Accompanying Paper Code

This vignette is a companion to the paper 'The maximally selected likelihood ratio test in random coefficient models.' The code to build all figures in that paper are presented. Discussions on these figures are conducted in the paper and are not addressed in this data supplement.

The stationary region was built using the following code.

```
set.seed(12345)
data_sr <- simulateStationaryRegion(nSims = 10000000, silent=T)
```

The results of that code includes the following figure.

```
data_sr[[3]]
```

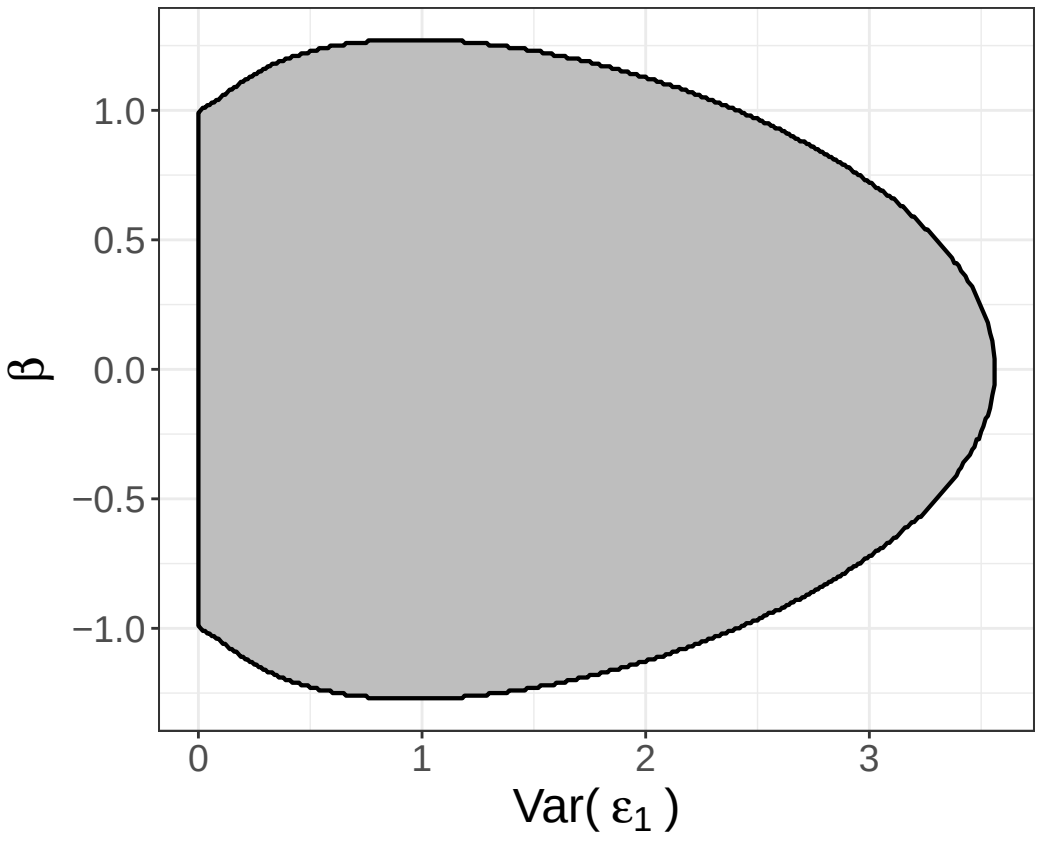

Figure 1: Simulation stationary region using Gaussian error based on $E \log |\beta + \epsilon_{0,1}|$

The empirical rejection frequencies under the null case was generated using the following code.

```
data_er <- simulationTable_NC(betas = c(0.5,0.75,1,1.05),
                              varProbRates = c(0.5, 0.5),
                              nSims=500, iterations = c(100, 200, 400, 800),
                              burnin = 1000, lowerEst=c(-Inf,0,10^-8,-Inf),
                              upperEst=c(Inf,Inf,Inf,Inf), alpha = 0.05,
                              errorTypes = c('Normal','Bernoulli','Exponential'),
                              CPLoc=0.5, seed=1234,trimAmt=10)
```

The results of that code is the following data.

```
data_er$errType <- ifelse(data_er$errType=='Normal','N',
                          ifelse(data_er$errType=='Bernoulli','B','E'))
tmp <- data_er %>%
  dplyr::select(-c(Lower,Upper)) %>%
  pivot_wider(names_from = c(iters,errType), values_from = Estim)

kable(tmp[,!(colnames(tmp)%in%c('beta'))], booktabs = TRUE) %>%
  pack_rows(index = table(tmp$beta))
```

