Describing and Estimating Migration Age Structures

Guy J. Abel

Rogers Castro migration age schedules

- Populations tend to experience demographic events, such fertility, mortality and migration, with persistent regularities in the age-specific rates
- Demographers have summarsied regularities in rates using mathematical expressions called model schedules.
- Rogers and Castro (1981) first proposed a migration model schedules via an analysis of over 500 age profiles of migration

Rogers Castro migration age schedules

Composed of curves based on migration of different life stages:

1. Pre-labor force
2. Labor force
3. Post-labor force
4. Post-retirement
5. A constant term

$$
\begin{aligned}
m(x) = & a_1 \exp(-\alpha_1 x) \\
& + a_2 \exp(-\alpha_2(x - \mu_2) - \exp(\lambda_2(x - \mu_2))) \\
& + a_3 \exp(-\alpha_3(x - \mu_3) - \exp(\lambda_3(x - \mu_3))) \\
& + a_4 \exp(\lambda_4 x) \\
& + c
\end{aligned}
$$

Rogers Castro migration age schedules

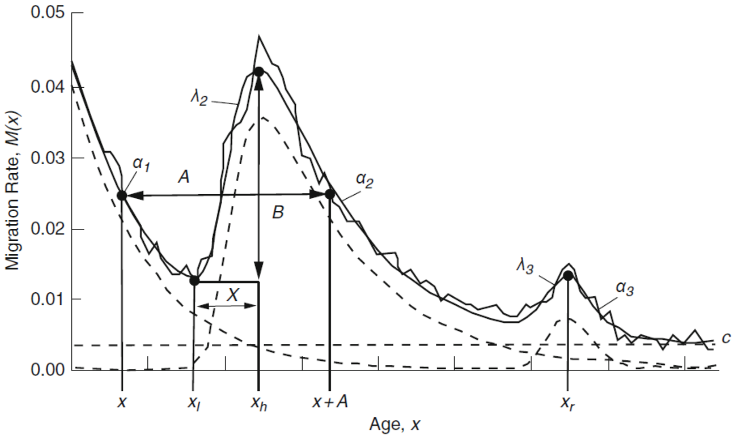


Fig. 2.4 The model migration schedule fitted to the observed out-migration rates of males leaving the Stockholm region, 1974.
(*Source*: Rogers & Castro, 1981)

Rogers Castro migration age schedules

- Most migration age patterns have a pre-labor force downward slope and labor force peak (and a constant)
  - 7-parameter model schedule
- In specific areas (in Western countries) migration age patterns have an additional retirement peak component
  - 11-parameter model schedule
- In other areas, instead of a retirement peak, age profiles have an upward slope at the end of life
  - 9-parameter model schedule
- In even fewer cases, some instances of both a retirement peak and a post-retirement upward slope Rogers and Watkins (1987)
  - 13-parameter model schedule
- Wilson (2010) introduced a 17-parameter model to incorporate a student peak before the labour force peak.

Rogers Castro migration age schedules

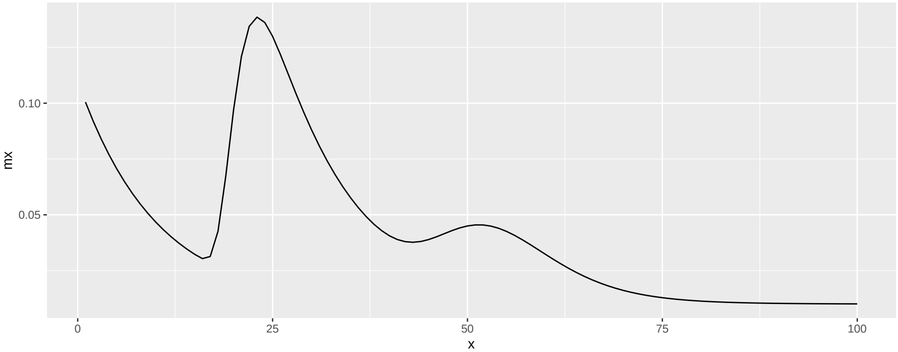
- The `mig_calculate_rc()` function in either the *rcbayes* package by Monica Alexander et. al. provide a quick method to calculate migration age schedules for a given parameter set
  - Also available in *DemoTools*.
  - Same functions by same authors. *DemoTools* not on CRAN presently.

Rogers Castro migration age schedules

```
> library(rcbayes)
> # define 11 parameters
> p <- c(a1 = 0.1, alpha1 = 0.1,
+        a2 = 0.2, alpha2 = 0.1, mu2 = 20, lambda2 = 0.5,
+        a3 = 0.05, alpha3 = 0.2, mu3 = 60, lambda3 = 0.1,
+        c = 0.01)
>
> # calculate model migration schedule with 11 parameters
> mx <- mig_calculate_rc(ages = 1:100, pars = p)
> mx
  [1] 0.10048374 0.09187308 0.08408182 0.07703200 0.07065307 0.06488116
  [7] 0.05965853 0.05493290 0.05065697 0.04678794 0.04328711 0.04011942
 [13] 0.03725318 0.03465970 0.03231470 0.03037404 0.03132289 0.04264948
 [19] 0.06746077 0.09710942 0.12091621 0.13442550 0.13855839 0.13616639
 [25] 0.12995494 0.12185875 0.11308333 0.10431583 0.09591794 0.08806055
 [31] 0.08080783 0.07416691 0.06811661 0.06262465 0.05765908 0.05319728
 [37] 0.04923336 0.04578295 0.04288297 0.04058442 0.03893812 0.03797602
 [43] 0.03769285 0.03803360 0.03889051 0.04011081 0.04151310 0.04290860
 [49] 0.04412234 0.04501067 0.04547234 0.04545269 0.04494158 0.04396660
 [55] 0.04258384 0.04086761 0.03890096 0.03676762 0.03454596 0.03230498
 [61] 0.03010206 0.02798231 0.02597892 0.02411422 0.02240126 0.02084543
 [67] 0.01944616 0.01819839 0.01709398 0.01612274 0.01527339 0.01453424
 [73] 0.01389365 0.01334046 0.01286417 0.01245512 0.01210453 0.01180452
 [79] 0.01154811 0.01132916 0.01114229 0.01098283 0.01084676 0.01073060
 [85] 0.01063138 0.01054657 0.01047399 0.01041181 0.01035847 0.01031264
```

