Barcelona Summer School of Demography

Module 2. Demography with R

3. Standardization and decomposition

Standardization and decomposition

Tim Riffe
tim.riffe@gmail.com

9 July 2025

Contents

1 Summary

Today we'll look at ways to make data more comparable (standardization) and ways to explain differences between summary measures (decomposition). As per the previous days, this material was originally prepared by the one-and-only Marie-Pier Bergeron-Boucher, credit is due to her for organizing the logic and rigor of this lesson. My contributions have been light edits to the text, occasional insertions where I though they would help, doing a full overhaul of the code to a tidy approach, and in considering extra decomposition methods.

2 Data

We will compare mortality in Taiwan and Japan. I downloaded their mortality rates from the HMD using the *demography* package (Hyndman et al. 2017) and tidified them using the approach from yesterday. I save you having to replicate that code and have posted the data as a `csv` on the github site. You can read it directly into `R`, below.

We *might* use the `DemoDecomp` package today if there's time and someone wants a demonstration of generalized decomposition. Just in case, feel free to install this, though it isn't strictly required for the prepared lesson.

```
install.packages("DemoDecomp")
```

I sometimes make updates to it without pushing to the main `R` repositories, so you could also get a more up-to-date version of the package here, if so inclined

```
install.packages("remotes")
library(remotes)
install_github("timriffe/DemoDecomp")
```

Get the data and load our beloved packages:

```r
library(tidyverse)
library(DemoDecomp)
# will copy this link into the google doc too
DAT <- read_csv("https://github.com/timriffe/BSSD2025Module2/raw/master/data/JPNTWN.csv")
```

3 Standardization

Standardization is a commonly-used technique when comparing rates or probabilities for groups with differences in composition. Standardization is used to avoid the confounding effect of the population structure by simply equalizing structure for all groups.

3.1 Problems with crude measures

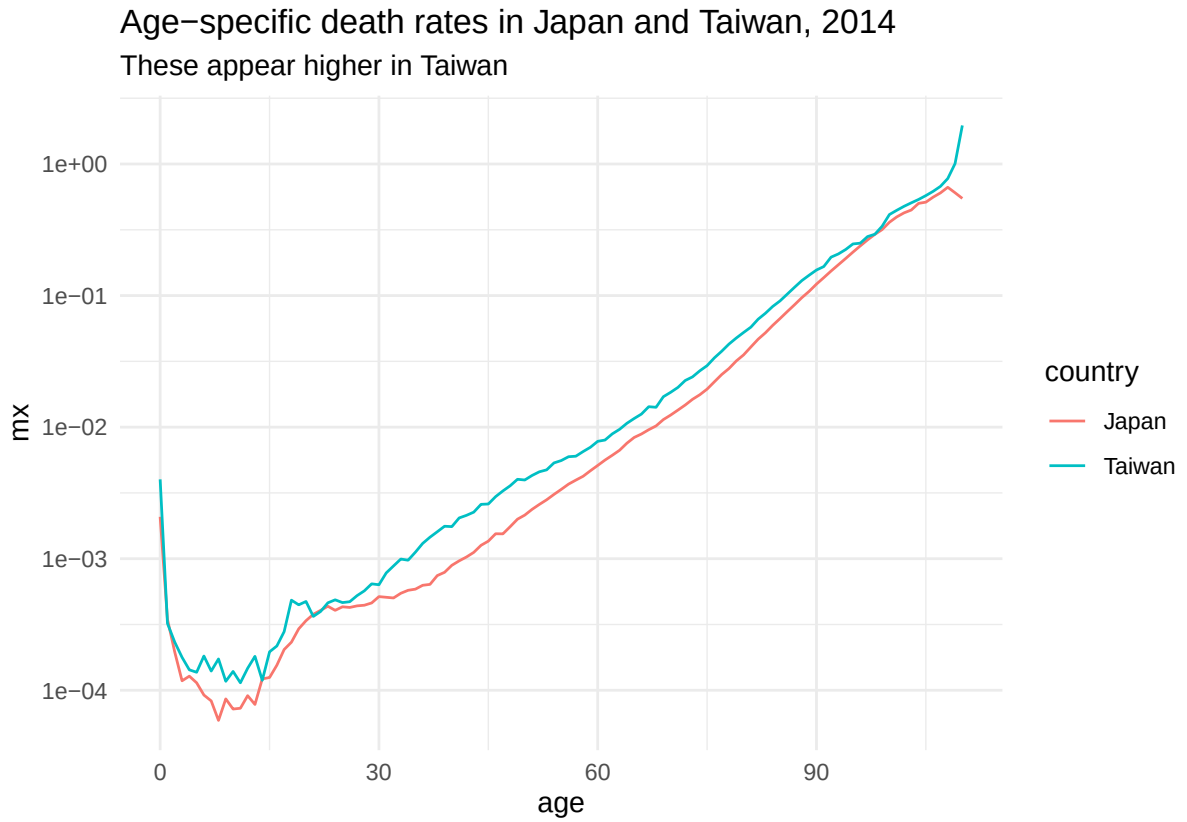
Let's start by comparing the crude mortality rates in Japan and Taiwan in 2014.

```r
DAT |>
  filter(sex == "total",
         year == 2014) |>
  mutate(deaths = mx * exposure) |>
  group_by(country) |>
  summarize(CDR = 1000 * sum(deaths) / sum(exposure))
```

```
## # A tibble: 2 x 2
##   country   CDR
##   <chr>   <dbl>
## 1 Japan    10.2
## 2 Taiwan    6.98
```

Japan has a higher CDR than Taiwan, which *the many* would interpret as Japan having higher mortality than Taiwan. However, if we look at the age-specific death rates, we have a different story.

```r
# Age-specific death rates
DAT |>
  filter(sex == "total",
         year == 2014) |>
  ggplot(aes(x = age, y = mx, color = country)) +
  geom_line() +
  scale_y_log10() +
  labs(title = "Age-specific death rates in Japan and Taiwan, 2014",
       subtitle = "These appear higher in Taiwan") +
  theme_minimal()
```

