Package ‘R4GoodPersonalFinances’

March 13, 2025

Title Make Better Financial Decisions

Version 0.2.0

Description Make informed, data-driven decisions for your personal or household finances. Use tools and methods that are selected carefully to align with academic consensus, bridging the gap between theoretical knowledge and practical application. They assist you in finding optimal asset allocation, preparing for retirement or financial independence, calculating optimal spending, and more.
For more details see:
Haghani V., White J. (2023, ISBN:978-1-119-74791-8),
Idzorek T., Kaplan P. (2024, ISBN:9781952927379).

License MIT + file LICENSE

URL https://www.r4good.academy/,
https://r4goodacademy.github.io/R4GoodPersonalFinances/,
https://github.com/R4GoodAcademy/R4GoodPersonalFinances

BugReports https://github.com/R4GoodAcademy/R4GoodPersonalFinances/issues

Depends R (>= 4.1.0)

Imports bsicons, bslib, dplyr, ggplot2, ggrepel, ggtext, glue, PrettyCols, scales, shiny, tidyr, readr, fs, purrr, stringr, withr

Suggests spelling, testthat (>= 3.0.0), vdiffr

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

LazyData true

NeedsCompilation no

Author Kamil Wais [aut, cre, cph, fnd]
(<https://orcid.org/0000-0002-4062-055X>),
Olesia Wais [aut] (<https://orcid.org/0000-0002-8741-8674>)

Maintainer Kamil Wais <kamil.wais@gmail.com>

Repository CRAN

Date/Publication 2025-03-13 20:50:05 UTC

Contents

`calc_gompertz_joint_parameters`
Calculating the Gompertz model parameters for joint survival

Description

Calculating the Gompertz model parameters for joint survival

Usage

```
calc_gompertz_joint_parameters(
  p1 = list(age = NULL, mode = NULL, dispersion = NULL),
  p2 = list(age = NULL, mode = NULL, dispersion = NULL),
  max_age = 120
)
```

Arguments

`p1`	A list with `age`, `mode` and `dispersion` parameters for the first person (p1).
`p2`	A list with `age`, `mode` and `dispersion` parameters for the second person (p2).
`max_age`	A numeric. The maximum age for the Gompertz model.

Value

A list containing:

- `data` — A data frame with survival rates for 'p1', 'p2', 'joint' survival, and the fitted Gompertz model
- `mode` — The mode of the joint Gompertz distribution
- `dispersion` — The dispersion parameter of the joint Gompertz distribution

Examples

```
calc_gompertz_joint_parameters(
  p1 = list(
    age        = 65,
    mode       = 88,
    dispersion = 10.65
  ),
  p2 = list(
    age        = 60,
    mode       = 91,
    dispersion = 8.88
  ),
  max_age = 110
)
```

---

calc_gompertz_parameters

Calculating Gompertz model parameters

Description

Calculating Gompertz model parameters

Usage

```
calc_gompertz_parameters(
  mortality_rates,
  current_age,
  estimate_max_age = FALSE
)
```

Arguments

- `mortality_rates` — A data frame with columns `mortality_rate` and `age`. Usually the output of `read_hmd_life_tables()` function or filtered data from life_tables object.
- `current_age` — A numeric. Current age.
- `estimate_max_age` — A logical. Should the maximum age be estimated?

Value

A list containing:

`data`	The input mortality rates data frame with additional columns like 'survival_rate' and 'probability_of_death'
`mode`	The mode of the Gompertz distribution
`dispersion`	The dispersion parameter of the Gompertz distribution
`current_age`	The current age parameter
`max_age`	The maximum age parameter

References

Blanchet, David M., and Paul D. Kaplan. 2013. "Alpha, Beta, and Now... Gamma." Journal of Retirement 1 (2): 29-45. doi:10.3905/jor.2013.1.2.029.

