# Package 'EloOptimized'

October 12, 2022

**Title** Optimized Elo Rating Method for Obtaining Dominance Ranks

**Version** 0.3.1

**Date** 2021-06-09

**Description** Provides an implementation of the maximum likelihood methods for deriving Elo scores as published in Foerster, Franz et al. (2016) <DOI:10.1038/srep35404>.

**Depends** R (>= 3.3.0)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Imports** dplyr, reshape2, BAMMtools, magrittr, lubridate, rlang

**URL** https://github.com/jtfeld/EloOptimized

**BugReports** https://github.com/jtfeld/EloOptimized/issues

**Suggests** knitr, rmarkdown, ggplot2

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Joseph Feldblum [aut, cre],
Steffen Foerster [aut],
Mathias Franz [aut]

**Maintainer** Joseph Feldblum <feldblum@umich.edu>

**Repository** CRAN

**Date/Publication** 2021-06-10 17:00:02 UTC

## R topics documented:

---

`cardinalize` *internal fn to create cardinal rank scores*

---

## Description

internal function for generating cardinal ranks

## Usage

```
cardinalize(x)
```

## Arguments

| | |
|---|---|
| x | input vector |

## Details

converts raw Elo scores into predicted number of individuals beaten (using Equation 1 from paper)

subtracting .5 is equivalent to removing the prob of winning against oneself because 1/(1 + exp(-0.01*0)) = 1/(1 + exp(0)) = 1/(1 + 1) = 1/2

## Value

returns new vector of cardinal rank scores

## chimpagg_f *Anonymized female chimpanzee pant-grunt data from Gombe*

### Description

Female data from Gombe National Park, Tanzania from 1969 to 2013. Data are submissive pant-grunt vocalizations.

### Usage

```
chimpagg_f
```

### Format

A data frame with 1015 rows and 3 variables:

**Date** date of interaction

**Winner** winning individual

**Loser** losing individual

### Source

Supplemental data published with Foerster, Franz et al. (2016). https://datadryad.org/stash/dataset/doi:10.5061/dryad.r4g74

## chimpagg_m *Anonymized male chimpanzee pant-grunt data from Gombe*

### Description

Data from Gombe National Park, Tanzania from 1978 to 2011. Data are submissive pant-grunt vocalizations.

### Usage

```
chimpagg_m
```

### Format

A data frame with 2741 rows and 3 variables:

**Date** date of interaction

**Winner** winning individual

**Loser** losing individual

## Source

Supplemental data published with Foerster, Franz et al. (2016). https://datadryad.org/stash/dataset/doi:10.5061/dryad.r4g74

---

# chimppres_f *Anonymized female chimpanzee presence data from Gombe*

---

## Description

Female presence data from Gombe National Park, Tanzania from 1969 to 2013. Presence criteria are given in Foerster, Franz et al. (2016)

## Usage

```
chimppres_f
```

## Format

A data frame with 44 rows and 3 variables:

**id** female code

**start_date** start date

**end_date** date of departure

## Source

Supplemental data published with Foerster, Franz et al. (2016). https://datadryad.org/stash/dataset/doi:10.5061/dryad.r4g74

---

# chimppres_m *Anonymized male chimpanzee presence data from Gombe*

---

## Description

Male presence data from Gombe National Park, Tanzania from 1978 to 2011. Presence criteria are given in Foerster, Franz et al. (2016)

## Usage

```
chimppres_m
```

## Format

A data frame with 22 rows and 3 variables:

**id** male code

**start_date** start date

**end_date** date of departure

## Source

Supplemental data published with Foerster, Franz et al. (2016). https://datadryad.org/stash/dataset/doi:10.5061/dryad.r4g74

---

# elo.m3_lik_vect *optimize k parameter and entry Elo scores, vectorized*

---

## Description

Function to optimize k parameter and entry Elo scores

## Usage

```
elo.m3_lik_vect(par, IA_data, all_ids)
```

## Arguments

| | |
|---|---|
| par | list of parameters, with par[1] being log(k), and par[2:length(par)] being the initial elo scores of individuals |
| IA_data | list of interaction data, with columns "Date", "Winner", and "Loser" (in that order) |
| all_ids | list of all ids to rank |

