Changepoint Approach

Jon Minton

2022-07-22

Aim

The aim of this appendix is to present results for an alternative method of attempting to detect discontinuities in the series of life expectancy changes for different countries. The main approach used was segmented regression using the `segmented` package. The segmented regressions were fit to the data itself y t. This alternative approach instead involves fitting data to the first differences $y(t) - y(t-1)$ t.

The alternative approach identified some similar breakpoints/changepoints to the main, segmented approach. However it differed for many populations in identifying adjacent or near-adjacent changepoints in the data. Further inspection of these pairs of changepoints indicated that the approach developed inadvertently operated as a form of outlier detector algorithm, removing from the series extreme values. Though extreme values are a genuine phenomenon within mortality change data, they were not the primary purpose of the analysis, and so this approach was not used within the main manuscript.

Description of approach

- We first fit a 'null' (no change) model to the data. These are models that represent the assumption that no change occurred in trends in life expectancy over time.
- Then we fit a series on one-break models to the data. These are models that represent the assumption that a single significant change occurred to the series over time. From this series of one-break models, we attempt to find the best one-break model.
- We then fit a series of two-break models to the data. These are models that represent the assumption that were were two significant changes to the series over time. From this series of two-break models, we attempt to find the best two-break models.
- We now have three models to compare: the null model (M_0), the best one-break model (M_1), and the best two-break model (M_2). Although both M_1 can be compared directly[1] with M_0, and M_2 can be compared directly with M_0, M_2 cannot necessarily be compared directly with M_1. Instead, the triplet of models (M_0, M_1 and M_2) are compared indirectly using BIC^[BIC stands for Bayesian Information Criterion, like the similar AIC (An Information Criterion or Akaike's Information Criterion), is a penalised model fit score. By penalised this means that the fit of the model to the data (more specifically its log likelihood) is calculated, then a 'penalty' is applied to this score based on the complexity of the model. BIC and AIC differ only according to how the penalty is applied, with BIC tending to penalise more complex models more severely than AIC. This means BIC will tend to be more conservative in selecting models, providing some protection against overfitting. Both AIC and BIC can be used to compare both nested and non-nested models based on the same dataset, unlike the F-test, the Lagrange Multiplier Test, and so on. Measures of model fit should *not* be used to compare models fit to different datasets.[2], which like AIC provides a penalised model fit score. Lower BIC scores

[1] By 'compared directly', we mean that one model specification can be expressed as a restricted/constrained version of another model, the unrestricted model, with one or more terms in the unrestricted model set to fixed values, usually zero, in the restricted model. Such models can be compared directly using an F-test

[2] For example, it would be wrong to conclude that a model with an R^2 or adjusted R^2 fit to one dataset of 0.80 is 'better' than a model with an R^2 or adjusted R^2 of 0.50 fit to a different dataset, even though R^2 and its variants are often (mis)interpreted in this way

indicate better fit to the data, and so, for each dataset, the model (M_0, M_1 or M_2) with the lowest BIC will be selected.
- For the best of the three models for each dataset, the breakpoint or breakpoints (if any), and the model predictions, will be presented and visualised.

Breaks in the data are estimated by fitting the following model specifications:

- M_0: $\frac{de_x}{dt} \sim \alpha$
- M_1: $\frac{de_x}{dt} \sim \alpha + \beta T$, where

$$T = \begin{cases} 1 & \text{if} \quad t \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

- M_2: $\frac{de_x}{dt} \sim \alpha + \beta_1 T_1 + \beta_2 T_2$, where

$$T_1 = \begin{cases} 1 & \text{if } \tau_1 \leq t < \tau_2 \\ 0 & \text{otherwise} \end{cases}$$

and

$$T_2 = \begin{cases} 1 & \text{if } t \geq \tau_2 \\ 0 & \text{otherwise} \end{cases}$$

i.e. the selection of τ_1 and τ_2 partitions the model into three sections: $\frac{de_x}{dt} \sim \alpha$ where $t < \tau_1$, $\frac{de_x}{dt} \sim \alpha + \beta_1$ where $\tau_1 \leq t < \tau_2$, and $\frac{de_x}{dt} \sim \alpha + \beta_2$ where $t \geq \tau_2$. As $\tau_1 < \tau_2$, only values of τ_2 which are greater than τ_1 are considered. Otherwise all whole integer values of τ are seached through for both M_1 and M_2.

The above analyses are performed for each population group, comprising different combinations of country, sex, and starting age.

Extracting relevant data

```
# load packages

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# load data
hmd_lt <- read_rds("https://github.com/JonMinton/change-in-ex/blob/main/data/lifetables.rds?raw=true")


# Labels for codes
country_code_lookup <-
  tribble(
    ~code, ~country,
```

```
    "DEUTNP", "Germany",
    "DEUTE", "East Germany",
    "DEUTW", "West Germany",
    "ESP", "Spain",
    "FRATNP", "France",
    "ITA", "Italy",
    "GBRTENW", "England & Wales",
    "GBR_SCO", "Scotland",
    "DEUTSYNTH", "Synthetic Germany",
    "NLD", "Netherlands"
  )

countries_of_interest <- c(
  "GBRTENW",
  "GBR_SCO",
  "GBR_UK",
  "FRATNP",
  "ESP",
  "ITA",
  "DEUTNP",
  "DEUTE",
  "DEUTW",
  "NLD"
)

source("https://raw.githubusercontent.com/JonMinton/change-in-ex/main/R/make_synthetic_germany_functions

source("https://raw.githubusercontent.com/JonMinton/change-in-ex/main/R/make_pop_selection.R")


change_in_ex_selected_countries <-
  hmd_ex_selected_countries_with_synth %>%
    group_by(code, x, sex) %>%
    arrange(year) %>%
    mutate(delta_ex = ex - lag(ex)) %>%
    ungroup()
```