Examples

```
mortality_rates <-
  dplyr::filter(
    life_tables,
    country == "USA" &
    sex     == "male" &
    year    == 2022
  )

calc_gompertz_parameters(
  mortality_rates = mortality_rates,
  current_age     = 65
)
```

`calc_gompertz_survival_probability`
Calculating Gompertz survival probability

Description

Calculating Gompertz survival probability

Usage

```
calc_gompertz_survival_probability(
  current_age,
  target_age,
  mode,
  dispersion,
  max_age = NULL
)
```

Arguments

`current_age`	Current age
`target_age`	Target age
`mode`	Mode of the Gompertz distribution
`dispersion`	Dispersion of the Gompertz distribution
`max_age`	Maximum age. Defaults to `NULL`.

Value

A numeric. The probability of survival from 'current_age' to 'target_age' based on the Gompertz distribution with the given parameters.

Examples

```
calc_gompertz_survival_probability(
  current_age = 65,
  target_age  = 85,
  mode        = 80,
  dispersion  = 10
)
```

---

calc_optimal_risky_asset_allocation

Calculate optimal risky asset allocation

Description

Calculates the optimal allocation to the risky asset using the Merton Share formula.

Usage

```
calc_optimal_risky_asset_allocation(
  risky_asset_return_mean,
  risky_asset_return_sd,
  safe_asset_return,
  risk_aversion
)
```

Arguments

`risky_asset_return_mean`
: A numeric. The expected (average) yearly return of the risky asset.

`risky_asset_return_sd`
: A numeric. The standard deviation of the yearly returns of the risky asset.

`safe_asset_return`
: A numeric. The expected yearly return of the safe asset.

`risk_aversion`
: A numeric. The risk aversion coefficient.

Details

Can be used to calculate the optimal allocation to the risky asset for vectors of inputs.

Value

A numeric. The optimal allocation to the risky asset. In case of `NaN()` (because of division by zero) the optimal allocation to the risky asset is set to 0.

See Also

- How to Determine Our Optimal Asset Allocation?
- Haghani V., White J. (2023) "The Missing Billionaires: A Guide to Better Financial Decisions." ISBN:978-1-119-74791-8.

Examples

```
calc_optimal_risky_asset_allocation(
  risky_asset_return_mean = 0.05,
  risky_asset_return_sd   = 0.15,
  safe_asset_return       = 0.02,
  risk_aversion           = 2
)

calc_optimal_risky_asset_allocation(
  risky_asset_return_mean = c(0.05, 0.06),
  risky_asset_return_sd   = c(0.15, 0.16),
  safe_asset_return       = 0.02,
  risk_aversion           = 2
)
```

`calc_purchasing_power` *Calculate purchasing power*

Description

Calculates changes in purchasing power over time, taking into account the real interest rate.

Usage

```
calc_purchasing_power(x, years, real_interest_rate)
```

Arguments

`x`	A numeric. The initial amount of money.
`years`	A numeric. The number of years.
`real_interest_rate`	A numeric. The yearly real interest rate.

Details

The real interest rate is the interest rate after inflation. If negative (e.g. equal to the average yearly inflation rate) it can show diminishing purchasing power over time. If positive, it can show increasing purchasing power over time, and effect of compounding interest on the purchasing power.

Value

A numeric. The purchasing power.

See Also

- How to Determine Our Optimal Asset Allocation?

Examples

```
calc_purchasing_power(x = 10, years = 30, real_interest_rate = -0.02)
calc_purchasing_power(x = 10, years = 30, real_interest_rate = 0.02)
```

---

calc_retirement_ruin *Calculating retirement ruin probability*

Description

Calculating retirement ruin probability

Usage

```
calc_retirement_ruin(
  portfolio_return_mean,
  portfolio_return_sd,
  age,
  gompertz_mode,
  gompertz_dispersion,
  portfolio_value,
  monthly_spendings,
  yearly_spendings = 12 * monthly_spendings,
  spending_rate = yearly_spendings/portfolio_value
)
```

Arguments

`portfolio_return_mean`
: A numeric. Mean of portfolio returns.

