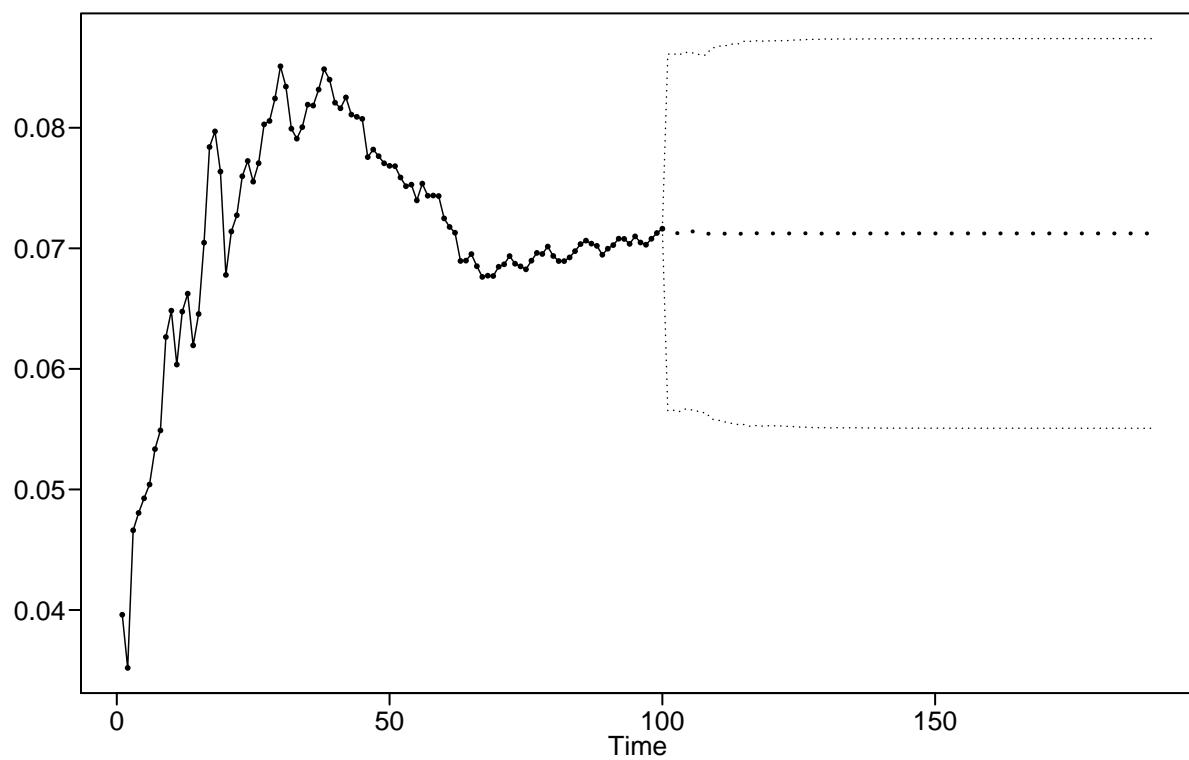


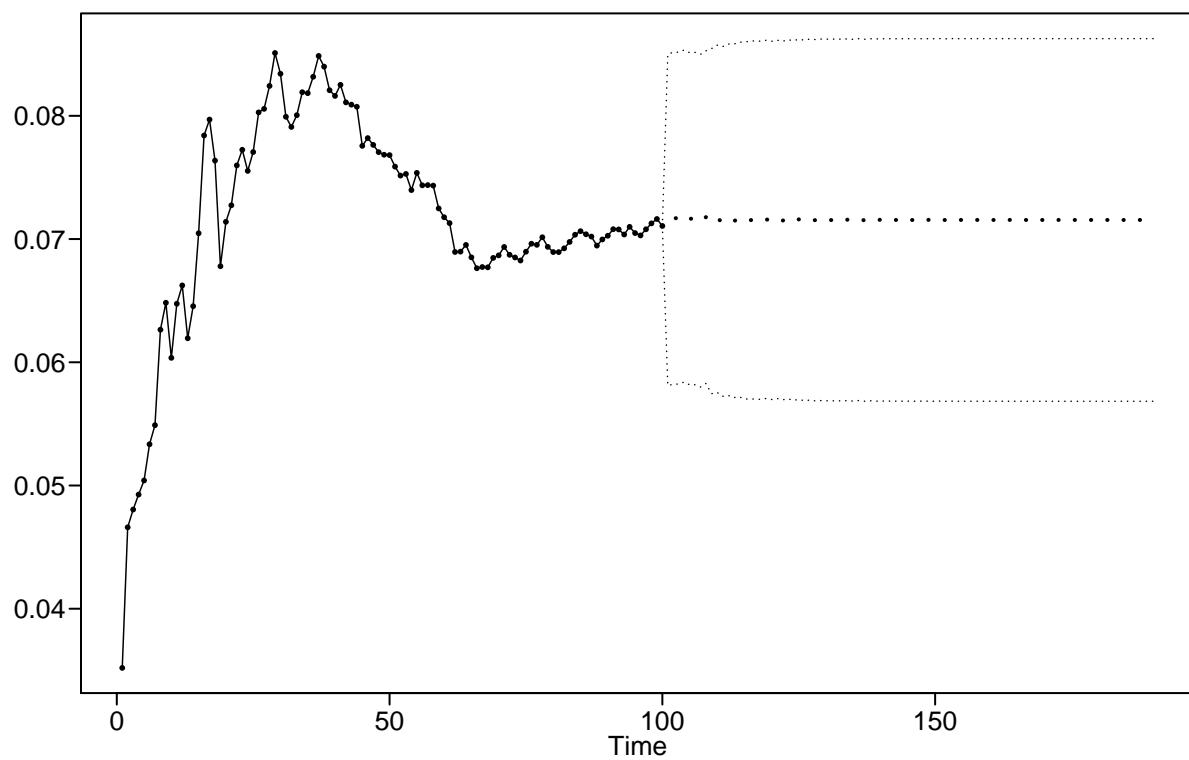


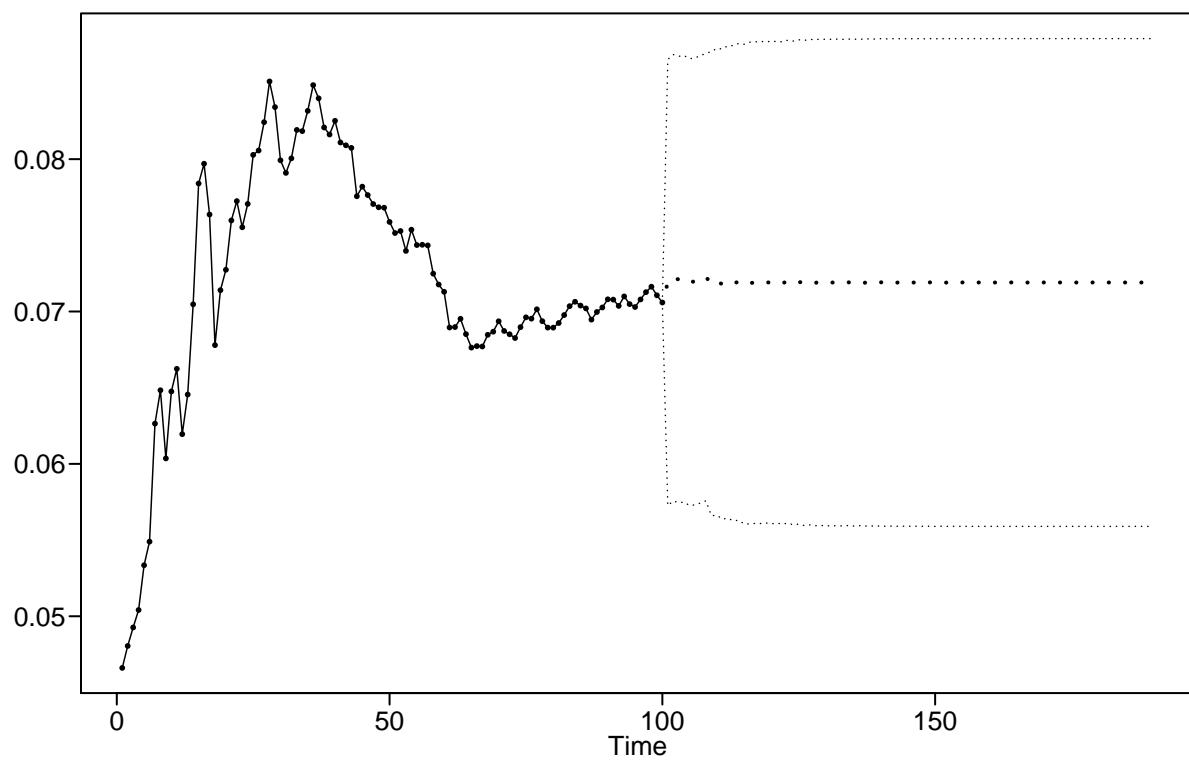


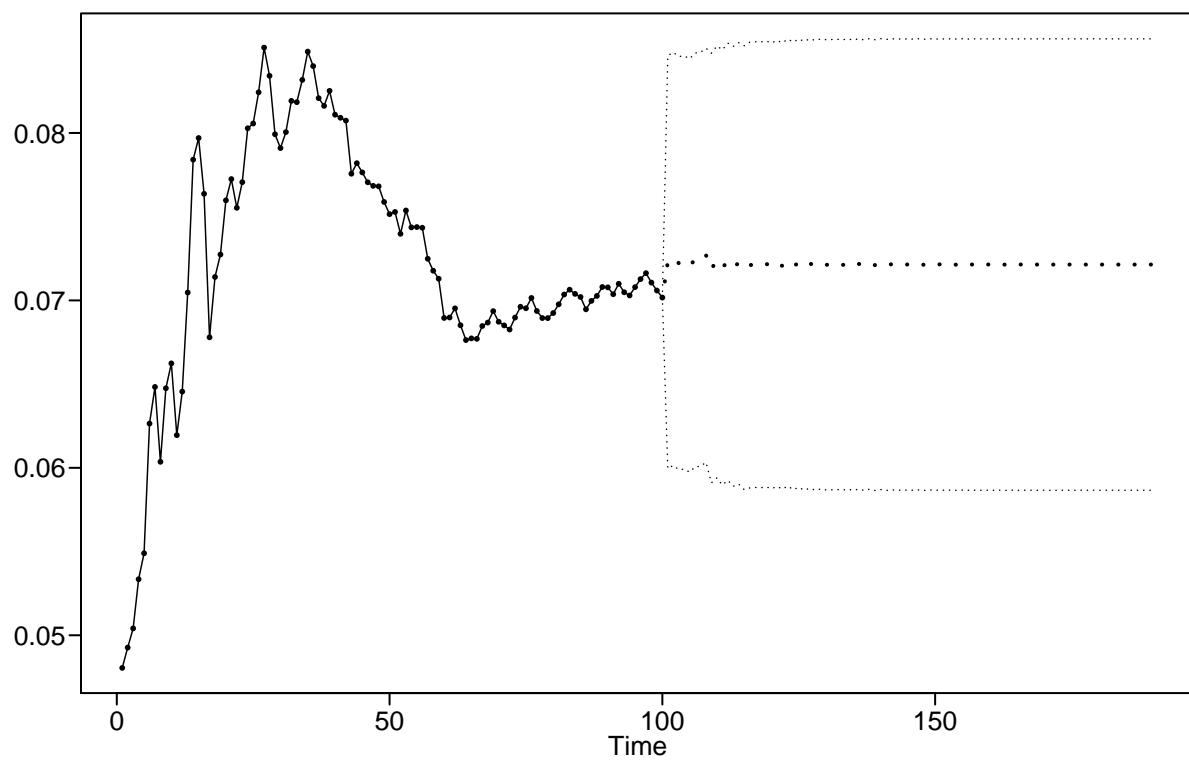


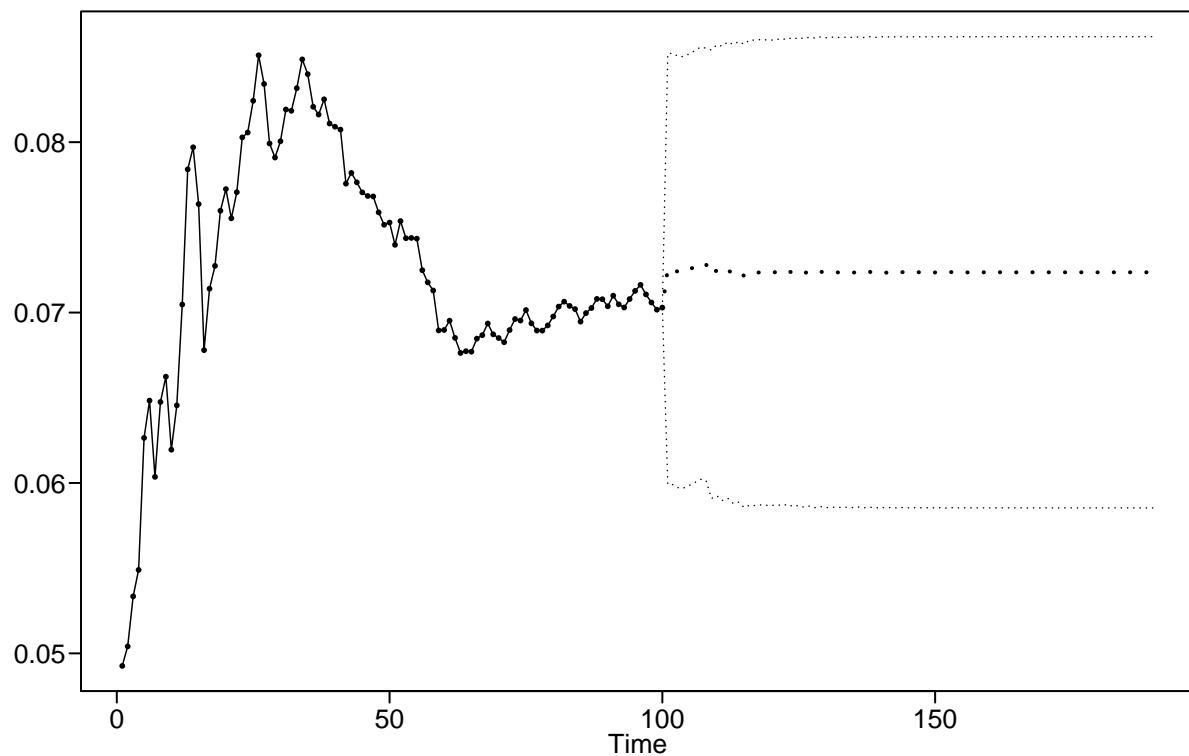


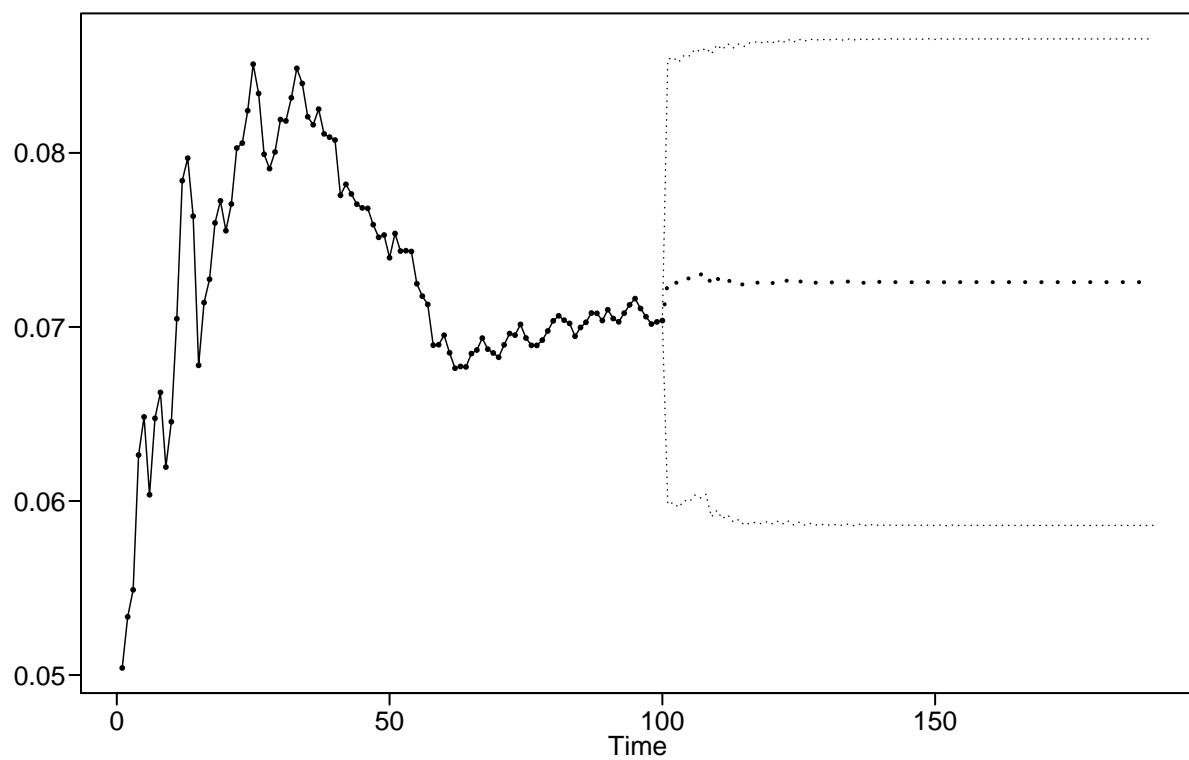


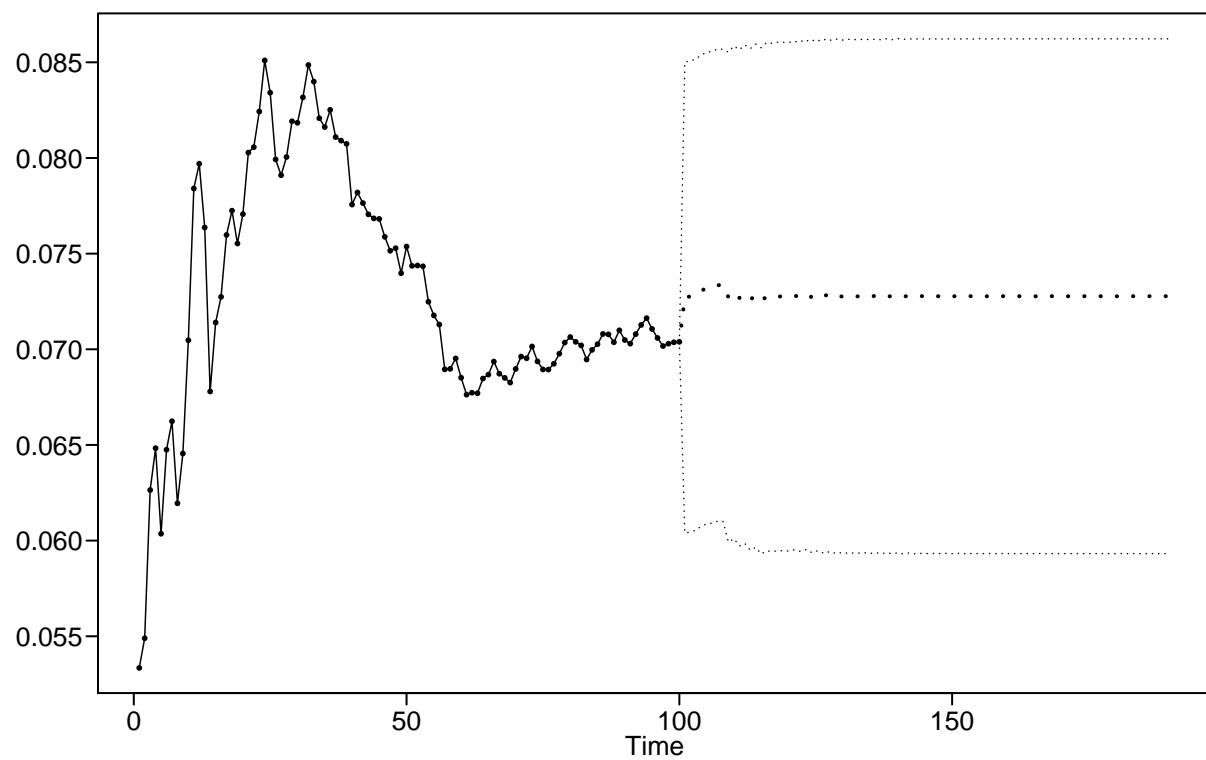


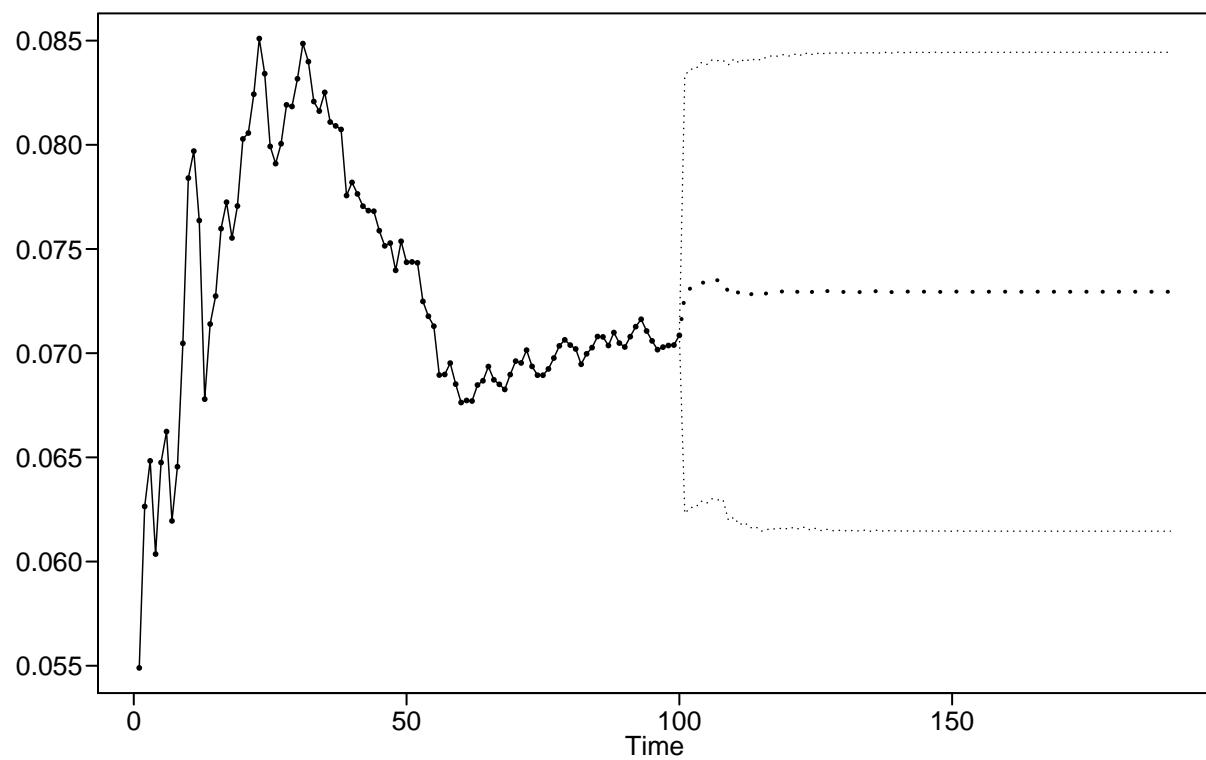


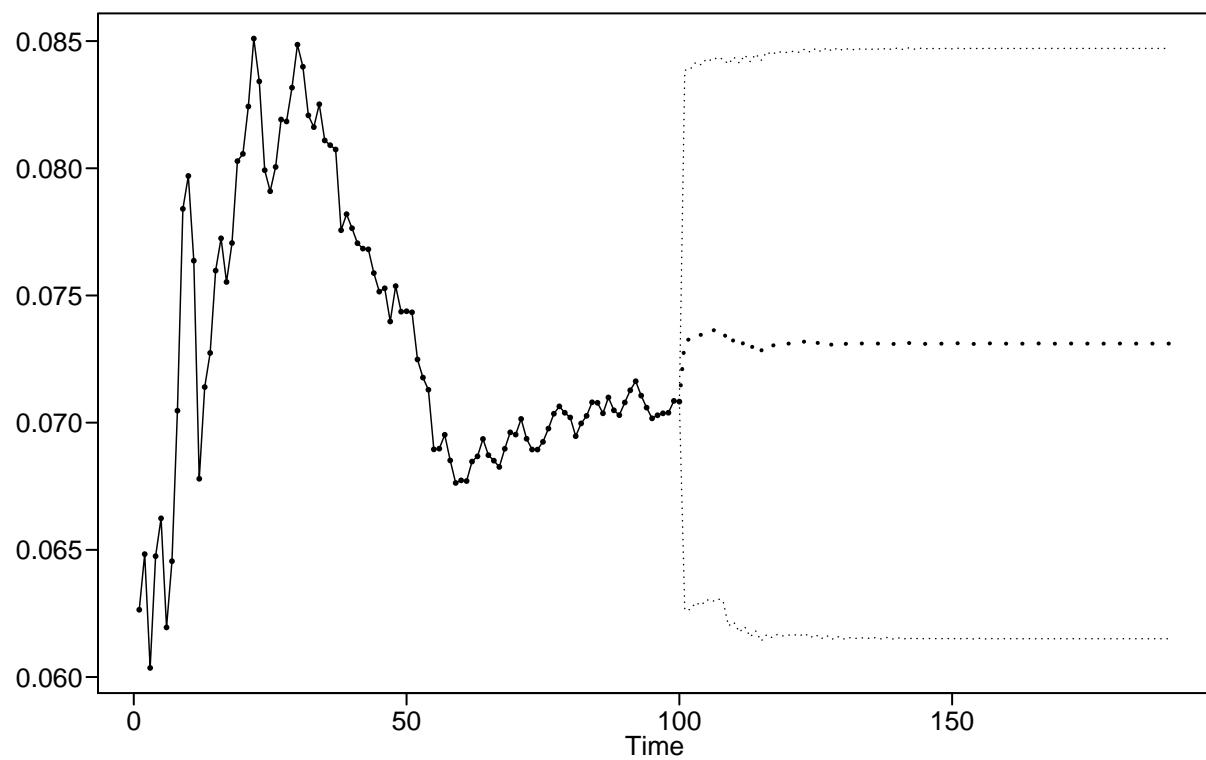


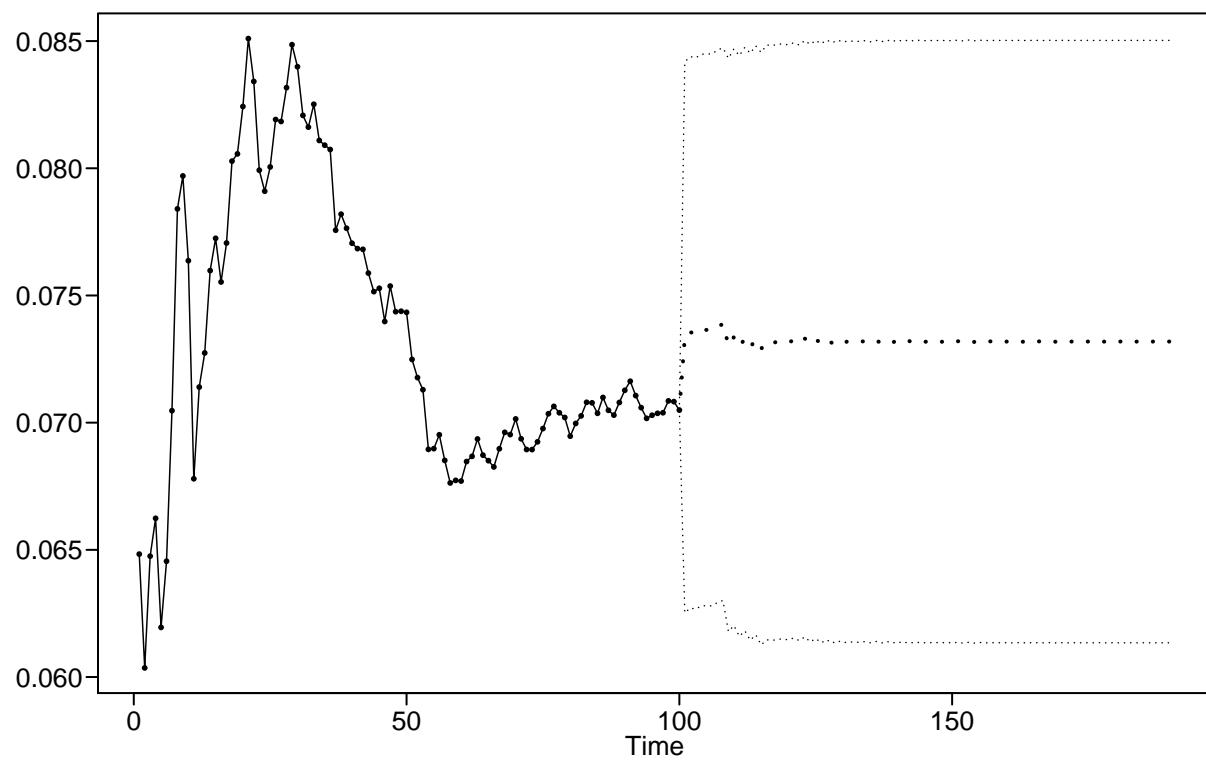


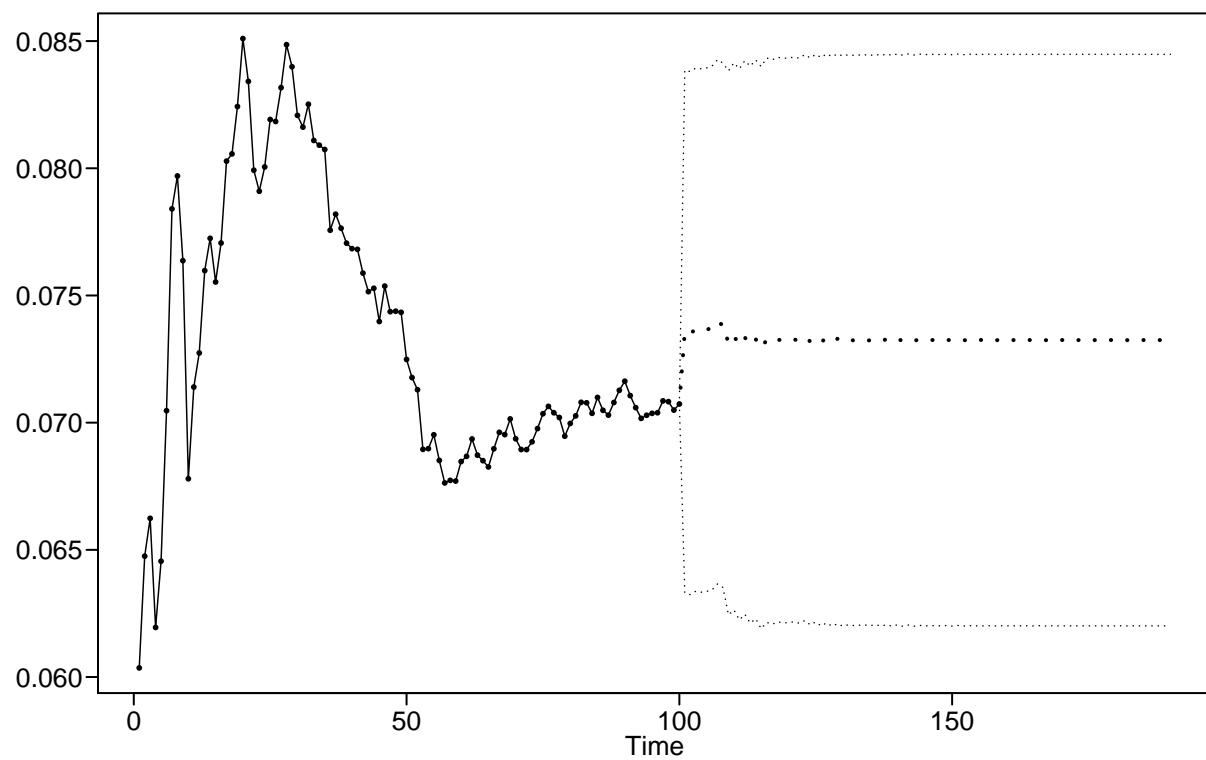


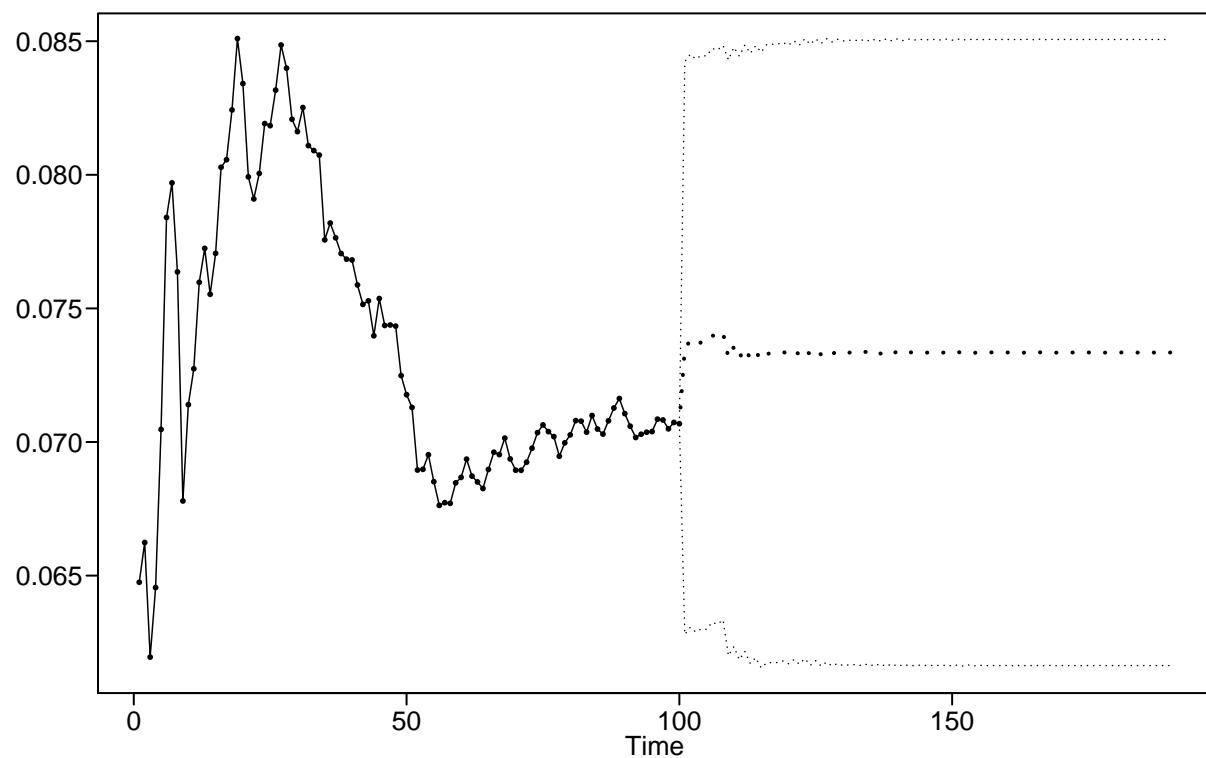


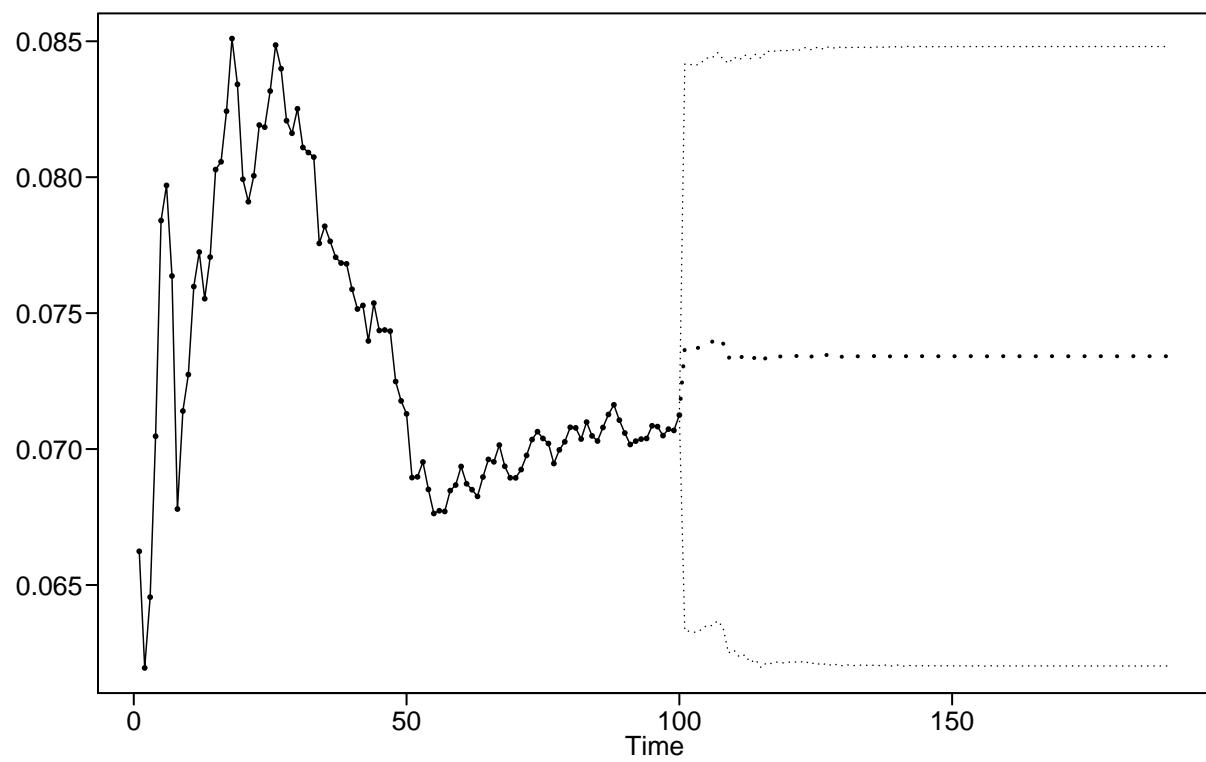


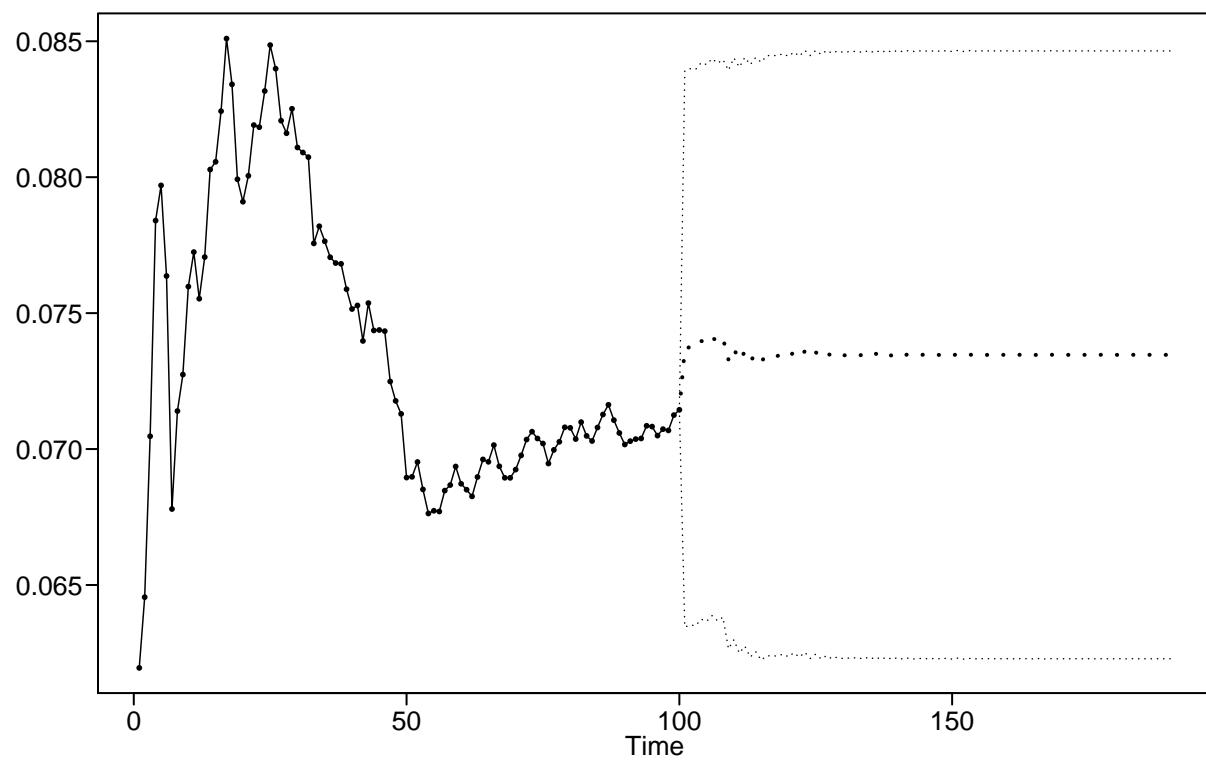


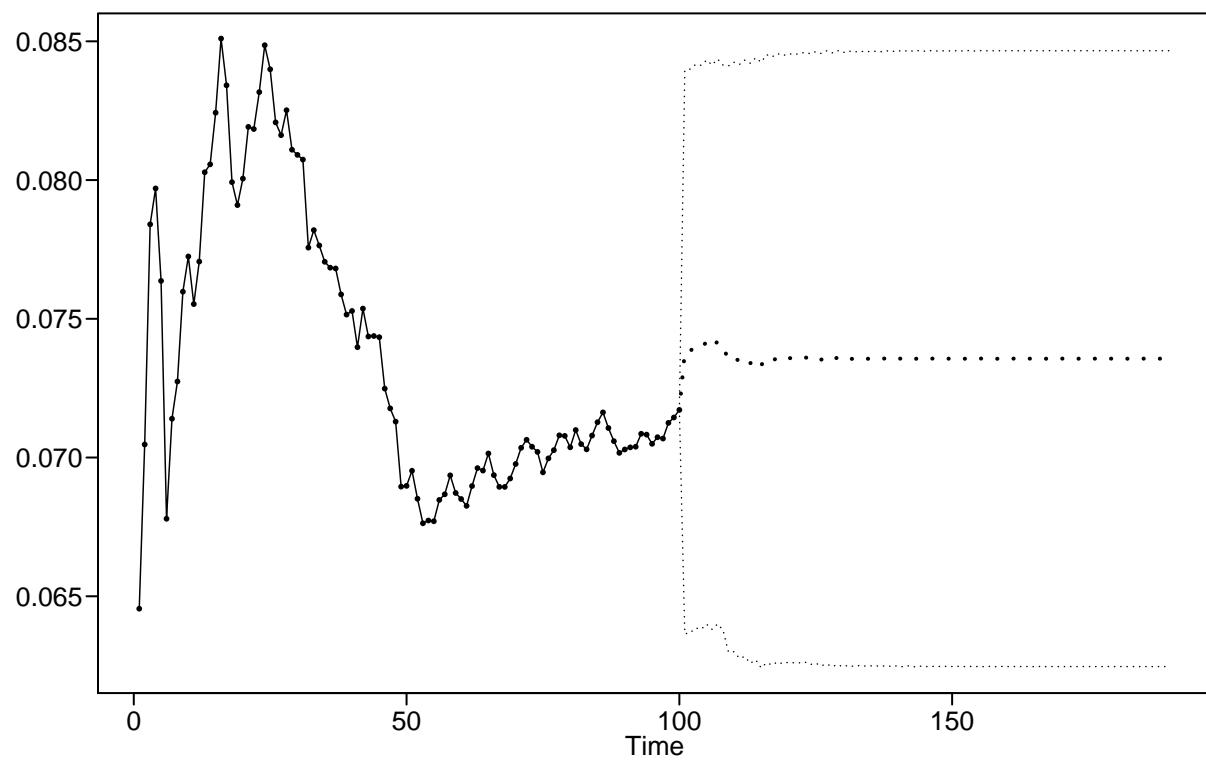


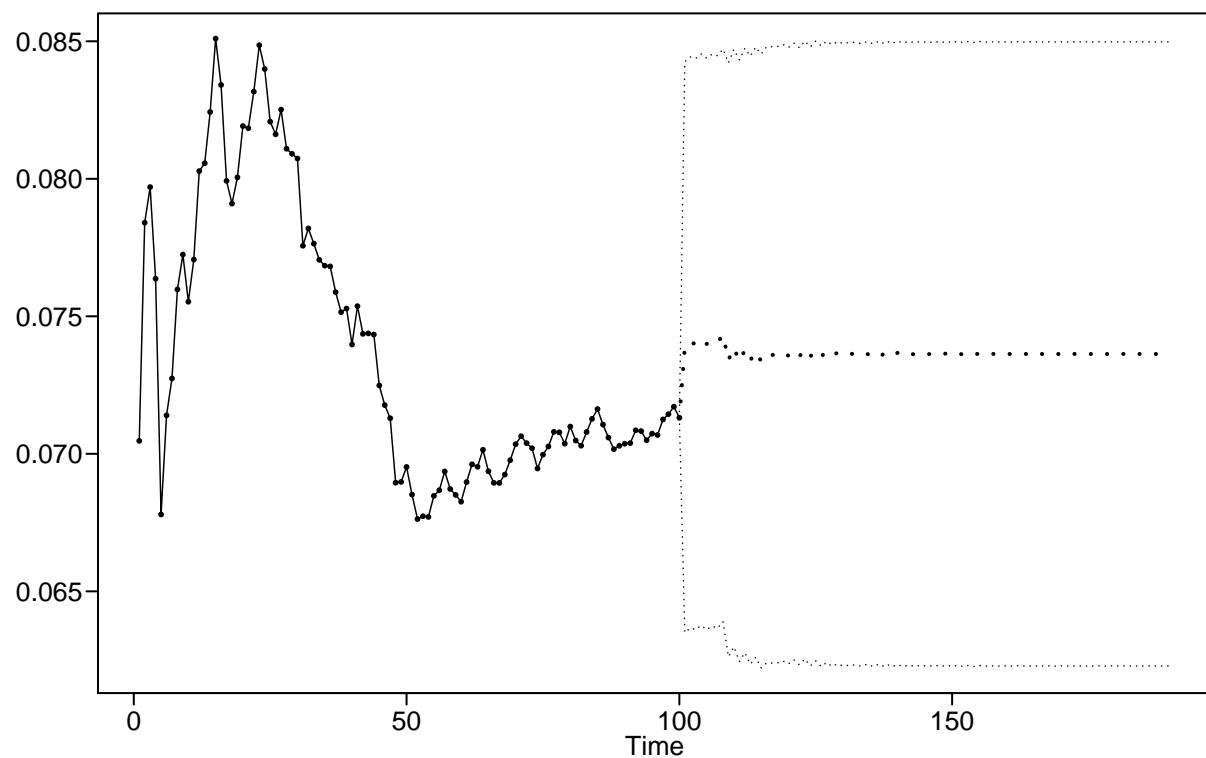


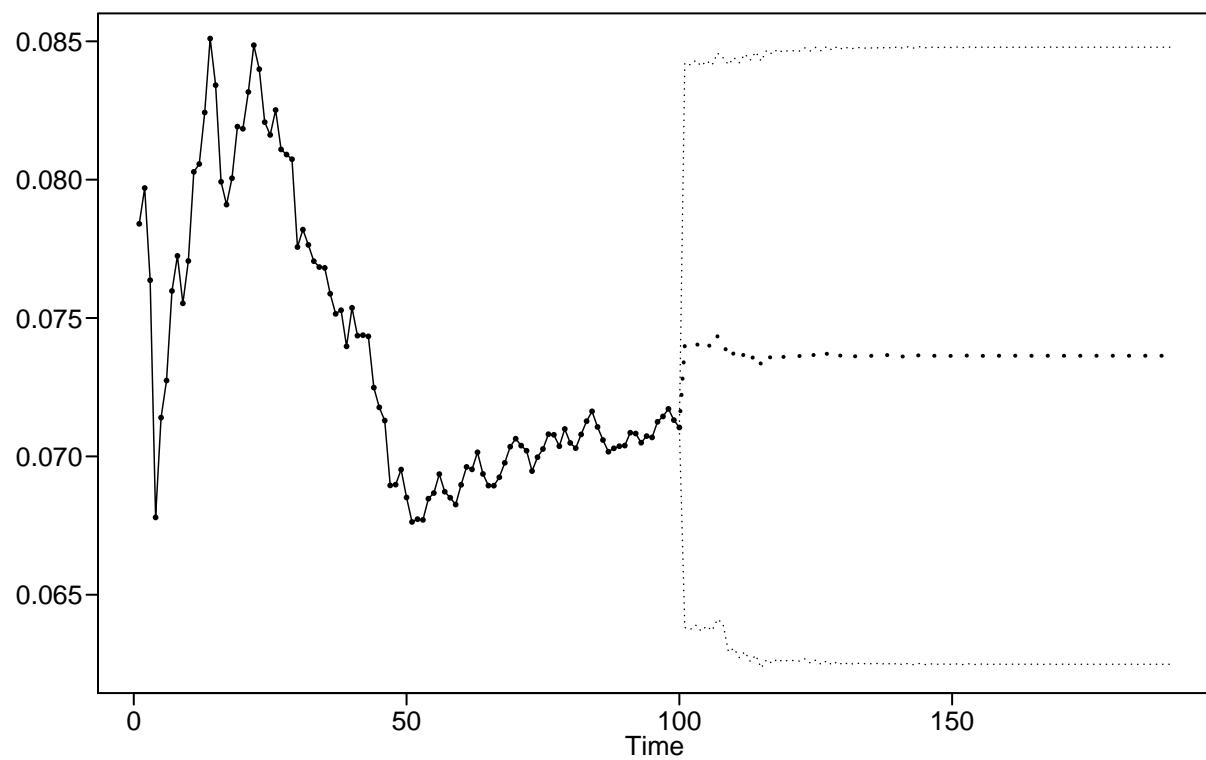


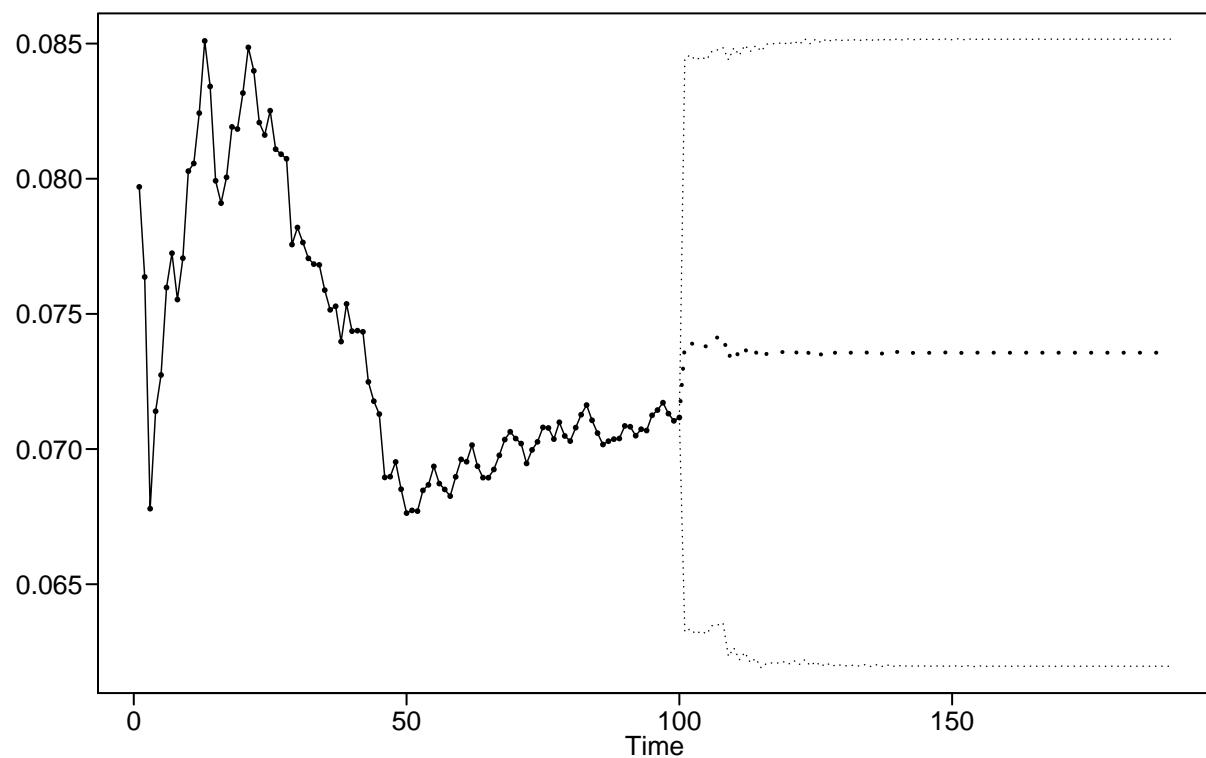


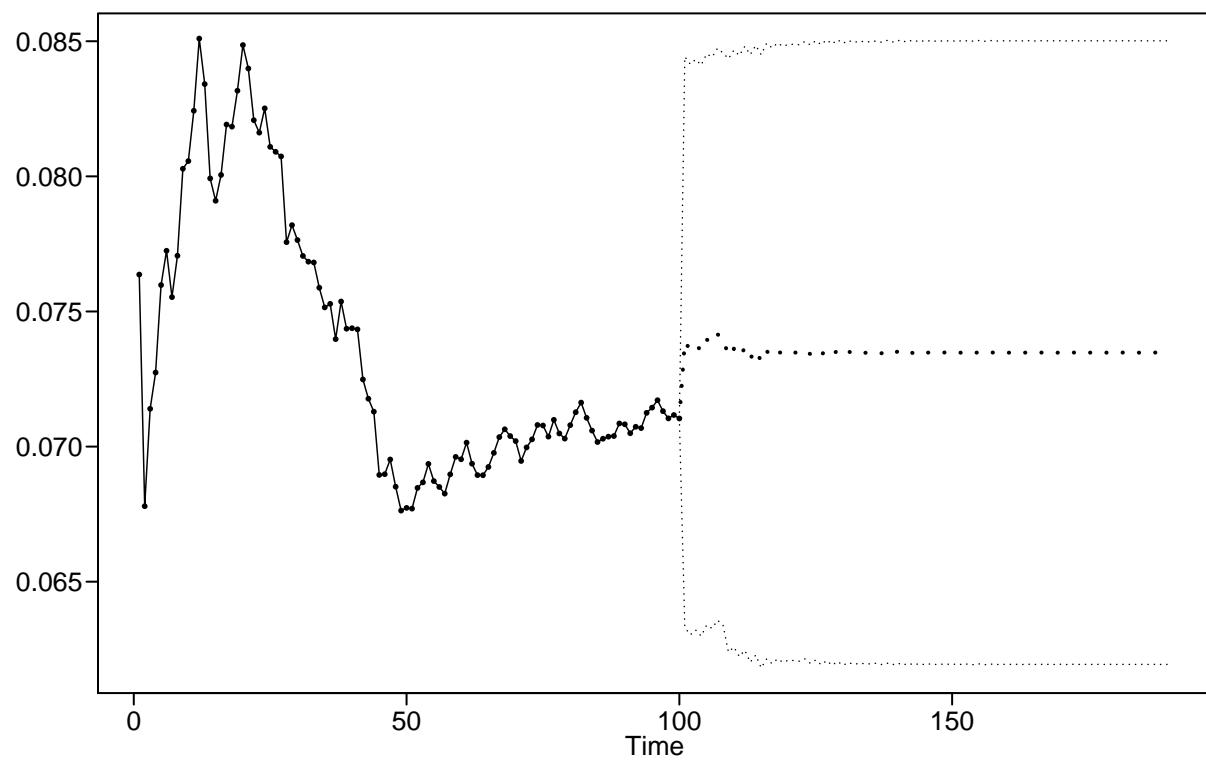


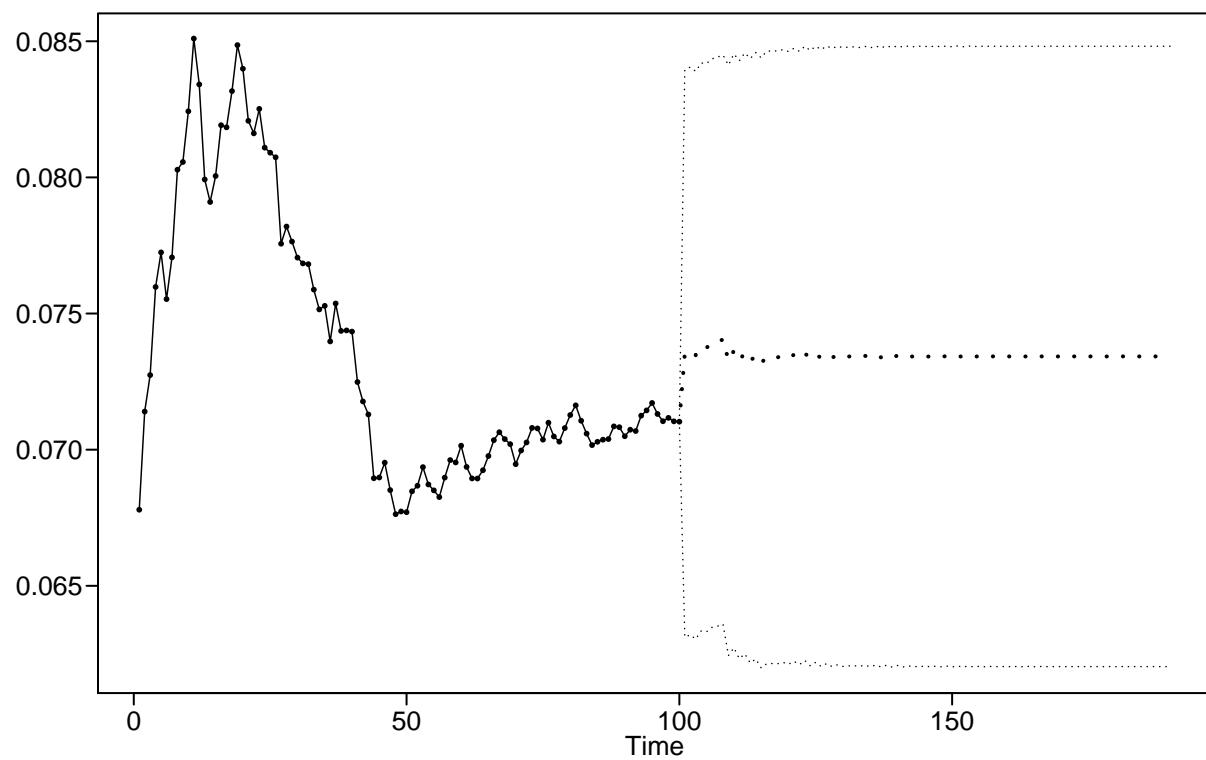


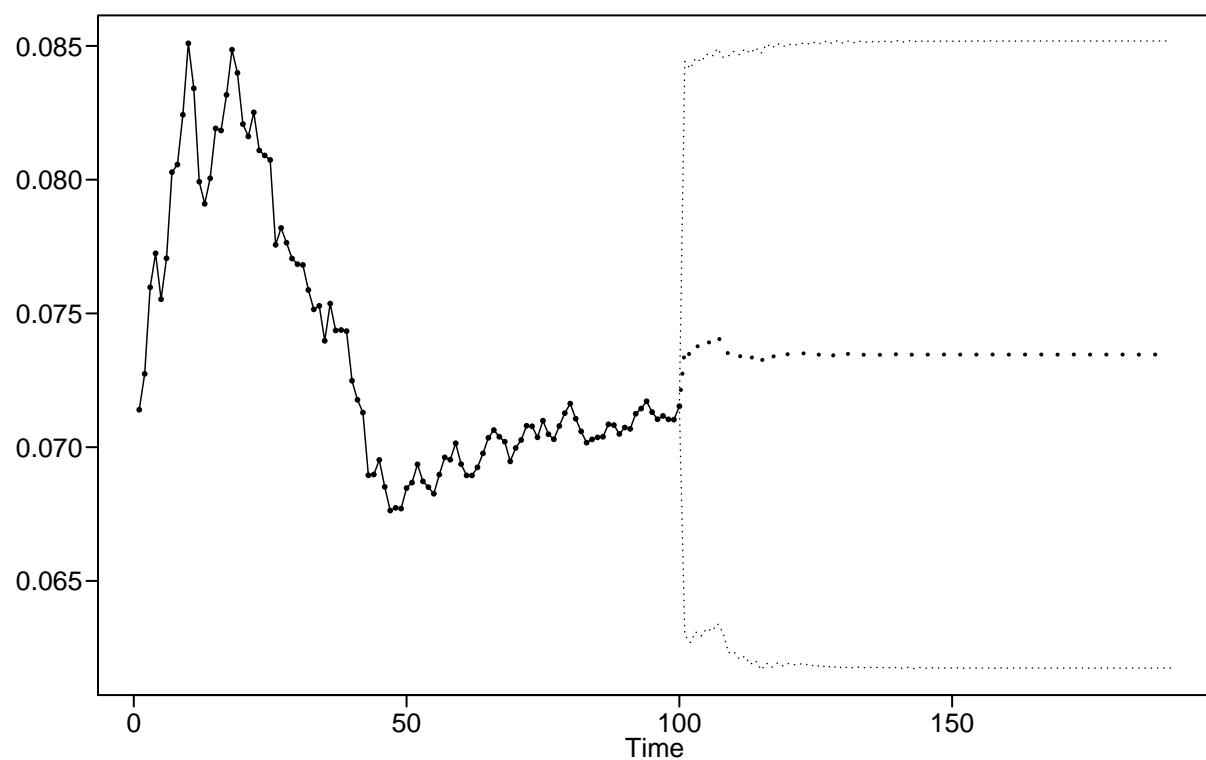


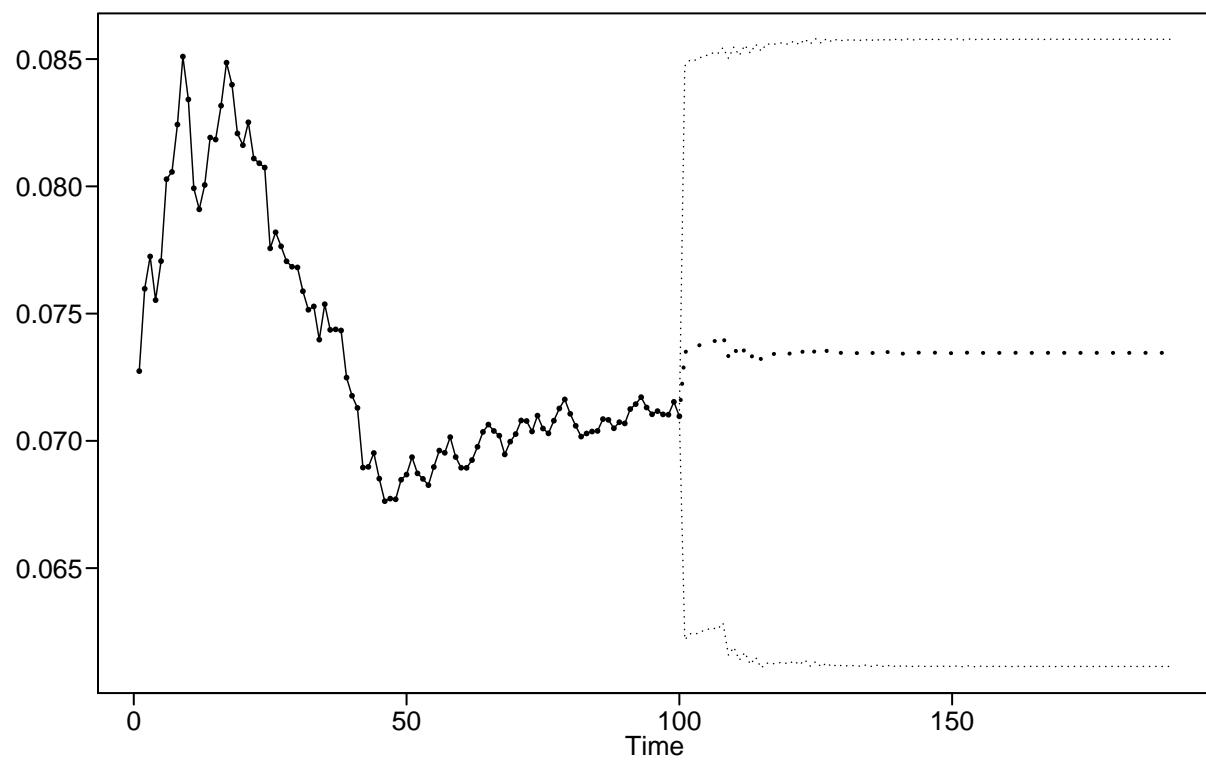


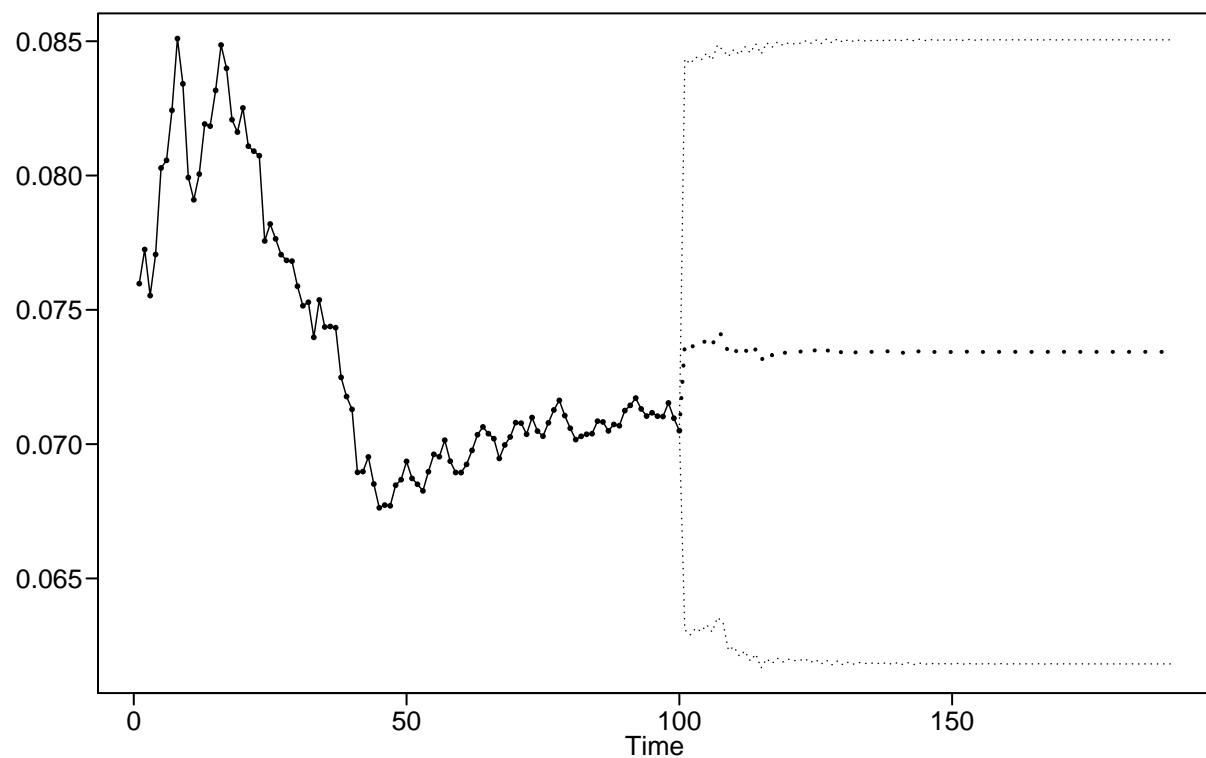