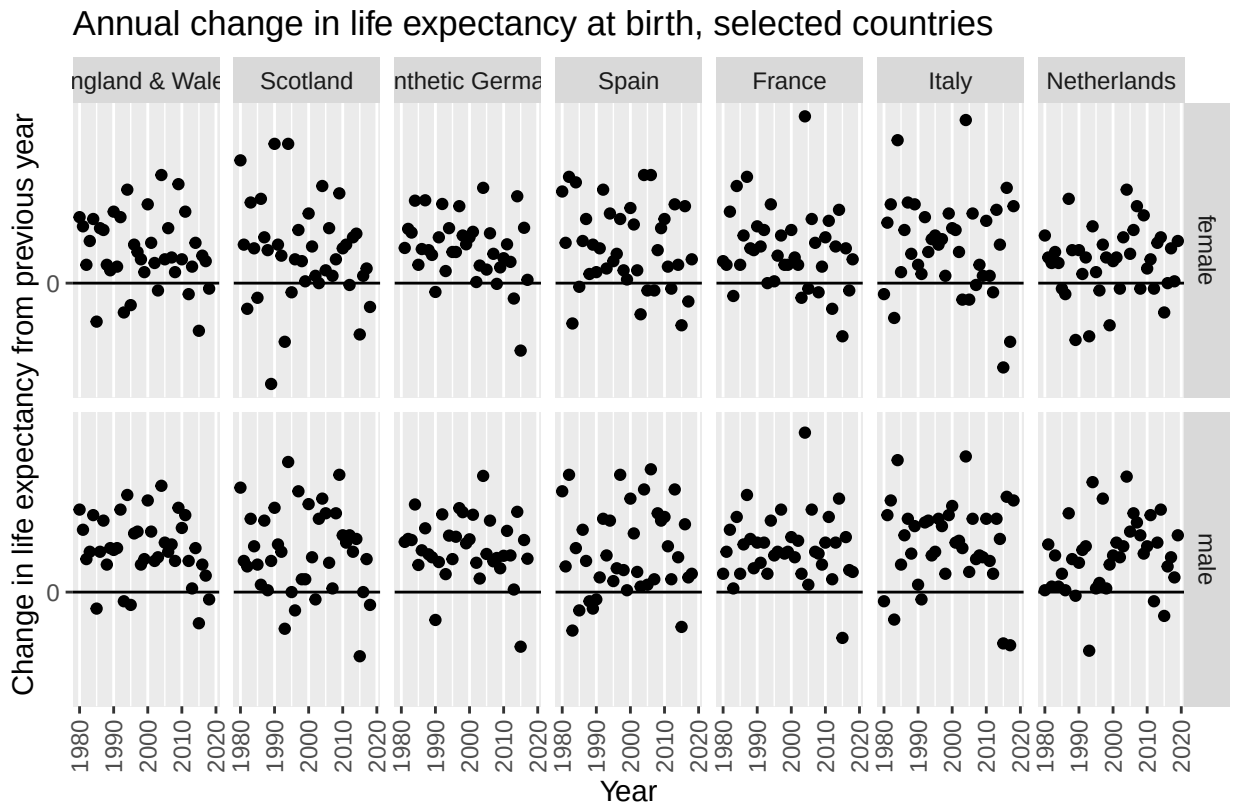
Visualise

```
change_in_ex_selected_countries %>%
  filter(x == 0) %>%
  left_join(country_code_lookup) %>%
  mutate(country = factor(country, levels = c("England & Wales", "Scotland", "Synthetic Germany", "Spai
  filter(!is.na(country)) %>%
  filter(between(year, 1980, 2020)) %>%
  ggplot(aes(x = year, y = delta_ex)) +
  geom_point() +
  facet_grid(sex~country) +
  geom_hline(yintercept = 0) +
  scale_y_continuous(breaks = seq(-30, 50, by = 10)) +
  theme(
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)
  ) +
  labs(
```

```
    x = "Year",
    y = "Change in life expectancy from previous year",
    title = "Annual change in life expectancy at birth, selected countries",
    caption =  "Source: Human Mortality Database. Synthetic Germany based on 20% East/80% West German po
  )
```

```
## Joining, by = "code"
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



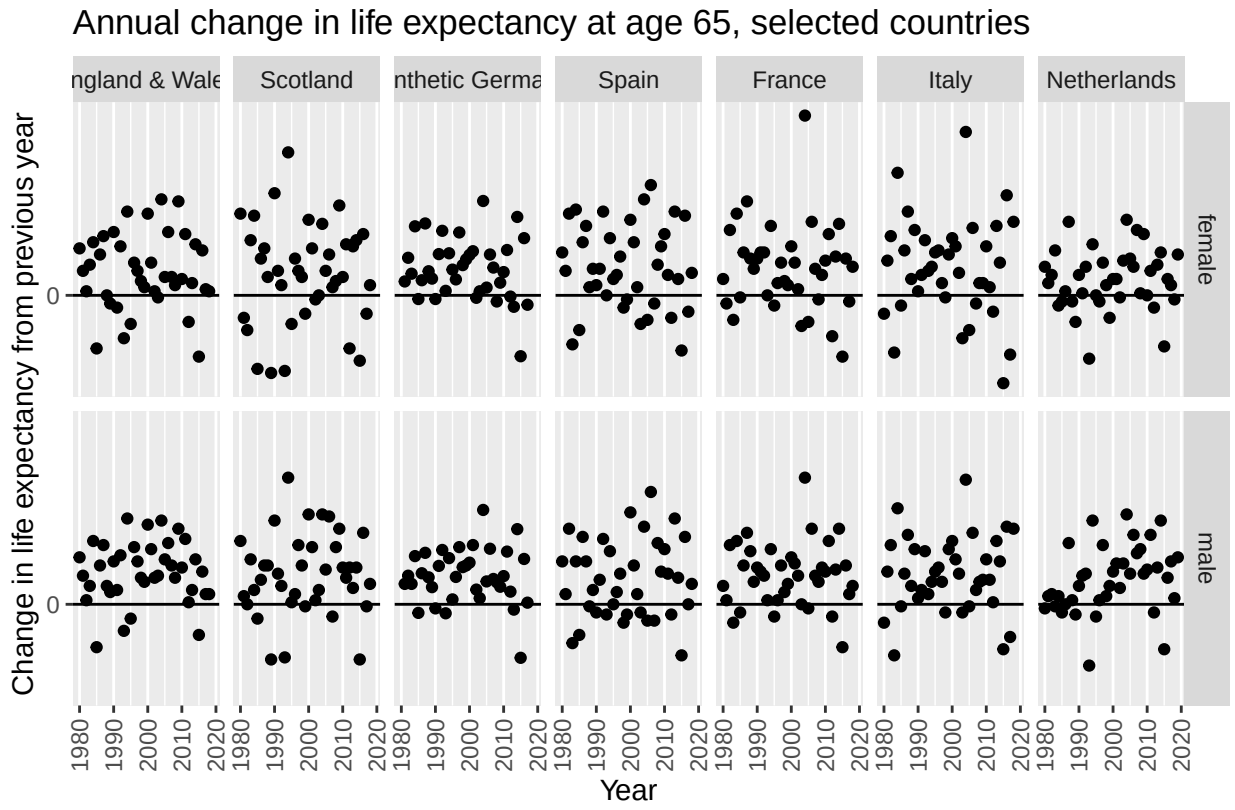
ce: Human Mortality Database. Synthetic Germany based on 20% East/80% West German population weighting

```
change_in_ex_selected_countries %>%
  filter(x == 65) %>%
  left_join(country_code_lookup) %>%
  mutate(country = factor(country, levels = c("England & Wales", "Scotland", "Synthetic Germany", "Spain
  filter(!is.na(country)) %>%
  filter(between(year, 1980, 2020)) %>%
  ggplot(aes(x = year, y = delta_ex)) +
  geom_point() +
  facet_grid(sex~country) +
  geom_hline(yintercept = 0) +
  scale_y_continuous(breaks = seq(-30, 50, by = 10)) +
  theme(
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)
  ) +
  labs(
    x = "Year",
    y = "Change in life expectancy from previous year",
```

```
    title = "Annual change in life expectancy at age 65, selected countries",
    caption =  "Source: Human Mortality Database. Synthetic Germany based on 20% East/80% West German po
  )
```

```
## Joining, by = "code"
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



ce: Human Mortality Database. Synthetic Germany based on 20% East/80% West German population weighting