type	100_N	200_N	400_N	800_N	100_B	200_B	400_B	800_B	100_E	200_E	400_E	800_E
0.5												
MLE	0.002	0.012	0.004	0.010	0.002	0.006	0.006	0.004	0.006	0.010	0.028	0.016
Vost	0.026	0.046	0.024	0.032	0.032	0.032	0.034	0.028	0.028	0.026	0.042	0.052
WLS	0.004	0.012	0.002	0.018	0.012	0.012	0.012	0.010	0.004	0.006	0.014	0.016
0.75												
MLE	0.002	0.010	0.002	0.010	0.010	0.006	0.004	0.004	0.004	0.008	0.022	0.026
Vost	0.030	0.028	0.028	0.036	0.044	0.024	0.032	0.026	0.026	0.024	0.048	0.052
WLS	0.020	0.032	0.026	0.034	0.034	0.024	0.056	0.040	0.006	0.004	0.018	0.024
1												
MLE	0.008	0.010	0.002	0.006	0.006	0.010	0.008	0.012	0.006	0.008	0.016	0.020
Vost	0.022	0.040	0.032	0.036	0.048	0.032	0.036	0.038	0.030	0.036	0.054	0.040
WLS	0.030	0.054	0.058	0.058	0.076	0.098	0.138	0.142	0.002	0.004	0.018	0.016
1.05												
MLE	0.002	0.016	0.012	0.012	0.010	0.010	0.008	0.008	0.004	0.010	0.020	0.024
Vost	0.024	0.052	0.032	0.044	0.046	0.038	0.038	0.036	0.028	0.036	0.058	0.048
WLS	0.026	0.078	0.058	0.066	0.066	0.090	0.136	0.150	0.004	0.006	0.014	0.016

The power curves were built using the following code.

```
tmp <- c(1, 0.9,0.75, seq(0.6,0.3,-0.1),0.25,0.1)
U4seq <- c(tmp, 0, rev(-tmp))
set.seed(1234)
pc1 <- simulatePowerCurve(Us3 = c(0, 0.5, 0.5), U4s = U4seq,
                      nSims = 500, lowerEst = c(-Inf,0,10^-8,-Inf),
                      upperEst = rep(Inf,4), alpha = 0.05,
                      burnin = 1000, k = 0.5 * 400, N = 400,
                      errorType = 'Normal', trimAmt=10,
                      silent=T)

set.seed(1234)
pc2 <- simulatePowerCurve(Us3 = c(0, 0.5, 0.5), U4s = U4seq,
                      nSims = 500, lowerEst = c(-Inf,0,10^-8,-Inf),
```

```
                      upperEst = rep(Inf,4), alpha = 0.05,
                      burnin = 1000, k = 0.9 * 400, N = 400,
                      errorType = 'Normal', trimAmt=10,
                      silent=T)

U4seq <- c(1.05,seq(1,0.6,-0.1), 0.5, seq(0.4,0,-0.1),-0.05)
set.seed(1234)
pc3 <- simulatePowerCurve(Us3 = c(0.5, 0.5, 0.5), U4s = U4seq,
                      nSims = 500, lowerEst = c(-Inf,0,10^-8,-Inf),
                      upperEst = rep(Inf,4), alpha = 0.05,
                      burnin = 1000, k = 0.5 * 400, N = 400,
                      errorType = 'Normal', trimAmt=10,
                      silent=T)
set.seed(1234)
pc4 <- simulatePowerCurve(Us3 = c(0.5, 0.5, 0.5), U4s = U4seq,
                      nSims = 500, lowerEst = c(-Inf,0,10^-8,-Inf),
                      upperEst = rep(Inf,4), alpha = 0.05,
                      burnin = 1000, k = 0.5 * 400, N = 400,
                      errorType = 'Bernoulli', trimAmt=10,
                      silent=T)
```

The resulting power figures are as follows (See the paper for details on each of the figures).

```
pc1[[1]]
```

```
pc2[[1]]
```

```
pc3[[1]]
```

```
pc4[[1]]
```

The US housing figures were built using the following code.

```
housingDataList <- list()
for(city in c('Boston','LosAngeles')){

  ## Load data
  data <- housing[housing$city==city,c(2,3)]
  data <- data[order(data$date),]

  ## Detect CP
  set.seed(12345)
  result_Vost <- binarySegmentationCPDetection(fullData=data,
                                       method='Vostrikova',
                                       lower=c(-Inf, 0, 10^-8, -Inf),
                                       upper=c(Inf, Inf, Inf, Inf),
                                       alpha=0.05, nStart=NA, nEnd=NA,
                                       trimAmt = 10, silent = T )

  data_plot <- renderStackedPlot(trueData = data,
```