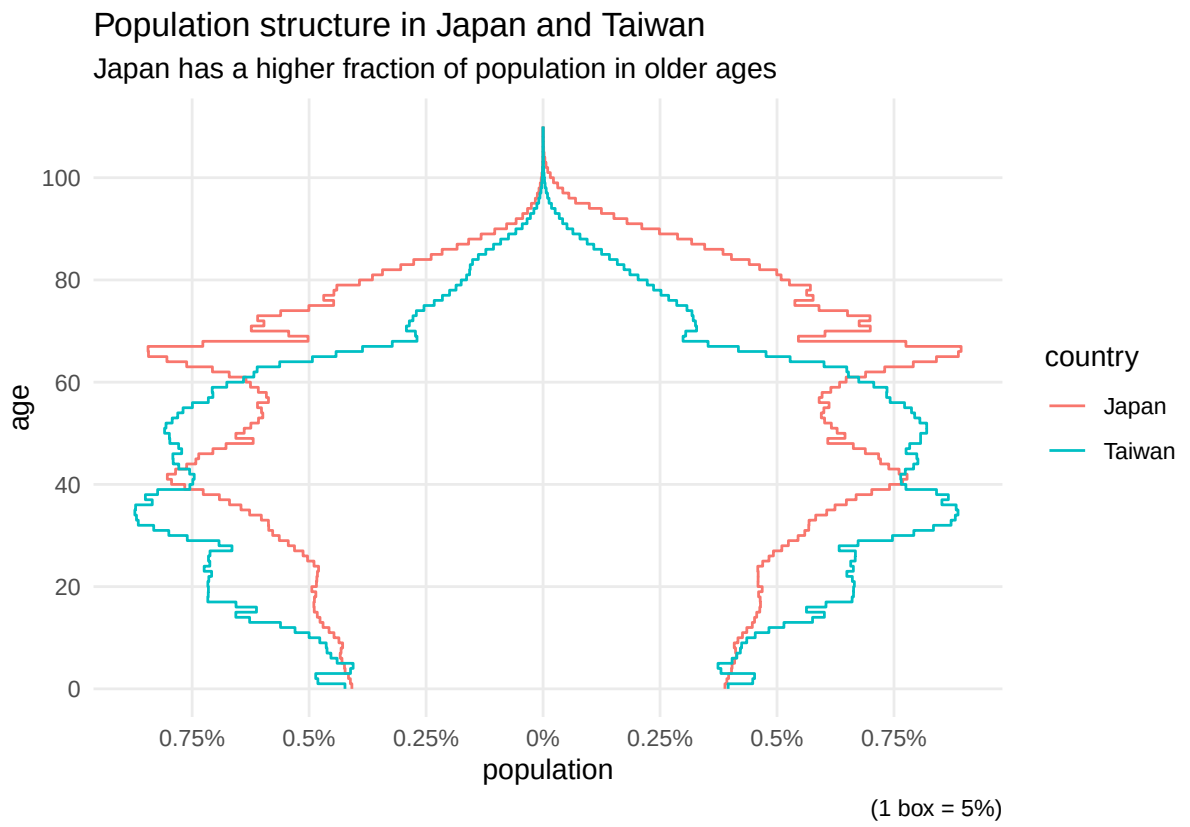


Here, we see that Japan has lower age-specific death rates than Taiwan at all ages, despite having a higher CDR. This occurs because 1) mortality has a strong age gradient: stronger than the international differences in this comparison, and 2) therefore the CDR is very sensitive to the population age structure, which is acting as an *implicit* weight for the mortality rates.

```
breaks = seq(-0.01, 0.01, 0.0025)

DAT |>
  filter(year == 2014,
         sex != "total") |>
  group_by(country) |>
  mutate(structure = exposure / sum(exposure),
         population = ifelse(sex == "male", -structure, structure)) |>
  ungroup() |>
  ggplot(aes(x = age,
             y = population,
             color = country,
             group = interaction(sex, country))) +
  geom_step() +
  coord_flip() +
  scale_y_continuous(breaks = seq(-0.01, 0.01, 0.0025),
                     labels = paste0(as.character(
                         c(seq(.01, 0, -.0025), seq(0.0025, 0.01, 0.0025))*100), "%")) +
  scale_x_continuous(breaks = seq(0,100,by=20)) +
  labs(title = "Population structure in Japan and Taiwan",
       subtitle = "Japan has a higher fraction of population in older ages",
```

```
      caption = "(1 box = 5%)") +
theme_minimal() +
theme(panel.grid.minor = element_blank())
```



The age pyramids indicate that Japan has an older age structure than Taiwan. In 2014, 26% of Japanese population was aged 65 years old or higher, compared with 12% in Taiwan. As death rates are much higher in older ages, an older population will have a higher CDR than younger population. Recall variation over age is typically higher than variation between countries!

3.2 Direct standardization

To avoid the confounding effect of population structure (e.g. age structure) when comparing rates, direct standardization can be used. This method allows us to estimate what the crude rate *would be* if both populations had the same structure.

An important relation between structure-specific rates (r_c) and crude rates (R) is:

$$R = \sum_{c}^{\infty} r_c s_c \tag{1}$$

where s_c is the population structure by component c (for example age, or age and sex). For the crude death rates,

$$CDR = \frac{\sum D_x}{\sum P_x} = \sum_{x}^{\infty} m_x s_x$$

.

where $s_x = \frac{P_x}{\sum(P_x)}$, i.e. the population structure net of its size.

The direct standardization method consists in:

- Finding a *standard* structure (s_c^A), e.g. an average structure between the population compared or the structure of one of these populations.
- Multiplying the component-specific rates (r_c) of the studied population by the standard structure.
- The standardized crude rates are found by summing $s_c^A r_c$

```
# Standardizing CDR of Taiwan and Japan,
# using avg structure as the standard
DAT |>
  filter(year == 2014,
         sex != "total") |>
  # 1. calc structure per country
  group_by(country) |>
  mutate(structure = exposure / sum(exposure)) |>
  # 2. calc the standard, per age (standard is not sex-specific)
  group_by(age) |>
  mutate(standard = mean(structure)) |>
  # 3. rescale standard to sum to 1 per country;
  # note: it's the same standard for each subset
  group_by(country) |>
  mutate(standard = standard / sum(standard)) |>
  # 4. calculate and compare rates
  summarize(CDR = 1000 * sum(mx * structure),
            ASDR = 1000 * sum(mx * standard))
```

```
## # A tibble: 2 x 3
##   country   CDR  ASDR
##   <chr>   <dbl> <dbl>
## 1 Japan   10.2   7.95
## 2 Taiwan   6.98 10.7
```

After standardization, Japan has a lower CDR than Taiwan, the CDR being now consistent with what observed at the age-specific level.

