# Life as a Pachinko machine: modelling lifetime income using HILDA data

*Hugh Parsonage and William Young*

*2018-07-03*

This package outlines the formation of lifetime incomes as used in the retirement income models.

## Outline

The distribution of lifetime income of individuals is calculated by:

1. Selecting a relevant subset of the longitudinal sample of the HILDA survey
2. For each individual in this subset, calculate their average income between 2005 and 2009 and their average income between 2010 and 2014.
3. Determine the income percentile of each individual in each period.
4. Construct the observed transition matrix of income for each age transition from 25-29 to 30-34 and from 55-59 to 60-64.
5. Smooth these matrix using a tensor product smooth. It may be more appropriate to use a different 2D kernel smoother, but this is the one that was chosen.
6. Apply these matrices across each age to end up with a `machine_widget`

```
# Set working directory to source file location
# This script:
#   - writes several transition tables to disk (through the source() line below)
#   - creates objects
#     * machine_widget (every row is a simulation's instance of a person;
#       the columns marked with .1 are the path that person took in that simulation)
#     * lifetime_pachinko that outputs for every Percentile the average lifetime_AWOTE across all simul
```

These are the global variables. That is, only consider HILDA individuals aged between 30 and 70 in the sample and perform 10,000 simulations to obtain our percentiles.

```
# 1000 is pretty good (100,000 individuals)
N_SIMULATIONS <- 10000
OLDEST.AGE <- 70L
YOUNGEST.AGE <- 30L
# tbl_dt [100 x 2]
#    Percentile lifetime_AWOTE
#         <int>          <dbl>
# 1           1       13.50444
# 2           2       14.17248
# 3           3       14.56721
# 4           4       15.00996
# 5           5       15.38401
# 6           6       15.93115
# 7           7       16.49369
# 8           8       16.91389
# 9           9       17.61472
# 10         10       17.99585
# # ... with 90 more rows
```

```
N_SIMULATIONS <- as.integer(N_SIMULATIONS)
stopifnot(N_SIMULATIONS >= 1)
randz <- runif(100 * N_SIMULATIONS)

library(hutils)
library(readr)
library(ggplot2)
library(scales)
#>
#> Attaching package: 'scales'
#> The following object is masked from 'package:readr':
#>
#>     col_factor
library(grattanCharts)
#>
#> Attaching package: 'grattanCharts'
#> The following object is masked from 'package:datasets':
#>
#>     Orange
library(viridis)
#> Loading required package: viridisLite
#>
#> Attaching package: 'viridis'
#> The following object is masked from 'package:scales':
#>
#>     viridis_pal
library(magrittr)
library(mgcv)
#> Loading required package: nlme
#> This is mgcv 1.8-23. For overview type 'help("mgcv-package")'.


library(data.table)
if (requireNamespace("hildaData", quietly = TRUE)) {
  library(hildaData)
} else {
  stop("package:hildaData is a local package, containing the raw Combined_* .dta files as data.tables.\
       "Either obtain this package from Hugh Parsonage or run", "\n",
       "\t",
       "Combined_n140c <- as.data.table(read.dta('path/to/your/Wave14/Combinedfile.dta'));", "\n",
       "\t",
       "Combined_m130c <- as.data.table(read.dta('path/to/your/Wave13/Combinedfile.dta'));", "\n",
       "\t", "...", "\n",
       "etc.")
}
library(hildaExtra)
library(grattan)
#>
#> Attaching package: 'grattan'
#> The following object is masked from 'package:CRIMpp':
#>
#>     IncomeTax
```

```
awote_by_year <-
  data.table(Year = 2014:2001,
             AWOTE = c(1477,
                       1437,
                       1396,
                       1330.1,
                       1275.2,
                       1226.8,
                       1158,
                       1098.6,
                       1044.1,
                       1011.8,
                       965,
                       929.1,
                       882.1,
                       842.6)) %>%
  .[, AWOTE_annual := 52.5 * AWOTE]

get_hilda_xwaveid_age_income <- function(year){
  wave <- letters[year - 2000]
  if (exists("Combined_n150c", where = "package:hildaData")) {
    NAME <- paste0("Combined_", wave, "150c")
  } else {
    if (year == 2014){
      NAME <- "Combined_n140c"
    } else {
      NAME <- paste0("Combined_", wave, "130c")
    }
  }

  get(NAME) %>%
    as.data.table %>%
    .[,.SD,
      .SDcols = c("xwaveid",
                  paste0(wave, "tifefp"), # income
                  paste0(wave, "hgage"),  # age
                  paste0(wave, "esbrd"),  # employment-status
                  paste0(wave, "wscmg")    # earnings from wage
      )] %>%
    setnames(paste0(wave, "tifefp"), "Income") %>%
    setnames(paste0(wave, "hgage"), "Age") %>%
    setnames(paste0(wave, "esbrd"), "Employment_status") %>%
    setnames(paste0(wave, "wscmg"), "Weekly_salary_wage_from_main_job") %>%
    .[, lapply(.SD, make_negatives_NA)] %>%
    .[, Year := get("year", mode = "numeric")]
}

if (exists("Combined_n150c", where = "package:hildaData")) {
  # alias
  Combined_n140c <- as.data.table(Combined_n150c)
}
```

```
xwaveid_by_lnwte <-
  Combined_n140c %>%
```

```r
  as.data.table %>%
  .[, .(xwaveid, WEIGHT = nlnwte)] %>%
  .[WEIGHT > 0] %>%
  setkey(xwaveid) %>%
  unique(by = "xwaveid")
```

```r
income_age_by_xwaveid_year <-
  lapply(2001:2014, get_hilda_xwaveid_age_income) %>%
  rbindlist(use.names = TRUE) %>%
  setkey(xwaveid) %>%
  .[xwaveid_by_lnwte, on = "xwaveid"] %>%
  .[, Income_percentile := weighted_ntile(Income, WEIGHT, n = 100), keyby = "Year"] %>%
  .[, Income_next_year := shift(Income, type = "lead"), by = "xwaveid"] %>%
  .[, Income_percentile_next_year := shift(Income_percentile, type = "lead"),
    by = "xwaveid"] %>%
  .[, DOB := Year - Age] %>%