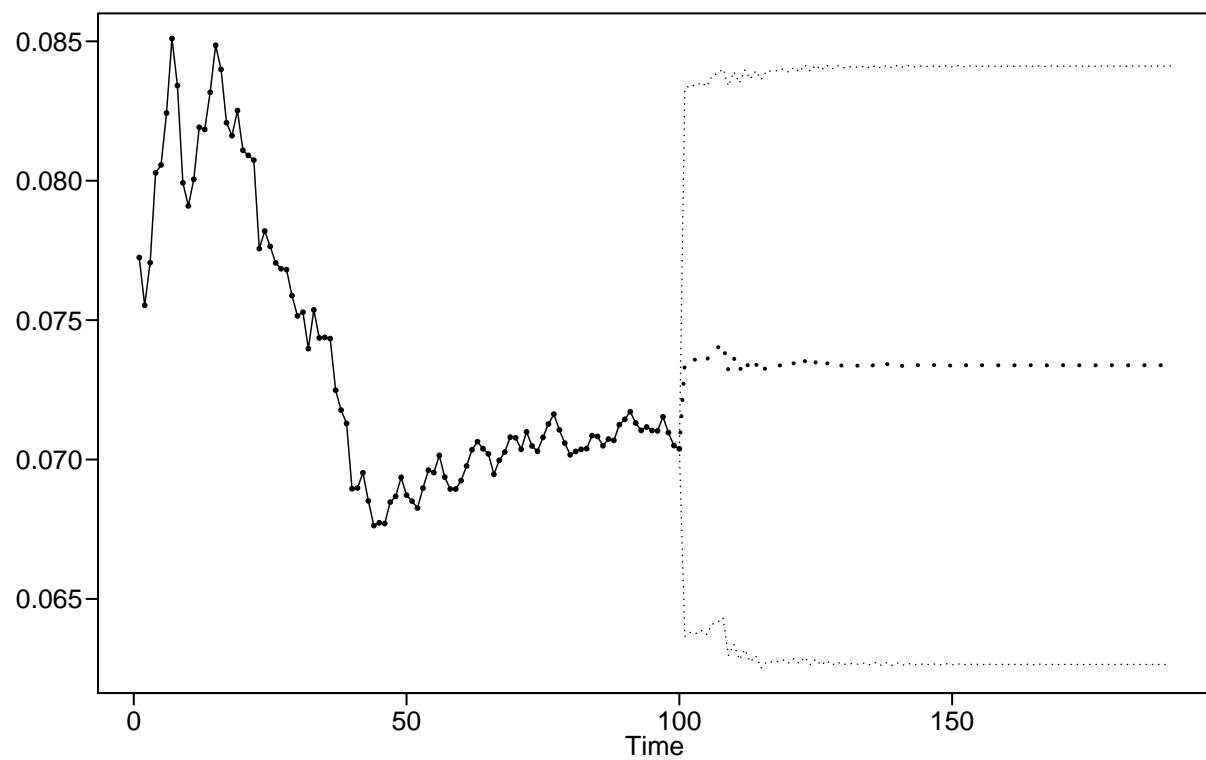


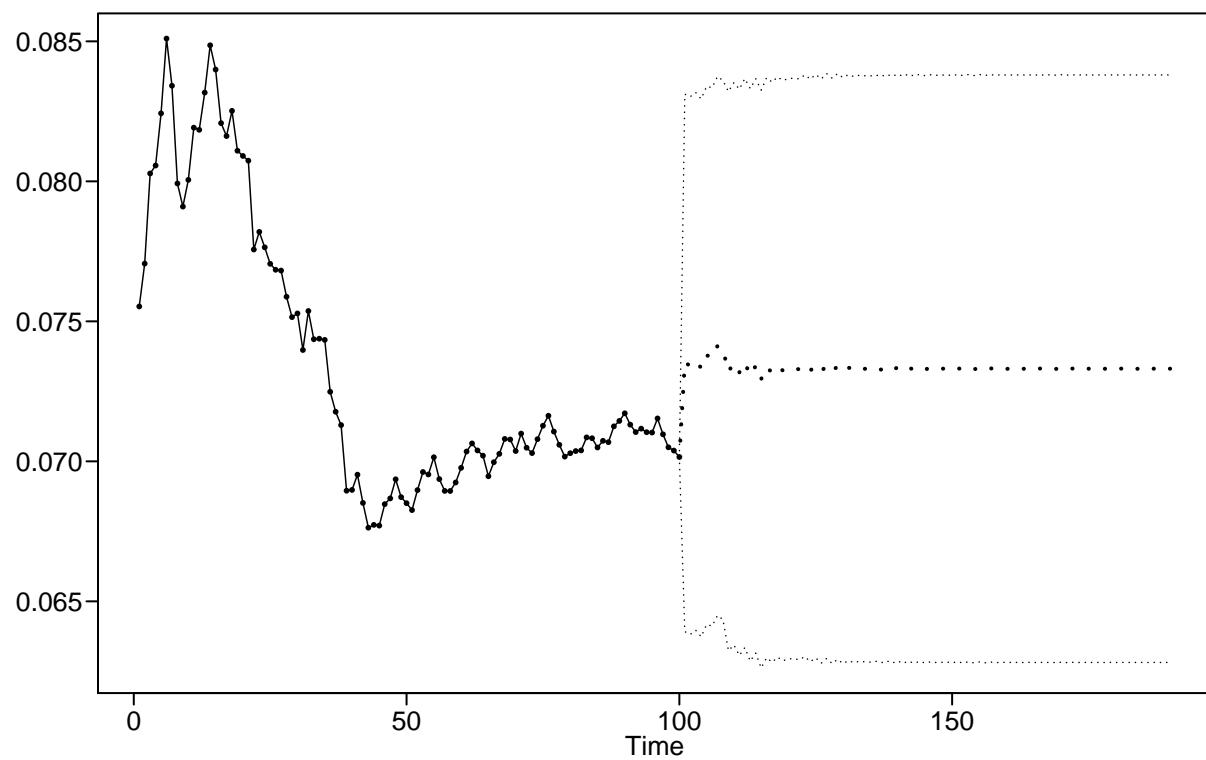


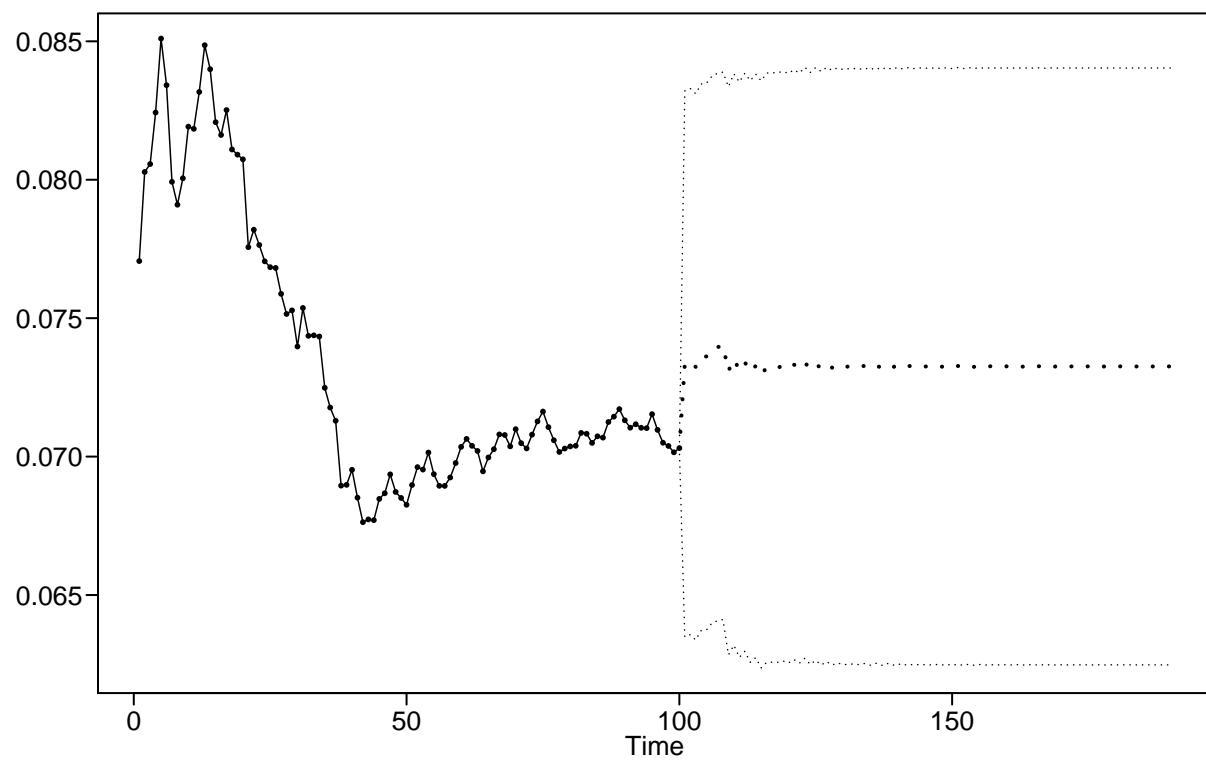


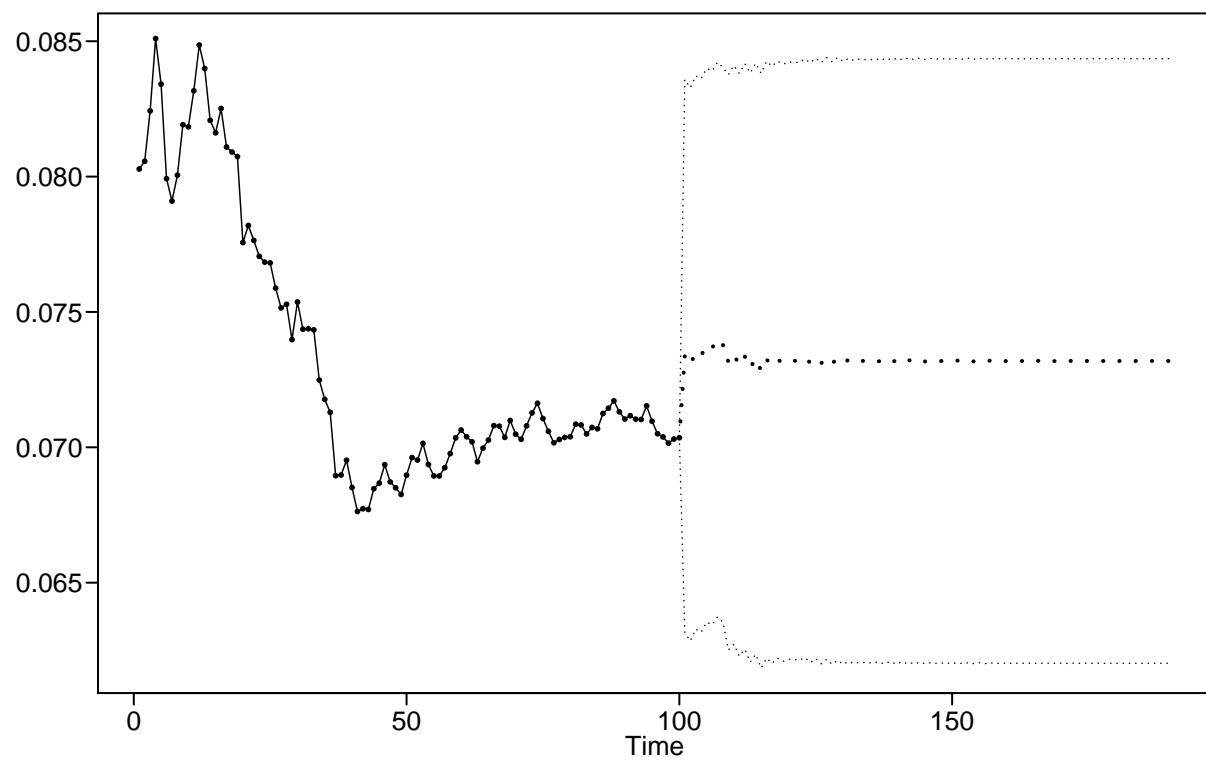


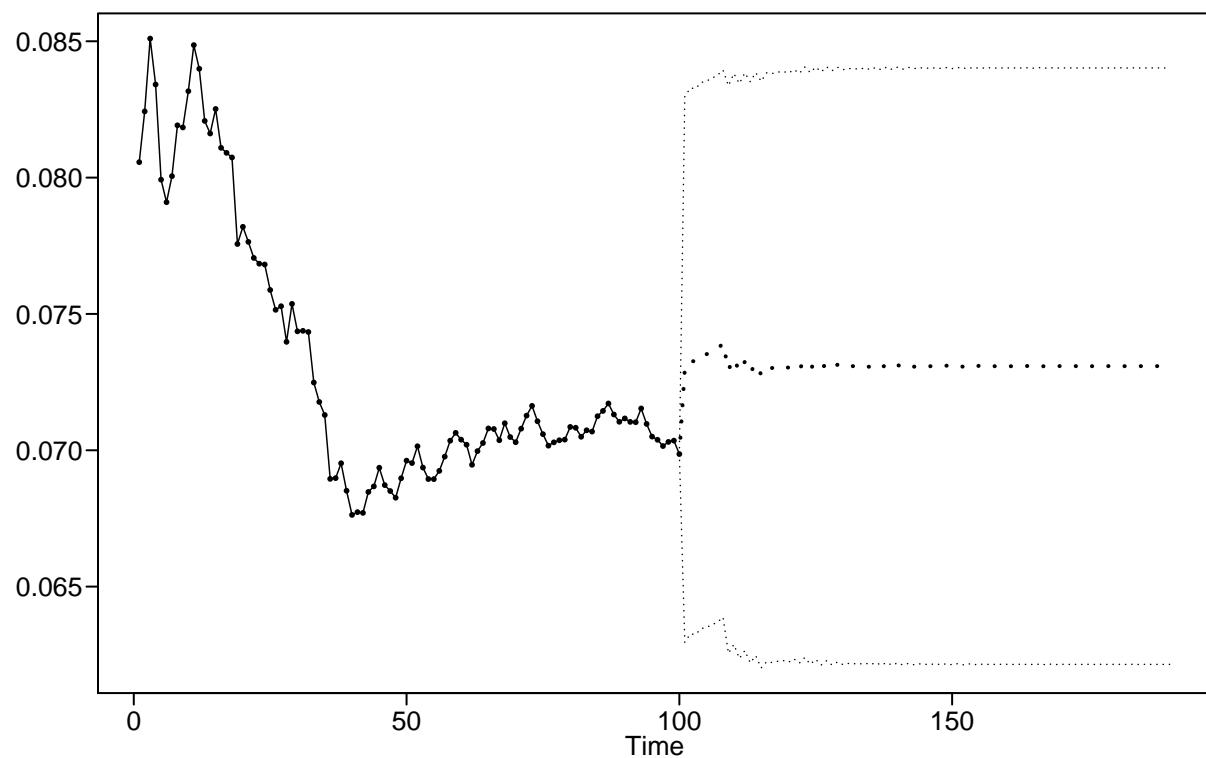


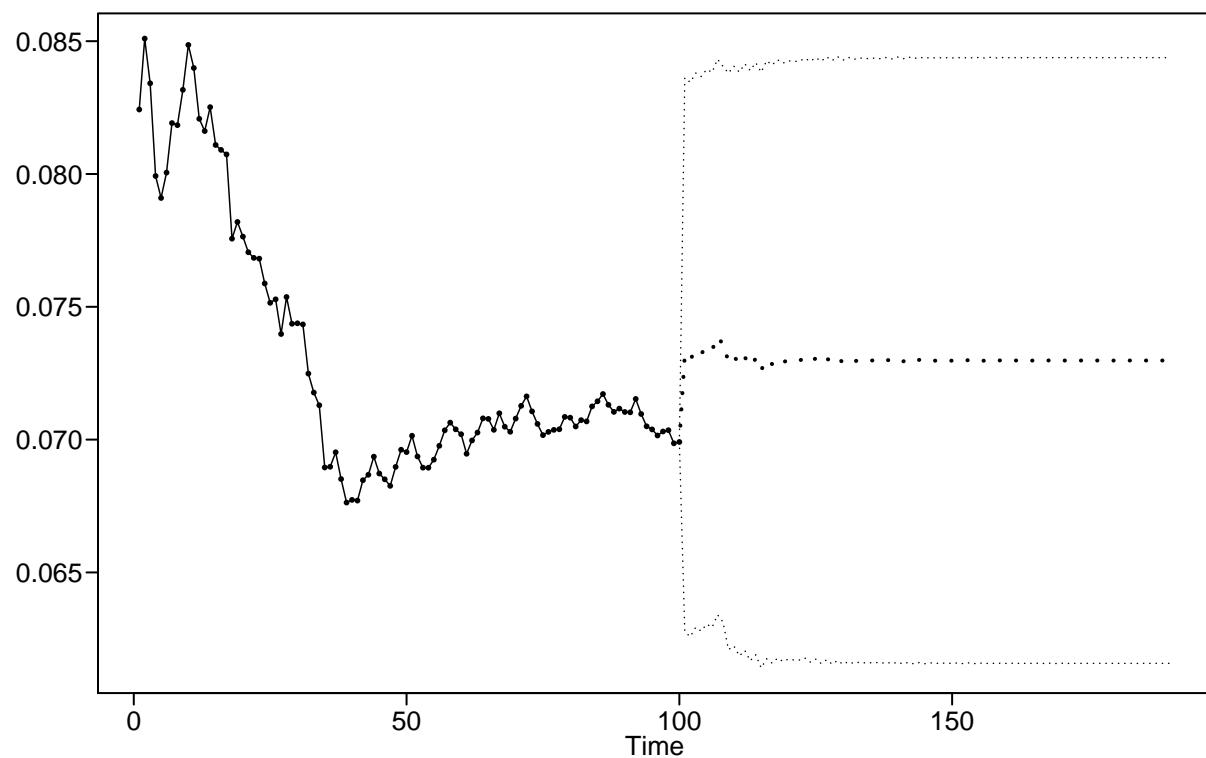


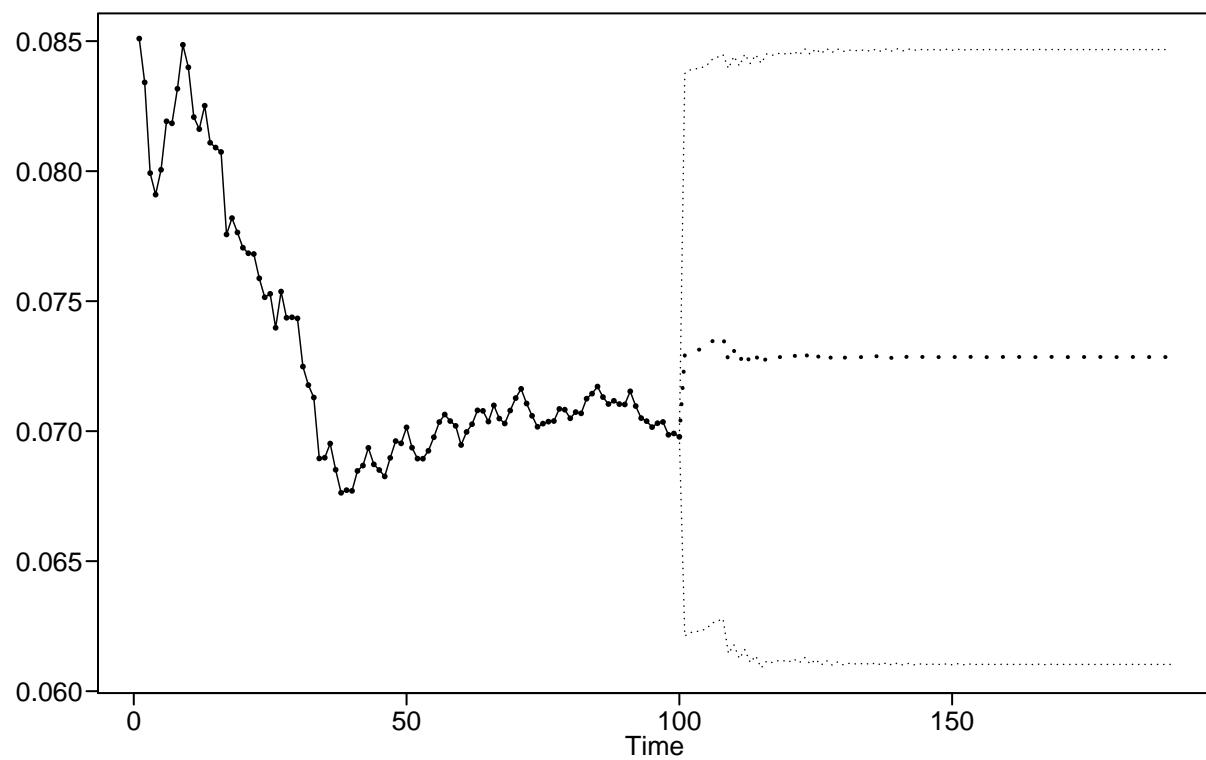


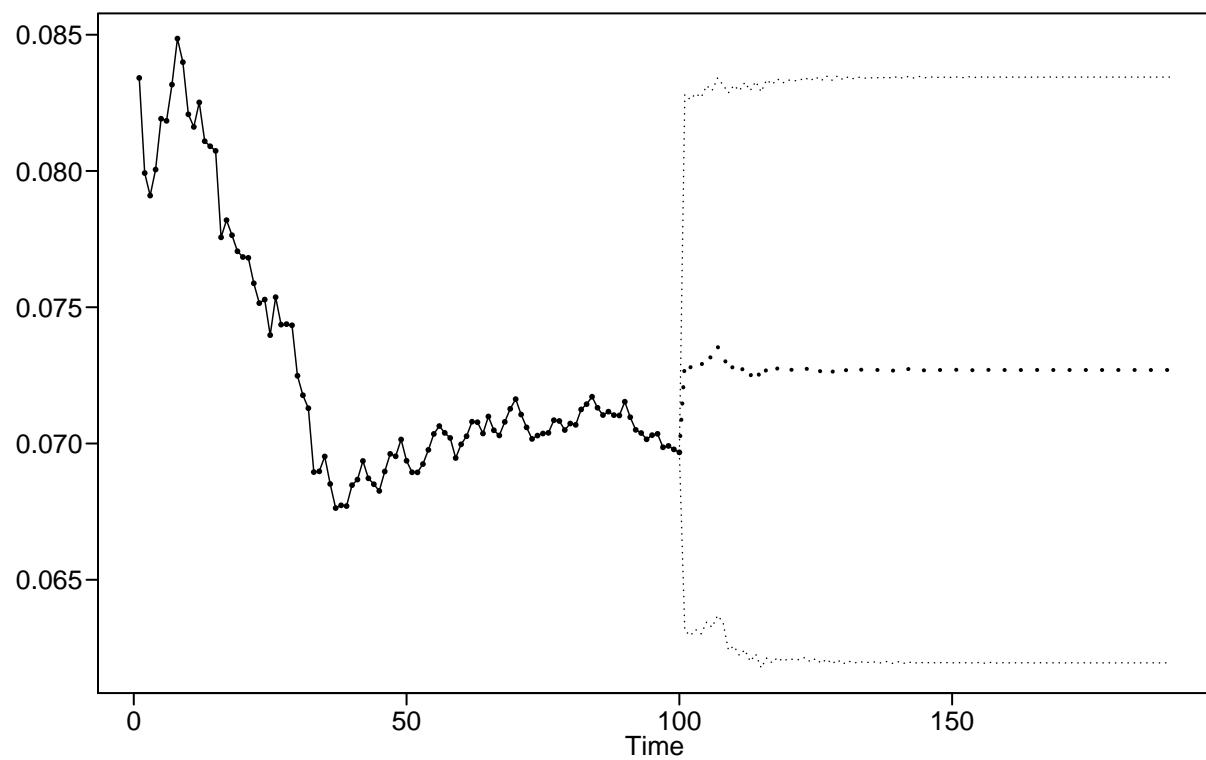


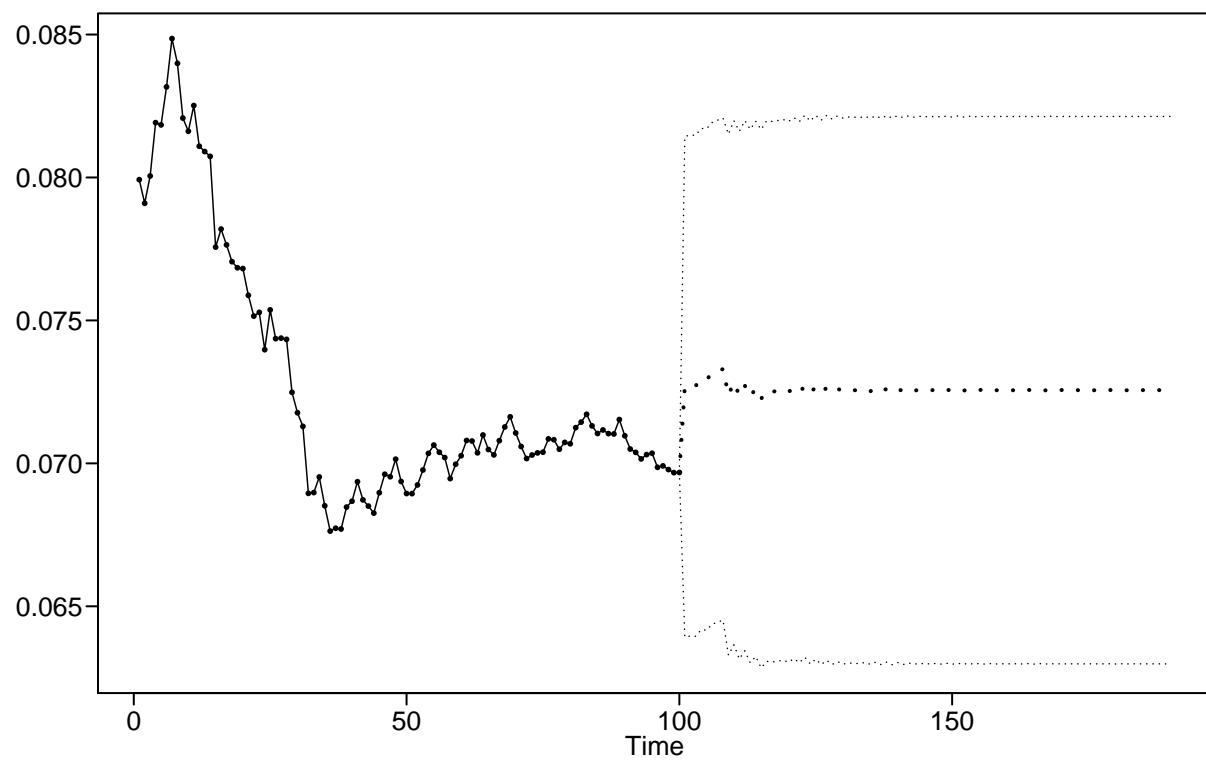


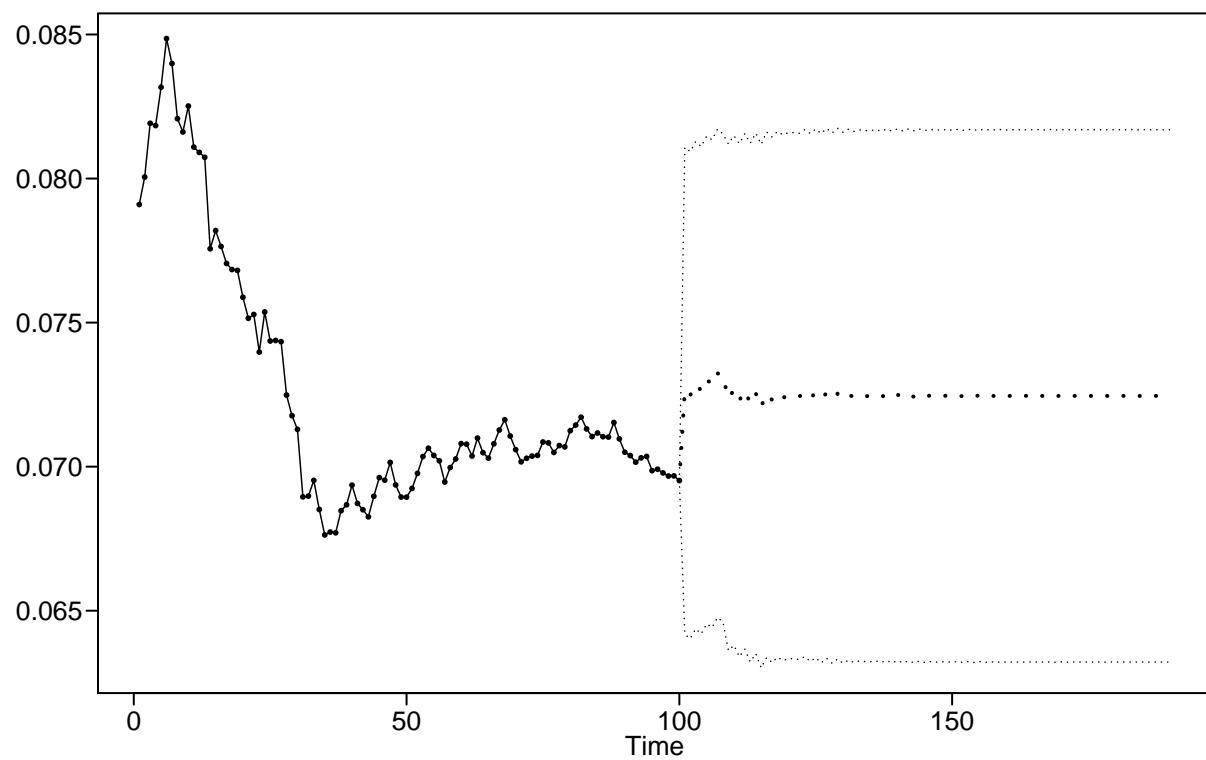


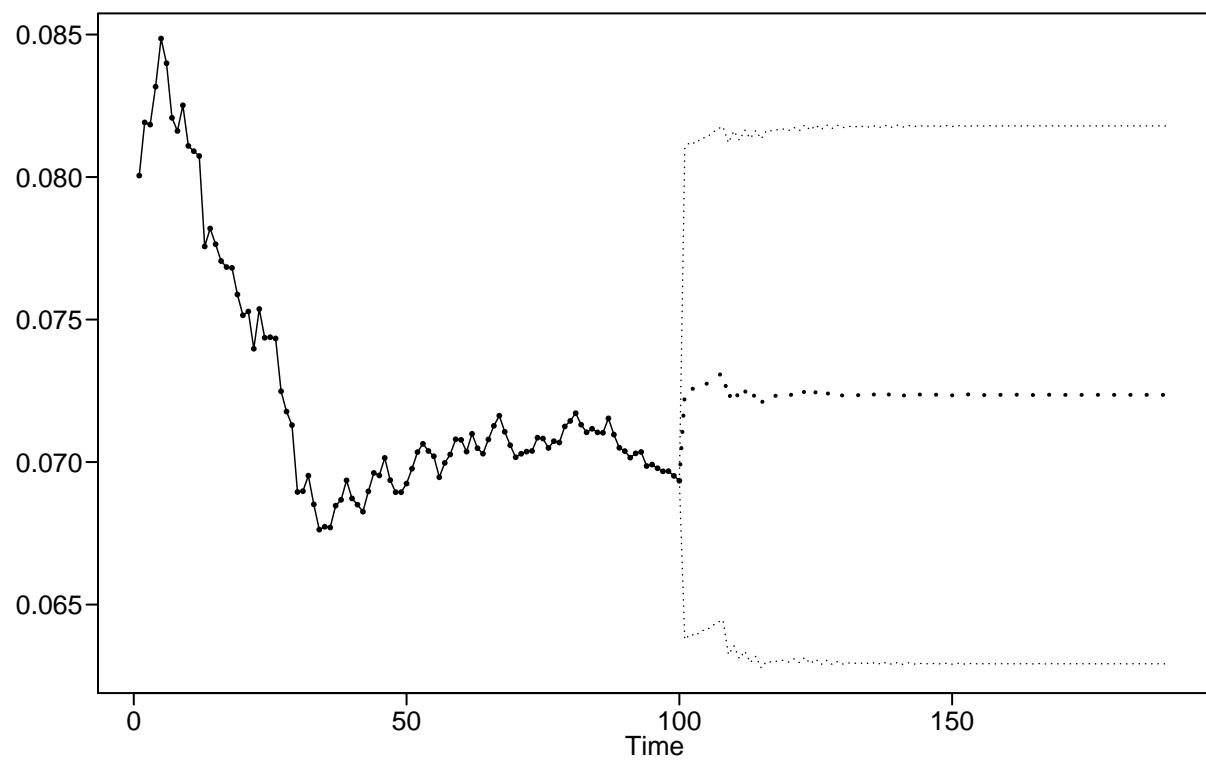


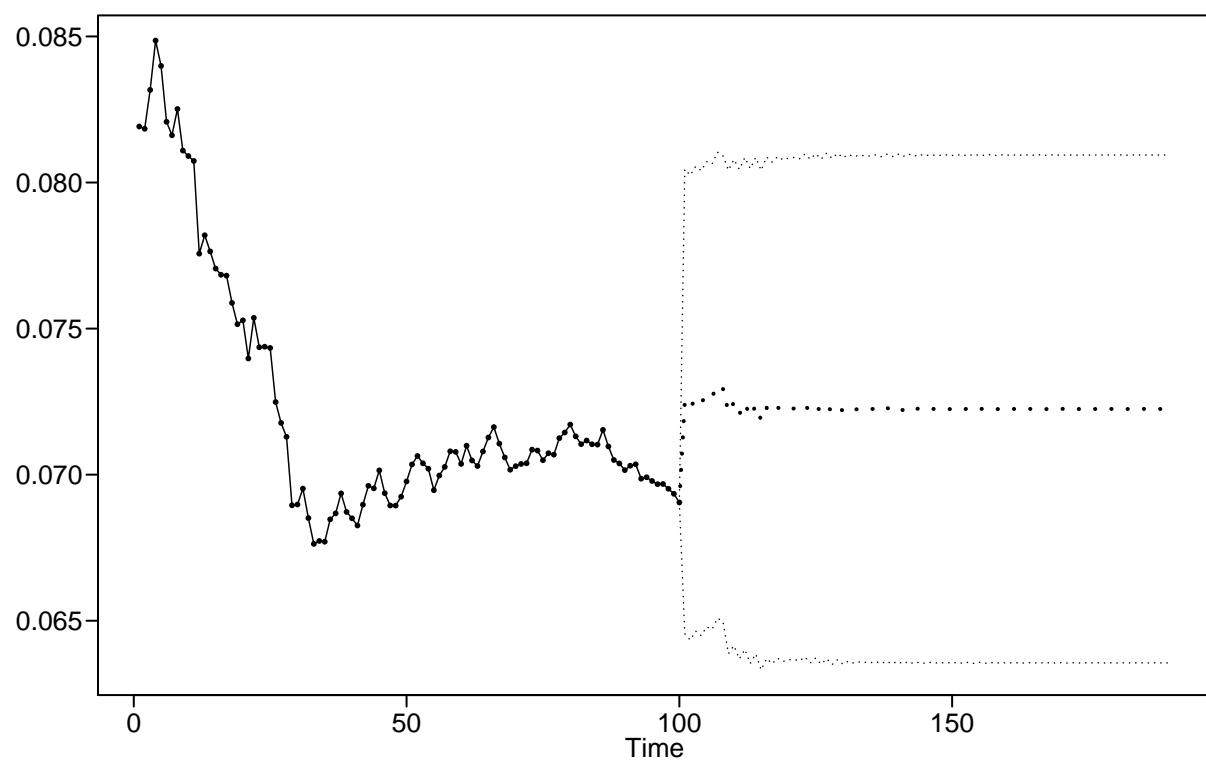


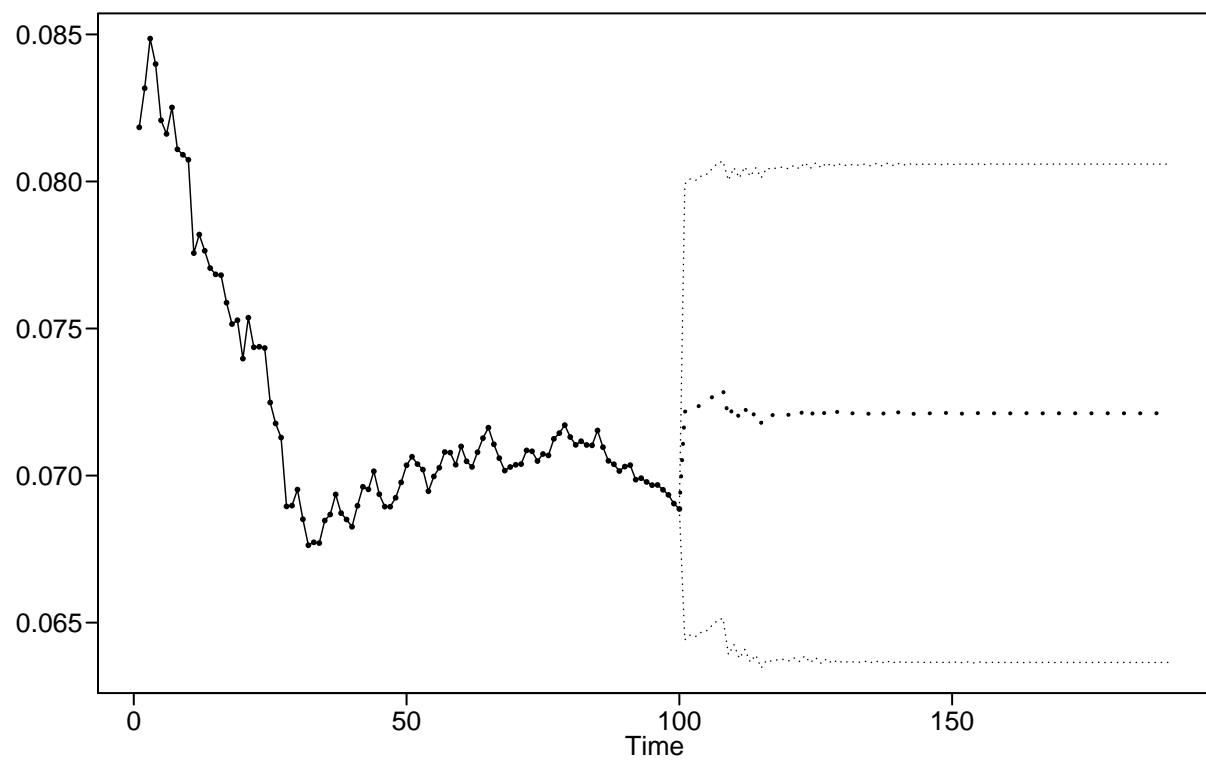


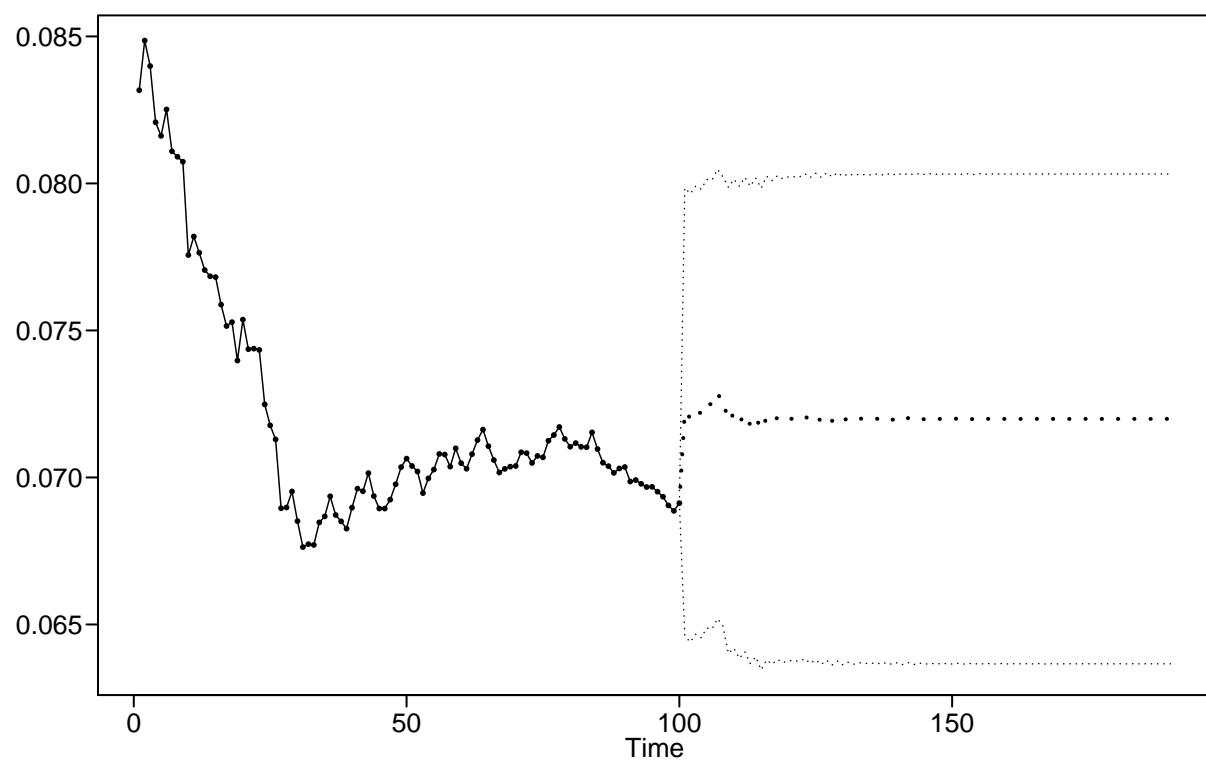


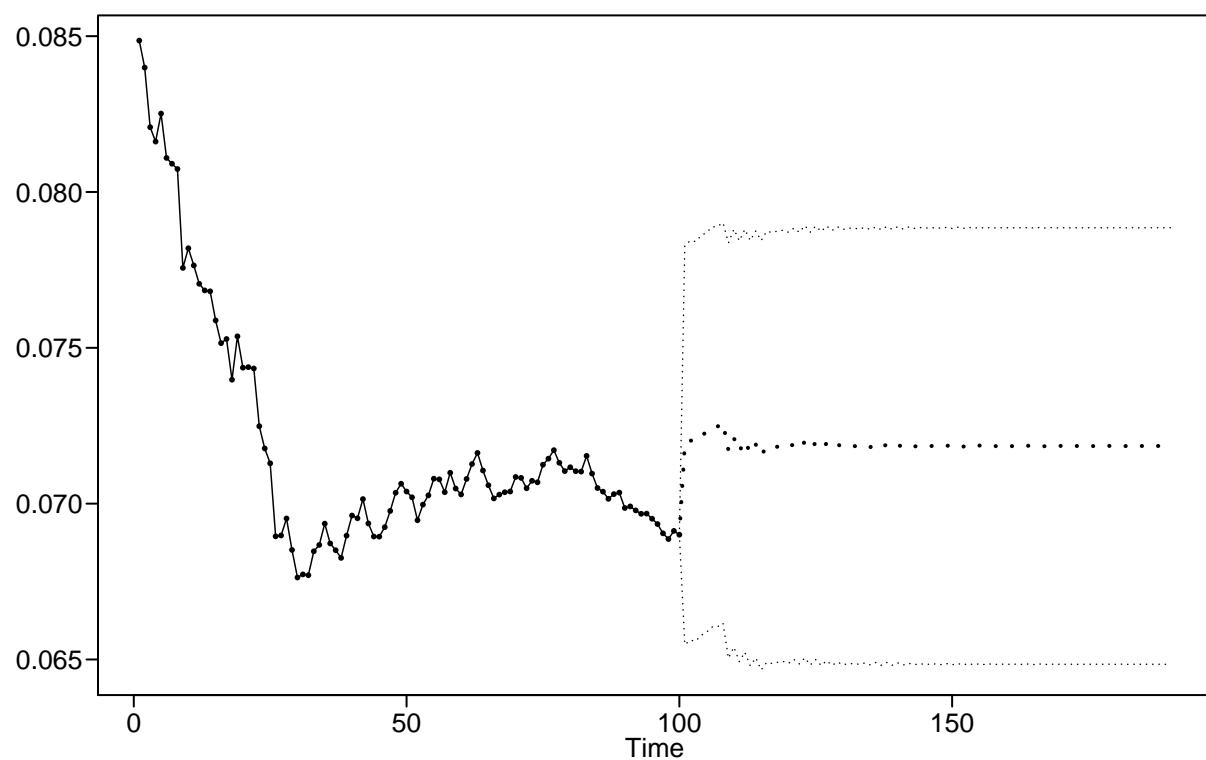


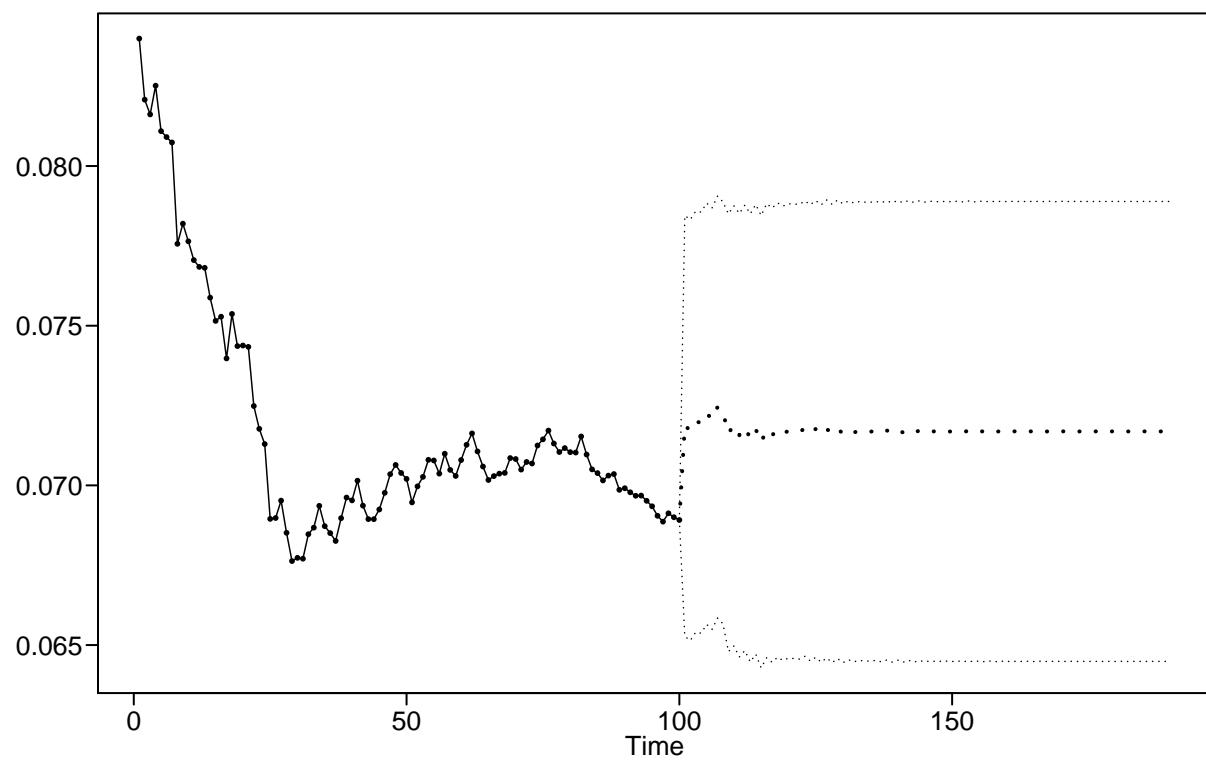


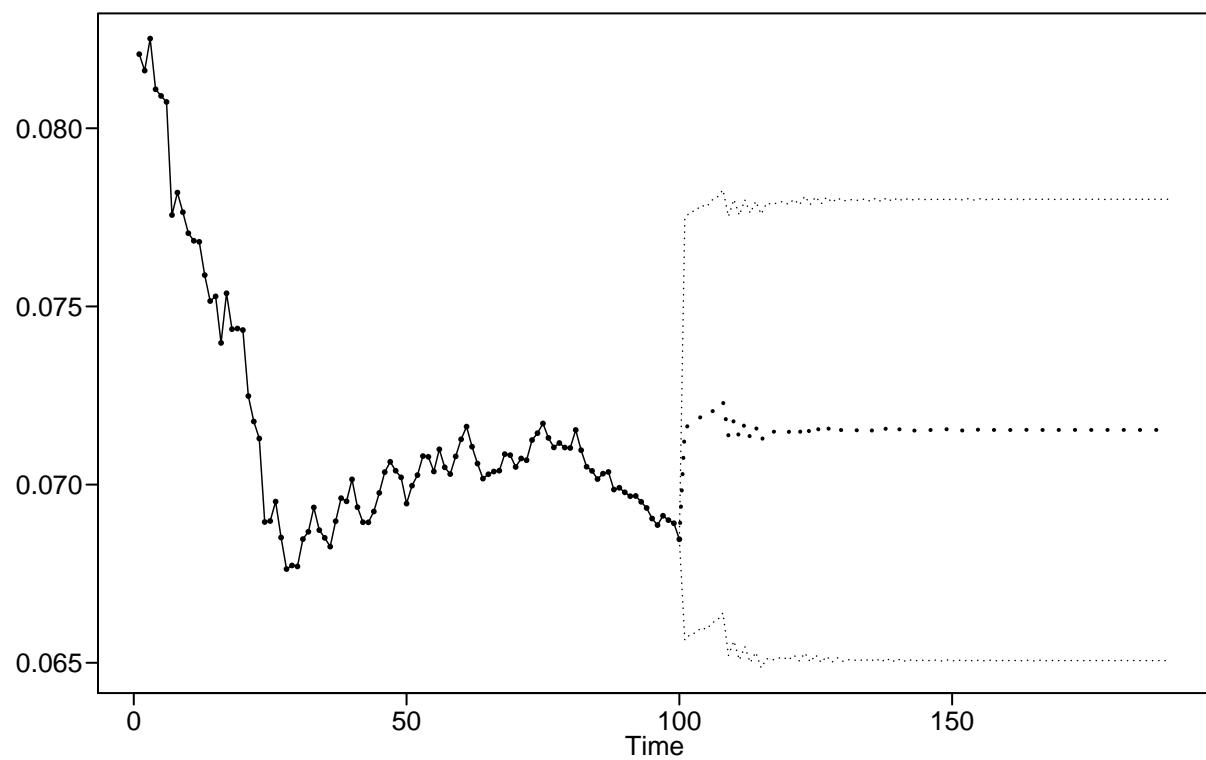


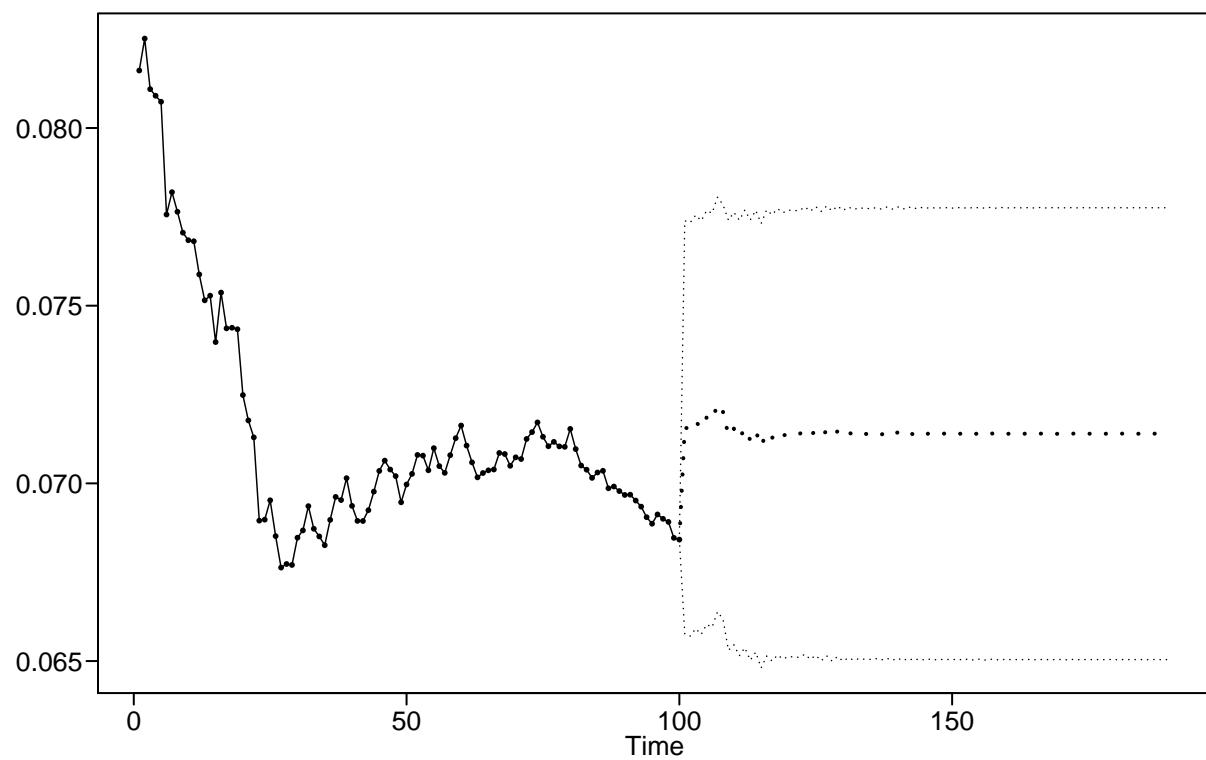


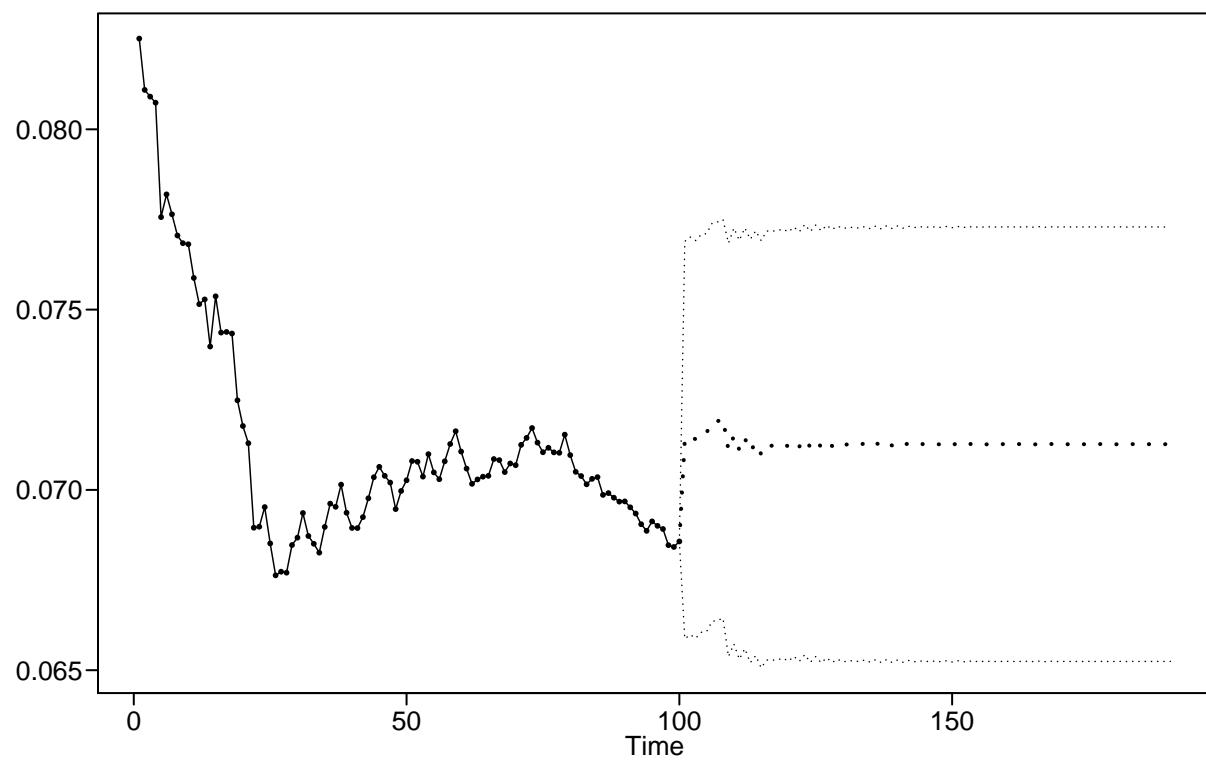


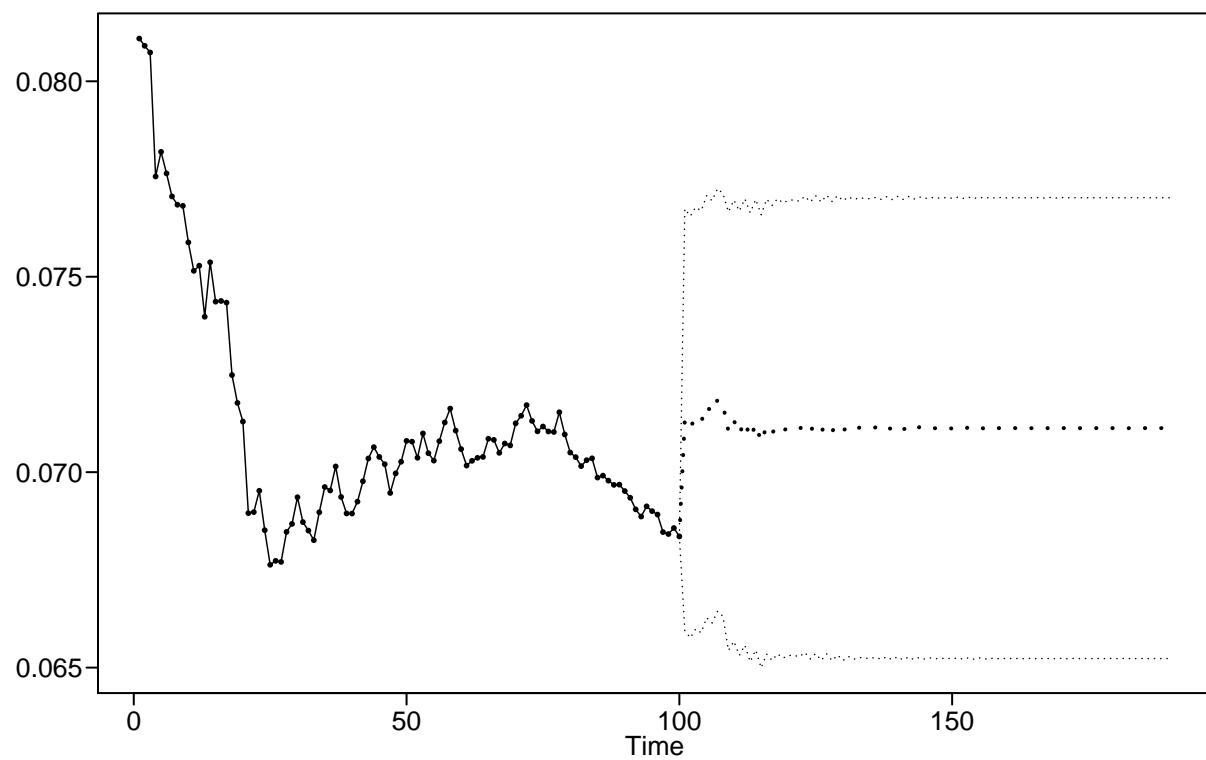


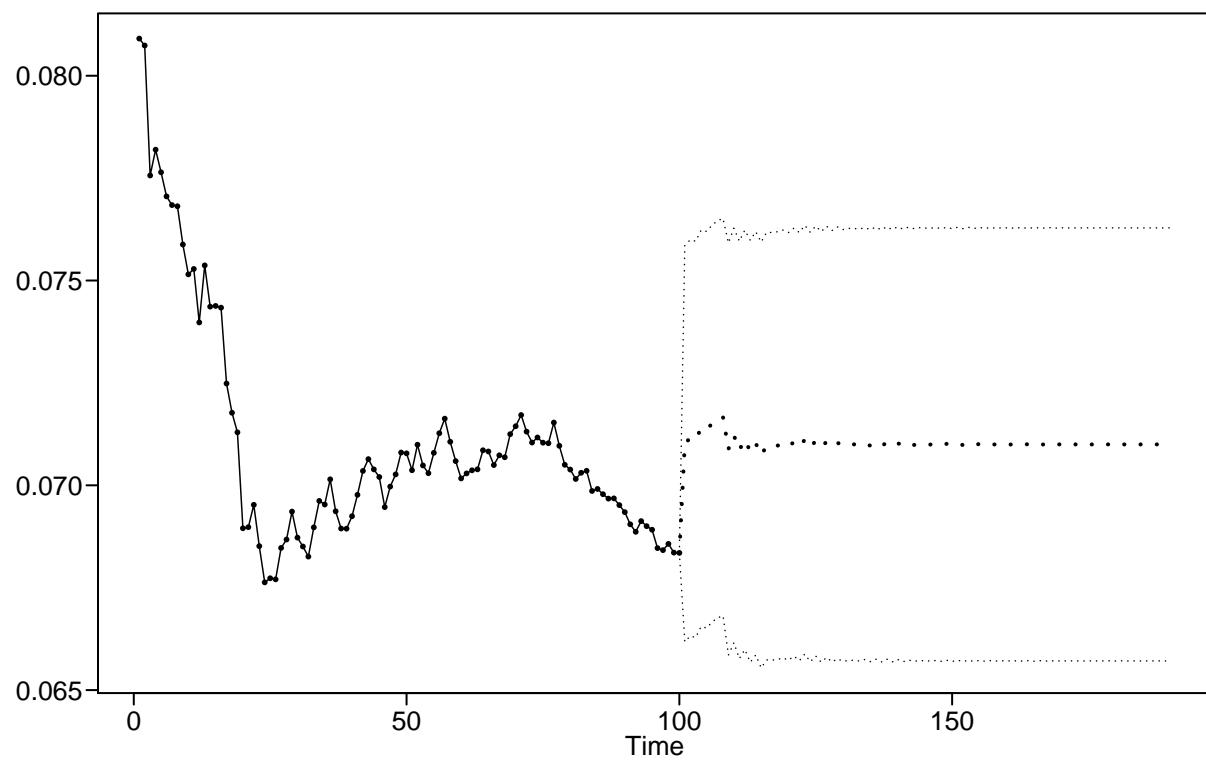


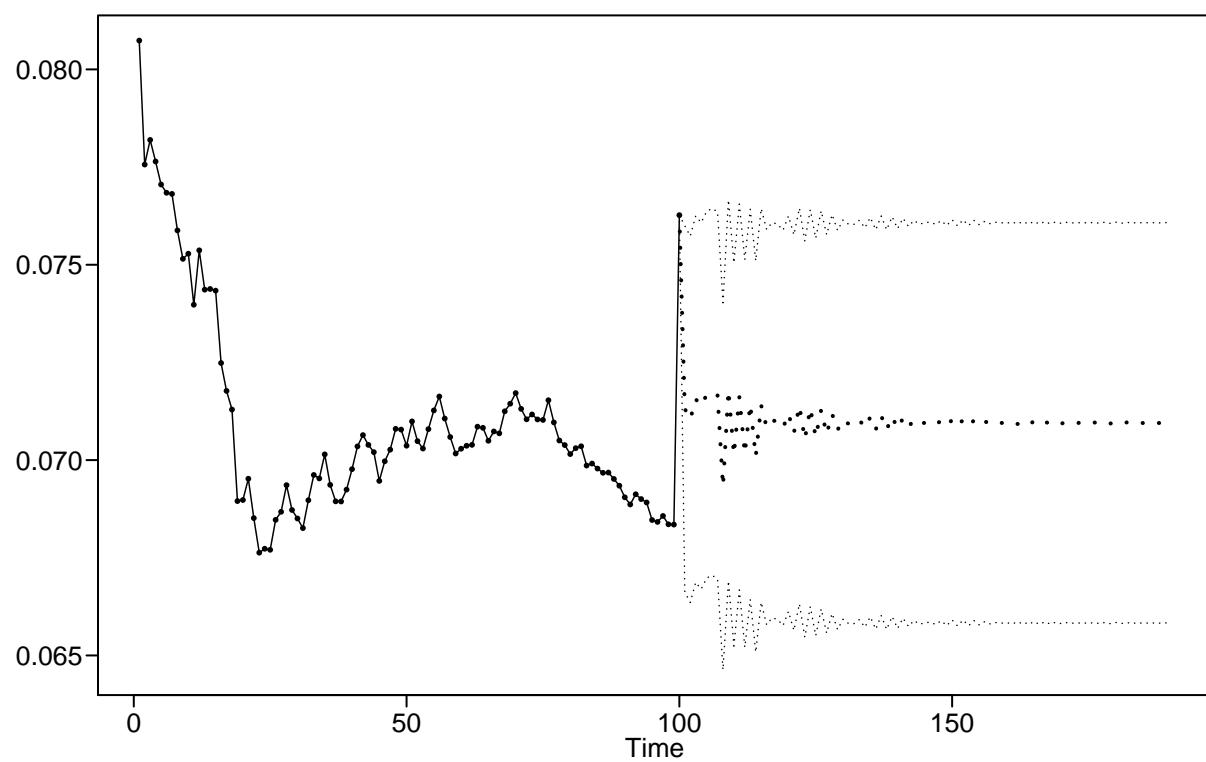


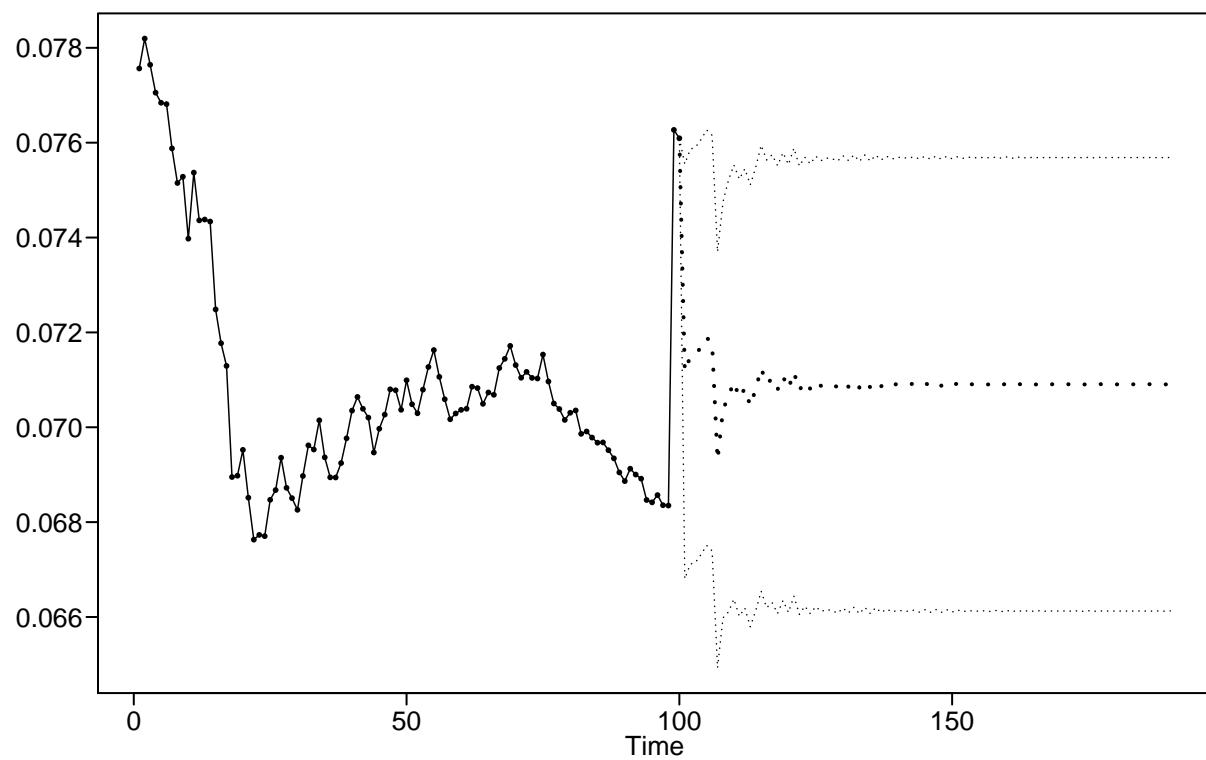