`portfolio_return_sd`
: A numeric. Standard deviation of portfolio returns.

`age`
: A numeric. Current age.

`gompertz_mode` A numeric. Gompertz mode.

`gompertz_dispersion` A numeric. Gompertz dispersion.

`portfolio_value` A numeric. Initial portfolio value.

`monthly_spendings` A numeric. Monthly spendings.

`yearly_spendings` A numeric. Yearly spendings.

`spending_rate` A numeric. Spending rate (initial withdrawal rate).

Value

A numeric. The probability of retirement ruin (between 0 and 1), representing the likelihood of running out of money during retirement.

References

Milevsky, M.A. (2020). Retirement Income Recipes in R: From Ruin Probabilities to Intelligent Drawdowns. Use R! Series. doi:10.1007/9783030514341.

Examples

```
calc_retirement_ruin(
  age                   = 65,
  gompertz_mode         = 88,
  gompertz_dispersion   = 10,
  portfolio_value       = 1000000,
  monthly_spendings     = 3000,
  portfolio_return_mean = 0.02,
  portfolio_return_sd   = 0.15
)
```

`calc_risk_adjusted_return` *Calculate risk adjusted return*

Description

Calculates the risk adjusted return for portfolio of given allocation to the risky asset.

Usage

```
calc_risk_adjusted_return(
  safe_asset_return,
  risky_asset_return_mean,
  risky_asset_allocation,
  risky_asset_return_sd = NULL,
  risk_aversion = NULL
)
```

Arguments

`safe_asset_return`
: A numeric. The expected yearly return of the safe asset.

`risky_asset_return_mean`
: A numeric. The expected (average) yearly return of the risky asset.

`risky_asset_allocation`
: A numeric. The allocation to the risky asset. Could be a vector. If it is the optimal allocation then parameters `risky_asset_return_sd` and `risk_aversion` can be omitted.

`risky_asset_return_sd`
: A numeric. The standard deviation of the yearly returns of the risky asset.

`risk_aversion`
: A numeric. The risk aversion coefficient.

Value

A numeric. The risk adjusted return.

See Also

- How to Determine Our Optimal Asset Allocation?
- Haghani V., White J. (2023) "The Missing Billionaires: A Guide to Better Financial Decisions." ISBN:978-1-119-74791-8.

Examples

```
calc_risk_adjusted_return(
  safe_asset_return = 0.02,
  risky_asset_return_mean = 0.04,
  risky_asset_return_sd = 0.15,
  risky_asset_allocation = 0.5,
  risk_aversion = 2
)

calc_risk_adjusted_return(
  safe_asset_return = 0.02,
  risky_asset_return_mean = 0.04,
  risky_asset_allocation = c(0.25, 0.5, 0.75),
  risky_asset_return_sd = 0.15,
  risk_aversion = 2
)
```

life_tables — *HMD life tables*

Description

A data frame based on: HMD. Human Mortality Database. Max Planck Institute for Demographic Research (Germany), University of California, Berkeley (USA), and French Institute for Demographic Studies (France). Available at www.mortality.org.

Usage

```
life_tables
```

Format

`life_tables`**:**
A data frame with 6 columns:

country Country name
sex Sex: "male", "female", "both"
year Year
age Age
mortality_rate Mortality rate
life_expectancy Life expectancy

Source

https://www.mortality.org

plot_gompertz_calibration — *Plotting the results of Gompertz model calibration*

Description

Plotting the results of Gompertz model calibration

Usage

```
plot_gompertz_calibration(params, mode, dispersion, max_age)
```

Arguments

`params`	A list returned by `calc_gompertz_parameters()` function.
`mode`	A numeric. The mode of the Gompertz model.
`dispersion`	A numeric. The dispersion of the Gompertz model.
`max_age`	A numeric. The maximum age of the Gompertz model.

Value

A ggplot2::ggplot() object showing the comparison between actual survival rates from life tables and the fitted Gompertz model.