  .[]
```

```r
prob_year_on_year <-
  income_age_by_xwaveid_year %>%
  .[between(DOB, 2014L - OLDEST.AGE, 2010L - YOUNGEST.AGE)] %>%
  .[Weekly_salary_wage_from_main_job > 38L] %>%
  .[, Age_group := age_grouper(Age, interval = 10, min_age = 25, max_age = 75)] %>%
  .[, .(n_persons = sum(WEIGHT)),
    keyby = .(Age_group, Income_percentile, Income_percentile_next_year)] %>%
  .[, prob := n_persons / sum(n_persons), keyby = .(Age_group, Income_percentile)] %>%
  .[]
```

```r
Age_group_2005_by_xwaveid <-
  income_age_by_xwaveid_year %>%
  .[Year == 2005L] %>%
  .[, .(Age_group_2005 = age_grouper(Age, interval = 5, min_age = 25, max_age = 75)),
    keyby = "xwaveid"]
```

```r
## How much do we lose by restricting to only those present in at
## least one year in 2005-2009 and 2010-2014.

ALL <- function(...) {
  Reduce(`&&`, list(...))
}
```

```r
InScope_by_xwaveid <-
  income_age_by_xwaveid_year %>%
  .[Weekly_salary_wage_from_main_job > 100L] %>%
  .[Employment_status == "[1] Employed"] %>%
  .[between(DOB, 2014L - OLDEST.AGE, 2010L - YOUNGEST.AGE)] %>%
  .[Year >= 2005L] %>%
  .[,
    .(in_scope = ALL(any(Year <= 2009L),
                     any(Year > 2009L),
                     .N >= 3L)),
    keyby = "xwaveid"]
```

```r
# Must be at least 60% (of workers at least 30)
stopifnot(mean(InScope_by_xwaveid$in_scope) > 0.6)
```

```r
income_AWOTE_by_age_percentile <-
  income_age_by_xwaveid_year %>%
  InScope_by_xwaveid[., on = "xwaveid", nomatch=0L] %>%
  .[(in_scope)] %>%
  .[Year >= 2005L] %>%
  awote_by_year[., on = "Year", nomatch=0L] %>%
  .[, Income_per_AWOTE_annual := Income / AWOTE_annual] %>%
  .[,
    .(AWOTE_annual_average = mean(Income_per_AWOTE_annual)),
    keyby = .(xwaveid, First_year_range = Year <= 2009)] %>%
  setkey(xwaveid) %>%
  xwaveid_by_lnwte[., on = "xwaveid", nomatch=0L] %>%
  Age_group_2005_by_xwaveid[., on = "xwaveid", nomatch=0L] %>%
  .[, AWOTE_annual_percentile := weighted_ntile(AWOTE_annual_average,
                                                weights = WEIGHT,
                                                n = 100),
    keyby = .(Age_group_2005, First_year_range)] %>%
  .[, Year_range := if_else(First_year_range, "Decile_ante_2009", "Decile_post_2009"),
    keyby = .(Age_group_2005, First_year_range)] %>%
  .[, .(xwaveid,
        Age_group_2005,
        Year_range,
        AWOTE_annual_percentile,
        AWOTE_annual_average)] %>%
  setkey(xwaveid) %>%
  xwaveid_by_lnwte[., on = "xwaveid", nomatch=0L] %>%
  .[, .(n_persons = sum(WEIGHT),
        AWOTE_average = weighted.mean(AWOTE_annual_average, WEIGHT)),
    keyby = .(Age_group_2005, Year_range, AWOTE_annual_percentile)]

# income_AWOTE_by_age_percentile %>%
#   write_csv("income-AWOTE-by-age-percentile.csv")
```

```r
prob_5year_on_year <-
  income_age_by_xwaveid_year %>%
  InScope_by_xwaveid[., on = "xwaveid", nomatch=0L] %>%
  awote_by_year[., on = "Year", nomatch=0L] %>%
  .[Year >= 2005L] %>%
  .[(in_scope)] %>%
  .[, Income_per_AWOTE_annual := Income / AWOTE_annual] %>%
  .[,
    .(AWOTE_annual_average = mean(Income_per_AWOTE_annual)),
    keyby = .(xwaveid, First_year_range = Year <= 2009)] %>%
  setkey(xwaveid) %>%
  xwaveid_by_lnwte[., on = "xwaveid", nomatch=0L] %>%
  Age_group_2005_by_xwaveid[., on = "xwaveid", nomatch=0L] %>%
  .[,
    AWOTE_annual_decile := weighted_ntile(AWOTE_annual_average,
                                          weights = WEIGHT,
                                          n = 10),
    keyby = .(Age_group_2005, First_year_range)] %>%
  .[, Year_range := if_else(First_year_range,
                            "Decile_ante_2009",
                            "Decile_post_2009")] %>%
  .[, .(xwaveid, Age_group_2005, Year_range, AWOTE_annual_decile)] %>%
  dcast.data.table(xwaveid + Age_group_2005 ~  Year_range,
                   value.var = "AWOTE_annual_decile") %>%
  setkey(xwaveid) %>%
  xwaveid_by_lnwte[., on = "xwaveid", nomatch=0L] %>%
  .[, .(n_persons = sum(WEIGHT)),
    keyby = .(Age_group_2005, Decile_ante_2009, Decile_post_2009)] %>%
  .[, prob := n_persons / sum(n_persons),
    keyby = .(Age_group_2005, Decile_ante_2009)] %>%
  .[]
```

```
prob_5year_on_year_percentile <-
  income_age_by_xwaveid_year %>%
  InScope_by_xwaveid[., on = "xwaveid", nomatch=0L] %>%
  .[(in_scope)] %>%
  .[Year >= 2005L] %>%
  awote_by_year[., on = "Year", nomatch=0L] %>%
  .[, Income_per_AWOTE_annual := Income / AWOTE_annual] %>%
  .[, .(AWOTE_annual_average = mean(Income_per_AWOTE_annual)),
    keyby = .(xwaveid, First_year_range = Year <= 2009)] %>%
  setkey(xwaveid) %>%
  merge(xwaveid_by_lnwte) %>%
  merge(Age_group_2005_by_xwaveid) %>%
  .[, Year_range := if_else(First_year_range, "Percentile_ante_2009", "Percentile_post_2009")] %>%
  .[, AWOTE_annual_percentile := weighted_ntile(AWOTE_annual_average,
                                                weights = WEIGHT,
                                                n = 100),
    keyby = .(Age_group_2005, Year_range)] %>%
  .[, .(xwaveid, Age_group_2005, Year_range, AWOTE_annual_percentile)] %>%
  dcast.data.table(xwaveid + Age_group_2005 ~  Year_range,
                   value.var = "AWOTE_annual_percentile") %>%
  setkey(xwaveid) %>%
  xwaveid_by_lnwte[., on = "xwaveid", nomatch=0L] %>%
  .[, .(n_persons = sum(WEIGHT)),
    keyby = .(Age_group_2005,
              Percentile_ante_2009 = factor(Percentile_ante_2009),
              Percentile_post_2009 = factor(Percentile_post_2009))] %>%
  .[, prob := n_persons / sum(n_persons),
    keyby = .(Age_group_2005, Percentile_ante_2009)] %>%
  .[]
```

```
provide.dir("age-transition-matrices")
```

```
nan_to_zero <- function(x) {
  if (is.numeric(x)){
    x[is.nan(x)] <- 0
  }
  x
}
```

```
get_transition_table <- function(age_range, filename){
  if (missing(age_range)) {
```

