Chapter 12 Life Expectancy in Portugal

John Bryant and Junni L. Zhang

2020-03-08

Setup

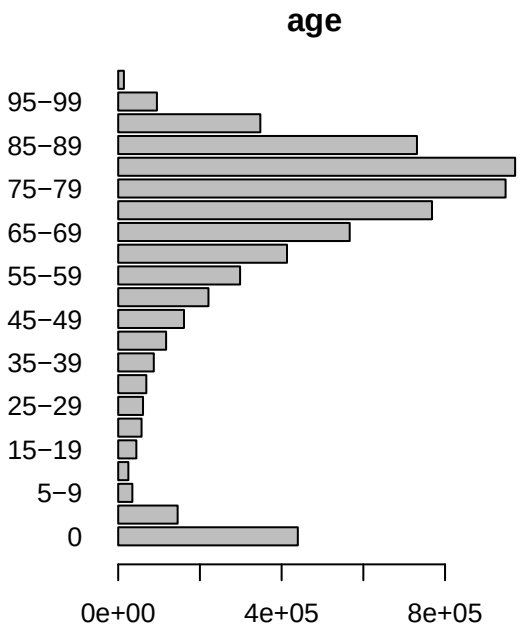
```
library(bdefdata)
library(demest)
library(demlife)
library(dplyr)
library(tidyr)
library(latticeExtra)
```

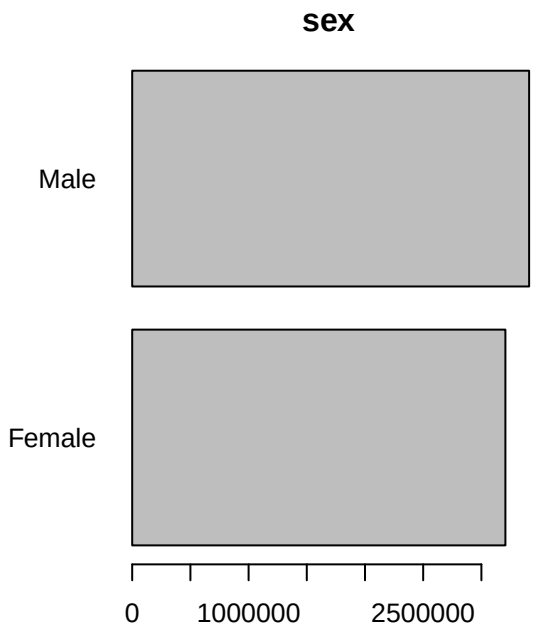
Get data

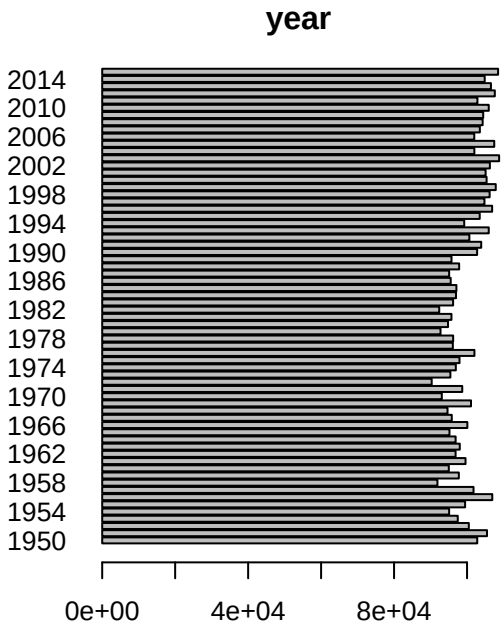
```
deaths <- bdefdata::portugal_deaths %>%
  Counts(dimscales = c(year = "Intervals"))
summary(deaths)
```

```
##
## name:     age       sex    year
## length:   22        2      66
## dimtype:  age       sex    time
## dimscale: Intervals Sexes  Intervals
## first:    0         Female 1950
## last:     100+      Male   2015
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     3.0   332.8   976.5  2278.3  3469.0 12885.0
```

```
plot(deaths)
```





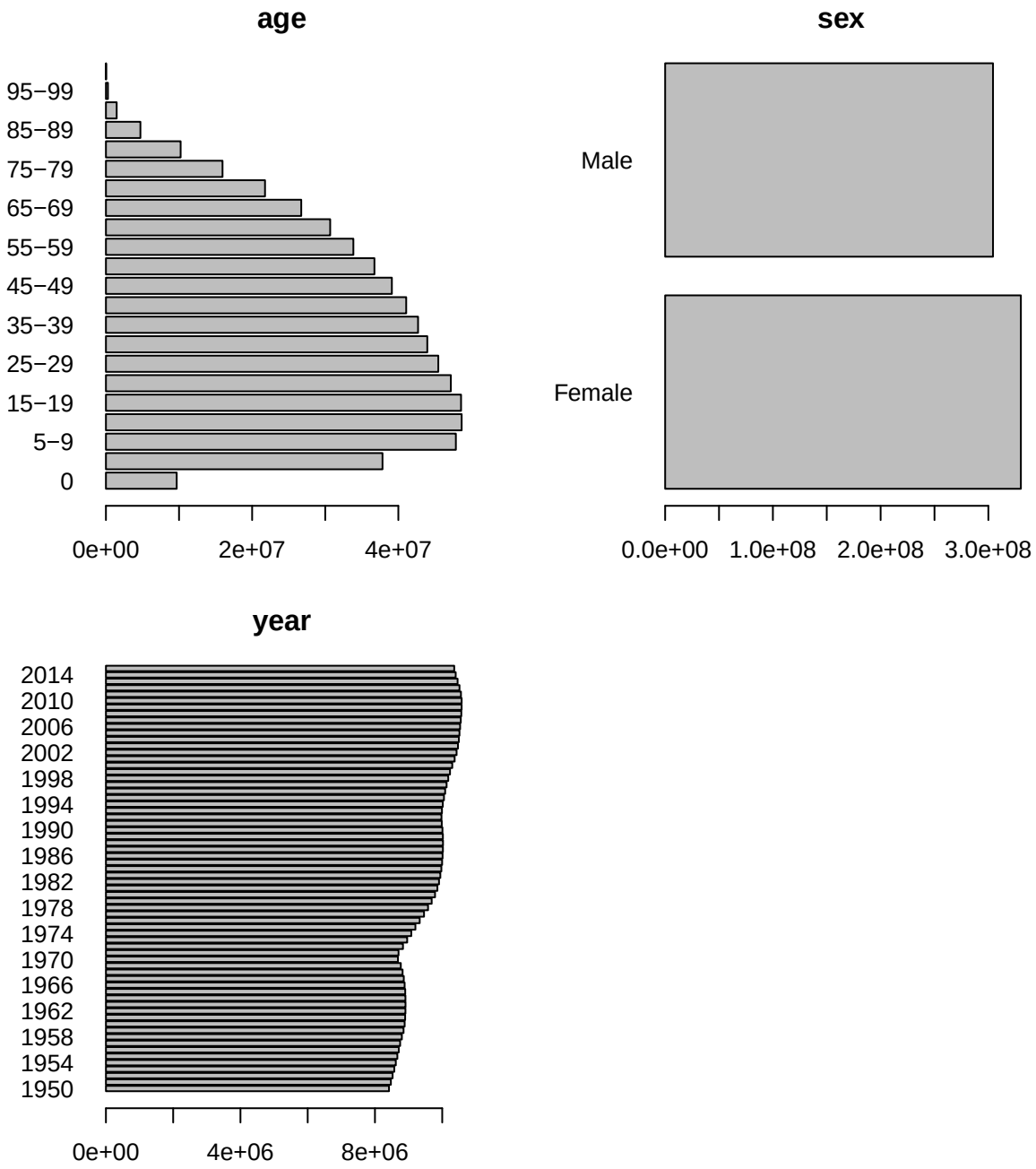



```
exposure <- bdefdata::portugal_exposure %>%
  Counts(dimscales = c(year = "Intervals"))
summary(exposure)
```

```
##
## name:     age       sex    year
## length:   22        2      66
## dimtype:  age       sex    time
## dimscale: Intervals Sexes  Intervals
## first:    0         Female 1950
```

```
## last:     100+      Male   2015
##
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     12.6  80485.7 255696.5 218496.9 334494.5 441351.0
```