Examples

```
mortality_rates <-
  dplyr::filter(
    life_tables,
    country == "USA" &
    sex     == "female" &
    year    == 2022
  )

params <- calc_gompertz_parameters(
  mortality_rates = mortality_rates,
  current_age     = 65
)

plot_gompertz_calibration(params = params)
```

---

plot_joint_survival *Plotting the results of Gompertz model calibration for joint survival*

Description

Plotting the results of Gompertz model calibration for joint survival

Usage

```
plot_joint_survival(params, include_gompertz = FALSE)
```

Arguments

params	A list returned by calc_gompertz_joint_parameters() function.
include_gompertz	A logical. Should the Gompertz survival curve be included in the plot?

Value

A ggplot2::ggplot() object showing the survival probabilities for two individuals and their joint survival probability.

Examples

```
params <- calc_gompertz_joint_parameters(
  p1 = list(
    age        = 65,
    mode       = 88,
    dispersion = 10.65
  ),
  p2 = list(
    age        = 60,
    mode       = 91,
    dispersion = 8.88
  ),
  max_age = 110
)

plot_joint_survival(params = params, include_gompertz = TRUE)
```

---

plot_purchasing_power *Plotting changes to the purchasing power over time*

Description

Plots the effect of real interest rates (positive or negative) on the purchasing power of savings over the span of 50 years (default).

Usage

```
plot_purchasing_power(
  x,
  real_interest_rate,
  years = 50,
  legend_title = "Real interest rate",
  seed = NA
)
```

Arguments

`x`	A numeric. The initial amount of money.
`real_interest_rate`	A numeric. The yearly real interest rate.
`years`	A numeric. The number of years.
`legend_title`	A character.
`seed`	A numeric. Seed passed to `geom_label_repel()`.

Value

A `ggplot2::ggplot()` object.

See Also

- How to Determine Our Optimal Asset Allocation?

Examples

```
plot_purchasing_power(
  x = 10,
  real_interest_rate = seq(-0.02, 0.04, by = 0.02)
)
```

`plot_retirement_ruin` *Plotting retirement ruin*

Description

Plotting retirement ruin

Usage

```
plot_retirement_ruin(
  portfolio_return_mean,
  portfolio_return_sd,
  age,
  gompertz_mode,
  gompertz_dispersion,
  portfolio_value,
  monthly_spendings = NULL
)
```

Arguments

`portfolio_return_mean`
: A numeric. Mean of portfolio returns.

`portfolio_return_sd`
: A numeric. Standard deviation of portfolio returns.

`age`
: A numeric. Current age.

`gompertz_mode`
: A numeric. Gompertz mode.

`gompertz_dispersion`
: A numeric. Gompertz dispersion.

`portfolio_value`
: A numeric. Initial portfolio value.

`monthly_spendings`
: A numeric. Monthly spendings.

Value

A ggplot2::ggplot() object showing the probability of retirement ruin for different monthly spending levels. If a specific 'monthly_spendings' value is provided, it will be highlighted on the plot with annotations.

Examples

```
plot_retirement_ruin(
  portfolio_return_mean = 0.034,
  portfolio_return_sd   = 0.15,
  age                   = 65,
  gompertz_mode         = 88,
  gompertz_dispersion   = 10,
  portfolio_value       = 1000000,
  monthly_spendings     = 3000
)
```

---

plot_risk_adjusted_returns

Plotting risk adjusted returns

Description

Plots the risk adjusted returns for portfolios of various allocations to the risky asset.

Usage

```
plot_risk_adjusted_returns(
  safe_asset_return,
  risky_asset_return_mean,
  risky_asset_return_sd,
  risk_aversion = 2,
  current_risky_asset_allocation = NULL
)
```

Arguments

`safe_asset_return`
: A numeric. The expected yearly return of the safe asset.

`risky_asset_return_mean`
: A numeric. The expected (average) yearly return of the risky asset.