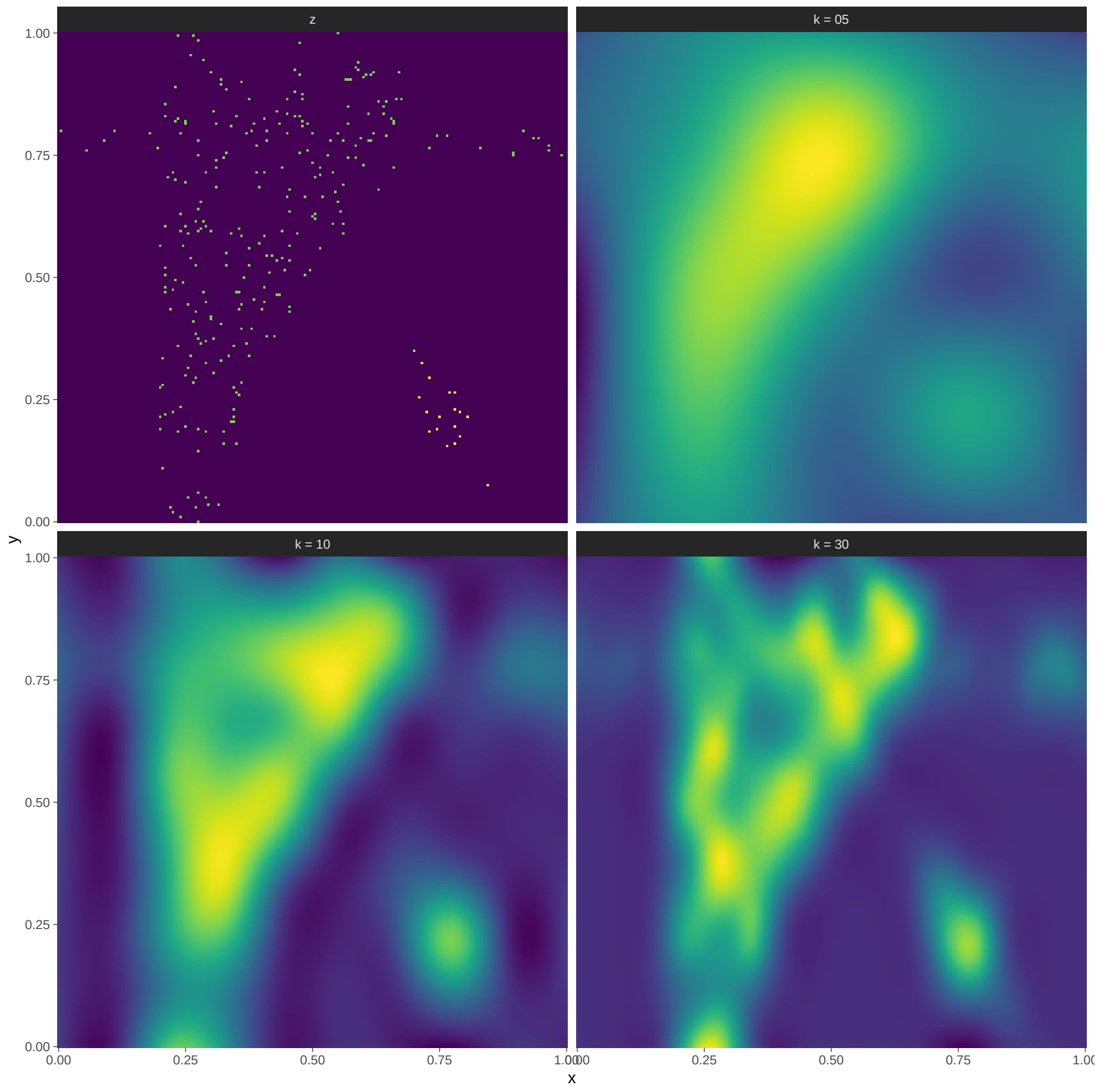
```r
    age.range <- gsub("^.*([0-9]{2}.[0-9]{2}).transition.matrix\\.csv$", "\\1", filename)

    fread(filename, header = TRUE) %>%
      melt.data.table(id.vars = "Percentile_pre_2009",
                      variable.name = "Percentile_post_2009",
                      value.name = "prob") %>%
      arrange(Percentile_pre_2009) %>%
      group_by(Percentile_pre_2009) %>%
      mutate(cumprob = order_by(Percentile_post_2009, cumsum(prob))) %>%
      setkey(Percentile_pre_2009, cumprob) %>%
      mutate(age_range = age.range)
  } else {
    prob_5year_on_year_percentile[Age_group_2005 == age_range] %>%
      .[, Percentile_ante_2009 := factor(Percentile_ante_2009, levels = 1:100)] %>%
      .[, Percentile_post_2009 := factor(Percentile_post_2009, levels = 1:100)] %>%
      # dcast + melt = expand.grid (so that zero probabilities )
      dcast.data.table(Percentile_ante_2009 ~ Percentile_post_2009,
                       fun = mean,
                       drop = FALSE,
                       fill = NaN,
                       value.var = "prob") %>%
      .[, lapply(.SD, nan_to_zero)] %>%
      melt.data.table(id.vars = "Percentile_ante_2009",
                      variable.name = "Percentile_post_2009",
                      value.name = "prob") %>%
      .[, Percentile_ante_2009 := as.integer(Percentile_ante_2009)] %>%
      .[, Percentile_post_2009 := as.integer(Percentile_post_2009)] %>%
      setkey(Percentile_ante_2009) %>%
      .[, cumprob := cumsum(prob), keyby = "Percentile_ante_2009"] %>%
      setkey(Percentile_ante_2009, cumprob) %>%
      .[]
  }
}
```

```r
transition_tables_all <-
  lapply(unique(income_AWOTE_by_age_percentile$Age), get_transition_table) %>%
  rbindlist(use.names = TRUE, fill = TRUE)
```