Functions

The following are functions used to try to identify the best possible breakpoint/changepoint years

```
# Breakpoint functions

run_alt_with_given_tau <- function(tau, df){
  df <-
    df %>%
    filter(!is.na(delta_ex)) %>%
    mutate(
      T_param = ifelse(year < tau, FALSE, TRUE)
    )

  lm(delta_ex ~ T_param, data  = df)
}
```

```r
# This function was not used in the end. The reason is because the fitness landscape is
# difficult to navigate. Instead a gridsearch approach was used

get_best_tau_from_optim <- function(df, buffer = 2, what = c("tau", "optim")){
  # We can put in some data validation checks too

  what <- match.arg(what)

  years <- df$year
  tau_lower <- min(years) + buffer # the lower bound
  tau_upper <- max(years) - buffer # the upper bound
  tau_start <- (tau_upper + tau_lower) / 2 # start just in the middle
  # (This shouldn't matter if the algorithm is robust)

  get_result_from_model <- function(par, df, what = c("AIC", "BIC", "model")){
    # Again, can carry out more validation checks
    what <- match.arg(what)

    tau <- par[["tau"]] # This is how parameters are packed up by optim
    model <-
      df %>%
      arrange(year) %>%
      mutate(delta_ex = ex - lag(ex)) %>%
      filter(!is.na(delta_ex)) %>%
      mutate(T_param = ifelse(year < tau, FALSE, TRUE)) %>%
      lm(delta_ex ~ T_param, data = .)
    if (what == "model") {return(model)      }
    if (what == "BIC")   {return(BIC(model)) }
    if (what == "AIC")   {return(AIC(model)) }

    # NULL should never be returned! If it has something's gone wrong!
    NULL
  }

  optim_obj <-
    optim(
      par = list(tau = tau_start),
      fn = get_result_from_model,
      method = "L-BFGS-B",
      lower = tau_lower,
      upper = tau_upper,
      df = df
    )
  # Need to make the result dependent on the what argument

  if (what == "tau") {return(optim_obj[["par"]])}
  else if (what == "optim"){ return(optim_obj) }

  # Again, the following should never be triggered
  NULL
}
```

```r
# This is the function actually used, for a single changepoint value

get_best_tau_from_gridsearch <- function(df, buffer = 2, what = c("best", "all")){
  # We can put in some data validation checks too

  what <- match.arg(what)

  years <- df$year
  tau_lower <- min(years) + buffer # the lower bound
  tau_upper <- max(years) - buffer # the upper bound
  tau_searchrange <- tau_lower:tau_upper
  get_result_from_model <- function(tau, df, what = c("BIC", "AIC", "model")){ #I've changed the order
    # Again, can carry out more validation checks
    what <- match.arg(what)

    model <-
      df %>%
      arrange(year) %>%
      mutate(delta_ex = ex - lag(ex)) %>%
      filter(!is.na(delta_ex)) %>%
      mutate(T_param = ifelse(year < tau, FALSE, TRUE)) %>%
      lm(delta_ex ~ T_param, data = .)
    if (what == "model") {return(model)      }
    if (what == "BIC")   {return(BIC(model)) }
    if (what == "AIC")   {return(AIC(model)) }

    # NULL should never be returned! If it has something's gone wrong!
    NULL
  }

  search_df <- tibble(
    tau = tau_searchrange
  ) %>%
    mutate(bic = map_dbl(tau, get_result_from_model, df = df))

  if (what == "all") {return(search_df)}

  if (what == "best") {
    out <-
      search_df %>%
      filter(bic == min(bic)) %>%
      select(tau, bic)
    return(out) # I've changed this to compare both tau and bic more easily with 2 cp models
  }
  # Again, the following should never be triggered
  NULL
}

# This is a modification of get_best_tau_from_gridsearch, for two taus
get_best_taus_from_gridsearch <- function(df, buffer = 2, what = c("best", "all")){
  # We can put in some data validation checks too

  what <- match.arg(what)
```

```r
years <- df$year
tau_lower <- min(years) + buffer # the lower bound
tau_upper <- max(years) - buffer # the upper bound

# Now need to list the permutations of taus to consider

tau_grid <- expand_grid(
  tau1 = tau_lower:tau_upper,
  tau2 = tau_lower:tau_upper
) %>%
  filter(tau2 > tau1) # tau2 should be greater than tau1

get_result_from_model <- function(tau1, tau2, df, what = c("BIC", "AIC", "model")){ #I've changed the
  # Again, can carry out more validation checks
  what <- match.arg(what)


  model <-
    df %>%
    arrange(year) %>%
    mutate(delta_ex = ex - lag(ex)) %>%
    filter(!is.na(delta_ex)) %>%
    mutate(T_param = ifelse(
      year < tau1,
      "bp0",
      ifelse(
        year < tau2, "bp1", "bp2"
        )
      )
    ) %>%
    lm(delta_ex ~ T_param, data = .)
  if (what == "model") {return(model)      }
  if (what == "BIC")   {return(BIC(model)) }
  if (what == "AIC")   {return(AIC(model)) }

  # NULL should never be returned! If it has something's gone wrong!
  NULL
}

search_df <- tau_grid %>%
  mutate(bic = map2_dbl(tau1, tau2, get_result_from_model, df = df))

if (what == "all") {return(search_df)}

if (what == "best") {
  out <-
    search_df %>%
    filter(bic == min(bic)) %>%
    select(bic, tau1, tau2)
  return(out)
}
# Again, the following should never be triggered
NULL
```

```
}
```