3.3 Indirect standardization

The indirect standardization is used to estimate what would be the crude rates if both populations had the same component-specific rates. This method allows quantifying the effect of population structure on mortality.

The method consists in:

- Finding *standard* component-specific rates (r_c^A).

- Multiplying the population structures (s_c) of the studied population by the standard component-specific rates.
- The standardized crude rates are found by summing $s_c r_c^A$

```
DAT |>
filter(year == 2014,
       sex == "total") |>
group_by(country) |>
mutate(structure = exposure / sum(exposure)) |>
group_by(age) |>
mutate(mx_standard = mean(mx)) |>
group_by(country) |>
summarize(CDR = 1000 * sum(mx * structure),
          ASDR_indirect = 1000 * sum(mx_standard * structure))
```

```
## # A tibble: 2 x 3
##   country   CDR ASDR_indirect
##   <chr>   <dbl>         <dbl>
## 1 Japan   10.2          12.2
## 2 Taiwan   6.98          5.88
```

4 Decomposition methods

Decomposition methods are common tools in demography, used to understand differences in a demographic measure between two or more populations. These methods allow quantifying the exact contribution of specific components, such as ages and causes of death, to this difference between populations.

4.1 Kitagawa decomposition: Decomposing differences in crude rates

Kitagawa decomposition (Kitagawa 1955) aims at quantifying how much of the difference between two crude rates is due to composition effects (e.g. difference in the age-structures) and how much is due to differences in the component-specific rates.

The Kitagawa decomposition (Kitagawa 1955) was the first to decompose the difference between two rates by a composition effect and a rate effect, using multiple standardizations. It brings together both direct and indirect standardizations.

For example, when applied to the CDR, the decomposition is written as:

$$CDR^J - CDR^T = \underbrace{\sum_x^\infty (m_x^J - m_x^T)(\frac{s_x^J + s_x^T}{2})}_{RE:\ rate\ effect} + \underbrace{\sum_x^\infty (s_x^J - s_x^T)(\frac{m_x^J + m_x^T}{2})}_{CE:\ composition\ effect}$$

The left hand side of the equation (named RE) captures how much of the difference in the CDR between Japan and Taiwan is due to difference in age-specific death rates (m_x). This is the same process as finding the difference between the two crude rate after direct standardization, using the average population structure as standard.

The right hand side of the equation (named CE) captures how much of the difference in the CDR is due to age-structure (s_x) differences. This is the same process as finding the difference between the two crude rate after indirect standardization, using the average age-specific rate as standard.

```
# Get data in convenient format for side-by side calcs
DAT_Dec <-
  DAT |>
  filter(year == 2014,
         sex == "total") |>
  group_by(country) |>
  mutate(sx = exposure / sum(exposure)) |>
  ungroup() |>
  select(-exposure) |>
  pivot_wider(names_from = country, values_from = c(mx, sx))
```

This bit of code performs the Kitagawa decomposition

```
DAT_Dec |>
  mutate(
    # calculate standards
         mx_avg = (mx_Taiwan + mx_Japan) / 2,
         sx_avg = (sx_Taiwan + sx_Japan) / 2,
    # weight differences
         RE = (mx_Japan - mx_Taiwan) * sx_avg,
         CE = (sx_Japan - sx_Taiwan) * mx_avg) |>
    # summarize decomp results, compare with original CDR
  summarize(RE = 1000 * sum(RE),
            CE = 1000 * sum(CE),
            CDR_Japan = 1000 * sum(mx_Japan * sx_Japan),
            CDR_Taiwan = 1000 * sum(mx_Taiwan * sx_Taiwan)) |>
  mutate(CDR_diff = CDR_Japan - CDR_Taiwan) |>
  select(-CDR_Japan, -CDR_Taiwan)
```

```
## # A tibble: 1 x 3
##      RE    CE CDR_diff
##   <dbl> <dbl>    <dbl>
## 1 -3.11  6.29     3.17
```

Any kind of *weighted mean* can be decomposed with the Kitagawa method. The CBR, GFR, survival rates/probabilities, neonatal mortality rates, to names only a few, can also be decomposed using this method, as long as the relation between component-specific rates and the component structure, as expressed in equation (1), holds. The components can be age, socioeconomic status, race, etc.

More than one structure/composition effects can also be included. For more information see Kitagawa (1955) and Gupta (1978).

4.2 Arriaga decomposition: Decomposing differences in life expectancy

The Arriaga method (Arriaga 1984) is designed to decompose a *change* in life expectancy by age.

The method is usually expressed using survival probabilities (l_x) and person-years ($_nL_x$ and T_x) in the life table as the primary ingredients. We will calculate a lifetable using tools we created in the second session.

```
#source("https://raw.githubusercontent.com/timriffe/BSSD2025Module2/master/02_lifetables.R")
source("02_lifetables.R")
LT <-
  DAT |>
  filter(sex != "total") |>
  # necessary hacks; better than this = smooth data
  mutate(mx = if_else(exposure == 0, 1, mx),
         mx = if_else(mx == 0,.5,mx)) |>
  group_by(country, year, sex) |>
  group_modify(~lt_full(data = .x, groups = .y)) |>
  ungroup()
```

The difference in life expectancy between Japan and Taiwan is greater than 4 years. The Arriaga method can help figure out which ages (or age-groups) contribute to this difference. BUT, the method can be described as directional (as opposed to symmetrical) because it makes a difference whether we compare Japan with Taiwan or Taiwan with Japan! This is fine, if we're comparing mortality in 2000 with mortality in 2023: the direction of time is ambiguous, and we can imagine a sort of long term consistent secular change. Likewise, one could decompose between an disadvantaged and advantaged population, since we would hope that the disadvantaged group would come to obtain the mortality of the advantaged group, but are not interested in the consequences of the opposite. To compare groups that are qualitatively different, we'd prefer a symmetrical response.

```
#step 2: find the difference in life expectancy
LT |>
  filter(age == 0, year == 1990, sex == "female") |>
  select(country, year, sex,ex)
```

```
## # A tibble: 2 x 4
##   country  year sex       ex
##   <chr>   <dbl> <chr>  <dbl>
## 1 Japan    1990 female  81.9
## 2 Taiwan   1990 female  76.7
```

Let's select just the columns we'll need, and move them side by side, like before:

```
LT_arriaga <-
  LT |>
  filter( year == 1990, sex == "female") |>
  mutate(country = substr(country, 1,1) |> tolower()) |>
```

```
  select(country, age, lx, Lx, Tx) |>
  pivot_wider(names_from = country, values_from = c(lx, Lx, Tx))
```

The method goes in two steps:

1) Find the direct effect.