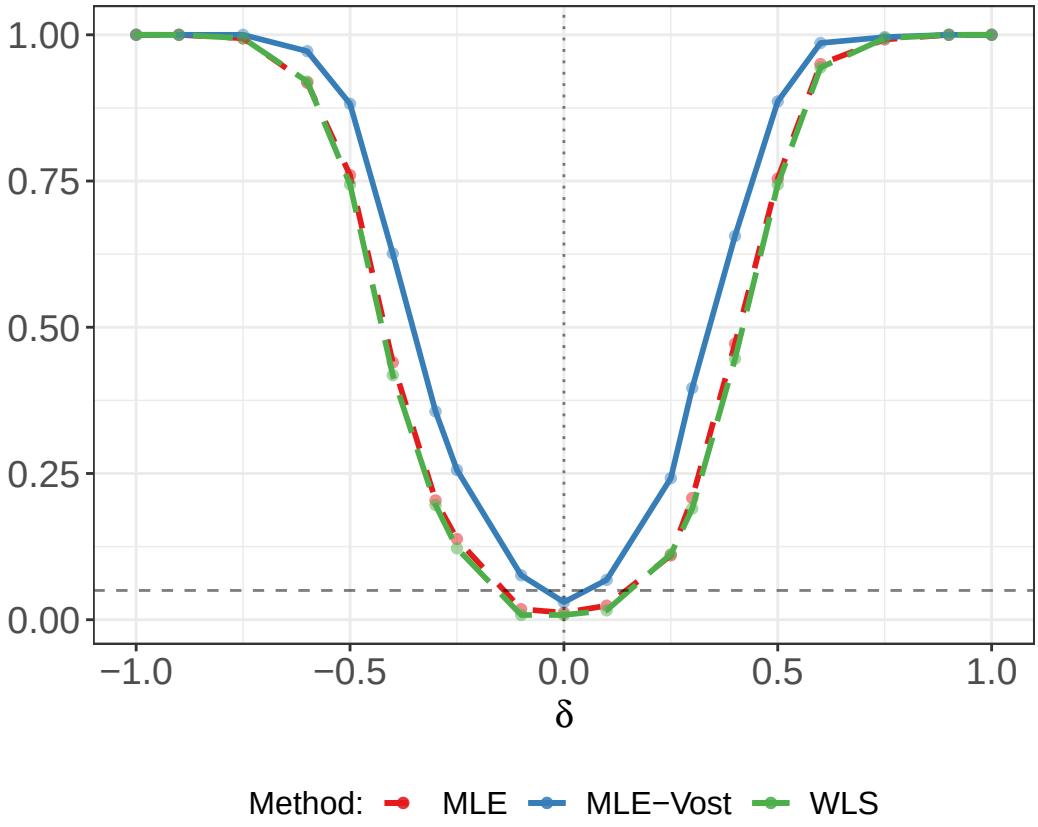

Figure 2: Power curves as a function of the change in RCA coefficient.

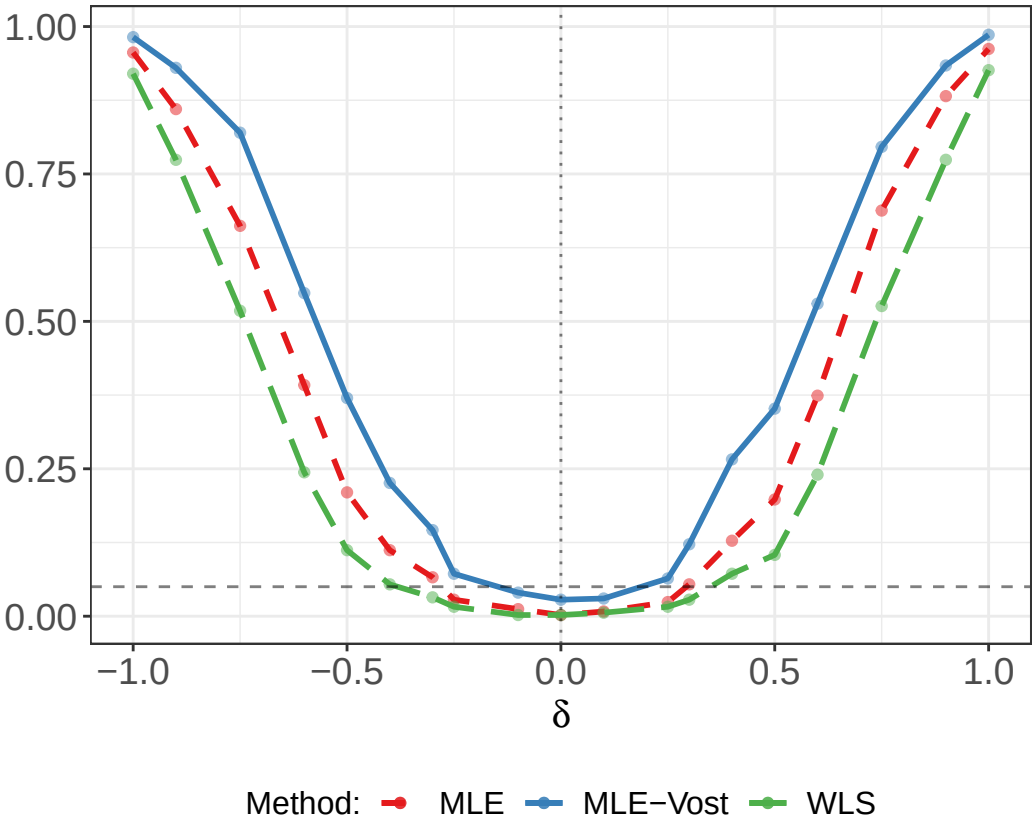

Figure 3: Power curves as a function of the change in RCA coefficient.