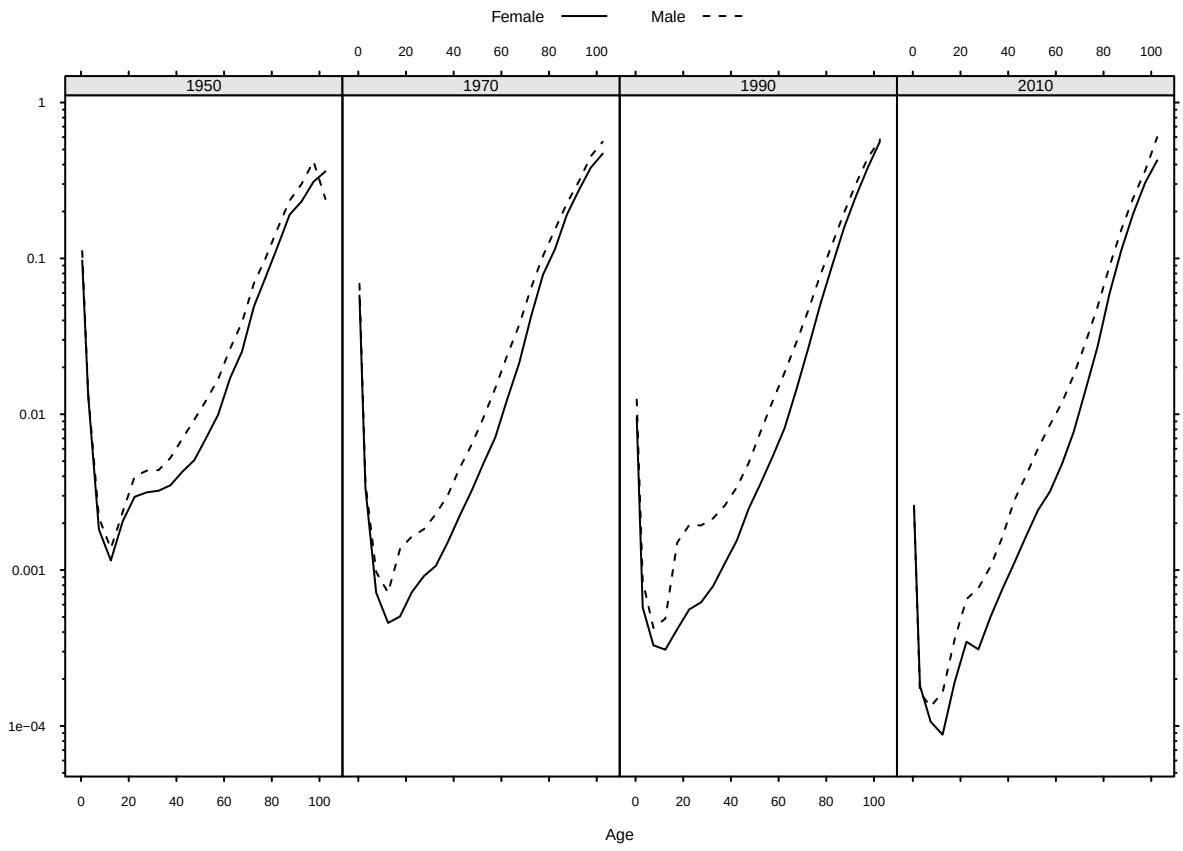
```
plot(exposure)
```



Direct Estimates

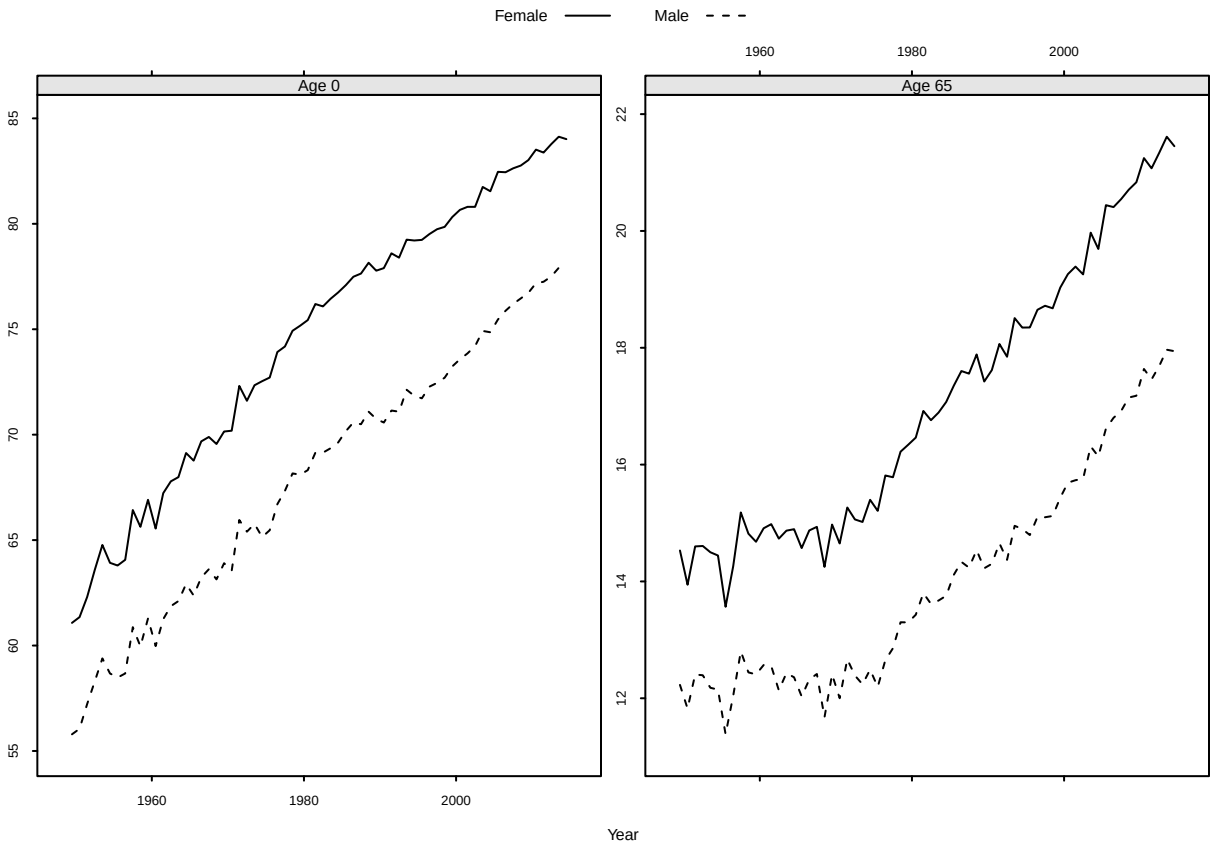
```
rates_direct <- deaths / exposure
```

```
lty <- c("solid", "dashed")
dplot(~ age | year,
      data = rates_direct,
      groups = sex,
      subarray = year %in% c("1950", "1970", "1990", "2010"),
      xlab = "Age",
      ylab = "",
      midpoints = "age",
      col = "black",
      lty = lty,
      layout = c(NA, 1),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90")),
      scales = list(tck = 0.3,
                    y = list(log = TRUE)),
      yscale.components = yscale.components.log10ticks,
      key = list(text = dimnames(rates_direct)["sex"],
                 lines = list(col = "black", lty = lty),
                 columns = 2))
```



```
  life_exp_direct <- rates_direct %>%
    collapseIntervals(dimension = "age", old = c("95-99", "100+"),
                      weights = exposure) %>%
    LifeTable() %>%
    lifeTableFun(fun = "ex")
```

```
life_exp_direct_df <- life_exp_direct %>%
    subarray(age %in% c("0", "65")) %>%
    as.data.frame(direction = "long", midpoints = "year") %>%
    mutate(age = paste("Age", age)) %>%
    mutate(sex = factor(sex, levels = c("Female", "Male")))
lty <- c("solid", "dashed")
xyplot(value ~ year | age,
            data = life_exp_direct_df,
            groups = sex,
            col = "black",
            type = "l",
            lty = lty,
            ylab = "",
            xlab = "Year",
            scales = list(tck = 0.3,
                          y = list(relation = "free")),
            midpoints = "year",
            par.settings = list(fontsize = list(text = 6, points = 4),
                                strip.background = list(col = "grey90")),
            key = list(text = list(levels(life_exp_direct_df$sex)),
                       lines = list(col = "black", lty = lty),
                       columns = 2))
```