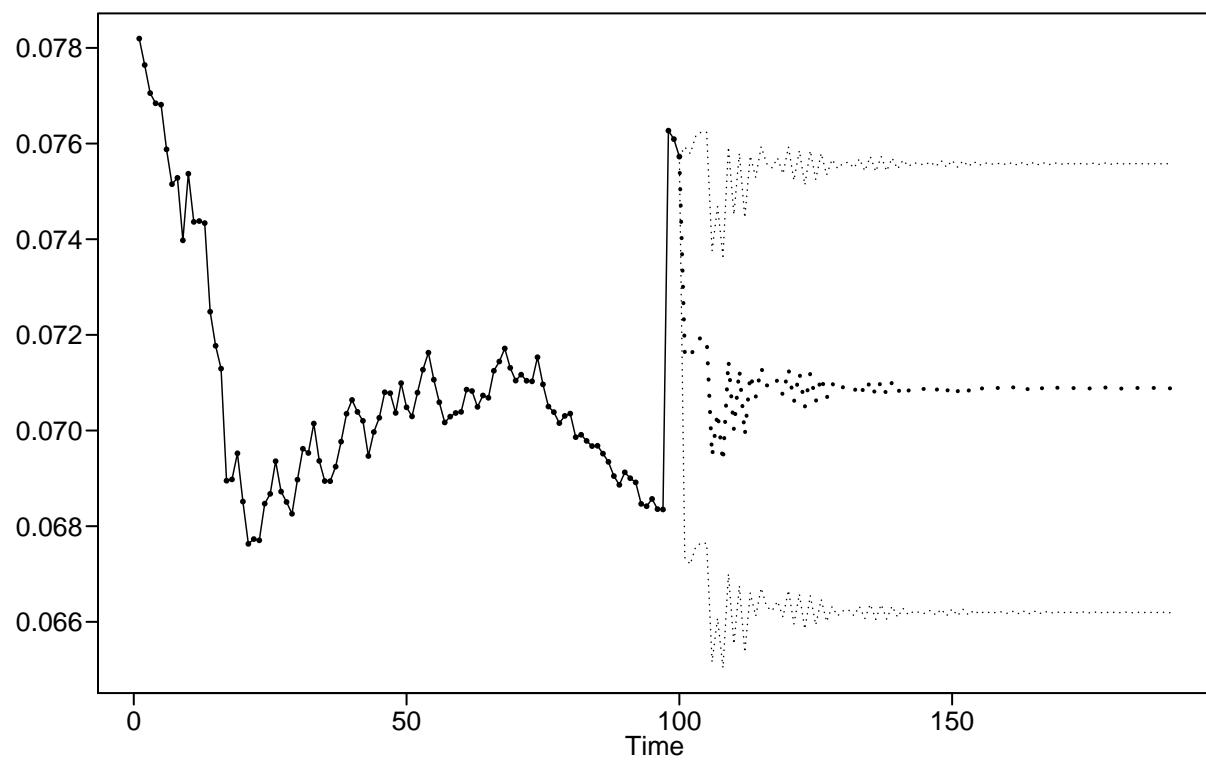


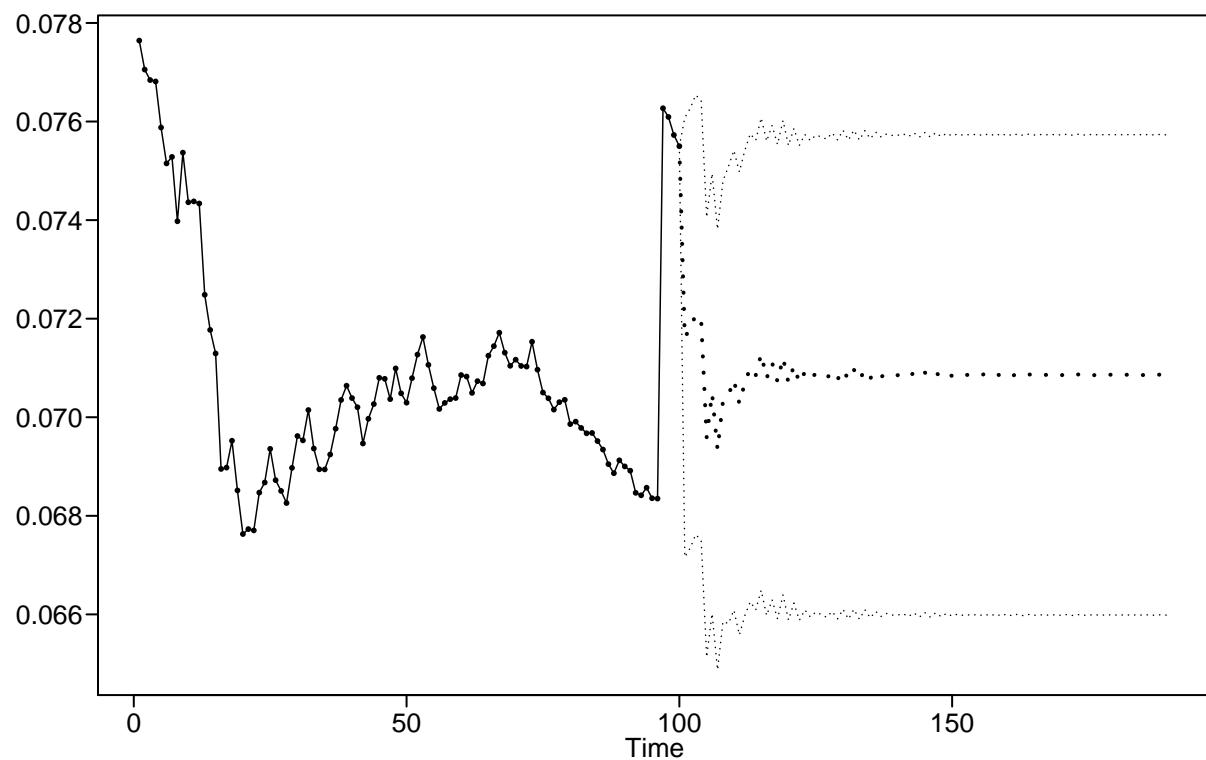


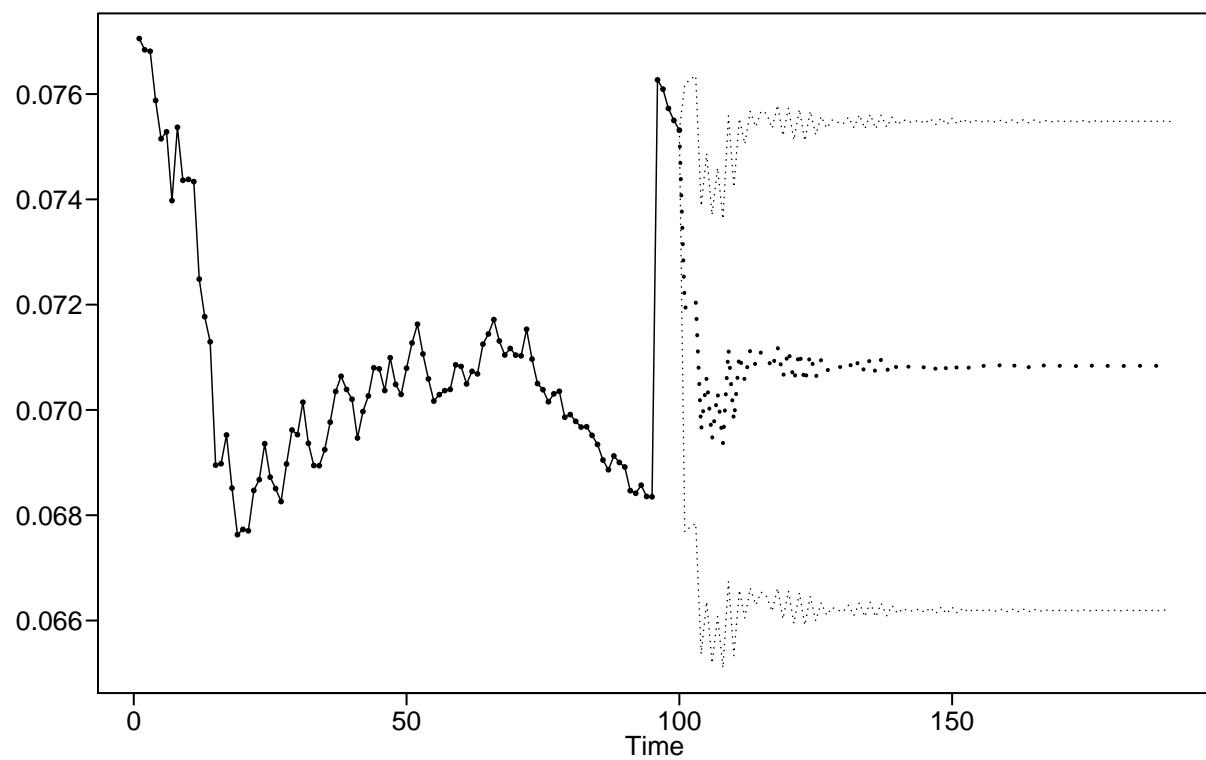


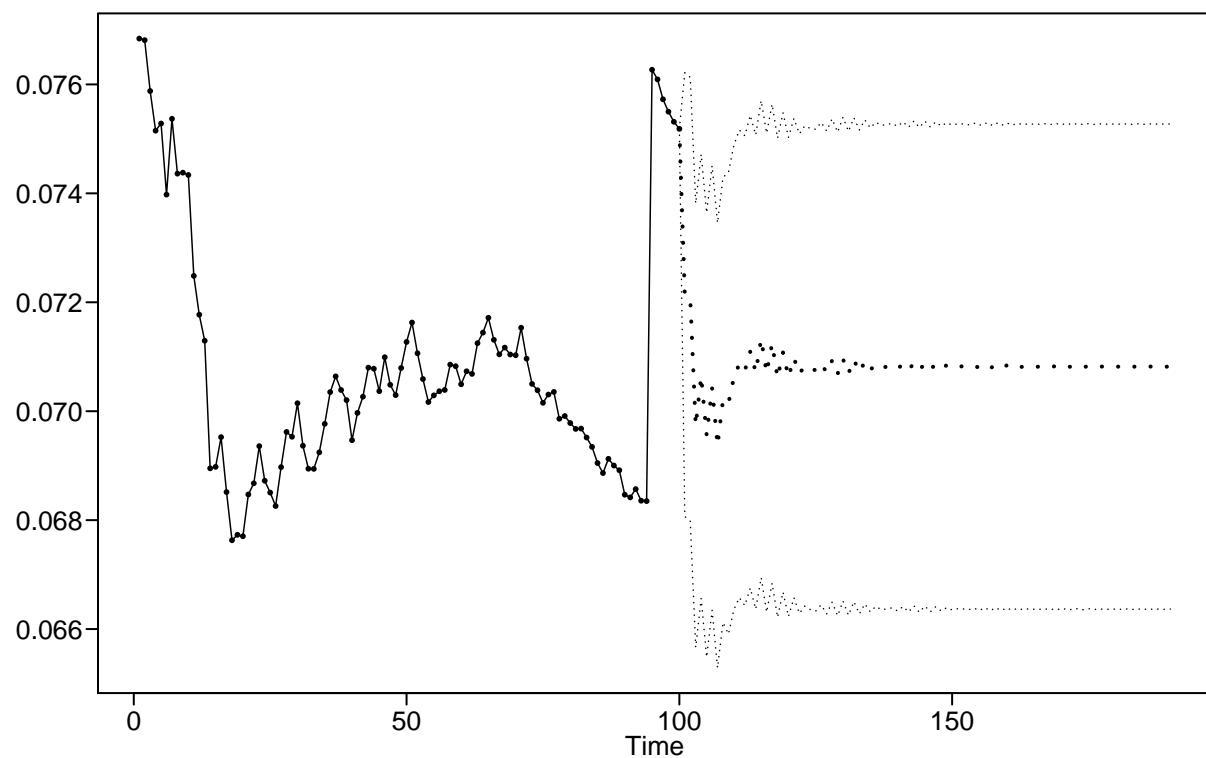


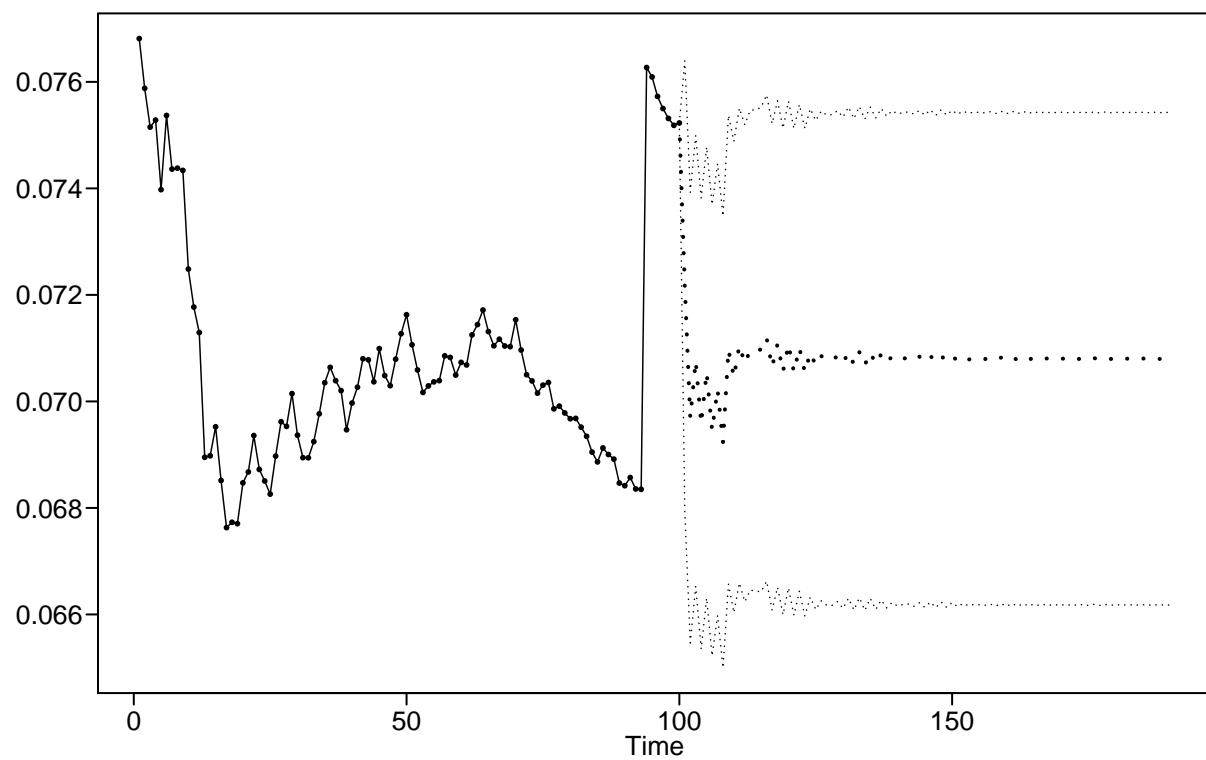


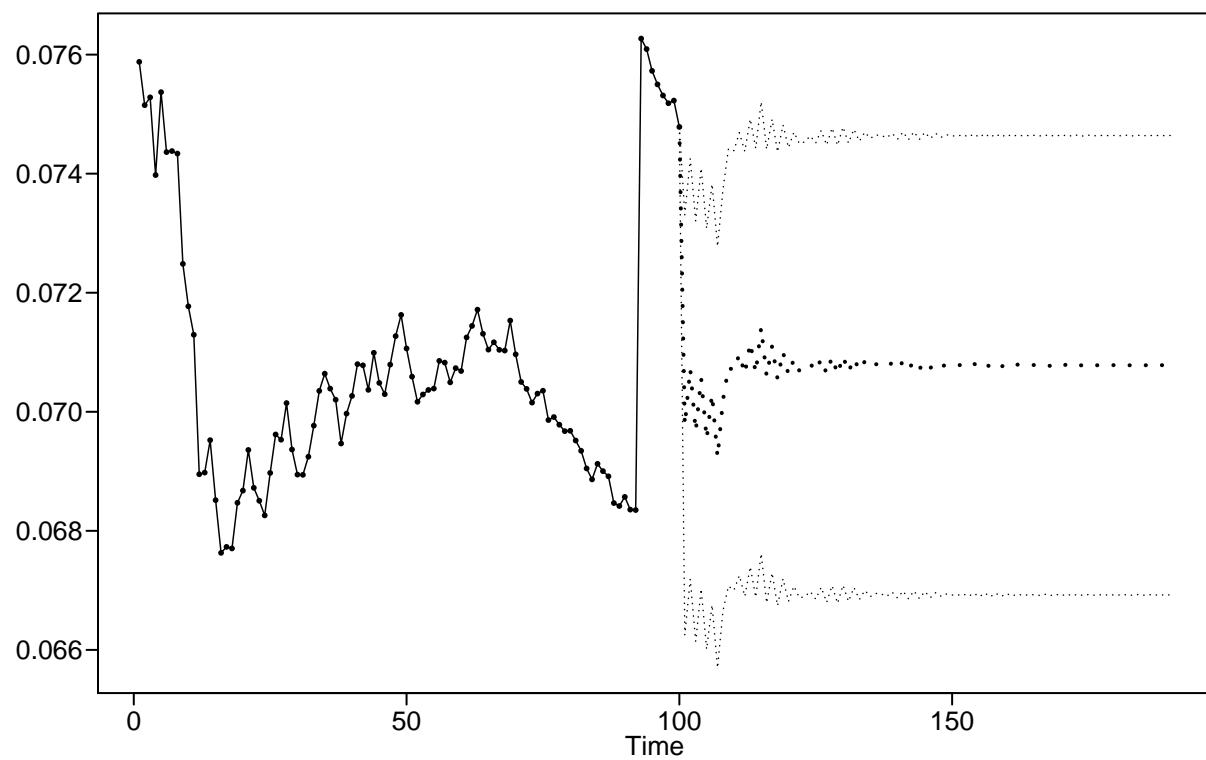


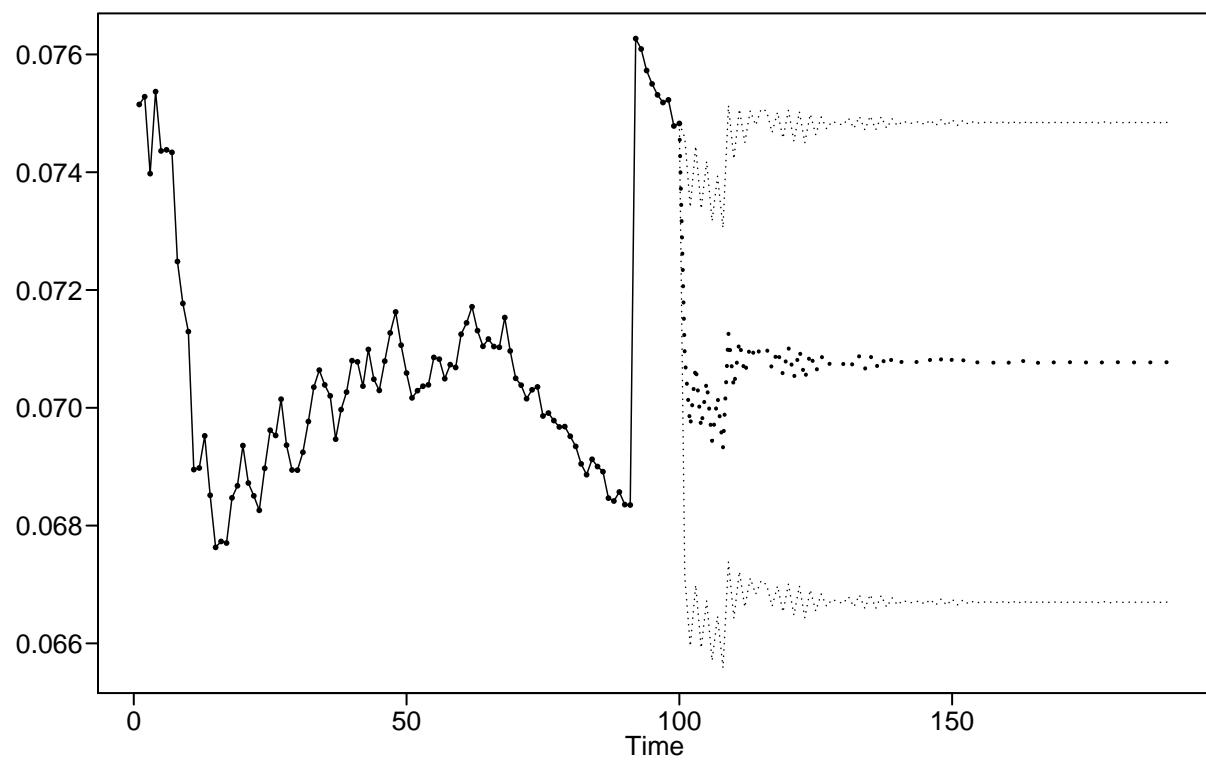


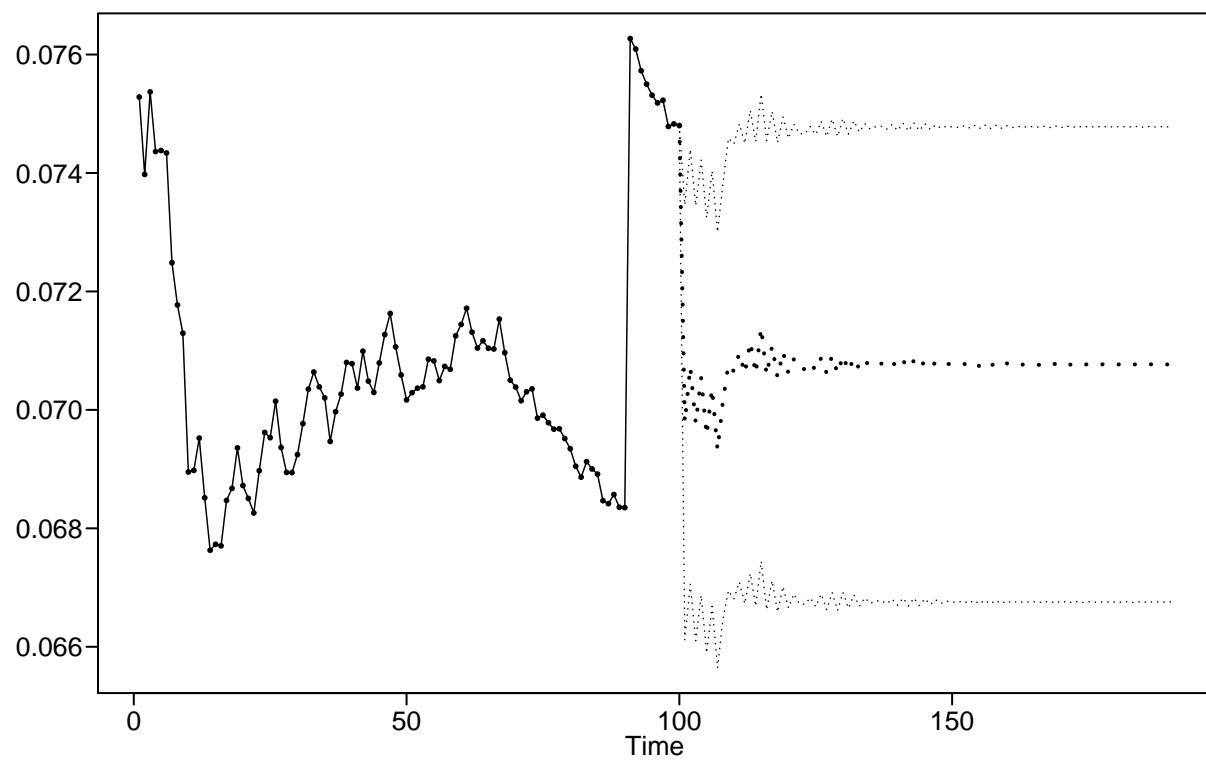


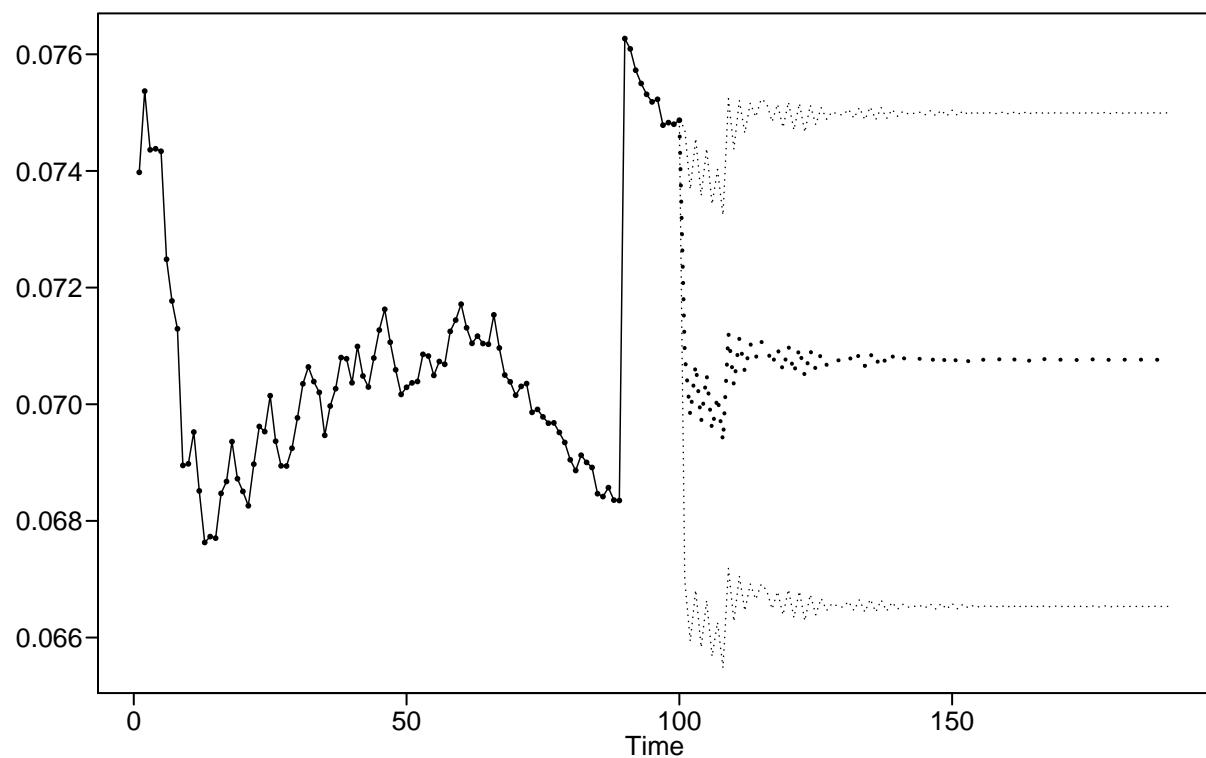


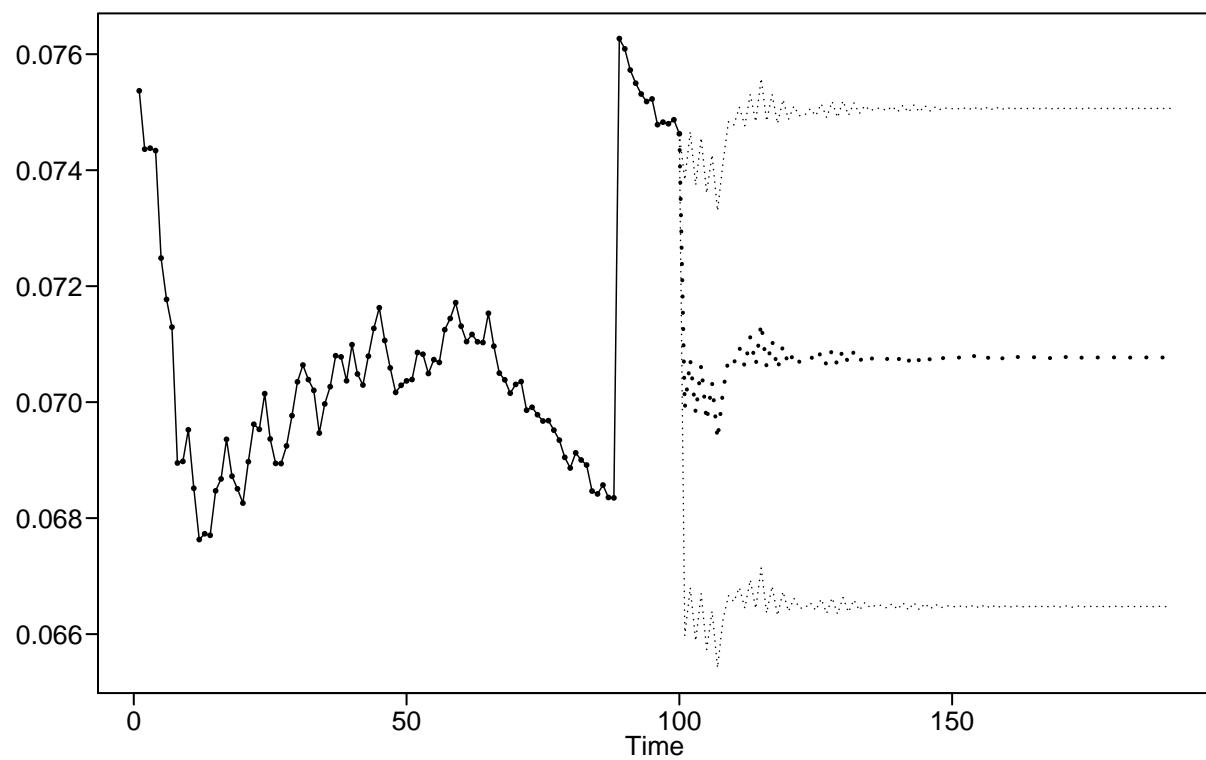


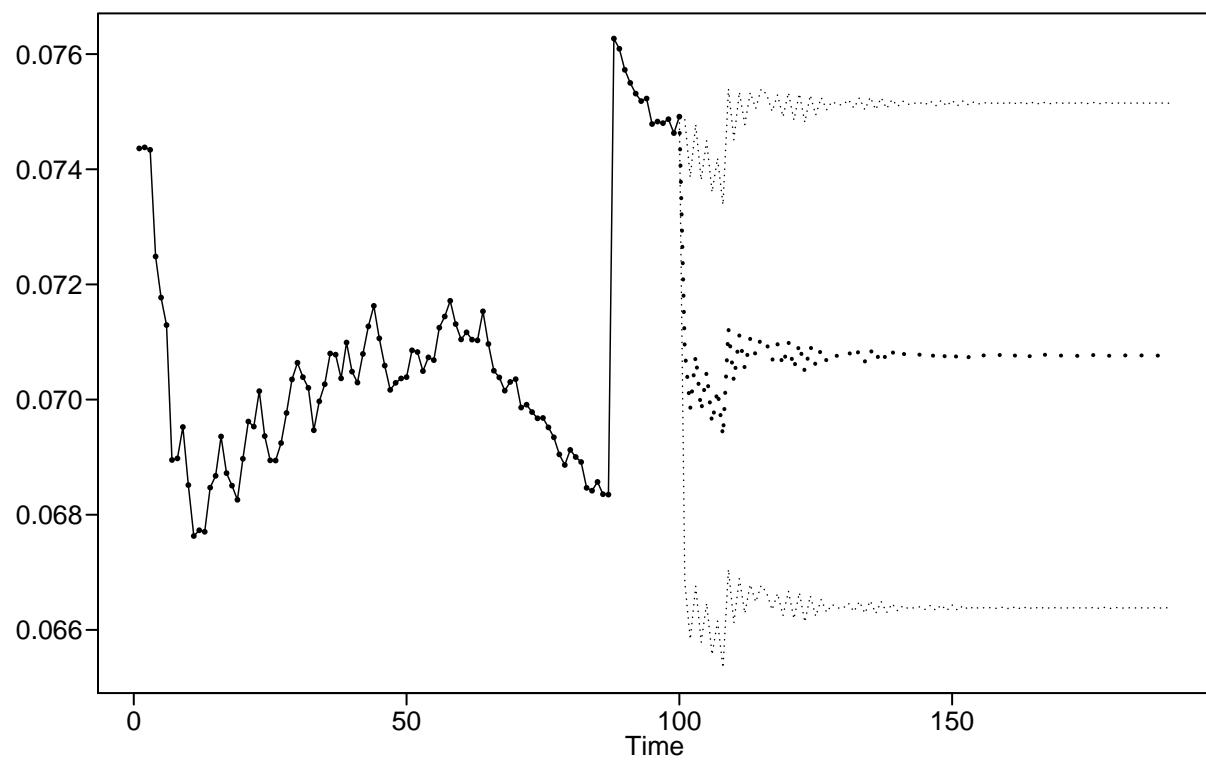


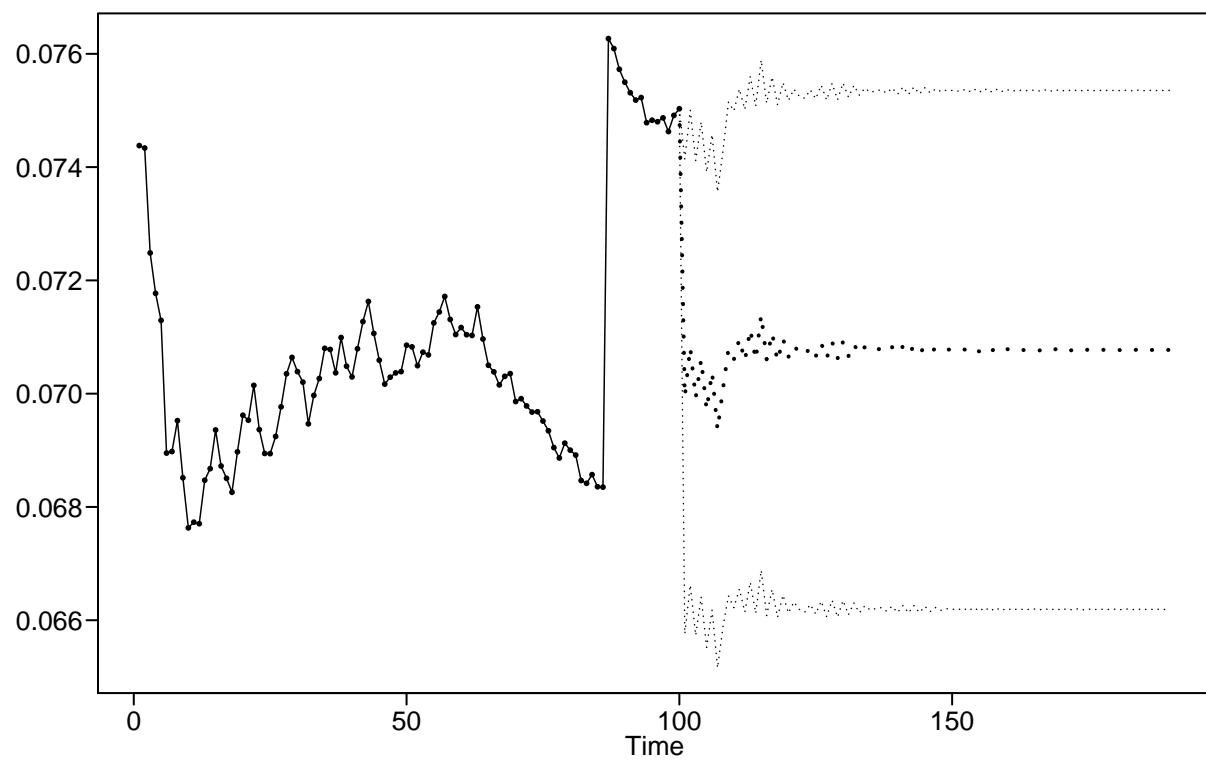


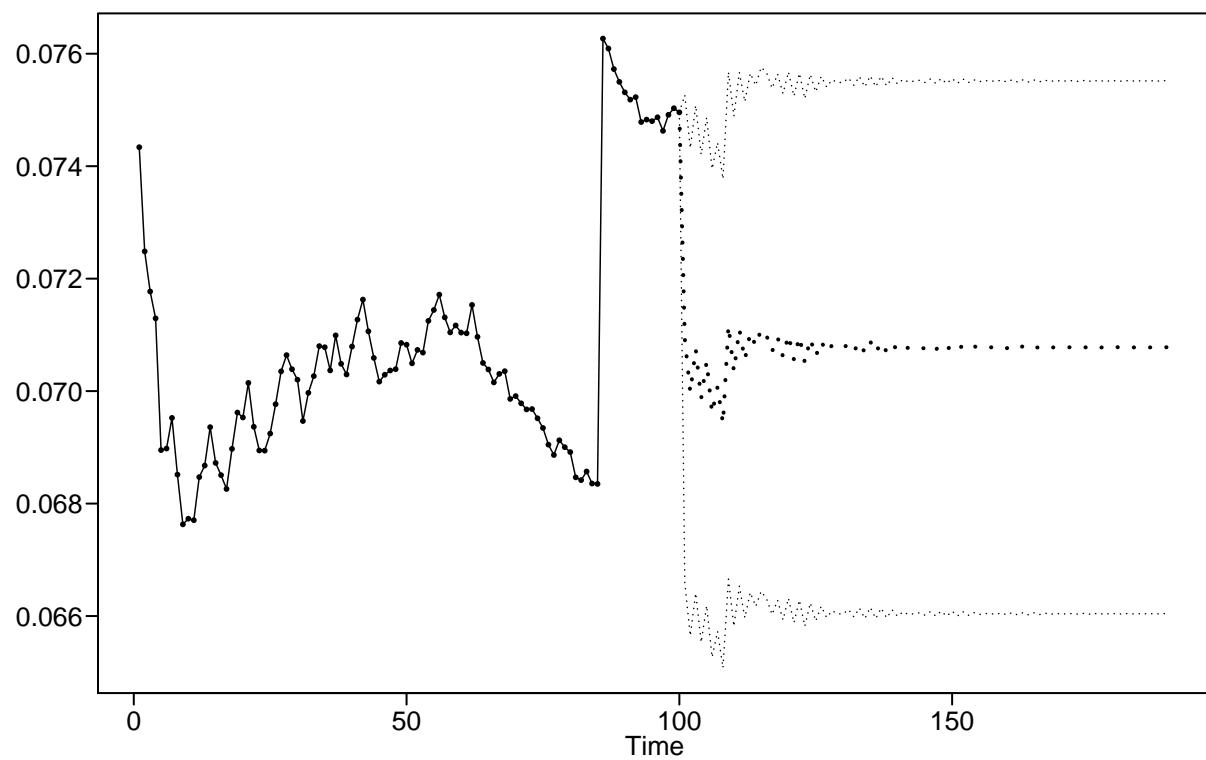


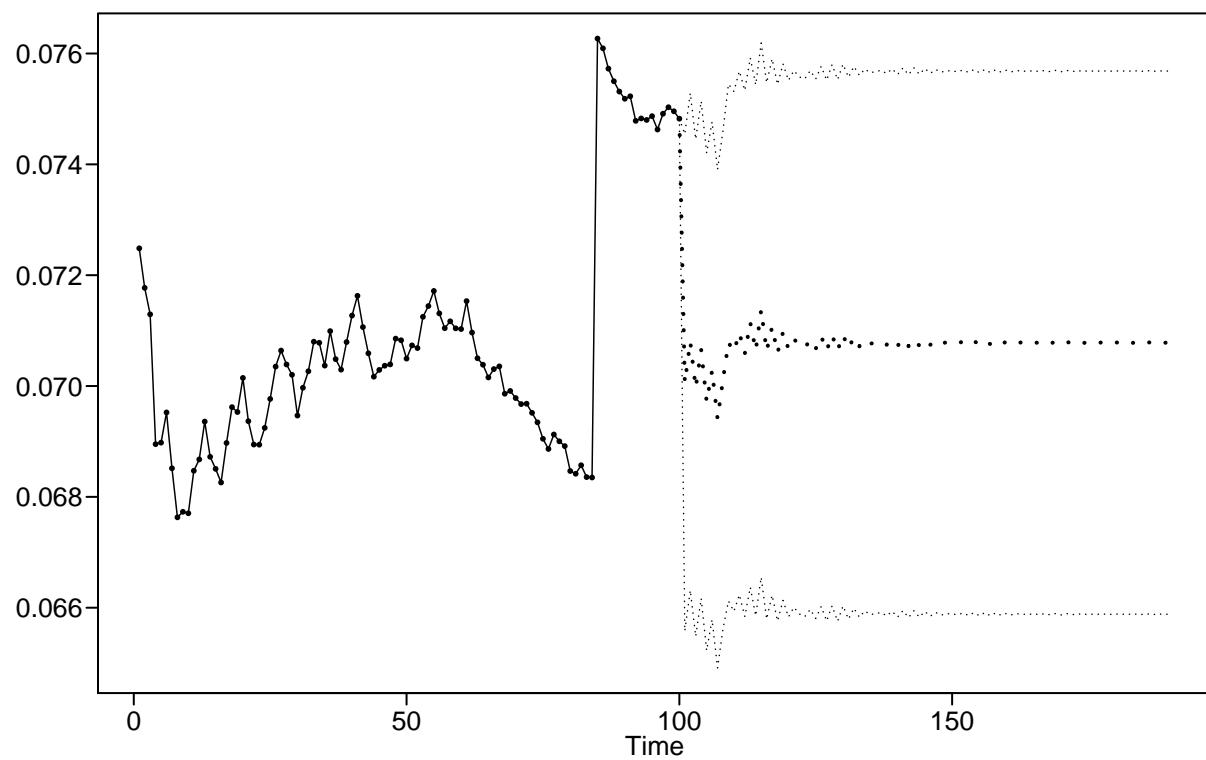






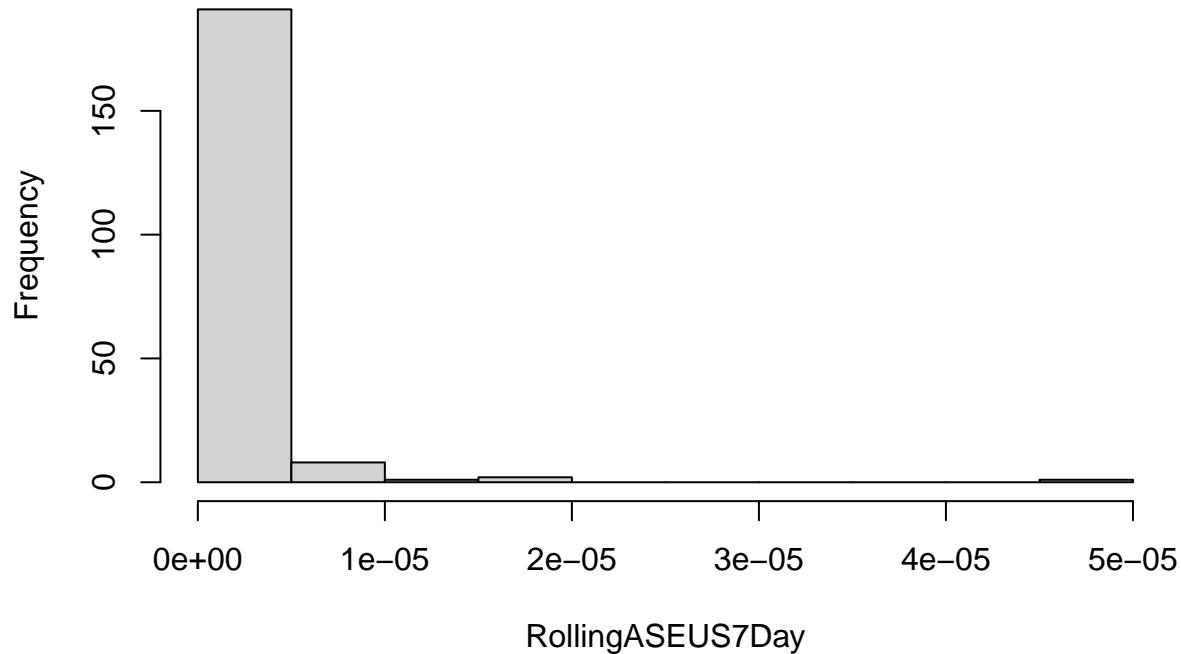






```
modelName = "NC Model - 90 Days"
GetRollingWindowASEInfo(modelName, RollingASENC90Day)
```

Histogram of NC Model – 90 Days Windowed ASE



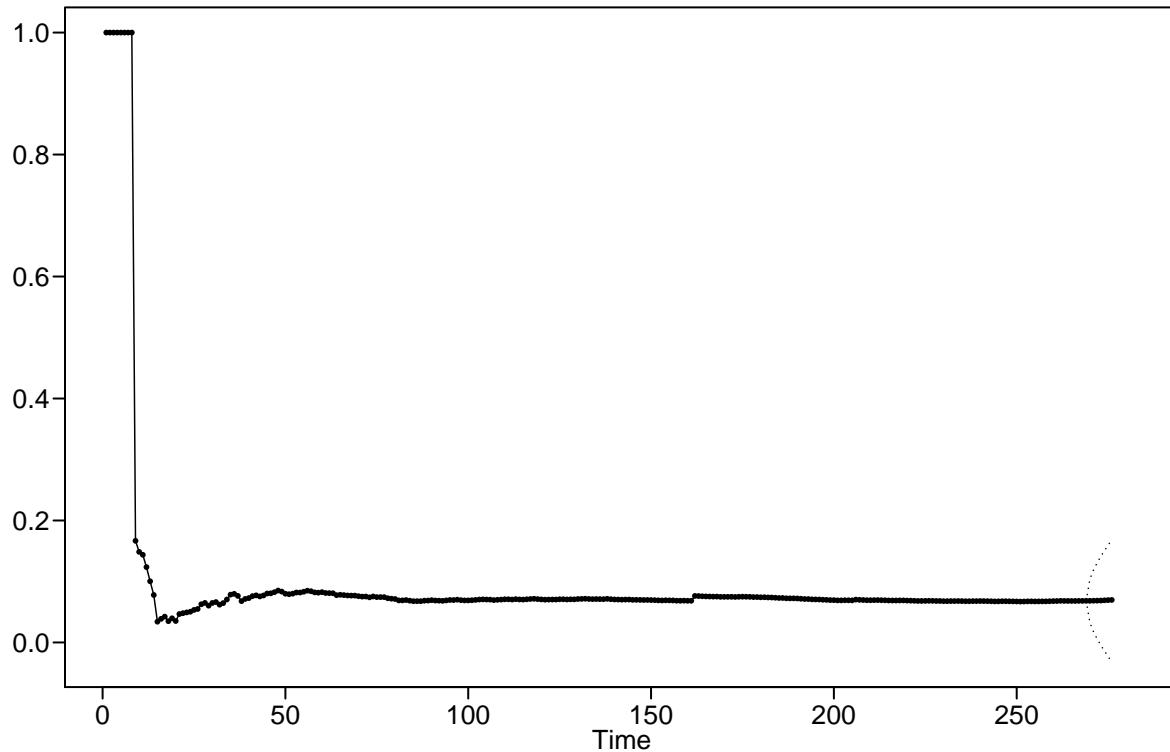
```
## [1] "NC Model - 90 Days Windowed ASE Mean: 0.000256546233646287"  
## [1] "NC Model - 90 Days Windowed ASE Median: 8.66800404447309e-06"
```

