Chapter 10 Infant Mortality in Sweden

John Bryant and Junni L. Zhang

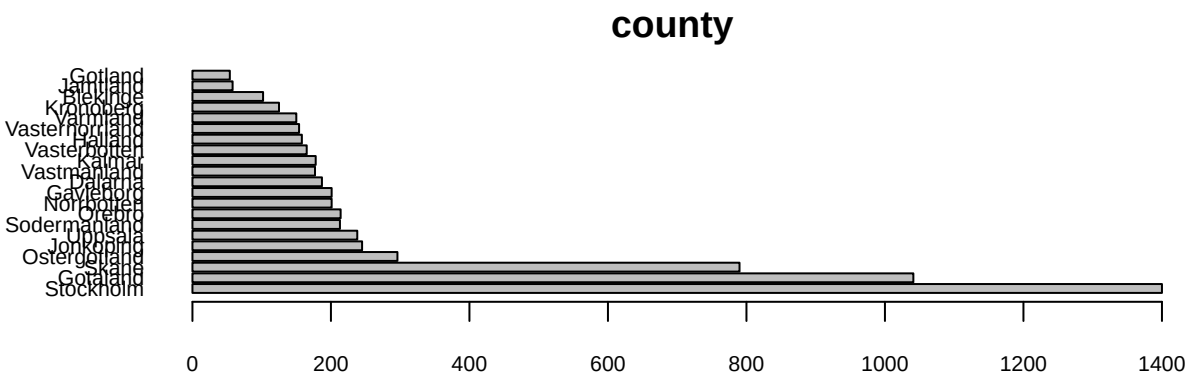
2018-08-05

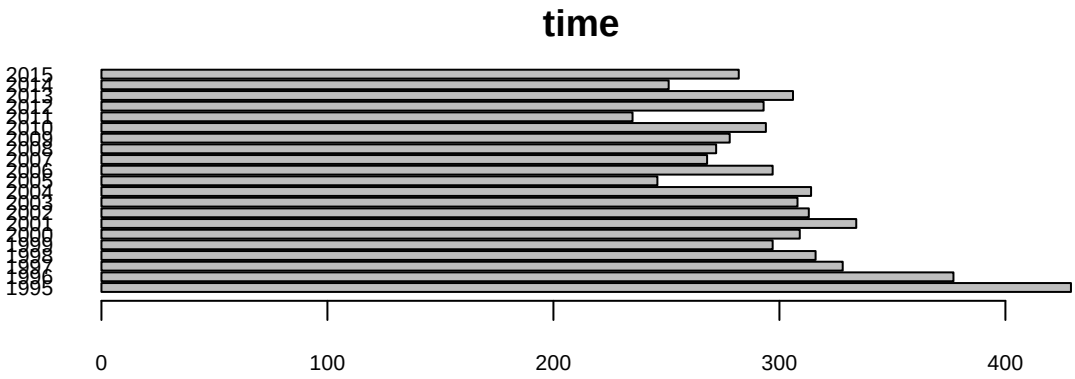
```
library(bdefdata)
library(demest)
library(dplyr)
library(purrr)
library(tidyr)
library(latticeExtra)
```

```
deaths <- bdefdata::sweden_deaths %>%
  Counts(dimscales = c(time = "Intervals"))
summary(deaths)
```

```
##
## name:     county     time
## length:   21         21
## dimtype:  state      time
## dimscale: Categories Intervals
## first:    Stockholm  1995
## last:     Gotland    2015
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    5.00    9.00   14.39   13.00   96.00
```

```
plot(deaths, cex.axis = 0.7, cex.names = 0.7)
```

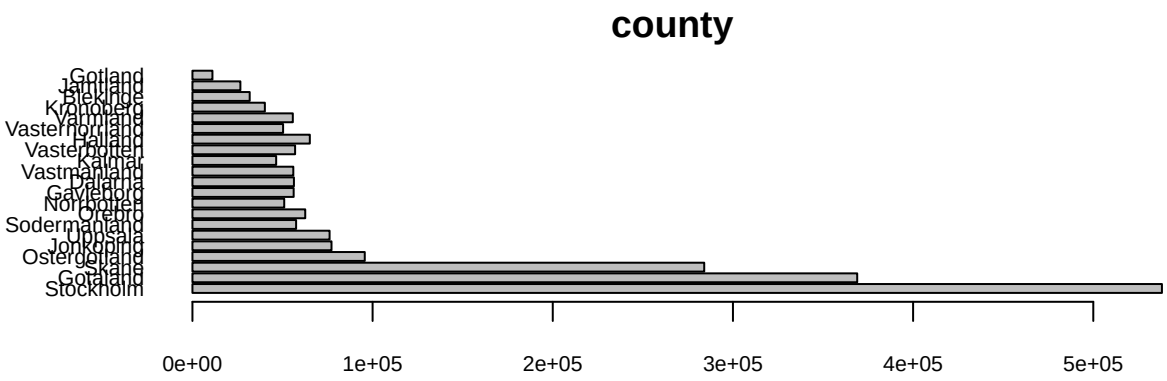


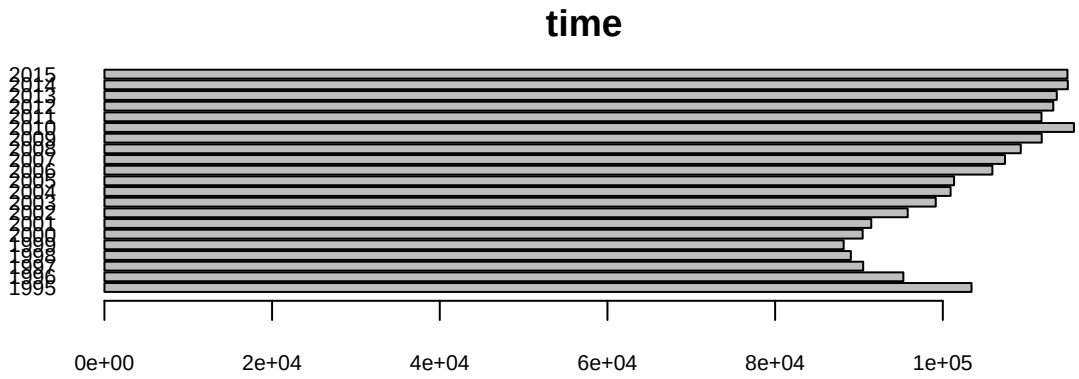



```
births <- bdefdata::sweden_births %>%
  Counts(dimscales = c(time = "Intervals"))
summary(births)
```

```
##
## name:     county     time
## length:   21         21
## dimtype:  state      time
## dimscale: Categories Intervals
## first:    Stockholm  1995
## last:     Gotland    2015
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     465    2325    2715    4907    3513   29619
```