Fit Models for Period to 1995

```
model_baseline <- Model(y ~ Poisson(mean ~ (age + sex + year)^2),
                  age ~ DLM(covariates = Covariates(infant=TRUE),
                            damp = NULL),
                  year ~ DLM(level = NULL,
                             damp = NULL),
                  age:sex ~ DLM(trend = NULL,
                                damp = NULL),
                  age:year ~ DLM(level = NULL),
                  sex:year ~ DLM(level = NULL),
                  jump = 0.02)
if(!file.exists("portugal_baseline_1995.est")) {
  set.seed(0)
  estimateModel(model_baseline,
                y = subarray(deaths, year <= 1995),
                exposure = subarray(exposure, year <= 1995),
                filename = "portugal_baseline_1995.est",
                nBurnin = 150000,
                nSim = 50000,
                nChain = 4,
                nThin = 100)
}
```

```r
options(width = 120)
fetchSummary("portugal_baseline_1995.est")
```

```
## ------------------------------------------------------------------
## model:
## y ~ Poisson(mean ~ (age + sex + year)^2)
## dimensions: age, sex, year
## ------------------------------------------------------------------
## y:
## Object of class "Counts"
## dimensions: age, sex, year
## n cells: 2024,  n missing: 0, integers: TRUE, n zeros: 0, median: 1046
## ------------------------------------------------------------------
## MCMC statistics:
## nBurnin: 150000,  nSim: 50000,  nChain: 4,  nThin: 100,  nCore: 4,  nIteration: 2000
##
## Metropolis-Hastings updates:
##                        jump acceptance autocorr
## model.likelihood.rate 0.02      0.664    0.044
##
## parameters:
##                                  Rhat                Est.                    N
##                                   med  max    n       min    med   max
## model.likelihood.rate               1 1.02   25   0.00023  0.009  0.63   2024
## model.prior.mean                 1.02 1.07   25      -8.4   -4.7 -0.51   2024
## model.prior.sd                      1            1             0.061            1
## model.hyper.age.scaleLevel       1.12            1              0.17            1
## model.hyper.age.scaleTrend       1.04            1              0.24            1
## model.hyper.age.coef             1.17            1               2.1            1
## model.hyper.age.scaleError        1.2            1              0.09            1
## model.hyper.year.scaleTrend      1.07            1            0.0021            1
## model.hyper.year.scaleError         1            1             0.033            1
## model.hyper.age:sex.scaleLevel   1.04            1             0.089            1
## model.hyper.age:sex.scaleError      1            1             0.015            1
## model.hyper.age:year.scaleTrend  1.04            1             0.011            1
## model.hyper.age:year.damp        1.04            1              0.84            1
## model.hyper.age:year.scaleError  1.05            1            0.0014            1
## model.hyper.sex:year.scaleTrend  1.07            1             0.001            1
## model.hyper.sex:year.damp         1.1            1              0.94            1
## model.hyper.sex:year.scaleError  1.01            1             0.002            1
## ------------------------------------------------------------------
```

```r
model_alternative <- Model(y ~ Poisson(mean ~ (age + sex + year)^2 - sex:year),
                     age ~ DLM(covariates = Covariates(infant=TRUE),
                               damp = NULL),
                     year ~ DLM(level = NULL,
                                damp = NULL),
                     age:sex ~ DLM(trend = NULL,
                                   damp = NULL),
                     age:year ~ DLM(level = NULL),
                     jump = 0.02)
if(!file.exists("portugal_alternative_1995.est")) {
  set.seed(0)
```

```
  estimateModel(model_alternative,
                y = subarray(deaths, year <= 1995),
                exposure = subarray(exposure, year <= 1995),
                filename = "portugal_alternative_1995.est",
                nBurnin = 150000,
                nSim = 50000,
                nChain = 4,
                nThin = 100)
}
```

```
options(width = 120)
fetchSummary("portugal_alternative_1995.est")
```

```
## ------------------------------------------------------------------
## model:
## y ~ Poisson(mean ~ (age + sex + year)^2 - sex:year)
## dimensions: age, sex, year
## ------------------------------------------------------------------
## y:
## Object of class "Counts"
## dimensions: age, sex, year
## n cells: 2024,  n missing: 0, integers: TRUE, n zeros: 0, median: 1046
## ------------------------------------------------------------------
## MCMC statistics:
## nBurnin: 150000,  nSim: 50000,  nChain: 4,  nThin: 100,  nCore: 4,  nIteration: 2000
##
## Metropolis-Hastings updates:
##                       jump acceptance autocorr
## model.likelihood.rate 0.02      0.686    0.041
##
## parameters:
##                                 Rhat                Est.                    N
##                                  med  max    n       min    med   max
## model.likelihood.rate              1 1.01   25   0.00024  0.009  0.67   2024
## model.prior.mean                1.02 1.08   25      -8.3   -4.7 -0.57   2024
## model.prior.sd                     1    1            0.086              1
## model.hyper.age.scaleLevel      1.04    1              0.2              1
## model.hyper.age.scaleTrend      1.04    1             0.21              1
## model.hyper.age.coef            1.59    1                2              1
## model.hyper.age.scaleError      1.15    1             0.16              1
## model.hyper.year.scaleTrend     1.13    1            0.003              1
## model.hyper.year.scaleError        1    1            0.031              1
## model.hyper.age:sex.scaleLevel  1.03    1            0.087              1
## model.hyper.age:sex.scaleError     1    1            0.013              1
## model.hyper.age:year.scaleTrend 1.02    1            0.011              1
## model.hyper.age:year.damp       1.01    1             0.85              1
## model.hyper.age:year.scaleError 1.11    1           0.0024              1
## ------------------------------------------------------------------
```