The direct effect quantifies how much the difference in the number of years lived between age x and $x+n$ contributes to the difference in life expectancy. It is the "*change in life years within a particular age group as a consequence of the mortality change in that age group*" (Arriaga 1984).

$$ {}_nD_x = \frac{l_x^T}{l_0^T}\left(\frac{{}_nL_x^J}{l_x^J} - \frac{{}_nL_x^T}{l_x^T}\right) $$

2) Finding the indirect effect

The indirect effect (and interaction effect) is the "*number of person-years added to a given life expectancy because the mortality change, within a specific age group, will produce a change in the number of survivors at the end of the age interval.*"(Arriaga 1984)

$$ {}_nI_x = \frac{T_{x+n}^J}{l_0^T}\left(\frac{l_x^T}{l_x^J} - \frac{l_{x+n}^T}{l_{x+n}^J}\right) $$

One way to do it with the tidy approach:

```
LT_arriaga <-
  LT_arriaga |>
  mutate(direct = lx_t * (Lx_j / lx_j - Lx_t / lx_t),
         indirect = lead(Tx_j, default = 0) *
           (lx_t / lx_j -
              lead(lx_t, default = 0) / lead(lx_j, default = 0)),
         # impute 0 in the final NA
         indirect = ifelse(is.nan(indirect),0,indirect))
```

The direct and indirect contributions sum to the total differences. The Arriaga formula is then written as:

$$ {}_n\Delta_x = \frac{l_x^T}{l_0^T}\left(\frac{{}_nL_x^J}{l_x^J} - \frac{{}_nL_x^T}{l_x^T}\right) + \frac{T_{x+n}^J}{l_0^T}\left(\frac{l_x^T}{l_x^J} - \frac{l_{x+n}^T}{l_{x+n}^J}\right) $$

where ${}_n\Delta_x$ is the contribution to the difference in life expectancy at birth in age group x to x+n. The last (and open) age-interval consists only of the direct effect.

The sum of the indirect and direct effect gives the total effect of mortality differences in each age.

```
arriaga <-
LT_arriaga  |>
  mutate(total = indirect + direct) |>
  select(age, total)
```

The method is exact in that the sum of age-specific effects equals the life expectancy difference:

```
# decomposition sum
arriaga$total |> sum()
```

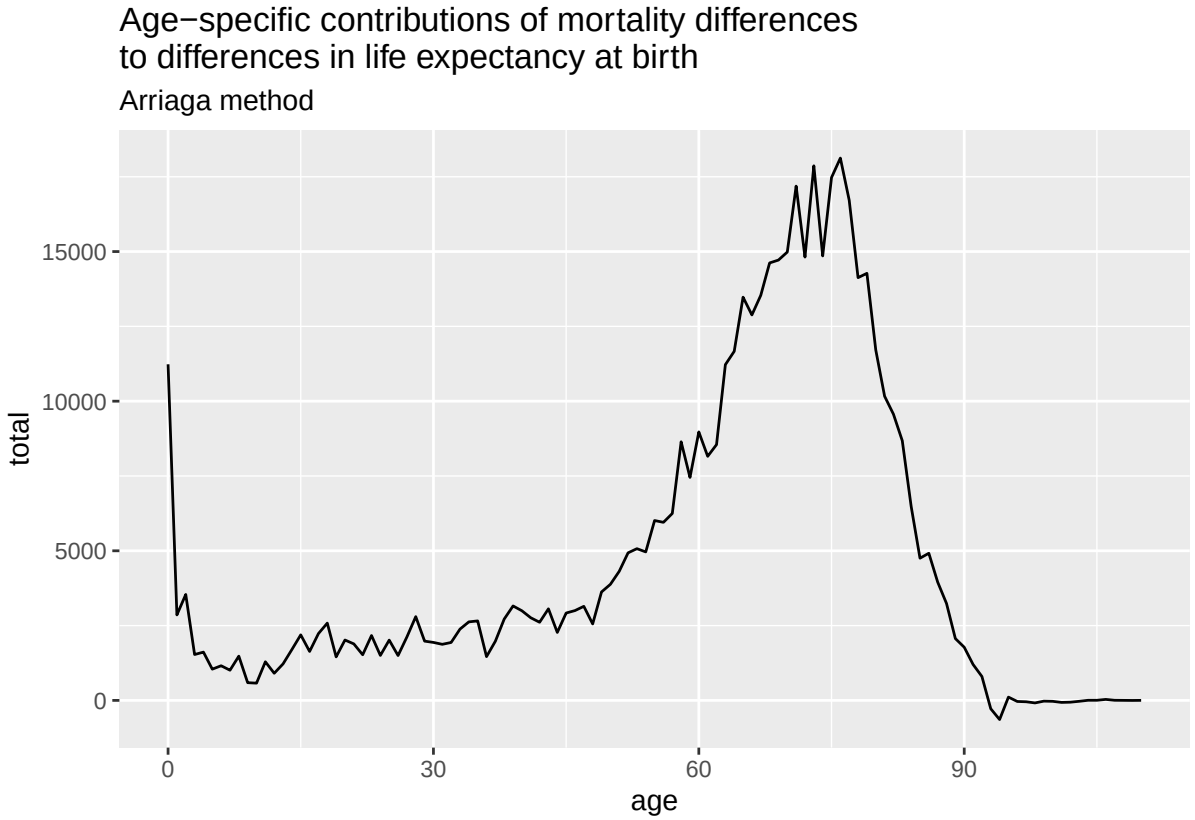
```
## [1] 516900.1
```

```
# it's exact!
LT |>
  filter(age == 0) |>
  pull(ex) |>
  diff()
```

```
##   [1] -5.361696  6.163845 -5.354420  5.814747 -5.415627  5.496955 -5.267310
##   [8]  5.607714 -5.114726  5.625004 -5.131319  5.544190 -5.160873  5.736459
##  [15] -5.197697  5.560498 -5.262833  5.800506 -5.334162  5.335098 -5.381783
##  [22]  5.784781 -5.360920  5.925580 -5.473388  5.569969 -5.589148  6.031215
##  [29] -5.673490  5.951698 -5.657382  6.095222 -5.717028  6.166706 -5.788418
##  [36]  5.687144 -5.749446  6.221954 -5.843357  5.907940 -5.932202  6.242941
##  [43] -6.025155  6.150328 -6.177141  6.323928 -6.190297  6.647412 -6.324557
##  [50]  6.205132 -6.378009  7.102343 -6.485957  6.720553 -6.509079  6.692138
##  [57] -6.719723  6.712750 -6.758499  7.370103 -6.857318  7.179956 -6.848047
##  [64]  7.148963 -6.882032  6.983833 -6.941460  7.188724 -6.907464  6.824305
##  [71] -6.935804  7.222525 -6.801266  6.979990 -6.794697  6.860828 -6.753990
##  [78]  7.136324 -6.833224  6.738183 -6.741667  6.354980 -6.459739  6.964278
##  [85] -6.467111  6.659435 -6.390291  6.604579 -6.327445  6.525507 -6.249471
##  [92]  6.401434 -6.195787  6.332224 -6.203099  6.260474 -6.102999  6.232177
##  [99] -6.082850  6.381342 -6.168324  6.031274 -6.129958  5.634889 -6.048196
## [106] -9.639158 -5.104087  5.528512 -4.972896  5.379824 -4.853320  5.066625
## [113] -4.920920  5.310434 -4.836342  5.226254 -4.816272  5.162514 -4.734391
## [120]  4.884924 -4.897226  5.386682 -4.895346  5.071853 -5.115888  5.186518
## [127] -4.936382  4.978351 -4.896824  5.385345 -4.859100  4.839922 -4.896696
## [134]  5.571427 -4.956965  5.072533 -4.848429  4.998484 -4.926336  5.276362
## [141] -5.088694  4.891205 -5.131879  5.459372 -5.257585  5.547593 -5.304385
## [148]  5.725962 -5.252503  5.264424 -5.292411  5.693865 -5.456746  5.780759
## [155] -5.759694  5.531695 -5.724600  5.908736 -5.735256  6.272587 -5.751356
## [162]  6.008130 -5.858290  5.872537 -5.702163  6.326428 -5.823221  6.162007
## [169] -5.896414  6.203915 -5.699853  5.844544 -5.618981  5.936871 -6.022797
## [176]  6.173311 -6.303849  7.072524 -6.442204  6.474018 -6.171411  6.409361
## [183] -6.232731  6.618564 -6.115129  6.468884 -6.154623  6.111918 -6.422858
## [190]  6.598497 -6.188154  6.634409 -6.197113  6.113487 -6.277466  6.631966
## [197] -6.358490  6.214768 -6.420625  6.713096 -6.198394  6.474959 -6.216982
## [204]  6.438354 -6.273664  6.741126 -6.211945  5.846452 -6.276218
```