Rogers Castro migration age schedules

```
> library(tidyverse)
> tibble(x = 1:100,
+        mx = mx) %>%
+   ggplot(mapping = aes(x = x, y = mx)) +
+   geom_line()
```



Model migration age schedules

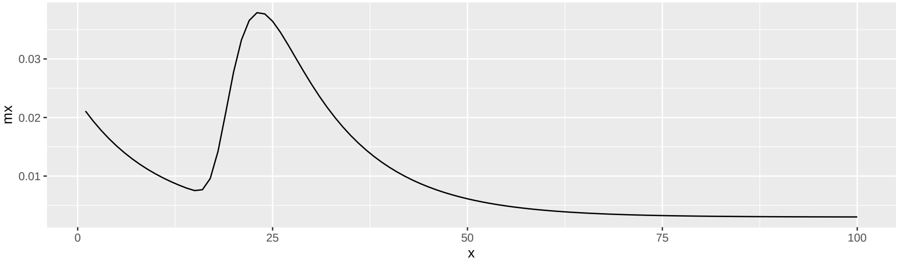
- The *migest* package contains two sets of parameters for model migration schedules.
- The `rc_model_fund` are the set of fundamental parameters proposed by Rogers and Castro to represent a typical migration age pattern, based on their analysis of over 500 migration flows

```
> library(migest)
> rc_model_fund
# A tibble: 7 x 2
  param     value
  <chr>     <dbl>
1 a1        0.02
2 alpha1    0.1
3 a2        0.06
4 alpha2    0.1
5 mu2      20
6 lambda2   0.4
7 c         0.003
```

Model migration age schedules

- Plot of model age schedule based on fundamental parameters

```
> # convert data frame to named vector
> p <- deframe(rc_model_fund)
> p
     a1  alpha1      a2  alpha2     mu2 lambda2       c
  2e-02   1e-01   6e-02   1e-01   2e+01   4e-01   3e-03
>
> tibble(x = 1:100,
+        mx = mig_calculate_rc(ages = x, pars = p)) %>%
+   ggplot(mapping = aes(x = x, y = mx)) +
+   geom_line()
```



Model migration age schedules

- Rogers and Castro describe the nice properties in the parameters and their relationships
- Peaking: early versus late peaking (μ_2)
  - $\mu_2 = 20$ in the fundamental parameters
- Dominance: $\gamma_{12} = a_1/a_2$ as the index of child dependency, and $1/\gamma_{12}$ as index of labor dominance
  - $\gamma_{12} = 1/3$ in fundamental parameters
- Labor asymmetry: $\sigma_2 = \lambda_2/\alpha_2$
  - $\sigma_2 = 4$ in fundamental parameters
- Regularity: $\beta_{12} = \alpha_1/\alpha_2$ how the migration rates of children match to the migration rates of parents
  - $\beta_{12} = 1$ in fundamental parameters
- Users can focus on these four measures (peaking, dominance, labor asymmetry and regularity) when describing or deriving their own model schedules

Model migration age schedules

- The `index_age_rc()` function in the migest package returns these ratios given a named vector of the parameters

```
> rc_model_fund %>%
+   deframe() %>%
+   index_age_rc()
# A tibble: 5 x 2
  measure           value
  <chr>             <dbl>
1 peaking          20
2 child_dependency  0.333
3 labor_dependency  3
4 labor_asymmetry   4
5 regularity        1
```

Model migration age schedules

- The `rc_model_un` are the set of fundamental parameters proposed in United Nations Department of Economic and Social Affairs Population Division (1992) for estimating age-specific migration flows in different contexts

```
> rc_model_un
# A tibble: 84 x 5
   schedule         schedule_abb sex   param     value
   <chr>            <chr>        <chr> <chr>     <dbl>
 1 Western standard ws           male  a1       0.0215
 2 Western standard ws           male  alpha1   0.105
 3 Western standard ws           male  a2       0.0694
 4 Western standard ws           male  alpha2   0.112
 5 Western standard ws           male  mu2     20.0
 6 Western standard ws           male  lambda2  0.391
 7 Western standard ws           male  c        0.0028
 8 Low dependency   ld           male  a1       0.0128
 9 Low dependency   ld           male  alpha1   0.105
10 Low dependency   ld           male  a2       0.0804
# ... with 74 more rows
# i Use `print(n = ...)` to see more rows
```

Model migration age schedules

- To calculate model schedules we can use
  - `nest()` to group together the parameters
  - `map()` to apply the parameters to the `mig_calculate_rc()` function for each group

```
> d <- rc_model_un %>%
+   select(-schedule_abb) %>%
+   nest(rc_param = c(param, value)) %>%
+   mutate(p = map(.x = rc_param, .f = ~deframe(.x)),
+          mx = map(.x = p,
+                   .f = ~mig_calculate_rc(ages = 1:80, pars = .x)),
+          age = list(1:80))
> d
# A tibble: 12 x 6
   schedule                              sex    rc_param p          mx     age
   <chr>                                 <chr>  <list>   <list>     <list> <list>
 1 Western standard                      male   <tibble> <dbl [7]> <dbl>  <int>
 2 Low dependency                        male   <tibble> <dbl [7]> <dbl>  <int>
 3 High dependency                       male   <tibble> <dbl [7]> <dbl>  <int>
 4 Young labour force entry              male   <tibble> <dbl [7]> <dbl>  <int>
 5 Old labour force entry                male   <tibble> <dbl [7]> <dbl>  <int>
 6 Low dependency low labour force entry male   <tibble> <dbl [7]> <dbl>  <int>
 7 Western standard                      female <tibble> <dbl [7]> <dbl>  <int>
 8 Low dependency                        female <tibble> <dbl [7]> <dbl>  <int>
 9 High dependency                       female <tibble> <dbl [7]> <dbl>  <int>
```