Forecasts from 1995

Forecasts of super-population rates

```
if (!file.exists("portugal_baseline_1995.pred")) {
  set.seed(0)
  predictModel(filenameEst = "portugal_baseline_1995.est",
               filenamePred = "portugal_baseline_1995.pred",
               n = 20)
}
rates_super_1995_baseline <- fetch("portugal_baseline_1995.pred",
                                   where = c("model", "likelihood", "rate"))
if (!file.exists("portugal_alternative_1995.pred")) {
  set.seed(0)
  predictModel(filenameEst = "portugal_alternative_1995.est",
               filenamePred = "portugal_alternative_1995.pred",
               n = 20)
}
rates_super_1995_alternative <- fetch("portugal_alternative_1995.pred",
                                      where = c("model", "likelihood", "rate"))
rates_super_1995 <- dbind(Baseline = rates_super_1995_baseline,
                          Alternative = rates_super_1995_alternative,
                          along = "model")
```

Forecasts of finite-population rates

```
exposure_1995 <- exposure %>%
  subarray(year > 1995) %>%
  dbind(Baseline = .,
        Alternative = .,
    along = "model")
expected_deaths <- rates_super_1995 * exposure_1995
deaths_finite <- rpois(n = length(expected_deaths),
                       lambda = expected_deaths) %>%
  array(dim = dim(expected_deaths),
        dimnames = dimnames(expected_deaths)) %>%
  Counts(dimscales = dimscales(expected_deaths))
rates_finite <- deaths_finite / exposure_1995
```

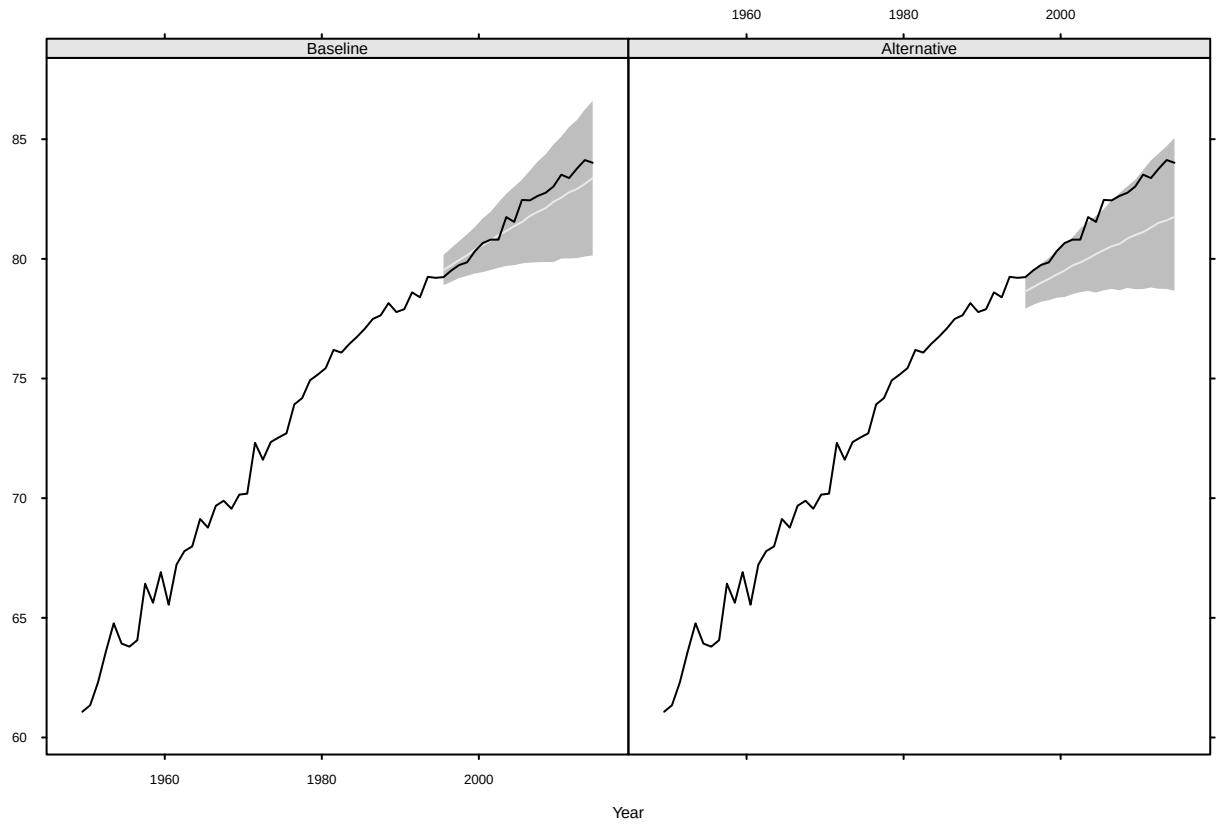
Forecasts of finite-population life expectancy

```
life_exp_forecast_finite <- rates_finite %>%
  collapseIntervals(dimension = "age", old = c("95-99", "100+"),
                    weights = exposure) %>%
  LifeTable() %>%
  lifeTableFun(fun = "ex")
```

```
life_exp_true_finite <- dbind(Baseline = life_exp_direct,
                              Alternative = life_exp_direct,
                              along = "model")
```