The rolling window ASEs are as follows:

- Rolling Window ASE - 7 Day: 1.95119e-06
- Rolling Window ASE - 90 Day: 0.000256

Now the individual forecasts for the short term and long term forecast horizons can be evaluated. The first evaluation will be the short term horizon and how well the forecast has been estimated.

```
#forecast 7 days ahead  
fore.NC7 = fore.aruma.wge(df_NC_Percent_Positive$Percent_Positive, phi=NC.est$phi, theta=NC.est$theta, d=NC.est$d)
```



In evaluating the forecasts above, it is difficult to tell from the full forecast just how well the forecast estimations are. The confidence intervals on the forecast are wide. In order to get a better view of the forecast compared to the original the plot below will provide better indications.

```

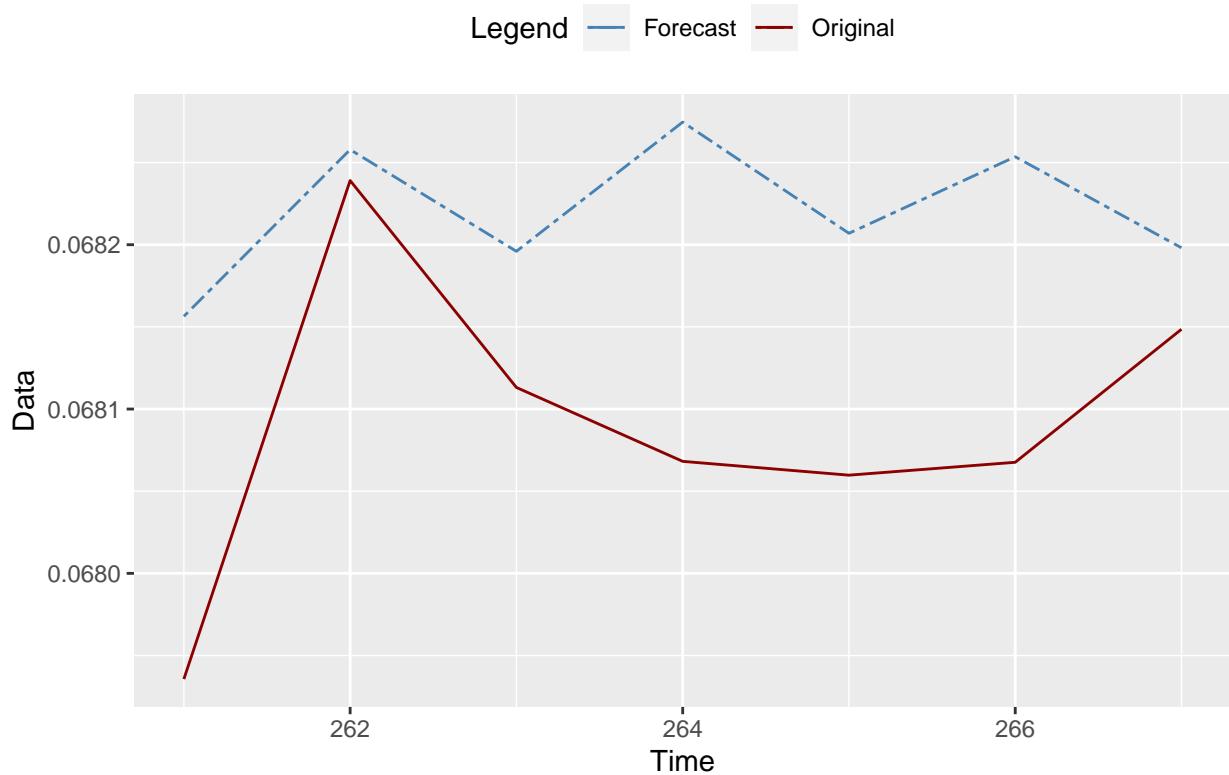
startPoint = length(df_NC_Final$Percent_Positive) - 6
endPoint = length(df_NC_Final$Percent_Positive)

tl = seq(startPoint,endPoint)