Running the functions

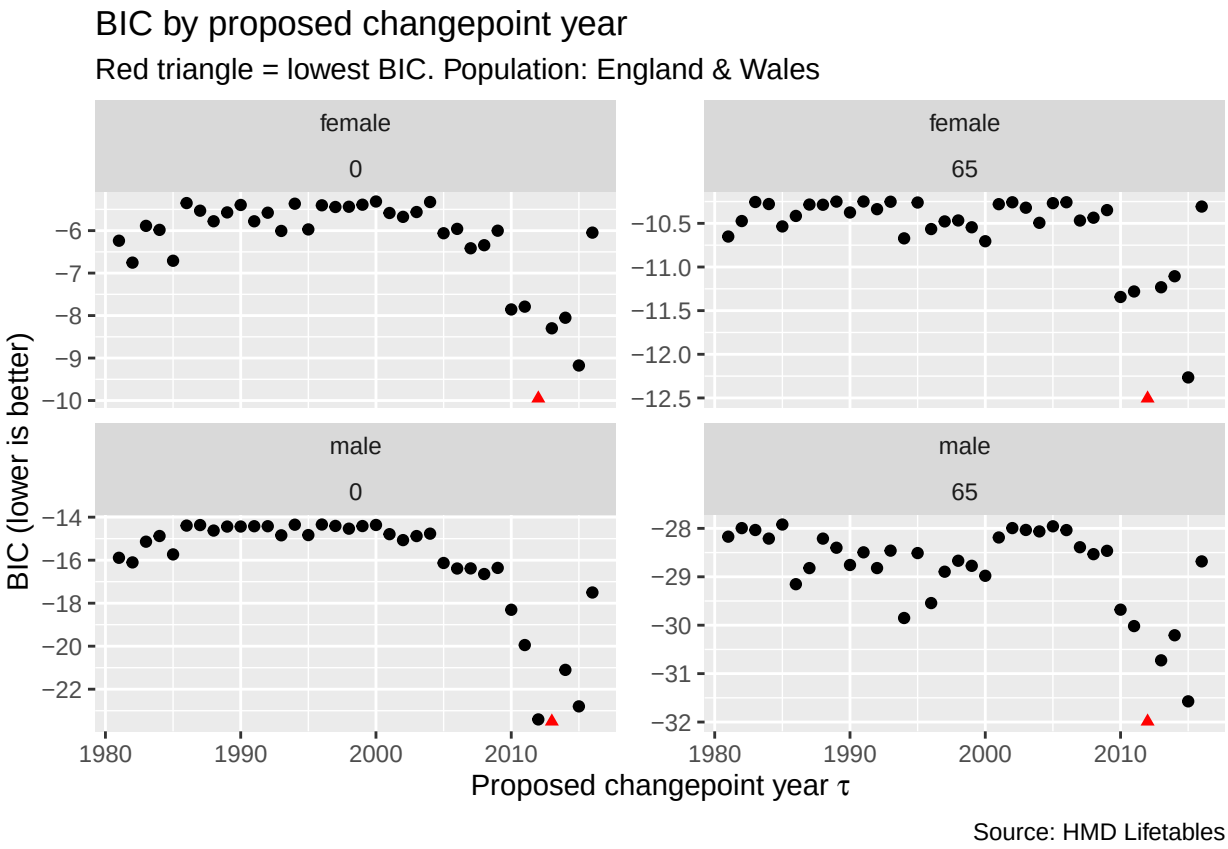
Single changepoint

The following calculates the best single breakpoint model for each population

```
changepoint_breakpoint_models <-
  hmd_ex_selected_countries_with_synth %>%
  filter(code != "DEUTNP") %>%
  filter(year >= 1979) %>%
  group_by(code, x, sex) %>%
  nest() %>%
  mutate(
    mdl_outputs = map(data, get_best_tau_from_gridsearch, what = "all")
  )
```

The relationship between proposed `tau` (i.e. changepoint year) and BIC is shown for England & Wales below

```
changepoint_breakpoint_models %>%
  filter(code == "GBRTENW") %>%
  unnest(mdl_outputs) %>%
  group_by(x, sex) %>%
  mutate(min_bic = bic == min(bic)) %>%
  ungroup() %>%
  ggplot(aes(tau, bic, colour = min_bic, shape = min_bic)) +
  geom_point() +
  facet_wrap(~ sex + x, scales = "free_y") +
  scale_colour_manual(values = c(`TRUE` = 'red', `FALSE` = "black")) +
  scale_shape_manual(values = c(`TRUE` = 'triangle', `FALSE` = 'circle')) +
  labs(
    title = "BIC by proposed changepoint year",
    subtitle = "Red triangle = lowest BIC. Population: England & Wales",
    x = expression(paste("Proposed changepoint year ", tau)),
    y = "BIC (lower is better)",
    caption = "Source: HMD Lifetables"
  ) +
  theme(legend.position = "none")
```



We can see here that 2012 is consistently identified as having the lowest BIC for each of the four subpopulations, except for males where x = 0, where 2013 is identified instead. We can also see that the 'fitness landscape' (i.e. how BIC changes as the proposed changepoing years change) is quite uneven, and so simple numerical optimisation algorithms like Newton-Ralphson are likely to get 'stuck' at local rather than the global optima, and so the results of such algorithms may be quite dependent on starting parameters.

For two changepoints

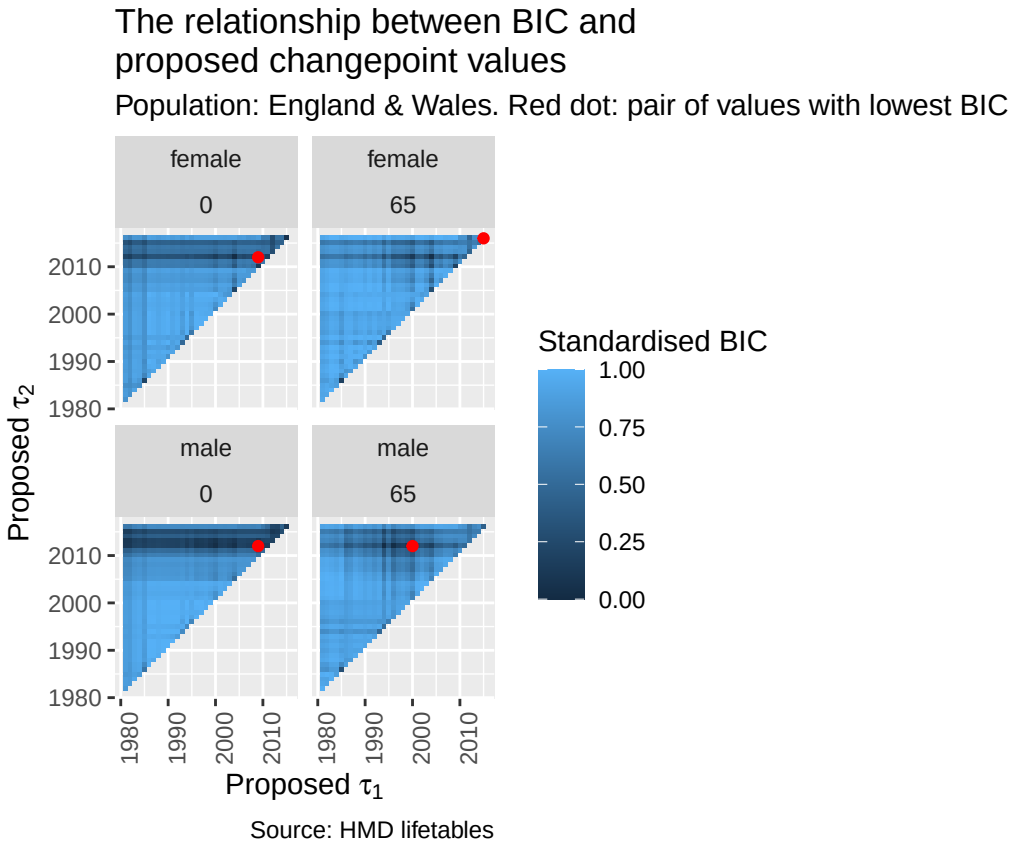
The following code identifies the best two changepoint model for each population

```
changepoint_breakpoints_models <-
  hmd_ex_selected_countries_with_synth %>%
  filter(code != "DEUTNP") %>%
  filter(year >= 1979) %>%
  group_by(code, x, sex) %>%
  nest() %>%
  mutate(
    mdl_outputs = map(data, get_best_taus_from_gridsearch, what = "all")
  )
```

Whereas for a single changepoint the relationship between BIC and proposed τ can be visualised as a scatterplot, for two values τ_1 and τ_2 they can be visualised as a heatmap instead. Our only known constraint is that $\tau_2 > \tau_1$, meaning within the heatmap only a triangle rather than rectangle of candidate values needs to be considered.