## Tensor product smooth

A tensor product smooth was used to convert the spikes in the sparse transition matrices to a smoothed version, to reflect the underlying distribution. Intuitively, each nonzero value in observed transition matrix is a spike on a square board. The tensor product smooth is the surface that would appear if a somewhat segmented sheet were loosely placed over the spikes on the board. Higher values of $k$ mean the sheet is 'tighter'.

An example of a the tensor product smooth applies to some dummy data is in the following figure. For the model, we used $k = 10$.



```
smooth_matrix <- function(age_range) {
  stopifnot(length(age_range) == 1L)

  gam.1 <-
    prob_5year_on_year_percentile %>%
    .[Age_group_2005 == age_range]
  # k = 10 for less smooth data
  gam(prob ~ te(Percentile_ante_2009, Percentile_post_2009), k = c(10, 10), data = .)
  out <- matrix(pmax(fitted(gam.1), 0), ncol = 100)
  out / rowSums(out)
}
```

```
smooth_table <- function(age_range) {
  gam.1 <-
```

```r
    get_transition_table(age_range) %>%
    # weight2rows("n_persons") %>%
    # k = 10 for less smooth data, somewhat computationally intensive than lower k
    # This is a tensor product smooth.
    gam(prob ~ te(Percentile_ante_2009, Percentile_post_2009, k = c(10, 10)),
        data = .)

  CJ(Percentile_ante_2009 = 1:100,
     Percentile_post_2009 = 1:100) %>%
    # gam will predict some values as (very slightly) negative.
    # We need to normalize them anyhow, so just force nonnegative.
    .[, prob := pmax(predict(gam.1, newdata = .), 0)] %>%
    .[, prob := prob / sum(prob), by = Percentile_ante_2009] %>%
    .[, cumprob := cumsum(shift(prob, n = 1, type =  "lag", fill = 0)),
      keyby = Percentile_ante_2009] %>%
    .[, .(Percentile_ante = Percentile_ante_2009,
          Percentile_post = Percentile_post_2009,
          cumprob)] %>%
    setkey(Percentile_ante, cumprob)
}
```

```r
input <-
  CJ(Percentile_ante = 1:100,
     simulation = 1:N_SIMULATIONS) %>%
  .[, cumprob := randz]

setkey(input, Percentile_ante, cumprob)

copy_column <- function(.data, new_name, column_copied){
  .data2 <- copy(.data)
  .data2[[new_name]] <- .data[[column_copied]]
  as.data.table(.data2)
}

age_the_simulation <- function(input.table, five_year_group = 2){
  Ages <- c("25-29", "30-34", "35-39", "40-44", "45-49", "50-54", "55-59", "60-64")
  age_range_fm <- Ages[five_year_group - 1]
  agerangefm <- sub("-", "", age_range_fm, fixed = TRUE)
  age_range_to <- Ages[five_year_group]
  agerangeto <- sub("-", "", age_range_to, fixed = TRUE)
  smooth_table(age_range_to)[input.table, roll = Inf] %>%
    setnames("Percentile_post", paste0("Percentile_", agerangeto)) %>%
    setnames("Percentile_ante", paste0("Percentile_", agerangefm)) %>%
    copy_column("Percentile_ante", paste0("Percentile_", agerangefm)) %>%
    # re roll the die
    .[, cumprob := runif(.N)] %>%
    setkey(Percentile_ante, cumprob)
}
```

```r
machine_widget <-
  input %>%
  setkey(Percentile_ante, cumprob) %>%
  age_the_simulation(2) %>%
  age_the_simulation(3) %>%
  age_the_simulation(4) %>%
  age_the_simulation(5) %>%
  age_the_simulation(6) %>%
  age_the_simulation(7) %>%
  age_the_simulation(8) %>%
  .[]
```

```r
awote_by_age <-
  income_AWOTE_by_age_percentile[Year_range == "Decile_post_2009",
                                 .(Age = Age_group_2005,
                                   Percentile = AWOTE_annual_percentile,
                                   AWOTE_average)] %>%
  setkey(Age, Percentile)  %>%
  # Ensure all percentiles are available
  # (Otherwise the benchmark might be non-monotonic.)
  {
    dot <- .
    gridExpanded <- CJ(Age = unique(dot$Age),
                       Percentile = 1:100)

    setkey(gridExpanded, Age, Percentile)

    dot[gridExpanded, on = c("Age", "Percentile")] %>%
      .[, AWOTE_average := if_else(is.na(AWOTE_average) & Percentile %in% c(1, 100),
                                   if_else(Percentile == 100,
                                           max(AWOTE_average, na.rm = TRUE),
                                           0),
                                   AWOTE_average), keyby = "Age"] %>%
      .[, AWOTE_average := zoo::na.approx(AWOTE_average, na.rm = FALSE), keyby = "Age"]
  }

stopifnot(nrow(awote_by_age) == 800L)
```

```r
lifetime_income_benchmark <-
  awote_by_age %>%
  setkey(Age, Percentile) %>%
  .[, .(lifetime_AWOTE = sum(AWOTE_average)), keyby = "Percentile"] %>%
  .[, lifetime_AWOTE := 5 * lifetime_AWOTE]
```

```r
lifetime_pachinko_density <-
  machine_widget %>%
  .[, .SD, .SDcols = c("simulation", "Percentile_2529", grep(".1", names(.), value = TRUE), "Percentile_
  setnames(old = grep(".1$", names(.), value = TRUE),
           new = gsub(".1$", "", names(.)[grep(".1$", names(.), value = FALSE)])) %>%
  .[, id := .I] %>%
  melt.data.table(id.vars = c("id", "simulation"), variable.name = "Age", value.name = "Percentile") %>
  .[, Age := gsub("Percentile_(..)(..)", "\\1-\\2", Age)] %>%
  setkeyv(c("Age", "Percentile")) %>%
  merge(awote_by_age)
```

```r
lifetime_income_pachinko <-
  lifetime_pachinko_density[,
                            .(lifetime_AWOTE = sum(AWOTE_average) * 5,
                              Percentile = first(Percentile)),
                            keyby = "id"] %>%
  .[, .(lifetime_AWOTE = mean(lifetime_AWOTE)), keyby = "Percentile"]
```