orig = df_NC_Final$Percent_Positive[startPoint:endPoint]
forecast = fore.NC7$f
df_NC7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

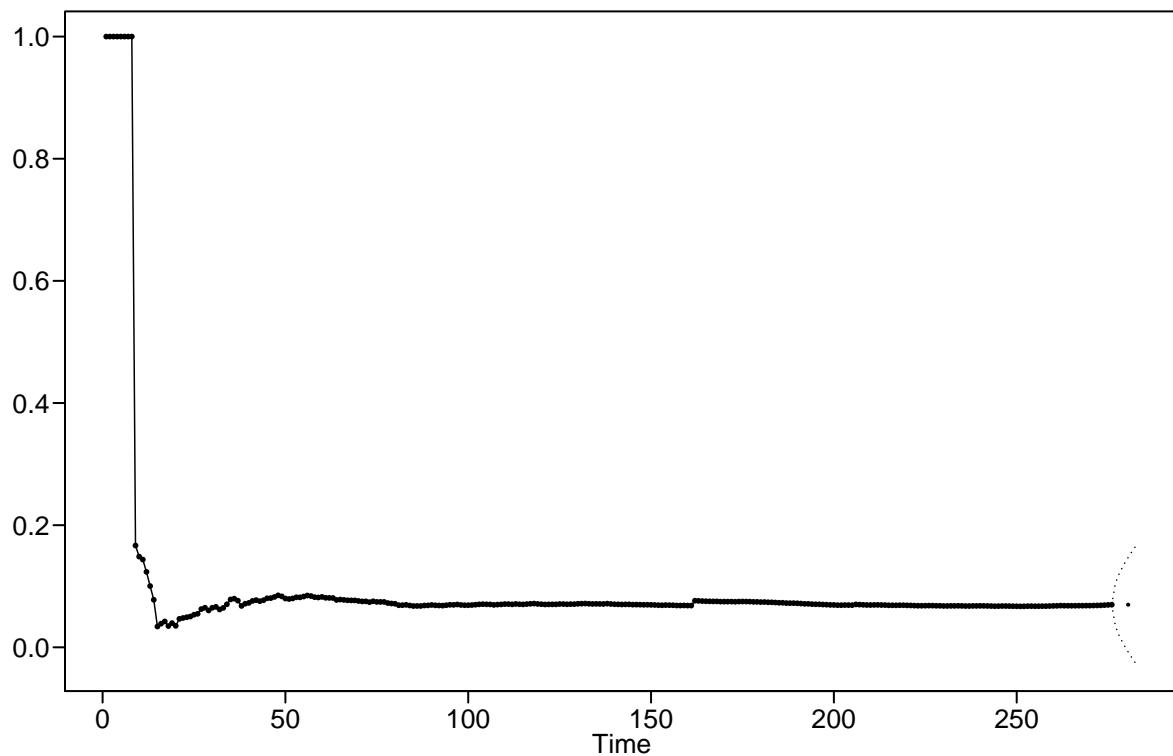
ggplot(df_NC7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Percent Positive: Original vs. Forecast - 7 Day Horizon") +
  scale_color_manual(values = colors)
  
```

NC Percent Positive: Original vs. Forecast – 7 Day Horizon



In comparing the 7 day forecast to the original, the forecast trend is higher than the original trend. The forecast does seem to rise as the original data rises and falls when the original data falls, however toward the end of the dataset the forecast and original data deviate. The original data begins to trend up while the forecast data begins to trend down. Leveraging the forecast to project ahead on the short term horizon (7-days), the plot below shows that there still may be a high forecast especially with the large confidence interval bands that are produced.