`risky_asset_return_sd`
: A numeric. The standard deviation of the yearly returns of the risky asset.

`risk_aversion`
: A numeric. The risk aversion coefficient.

`current_risky_asset_allocation`
: A numeric. The current allocation to the risky asset. For comparison with the optimal allocation.

Value

A `ggplot2::ggplot()` object.

See Also

- How to Determine Our Optimal Asset Allocation?
- Haghani V., White J. (2023) "The Missing Billionaires: A Guide to Better Financial Decisions." ISBN:978-1-119-74791-8.

Examples

```
plot_risk_adjusted_returns(
  safe_asset_return            = 0.02,
  risky_asset_return_mean      = 0.04,
  risky_asset_return_sd        = 0.15,
  risk_aversion                = 2,
  current_risky_asset_allocation = 0.8
)
```

`print_currency` *Printing currency values or percentages*

Description

Wrapper functions for printing nicely formatted values.

Usage

```
print_currency(
  x,
  suffix = "",
  big.mark = ",",
  accuracy = NULL,
  prefix = NULL,
  ...
)

print_percent(x, accuracy = 0.1, ...)
```

Arguments

`x`	A numeric vector
`big.mark`	Character used between every 3 digits to separate thousands.
`accuracy`	A number to round to. Use (e.g.) `0.01` to show 2 decimal places of precision. If `NULL`, the default, uses a heuristic that should ensure breaks have the minimum number of digits needed to show the difference between adjacent values. Applied to rescaled data.

`prefix, suffix`	Symbols to display before and after value.
`...`	Other arguments passed on to `base::format()`.

Value

A character. Formatted value.

A character. Formatted value.

See Also

`scales::dollar()`

`scales::percent()`

Examples

```
print_currency(2345678, suffix = " PLN")
print_percent(0.52366)
```

`read_hmd_life_tables` *Reading HMD life tables*

Description

Reading HMD life tables

Usage

```
read_hmd_life_tables(
  path = getwd(),
  files = c("mltper_1x1.txt", "fltper_1x1.txt", "bltper_1x1.txt")
)
```

Arguments

`path`	A character. Path to the folder with life tables.
`files`	A character. Names of files with life tables.

Value

A data frame containing mortality data with columns:

`sex`	Character - sex ('male', 'female', or 'both')
`year`	Integer - the year of the data
`age`	Integer - age
`mortality_rate`	Numeric - mortality rate
`life_expectancy`	Numeric - life expectancy

References

HMD. Human Mortality Database. Max Planck Institute for Demographic Research (Germany), University of California, Berkeley (USA), and French Institute for Demographic Studies (France). Available at www.mortality.org

Examples

```
## Not run:
# Download 'txt' files
# ("mltper_1x1.txt", "fltper_1x1.txt", "bltper_1x1.txt")
# for a given country to the working directory
# from https://www.mortality.org after registration.

read_hmd_life_tables(path = getwd())

## End(Not run)
```

`run_app` — *Run a package app*

Description

Run a package app

Usage

```
run_app(
  which = c("risk-adjusted-returns", "purchasing-power", "retirement-ruin"),
  res = 120,
  shinylive = FALSE
)
```

Arguments

`which`	A character. The name of the app to run. Currently available: • `risk-adjusted-returns` - Plotting risk-adjusted returns for various allocations to the risky asset allows you to find the optimal allocation. • `purchasing-power` - Plotting the effect of real interest rates (positive or negative) on the purchasing power of savings over time. • `retirement-ruin` - Plotting the probability of retirement ruin.
`res`	A numeric. The initial resolution of the plots.
`shinylive`	A logical. Whether to use `shinylive` for the app.

Value

A `shiny::shinyApp()` object if `shinylive` is `TRUE`. Runs the app if `shinylive` is `FALSE` with `shiny::runApp()`.

Examples

```
run_app("risk-adjusted-returns")
run_app("purchasing-power")
run_app("retirement-ruin")
```

Index