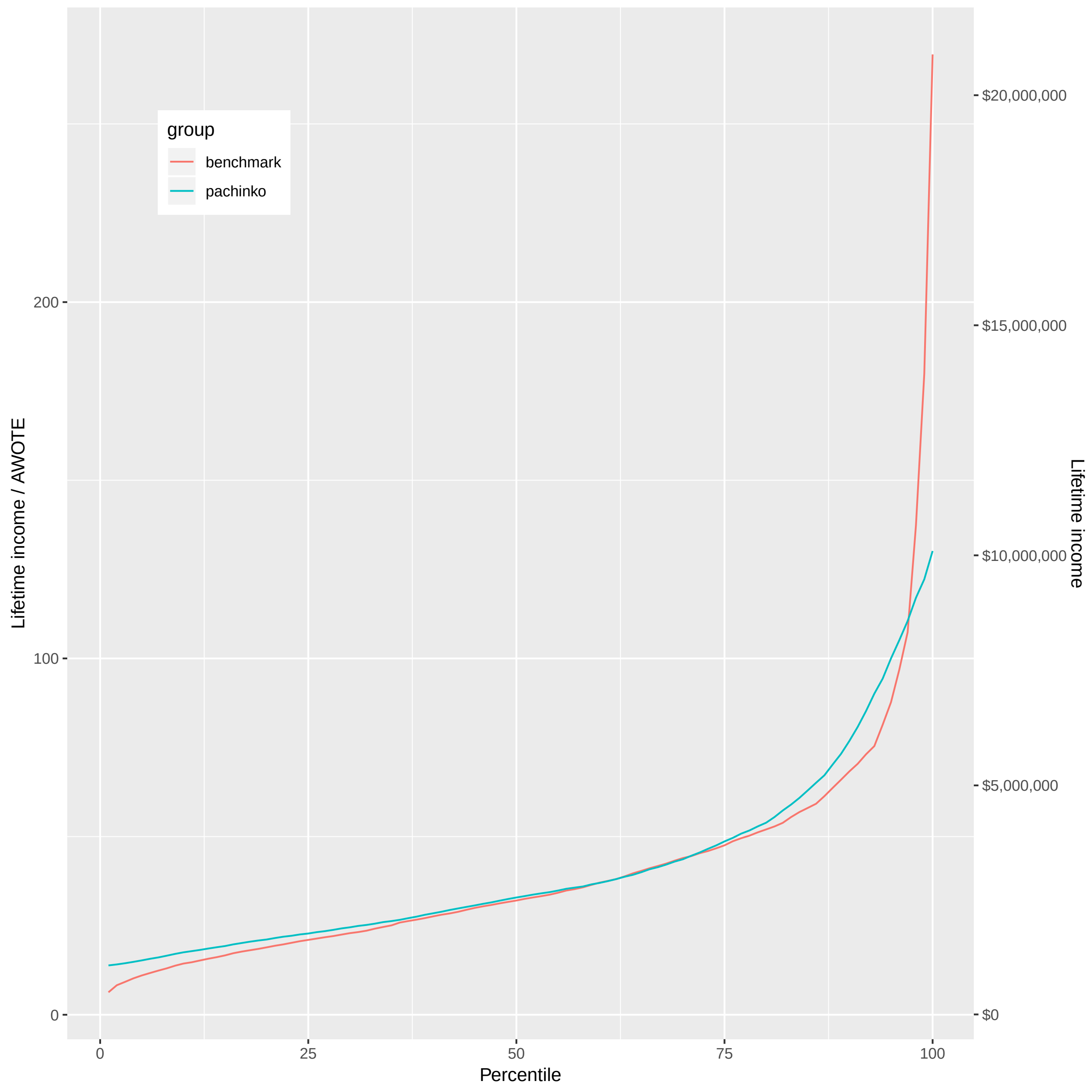
## Result

The average lifetime income is comparable to the average income if we assumed individuals remained in the same percentile throughout their lives.