The following shows this heatmap for England & Wales. A red dot is added to indicate the pair of candidate values $\{\tau_1, \tau_2\}$ with the lowest BIC. As the range of BIC differs with each dataset, the values are normalised for each subpopulations such that the highest value is 1 and the lowest 0.

```r
changepoint_breakpoints_models %>%
  filter(code == "GBRTENW") %>%
  unnest(mdl_outputs) %>%
  group_by(x, sex) %>%
  mutate(bic_std = (bic - min(bic)) / (max(bic) - min(bic))) %>%
  ggplot(aes(x = tau1, y = tau2, colour = bic_std, fill = bic_std)) +
  geom_raster() +
  facet_wrap(~ sex + x, ncol = 2) +
  scale_color_viridis_c() +
  geom_point(colour= 'red',
             data = changepoint_breakpoints_models %>%
              filter(code == "GBRTENW") %>%
              unnest(mdl_outputs) %>%
              group_by(x, sex) %>%
              mutate(bic_std = (bic - min(bic)) / (max(bic) - min(bic))) %>%
              filter(bic_std == min(bic_std))
  ) +
  coord_fixed() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(
    x = expression(paste("Proposed ", tau[1])),
    y = expression(paste("Proposed ", tau[2])),
    fill = "Standardised BIC",
    title = "The relationship between BIC and\nproposed changepoint values",
    subtitle = "Population: England & Wales. Red dot: pair of values with lowest BIC",
    caption = "Source: HMD lifetables"
  )
```

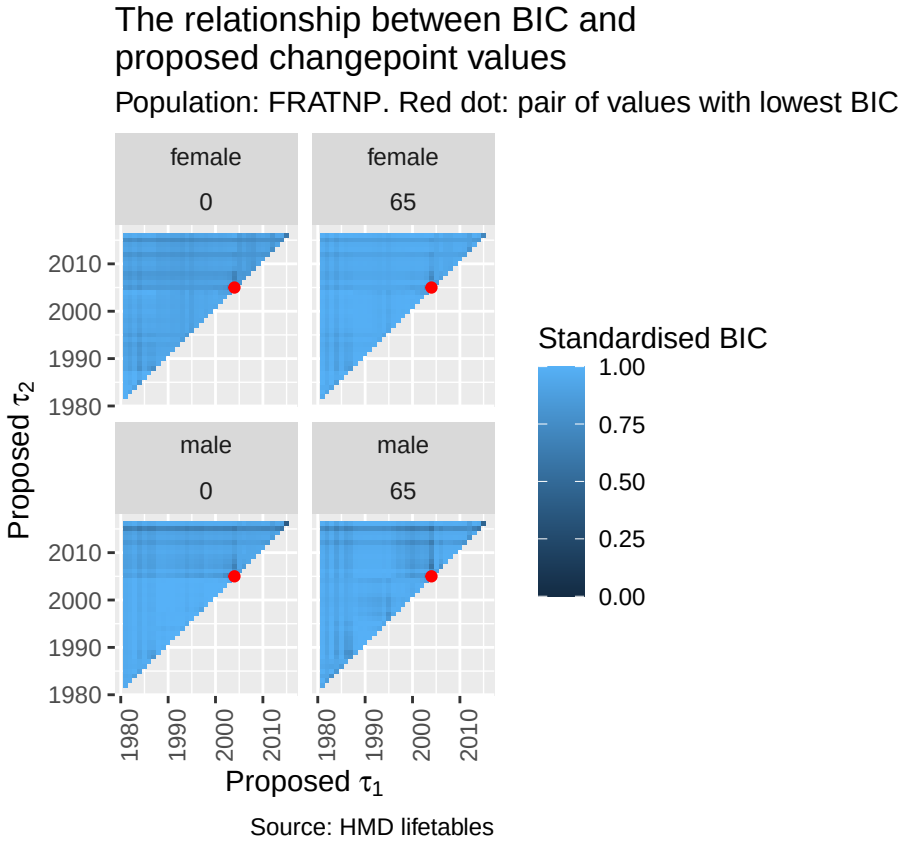


We can see in the above that, for males for x = 65, a $\{\tau_1, \tau_2\}$ pairing of `{2000, 2012}` has been identified. For males and females where x = 0, a near-contiguous pairing of `{2009, 2012}` has been identified. And for females where x = 65 a contigious pairing of `{2015, 2016}` had the lowest BIC.

Similar identification of contiguous and near-contiguous pairings were also identified for many other countries. For example, the following shows this for France.

```
changepoint_breakpoints_models %>%
  filter(code == "FRATNP") %>%
  unnest(mdl_outputs) %>%
  group_by(x, sex) %>%
  mutate(bic_std = (bic - min(bic)) / (max(bic) - min(bic))) %>%
  ggplot(aes(x = tau1, y = tau2, colour = bic_std, fill = bic_std)) +
  geom_raster() +
  facet_wrap(~ sex + x, ncol = 2) +
  scale_color_viridis_c() +
  geom_point(colour= 'red',
             data = changepoint_breakpoints_models %>%
              filter(code == "FRATNP") %>%
              unnest(mdl_outputs) %>%
              group_by(x, sex) %>%
              mutate(bic_std = (bic - min(bic)) / (max(bic) - min(bic))) %>%
              filter(bic_std == min(bic_std))
  ) +
  coord_fixed() +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(
```

```
    x = expression(paste("Proposed ", tau[1])),
    y = expression(paste("Proposed ", tau[2])),
    fill = "Standardised BIC",
    title = "The relationship between BIC and\nproposed changepoint values",
    subtitle = "Population: FRATNP. Red dot: pair of values with lowest BIC",
    caption = "Source: HMD lifetables"
  )
```