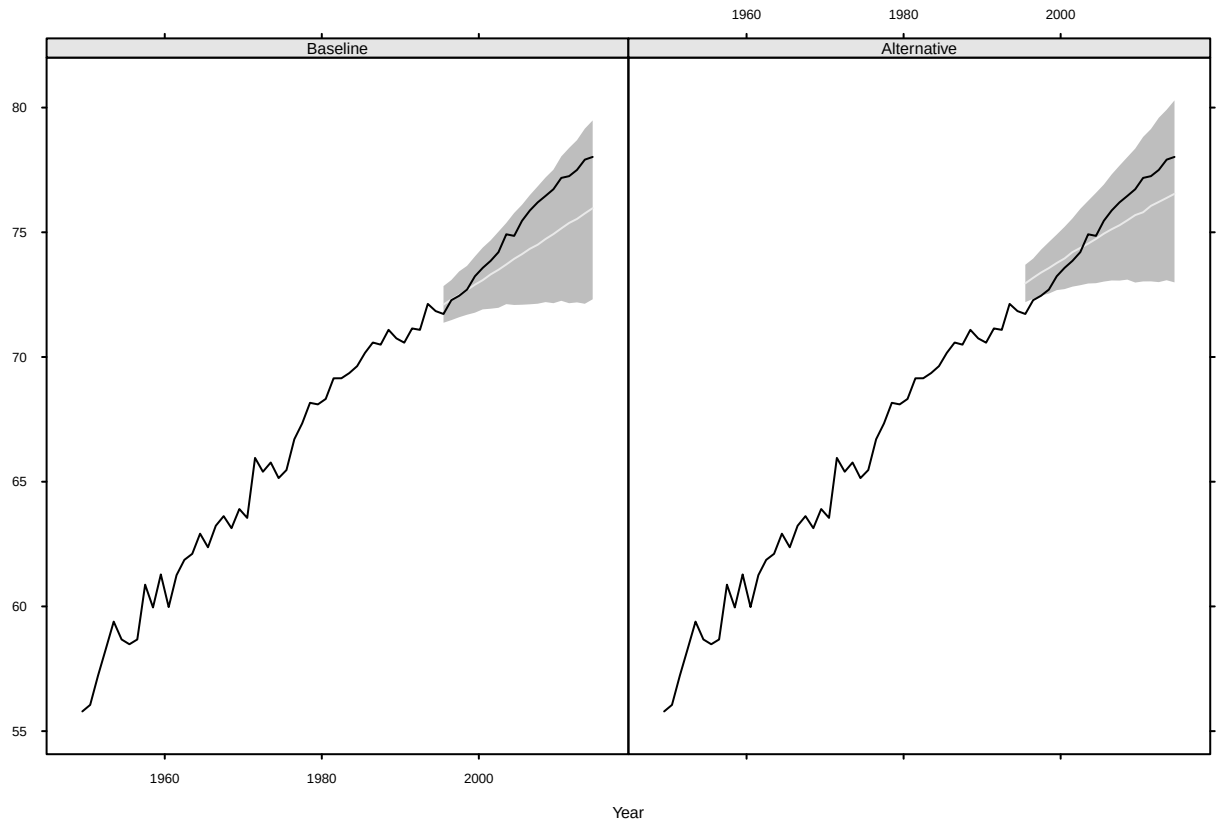
```
life_exp_forecast_age0_female <- life_exp_forecast_finite %>%
  subarray(age == 0 & sex == "Female" ) %>%
  extrapolate(labels = seq(1950, 1995), type = "missing")
life_exp_true_age0_female <- life_exp_true_finite %>%
  subarray(age == 0 & sex == "Female")
```

```
dplot(~ year | model,
      data = life_exp_forecast_age0_female,
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = life_exp_true_age0_female,
                     col = "black"))
```



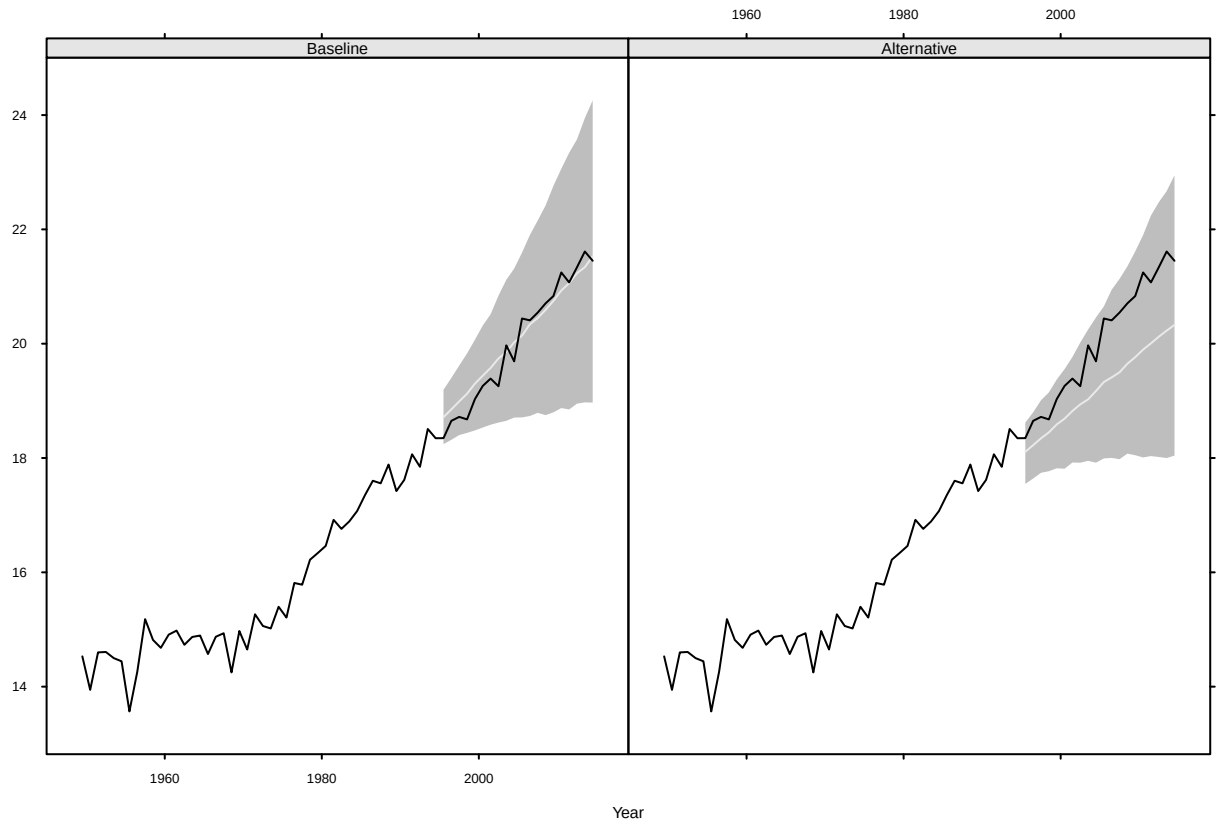

```
life_exp_forecast_age0_male <- life_exp_forecast_finite %>%
  subarray(age == 0 & sex == "Male" ) %>%
  extrapolate(labels = seq(1950, 1995), type = "missing")
life_exp_true_age0_male <- life_exp_true_finite %>%
  subarray(age == 0 & sex == "Male")
```

```
dplot(~ year | model,
      data = life_exp_forecast_age0_male,
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = life_exp_true_age0_male,
                     col = "black"))
```