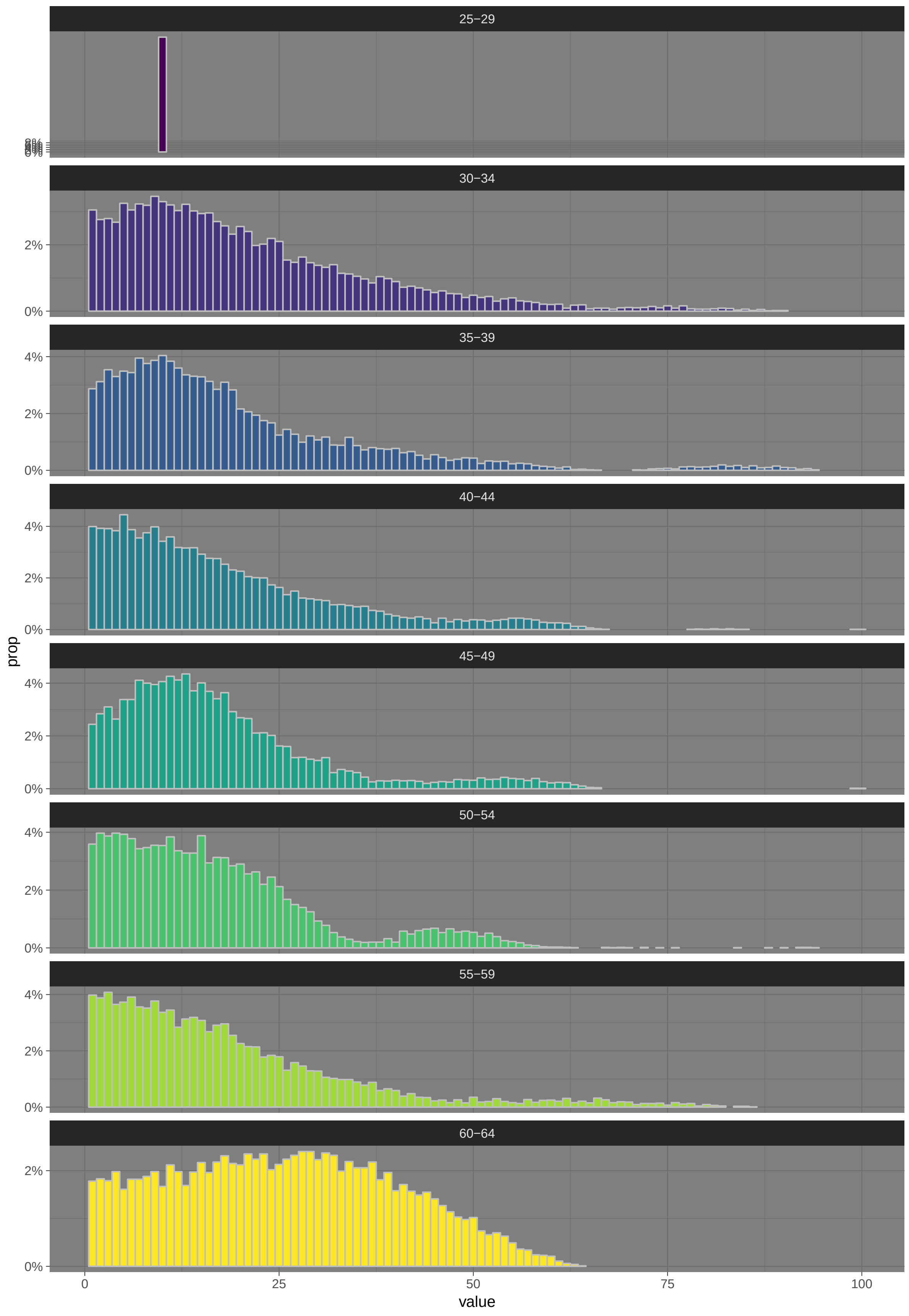
```
merge(lifetime_income_benchmark, lifetime_income_pachinko, by = "Percentile") %$%
  t.test(lifetime_AWOTE.x, lifetime_AWOTE.y)
#>
#>  Welch Two Sample t-test
#>
#> data:  lifetime_AWOTE.x and lifetime_AWOTE.y
#> t = -0.071412, df = 181.93, p-value = 0.9431
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#>  -8.926094  8.302541
#> sample estimates:
#> mean of x mean of y
#>  40.55624  40.86801
```

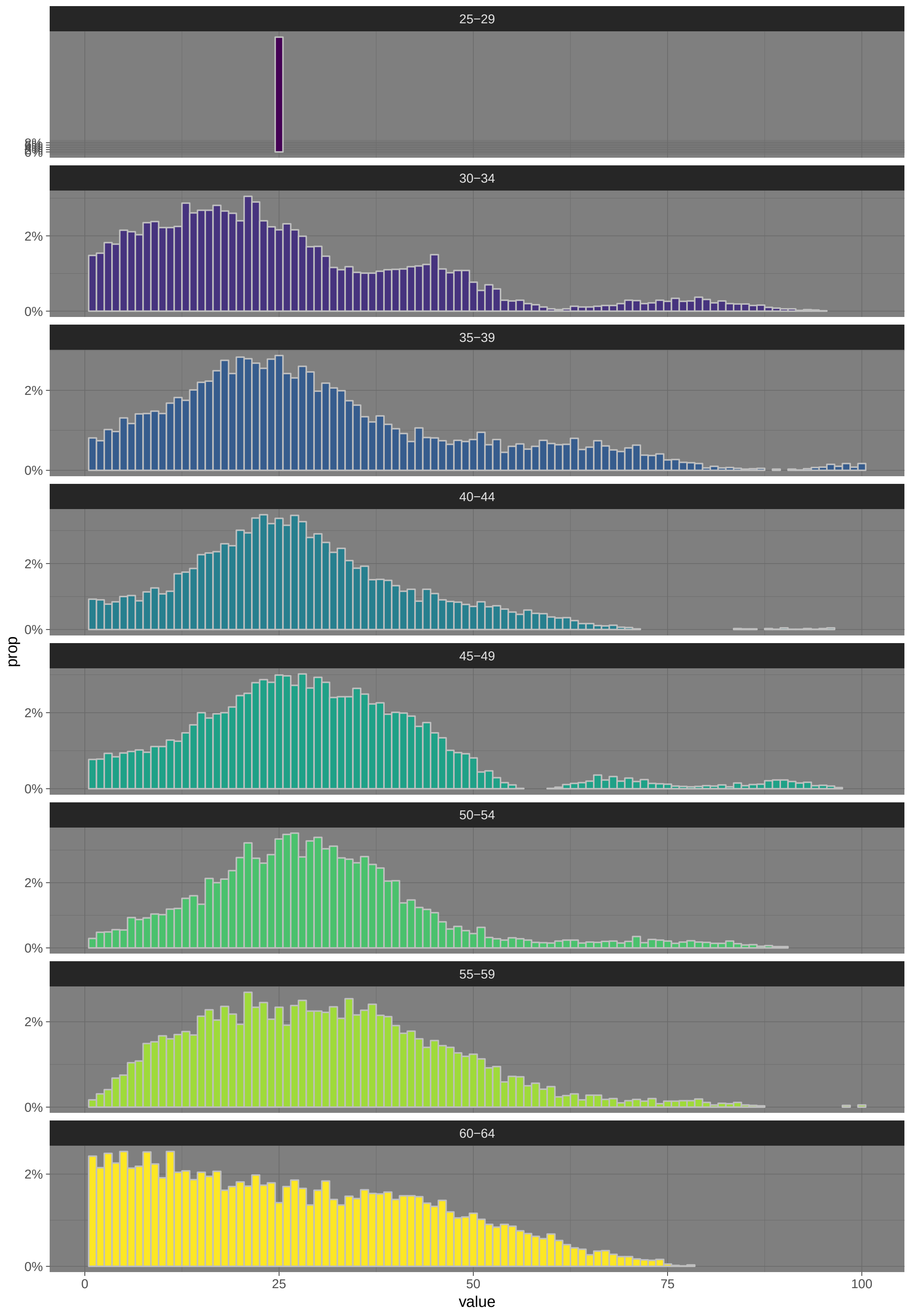
The difference is most pronounced at the tails of the distribution:

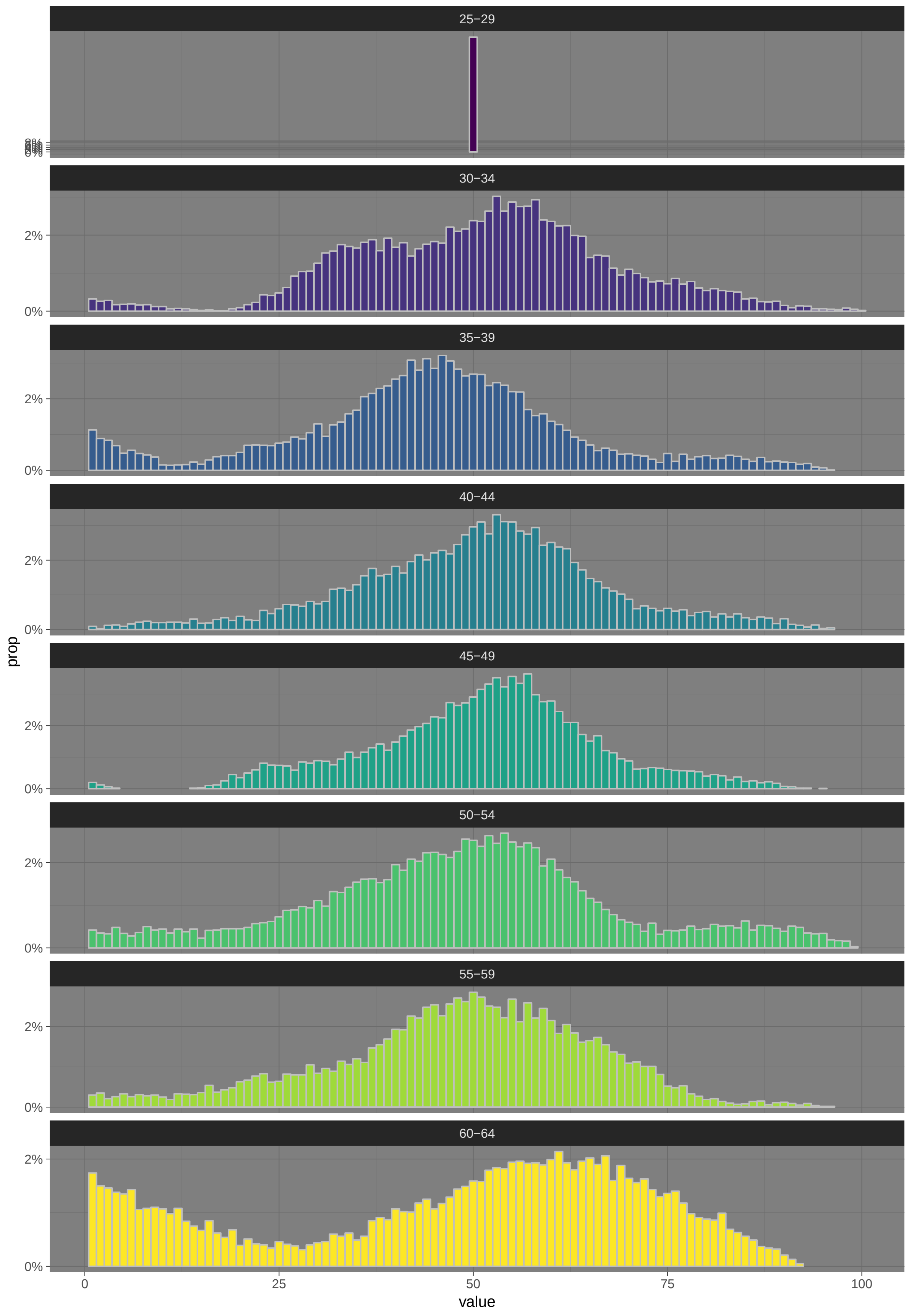
```
rbindlist(list("benchmark" = lifetime_income_benchmark,
               "pachinko"  = lifetime_income_pachinko),
          id = "group") %>%
  ggplot(aes(x = Percentile, y = lifetime_AWOTE, group = group, color = group)) +
  geom_line() +
  scale_y_continuous(name = "Lifetime income / AWOTE",
                     sec.axis = sec_axis(trans = ~.*awote_by_year[which.max(Year)][["AWOTE_annual"]],
                                         name = "Lifetime income",
                                         label = grattan_dollar)) +
  theme(legend.position = c(0.1, 0.9),
        legend.justification = c(0, 1))
```