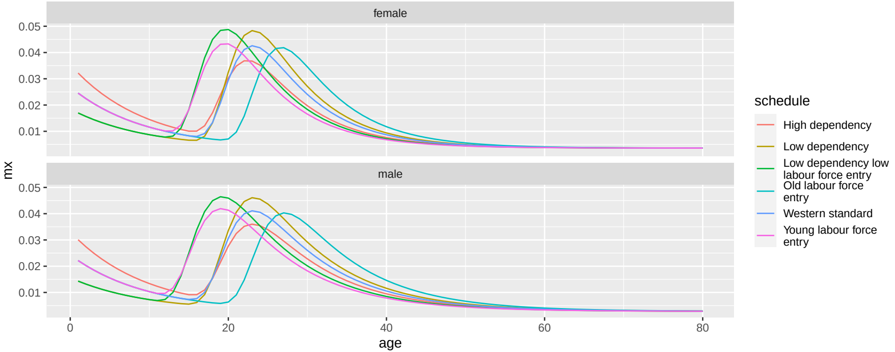
Model migration age schedules

```
> # first row parameters
> d$p[[1]]
     a1  alpha1      a2  alpha2     mu2 lambda2       c
 0.0215  0.1050  0.0694  0.1120 20.0400  0.3910  0.0028
>
> # data unnested
> d %>%
+   select(-rc_param, -p) %>%
+   unnest(c(mx, age))
# A tibble: 960 x 4
   schedule         sex       mx   age
   <chr>            <chr>  <dbl> <int>
 1 Western standard male  0.0222     1
 2 Western standard male  0.0202     2
 3 Western standard male  0.0185     3
 4 Western standard male  0.0169     4
 5 Western standard male  0.0155     5
 6 Western standard male  0.0143     6
 7 Western standard male  0.0131     7
 8 Western standard male  0.0121     8
 9 Western standard male  0.0112     9
10 Western standard male  0.0103    10
# ... with 950 more rows
# i Use `print(n = ...)` to see more rows
```

Model migration age schedules

- Use `unnest()` to create a data base varying by age for each model schedule and sex for plotting

```
> d %>%
+   unnest(c(mx, age)) %>%
+   mutate(schedule = str_wrap(schedule, width = 20)) %>%
+   ggplot(mapping = aes(x = age, y = mx, colour = schedule)) +
+   geom_line() +
+   facet_wrap(facets = "sex", ncol = 1)
```



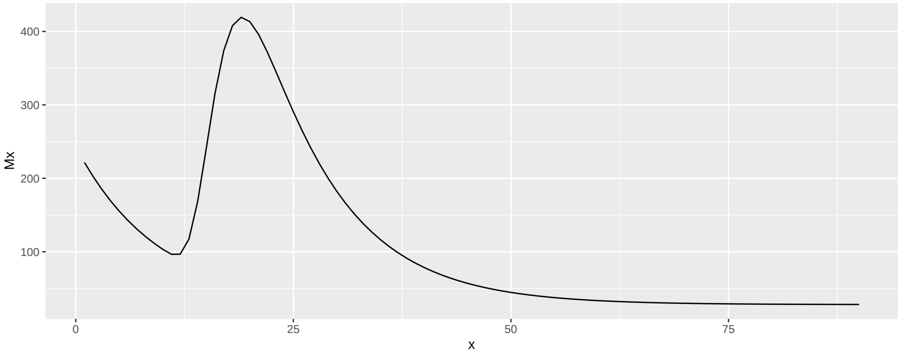
Model migration age schedules

- Model migration schedules are useful when we do not have any age information, but require an estimate of age specific migration
  - For example, in cohort component projections age specific migration rates are required but might not be available in any data source
- We may use an estimate or reported data on total migration to obtain age-specific migration
  - Design or select appropriate model age schedule based on existing knowledge of migration age patterns for the given flow.

```
> # example for males based on young labour force entry
> p <- rc_model_un %>%
+   filter(sex == "male", schedule_abb == "ylfe") %>%
+   select(param, value) %>%
+   deframe()
> p
     a1  alpha1      a2  alpha2     mu2 lambda2       c
 0.0215  0.1050  0.0691  0.1120 16.0900  0.3910  0.0028
```

Model migration age schedules

```
> tibble(x = 1:90,
+        mx = mig_calculate_rc(ages = x, pars = p),
+        # calculate number of migrants, given a total estimate of 10,000
+        Mx = 10000 * mx) %>%
+   ggplot(mapping = aes(x = x, y = Mx)) +
+   geom_line()
```



Fitting Roger Castro migration age schedules

- If we have age-specific migration data we might want to estimate the parameters of a Rogers Castro age schedule to
  - Smooth the data
  - Analyse the parameter estimates
  - Create projected age schedules based on past patterns of the age schedule parameters
- Fitting Rogers Castro migration age schedules can be difficult.
  - A number of different software has been used to fit age schedules including Rogers and Little (1994), TableCurve 2D Rogers and Raymer (1999), MATLAB Rogers, Raymer, and Little (2010), and Excel Wilson (2010).
- The `mig_estimate_rc()` function in rcbayes uses Stan, via the rstan package, a Bayesian probabilistic programming language
  - Estimation is carried out using MCMC sampling.
- Requires two arguments
  - `ages` a vector of migration ages
  - `mx` a vector of standardized migration intensities for the corresponding ages
  - Specify form of age schedule using the `pre_working_age`, `working_age`, `retirement` and `post_retirement` arguments - set to `TRUE` or `FALSE`