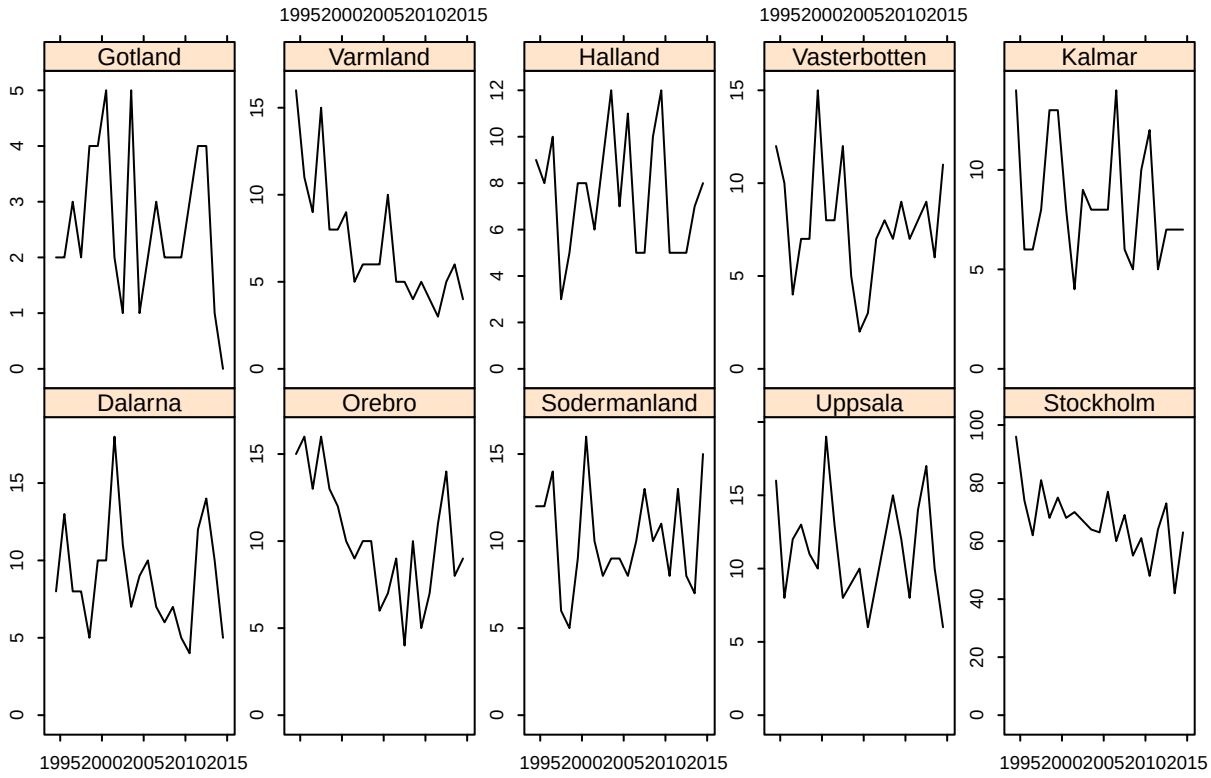
```
plot(births, cex.axis = 0.7, cex.names = 0.7)
```



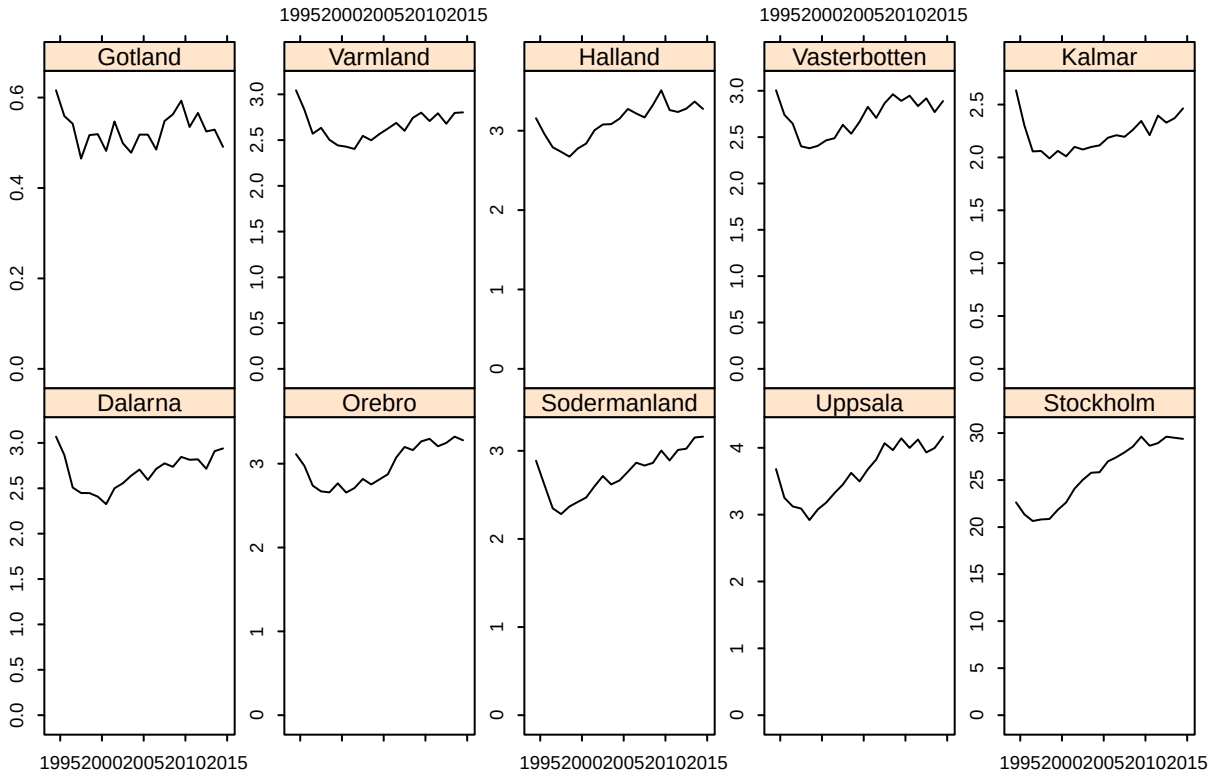



```
ten_counties <- c("Gotland", "Varmland", "Halland", "Vasterbotten", "Kalmar",
                  "Dalarna", "Orebro", "Sodermanland", "Uppsala", "Stockholm")
deaths_ten <- deaths %>%
  slab(dimension = "county", elements = ten_counties)
births_ten <- births %>%
  slab(dimension = "county", elements = ten_counties)
```