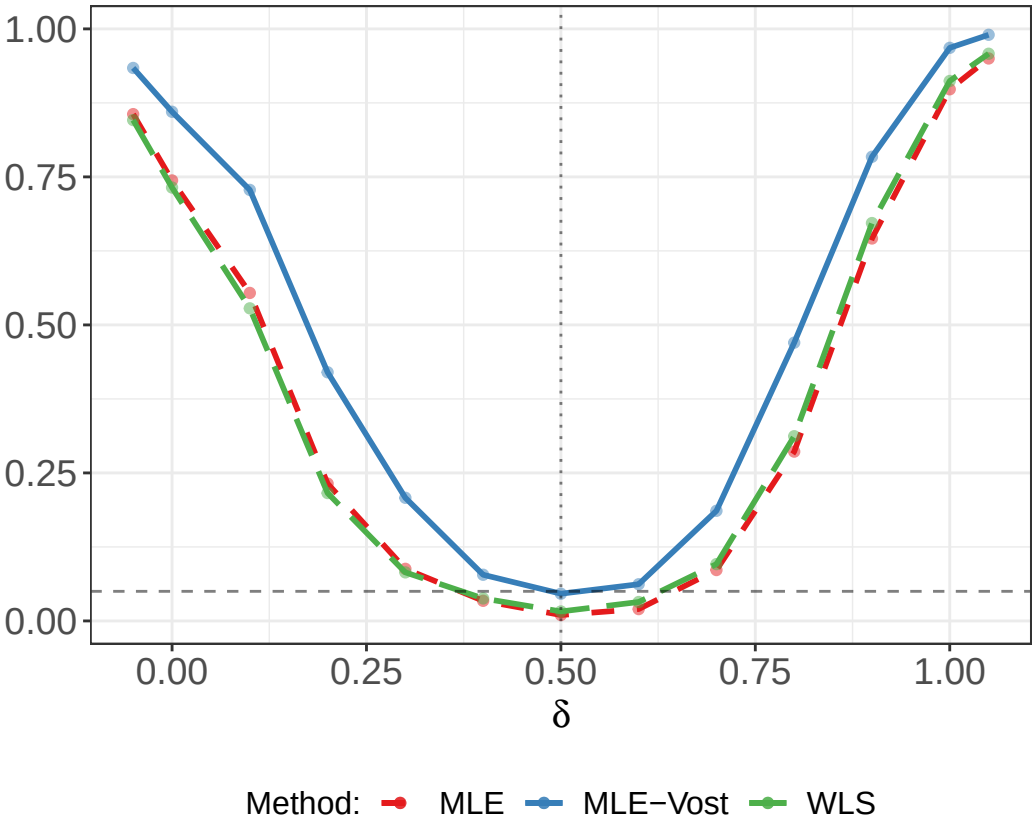

Figure 4: Power curves as a function of the change in RCA coefficient.