Fitting Roger Castro migration age schedules

- Demonstrate with five-year data from the `italy_area` data set in *migest*
  - Calculate the out-migration for Islands (Sicily and Sardinia) in 1970

```
> # include a numeric age column for mig_estimate_rc()
> i <- italy_area %>%
+   filter(year == 1970) %>%
+   group_by(age_grp) %>%
+   sum_region() %>%
+   filter(region == "Islands") %>%
+   separate(col = age_grp, into = c("age_min", "age_max"),
+            remove = FALSE, convert = TRUE)
> i
# A tibble: 20 x 8
# Groups:   age_grp [20]
   age_grp age_min age_max region  out_mig in_mig  turn   net
   <fct>     <int>   <int> <chr>     <dbl>  <dbl> <dbl> <dbl>
 1 0-4           0       4 Islands    7876   4532 12408 -3344
 2 5-9           5       9 Islands    7271   3592 10863 -3679
 3 10-14        10      14 Islands    5779   2228  8007 -3551
 4 15-19        15      19 Islands    8526   3064 11590 -5462
 5 20-24        20      24 Islands   15629   6861 22490 -8768
 6 25-29        25      29 Islands   11224   5891 17115 -5333
 7 30-34        30      34 Islands    7046   4042 11088 -3004
 8 35-39        35      39 Islands    4612   2480  7092 -2132
 9 40-44        40      44 Islands    3634   1737  5371 -1897
```

Fitting Roger Castro migration age schedules

- Requires a standardized age schedule (where values sum to one)
- Will take a few minutes and print out lots of messages from Stan

```
> m <- i$out_mig/sum(i$out_mig)
> m
 [1] 0.0965527387 0.0891359780 0.0708453881 0.1045211592 0.1915976070
 [6] 0.1375962340 0.0863776786 0.0565390085 0.0445496004 0.0341170990
[11] 0.0210366302 0.0194552052 0.0149193351 0.0113274163 0.0086058942
[16] 0.0060069632 0.0032854411 0.0014220566 0.0004290688 0.0016794979
>
> f <- mig_estimate_rc(ages = i$age_min + 2.5, mx = m,
+                      # set model components
+                      pre_working_age = TRUE, working_age = TRUE,
+                      retirement = FALSE, post_retirement = FALSE)
mig_estimate_rc is running normal model, Using arguments ages and mx

SAMPLING FOR MODEL 'rcmodel_normal' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 secon
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
```

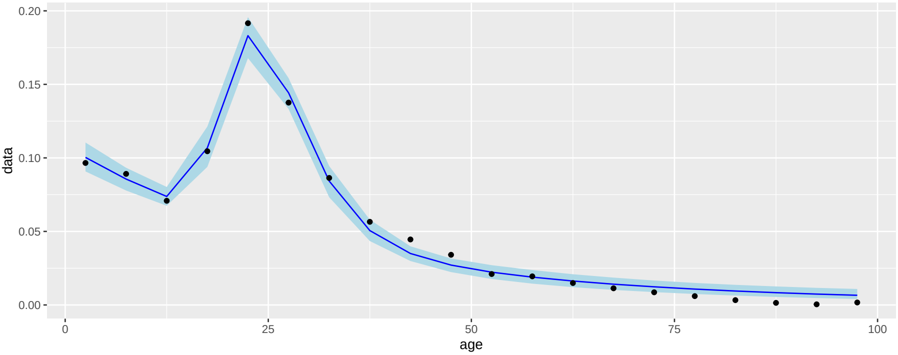
Fitting Roger Castro migration age schedules

The fitted object has two components

```
> # parameter estimates
> f[[1]]
# A tibble: 7 x 4
  variable   median      lower     upper
  <chr>       <dbl>      <dbl>     <dbl>
1 a1         0.107     0.0949     0.118
2 a2         0.341     0.277      0.382
3 alpha1     0.0322    0.0269     0.0428
4 alpha2     0.226     0.165      0.295
5 c          0.00154   0.0000646  0.00777
6 lambda2    0.185     0.150      0.259
7 mu2       24.5      21.4       27.0
>
> # fitted schedule
> f[[2]]
# A tibble: 20 x 6
     age    data  median   lower  upper     diff_sq
   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>       <dbl>
 1   2.5 0.0966   0.100   0.0908  0.110  0.0000144
 2   7.5 0.0891   0.0856  0.0778  0.0933 0.0000124
 3  12.5 0.0708   0.0738  0.0675  0.0802 0.00000856
 4  17.5 0.105    0.107   0.0939  0.121  0.00000548
 5  22.5 0.192    0.183   0.168   0.196  0.0000717
```

Fitting Roger Castro migration age schedules

```
> ggplot(data = f[[2]],
+        mapping = aes(x = age, y = data)) +
+   geom_ribbon(mapping = aes(ymin = lower, ymax = upper), fill = "lightblue") +
+   geom_line(mapping = aes(y = median), colour = "blue") +
+   geom_point()
```



Fitting Roger Castro migration age schedules

- The *migraR* package by Ruiz-Santacruz and Garcés also has functions to estimate parameters in Rogers Castro schedule
  - Also not on CRAN
  - Uses an optimization procedure (non-Bayesian)
  - Functions to select best form schedule
- Selecting the form of the schedule usually requires some form of visual inspection

Age Indices

- Number of criticisms of model age schedules for migration (Bell et al. (2002), Bernard, Bell, and Charles-Edwards (2014))
- Not always clear how many parameters should be included in model schedule
  - Parameter estimates sensitive to the choice of model form, making comparisons difficult
  - Use statistical accuracy measures to select best form, at the risk of over fitting
- Parameter estimates sensitive to initial values
  - Unlikely to be the case when using `mig_estimate_rc()`
- Unstable parameter estimates
  - Sensitive to measurement error in age-specific migration
- Interpretation of parameter estimates
  - The indexes in `index_age_rc()` have not been widely adopted, probably because of difficulty in fitting model schedules.