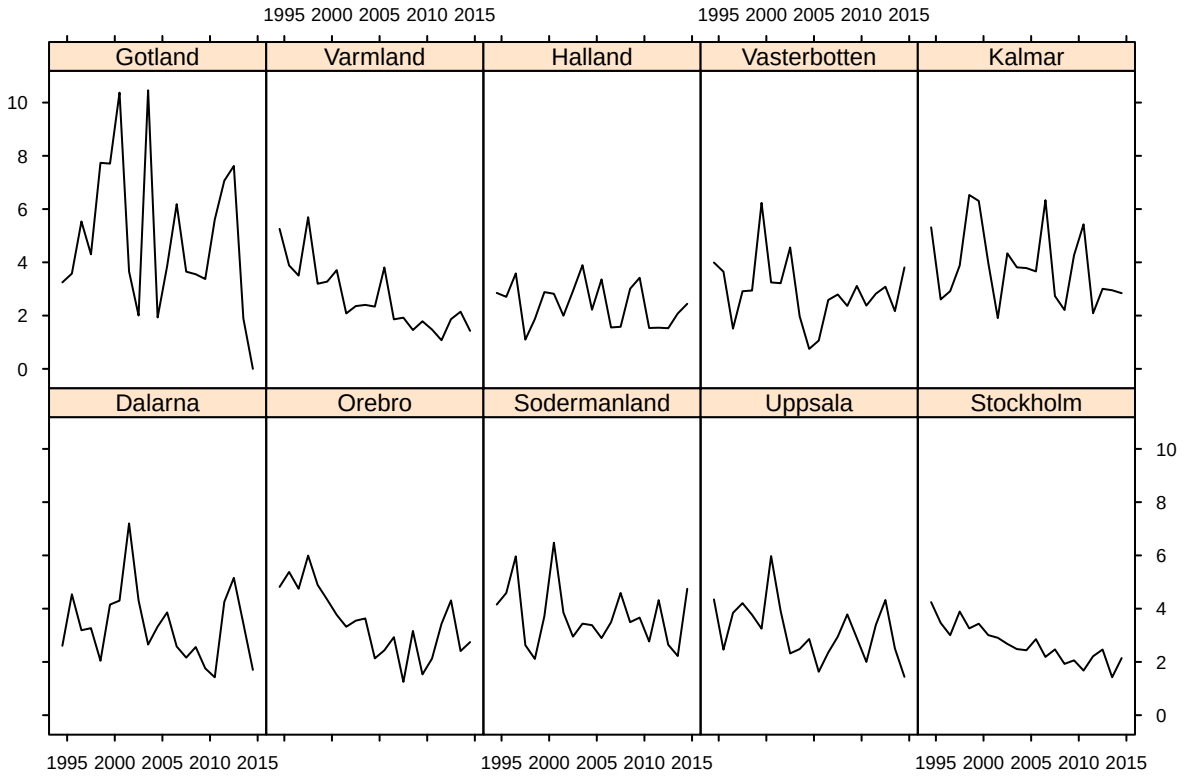
```
dplot(~ time | county,
      data = deaths_ten,
      midpoints = "time",
      scales = list(tck = 0.4,
                    y = list(relation = "free")),
      ylim = c(0, NA),
      layout = c(NA, 2),
      col = "black",
      xlab = "",
      ylab = "",
      par.settings = list(fontsize = list(text = 9)),
      as.table = TRUE)
```




```
dplot(~ time | county,
      data = births_ten / 1000,
      midpoints = "time",
      scales = list(tck = 0.4,
                    y = list(relation = "free")),
      ylim = c(0, NA),
      layout = c(NA, 2),
      col = "black",
      xlab = "",
      ylab = "",
      par.settings = list(fontsize = list(text = 9)),
      as.table = TRUE)
```




```
dplot(~ time | county,
      data = 1000 * deaths_ten / births_ten,
      midpoints = "time",
      col = "black",
      layout = c(NA, 2),
      scales = list(tck = 0.4),
      xlab = "",
      ylab = "",
      par.settings = list(fontsize = list(text = 9)),
      as.table = TRUE)
```




```
model_initial <- Model(y ~ Binomial(mean ~ county + time),
                       time ~ DLM(damp = NULL),
                       jump = 0.11)
model_initial
```

```
## An object of class "SpecBinomialVarying"
##
##             y[i] ~ binomial(exposure[i], prob[i])
##   logit(prob[i]) ~ N(county[j[i]] + time[j[i]], sd^2)
##
##          time[j] = level[j] + error[j]
##         level[j] = level[j-1] + trend[j-1] + errorLevel[j]
##         trend[j] = trend[j-1] + errorTrend[j]
##         level[0] ~ N(0, NA)
##         trend[0] ~ N(0, NA)
##    errorLevel[j] ~ N(0, scaleLevel^2)
##    errorTrend[j] ~ N(0, scaleTrend^2)
##       scaleLevel ~ trunc-half-t(7, NA, NA)
##       scaleTrend ~ trunc-half-t(7, NA, NA)
##         error[j] ~ N(0, scaleError^2)
##       scaleError ~ trunc-half-t(7, NA, NA)
##
##               sd ~ trunc-half-t(7, 1, 5.408)
##
## jump:
##     prob[i]: 0.11
```

```
set.seed(0)
```

```
if (!file.exists("sweden_initial.est")) {
  estimateModel(model_initial,
                y = deaths,
                exposure = births,
                filename = "sweden_initial.est",
                nBurnin = 100000,
                nSim = 100000,
                nChain = 4,
                nThin = 200)
}
```

```
fetchSummary("sweden_initial.est")
```