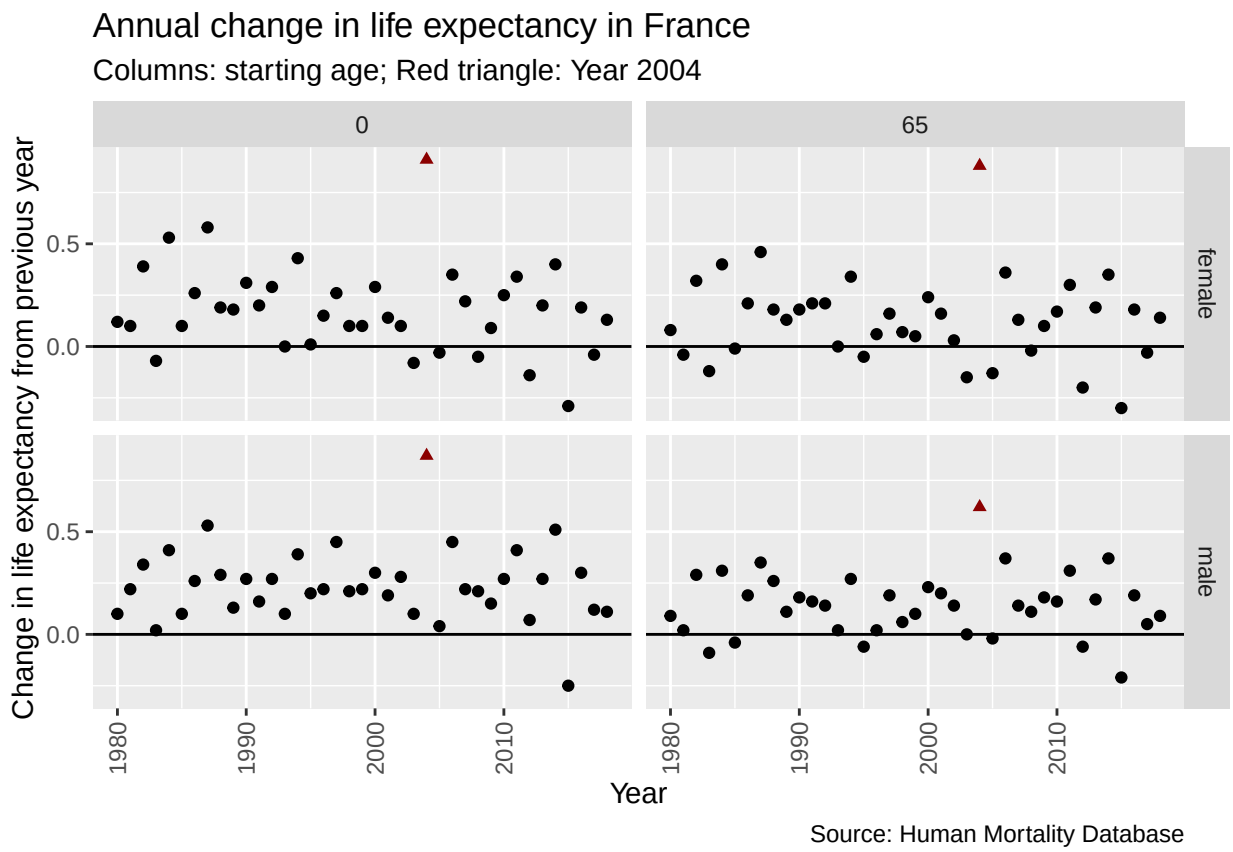


For France, for all four subpopulations, a pairing of `{2004, 2005}` was identified. If we look at the series for France, we can see why this pair of years may have been identified

```
change_in_ex_selected_countries %>%
  filter(code == "FRATNP") %>%
  filter(year >= 1979) %>%
  left_join(country_code_lookup) %>%
  mutate(country = factor(country, levels = c("France"))) %>%
  filter(!is.na(country)) %>%
  filter(between(year, 1980, 2020)) %>%
  mutate(is_2004 = year == 2004) %>%
  ggplot(aes(x = year, y = delta_ex)) +
  geom_point(aes(shape = is_2004, colour = is_2004)) +
  facet_grid(sex ~ x) +
  scale_shape_manual(values =c(`TRUE` = "triangle", `FALSE` = "circle")) +
  scale_colour_manual(values =c(`TRUE` = "darkred", `FALSE` = "black")) +
  geom_hline(yintercept = 0) +
  theme(
```

```
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        legend.position = 'none'
  ) +
  labs(
    x = "Year",
    y = "Change in life expectancy from previous year",
    title = "Annual change in life expectancy in France",
    subtitle = "Columns: starting age; Red triangle: Year 2004",
    caption =  "Source: Human Mortality Database"
  )
```

```
## Joining, by = "code"
```



We can see that for each of this subpopulation, year = 2004 (i.e. the change in ex from 2003 to 2004) was much higher than all other years, and so these points are outliers. So, it appears the changepoint approach has inadertently 'detected' this outlier.

Best model for each population

The following code shows the best performing model, from the best zero, one and two changepoint models, for each of the populations.

```
estimate_null <- function(df){
  df %>%
    arrange(year) %>%
    mutate(delta_ex = ex - lag(ex)) %>%
```

```
    filter(!is.na(delta_ex)) %>%
    lm(delta_ex ~ 1, data = .)
}

  hmd_ex_selected_countries_with_synth %>%
    filter(code != "DEUTNP") %>%
    filter(year >= 1979) %>%
    group_by(code, x, sex) %>%
    nest() %>%
    mutate(
      mdl_0 = map(data, estimate_null),
      bic_0    = map_dbl(mdl_0, BIC)
    ) %>%
    select(code, x, sex, mdl_0, bic_0) %>%
    left_join(
      changepoint_breakpoint_models %>%
        unnest(mdl_outputs) %>%
        group_by(code, x, sex) %>%
        filter(bic == min(bic)) %>%
        select(code, x, sex, tau1 = tau, bic_1 = bic)
    ) %>%
    left_join(
      changepoint_breakpoints_models %>%
        unnest(mdl_outputs) %>%
        group_by(code, x, sex) %>%
        filter(bic == min(bic)) %>%
        select(code, x, sex, tau1_2 = tau1, tau2_2 = tau2, bic_2 = bic)
    ) %>%
  select(-mdl_0) %>%
  ungroup() # %>%
```

```
## Joining, by = c("code", "x", "sex")
## Joining, by = c("code", "x", "sex")
```

```
## # A tibble: 36 x 9
##    code       x sex     bic_0  tau1  bic_1 tau1_2 tau2_2  bic_2
##    <chr>  <dbl> <chr>   <dbl> <int>  <dbl>  <int>  <int>  <dbl>
##  1 DEUTE      0 female   4.19  2005   3.67   1991   2002  -3.14
##  2 DEUTE     65 female -20.6   1987 -22.8    1987   2002 -26.9
##  3 DEUTW      0 female -16.0   1988 -17.1    1988   2015 -16.8
##  4 DEUTW     65 female -27.4   2015 -26.8    2004   2005 -27.5
##  5 ESP        0 female   1.99  1983   1.99   1983   1984   1.95
##  6 ESP       65 female  -5.81  2014  -3.32   1983   1984  -2.94
##  7 FRATNP     0 female  -2.08  2015  -1.97   2004   2005 -10.5
##  8 FRATNP    65 female  -4.73  2015  -3.05   2004   2005 -14.2
##  9 ITA        0 female  13.4   2005  14.3    2004   2005  10.7
## 10 ITA       65 female   6.71  2005   9.02   2004   2005   4.41
## # ... with 26 more rows
```

```
run_tau_1 <- function(tau, df){
  df %>%
    arrange(year) %>%
    mutate(delta_ex = ex - lag(ex)) %>%
    filter(!is.na(delta_ex)) %>%
    mutate(T_param = ifelse(year < tau, FALSE, TRUE)) %>%
```

```
    lm(delta_ex ~ T_param, data = .)
}

run_tau_2 <- function(tau1, tau2, df){
  df %>%
    arrange(year) %>%
    mutate(delta_ex = ex - lag(ex)) %>%
    filter(!is.na(delta_ex)) %>%
    mutate(T_param = ifelse(
      year < tau1,
      "bp0",
      ifelse(
        year < tau2, "bp1", "bp2"
        )
      )
    ) %>%
    lm(delta_ex ~ T_param, data = .)
}

make_path_coords <- function(df, taus = NULL, mdl){
  stopifnot(
    "wrong number of coefficients for the model type" =
    length(coefficients(mdl)) == length(taus) + 1
  )

  min_x = min(df$year)
  max_x = max(df$year)

  coeffs <- coefficients(mdl)

  y0 <- coeffs[1]

  out <- tribble(
    ~x, ~y,
    min_x, y0
  )

  if(length(coeffs) == 1){
    out <- out %>%
      bind_rows(
        tribble(
          ~x, ~y,
          max_x, y0
        )
      )
    return(out)

  } else if (length(coeffs) == 2){
    out <- out %>%
      bind_rows(
        tribble(
          ~x, ~y,
          taus[1], y0,
```

```
          taus[1], y0 + coeffs[2],
          max_x, y0 + coeffs[2]
        )
      )
    return(out)

  } else if (length(coeffs) == 3){
    out <- out %>%
      bind_rows(
        tribble(
          ~x, ~y,
          taus[1], y0,
          taus[1], y0 + coeffs[2],
          taus[2], y0 + coeffs[2],
          taus[2], y0 + coeffs[3],
          max_x, y0 + coeffs[3]

        )
      )

    return(out)
  }
  return(NULL) # should not be triggered
}

# Let's try this with 0cp

paths_0cp <-
  hmd_ex_selected_countries_with_synth %>%
    filter(code != "DEUTNP") %>%
    filter(year >= 1979) %>%
    group_by(code, x, sex) %>%
    nest() %>%
    mutate(
      mdl_0 = map(data, estimate_null)
    ) %>%
    mutate(bic = map_dbl(mdl_0, BIC)) %>%
    mutate(
      tau_paths = pmap(.l = list(df = data, mdl = mdl_0), make_path_coords)
    ) %>%
  select(code, x, sex, bic_0cp = bic, paths_0cp = tau_paths)

# Now 1cp
paths_1cp <-
  changepoint_breakpoint_models %>%
    unnest(mdl_outputs) %>%
    group_by(code, x, sex) %>%
    filter(bic == min(bic)) %>%
    ungroup() %>%
    mutate(
      mdl_with_best_1cp = map2(tau, data, run_tau_1)
    ) %>%
    mutate(
```

```
      tau_paths = pmap(
        .l = list(
          df = data,
          mdl = mdl_with_best_1cp,
          taus = c(tau)
        ),
        make_path_coords
      )
    ) %>%
    select(
      code, x, sex, bic_1cp = bic, paths_1cp = tau_paths
    )

# and now for 2cp
paths_2cp <-
  changepoint_breakpoints_models %>%
    unnest(mdl_outputs) %>%
    group_by(code, x, sex) %>%
    filter(bic == min(bic)) %>%
    ungroup() %>%
    mutate(mdl_with_best_2cp = pmap(.l = list(tau1=tau1, tau2=tau2, df=data), run_tau_2)) %>%
    mutate(taus = map2(tau1, tau2, c)) %>%
    mutate(
      tau_paths = pmap(
        .l = list(
          df = data,
          mdl = mdl_with_best_2cp,
          taus = taus
        ),
        make_path_coords
      )
    ) %>%
    select(code, x, sex, bic_2cp = bic, paths_2cp = tau_paths)

best_cp_paths <-
  reduce(list(paths_0cp, paths_1cp, paths_2cp), left_join) %>%
    rename(start_age = x) %>%
    pivot_longer(cols = starts_with(c("path", "bic")), names_to = c(".value", "path_type"), names_sep =
    group_by(code, start_age, sex) %>%
    mutate(
      bic_rank = as.character(rank(bic))
    ) %>%
    ungroup() %>%
    unnest(paths) %>%
    left_join(country_code_lookup) %>%
  mutate(country = factor(country, levels = c("England & Wales", "Scotland", "Synthetic Germany", "Spai
  filter(!is.na(country)) %>%
  group_by(country, sex, start_age) %>%
  filter(bic_rank == 1)
```