Age Indices

- A number of other measures of age specific migration have been proposed that do not require fitting model age schedules.
- Most a dependent on the migration intensity m_{as}, the number of migrants in a age group and given time period as a percentage of the population at risk of moving.
- Rogers (1975) proposed a Gross Migraproduction Rate (GMR) based on the sum of age-specific (and sex-specific) migration intensities

$$GMR = \sum_{as} m_{as}$$

- Bell et al. (2002) introduced
  - Peak migration intensity, the largest age-specific migration intensity of any age-group
  - Peak age, the corresponding age of the peak migration intensity

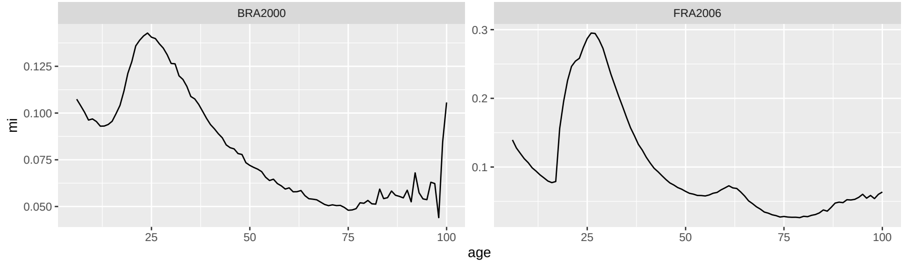
Age Indices

- Bell and Muhidin (2009) proposed and additional measures
  - Breadth of peak based on the sum of the peak migration intensity at the peak age and the five age-groups before and after the peak.
  - Peak share based on the percentage of the normalized migration age schedule covered by the peak age and the five age-groups before and after the peak.
- Bernard, Bell, and Charles-Edwards (2014) provide three additioanl measures
  - The Maximum Upward Rate of Change (MURC) for the largest gradient in the slope of the labour force peak before the peak age
  - The Maximum Downward Rate of Change (MDRC) for the largest gradient in the slope of the labour force peak after the peak age
  - The asymmetry of the labour force peak based on the ratio of MURC and MDRC
- Each of these measures area calculated in the `age_index()` function in the *migest* package

Age Indices

- To demonstrate we use the age schedule data of Brazil 2000 and France 2006 in the `ipumsi_age` data frame of the *migest* package
  - Migration based on five-year transitions between any minor (and major) administrative units.

```
> ipumsi_age %>%
+   mutate(mi = migrants/population) %>%
+   filter(age > 5) %>%
+   ggplot(mapping = aes(x = age, y = mi)) +
+   geom_line() +
+   facet_wrap(facets = "sample", scales = "free")
```



Age Indices

- Bernard, Bell, and Charles-Edwards (2014) recommends smoothing age schedules before calculating index values
  - Get very similar results without smoothing - at least in these examples
- `index_age()` by default ignores values above 65 (and below 5) when calculating peak index statistics
  - GMR still sensitive for outliers (e.g. oldest in Brazil)
- Index values for Brazil 2000

```
> ipumsi_age %>%
+   filter(sample == "BRA2000") %>%
+   mutate(mi = migrants/population) %>%
+   index_age()
# A tibble: 8 x 2
  measure      value
  <chr>        <dbl>
1 gmr           7.82
2 peak_mi      14.3
3 peak_age     24
4 peak_breadth 147.
5 peak_share   18.8
6 murc         19
7 mdrc         32
8 asymmetry     0.594
```

Age Indices

- Index values are most useful for comparing age-specific migration in different countries (or regions or time periods)

```
> ipumsi_age %>%
+   mutate(mi = migrants/population) %>%
+   group_nest(sample) %>%
+   rowwise() %>%
+   mutate(i = list(index_age(data))) %>%
+   unnest(i) %>%
+   select(-data) %>%
+   pivot_wider(names_from = sample, values_from = value)
# A tibble: 8 x 3
  measure      BRA2000 FRA2006
  <chr>          <dbl>   <dbl>
1 gmr            7.82    9.55
2 peak_mi       14.3    29.5
3 peak_age      24      26
4 peak_breadth 147.    295.
5 peak_share    18.8    30.8
6 murc          19      18
7 mdrc          32      30
8 asymmetry      0.594   0.6
```

General purpose smoothing functions

- There are many non-parametric smoothing functions in R that can be used to smooth data.
- The `stats` package, which is loaded when R opens, includes
  - `ksmooth()` is a kernel regression smoother
  - `loess.smooth()` is a Local Polynomial Regression Fitting method
  - `smooth.spline` a cubic spline fit
- The DemoTools package contains a `smooth_age_5()` that is particularly useful for age-heaped data.