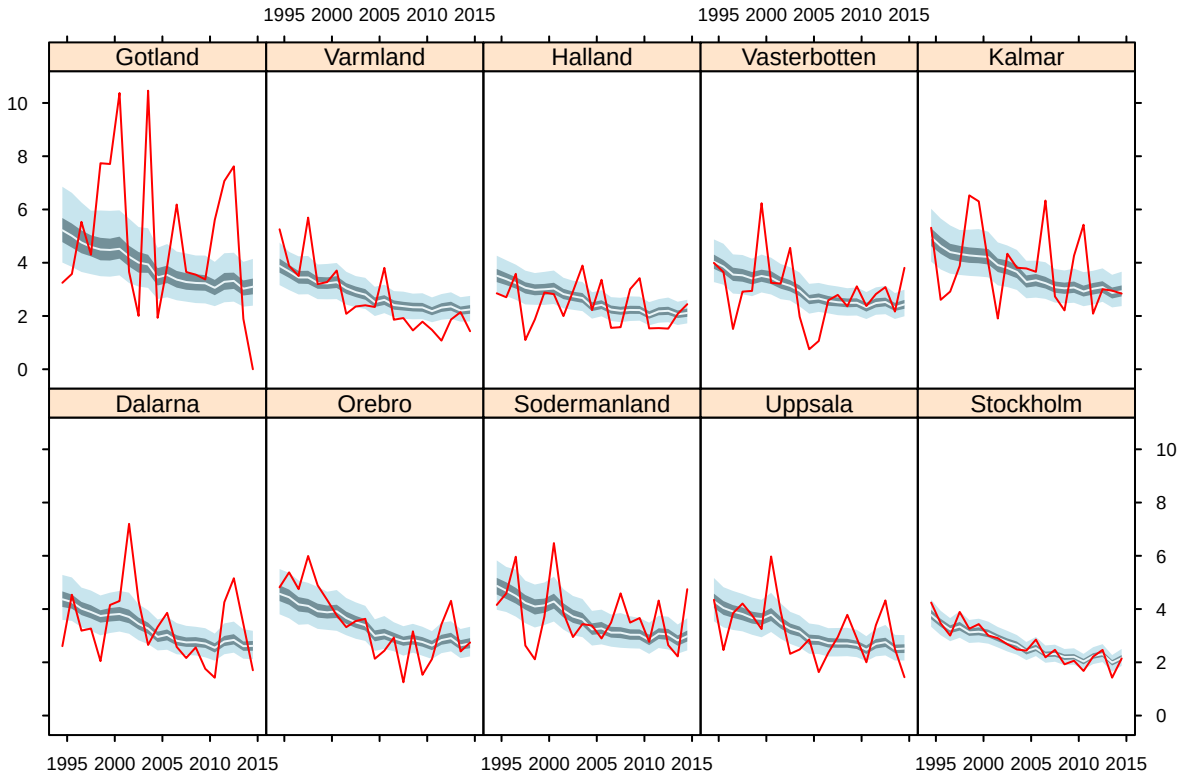
```
## ----------------------------------------------------
## model:
## y ~ Binomial(mean ~ county + time),
## time ~ DLM(damp = NULL),
## 0.11
## dimensions: county, time
## ----------------------------------------------------
## y:
## Object of class "Counts"
## dimensions: county, time
## n cells: 441,  n missing: 0, integers: TRUE, n zeros: 3, median: 9
## ----------------------------------------------------
## MCMC statistics:
## nBurnin: 100000,  nSim: 100000,  nChain: 4,  nThin: 200,  nIteration: 2000
##
## Metropolis-Hastings updates:
##                        jump acceptance autocorr
## model.likelihood.prob 0.11      0.511    0.104
##
## parameters:
##                                  Rhat    2.5%     50%   97.5% length
## model.likelihood.prob           1.01 0.00204 0.00305 0.00488    441
## model.prior.mean                1.00   -6.16   -5.79   -5.33    441
## model.prior.sd                  1.01   0.031  0.0646   0.106      1
## model.hyper.county.scaleError   1.00  0.0878   0.132   0.185      1
## model.hyper.time.scaleLevel     1.00 0.00145  0.0249  0.0656      1
## model.hyper.time.scaleTrend     1.00 0.00185  0.0138  0.0299      1
## model.hyper.time.scaleError     1.00 0.00161  0.0265  0.0708      1
## ----------------------------------------------------
```

```
county_names_reversed <- rev(dimnames(births)$county) # to match book
```

```
rates_modelled_initial <- fetch("sweden_initial.est",
                                where = c("model", "likelihood", "prob")) %>%
  slab(dimension = "county", elements = county_names_reversed)
```

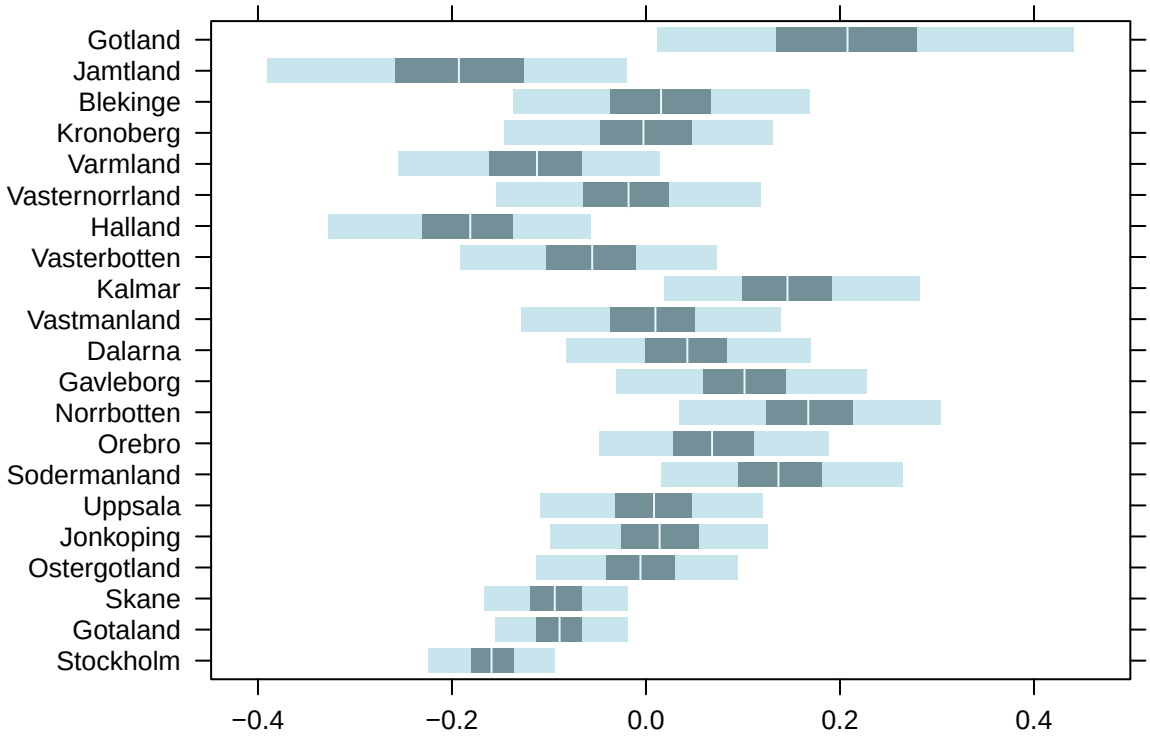
```
y <- fetch("sweden_initial.est", where = "y")
exposure <- fetch("sweden_initial.est", where = "exposure")
rates_direct <- (y / exposure) %>%
  slab(dimension = "county", elements = county_names_reversed)
```

```
dplot( ~ time | county,
       data = 1000 * rates_modelled_initial,
       subarray = county %in% ten_counties,
       midpoints = "time",
       col = "light blue",
       as.table = TRUE,
       layout = c(NA, 2),
       scales = list(tck = 0.4),
       xlab = "",
       ylab = "",
       par.settings = list(fontsize = list(text = 9)),
       overlay = list(values = 1000 * rates_direct,
                      col = "red"))
```

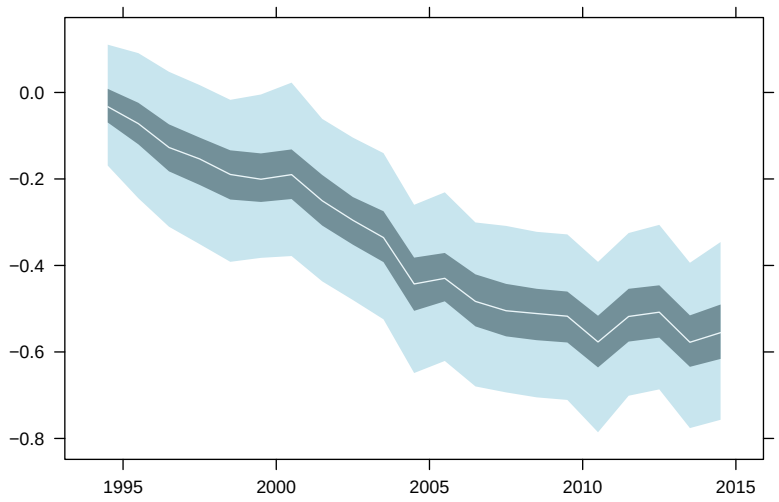



```
intercept_initial <- fetch("sweden_initial.est",
                           where = c("model", "prior", "(Intercept)"))
intercept_initial <- addDimension(intercept_initial,
                                  name = "intercept",
                                  labels = "x")
dplot( ~ intercept,
       data = intercept_initial,
       col = "light blue",
       as.table = TRUE,
       xlab = "",
       ylab = "",
       par.settings = list(fontsize = list(text = 9)),
       scales = list(x = list(tck = 0, labels = "")))
```