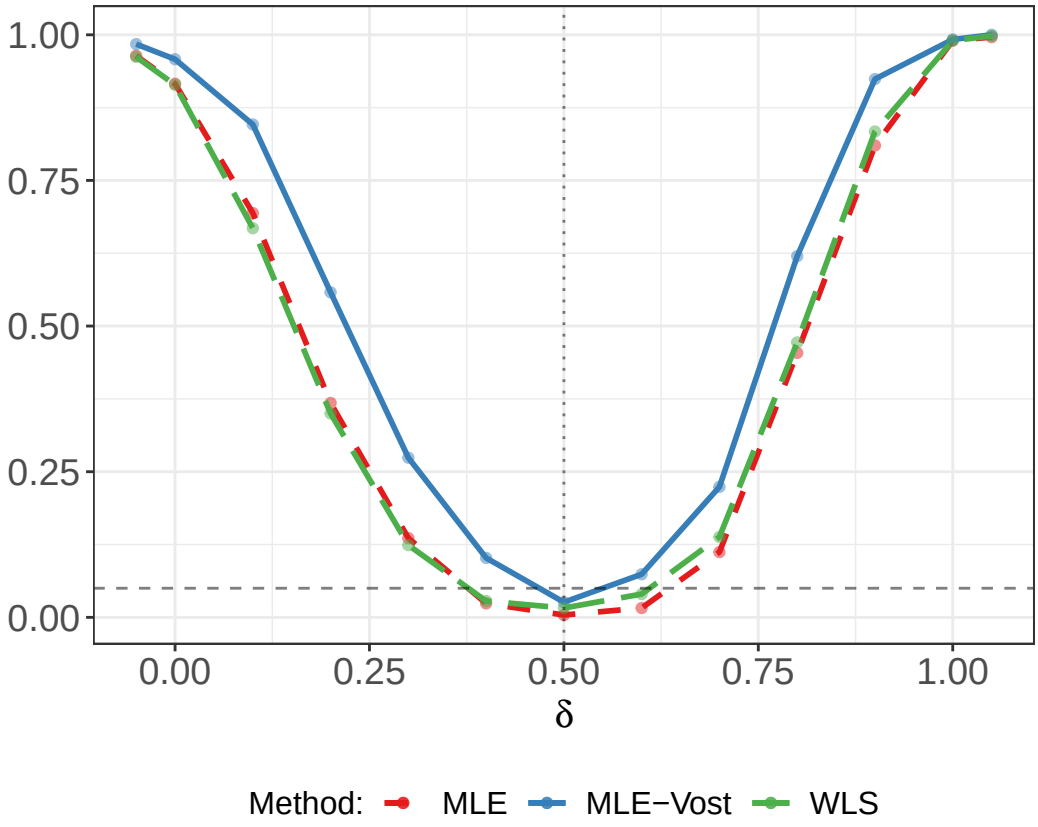

Figure 5: Power curves as a function of the change in RCA coefficient.

```
                    parCPData = result_Vost[,c(1:2,7:10)],
                    title = NULL,
                    subtitle = NULL,
                    varPlots = FALSE)

  ## Save data
  housingDataList <- append(housingDataList, list(list(result_Vost,data_plot)))
}
```

The results were as follows.

```
housingDataList[[1]][[2]]
```

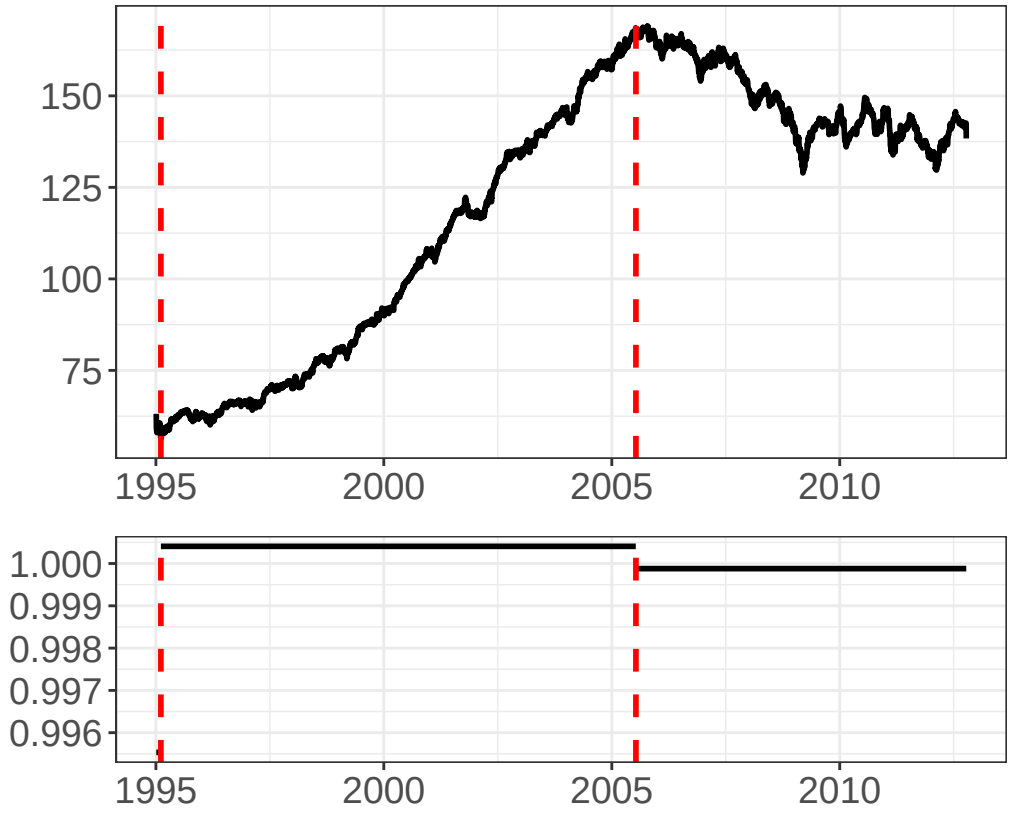

Figure 6: Daily US housing price indices in Boston.

```
housingDataList[[2]][[2]]
```

The UK covid-19 figures were built using the subsequent code.

```
UKCovidList <- list()
for(nation in c('England','Northern Ireland','Scotland','Wales')){

  ## Load data
```