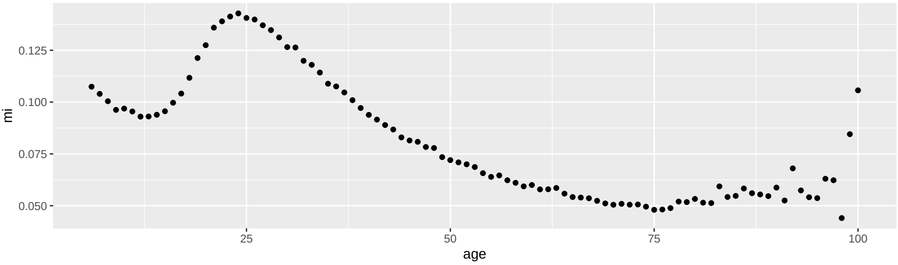
General purpose smoothing functions

- Smoothing methods perform some form of weighting data points on separate subsections (windows or bandwidths of the data)
- In a migration age schedule context, this involves some form of simple local regression or averaging of migration intensities at each age, given data from nearby ages.
- Careful consideration is usually required in choosing the bandwidth.
  - The default values are not always sensible for migration age schedules
- Might consider censoring the very oldest values where values can become volatile due to small numbers

General purpose smoothing functions

- Use Brazil 2000 IPUMS International sample data to demonstrate
  - Particularly rough at older age groups

```
> b <- ipumsi_age %>%
+   filter(sample == "BRA2000",
+          age > 5) %>%
+   mutate(mi = migrants/population)
>
> ggplot(data = b, mapping = aes(x = age, y = mi)) +
+   geom_point()
```



General purpose smoothing functions

- Most smoothing function in R require two vectors (`x` and `y`)
  - Optional arguments to control the smoothness of the fit(names differ for different smoothing functions)
- Will return a list with two components (`x` and `y`), where the length of `x` may differ from the original vector provided
  - Set a output length argument (names differ for different smoothing functions)
  - The `x` component will match age values
  - Can use within `mutate()`

```
> k1 <- ksmooth(x = b$age, y = b$mi)
> str(k1)
List of 2
 $ x: num [1:100] 6 6.95 7.9 8.85 9.8 ...
 $ y: num [1:100] 0.1074 0.104 0.1004 0.0962 0.0969 ...
>
> k2 <- ksmooth(x = b$age, y = b$mi, n.points = nrow(b))
> str(k2)
List of 2
 $ x: num [1:95] 6 7 8 9 10 11 12 13 14 15 ...
 $ y: num [1:95] 0.1074 0.104 0.1004 0.0962 0.0969 ...
```

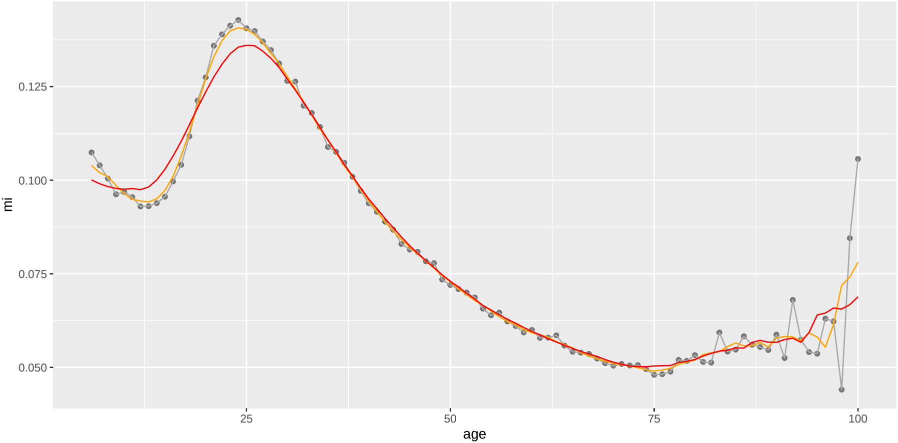
Kernal smoothing

- The `ksmooth` function is unlikely to smooth a migration age schedule as the default bandwidth parameter is too small
  - Increase for a more suitable fit

```
> b <- b %>%
+   mutate(
+     k_default = ksmooth(x = age, y = mi, n.points = n())$y,
+     k_bw5 = ksmooth(x = age, y = mi, n.points = n(), bandwidth = 5)$y,
+     k_bw10 = ksmooth(x = age, y = mi, n.points = n(), bandwidth = 10)$y
+   )
> b
# A tibble: 95 x 8
   sample    age migrants population     mi k_default  k_bw5 k_bw10
   <chr>   <dbl>    <dbl>      <dbl>  <dbl>     <dbl>  <dbl>  <dbl>
 1 BRA2000     6  355723.   3311728. 0.107     0.107  0.104  0.100
 2 BRA2000     7  343852.   3307567. 0.104     0.104  0.102  0.0990
 3 BRA2000     8  327166.   3258046. 0.100     0.100  0.101  0.0983
 4 BRA2000     9  314905.   3272305. 0.0962    0.0962 0.0986 0.0978
 5 BRA2000    10  324066.   3345583. 0.0969    0.0969 0.0964 0.0976
 6 BRA2000    11  329525.   3451739. 0.0955    0.0955 0.0949 0.0978
 7 BRA2000    12  327113.   3518160. 0.0930    0.0930 0.0944 0.0975
 8 BRA2000    13  323180.   3473133. 0.0931    0.0931 0.0942 0.0982
 9 BRA2000    14  334783.   3566239. 0.0939    0.0939 0.0950 0.100
10 BRA2000    15  337297.   3528845. 0.0956    0.0956 0.0973 0.103
# ... with 85 more rows
```

Kernal smoothing

```
> ggplot(data = b, mapping = aes(x = age, y = mi)) +
+   geom_point(alpha = 0.5) +
+   geom_line(mapping = aes(y = k_default), col = "darkgrey") +
+   geom_line(mapping = aes(y = k_bw5), col = "orange") +
+   geom_line(mapping = aes(y = k_bw10), col = "red")
```