Take a look at the age patterns of these contributions to the difference:

```
# age pattern
arriaga |>
  ggplot(aes(x= age, y= total)) +
  geom_line() +
  labs(title = "Age-specific contributions of mortality differences\nto differences in life
       subtitle = "Arriaga method")
```



An extension of the Arriaga method decomposing life expectancy by age AND cause of death is also available (see Preston, Heuveline, and Guillot (2001)). There is also an extension for healthy life expectancy (see Shkolnikov, Andreev, et al. (2017)).

5 NEW: Symmetry in decomposition

It will be convenient to implement the Arriaga steps into a function, for the sake of demonstrating some properties of the method. But I already did that in a separate R package in development, so let's just install that and use it instead. You can install it from GitHub like so. Note: Windows users may need to install RTools from here https://cran.r-project.org/bin/windows/Rtools/ to be able to install R packages from GitHub.

```
library(remotes)
install_github("timriffe/LEdecomp")
```

In this package, everything is a function of rates (yay!), because recall, the three lifetable columns used in the Arriaga method can each be derived from rates, voila:

```
library(LEdecomp)
```

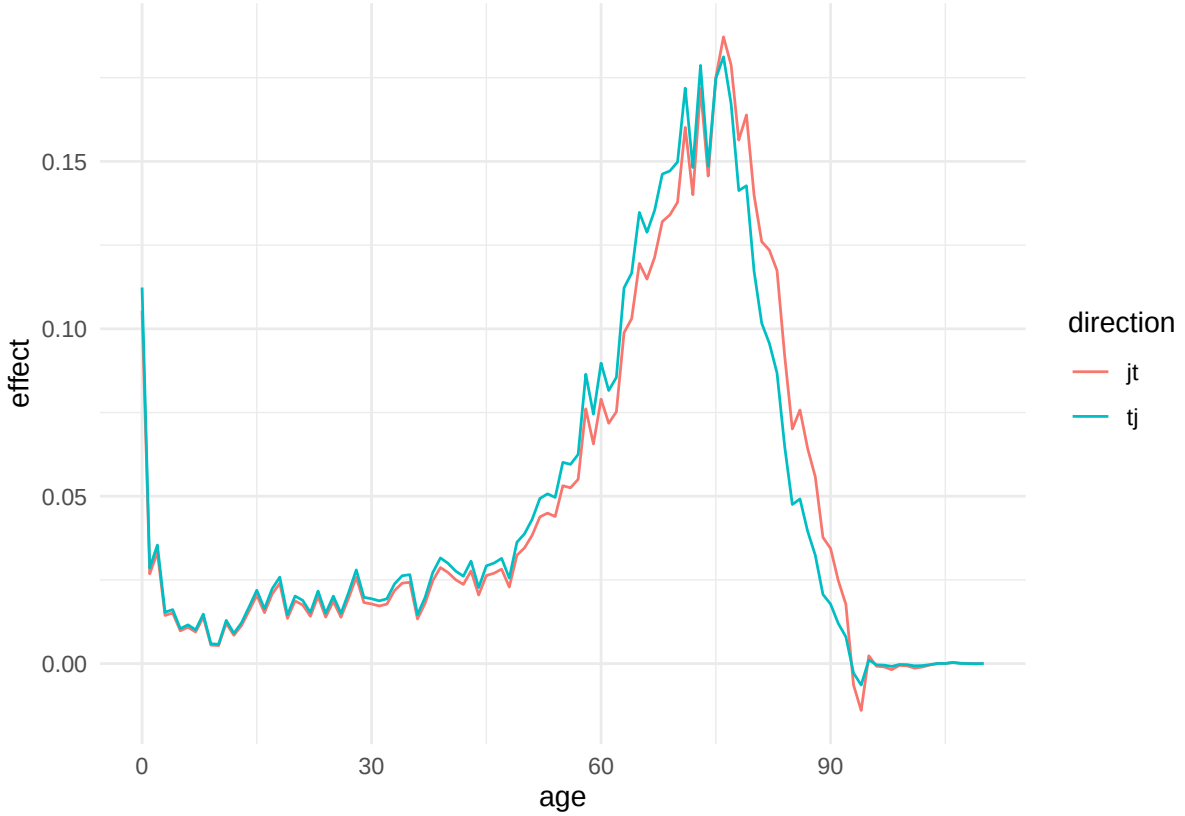
```
##
## Attaching package: 'LEdecomp'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     mx_to_ax
```

```
directional <-
  LT |>
  filter( year == 1990, sex == "female") |>
  select(country, sex, age, mx) |>
  pivot_wider(names_from = country, values_from = mx) |>
  group_by(sex) |>
  mutate(
    sex = substr(sex,1,1),
    jt = -arriaga(Japan, Taiwan, age, sex1 = sex[1]),
    tj = arriaga(Taiwan, Japan, age, sex1 = sex[1])) |>
  select(sex, age, jt, tj) |>
  pivot_longer(c(jt, tj),
               names_to = "direction",
               values_to = "effect")
```