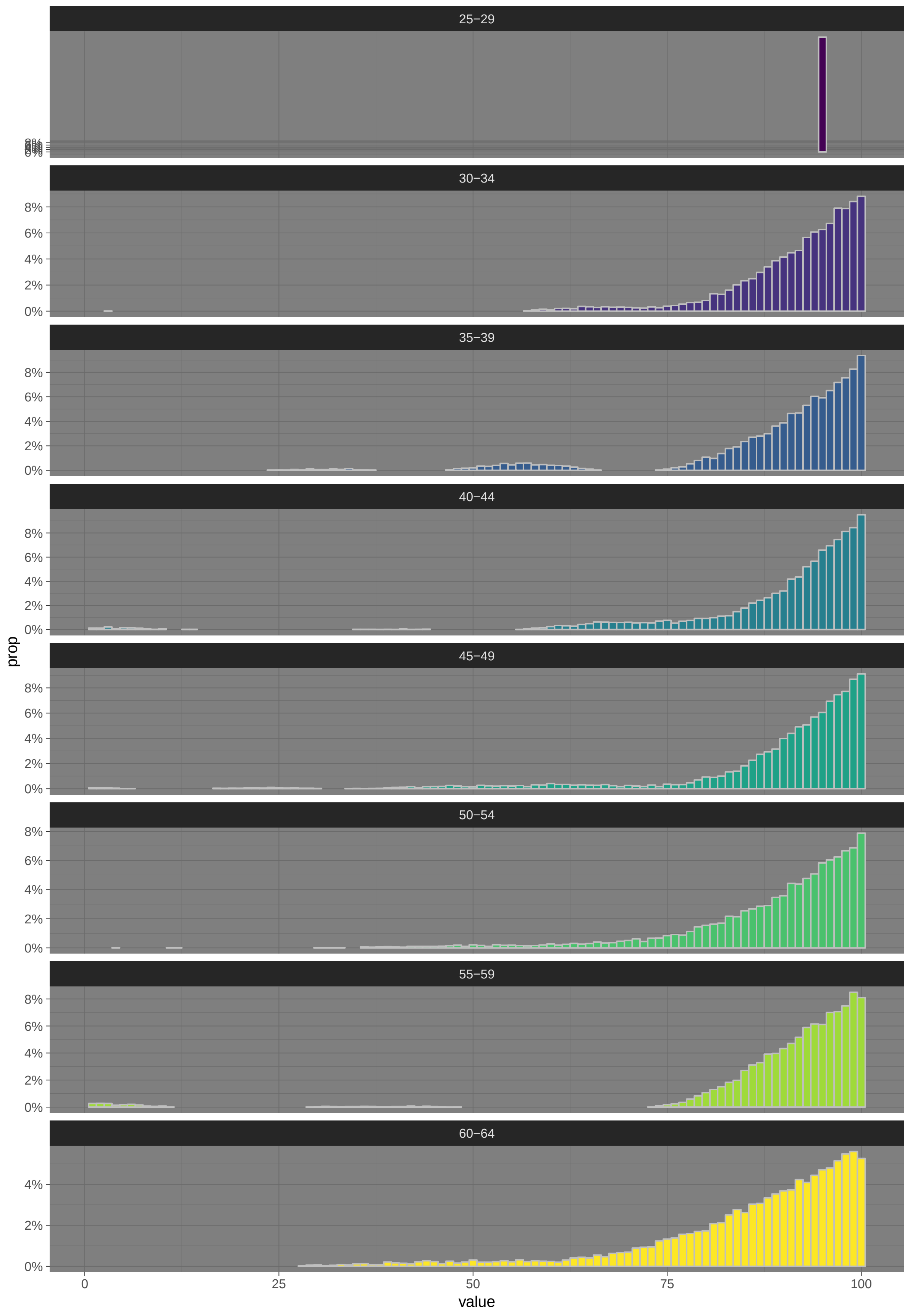


```r
for (the_percentile in c(10, 25, 50, 95)) {
  the_percentile_plot <-
    machine_widget[Percentile_2529 %in% the_percentile,
                   .SD,
                   .SDcols = grep("^((Percentile_[0-9]{4}\\.1)|(.*(6064|2529)))$",
                                  names(machine_widget)),
                   key = "simulation"] %>%
    melt.data.table(id.vars = "simulation") %>%
    .[, variable := sub("^Percentile_(..)(..).*$", "\\1-\\2", variable)] %>%
    .[order(-variable)] %>%
    .[, variable := factor(variable,
                           levels = rev(unique(.$variable)),
                           ordered = TRUE)] %>%
    ggplot(aes(x = value, fill = variable)) +
    geom_bar(width = 1,
             color = grey(level = 0.75),
             aes(y = ..prop..)) +
    facet_wrap( ~ variable, scales = "free_y", ncol = 1) +
    theme_dark()  +
    theme(legend.position = "none") +
    scale_y_continuous(labels = percent, breaks = c(0, 0.02, 0.04, 0.06, 0.08))

  print(the_percentile_plot)
}
```


25−29
30−34
35−39
40−44
45−49
50−54
55−59
60−64
prop
0%
2%
4%
0
25
50
75
100
value


25–29
30–34
35–39
40–44
45–49
50–54
55–59
60–64
prop
0%
2%
0
25
50
75
100
value


25–29
8%
6%
4%
2%
0%
30–34
2%
0%
35–39
2%
0%
40–44
2%
0%
45–49
2%
0%
50–54
2%
0%
55–59
2%
0%
60–64
2%
0%
prop
0
25
50
75
100
value


25−29
30−34
35−39
40−44
45−49
50−54
55−59
60−64
8%
6%
4%
2%
0%
prop
value
0
25
50
75
100