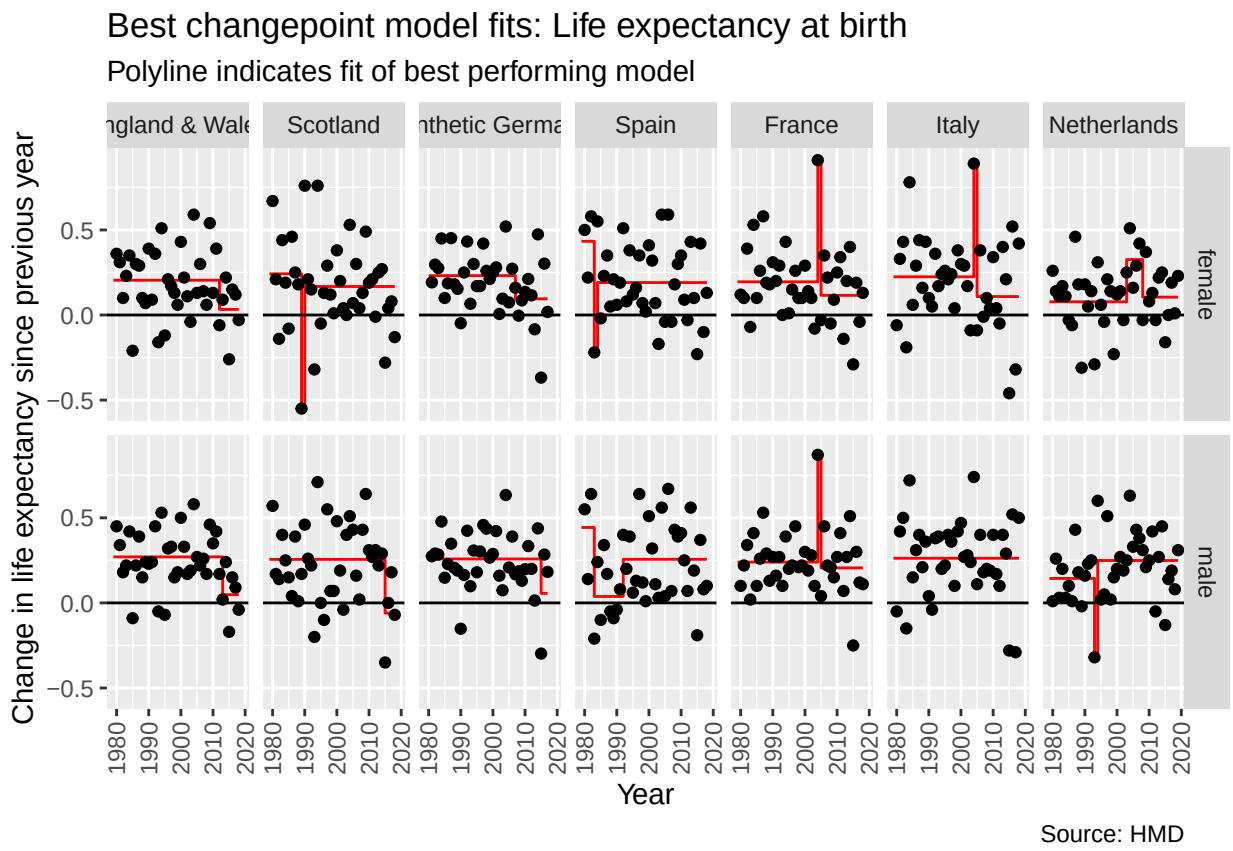
```
## Joining, by = c("code", "x", "sex")
## Joining, by = c("code", "x", "sex")
## Joining, by = "code"
```