Now note how it makes a difference which direction we decompose (and that you need to flip the sign for one of them to be consistent).

```
directional |>
  ggplot(aes(x=age, y = effect, color = direction)) +
  geom_line() +
  theme_minimal()
```



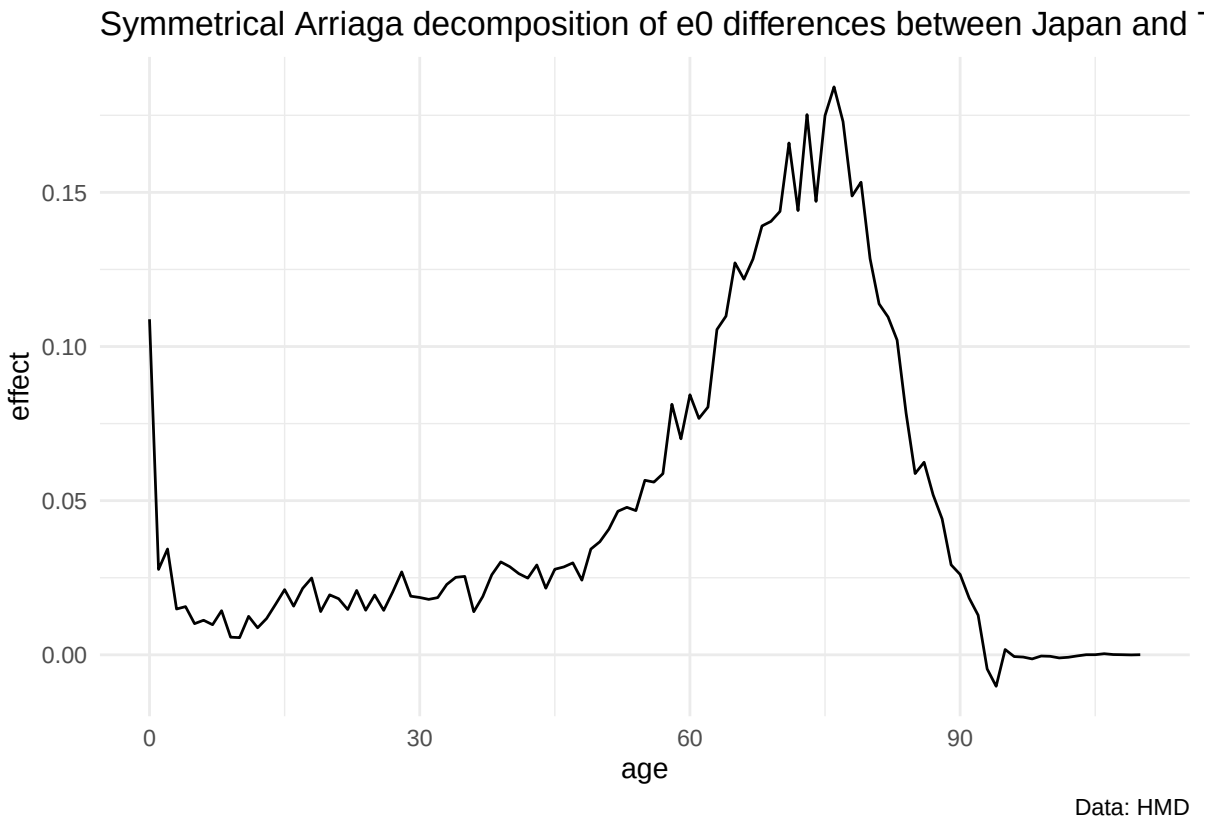
This difference isn't huge: you'd still arrive at the same narrative for what ages contribute the most to life expectancy differences, but for this comparison, the slightly more rigorous thing to do would be to average the effects for each age:

```
directional |>
  group_by(age) |>
  summarize(effect_sym = mean(effect))
```

```
## # A tibble: 111 x 2
##      age effect_sym
##    <dbl>      <dbl>
##  1     0    0.109
##  2     1    0.0277
##  3     2    0.0343
##  4     3    0.0149
##  5     4    0.0156
##  6     5    0.0101
##  7     6    0.0112
##  8     7    0.00977
##  9     8    0.0143
## 10     9    0.00573
## # i 101 more rows
```

Actually the same package has a function for doing exactly this:

```r
LT |>
  filter( year == 1990, sex == "female") |>
  select(country, sex, age, mx) |>
  pivot_wider(names_from = country, values_from = mx) |>
  group_by(sex) |>
  mutate(
    sex = substr(sex,1,1),
    sensitivity = sen_arriaga_sym(Japan, Taiwan, sex1 = sex[1]),
    effect = sensitivity * (Japan - Taiwan)) |>
  ggplot(aes(x=age, y = effect)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Symmetrical Arriaga decomposition of e0 differences between Japan and Taiwan
       caption = "Data: HMD")
```



6 Generalized decomposition

Kitagawa decomposition is applicable to quantities calculated as weighted means, for example where the weights might differ in the groups compared. Arriaga decomposition works with life tables and life expectancy, and so I call it a *bespoke* analytic decomposition. That means that an direct solution has been derived for these settings that allows for the decomposition to be calculated.