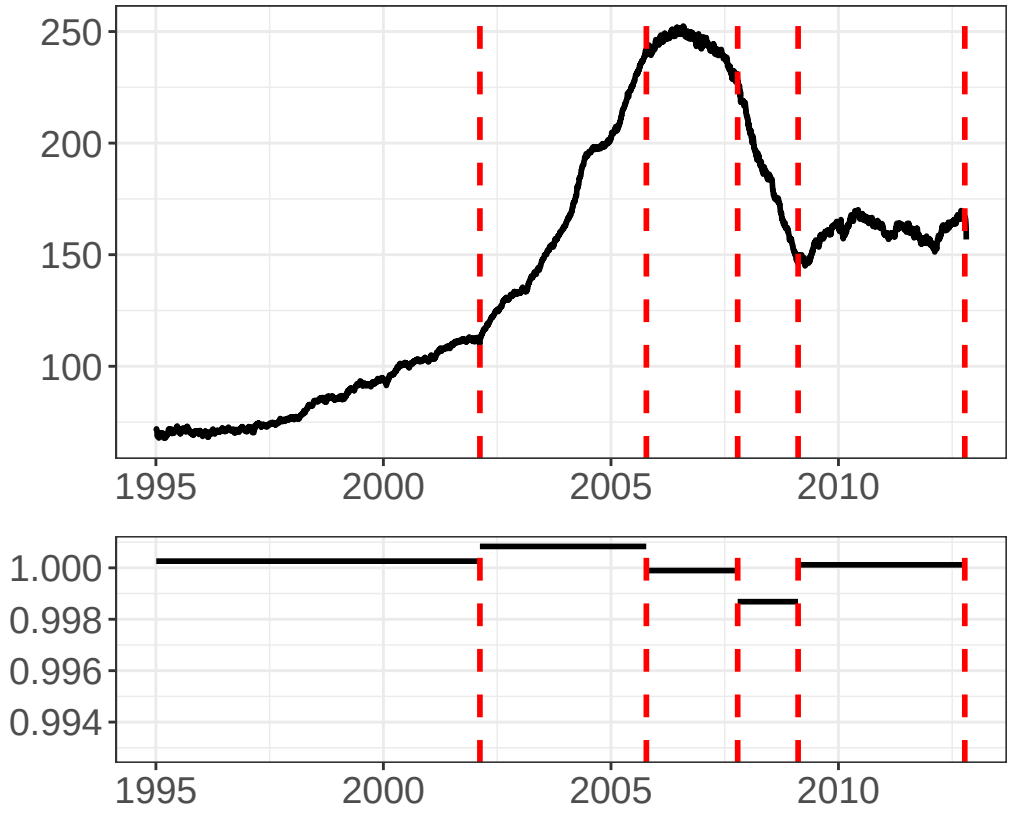

Figure 7: Daily US housing price indices in Los Angeles.

```
  data <- UKcovid[UKcovid$nation==nation,c(2,3)]
  data <- data[order(data$date),]

  ## Detect CP
  set.seed(12345)
  result_Vost <- binarySegmentationCPDetection(fullData=data,
                                       method='Vostrikova',
                                       lower=c(-Inf, 0, 10^-8, -Inf),
                                       upper=c(Inf, Inf, Inf, Inf),
                                       alpha=0.05,
                                       trimAmt = 10, silent = T )
  data_plot <- renderStackedPlot(trueData = data,
                      parCPData = result_Vost[,c(1:2,7:10)],
                      title = NULL,
                      subtitle = NULL,
                      varPlots = FALSE)

  ## Save data
  UKCovidList <- append(UKCovidList, list(list(result_Vost,data_plot)))
}
```

With plots produced as follows.

```
UKCovidList[[1]][[2]]
```

```
UKCovidList[[2]][[2]]
```

```
UKCovidList[[3]][[2]]
```

```
UKCovidList[[4]][[2]]
```