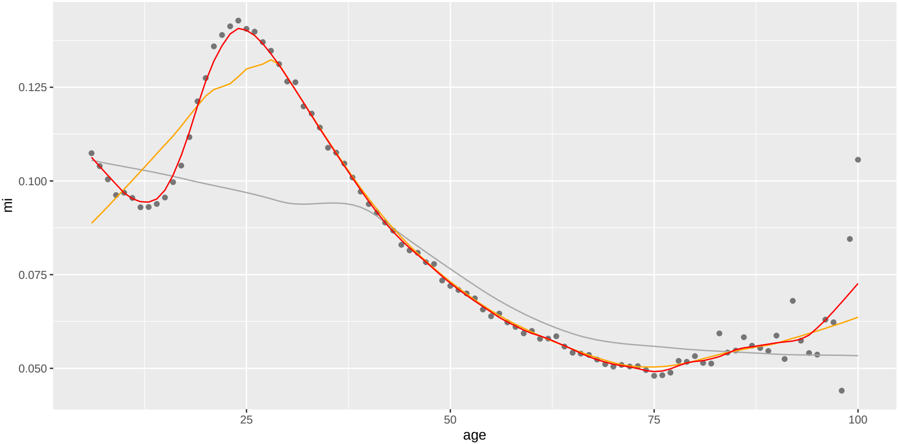
Loess smoothing

- The `loess.smooth` function is also unlikely to smooth a migration age schedule as the default span parameter is too small
  - Adjust the smoothing parameter using spar (between 0 and 1), default is 2/3
  - Use `evaluation` to set the number of predicted values

```
> b <- b %>%
+   mutate(
+     lo_default = loess.smooth(x = age, y = mi, evaluation = n())$y,
+     lo_sp2 = loess.smooth(x = age, y = mi, evaluation = n(), span = 0.2)$y,
+     lo_sp1 = loess.smooth(x = age, y = mi, evaluation = n(), span = 0.1)$y,
+   )
```

Loess smoothing

```
> ggplot(data = b,
+        mapping = aes(x = age, y = mi)) +
+   geom_point(alpha = 0.5) +
+   geom_line(mapping = aes(y = lo_default), col = "darkgrey") +
+   geom_line(mapping = aes(y = lo_sp2), col = "orange") +
+   geom_line(mapping = aes(y = lo_sp1), col = "red")
```



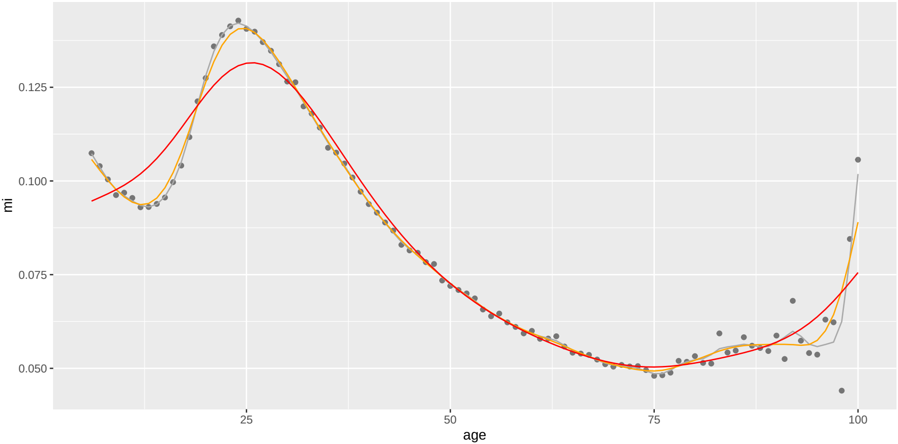
Cubic spline smoothing

- The `smooth.spline` function might have a nice smooth fit to migration age schedule
  - Adjust the smoothing parameter using `spar` (between 0 and 1)
  - Use `n` to set the number of predicted values

```
> b <- b %>%
+   mutate(
+     s_default = smooth.spline(x = age, y = mi, n = n())$y,
+     s_sp6 = smooth.spline(x = age, y = mi, n = n(), spar = 0.6)$y,
+     s_sp8 = smooth.spline(x = age, y = mi, n = n(), spar = 0.8)$y)
```

Cubic spline smoothing

```
> ggplot(data = b,
+        mapping = aes(x = age, y = mi)) +
+   geom_point(alpha = 0.5) +
+   geom_line(mapping = aes(y = s_default), col = "darkgrey") +
+   geom_line(mapping = aes(y = s_sp6), col = "orange") +
+   geom_line(mapping = aes(y = s_sp8), col = "red")
```



Graduating

- If you require single year migration age data, but only have data by age groups, then graduating methods can be used to estimate migration for each age that sum to the reported age group totals.
- There a multiple graduating methods available in the `graduate()` function in the DemoTools package
  - Built for interpolating population totals, but also suitable for migration flows
  - See the guide for more detail on different methods
- Requires users to provide `Value` and minimum `Age`.
- Can also specify the maximum value of final open age group, if exists, for certain methods such as `pclm`.

Graduating

- DemoTools is not on CRAN. Difficult to install.

```
> # install devtools from CRAN
> # install.packages("devtools")
> library(devtools)
>
> install_github("timriffe/DemoTools")
```

Graduating

- Using the out-migration to Italian islands area in 1970

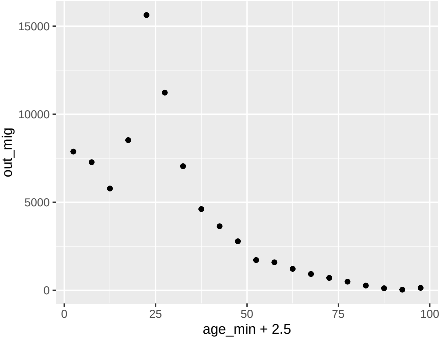
```
> library(DemoTools)
> head(i)
# A tibble: 6 x 8
# Groups:   age_grp [6]
  age_grp age_min age_max region  out_mig in_mig  turn   net
  <fct>     <int>   <int> <chr>     <dbl>  <dbl> <dbl> <dbl>
1 0-4           0       4 Islands    7876   4532 12408 -3344
2 5-9           5       9 Islands    7271   3592 10863 -3679
3 10-14        10      14 Islands    5779   2228  8007 -3551
4 15-19        15      19 Islands    8526   3064 11590 -5462
5 20-24        20      24 Islands   15629   6861 22490 -8768
6 25-29        25      29 Islands   11224   5891 17115 -5333
>
> mx <- graduate(Value = i$out_mig, Age = i$age_min,
+                method = "pclm", OAG =  TRUE, OAnew = 100)
> mx
          0           1           2           3           4           5
1540.822616 1563.081967 1582.487050 1594.810184 1594.859870 1576.283303
          6           7           8           9          10          11
1534.233098 1470.025351 1388.679355 1301.771896 1221.108012 1158.023181
         12          13          14          15          16          17
1121.942083 1119.554144 1158.435391 1247.103598 1398.284195 1623.913611
         18          19          20          21          22          23
```