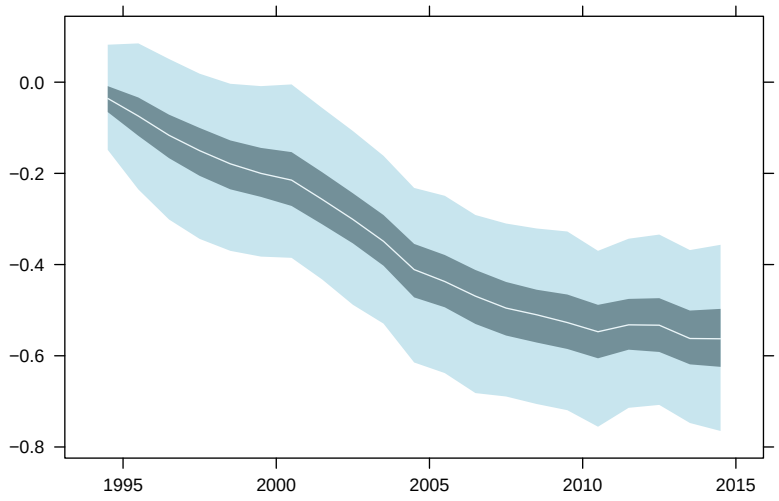
```
county_effect_initial <- fetch("sweden_initial.est",
                               where = c("model", "prior", "county"))
dplot( ~ county,
       data = county_effect_initial,
       col = "light blue",
       horizontal = TRUE,
       xlab = "",
       ylab = "")
```



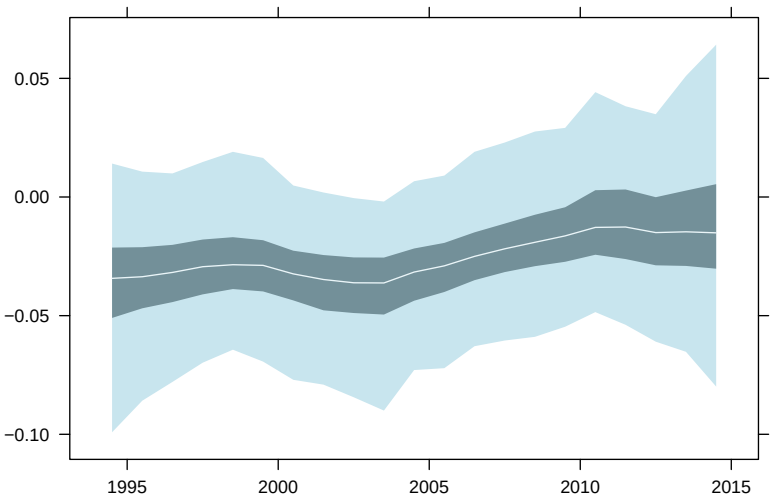

```
time_effect_initial <- fetch("sweden_initial.est",
                             where = c("model", "prior", "time"))
dplot( ~ time,
       data = time_effect_initial,
       midpoints = "time",
       col = "light blue",
       xlab = "",
       ylab = "")
```



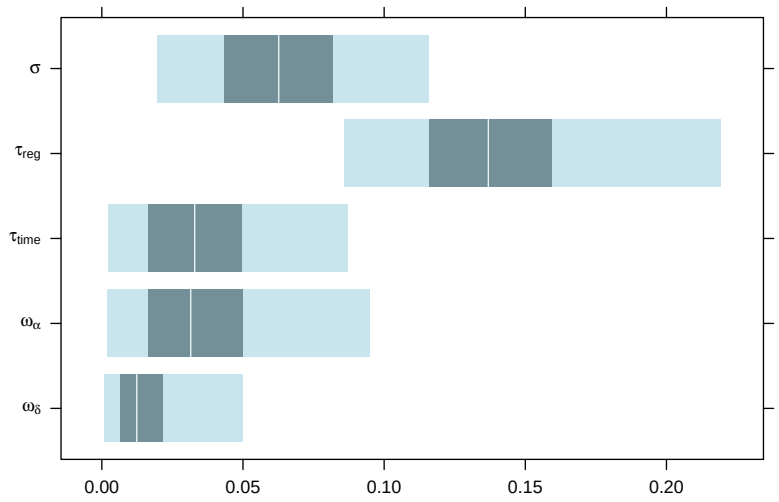

```
level_initial <- fetch("sweden_initial.est",
                       where = c("model", "hyper", "time", "level"))
dplot( ~ time,
       data = level_initial,
       midpoints = "time",
       col = "light blue",
       xlab = "",
       ylab = "")
```