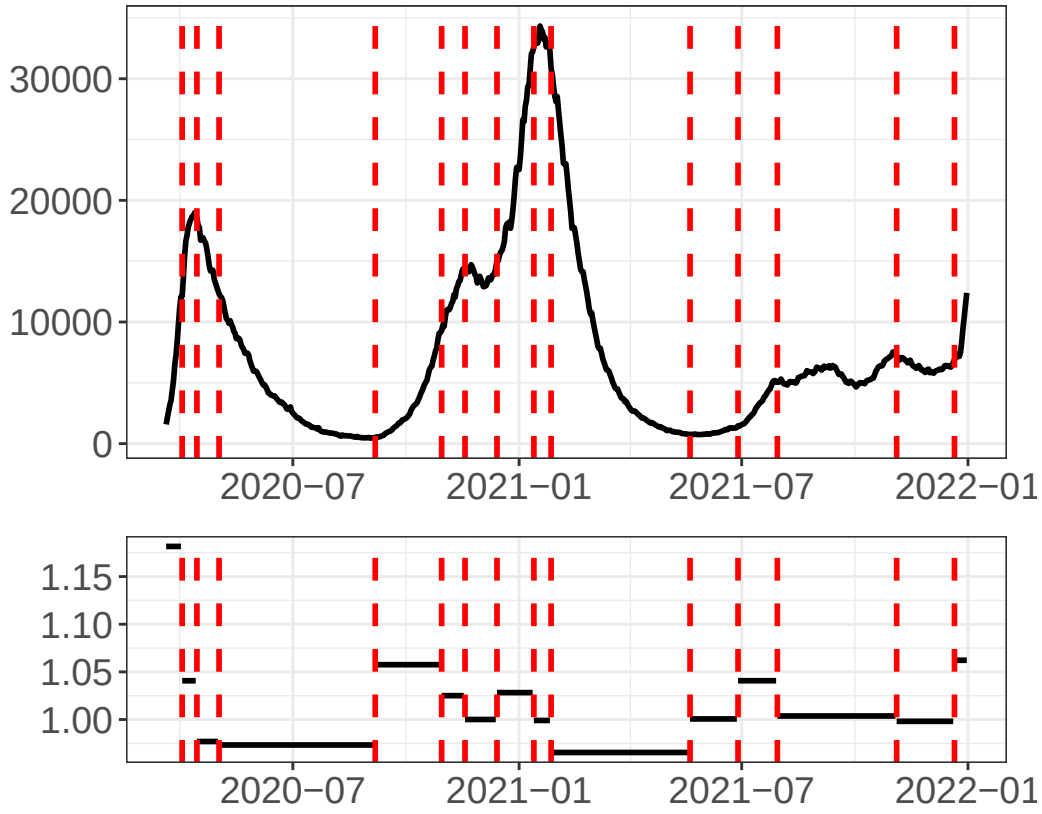

Figure 8: Daily Covid-19 patients hospitalised in England.

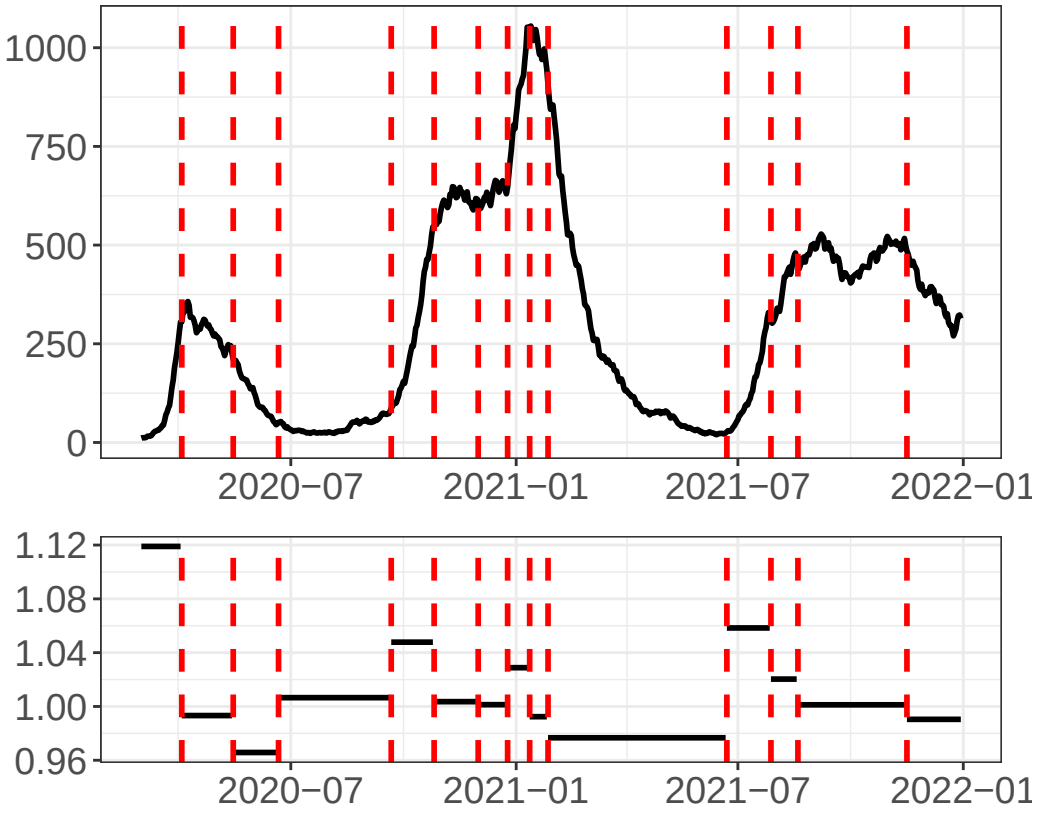

Figure 9: Daily Covid-19 patients hospitalised in Northern Ireland.