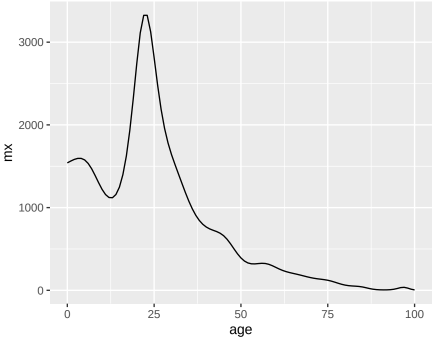
Graduating

```
> # check for close match between graduate values and out_mig
> # 0-4
> sum(mx[1:5])
[1] 7876.062
> # 5-9
> sum(mx[6:10])
[1] 7270.993
>
> select(i, age_grp, out_mig)
# A tibble: 20 x 2
# Groups:   age_grp [20]
   age_grp out_mig
   <fct>     <dbl>
 1 0-4        7876
 2 5-9        7271
 3 10-14      5779
 4 15-19      8526
 5 20-24     15629
 6 25-29     11224
 7 30-34      7046
 8 35-39      4612
 9 40-44      3634
10 45-49      2783
11 50-54      1716
12 55-59      1587
```

Graduating

```
> # different scales in y-axis
> ggplot(data = i,
+        mapping = aes(x = age_min + 2.5, y = out_mig)) +
+   geom_point()
>
> tibble(age = 0:100, mx = mx) %>%
+   ggplot(mapping = aes(x = age, y = mx)) +
+   geom_line()
```





Exercise 5 (ex5.R)

```
# 0.  a) Load the KOSTAT2022.Rproj file.
#     Run the getwd() below. It should print the directory where the
#     KOSTAT2022.Rproj file is located.
getwd()
#     b) Load the packages used in this exercise
library(tidyverse)
library(migest)
library(DemoTools)
##
##
##
# 1. Run the code below to read in the population age structure data for flows
#    from Florida to New York based on the 2015 American Community Survey
flny <- read_csv("./data/florida_new_york_acs_2015.csv")
flny
# 2. Run the code below to plot the age schedule for migration from New York to
#    Florida. Note, the uneven spread of the age groups
ggplot(data = x, mapping = aes(x = AGE_label, y = mig_in, group = 1)) +
  geom_point() +
  geom_line() +
  theme(axis.text = element_text(angle = 45, hjust = 1))
# 3. Estimate the age schedule based on single years up to 100, based on a
#    graduation of the migration data in flny
mx <- #####(Value = flny$#####, Age = #####$age_min,
               method = "pclm", OAG =  TRUE, OAnew = #####)
```

References I

Bell, Martin, Marcus Blake, Paul Boyle, O. Duke-Williams, Philip H. Rees, John Stillwell, and Graeme John Hugo. 2002. "Cross-national comparison of internal migration: issues and measures." *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 165 (3): 435–64. https://doi.org/10.1111/1467-985X.00247.

Bell, Martin, and Salut Muhidin. 2009. "Cross-National Comparisons of Internal Migration." Human Development Reports. United Nations Development Programme.

Bernard, Aude, Martin Bell, and Elin Charles-Edwards. 2014. "Improved measures for the cross-national comparison of age profiles of internal migration." *Population Studies* 68 (2): 179–95. https://doi.org/10.1080/00324728.2014.890243.

Rogers, Andrei. 1975. *Introduction to Multiregional Mathematical Demography*. New York, New York, USA: Wiley.

Rogers, Andrei, and Luis J. Castro. 1981. "Model Migration Schedules." RR-81-30. Vol. 81. Laxenburg, Austria: International Institute for Applied Systems Analysis. http://webarchive.iiasa.ac.at/Admin/PUB/Documents/RR-81-030.pdf.

Rogers, Andrei, and Jani S Little. 1994. "An International Journal of Parameterizing age patterns of demographic rates with the multiexponential model schedule." *Mathematical Population Studies* 4 (3): 175–95. https://doi.org/10.1080/08898489409525372.

Rogers, Andrei, and James Raymer. 1999. "Estimating the regional migration patterns of the foreign-born population in the United States: 1950-1990." *Mathematical Population Studies* 7 (3): 181–216, 307. https://doi.org/10.1080/08898489909525457.

Rogers, Andrei, James Raymer, and Jani Little. 2010. *The Indirect Estimation of Migration*. Vol. 26. The Springer Series on Demographic Methods and Population Analysis. Dordrecht: Springer Netherlands. https://doi.org/10.1007/978-90-481-8915-1.

Rogers, Andrei, and John Watkins. 1987. "General Versus Elderly Interstate Migration and Population Redistribution in the United States." *Research on Aging* 9 (4): 483–529. https://doi.org/10.1177/0164027587094002.

References II

United Nations Department of Economic and Social Affairs Population Division. 1992. *Preparing Migration Data for Subnational Population Projections*. http://www.un.org/esa/population/techcoop/IntMig/migdata_popproj/migdata_popproj.html.

Wilson, Tom. 2010. "Model migration schedules incorporating student migration peaks." *Demographic Research* 23 (8): 191–222. https://doi.org/10.4054/DemRes.2010.23.8.