## Examples

```
# for internal use
```

`elo.model1` *Optimize k parameter in Elo rating method*

## Description

Function to optimize k parameter in Elo Rating Method

## Usage

```
elo.model1(par, burn_in=100, init_elo = 1000, IA_data, all_ids, p_function = "sigmoid",
  return_likelihood = T)
```

## Arguments

| | |
|---|---|
| `par` | initial value of log(k) |
| `burn_in` | burn in period for establishing initial elo scores. Defaults to 100 |
| `init_elo` | Initial Elo score for all individuals. Defaults to 1000 |
| `IA_data` | Data frame with Date, Winner, and Loser |
| `all_ids` | list of all IDs in sample |
| `p_function` | function used to calculate probability of winning. Defaults to sinusoidal function, but use "pnorm" to use the `pnorm`-based method implemented in the EloRating package. |
| `return_likelihood` | Logical; if TRUE, returns log likelihood based on given par, if FALSE returns agonistic interactions table with elo scores based on given value of par |

## Examples

```
#for internal use
```

`elo.model3` *optimize k parameter and entry Elo scores*

## Description

Function to optimize k parameter and entry Elo scores

## Usage

```
elo.model3(par, IA_data, all_ids, return_likelihood = T)
```

### Arguments

| | |
|---|---|
| `par` | list of parameters, with par[1] being log(k), and par[2:length(par)] being the initial elo scores of individuals |
| `IA_data` | list of interaction data, with columns "Date", "Winner", and "Loser" (in that order) |
| `all_ids` | list of all ids to rank |
| `return_likelihood` | If TRUE, returns the total likelihood based on all interactions given a particular set of parameters. If FALSE, returns a table of Elo scores based on a given set of parameters. |

### Examples

```
# for internal use
```

---

`EloOptimized` *EloOptimized: ML fitting of Elo Scores*

---

### Description

This package implements the maximum likelihood methods for deriving Elo scores as published in Foerster, Franz et al. (2016). Chimpanzee females queue but males compete for social status. Scientific Reports 6, 35404, doi:10.1038/srep35404

### Primary functions

- `eloratingopt`: main function
- `eloratingfixed`: traditional Elo scores function
- `elo.model1`: internal function for fitting model type 1
- `elo.model3`: internal function for fitting model type 3
- `elo.m3_lik_vect`: vectorized internal function for fitting mod type 3

### Plans for future development

- Make package more modular, with a more flexible wrapper function.
- Option to specify K during burn-in period when fitting only K
- Add additional example data
- Create vignette, other package doohickies
- Add additional user control of the optimization procedure, allowing for specification of the burn in period, optimization algorithm, and initial values for optimization.
- Add functionality to plot Elo trajectories from within package.

`eloratingfixed` *Create daily elo ranks and multiple derivatives with user-defined parameter values*

## Description

Conducts traditional elo rating analyses using specified K value and outputs raw, normalized, cardinal, and categorical ranks as a list object in R or in an output file. For optimized Elo parameters, use `eloratingopt`.

## Usage

```
eloratingfixed(agon_data, pres_data, k = 100, init_elo = 1000, outputfile = NULL,
  returnR = TRUE, p_function = "sigmoid")
```

## Arguments

| | |
|---|---|
| `agon_data` | Input data frame with dominance interactions, should only contain Date, Winner, Loser. Date should be formatted as MONTH/DAY/YEAR, or already as Date class. |
| `pres_data` | Input data frame with columns "id", "start_date" and "end_date". Date columns should be formatted as MONTH/DAY/YEAR, or already as Date class. If all IDs are present the whole time, you ignore this and a pres_data table will be automatically generated. |
| `k` | Specified value of the k parameter, default is 100 |
| `init_elo` | The starting Elo value for all individuals, default is 1000 |
| `outputfile` | Name of csv file to save ranks to. Default is NULL, in which case the function will only return a table in R. If you supply an output file name the function will save the results as a csv file in your working directory. |
| `returnR` | whether to return an R object from the function call. Default is TRUE |
| `p_function` | function defining probability of winning. Default "sigmoid" is equation (1) from Foerster, Franz et al 2016. Use "pnorm" to use the `pnorm`-based method implemented in the EloRating package. |