A generalized decomposition method is one that applied *any* deterministic function of parame-

ters that produces some synthetic measure based on them. For example, in the case of Kitagawa the parameters are a vector of age structure and demographic rates. To do a life expectancy decomposition with a generalized method, you need a function that converts rates to life expectancy (not the whole lifetable).

Three generalized methods are 1. The method of difference-scaled partial derivatives Caswell (1989) | This method was originally called the lifetable response experiment (LTRE), but it's totally general. | We might also call it the *Caswell* method. 2. The method of step-wise parameter replacement (Andreev, Shkolnikov, and Begun (2002) and Andreev and Shkolnikov (2012)) 3. The method of gradual perturbation (Horiuchi, Wilmoth, and Pletcher (2008)) | This is also known as pseudo-continuous decomposition, as the linear integral method, or else we just call it the *Horiuchi* method.

All three of these are implemented in the `DemoDecomp` `R` package (Riffe (2023)), with the functions `ltre()`, `stepwise_replacement()`, and `horiuchi()`. There's no paper really comparing them, but here's Tim's hand-wavy explanation of their differences:

6.1 LTRE `ltre()`

This method numerically calculates the partial derivatives of your function's parameters half way between the first and second set of parameters, then multiplies them by the observed difference in each parameter. This is a decent approximation of the contribution of each parameter to the difference in the quantity calculated. This method can be blazing fast if you have an analytic partial derivative function on hand. It can also be arbitrarily exact if you either (1) repeat the whole process in small steps between your two steps of parameters, or (2) find the optimal midpoint at which to evaluate the sensitivity.

6.2 Stepwise replacement `stepwise_replacement()`

This method works by swapping out elements of the first parameters, incrementally turning them into the second set of parameters. At each parameter replacement, we recalculate the result. You end up with your result calculated as many times as you have parameters. The moving first differences on this result vector approximates the leverage of that parameter's difference on the result. Since it makes a difference what order you swap the results out, usually one averages the results of going *up* and *down* the parameters. If you have n parameters, it recalculates the result $2 * n$ times. The sum of the parameter-specific contributions is equal to the difference in the summary result. Each-parameter's contribution is approximate, but the sum is exact. Theoretically, if you repeat this over all possible swap order permutations and average the results, then the contribution of each parameter is exact, but this isn't computationally practical.

6.3 Gradual perturbation `horiuchi()`

This method works by interpolating between your n first and second parameters in `N` equal steps. This then becomes the $(n * N)$ *background* against which rate differences are perturbed. At each interpolation point and for each parameter we perturb the parameter both up and down by $1/(2 * N)$ of the amount by which it changed and recalculate the result. The difference between these two calculations is an $n * N$ approximation of the contribution of each parameter at each interpolation point, and summing over the interpolated space within parameters approximates the contribution of that parameter to the difference in your result. For n parameters and N

interpolation points, your result ends up recalculated $2 * n * N$ times, so for large numbers of parameters and large `N` this method can be slow. The result is arbitrarily precise as N increases, and usually $N = 20$ gives a usable result.

6.4 How they work

As far as we're concerned, to use these decomposition methods from `DemoDecomp` package you need to be able to write your code in the form of a function that takes a single vector of parameters.

Here's a function that calculates a crude rate, so we can compare it with Kitagawa. Kitagawa needs two pieces of information, structure and rates (weights and the thing being weighted). We should write the function so that these are stacked in a single vector, e.g. `c(rates, structure)` or vice versa. Your function then needs to be able to unpack the vector and use it to calculate your result.

```
my_CrudeRate <- function(params){

  # first we need to sort out which parameter is which
  n <- length(params)

  # we stacked M on top of Sx, so reshape to a 2 column matrix
  dim(params) <- c(n / 2, 2)

  # return one summary measure
  sum(params[,1] * params[,2])
}
```

You can use the decomposition functions in base or in a tidy setup. For either, you'll want the parameters ordered in the way expected by your function.

```
DAT_Dec2 <-
  DAT |>
  filter(year == 2014,
         sex == "total") |>
  group_by(country) |>
  mutate(sx = exposure / sum(exposure)) |>
  ungroup() |>
  select(-exposure) |>
  pivot_longer(mx:sx,
               names_to = "variable",
               values_to = "pars") |>
  pivot_wider(names_from = country, values_from = pars) |>
  arrange(variable, age)
```

In base, let's check they all at least are additive in the desired way:

```
1000 * (my_CrudeRate(DAT_Dec2$Japan) -
my_CrudeRate(DAT_Dec2$Taiwan))
```

```
## [1] 3.173558
```

```
1000 * ltre(func = my_CrudeRate,
     pars1 = DAT_Dec2$Taiwan,
     pars2 =DAT_Dec2$Japan,
     N = 1) |>
     sum()
```

```
## [1] 3.173558
```

```
1000 * stepwise_replacement(
     func = my_CrudeRate,
     pars1 = DAT_Dec2$Taiwan,
     pars2 =DAT_Dec2$Japan) |>
     sum()
```

```
## [1] 3.173558
```