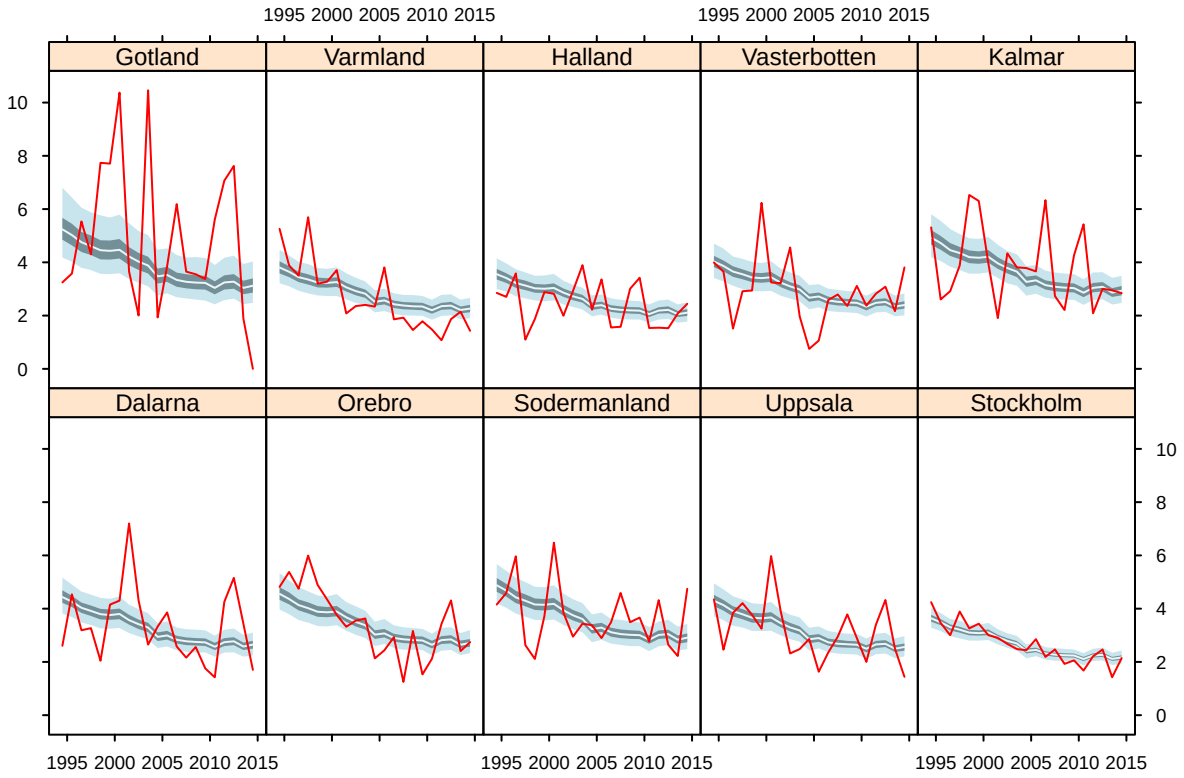

```
trend_initial <- fetch("sweden_initial.est",
                       where = c("model", "hyper", "time", "trend"))
dplot( ~ time,
       data = trend_initial,
       midpoints = "time",
       col = "light blue",
       xlab = "",
       ylab = "")
```




```
sigma_initial <- fetch("sweden_initial.est",
                       where = c("model", "prior", "sd"))
tau_county_initial <- fetch("sweden_initial.est",
                            where = c("model", "hyper", "county", "scaleError"))
tau_time_initial <- fetch("sweden_initial.est",
                          where = c("model", "hyper", "time", "scaleError"))
omega_level_initial <- fetch("sweden_initial.est",
                             where = c("model", "hyper", "time", "scaleLevel"))
omega_trend_initial <- fetch("sweden_initial.est",
                             where = c("model", "hyper", "time", "scaleTrend"))
sd_initial <- dbind(omega_trend_initial,
                    omega_level_initial,
                    tau_time_initial,
                    tau_county_initial,
                    sigma_initial,
                    along = "variant")

yscale_components <- function(...) {
  ans <- lattice:::yscale.components.default(...)
  ans$left$labels$labels <- c(expression(omega[delta]),
                              expression(omega[alpha]),
                              expression(tau[time]),
                              expression(tau[reg]),
                              expression(sigma))
  ans
}

dplot(value ~ variant,
      data = sd_initial,
      col = "light blue",
      horizontal = TRUE,
      as.table = TRUE,
      yscale.components = yscale_components,
      xlab = "",
      ylab = "")
```

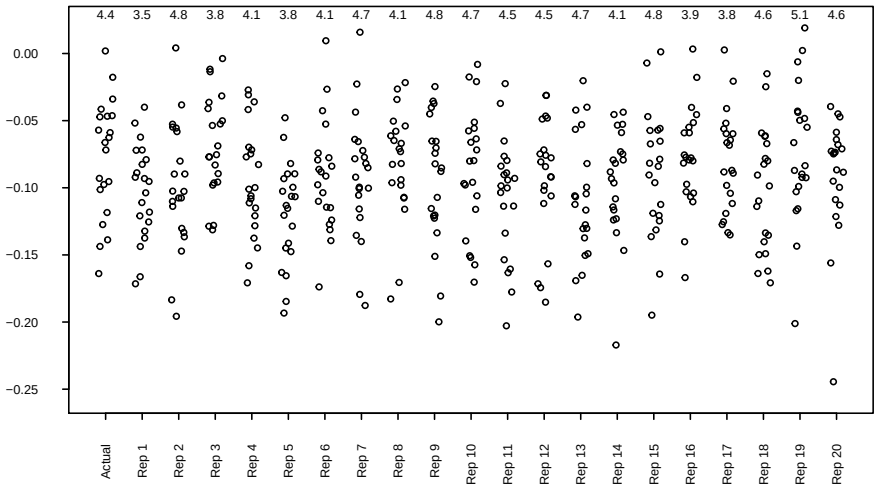



```
means_initial <- fetch("sweden_initial.est", where = c("model", "prior", "mean"))
inv_logit <- function(x) exp(x) / (1 + exp(x))
predicted_rates_initial <- inv_logit(means_initial)
predicted_rates_initial_ten_counties <- predicted_rates_initial %>%
  slab(dimension = "county", elements = county_names_reversed)

dplot( ~ time | county,
       data = 1000 * predicted_rates_initial_ten_counties,
       subarray = county %in% ten_counties,
       midpoints = "time",
       col = "light blue",
       as.table = TRUE,
       layout = c(NA, 2),
       xlab = "",
       ylab = "",
       scales = list(tck = 0.4),
       par.settings = list(fontsize = list(text = 9)),
       overlay = list(values = 1000 * rates_direct,
                      col = "red"))
```