## Details

This function accepts a data frame of date-stamped dominance interactions and (optionally) a data frame of start and end dates for each individual to be ranked, and outputs daily Elo scores with parameters specified by the user. The default function used to determine probability of winning is equation (1) from Foerster, Franz et al. 2016, but for ease of comparison with the EloRating package, we also added the option to use the `pnorm`-based method implemented in the EloRating package, and future development will add the option to use the original function from Elo 1978 (as implemented in the elo package). This function does not require large presence matrices, and efficiently calculates a series of additional indices (described below).

As opposed to the eloratingopt function, this procedure only requires that included individuals have at least one win *or* one loss.

A detailed description of the function output is given in the **Value** section of this help file:

## Value

Returns a list with six elements:

- **elo** Data frame with all IDs and dates they were present, with the following columns:
  - Date: Dates of study period
  - Individual: the names of each ranked individual, for each date they were present
  - Elo: fitted Elo scores for each individual on each day
  - EloOrdinal: Daily ordinal rank based on Elo scores
  - EloScaled: Daily Elo scores rescaled between 0 and 1 according to

    $$([individualElo]-min([dailyEloscores]))/(max([dailyEloscores])-min([dailyEloscores]))$$

  - ExpNumBeaten: expected number of individuals in the group beaten, which is the sum of winning probabilities based on relative Elo scores of an individual and all others, following equation (4) in Foerster, Franz et al. 2016
  - EloCardinal: ExpNumBeaten values rescaled as a percentage of the total number of ranked individuals present in the group on the day of ranking. We encourage the use of this measure.
  - JenksEloCardinal: Categorical rank (high, mid, or low) using the Jenks natural breaks classification method implemented in the R package BAMMtools. See getJenksBreaks
- **k** User-defined value of the k parameter
- **init_elo** User-defined initial Elo score when individuals enter the hierarchy
- **pred_accuracy** Proportion of correctly predicted interactions
- **logL** The overall log-likelihood of the observed data given the user-supplied parameter values based on winning probabilities (as calculated in equation (1) of Foerster, Franz et al 2016) for all interactions

## Examples

```
nbadata = EloOptimized::nba #nba wins and losses from the 1995-96 season
nbaelo = eloratingfixed(agon_data = nbadata)
# generates traditional Elo scores (with init_elo = 1000 & k = 100) and saves
#   them as "nbaelo"
```

`eloratingopt` *Create daily ML fitted Elo ranks and multiple derivatives*

## Description

Conducts **optimized** elo rating analyses as per Foerster, Franz et al and outputs raw, normalized, cardinal, and categorical ranks as a list object in R or in an output file. For non-optimized Elo score calculation, use `eloratingfixed`.

## Usage

```
eloratingopt(agon_data, pres_data, fit_init_elo = FALSE, outputfile = NULL,
  returnR = TRUE)
```

## Arguments

| | |
|---|---|
| `agon_data` | Input data frame with dominance interactions, should only contain Date, Winner, Loser. Date should be formatted as MONTH/DAY/YEAR, or already as Date class. |
| `pres_data` | Input data frame with columns "id", "start_date" and "end_date". Date columns should be formatted as MONTH/DAY/YEAR, or already as Date class. If all IDs are present the whole time, you can ignore this and a pres_data table will be automatically generated. |
| `fit_init_elo` | If FALSE (the default), fits only the K parameter, with a default starting Elo score of 1000 for each individual. If TRUE, fits K and starting Elo for each individual. The latter option is *much* slower. |
| `outputfile` | Name of csv file to save ranks to. Default is NULL, in which case the function will only return a table in R. If you supply an output file name the function will save the results as a csv file in your working directory. |
| `returnR` | whether to return an R object from the function call. Default is TRUE |