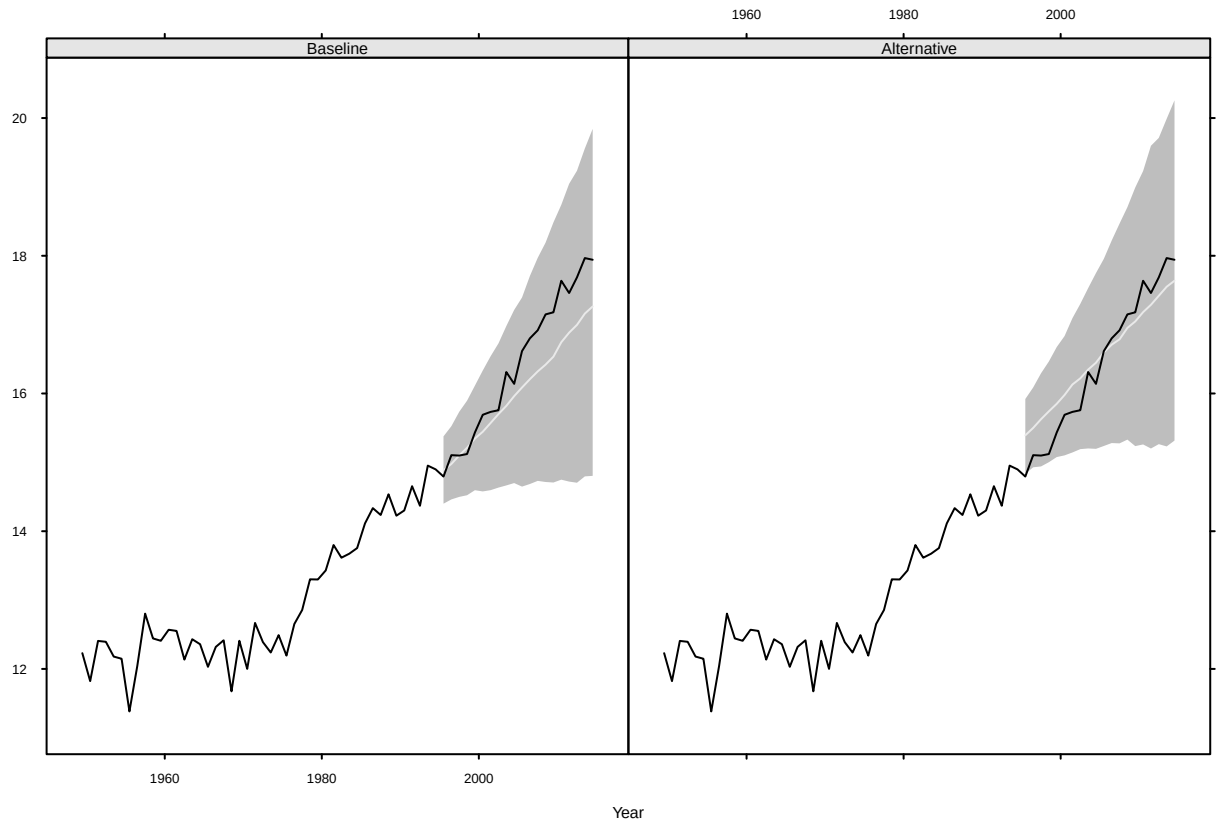

```
life_exp_forecast_age65_female <- life_exp_forecast_finite %>%
  subarray(age == 65 & sex == "Female" ) %>%
  extrapolate(labels = seq(1950, 1995), type = "missing")
life_exp_true_age65_female <- life_exp_true_finite %>%
  subarray(age == 65 & sex == "Female")
```

```r
dplot(~ year | model,
      data = life_exp_forecast_age65_female,
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = life_exp_true_age65_female,
                     col = "black"))
```




```r
life_exp_forecast_age65_male <- life_exp_forecast_finite %>%
  subarray(age == 65 & sex == "Male" ) %>%
  extrapolate(labels = seq(1950, 1995), type = "missing")
life_exp_true_age65_male <- life_exp_true_finite %>%
  subarray(age == 65 & sex == "Male")
```

```
dplot(~ year | model,
      data = life_exp_forecast_age65_male,
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = life_exp_true_age65_male,
                     col = "black"))
```



Performance

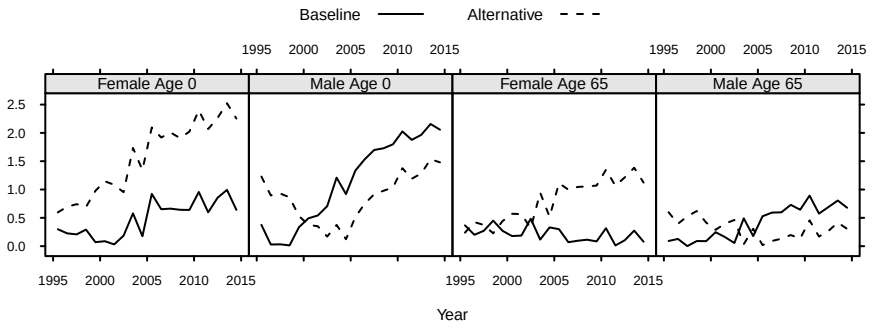
```
truth <- life_exp_direct %>%
  subarray(age %in% c(0, 65)) %>%
```

```
  subarray(year > 1995) %>%
  dbind(Baseline = .,
        Alternative = .,
    along = "model")
truth_df <- as.data.frame(truth,
                          responseName = "truth")
```

```
life_exp_forecast_finite_0_65 <- life_exp_forecast_finite %>%
  subarray(age %in% c(0, 65))

abs_error <- life_exp_forecast_finite_0_65 %>%
  collapseIterations(FUN = median) %>%
  MSE(truth = truth) %>%
  sqrt()

int_score <- life_exp_forecast_finite_0_65 %>%
  collapseIterations(prob = c(0.1, 0.9)) %>%
  intervalScore(truth = truth) %>%
  as("Values")

stepprob <- 0.01
probs <- seq(from = stepprob, by = stepprob, to = 1)
life_exp_forecast_finite_0_65_quantile_df <- life_exp_forecast_finite_0_65 %>%
  collapseIterations(prob = probs) %>%
  as.data.frame(responseName = "forecast") %>%
  mutate(prob = sub("%", "", quantile),
         prob = as.numeric(prob),
         prob = prob / 100)

crps <- inner_join(life_exp_forecast_finite_0_65_quantile_df,
                   truth_df,
                   by = c("age", "sex", "year", "model")) %>%
  group_by(age, sex, year, model) %>%
  mutate(val = 2 * stepprob *
           if_else(forecast <= truth, prob, 1 - prob) *
           abs(forecast - truth)) %>%
  summarise(crps = sum(val)) %>%
  ungroup() %>%
  xtabs(crps ~ age + sex + year + model, data = .) %>%
  Values(dimtypes = dimtypes(abs_error),
         dimscales = dimscales(abs_error))

performance <- dbind("Absolute Error" = abs_error,
                     "Interval Score" = int_score,
                     "CPRS" = crps,
                     along = "measure")

performance_df <- performance %>%
    as.data.frame(direction = "long", midpoints = "year") %>%
    unite(col = sex_age, sex, age, sep = " ") %>%
    mutate(model = factor(model, levels = c("Baseline", "Alternative")),
           sex_age = factor(sex_age,
                            levels = c("Female 0", "Male 0",
```