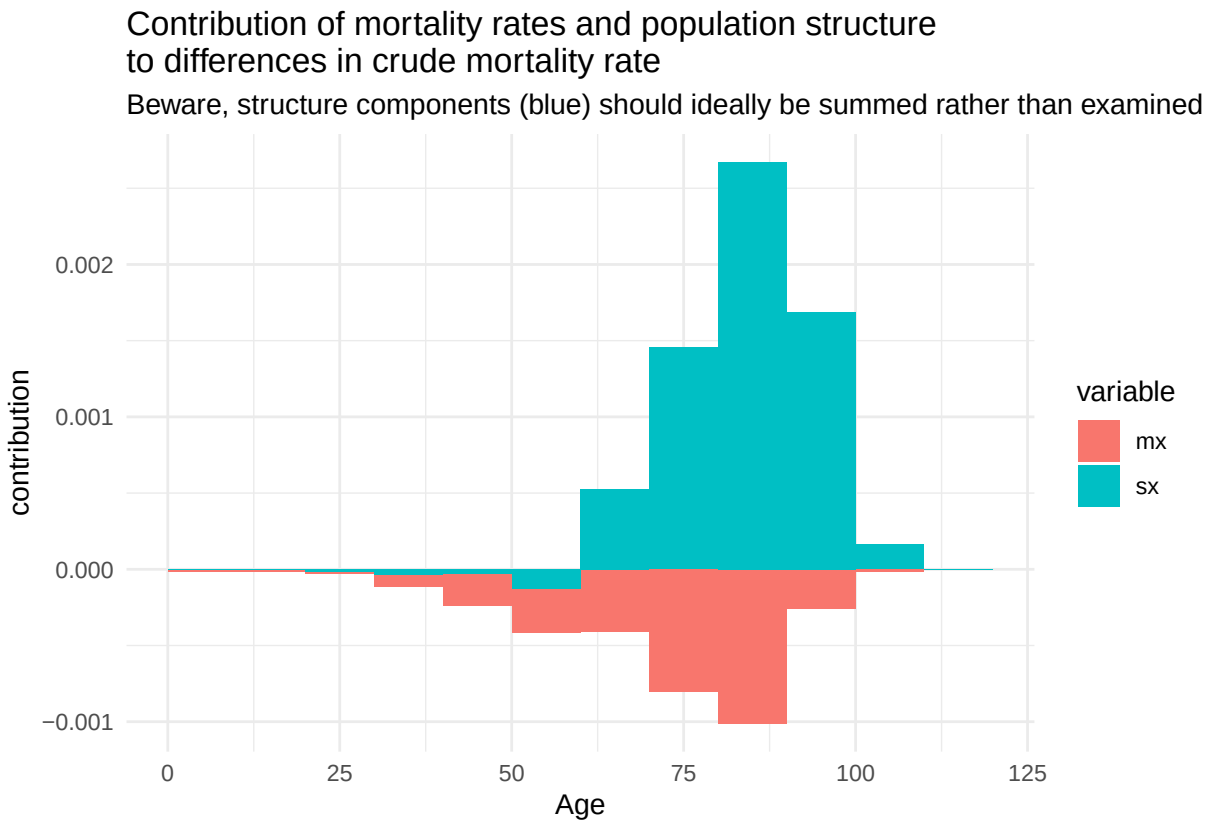
```
1000 * horiuchi(
     func = my_CrudeRate,
     pars1 = DAT_Dec2$Taiwan,
     pars2 =DAT_Dec2$Japan,
     N = 10) |>
     sum()
```

```
## [1] 3.173558
```

Since you end up with one contribution per parameter, decomposition results can sometimes be multidimensional, and can benefit from further aggregation and processing (remember they're additive). Otherwise, you end up with more results than you know what to do with. For that reason, you might do well to just stay in the tidy framework:

```
DAT_Dec2 |>
  mutate(contribution = horiuchi(
     func = my_CrudeRate,
     pars1 = Taiwan,
     pars2 = Japan,
     N = 10
  ),
  # group to 10-year age groups
  age = age - age %% 10) |>
  # sum contributions in groups and by variable
  group_by(variable, age) |>
```

```r
  summarize(contribution = sum(contribution), .groups = "drop") |>
  ggplot(aes(x = age+5, y = contribution, fill = variable)) +
  geom_col(width = 10) +
  xlab("Age") +
  labs(title = "Contribution of mortality rates and population structure\nto differences in
       subtitle = "Beware, structure components (blue) should ideally be summed rather than
  theme_minimal()
```



Exercises

Choose one country from the HMD and select 2 years (ideally over 15 years apart).

Exercise 1

1) Create a function calculating the CDR, standardized death rate (direct) and the Kitagawa decomposition.
2) Calculate the age-specific rate effect and total composition effect of the difference.
3) What factors allowed the CDR to decrease (or increase) over time?

Exercise 2

1) Calculate the life table from these two years.

2) Calculate the age-specific contributions for the change in life expectancy over time using the *directional* Arriaga method.
3) Plot and interpret the results.

References

Andreev, Evgueni M, and Vladimir M Shkolnikov. 2012. "An Excel Spreadsheet for the Decomposition of a Difference Between Two Values of an Aggregate Demographic Measure by Stepwise Replacement Running from Young to Old Ages." *Max Planck Institute for Demographic Research (MPIDR Technical Report TR–2012–002)*.

Andreev, Evgueni M, Vladimir M Shkolnikov, and Alexander Z Begun. 2002. "Algorithm for Decomposition of Differences Between Aggregate Demographic Measures and Its Application to Life Expectancies, Healthy Life Expectancies, Parity-Progression Ratios and Total Fertility Rates." *Demographic Research* 7: 499–522.

Arriaga, Eduardo E. 1984. "Measuring and Explaining the Change in Life Expectancies." *Demography* 21 (1): 83–96.

Caswell, Hal. 1989. "Analysis of Life Table Response Experiments i. Decomposition of Effects on Population Growth Rate." *Ecological Modelling* 46 (3-4): 221–37.

Gupta, Prithwis Das. 1978. "A General Method of Decomposing a Difference Between Two Rates into Several Components." *Demography* 15 (1): 99–112.

Horiuchi, Shiro, John R Wilmoth, and Scott D Pletcher. 2008. "A Decomposition Method Based on a Model of Continuous Change." *Demography* 45 (4): 785–801.

Hyndman, Rob J., Heather Booth, Leonie Tickle, and John Maindonald. 2017. *Package 'demography'*. https://cran.r-project.org/web/packages/demography/index.html.

Kitagawa, Evelyn M. 1955. "Components of a Difference Between Two Rates." *Journal of the American Statistical Association* 50 (272): 1168–94.

Preston, Samuel H., Patrick Heuveline, and Michel Guillot. 2001. *Demography: Measuring and Modeling Population Processes*. Oxford: Blackwell.

Riffe, Tim. 2023. *DemoDecomp: Decompose Demographic Functions*. https://github.com/timriffe/DemoDecomp.

Shkolnikov, Vladimir M, Evgeny M Andreev, et al. 2017. "The Decomposition of the Difference Between Two Healthy Life Expectancies. Which Formula Is Right." *Max Planck Institute for Demographic Research*.