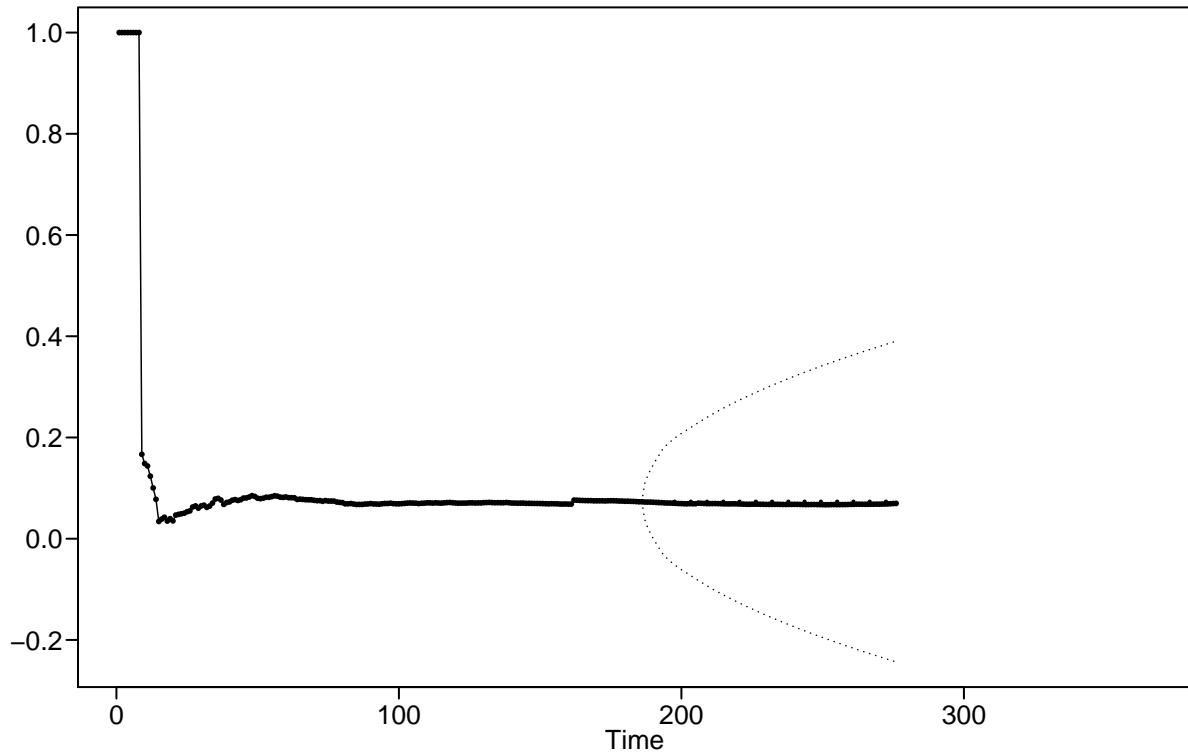
```
#forecast 7 days ahead
fore.NC7 = fore.aruma.wge(df_NC_Percent_Positive$Percent_Positive, phi=NC.est$phi, theta=NC.est$theta, d=NC.est$d, h=7)
```



Moving to the long term horizon (90-day) the same approach will be taken. First the comparison of the forecast to the original and then a projected forecast of the data. From the comparison of forecast data to original data the trends seem to be that the long term forecast is close to the original data. The confidence intervals are still wide which leaves some room for changes.

```
#forecast 90 days (3 months) ahead
```

```
fore.NC90 = fore.aruma.wge(df_NC_Percent_Positive$Percent_Positive, phi=NC.est$phi, theta=NC.est$theta, d=1)
```



Zooming into the forecast projection, below, the forecast trend is still higher than the original data. The forecast also seems to be at a mean value of about 0.08 which would be expected with the model type chosen. The overall trend of the forecast matches the original data however the increase above the original data may be a concern.

```

startPoint = length(df_NC_Percent_Positive$Percent_Positive)-89
endPoint = length(df_NC_Percent_Positive$Percent_Positive)

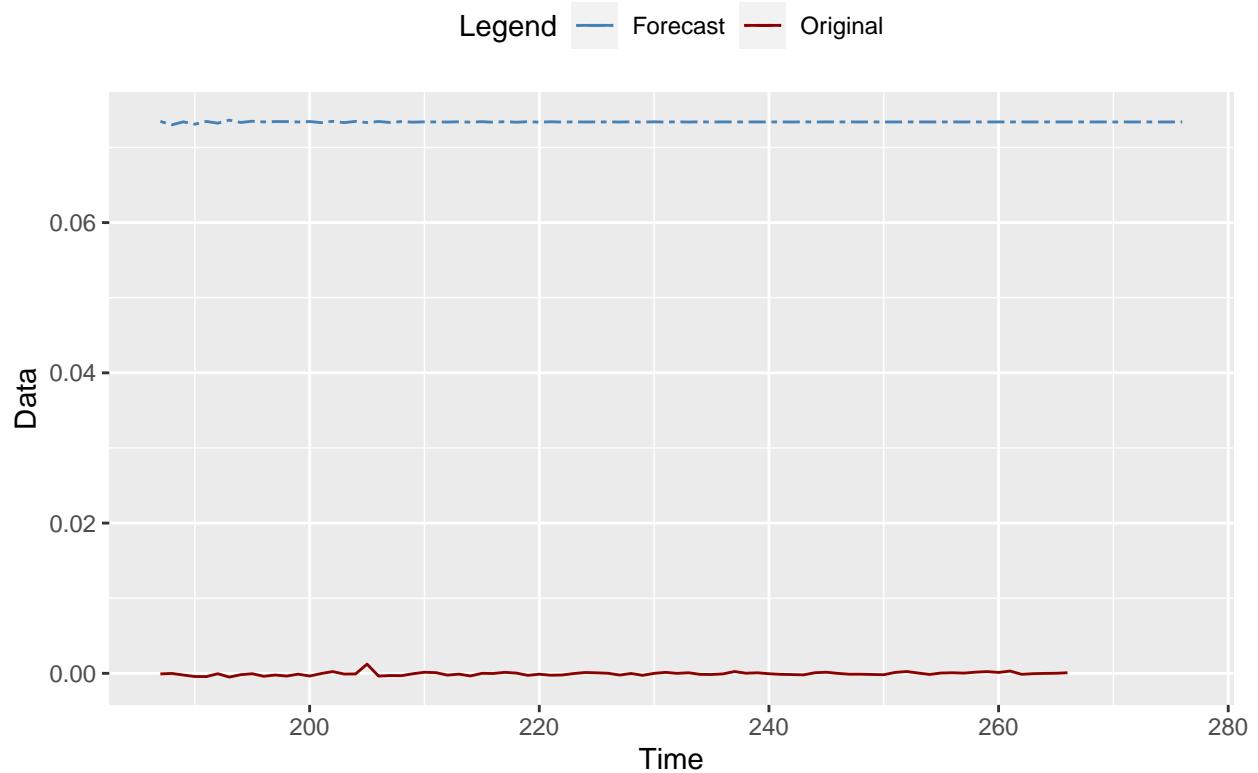
tl = seq(startPoint,endPoint)
orig = dfNCCasesDiff[startPoint:endPoint]
forecast = fore.NC90$f
df_NC90 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_NC90,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Percent Positive: Original vs. Forecast - 90 Day Horizon") +
  scale_color_manual(values = colors)

```

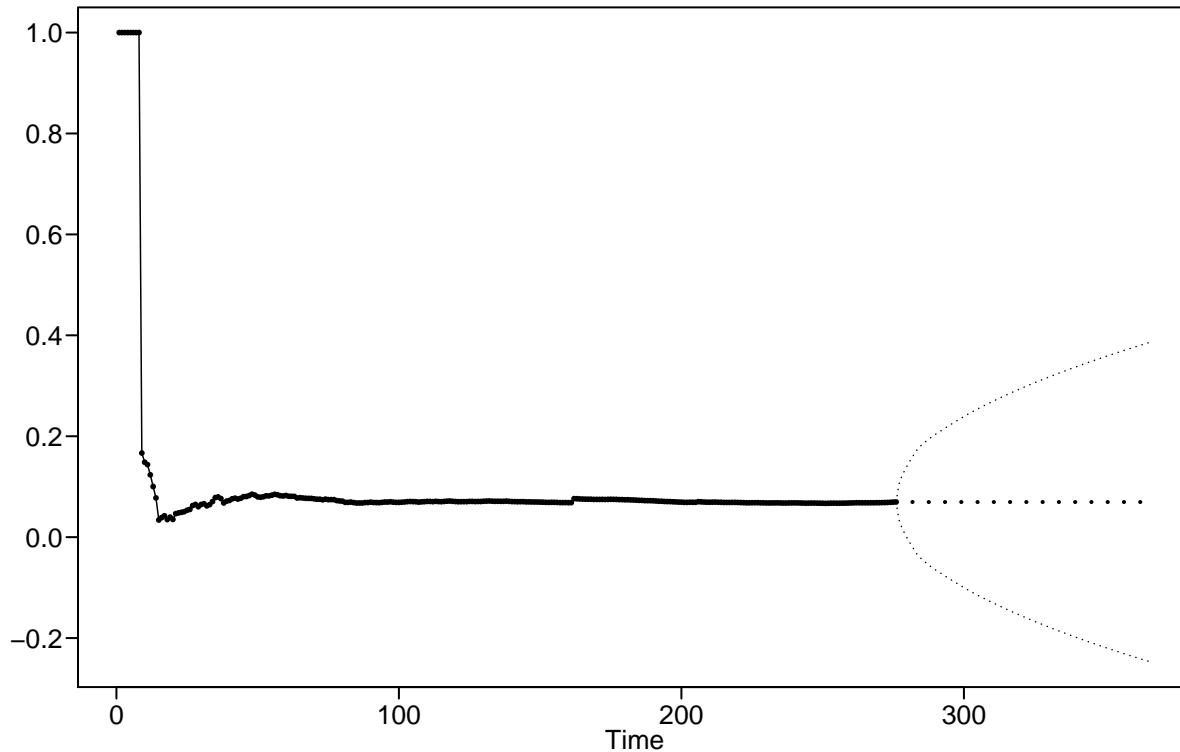
NC Percent Positive: Original vs. Forecast – 90 Day Horizon



As shown below, the projected forecast seems maintain the flat trend in the out time periods. The confidence interval bands do get wider as time progresses thus leaving room for further deviation.

```
#forecast 90 days (3 months) ahead
```

```
fore.NC90 = fore.aruma.wge(df_NC_Percent_Positive$Percent_Positive,phi=NC.est$phi,theta=NC.est$theta,d=
```



Multi-Layer Perceptron (MLP)

Another time series modeling approach that can be taken is to leverage a neural network, also called a Multi-Layer Perceptron (MLP). Through this model an input layer is defined. Each input layer is connected to a “hidden layer”. The connection between the input layer and the “hidden layer” contains a weighting. The “hidden layer” is used to approximate any continuous function. Within the MLP model there can be one or more hidden layers. The “hidden layers” then all connect to an output layer. The output layer provides the final result of the model.

As was done for the ARIMA modeling, the MLP modeling will be very similar. First the model will be evaluated. The model will be set to perform 50 repetitions of the MLP. Each of these repetition results will be combined based on the mean calculation. In the case of the MLP, the models will be executed twice. Once to see evaluate the model on the base data and secondly to take into account the differencing that was seen in the initial ARIMA evaluation.

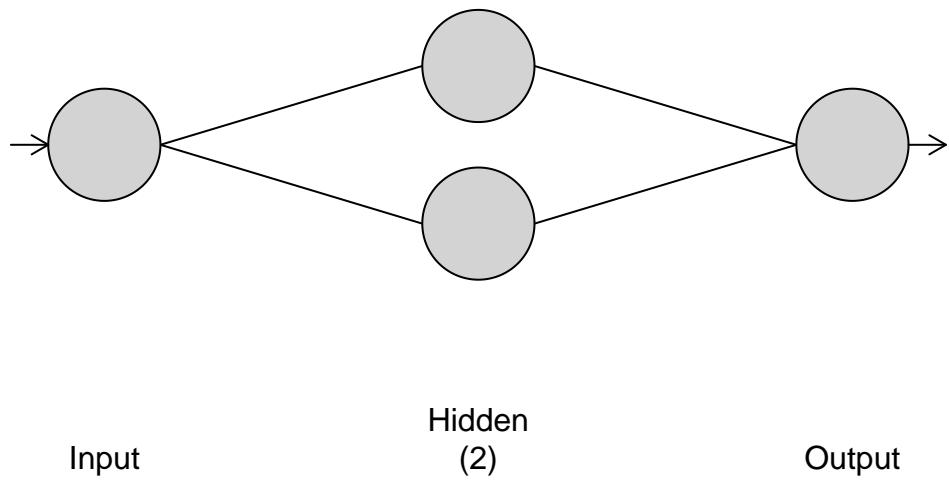
Evaluating the model with the base data results in the following:

```
set.seed(100)
fit.mlp.NC = mlp(ts(df_NC_Final$Percent_Positive), reps = NN_REPS, comb = "mean", hd.auto.type="cv")
fore.mlp.NC7 = forecast(fit.mlp.US, h = SHORT_TERM_FORECAST_HORIZON)
fore.mlp.NC90 = forecast(fit.mlp.US, h = LONG_TERM_FORECAST_HORIZON)
```

The visual below represents the MLP that was leveraged for the analysis. The MLP model has one (1) input layers, two (2) hidden layers and an output layer.

```
plot(fit.mlp.NC)
```

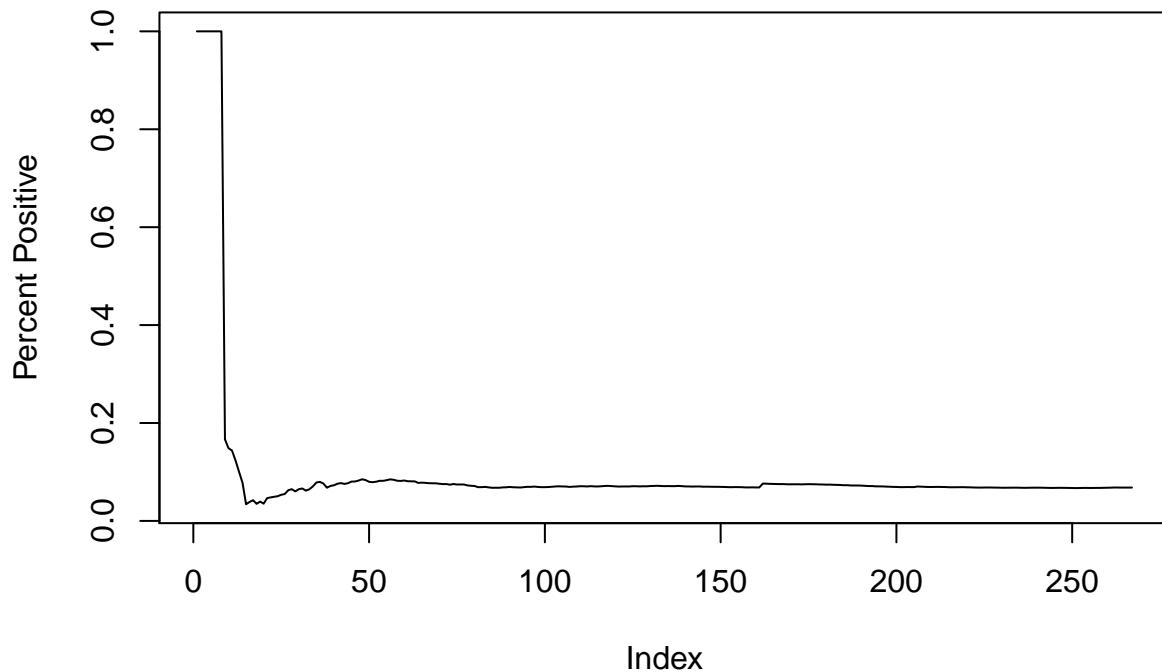
MLP



As a reminder, the plot below shows the current realization of the percent positive data for the US.

```
plot(df_NC_Final$Percent_Positive, type = "l", ylab="Percent Positive", main="Percent Positive - Total Positive")
```

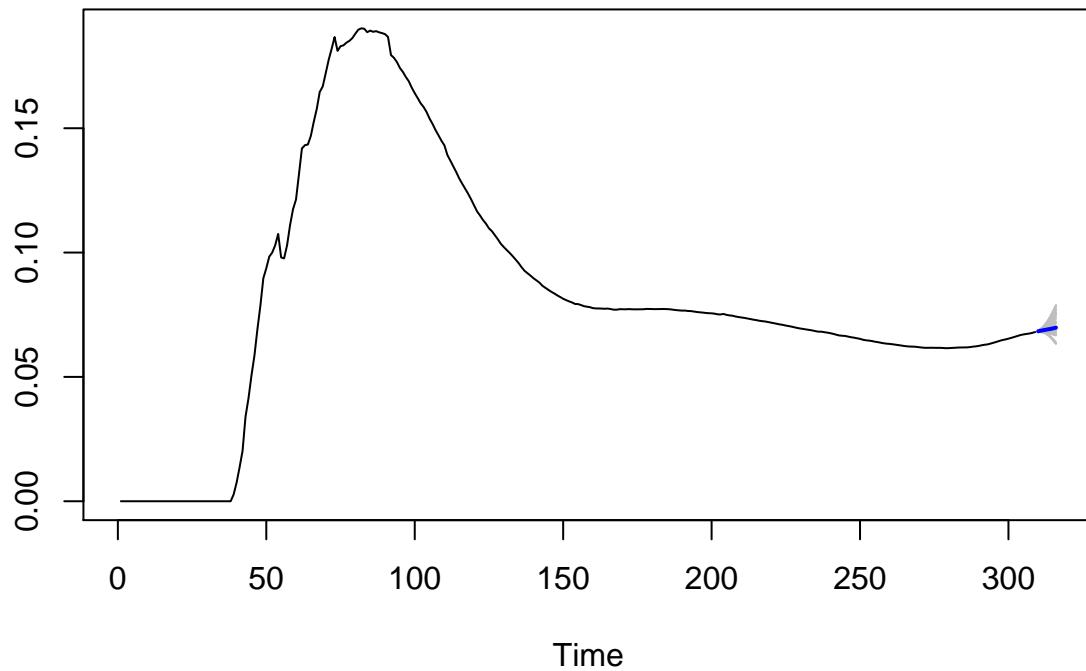
Percent Positive – Total NC



The plot below shows the 7-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow the curve fairly well.

```
plot(fore.mlp.NC7, ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total US")
```

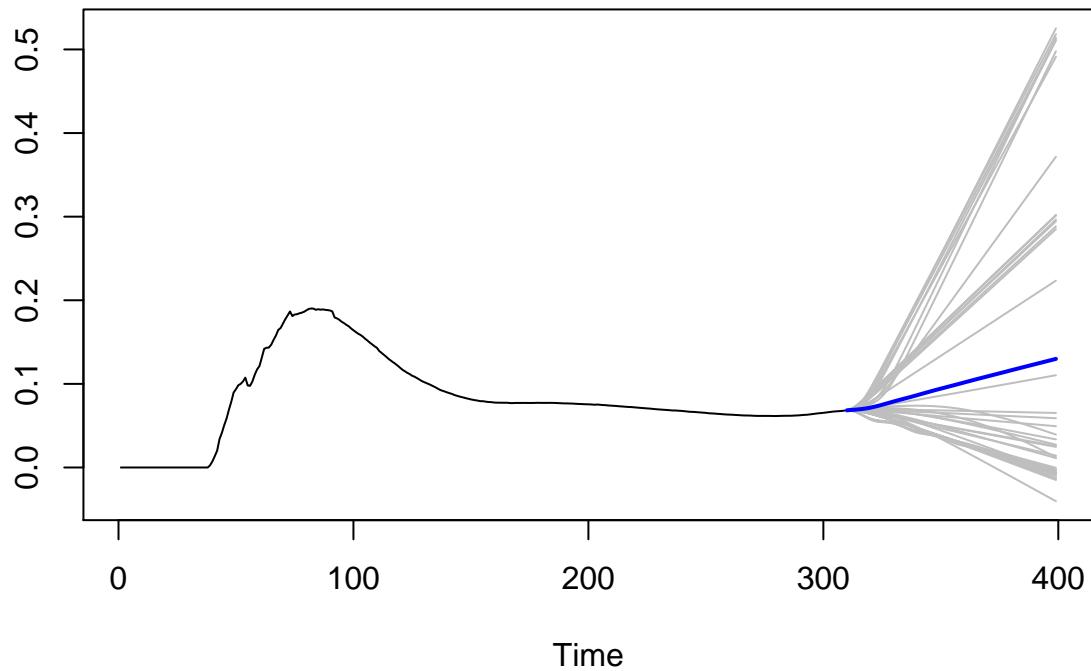
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to rise sharply versus following the trend.

```
plot(fore.mlp.NC90,ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total NC")
```

Forecasts from MLP

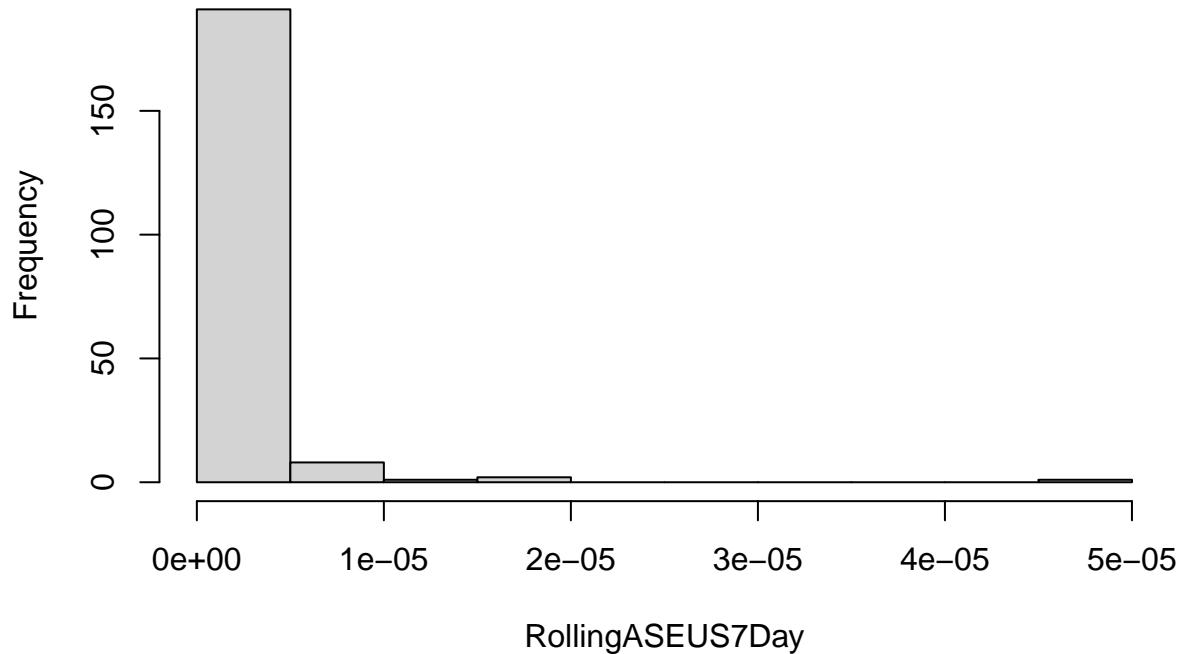


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcNCMLP7Day = RollingASECalcMLP(fit.mlp.NC$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT_7_DAY)
RollingASECalcNCMLP90Day = RollingASECalcMLP(fit.mlp.NC$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG_90_DAY)
```

```
modelName = "NC Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcNCMLP7Day)
```

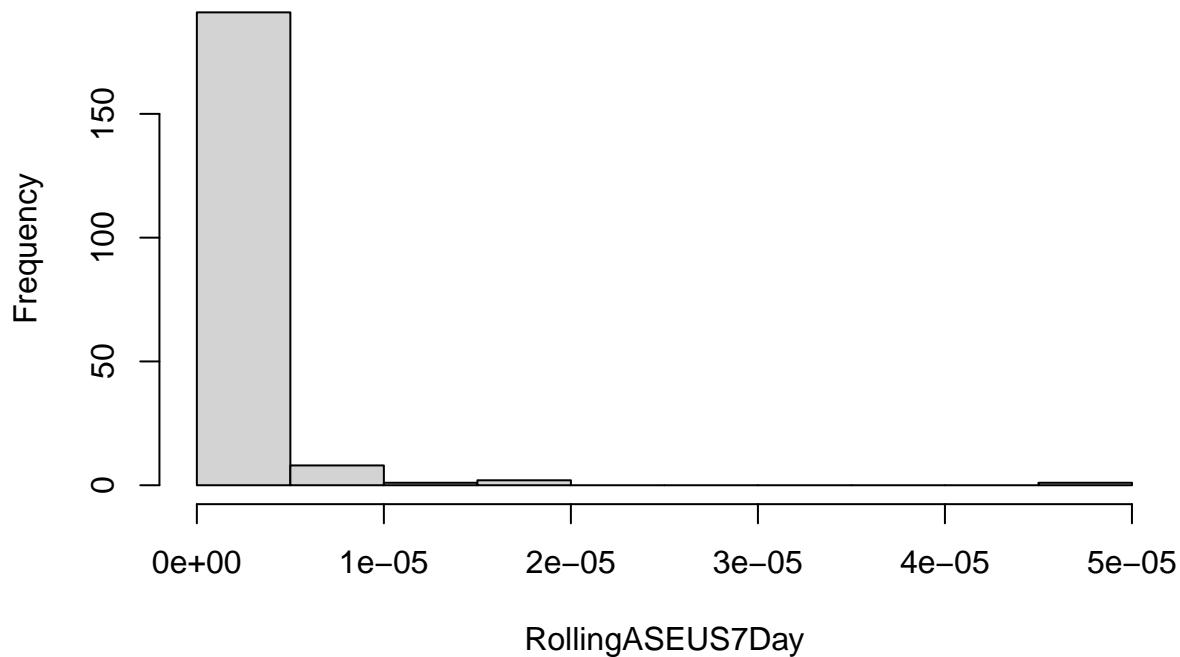
Histogram of NC Model – 7 Days Windowed ASE



```
## [1] "NC Model - 7 Days Windowed ASE Mean: 1.71739831045227e-06"  
## [1] "NC Model - 7 Days Windowed ASE Median: 2.56804026802787e-07"
```

```
modelName = "NC Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcNCMLP90Day)
```

Histogram of NC Model – 90 Days Windowed ASE



```
## [1] "NC Model - 90 Days Windowed ASE Mean: 1.30737558057811e-05"  
## [1] "NC Model - 90 Days Windowed ASE Median: 7.4961861399615e-06"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 1.717398e-06
- Rolling Window ASE - 90 Day: 1.30737e-05

In comparison to the ARIMA(9,1,2) model that was developed these ASE values are much lower.

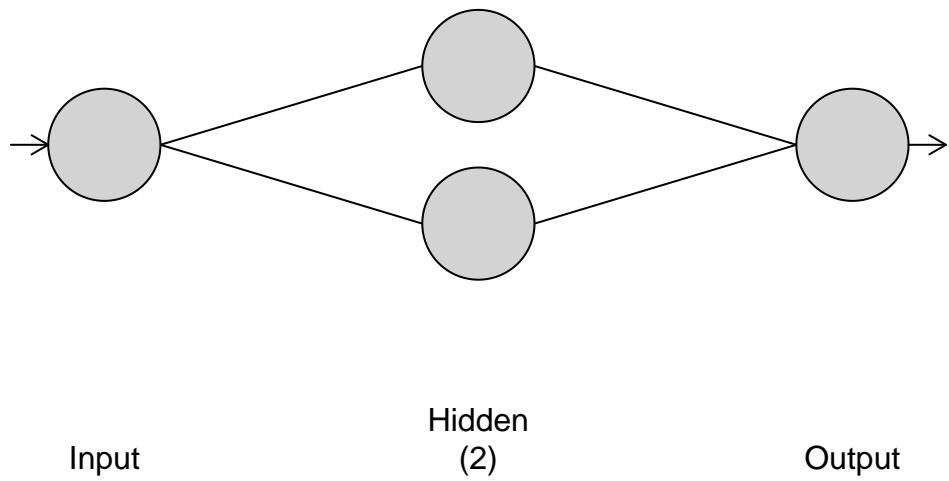
MLP Analysis - Differenced Data The MLP model will now be executed taking into account the differencing that was seen through the ARIMA analysis.

```
set.seed(100)  
fit.mlp.NC = mlp(ts(df_NC_Final$Percent_Positive), reps = NN_REPS, comb = "mean", hd.auto.type="cv", diff=1)  
fore.mlp.NC7Diff = forecast(fit.mlp.NC, h = SHORT_TERM_FORECAST_HORIZON)  
fore.mlp.NC90Diff = forecast(fit.mlp.NC, h = LONG_TERM_FORECAST_HORIZON)
```

The visual below represents the MLP that was leveraged for the analysis. The MLP model has one(1) input layers, two (2) hidden layers and an output layer.

```
plot(fit.mlp.NC)
```

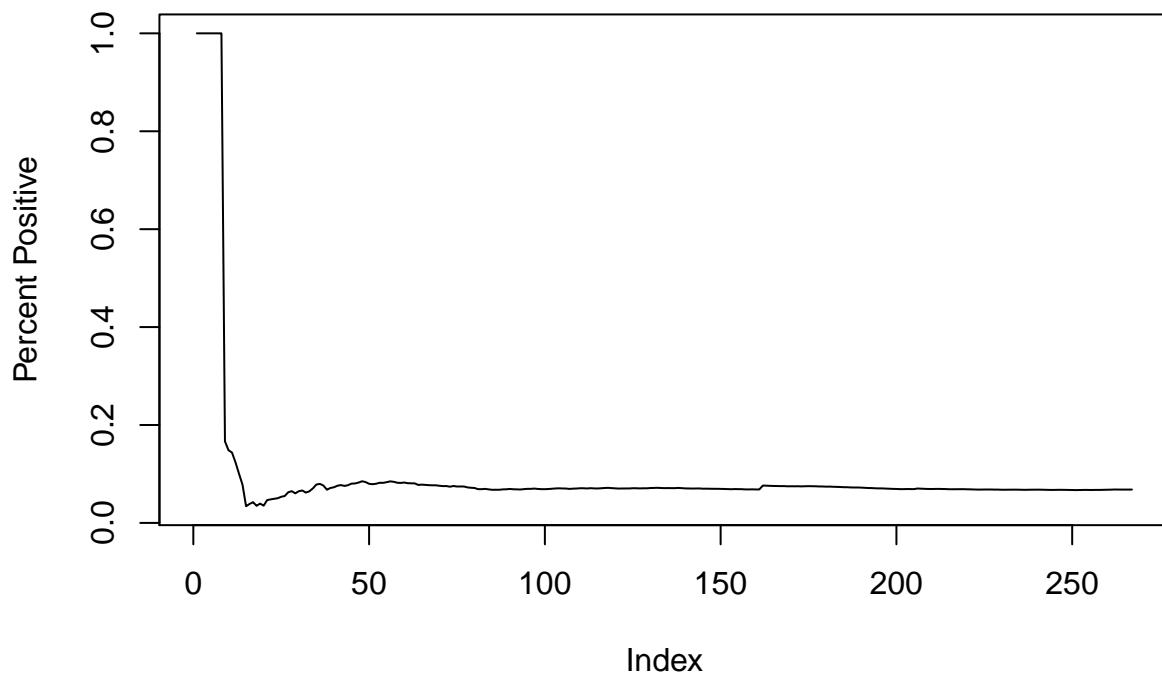
MLP



As a reminder, the plot below shows the current realization of the percent positive data for NC.

```
plot(df_NC_Final$Percent_Positive, type = "l", ylab="Percent Positive", main="Percent Positive - Total Positive")
```

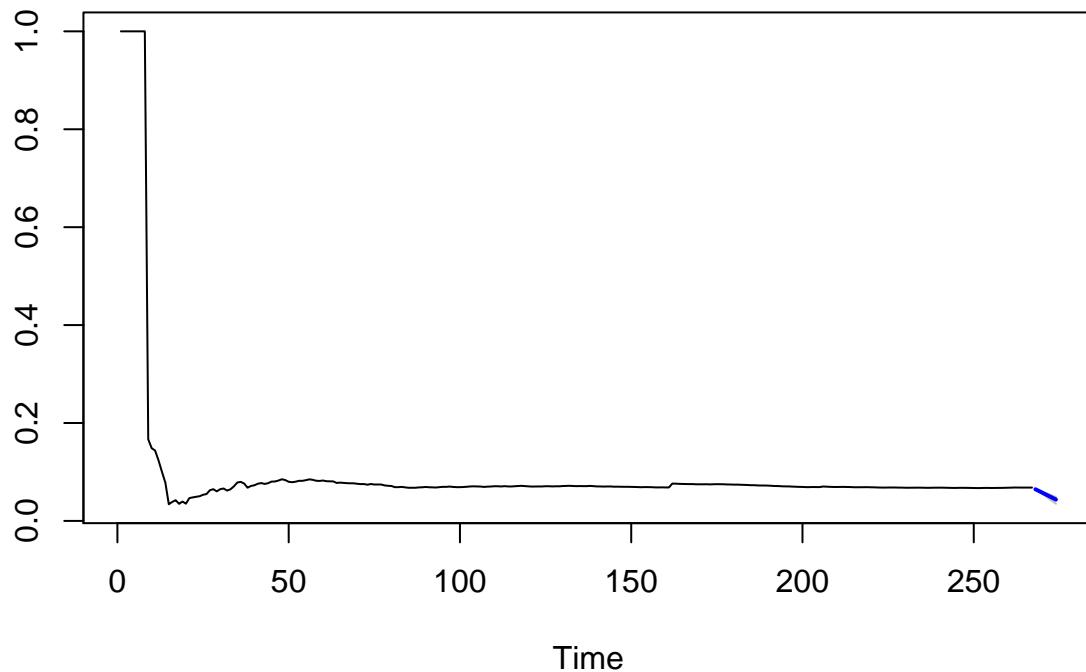
Percent Positive – Total NC



The plot below shows the 7-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to drop off sharply.

```
plot(fore.mlp.NC7Diff, ylab="Percent Positive", main="Percent Positive - 7 Day Forecast - Total NC")
```

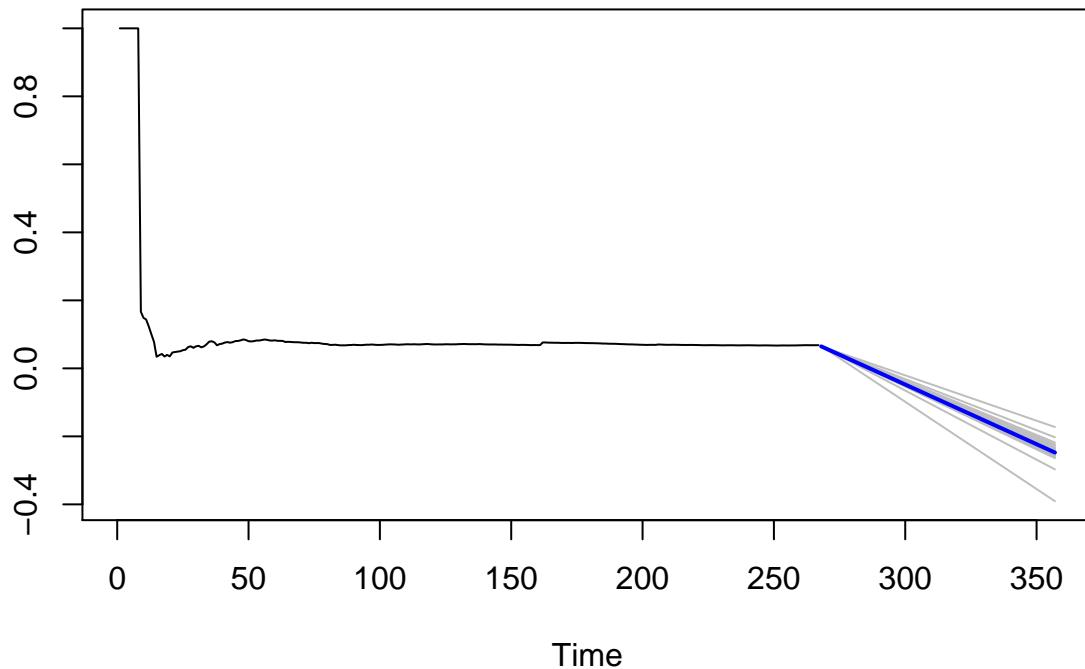
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow a more downward trajectory than the direction the original realization is moving toward.

```
plot(fore.mlp.NC90Diff, ylab="Percent Positive", main="Percent Positive - 90 Day Forecast - Total NC")
```

Forecasts from MLP

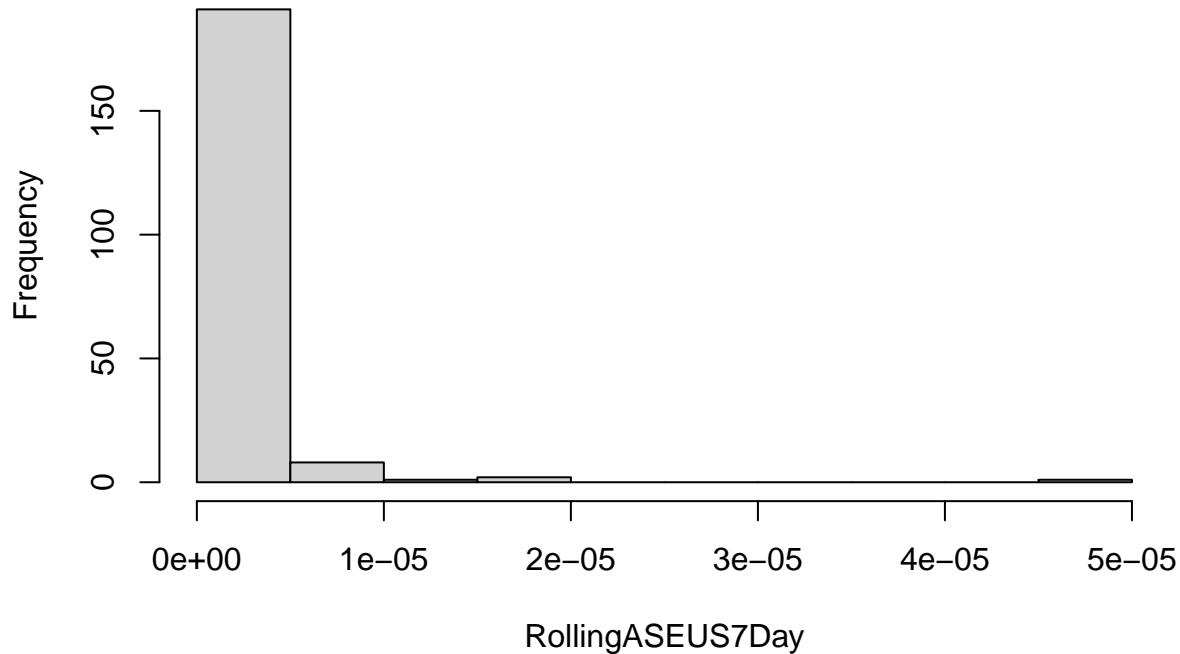


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcNCMLP7Day = RollingASECalcMLP(fit.mlp.NC$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT)
RollingASECalcNCMLP90Day = RollingASECalcMLP(fit.mlp.NC$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG)
```

```
modelName = "NC Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcNCMLP7Day)
```

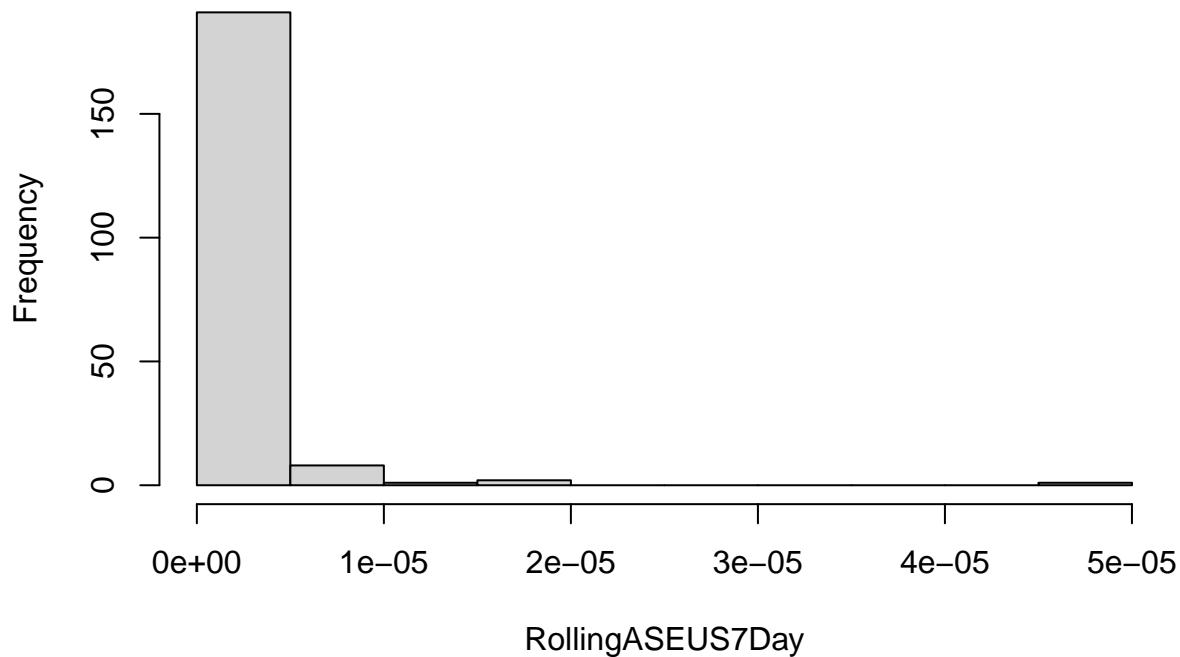
Histogram of NC Model – 7 Days Windowed ASE



```
## [1] "NC Model - 7 Days Windowed ASE Mean: 1.71739831045227e-06"  
## [1] "NC Model - 7 Days Windowed ASE Median: 2.56804026802787e-07"
```

```
modelName = "NC Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcNCMLP90Day)
```

Histogram of NC Model – 90 Days Windowed ASE



```
## [1] "NC Model - 90 Days Windowed ASE Mean: 1.30737558057811e-05"  
## [1] "NC Model - 90 Days Windowed ASE Median: 7.4961861399615e-06"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 1.717398e-06
- Rolling Window ASE - 90 Day: 1.30737e-05

In comparison to the ARIMA(13,2,0) model that was developed these ASE values are much lower. The ASEs are approximately the same as the original MLP model. From a visual perspective the original MLP model seems to forecast the data trend better given that some of the non mean analysis seems to follow the realization better.