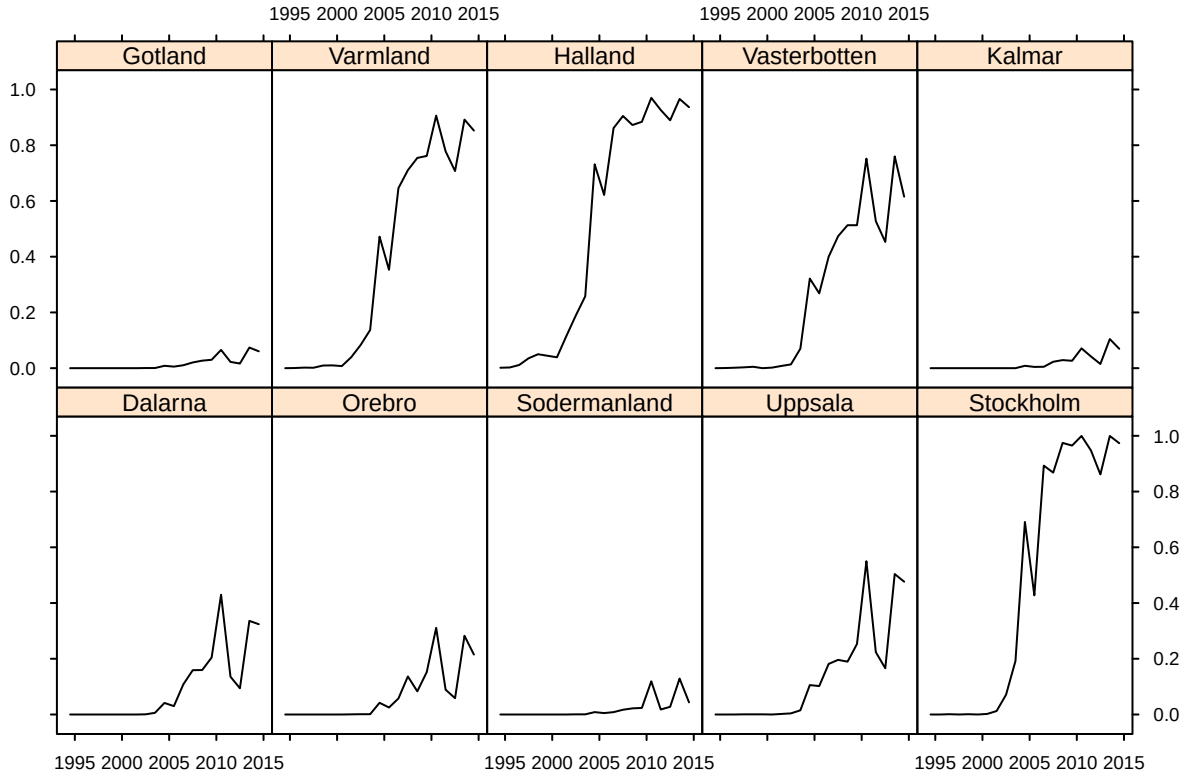

```
set.seed(0)
n_iter <- 20

predicted_replicate <- predicted_rates_initial %>%
  thinIterations(n = n_iter)

y_replicate <- rbinom(n = length(predicted_replicate),
                      prob = predicted_replicate,
                      size = exposure) %>%
  array(dim = dim(predicted_replicate),
        dimnames = dimnames(predicted_replicate)) %>%
  Counts(dimscales = c(time = "Intervals"))

y_all <- dbind(y, y_replicate, along = "iteration")

slopes <- (y_all / exposure) %>%
  as.data.frame(direction = "long", stringsAsFactors = FALSE) %>%
  as_tibble() %>%
  mutate(iteration = as.integer(iteration)) %>%
  mutate(time = as.integer((time))) %>%
  nest(time, value) %>%
  mutate(model = map(data, ~lm(value ~ time, data = .)),
         coef = map(model, coef),
         slope = 1000 * map_dbl(coef, "time"))

sd <- slopes %>%
  group_by(iteration) %>%
  summarise(sd = sd(slope)) %>%
  mutate(sd = sprintf("%2.1f", 100 * sd)) %>%
```

```
  pull(sd)

ylim <- slopes %>%
  filter(is.finite(slope)) %>%
  pull(slope) %>%
  range() %>%
  `*`(c(1.05, 1.25))

plot(slope ~ jitter(iteration),
     data = slopes,
     col = "black",
     cex = 0.5,
     ylim = ylim,
     xaxt = "n",
     cex.axis = 0.5,
     cex.lab = 0.7,
     las = 2,
     xlab = "",
     ylab = "",
     tcl = -0.2)
axis(side = 1,
     at = seq_len(n_iter + 1),
     las = 2,
     labels = c("Actual", paste("Rep", seq_len(n_iter))),
     tcl = -0.2,
     cex.axis = 0.5,
     cex.lab = 0.7)
mtext(text = sd,
      line = -0.6,
      at = 1:21,
      cex = 0.5)
```




```
cutoff <- 0.0025
prob <- fetch(filename = "sweden_initial.est",
              where = c("model", "likelihood", "prob"))
get_propn_below_cutoff <- function(x) mean(x < cutoff)
propn_below_cutoff <- collapseIterations(prob, FUN = get_propn_below_cutoff) %>%
  slab(dimension = "county", elements = county_names_reversed)
```

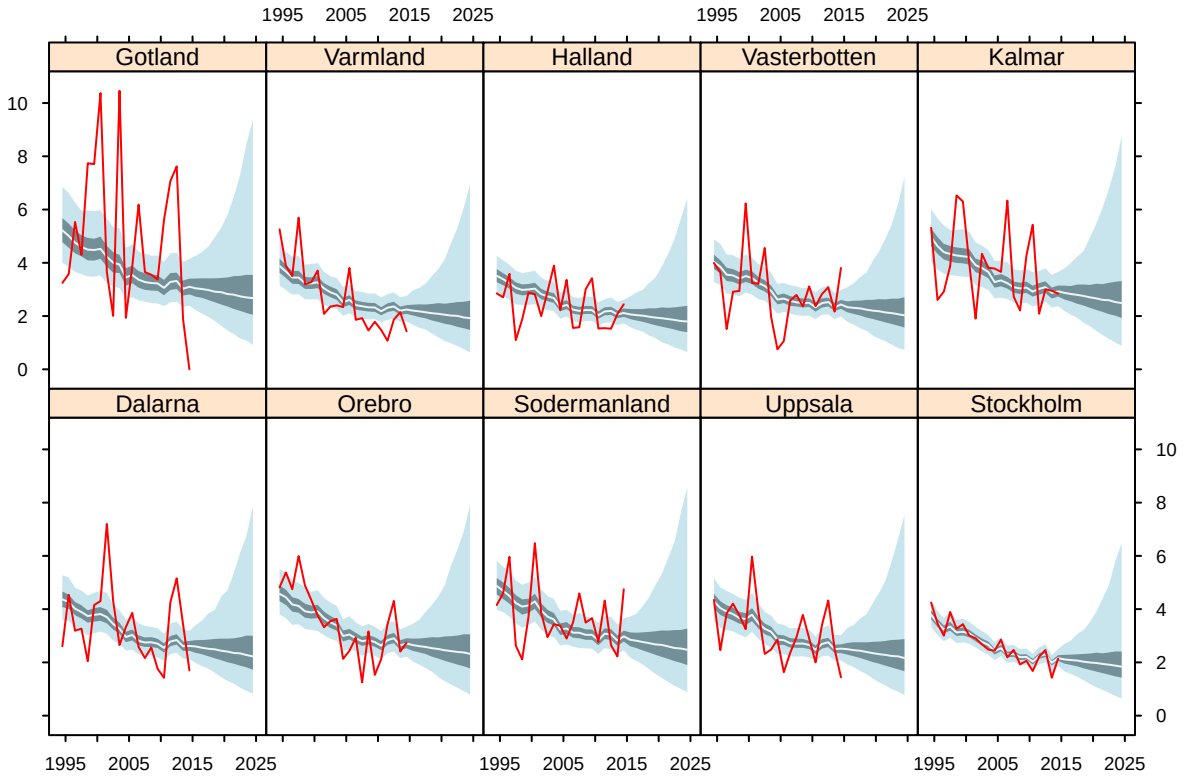
```
dplot( ~ time | county,
       data = propn_below_cutoff,
       subarray = county %in% ten_counties,
       midpoints = "time",
       col = "black",
       as.table = TRUE,
       layout = c(NA, 2),
       scales = list(tck = 0.4),
       par.settings = list(fontsize = list(text = 9)),
       xlab = "",
       ylab = "")
```




```
set.seed(0)
predictModel(filenameEst = "sweden_initial.est",
             filenamePred = "sweden_initial.pred",
             n = 10)
```

```
rates_modelled_all_initial <- fetchBoth(filenameEst = "sweden_initial.est",
                                        filenamePred = "sweden_initial.pred",
                                        where = c("model", "likelihood", "prob")) %>%
  slab(dimension = "county", elements = ten_counties)
dplot( ~ time | county,
       data = 1000 * rates_modelled_all_initial,
       midpoints = "time",
       col = "light blue",
       as.table = TRUE,
       layout = c(NA, 2),
       xlab = "",
       ylab = "",
```

```
       scales = list(tck = 0.4),
       par.settings = list(fontsize = list(text = 9)),
       overlay = list(values = 1000 * rates_direct,
                      col = "red"))
```




```
model_revised <- Model(y ~ Binomial(mean ~ county + time),
                       time ~ DLM(level = Level(scale = HalfT(scale = 0.01)),
                                  trend = Trend(scale = HalfT(scale = 0.01)),
                                  error = Error(scale = HalfT(scale = 0.01))),
                       jump = 0.12)
model_revised
```

```
## An object of class "SpecBinomialVarying"
##
##             y[i] ~ binomial(exposure[i], prob[i])
##   logit(prob[i]) ~ N(county[j[i]] + time[j[i]], sd^2)
##
##          time[j] = level[j] + error[j]
##         level[j] = level[j-1] + trend[j-1] + errorLevel[j]
##         trend[j] = damp * trend[j-1] + errorTrend[j]
##         level[0] ~ N(0, NA)
##         trend[0] ~ N(0, NA)
##    dampTransform = (damp-0.8)/(1-0.8)
##    dampTransform ~ Beta(2,2)
##    errorLevel[j] ~ N(0, scaleLevel^2)
##    errorTrend[j] ~ N(0, scaleTrend^2)
##       scaleLevel ~ trunc-half-t(7, 0.01^2, 0.05408)
##       scaleTrend ~ trunc-half-t(7, 0.01^2, 0.05408)
##         error[j] ~ N(0, scaleError^2)
```

```
##       scaleError ~ trunc-half-t(7, 0.01^2, 0.05408)
##
##               sd ~ trunc-half-t(7, 1, 5.408)
##
## jump:
##     prob[i]: 0.12
```