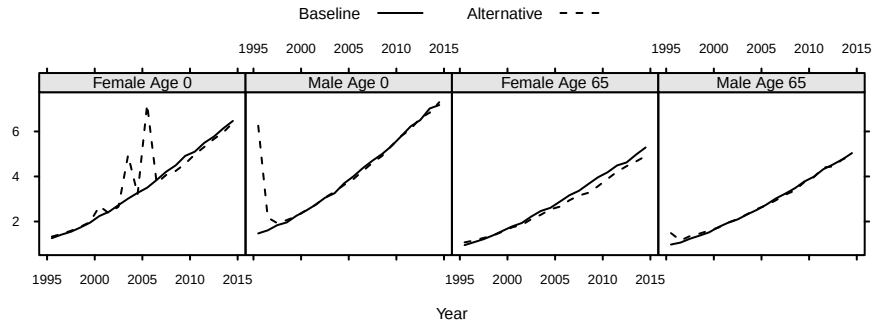
```
                                    "Female 65", "Male 65"),
                          labels = c("Female Age 0", "Male Age 0",
                                     "Female Age 65", "Male Age 65")))
```

```
lty <- c("solid", "dashed")
xyplot(value ~ year | sex_age,
       data = performance_df,
       subset = measure == "Absolute Error",
       groups = model,
       type = "l",
       col = "black",
       lty = lty,
       ylab = "",
       xlab = "Year",
       layout = c(NA, 1),
       scales = list(tck = 0.3),
       par.settings = list(fontsize = list(text = 6),
                           strip.background = list(col = "grey90"),
                           layout.widths = list(ylab.axis.padding = 0,
                                                left.padding = 0,
                                                axis.key.padding = 0,
                                                right.padding = 0)),
       key = list(text = list(levels(performance_df$model)),
                  lines = list(col = "black", lty = lty),
                  columns = 2))
```

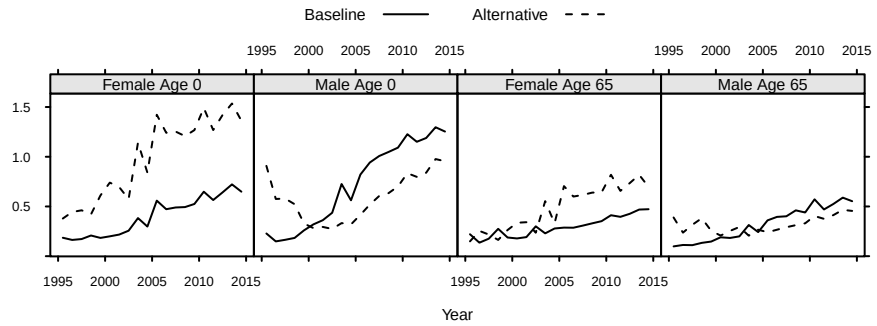



```
lty <- c("solid", "dashed")
xyplot(value ~ year | sex_age,
       data = performance_df,
       subset = measure == "Interval Score",
       groups = model,
       type = "l",
       col = "black",
       lty = lty,
       ylab = "",
       xlab = "Year",
       layout = c(NA, 1),
       scales = list(tck = 0.3),
       par.settings = list(fontsize = list(text = 6),
                           strip.background = list(col = "grey90"),
                           layout.widths = list(ylab.axis.padding = 0,
```

```
                                                left.padding = 0,
                                                axis.key.padding = 0,
                                                right.padding = 0)),
       key = list(text = list(levels(performance_df$model)),
                  lines = list(col = "black", lty = lty),
                  columns = 2))
```




```
lty <- c("solid", "dashed")
xyplot(value ~ year | sex_age,
       data = performance_df,
       groups = model,
       subset = measure == "CPRS",
       type = "l",
       col = "black",
       lty = lty,
       ylab = "",
       xlab = "Year",
       layout = c(NA, 1),
       scales = list(tck = 0.3),
       par.settings = list(fontsize = list(text = 6),
                           strip.background = list(col = "grey90"),
                           layout.widths = list(ylab.axis.padding = 0,
                                                left.padding = 0,
                                                axis.key.padding = 0,
                                                right.padding = 0)),
       key = list(text = list(levels(performance_df$model)),
                  lines = list(col = "black", lty = lty),
                  columns = 2))
```



Fit Models for Period to 2015

```
if(!file.exists("portugal_baseline_2015.est")) {
  set.seed(0)
  estimateModel(model_baseline,
                y = deaths,
                exposure = exposure,
                filename = "portugal_baseline_2015.est",
                nBurnin = 150000,
                nSim = 50000,
                nChain = 4,
                nThin = 100)
}
```

```
fetchSummary("portugal_baseline_2015.est")
```

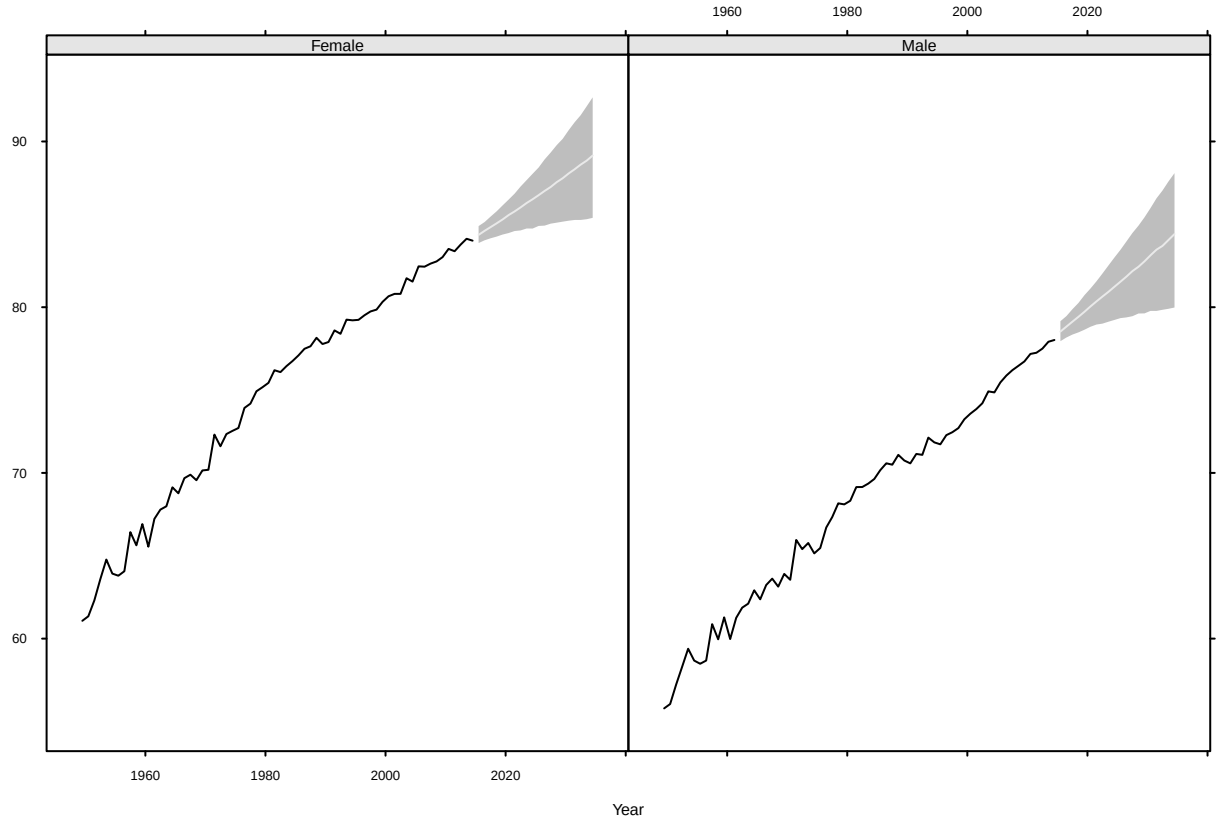
```
## ------------------------------------------------------------------
## model:
## y ~ Poisson(mean ~ (age + sex + year)^2)
## dimensions: age, sex, year
## ------------------------------------------------------------------
## y:
## Object of class "Counts"
## dimensions: age, sex, year
## n cells: 2904,  n missing: 0, integers: TRUE, n zeros: 0, median: 976.5
## ------------------------------------------------------------------
## MCMC statistics:
## nBurnin: 150000,  nSim: 50000,  nChain: 4,  nThin: 100,  nCore: 4,  nIteration: 2000
##
## Metropolis-Hastings updates:
##                        jump acceptance autocorr
## model.likelihood.rate 0.02      0.646    0.054
##
## parameters:
##                                  Rhat                Est.                    N
##                                   med  max    n        min     med   max
## model.likelihood.rate               1 1.02   25   0.000067  0.0068  0.61   2904
## model.prior.mean                 1.02 1.06   25       -9.6      -5 -0.51   2904
## model.prior.sd                      1         1              0.057            1
## model.hyper.age.scaleLevel       1.23         1               0.17            1
## model.hyper.age.scaleTrend       1.09         1               0.24            1
## model.hyper.age.coef             1.19         1                2.1            1
## model.hyper.age.scaleError       1.75         1              0.083            1
## model.hyper.year.scaleTrend      1.09         1             0.0045            1
## model.hyper.year.scaleError         1         1              0.027            1
## model.hyper.age:sex.scaleLevel   1.03         1              0.084            1
## model.hyper.age:sex.scaleError   1.01         1              0.012            1
## model.hyper.age:year.scaleTrend  1.04         1               0.01            1
## model.hyper.age:year.damp        1.05         1               0.87            1
## model.hyper.age:year.scaleError   1.1         1             0.0015            1
## model.hyper.sex:year.scaleTrend  1.25         1            0.00057            1
## model.hyper.sex:year.damp        1.32         1               0.96            1
```

```
## model.hyper.sex:year.scaleError    1         1              0.0017            1
## ------------------------------------------------------------------------
```