## Details

This function accepts a data frame of date-stamped dominance interactions and (optionally) a data frame of start and end dates for each individual to be ranked, and outputs daily Elo scores with K parameter, and optionally initial elo scores, fitted using a maximum likelihood approach. The optimization procedure uses the `optim()` function, with a burn in period of 100 interactions. We use the "Brent" method when fitting only the K parameter, and the "BFGS" method for fitting both K and initial Elo scores. See `optim` for more details. Future package development will add additional user control of the optimization procedure, allowing for specification of the burn in period, optimization algorithm, and initial values for optimization.

Note also that the fitting procedure requires each individual to have at least one win and one loss, so any individual that doesn't meet those criteria is automatically removed. Additionally, any instance of an individual winning against itself is cleaned from the data, and several other checks of the data are performed before the optimization procedure is run.

A detailed description of the function output is given in the **Value** section of this help file:

## Value

Returns a list with five or six elements (depending on input):

- **elo** Data frame with all IDs and dates they were present, with the following columns:
  - Date: Dates of study period
  - Individual: the names of each ranked individual, for each date they were present
  - Elo: fitted Elo scores for each individual on each day
  - EloOrdinal: Daily ordinal rank based on Elo scores
  - EloScaled: Daily Elo scores rescaled between 0 and 1 according to

    $$([individualElo]-min([dailyEloscores])/(max([dailyEloscores])-min([dailyEloscores]))$$

  - ExpNumBeaten: expected number of individuals in the group beaten, which is the sum of winning probabilities based on relative Elo scores of an individual and all others, following equation (4) in Foerster, Franz et al. 2016
  - EloCardinal: ExpNumBeaten values rescaled as a percentage of the total number of ranked individuals present in the group on the day of ranking. We encourage the use of this measure.
  - JenksEloCardinal: Categorical rank (high, mid, or low) using the Jenks natural breaks classification method implemented in the R package BAMMtools. See `getJenksBreaks`
- **k** The maximum-likelihood fitted k parameter value
- **pred_accuracy** Proportion of correctly predicted interactions
- **maxLogL** The overall log-likelihood of the observed data given the fitted parameter values based on winning probabilities (as calculated in equation (1) of Foerster, Franz et al 2016) for all interactions
- **AIC** Akaike's Information Criterion value as a measure of model fit
- **init_elo** (*Only returned if you fit initial Elo scores*) initial Elo for each individual

## Examples

```
nbadata = EloOptimized::nba #nba wins and losses from the 1995-96 season
nbaelo = eloratingopt(agon_data = nbadata, fit_init_elo = FALSE)
# generates optimized elo scores (optimizing only K) and saves them as "nbaelo"
```

---

`jenksify` *internal fn to generate categorical ranks*

---

## Description

internal function for generating categorical ranks using jenks natural breaks algorithm

## Usage

```
jenksify(x)
```

### Arguments

| | |
|---|---|
| x | input vector |

### Details

creates categorical ranks using jenks natural breaks algorithm

### Value

returns new vector of categorical ranks (high/medium/low)

---

## nba *NBA games 1995-96*

---

### Description

Outcome of NBA games during the 1995-1996 regular season, adapted from a dataset from fivethirtyeight

### Usage

```
nba
```

### Format

A data frame with 1189 rows and 3 variables:

**Date** date of game

**Winner** winning team

**Loser** losing team

### Source

https://github.com/fivethirtyeight/data/blob/master/nba-elo/nbaallelo.csv

`relativize` *internal fn to relativize rank scores*

## Description

internal function for generating scaled cardinal ranks

## Usage

```
relativize(x)
```

## Arguments

| | |
|---|---|
| `x` | input vector |

## Details

scales cardinal Elo scores between 0 and 1

## Value

returns new vector of scaled rank scores

# Index