Univariate Ensemble Ensemble modeling is a way of combining models to see if the positives of both models will make the forecasts better. In the univariate modeling case the ARIMA(13,2,0) model and the non-differenced MLP model will be combined to see if there is better performance. Ensemble models will be executed for both the short-term and long-term forecasts.

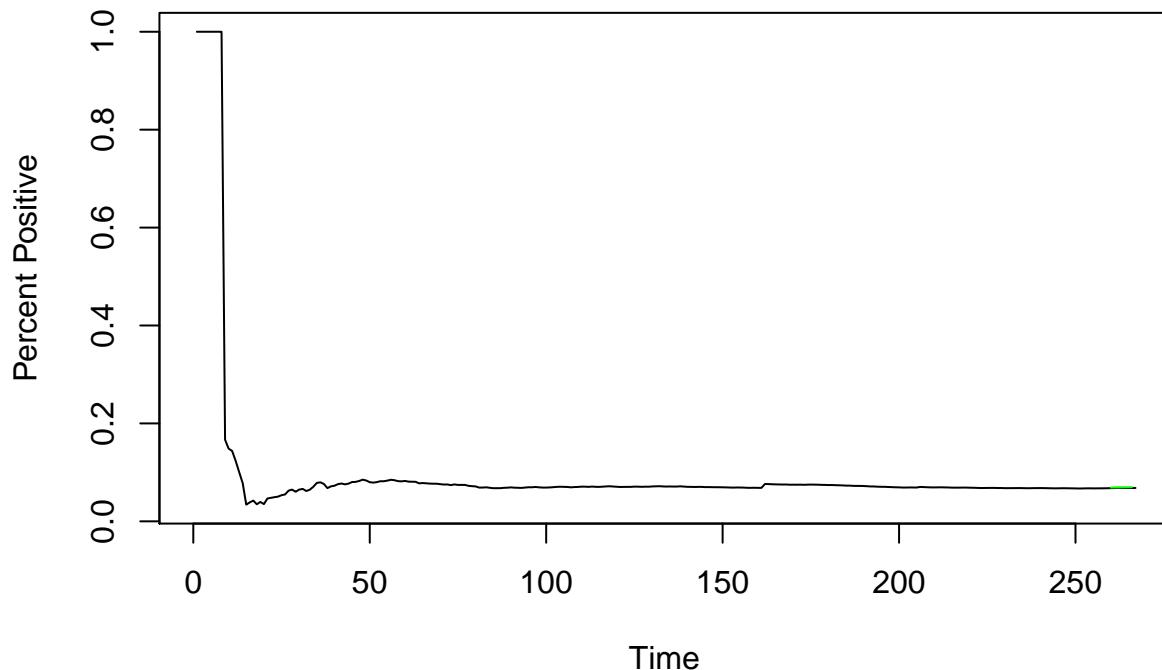
Start with the short-term, 7-day, forecast. Visually, the ensemble seems to predict the 7-day forecast fairly well.

```
startPos = length(df_NC_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON  
endPos = length(df_NC_Final$Percent_Positive)-1
```

```
NCUV7Ensemble = (fore.NC7$f + fore.mlp.NC7$mean)/2
```

```
plot(seq(1,length(df_NC_Final$Percent_Positive),1),df_NC_Final$Percent_Positive, type = "l",xlim = c(1,1),  
lines(seq(startPos,endPos,1), NCUV7Ensemble, type = "l", col = "green")
```

Total NC – Percent Positive – 7 Day Forecast – Ensemble Model



```
NCUV7EnsembleASE = mean((df_NC_Final$Percent_Positive[startPos:endPos] - NCUV7Ensemble)^2)
```

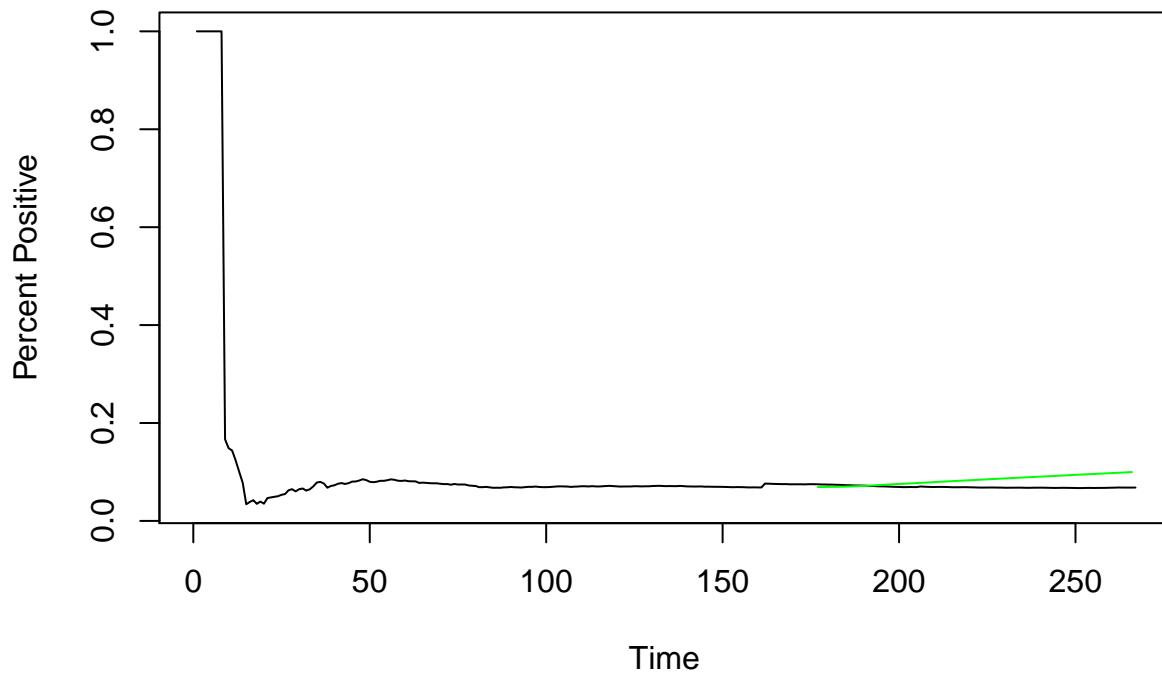
Now move to the long-term, 90 day, forecast. Initially the ensemble model trends a little lower than the original realization, then moves a bit higher than the realization. Visually this does not look like an extremely large over shoot on the forecast.

```
startPos = length(df_NC_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON  
endPos = length(df_NC_Final$Percent_Positive)-1
```

```
NCUV90Ensemble = (fore.NC90$f + fore.mlp.NC90$mean)/2
```

```
plot(seq(1,length(df_NC_Final$Percent_Positive),1),df_NC_Final$Percent_Positive, type = "l",xlim = c(1,1),  
lines(seq(startPos,endPos,1), NCUV90Ensemble, type = "l", col = "green")
```

Total NC – Percent Positive – 90 Day Forecast – Ensemble Model



```
NCUV90EnsembleASE = mean((df_NC_Final$Percent_Positive[startPos:endPos] - NCUV90Ensemble)^2)
```

ASEs for the ensemble models are:

```
print(paste("ASE Ensemble 7-Day: ", NCUV7EnsembleASE))
```

```
## [1] "ASE Ensemble 7-Day: 1.90583076081151e-06"
```

```
print(paste("ASE Ensemble 90-Day: ", NCUV90EnsembleASE))
```

```
## [1] "ASE Ensemble 90-Day: 0.000332540335257684"
```

NC Univariate Modeling Summary In summary for NC univariate modeling, the following models were developed with ASEs for each of the forecast periods. As shown, the MLP models seemed to perform the best on both forecasts.

Method	ASE - 7 Day	ASE - 90 Day
ARIMA(9,1,2)	1.95119e-06	0.000256
MLP - No Difference	1.717398e-06	1.30737e-05
MLP - Difference = 1	1.717398e-06	1.30737e-05
Ensemble	1.905830e-06	0.000332

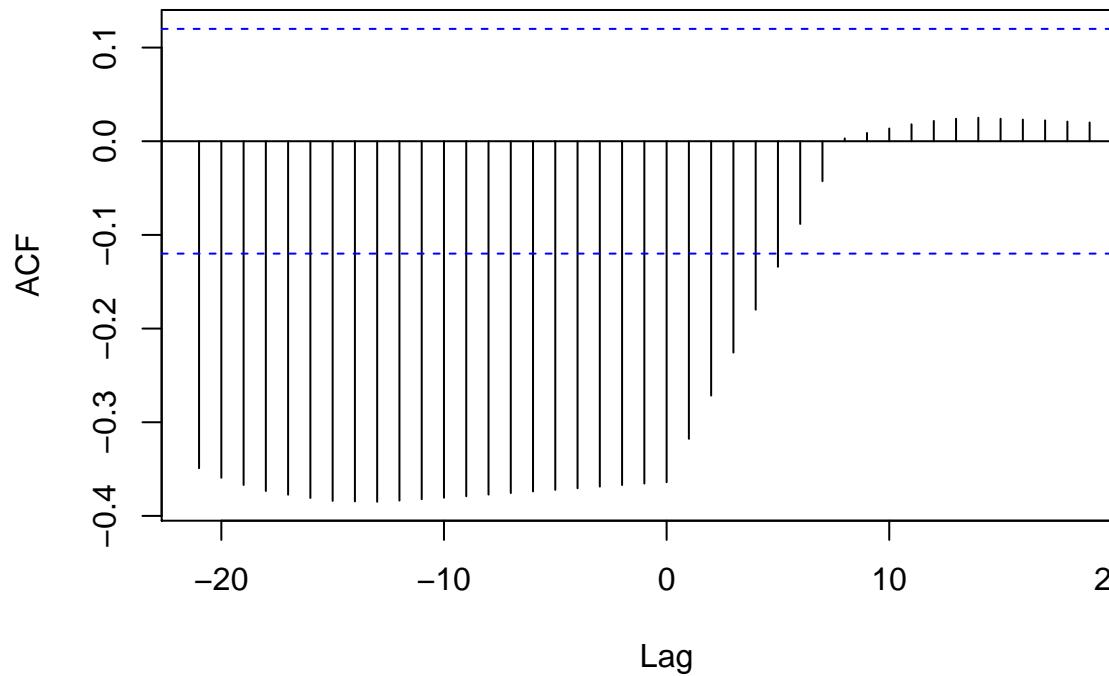
Multi-variate analysis

Progressing forward in the NC Analysis, the time series modeling will now shift to multivariate analysis. In multivariate analysis additional variables are added to the analysis to continue to refine the model. For this analysis the number of patients hospitalized will be used as the external variable. The goal is to see if the number of hospitalizations is a leading or lagging indicator for the percent positive metric.

Initially a cross correlation plot will be built to see if there is any correlation between the two variables. As shown below, the cross correlation plot seems to indicate that there is a correlation around lag -15, -14, -13. All three of these lags have the exact same value.

```
#x1 = dependent var
#x2 = independent var
# call is ccf(x2,x1)
ccf(df_NC_Final$Percent_Positive, df_NC_Final$Hospitalized_Count) # looks like correlation may be at
```

df_NC_Final\$Percent_Positive & df_NC_Final\$Hospitalized_Count



Cross Correlation Plots

Now that it is understood that the hospitalization count and percent positive rate are correlated, multivariate analysis can begin. In executing the multivariate analysis two different types of model development will occur. Just as with the univariate model a MLP model will be build for the multivariate analysis. In addition to the MLP model a VAR model will be developed. A VAR model allows for understanding the relationships between all variable types, independent or dependent. These variables are all treated the same in a VAR model and thus could result in a better application of forecasting as all variable interactions are accounted for.

As performed in the univariate analysis forecasts for short term (7-days) and long term (90-day) horizons will be leveraged. Rolling window ASE will be used as the metric

VAR Analysis - 7 Day

The VAR model for the 7-day horizon will be first evaluated. The default model settings will be used with a maximum lag of 15 for the model to select from.

```
# We know out to length of df_US_Final$Percent_Positive
backLen = length(df_NC_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON

X = cbind(df_NC_Final$Percent_Positive[1:backLen], df_NC_Final$Hospitalized_Count[1:backLen])
names(X) = c(COL_PERCENT_POSITIVE, COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut

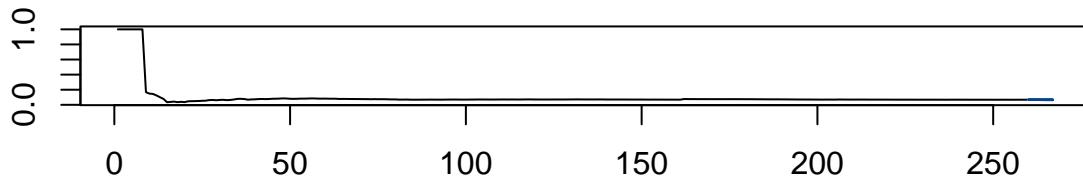
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      14      14      1      14
##
## $criteria
##           1          2          3          4          5
## AIC(n) -5.412516240 -5.410775251 -5.397098428 -5.427405393 -5.400635332
## HQ(n)   -5.377986762 -5.353226121 -5.316529648 -5.323816961 -5.274027248
## SC(n)   -5.326771141 -5.267866752 -5.197026531 -5.170170096 -5.086236636
## FPE(n)  0.004460414  0.004468226  0.004529846  0.004394774  0.004514258
##           6          7          8          9          10
## AIC(n) -5.51888191 -5.504586281 -5.53400288 -5.542525837 -5.662832916
## HQ(n)   -5.36925417 -5.331938893 -5.33833585 -5.323839147 -5.421126573
## SC(n)   -5.14731981 -5.075860785 -5.04811399 -4.999473543 -5.062617222
## FPE(n)  0.00401113  0.004069319  0.00395191  0.003919067  0.003475595
##           11         12         13         14         15
## AIC(n) -5.649217597 -5.689636509 -5.804518863 -5.84913037 -5.839686708
## HQ(n)   -5.384491603 -5.401890863 -5.493753565 -5.51534542 -5.482882107
## SC(n)   -4.991838504 -4.975094017 -5.032812971 -5.02026108 -4.953654018
## FPE(n)  0.003524176  0.003385645  0.003019327  0.00288884  0.002917701
```

From the output above, the VAR model selects a VAR(14) model based on the AIC criteria. Fitting that model to the predictions for the individual variables produces the outputs below. In this instance $y_1 = \text{Percent Positive}$ and $y_2 = \text{Hospital Count}$. From the plots, it is shown that the VAR(14) model seems to predict those values fairly well.

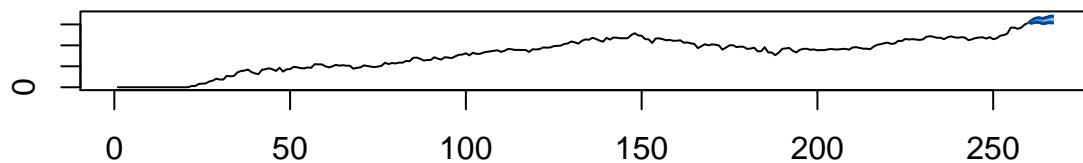
```
lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="const")
VARPredsNC7Day=predict(lsfit,n.ahead=SHORT_TERM_FORECAST_HORIZON)

#Plot forecast predictions
fanchart(VARPredsNC7Day, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make dist
```

Fanchart for variable y1



Fanchart for variable y2



Zooming in on the 7-day forecast for the percent positive from the VAR(14) model, the projected forecast trends much lower than the original data for percent positive.

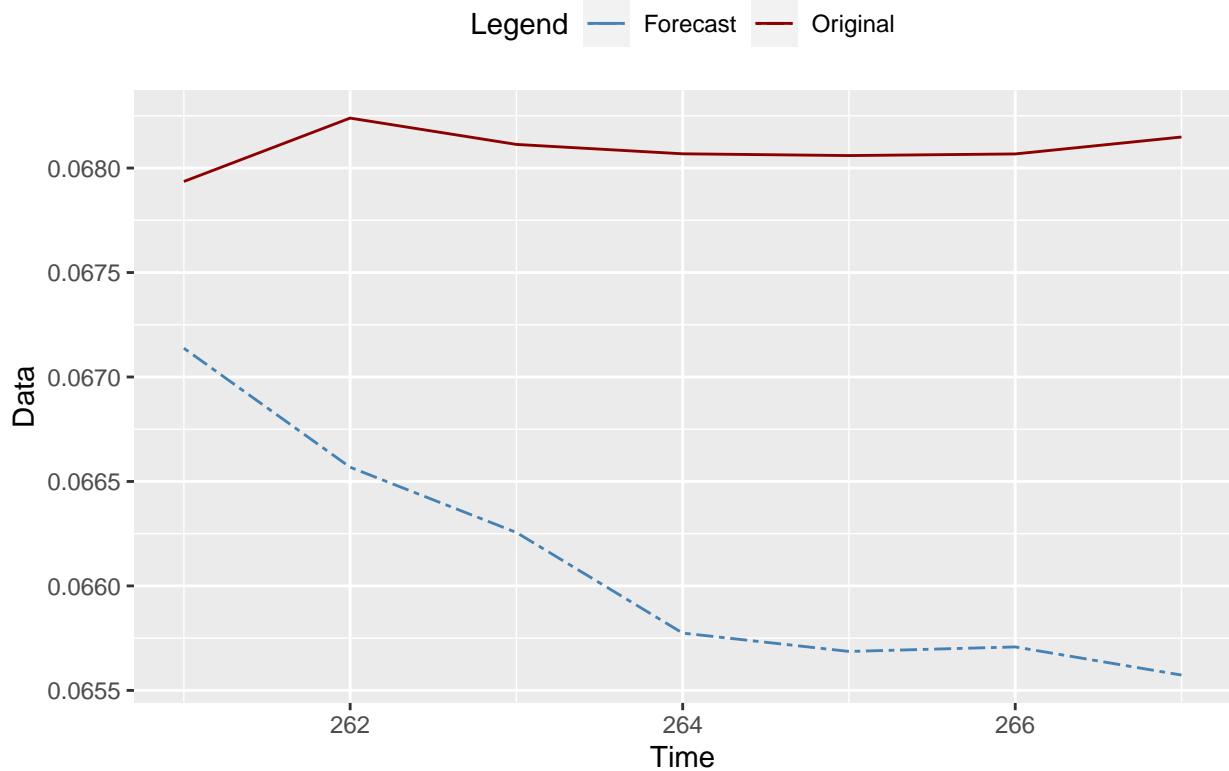
```
# Zoom In Percent Positive - 7day
startPoint = length(df_NC_Final$Percent_Positive) - 6
endPoint = length(df_NC_Final$Percent_Positive)

tl = seq(startPoint,endPoint)
orig = df_NC_Final$Percent_Positive[startPoint:endPoint]
forecast = VARPredsNC7Day$fcst$y1[,1]
df_NC7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_NC7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Percent Positive: Original vs. Forecast - 7 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

NC Percent Positive: Original vs. Forecast – 7 Day Horizon – VAR Model



Zooming in on the 7-day forecast for the hospitalization count from the VAR(14) model, the projected forecast trends well with the original values but towards the end of the realization begins to deviate lower than the original data.

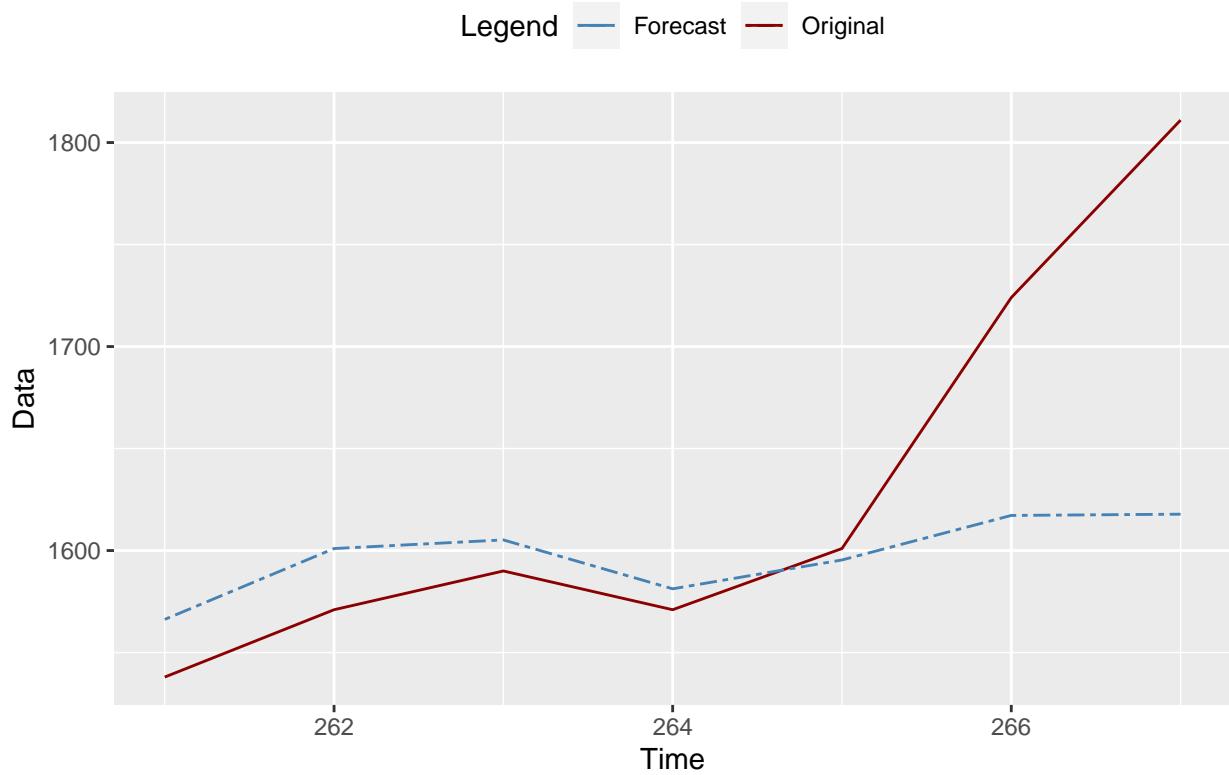
```
# Zoom In Hospital - 7day
startPoint = length(df_NC_Final$Hospitalized_Count) - 6
endPoint = length(df_NC_Final$Hospitalized_Count)

tl = seq(startPoint,endPoint)
orig = df_NC_Final$Hospitalized_Count[startPoint:endPoint]
forecast = VARPredsNC7Day$fcst$y2[,1]
df_NC7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_NC7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Hospitalization: Original vs. Forecast - 7 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

NC Hospitalization: Original vs. Forecast – 7 Day Horizon – VAR Model



Forecasting out the full 7-day period, the following is shown. The VAR model still selects the VAR(14) for the full dataset. The predictions for the 7-day horizon still seem to trend well with the variables.

```
# Full Forward 7-day
X = cbind(df_NC_Final$Percent_Positive, df_NC_Final$Hospitalized_Count)
names(X) = c(COL_PERCENT_POSITIVE, COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      14      14      1      14
##
## $criteria
##           1          2          3          4          5
## AIC(n) -5.405118270 -5.396140789 -5.381149899 -5.40828005 -5.381013115
## HQ(n)  -5.371304719 -5.339784870 -5.302251613 -5.30683940 -5.257030095
## SC(n)  -5.321084244 -5.256084079 -5.185070505 -5.15617797 -5.072888353
## FPE(n)  0.004493533  0.004534092  0.004602658  0.00447961  0.004603667
##           6          7          8          9          10
## AIC(n) -5.496557152 -5.485775446 -5.533145792 -5.552820583 -5.670716888
## HQ(n)  -5.350031764 -5.316707691 -5.341535670 -5.338668093 -5.434022031
## SC(n)  -5.132409706 -5.065605316 -5.056952979 -5.020605085 -5.082478707
## FPE(n)  0.004101618  0.004146488  0.003955156  0.003878731  0.003448065
##           11         12         13         14         15
## AIC(n) -5.655170001 -5.691387041 -5.800479304 -5.834020321 -5.832191365
```

```

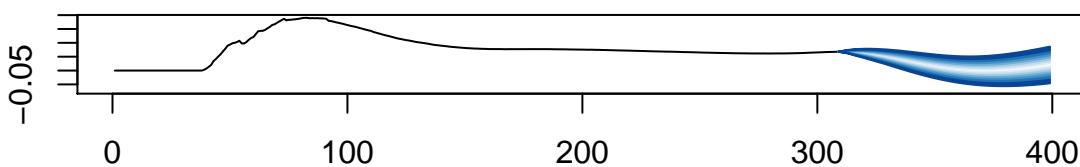
## HQ(n) -5.395932777 -5.409607449 -5.496157345 -5.507155995 -5.482784672
## SC(n) -5.010909136 -4.991103491 -5.044173071 -5.021691404 -4.963839764
## FPE(n) 0.003502945 0.003379331 0.003031105 0.002932289 0.002938998

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="const")
VARPredsNCT7DayForward=predict(lsfit,n.ahead=SHORT_TERM_FORECAST_HORIZON)

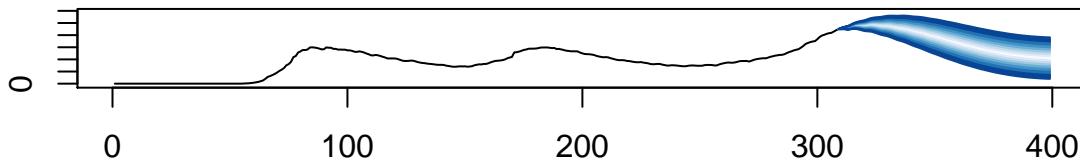
#Plot forecast predictions
fanchart(preds, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make distinguishable

```

Fanchart for variable y1



Fanchart for variable y2



VAR Analysis - 90 Day. The VAR model for the 90-day horizon will be first evaluated. The default model settings will be used with a maximum lag of 15 for the model to select from.

```

# We know out to length of df_US_Final$Percent_Positive
backLen = length(df_NC_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON

X = cbind(df_NC_Final$Percent_Positive[1:backLen],df_NC_Final$Hospitalized_Count[1:backLen])
names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "both",season = NULL, exogen = NULL)
vsOut

## $selection
## AIC(n)  HQ(n)  SC(n)  FPE(n)
##      14      1      1     14

```

```

##  

## $criteria  

##  

##          1           2           3           4           5  

## AIC(n) -5.01939137 -5.019078144 -4.988325386 -5.025209593 -4.989143216  

## HQ(n)  -4.95748465 -4.926218072 -4.864511957 -4.870442806 -4.803423072  

## SC(n)  -4.86691748 -4.790367304 -4.683377600 -4.644024860 -4.531721537  

## FPE(n)  0.00660868  0.006611066  0.006818167  0.006572275  0.006815202  

##  

##          6           7           8           9           10  

## AIC(n) -5.119005572 -5.079761847 -5.076034461 -5.087849664 -5.199354463  

## HQ(n)  -4.902332071 -4.832134988 -4.797454245 -4.778316091 -4.858867532  

## SC(n)  -4.585346946 -4.469866274 -4.389901942 -4.325480199 -4.360748051  

## FPE(n)  0.005987142  0.006229435  0.006256141  0.006186922  0.005538799  

##  

##          11          12          13          14          15  

## AIC(n) -5.181446833 -5.222617313 -5.298998932 -5.321620540 -5.317487891  

## HQ(n)  -4.810006545 -4.820223668 -4.865651929 -4.857320180 -4.822234173  

## SC(n)  -4.266603474 -4.231537008 -4.231681681 -4.178066342 -4.097696746  

## FPE(n)  0.005644614  0.005423475  0.005031745  0.004927249  0.004956954

```

From the output above, the VAR model selects a VAR(14) model based on the AIC criteria. Fitting that model to the predictions for the individual variables produces the outputs below. In this instance y_1 = Percent Positive and y_2 = Hospital Count. From the plots, it is shown that the VAR(14) model seems to predict those values fairly well.

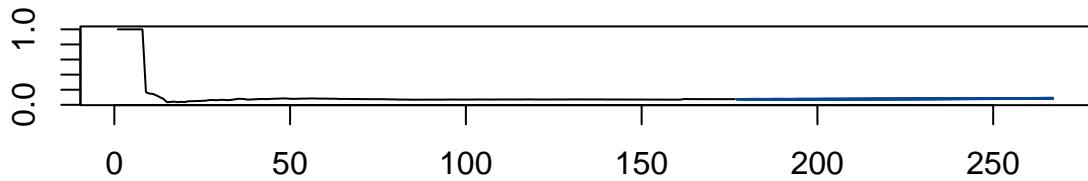
```

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="both")
VARPredsNC90Day=predict(lsfit,n.ahead=LONG_TERM_FORECAST_HORIZON)

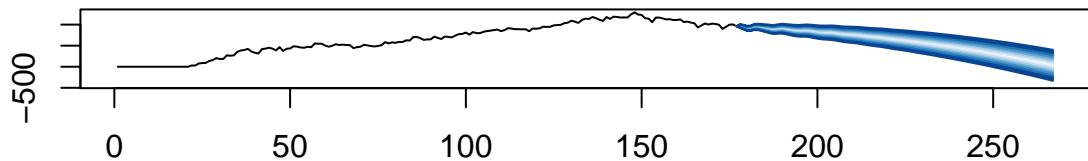
#Plot forecast predictions
fanchart(VARPredsNC90Day, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to make dis

```

Fanchart for variable y1



Fanchart for variable y2



Zooming in on the 90-day forecast for the percent positive from the VAR(14) model, the projected forecast initially tracks well but then deviates and over forecasts when compared to the original.

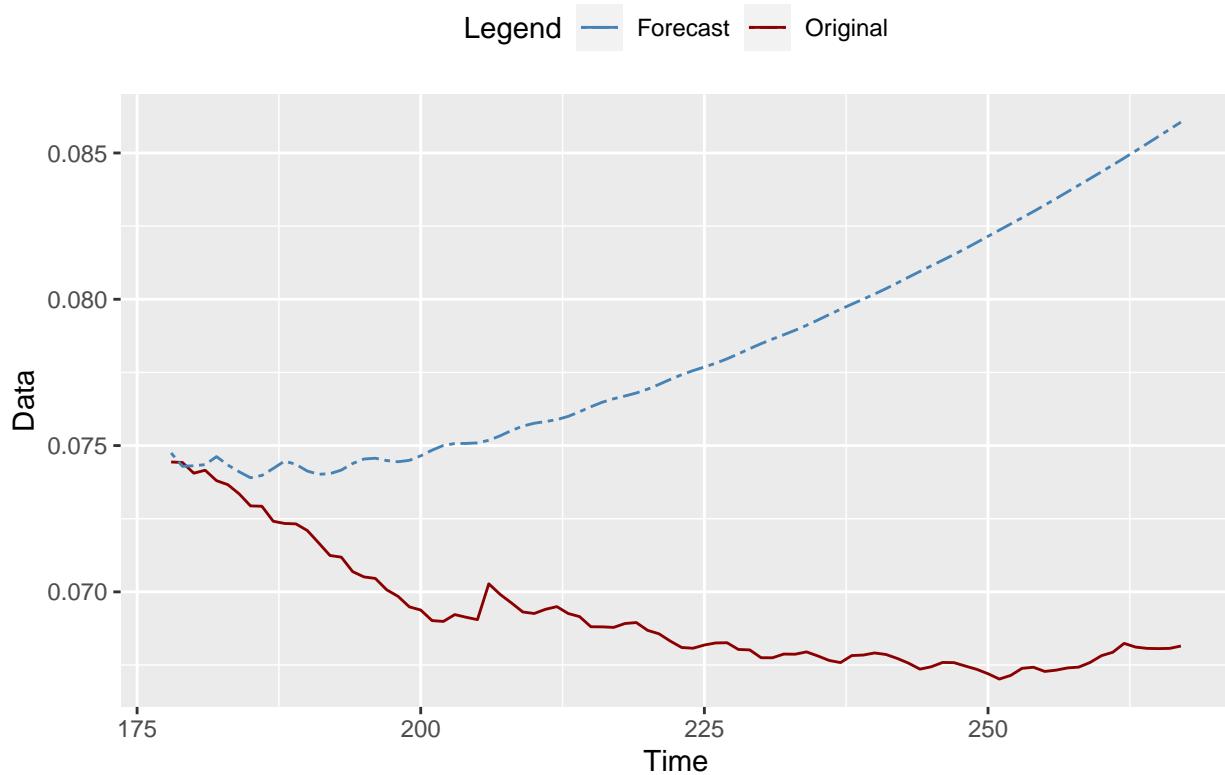
```
# Zoom In Percent Positive - 90day
startPoint = length(df_NC_Final$Percent_Positive) - 89
endPoint = length(df_NC_Final$Percent_Positive)

tl = seq(startPoint,endPoint)
orig = df_NC_Final$Percent_Positive[startPoint:endPoint]
forecast = VARPredsNC90Day$fcst$y1[,1]
df_NC7 = data.frame(tl,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_NC7,aes(x=tl)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Percent Positive: Original vs. Forecast - 90 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

NC Percent Positive: Original vs. Forecast – 90 Day Horizon – VAR Mode



Zooming in on the 90-day forecast for the hospitalization count from the VAR(14) model, the project forecast does not trend well at all with the original data. The hospitalization count initially is over projected but does start to move to the original data.

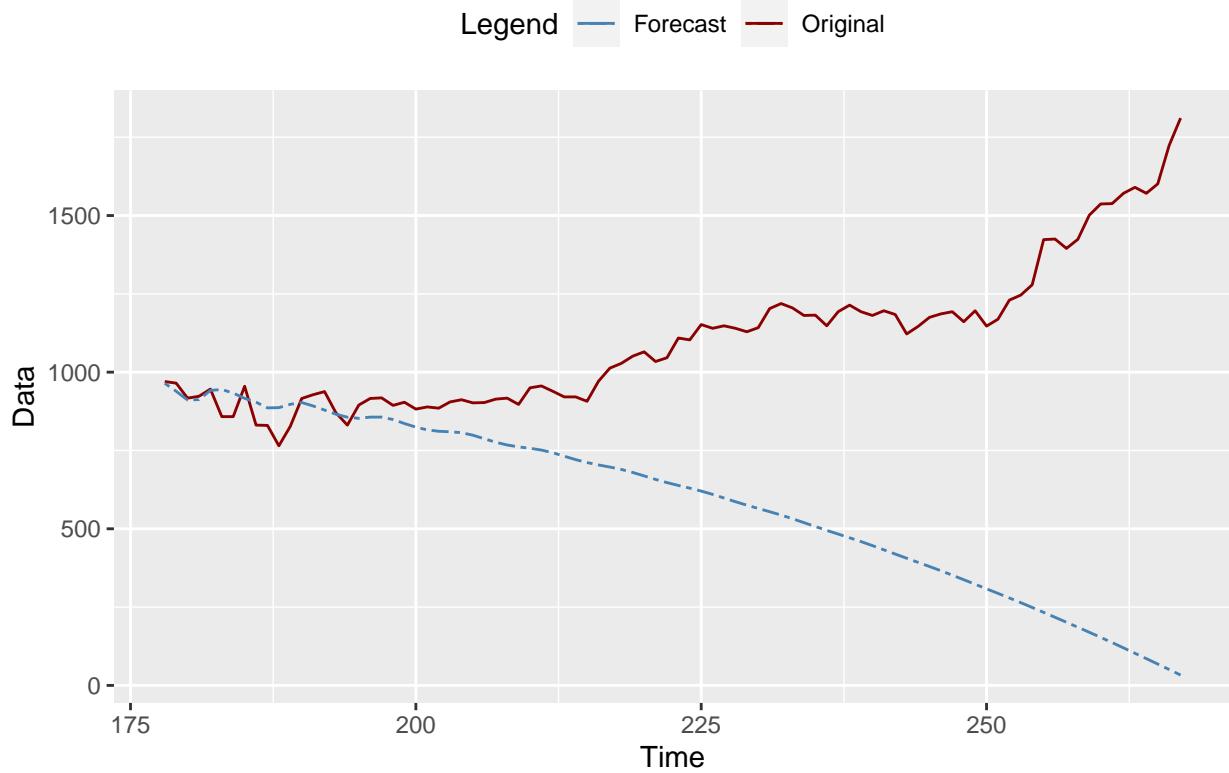
```
# Zoom In Percent Positive - 90day
startPoint = length(df_NC_Final$Hospitalized_Count) - 89
endPoint = length(df_NC_Final$Hospitalized_Count)

t1 = seq(startPoint,endPoint)
orig = df_NC_Final$Hospitalized_Count[startPoint:endPoint]
forecast = VARPredsNC90Day$fcst$y2[,1]
df_NC7 = data.frame(t1,orig,forecast)

colors <- c("Original" = "darkred", "Forecast" = "steelblue")

ggplot(df_NC7,aes(x=t1)) +
  geom_line(aes(y=orig, color="Original")) +
  geom_line(aes(y=forecast, color="Forecast"),linetype="twodash") +
  theme(legend.position="top") +
  labs(x="Time",y="Data",color="Legend") +
  ggtitle("NC Hospitalization Count: Original vs. Forecast - 90 Day Horizon - VAR Model") +
  scale_color_manual(values = colors)
```

NC Hospitalization Count: Original vs. Forecast – 90 Day Horizon – VAR M



Forecasting out the full 90-day period, the following is shown. The VAR model still selects the VAR(14) for the full dataset. The predictions for the 90-day horizon still seem to trend well with the variables. As expected the confidence intervals continue to increase the further away from the original data plots the forecast moves.