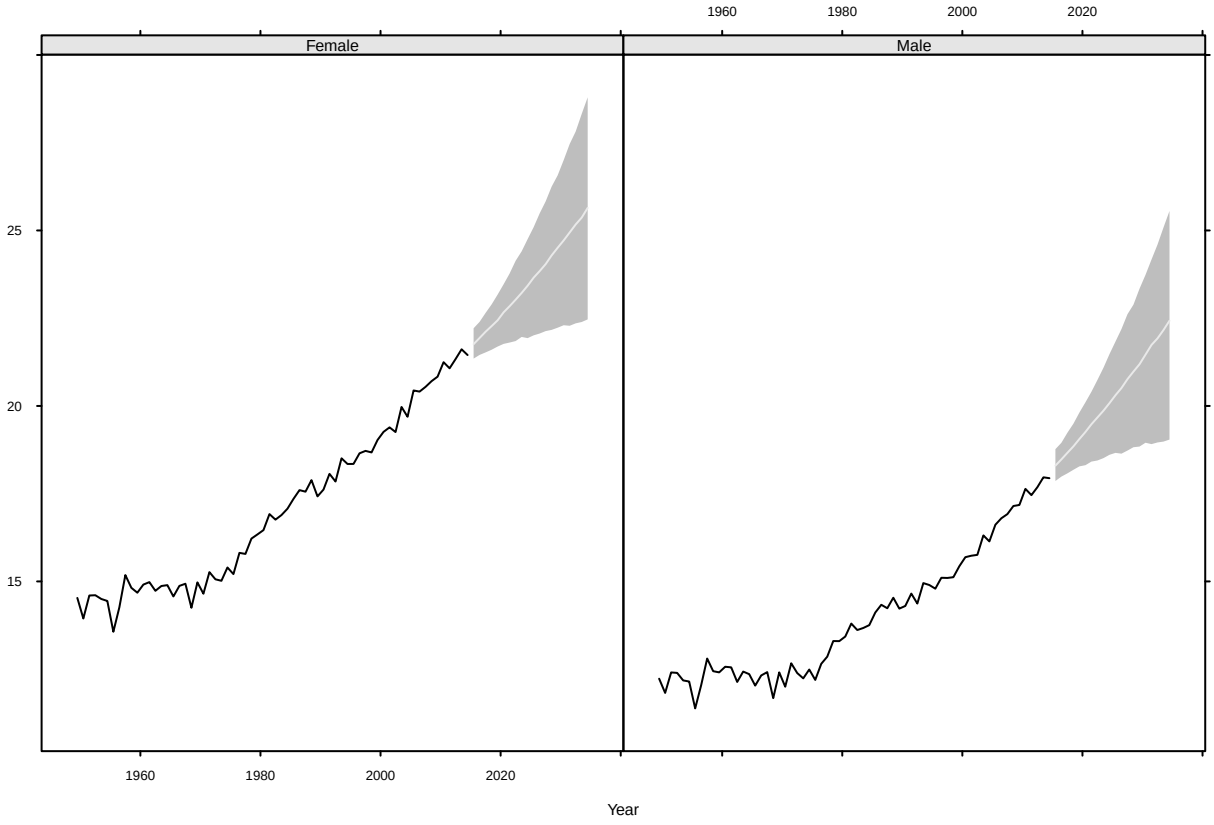
Forecasts from 2015

```
if (!file.exists("portugal_baseline_2015.pred")) {
  set.seed(0)
  predictModel(filenameEst = "portugal_baseline_2015.est",
               filenamePred = "portugal_baseline_2015.pred",
               n = 20)
}
rates_super_2015 <- fetch("portugal_baseline_2015.pred",
                          where = c("model", "likelihood", "rate"))

life_exp_forecast_2015 <- rates_super_2015 %>%
  LifeTable() %>%
  lifeTableFun(fun = "ex") %>%
  extrapolate(labels = dimnames(exposure)$year, type = "missing")
```

```
dplot(~ year | sex,
      data = subarray(life_exp_forecast_2015, age == 0),
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = subarray(life_exp_direct, age == 0),
                     col = "black"))
```




```
dplot(~ year | sex,
      data = subarray(life_exp_forecast_2015, age == 65),
      col = "grey",
      prob = c(0.1, 0.5, 0.9),
      na.rm = TRUE,
      ylab = "",
      xlab = "Year",
      as.table = TRUE,
      midpoints = "year",
      scales = list(tck = 0.3),
      par.settings = list(fontsize = list(text = 6),
                          strip.background = list(col = "grey90"),
                          layout.widths = list(ylab.axis.padding = 0,
                                               left.padding = 0,
                                               axis.key.padding = 0,
                                               right.padding = 0)),
      overlay = list(values = subarray(life_exp_direct, age == 65),
                     col = "black"))
```


Female
Male
25
20
15
1960
1980
2000
2020
Year