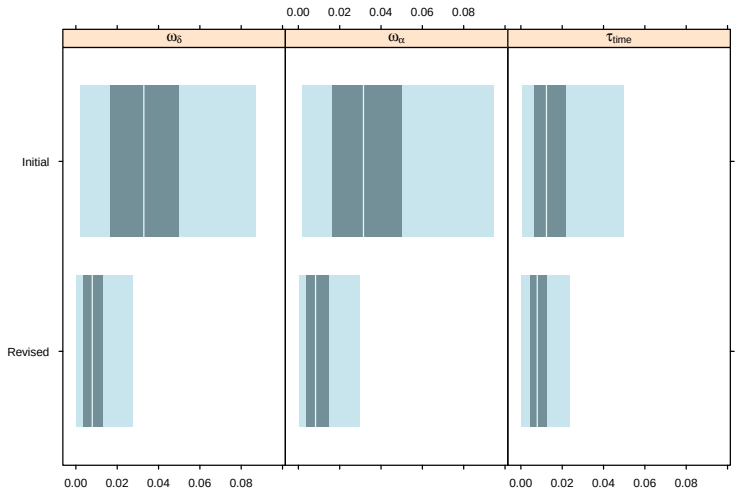
```
if (!file.exists("sweden_revised.est")) {
  estimateModel(model_revised,
                y = deaths,
                exposure = births,
                filename = "sweden_revised.est",
                nBurnin = 100000,
                nSim = 100000,
                nChain = 4,
                nThin = 200)
}
```

```
fetchSummary("sweden_revised.est")
```

```
## ----------------------------------------------------
## model:
## y ~ Binomial(mean ~ county + time),
## time ~ DLM(level = Level(scale = HalfT(scale = 0.01)), trend = Trend(scale = HalfT(scale = 0.01)),
##     error = Error(scale = HalfT(scale = 0.01))),
## 0.12
## dimensions: county, time
## ----------------------------------------------------
## y:
## Object of class "Counts"
## dimensions: county, time
## n cells: 441,  n missing: 0, integers: TRUE, n zeros: 3, median: 9
## ----------------------------------------------------
## MCMC statistics:
## nBurnin: 100000,  nSim: 100000,  nChain: 4,  nThin: 200,  nIteration: 2000
##
## Metropolis-Hastings updates:
##                         jump acceptance autocorr
## model.likelihood.prob 0.12      0.479    0.141
##
## parameters:
##                                  Rhat     2.5%     50%  97.5% length
## model.likelihood.prob           1.02  0.00206 0.00305 0.0049    441
## model.prior.mean                1.02    -6.16   -5.79  -5.34    441
## model.prior.sd                  1.03   0.0119  0.0648  0.116      1
## model.hyper.county.scaleError   1.00   0.0939    0.16  0.233      1
## model.hyper.time.scaleLevel     1.00  0.00245 0.00944 0.0202      1
## model.hyper.time.scaleTrend     1.00 0.000531 0.00603 0.0179      1
## model.hyper.time.damp           1.00    0.833   0.915  0.969      1
## model.hyper.time.scaleError     1.00   0.0011   0.009 0.0242      1
## ----------------------------------------------------
```

```
sd_initial <- dbind(tau_time = tau_time_initial,
                    omega_level = omega_level_initial,
                    omega_trend = omega_trend_initial,
```

```
                    along = "variant")
tau_time_revised <- fetch("sweden_revised.est",
                          where = c("model", "hyper", "time", "scaleError"))
omega_level_revised <- fetch("sweden_revised.est",
                             where = c("model", "hyper", "time", "scaleLevel"))
omega_trend_revised <- fetch("sweden_revised.est",
                             where = c("model", "hyper", "time", "scaleTrend"))
sd_revised <- dbind(tau_time = tau_time_revised,
                    omega_level = omega_level_revised,
                    omega_trend = omega_trend_revised,
                    along = "variant")
sd <- dbind(Revised = sd_revised,
            Initial = sd_initial,
            along = "model")

panel.labels <- c(expression(omega[delta]),
                  expression(omega[alpha]),
                  expression(tau[time]))
dplot(value ~ model | variant,
      data = sd,
      col = "light blue",
      horizontal = TRUE,
      as.table = TRUE,
      strip = strip.custom(factor.levels = panel.labels),
      scales = list(tck = 0.4),
      par.settings = list(fontsize = list(text = 9)),
      xlab = "",
      ylab = "")
```




```
set.seed(0)
predictModel(filenameEst = "sweden_revised.est",
             filenamePred = "sweden_revised.pred",
             n = 10)
```

```
rates_modelled_all_revised <- fetchBoth(filenameEst = "sweden_revised.est",
                                        filenamePred = "sweden_revised.pred",
                                        where = c("model", "likelihood", "prob")) %>%
  slab(dimension = "county", elements = ten_counties)
```

```
dplot( ~ time | county,
       data = 1000 * rates_modelled_all_revised,
       midpoints = "time",
       col = "light blue",
       as.table = TRUE,
       layout = c(NA, 2),
       xlab = "",
       ylab = "",
       scales = list(tck = 0.4),
       par.settings = list(fontsize = list(text = 9)),
       overlay = list(values = 1000 * rates_direct,
                      col = "red"))
```