```
# Full Forward 90-day
X = cbind(df_NC_Final$Percent_Positive,df_NC_Final$Hospitalized_Count)
names(X) = c(COL_PERCENT_POSITIVE,COL_HOSPITAL_COUNT)
vsOut = VARselect(X, lag.max = 15, type = "const", season = NULL, exogen = NULL)
vsOut
```

```
## $selection
## AIC(n)  HQ(n)  SC(n)  FPE(n)
##      14      14      1      14
##
## $criteria
##           1          2          3          4          5
## AIC(n) -5.405118270 -5.396140789 -5.381149899 -5.40828005 -5.381013115
## HQ(n)  -5.371304719 -5.339784870 -5.302251613 -5.30683940 -5.257030095
## SC(n)  -5.321084244 -5.256084079 -5.185070505 -5.15617797 -5.072888353
## FPE(n)  0.004493533  0.004534092  0.004602658  0.00447961  0.004603667
##           6          7          8          9         10
## AIC(n) -5.496557152 -5.485775446 -5.533145792 -5.552820583 -5.670716888
## HQ(n)  -5.350031764 -5.316707691 -5.341535670 -5.338668093 -5.434022031
## SC(n)  -5.132409706 -5.065605316 -5.056952979 -5.020605085 -5.082478707
## FPE(n)  0.004101618  0.004146488  0.003955156  0.003878731  0.003448065
```

```

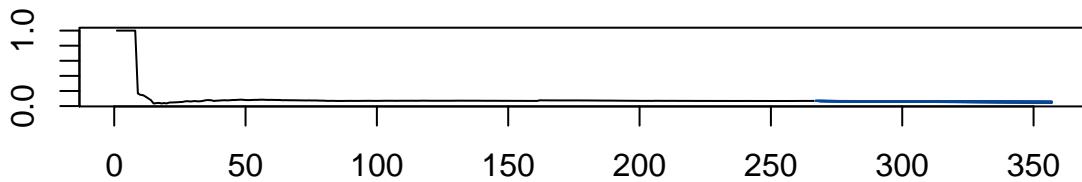
##          11          12          13          14          15
## AIC(n) -5.655170001 -5.691387041 -5.800479304 -5.834020321 -5.832191365
## HQ(n)  -5.395932777 -5.409607449 -5.496157345 -5.507155995 -5.482784672
## SC(n)  -5.010909136 -4.991103491 -5.044173071 -5.021691404 -4.963839764
## FPE(n)  0.003502945  0.003379331  0.003031105  0.002932289  0.002938998

lsfit=VAR(X,p=vsOut$selection[1][["AIC(n)"]],type="const")
VARPredsNC90DayForward=predict(lsfit,n.ahead=LONG_TERM_FORECAST_HORIZON)

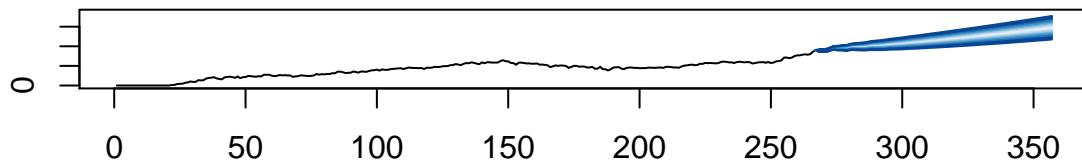
#Plot forecast predictions
fanchart(VARPredsNC90DayForward, colors = brewer.pal(n = 8, name = "Blues")) # Change color pallet to m

```

Fanchart for variable y1



Fanchart for variable y2



For comparison purposes calculate the rolling window ASEs for the VAR models.

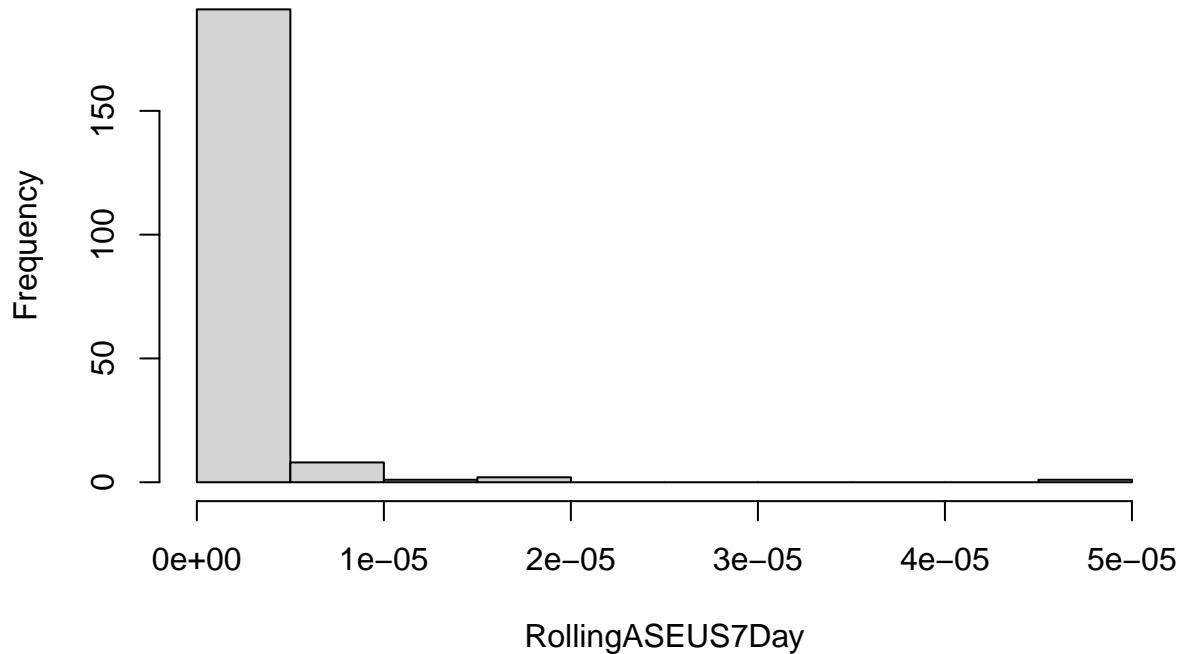
```

RollingASEVARNC7Day = RollingASECalcVAR(df_NC_Final$Percent_Positive,df_NC_Final$Hospitalized_Count,pVa
RollingASEVARNC90Day = RollingASECalcVAR(df_NC_Final$Percent_Positive,df_NC_Final$Hospitalized_Count,pVa

```

```
GetRollingWindowASEInfo("7-Day VAR",RollingASEVARNC7Day)
```

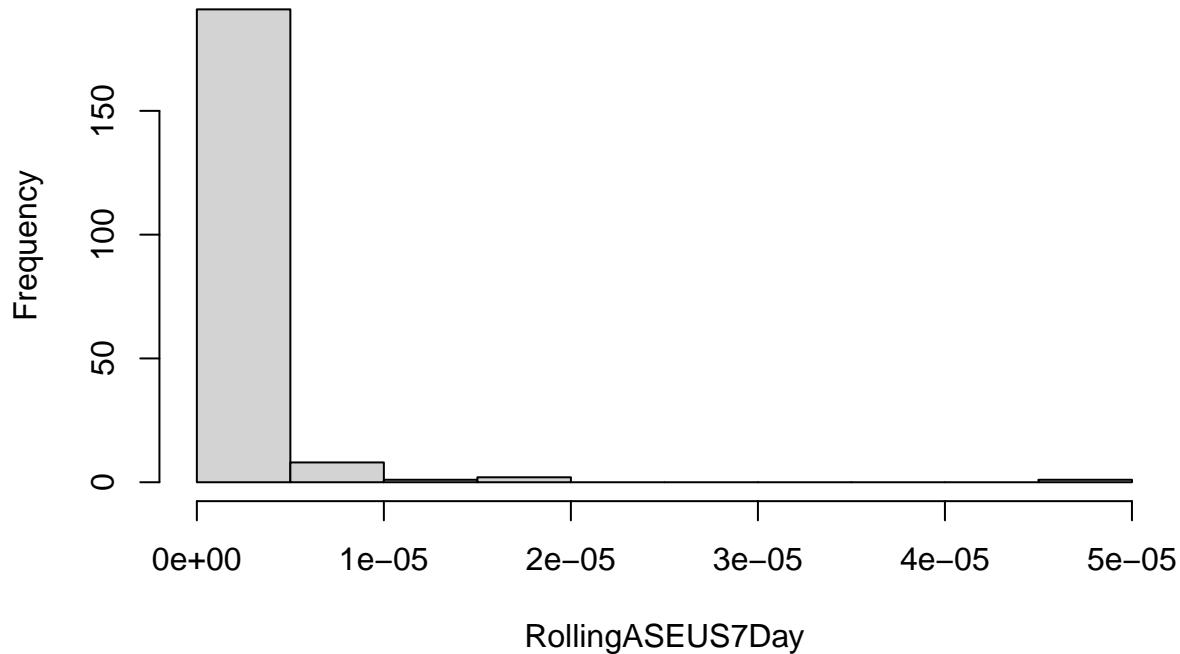
Histogram of 7-Day VAR Windowed ASE



```
## [1] "7-Day VAR Windowed ASE Mean: 1.04597118062974e-05"  
## [1] "7-Day VAR Windowed ASE Median: 5.19603732993768e-06"
```

```
GetRollingWindowASEInfo("90-Day VAR", RollingASEVARNC90Day)
```

Histogram of 90-Day VAR Windowed ASE



```
## [1] "90-Day VAR Windowed ASE Mean: 0.00293169943914712"  
## [1] "90-Day VAR Windowed ASE Median: 1.59022926681072e-05"
```

The rolling window ASE plots for the VAR(14) model are:

- Rolling Window ASE - 7 Day: 1.045971e-05
- Rolling Window ASE - 90 Day: 0.002931

Multi-Layer Perceptron (MLP) Another time series modeling approach that can be taken is to leverage a neural network, also called a Multi-Layer Perceptron (MLP). Through this model an input layer is defined. Each input layer is connected to a “hidden layer”. The connection between the input layer and the “hidden layer” contains a weighting. The “hidden layer” is used to approximate any continuous function. Within the MLP model there can be one or more hidden layers. The “hidden layers” then all connect to an output layer. The output layer provides the final result of the model.

First the model will be evaluated. The model will be set to perform 50 repetitions of the MLP. Each of these repetition results will be combined based on the mean calculation. For this multivariate MLP model the, similar to the VAR model, the Hospitalization count will be added to the model.

Evaluating the model with the base data results in the following:

```
# FULL FORWARD FORECAST  
endPoint90Day = length(df_NC_Final$Hospitalized_Count) - LONG_TERM_FORECAST_HORIZON -1  
endPoint7Day = length(df_NC_Final$Hospitalized_Count) - SHORT_TERM_FORECAST_HORIZON-1  
fullData = data.frame(hosp=ts(df_NC_Final$Hospitalized_Count))
```

```

# Build for 7 Day Horizon
tl = ts(df_NC_Final$Percent_Positive[1:endPoint7Day])
Sx = data.frame(hosp=ts(df_NC_Final$Hospitalized_Count[1:endPoint7Day]))
fit.mlp.MVNC7 = mlp(tl,xreg=Sx,sel.lag = TRUE,reps=NN_REPS,hd.auto.type="cv")
fore.mlp.MVNC7 = forecast(fit.mlp.MVNC7, h = SHORT_TERM_FORECAST_HORIZON,xreg=fullData)

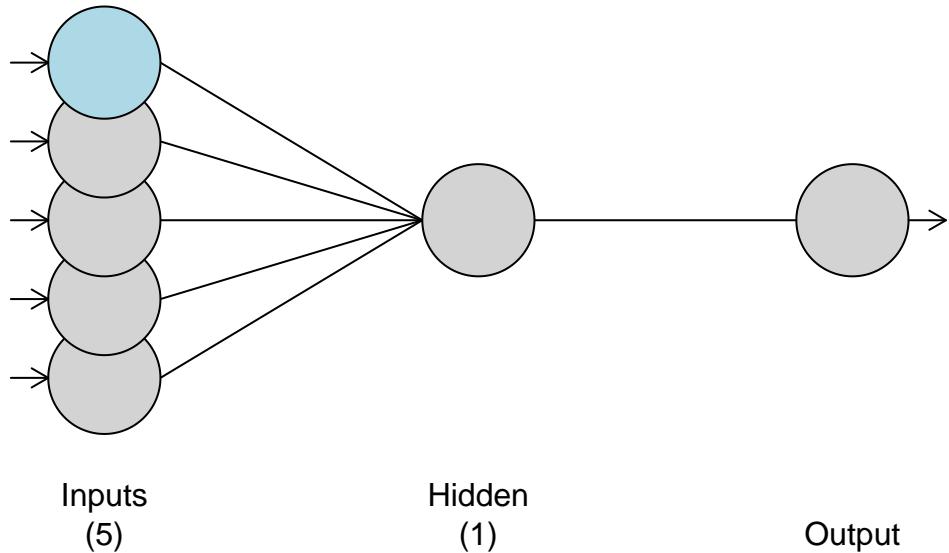
# Build for 90 Day Horizon
tl = ts(df_NC_Final$Percent_Positive[1:endPoint90Day])
Sx = data.frame(hosp=ts(df_NC_Final$Hospitalized_Count[1:endPoint90Day]))
fit.mlp.MVNC90 = mlp(tl,xreg=Sx,sel.lag = TRUE,reps=NN_REPS,hd.auto.type="cv")
fore.mlp.MVNC90 = forecast(fit.mlp.MVNC90, h = LONG_TERM_FORECAST_HORIZON,xreg=fullData)

```

The visual below represents the MLP configuration used for the short-term (7-day) forecast. The model contains five (5) input layers and one (1) hidden layer.

```
plot(fit.mlp.MVNC7)
```

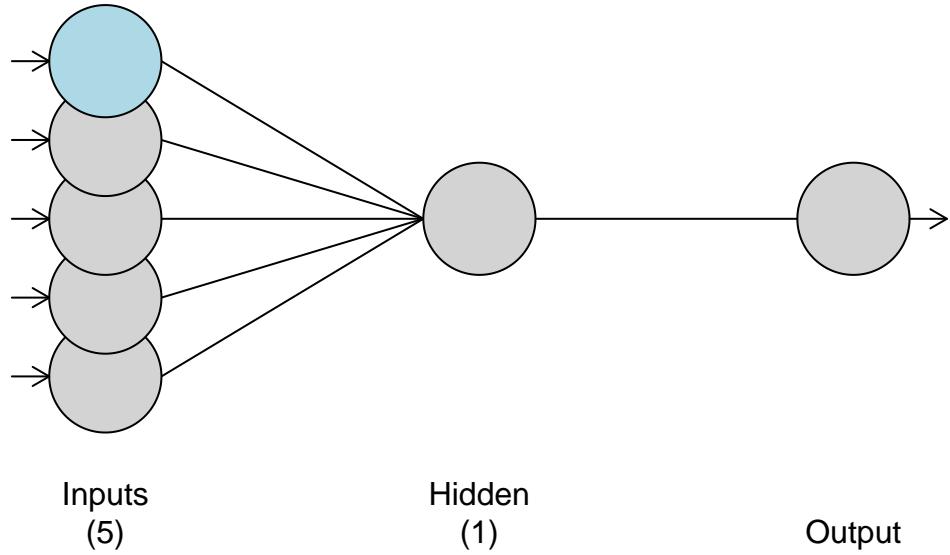
MLP



The visual below represents the MLP configuration used for the long-term (90-day) forecast. The model contains five (5) input layers and one (1) hidden layer.

```
plot(fit.mlp.MVNC90)
```

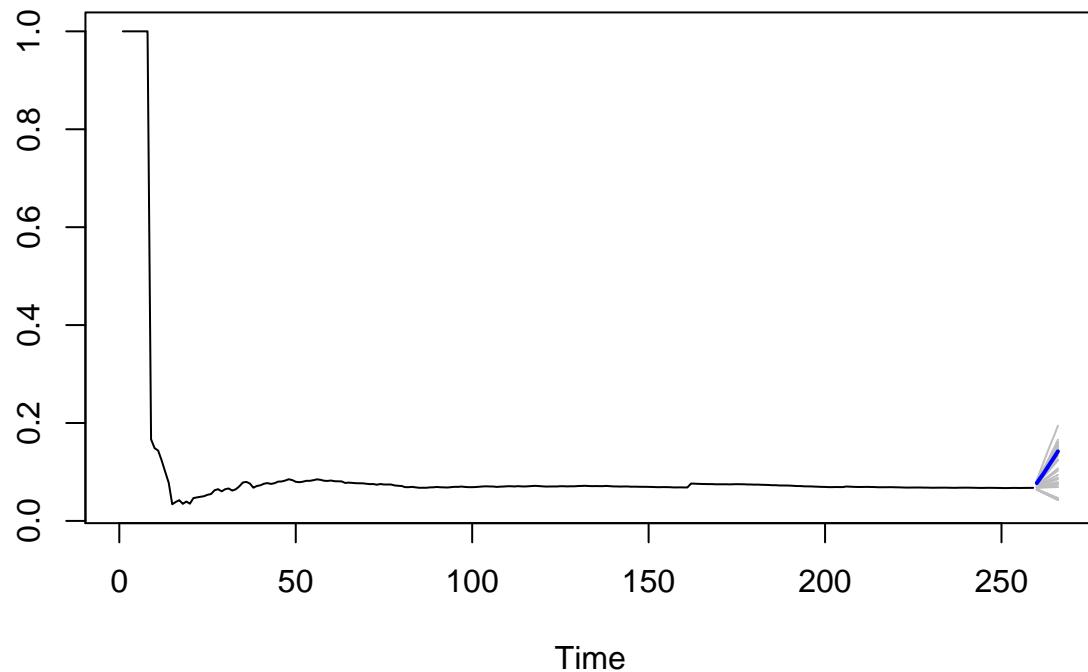
MLP



The plot below shows the 7-day forecasts that were calculated from the MLP model. From the visual is seems that the majority of the projections for the short-term forecast were trending in the same direction. The blue color on the plot represents the mean of the projections. It does seem that this forecast begins an upward trend.

```
plot(fore.mlp.MVNC7)
```

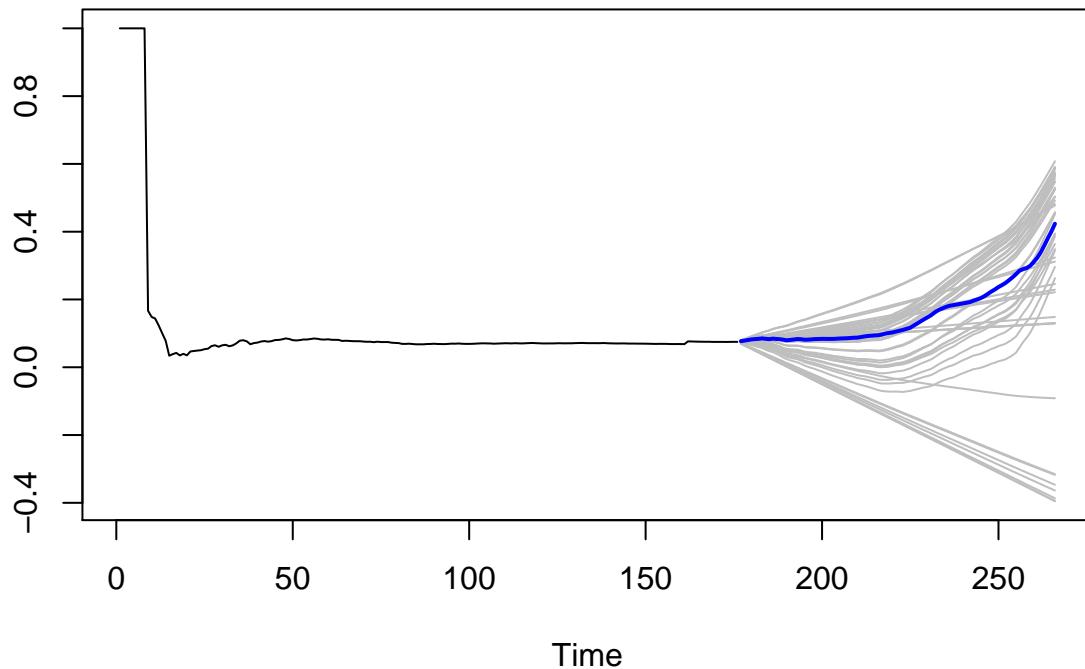
Forecasts from MLP



The plot below shows the 90-day forecasts that were calculated from the MLP model. From this it can be seen that there were multiple projections, both higher and lower (gray lines) but that the mean trend (blue line) seems to follow the original realization and then begin a climb upward.

```
plot(fore.mlp.MVNC90)
```

Forecasts from MLP

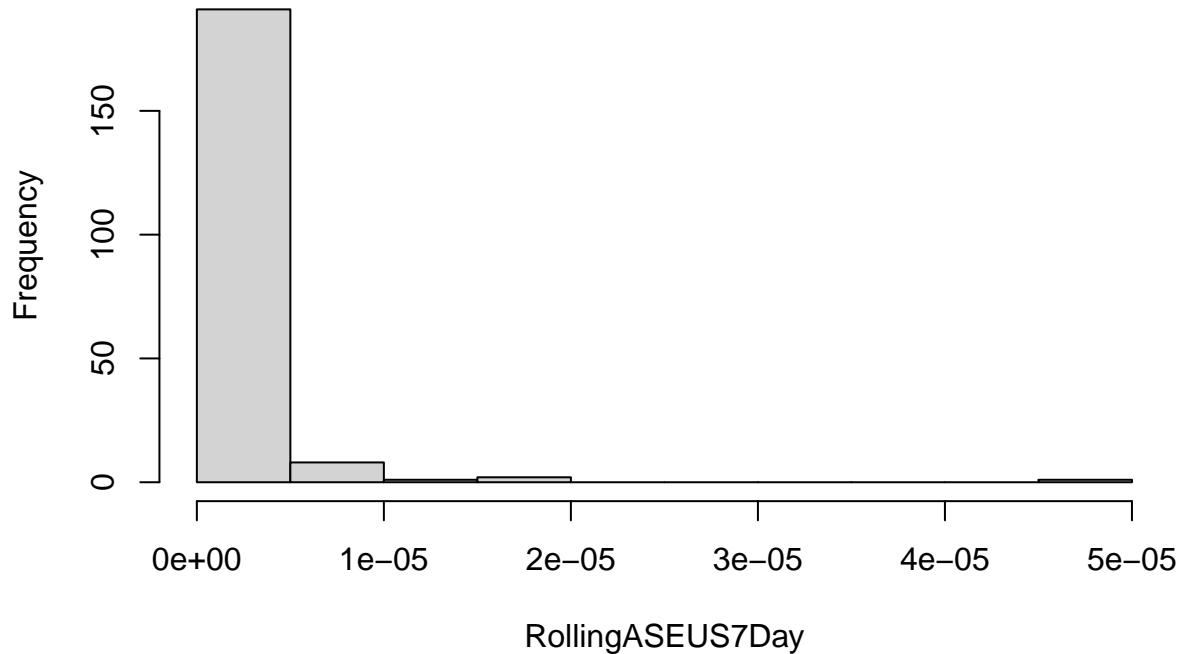


For the comparisons to the other models, the rolling window ASE will be calculated.

```
RollingASECalcMVNCMLP7Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=SHORT)
RollingASECalcMVNCMLP90Day = RollingASECalcMLP(fit.mlp.US$y, sampleSize = SAMPLE_SIZE_WINDOW, horizon=LONG)
```

```
modelName = "US Model - 7 Days"
GetRollingWindowASEInfo(modelName, RollingASECalcMVNCMLP7Day)
```

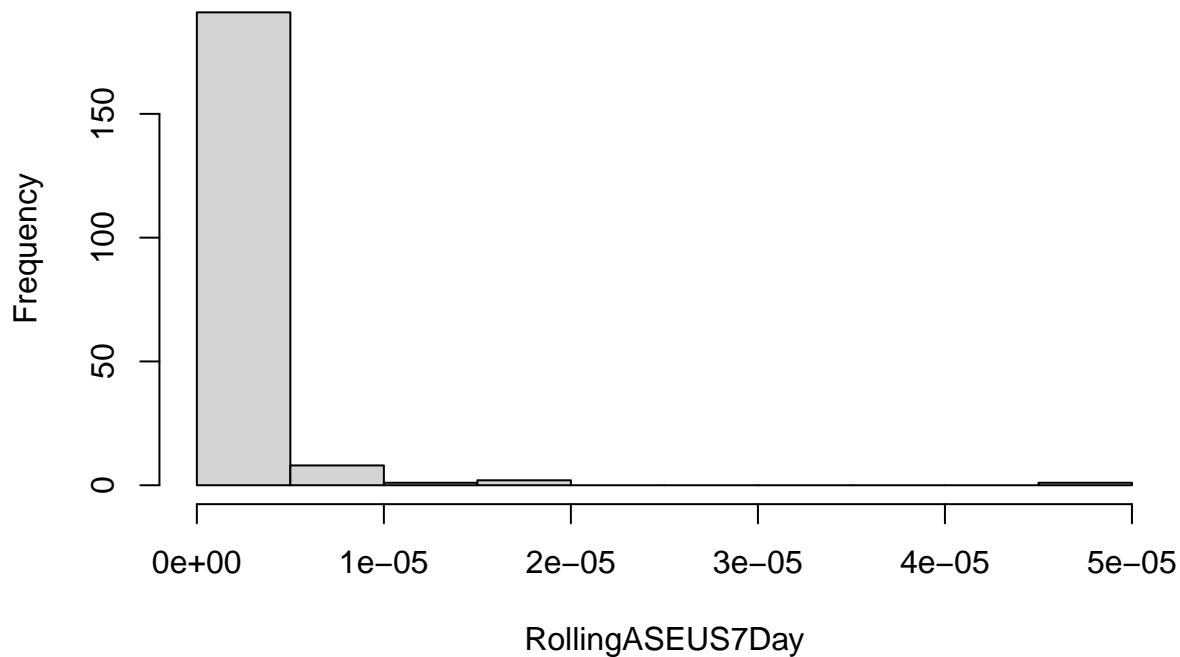
Histogram of US Model – 7 Days Windowed ASE



```
## [1] "US Model - 7 Days Windowed ASE Mean: 2.01956844442391e-06"  
## [1] "US Model - 7 Days Windowed ASE Median: 1.14642944922895e-07"
```

```
modelName = "US Model - 90 Days"  
GetRollingWindowASEInfo(modelName, RollingASECalcMVNCMLP90Day)
```

Histogram of US Model – 90 Days Windowed ASE



```
## [1] "US Model - 90 Days Windowed ASE Mean: 0.000530849683581231"
## [1] "US Model - 90 Days Windowed ASE Median: 5.60126383351628e-05"
```

The rolling window ASE plots for this iteration of the MLP are:

- Rolling Window ASE - 7 Day: 2.019568e-06
- Rolling Window ASE - 90 Day: 0.000530

In comparison to the VAR(14) model that was developed these ASE values are much lower for the MLP model.

Multivariate Ensemble Modeling Ensemble modeling is a way of combining models to see if the positives of both models will make the forecasts better. As was done in the univariate case, models used in the multivariate analysis will be combined to see if there is better performance. Ensemble models will be executed for both the short-term and long-term forecasts.

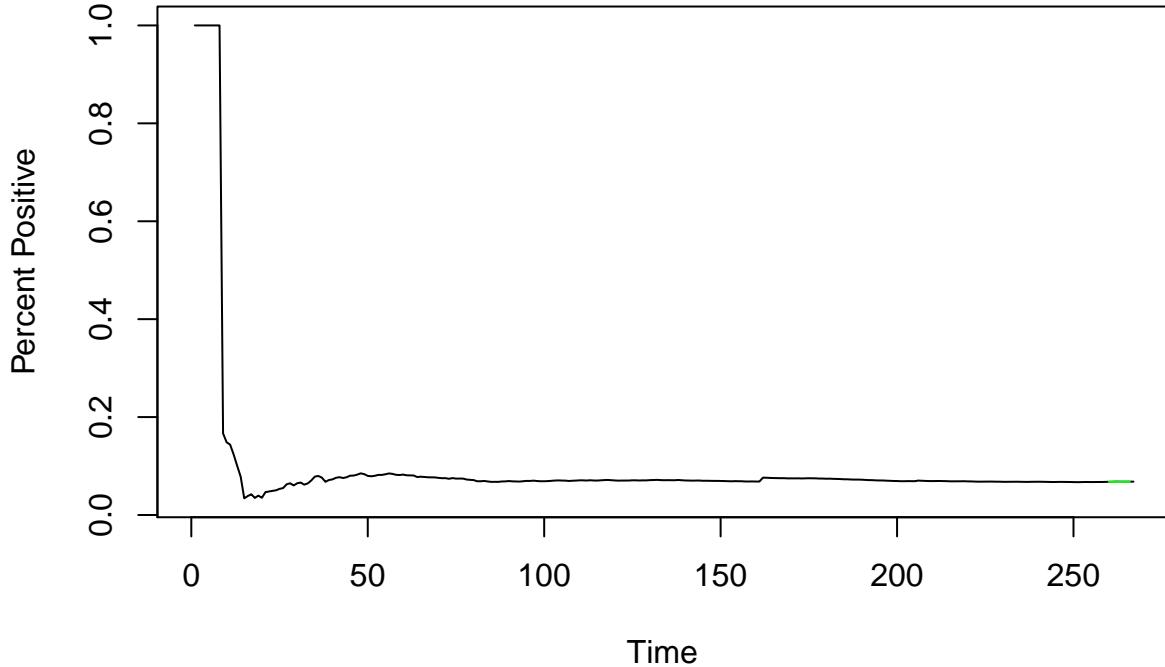
Start with the short term forecast. The ensemble projection, shown in green, seems to follow the actual realization fairly well.

```
startPos = length(df_NC_Final$Percent_Positive) - SHORT_TERM_FORECAST_HORIZON
endPos = length(df_NC_Final$Percent_Positive)-1

NCMV7Ensemble = (VARPredsNC7Day$fcst$y1[,1] + fore.mlp.NC7$mean)/2
```

```
plot(seq(1,length(df_NC_Final$Percent_Positive),1),df_NC_Final$Percent_Positive, type = "l",xlim = c(1,275)
lines(seq(startPos,endPos,1), NCMV7Ensemble, type = "l", col = "green")
```

Total NC – Percent Positive – 7 Day Forecast – Ensemble Model



```
NCMV7EnsembleASE = mean((df_NC_Final$Percent_Positive[startPos:endPos] - NCMV7Ensemble)^2)
```

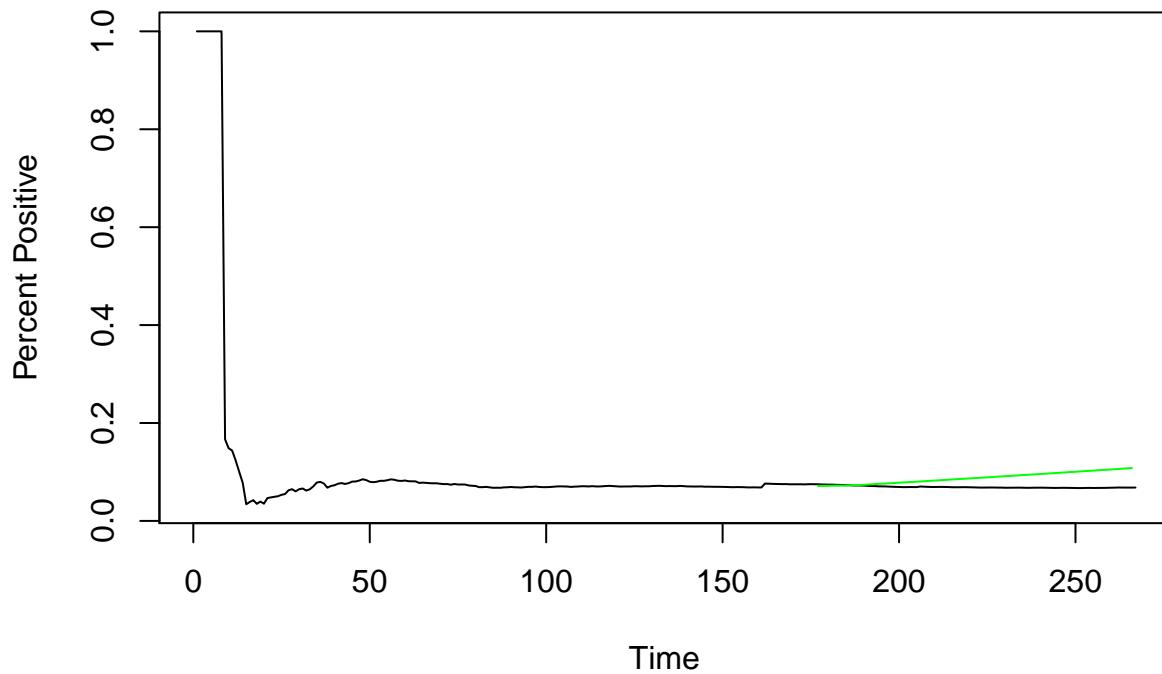
Now move to the 90 day forecast. The long-term forecast for the ensemble seems to track reasonably well with the observations. The forecast eventually rises above the original realization, but not to a large degree and it seems to be a gradual increase.

```
startPos = length(df_NC_Final$Percent_Positive) - LONG_TERM_FORECAST_HORIZON
endPos = length(df_NC_Final$Percent_Positive)-1

NCMV90Ensemble = (VARPredsNC90Day$fcst$y1[,1] + fore.mlp.NC90$mean)/2

plot(seq(1,length(df_NC_Final$Percent_Positive),1),df_NC_Final$Percent_Positive, type = "l",xlim = c(1,275)
lines(seq(startPos,endPos,1), NCMV90Ensemble, type = "l", col = "green")
```

Total NC – Percent Positive – 90 Day Forecast – Ensemble Model



```
NCMV90EnsembleASE = mean((df_NC_Final$Percent_Positive[startPos:endPos] - NCMV90Ensemble)^2)
```

ASEs for the ensemble models are:

```
print(paste("ASE Ensemble 7-Day: ", NCMV7EnsembleASE))
```

```
## [1] "ASE Ensemble 7-Day: 2.57810968815451e-07"
```

```
print(paste("ASE Ensemble 90-Day: ", NCMV90EnsembleASE))
```

```
## [1] "ASE Ensemble 90-Day: 0.000517703886833119"
```

NC Multivariate Modeling Summary In summary for NC multivariate modeling, the following models were developed with ASEs for each of the forecast periods. As shown, the Ensemble models generally seemed to perform the best on both forecasts.

Method	ASE - 7 Day	ASE - 90 Day
VAR(14)	1.045971e-05	0.002931
MLP	2.019568e-06	0.000530
Ensemble	2.578109e-07	0.000517

Conclusion

The analysis above has shown that multiple time series models have been built on the COVID-19 positivity rate data for both the United States and North Carolina. A variety of time series models have been leveraged to show both univariate and multivariate modeling techniques, with hospitalization data added for multivariate analysis.

In reviewing the US models, the MLP models performed the best based on ASE calculations for both the univariate and multivariate calculations over both forecast horizons. For reference the Univariate and Multivariate ASE summary tables are shown below.

Analysis Type	Method	ASE - 7 Day	ASE - 90 Day
Univariate	ARIMA(13,2,0)	1.26780e-06	0.00173
Univariate	MLP - No Difference	2.01956e-06	0.00053
Univariate	MLP - Difference = 2	2.01956e-06	0.00053
Univariate	Ensemble	1.37443e-05	0.00099
Multivariate	VAR(15)	0.00010	2.6734e+35
Multivariate	MLP	2.019568e-06	0.000530
Multivariate	Ensemble	1.135600e-05	0.000512

In reviewing the NC models, the MLP models performed the Ensemble based method performed the best in a multivariate scenario while the MLP model performed best in the univariate scenario.

Analysis Type	Method	ASE - 7 Day	ASE - 90 Day
Univariate	ARIMA(9,1,2)	1.95119e-06	0.000256
Univariate	MLP - No Difference	1.717398e-06	1.30737e-05
Univariate	MLP - Difference = 1	1.717398e-06	1.30737e-05
Univariate	Ensemble	1.905830e-06	0.000332
Multivariate	VAR(14)	1.045971e-05	0.002931
Multivariate	MLP	2.019568e-06	0.000530
Multivariate	Ensemble	2.578109e-07	0.000517

In conclusion, the modeling of COVID-19 positivity rates is possible with even simple models such as the ones in this analysis. These models may be leveraged as a baseline for further model building. From a univariate perspective the models are relatively straight forward. It will be interesting to see, as more data becomes available, if there will be seasonal patterns that emerge. At this time none have been seen in the modeling performed in this analysis. On the multivariate front, there are many other external variables that could be included in future analysis. Understanding the impacts of weather, pollution, mask enforcement and other policy restrictions could all be added to the multivariate models to further enhance them.