The best changepoint model (if any), along with the points, are shown for life expectancy at birth below

```
best_cp_paths %>%
  filter(start_age == 0) %>%
  ggplot(aes(x, y)) +
  geom_path(colour = "red") +
  facet_grid(sex ~ country ) +
  expand_limits(y  = 0) +
  geom_hline(yintercept = 0) +
  theme(
      axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
      legend.position = 'none'
  ) +
  geom_point(
    aes(year, delta_ex),
    inherit.aes = FALSE,
    data = change_in_ex_selected_countries %>%
      filter(x == 0) %>%
      filter(year >= 1979) %>%
      left_join(country_code_lookup) %>%
      mutate(
        country = factor(
          country,
          levels = c("England & Wales", "Scotland", "Synthetic Germany", "Spain", "France", "Italy", "N
          )
        ) %>%
        filter(!is.na(country))   %>%
        filter(between(year, 1980, 2020))
  ) +
  labs(
    x = "Year",
    y = "Change in life expectancy since previous year",
    title = "Best changepoint model fits: Life expectancy at birth",
    subtitle = "Polyline indicates fit of best performing model",
    caption = "Source: HMD"
  )
```

```
## Joining, by = "code"
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



We can see that the best-performing model is a two changepoing model with contiguous years for Scottish females, Spanish females, French females, Italian females, French males and Dutch males. For each of these models it is clear that the model in effect 'detected' an outlier of either exceptionally high annual change (France and Italy) or exceptionally low or negative annual change (Scottish males, Spanish females, Dutch males).

For England & Wales (both males and females) and for Scottish females, a single changepoint model was selected, with lower rates of annual life expectancy gain after around 2011 than before. For Scottish males this fell to negative values (declining rather than stalling mortality). A somewhat similar pattern is observed for Germany, but this appears to be less consistent in term and less severe than for England & Wales.

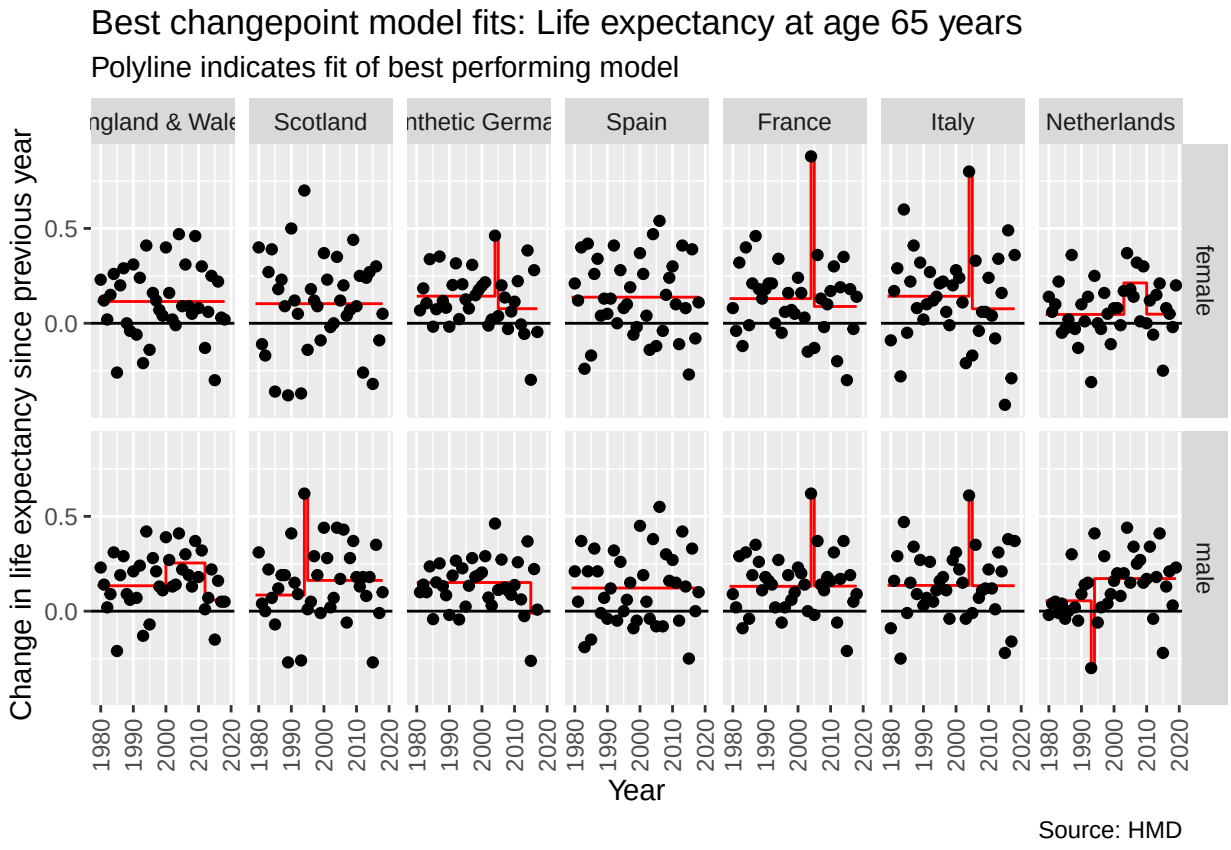
The equivalent graph for life expectancy from age 65 is shown below:

```
best_cp_paths %>%
  filter(start_age == 65) %>%
  ggplot(aes(x, y)) +
  geom_path(colour = "red") +
  facet_grid(sex ~ country ) +
  expand_limits(y  = 0) +
  geom_hline(yintercept = 0) +
  theme(
      axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
      legend.position = 'none'
  ) +
  geom_point(
    aes(year, delta_ex),
    inherit.aes = FALSE,
    data = change_in_ex_selected_countries %>%
```

```
      filter(x == 65) %>%
      filter(year >= 1979) %>%
      left_join(country_code_lookup) %>%
      mutate(
        country = factor(
          country,
          levels = c("England & Wales", "Scotland", "Synthetic Germany", "Spain", "France", "Italy", "N
          )
        ) %>%
        filter(!is.na(country))   %>%
        filter(between(year, 1980, 2020))
  ) +
  labs(
    x = "Year",
    y = "Change in life expectancy since previous year",
    title = "Best changepoint model fits: Life expectancy at age 65 years",
    subtitle = "Polyline indicates fit of best performing model",
    caption = "Source: HMD"
  )
```

```
## Joining, by = "code"
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



For changes in conditional life expectancy from age 65 two changepoint models with contiguous years were identified for females in Germany, France and Italy; and for males in Scotland, France, Italy and the Netherlands.

Summary and conclusion

This appendix has show the implementation and results of applying an alternative method to identifying distinct changes in trends in life expectancy for the selected countries. In many cases, a two changepoint model was identified. However, often these two changepoint models involved contiguous years, and appear to in effect be operating as outlier detection algorithms, picking out exceptional years from the series.

Although the fact these individual years are outliers is important to note, as it illustrates how varied life expectancy change series can be when single years are used, these single years cannot, by definition, define or constitute a change in trend, which was the purpose of this exercise. It is for this reason that the segmented approach was used instead of this approach in the main paper.

The changepoint approach could be adapted to be less 'sensitive' to outlier years by narrowing the search space for τ_2 conditional on τ_1 so as to be two or more years later. (e.g. if τ_1 were 1990 then candidate values for τ_2 would start from 1992 instead of 1991.) However, the choice of how much to narrow down the search space in this way is somewhat arbitrary, so was not adapted further for this paper.