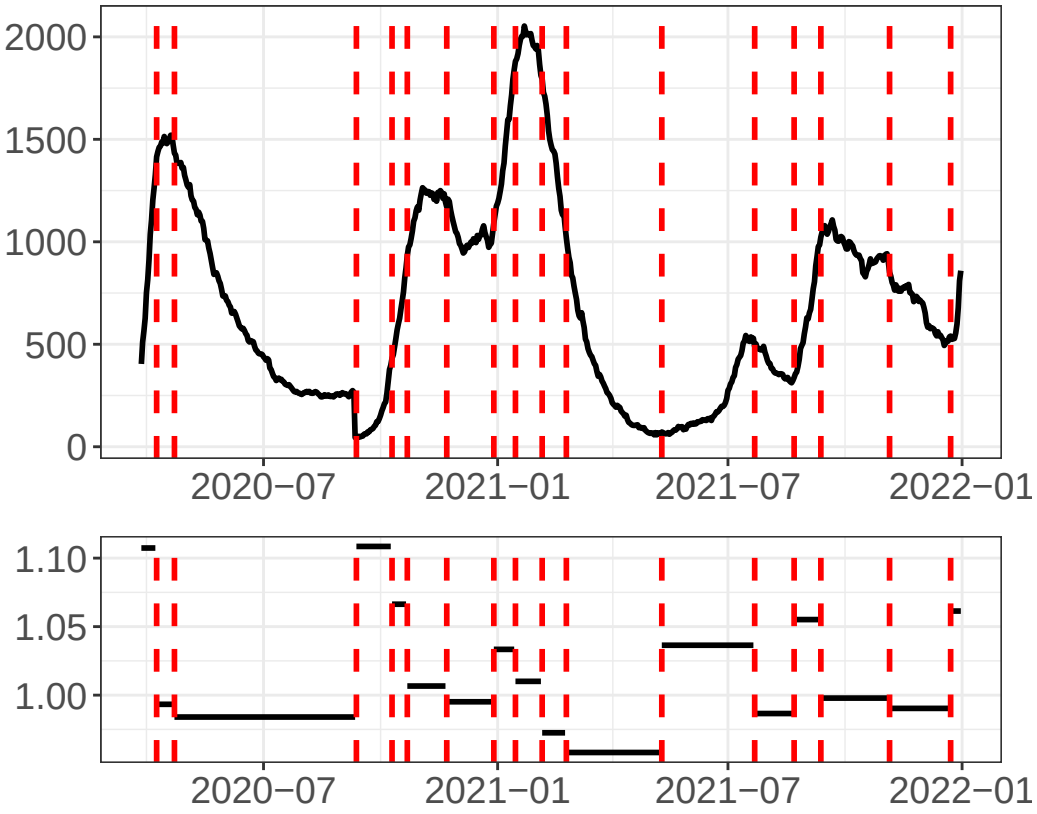

Figure 10: Daily Covid-19 patients hospitalised in Scotland.

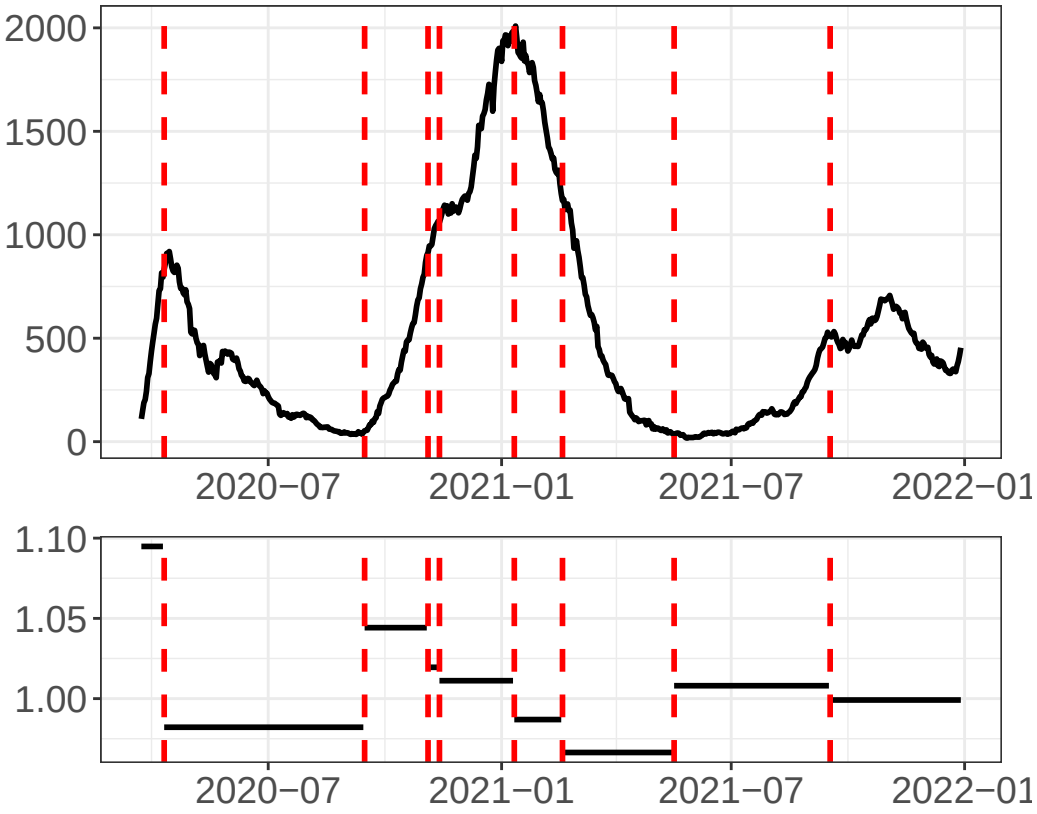

Figure 11: Daily Covid-19 patients hospitalised in Wales.