

# Package ‘mRchmadness’

March 20, 2017

**Title** Numerical Tools for Filling Out an NCAA Basketball Tournament Bracket

**Version** 1.0

**URL** <https://github.com/eshayer/mRchmadness>

**Imports** dplyr, glmnet, Matrix, rvest, xml2

**Description** Scrape season results, estimate win probabilities, and find a competitive bracket for your office pool. Additional utilities include: scraping population picks; simulating tournament results; and testing your bracket in simulation.

**Depends** R (>= 3.3.2)

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Eli Shayer [aut, cre],  
Scott Powers [aut]

**Maintainer** Eli Shayer <eshayer@stanford.edu>

## R topics documented:

bradley.terry . . . . .	2
draw.bracket . . . . .	2
find.bracket . . . . .	3
fold . . . . .	4
score.bracket . . . . .	4
scrape.game.results . . . . .	5
scrape.population.distribution . . . . .	5
scrape.team.game.results . . . . .	6
scrape.teams . . . . .	6
sim.bracket . . . . .	6
test.bracket . . . . .	7
unfold . . . . .	7

<b>Index</b>	<b>9</b>
--------------	----------

---

bradley.terry	<i>Fit a Bradley-Terry model on game score data</i>
---------------	---

---

**Description**

Fit a Bradley-Terry model on game score data

**Usage**

```
bradley.terry(games)
```

**Arguments**

games	data.frame with the following columns: game.id, home.id, away.id, home.score, away.score, neutral, ot (matched by output of scrape.game.results)
-------	--

**Author(s)**

sspowers

---

draw.bracket	<i>Plot bracket to device</i>
--------------	-------------------------------

---

**Description**

Plot bracket to device

**Usage**

```
draw.bracket(bracket.empty, bracket.filled = NULL)
```

**Arguments**

bracket.empty	a length-64 character vector giving the field of 64 teams in the tournament, in order of initial overall seeding
bracket.filled	an optional length-63 character vector encoding tournament results (matching output from simulate.bracket)

**Author(s)**

sspowers

---

find.bracket	<i>Fill out a bracket based on some criteria</i>
--------------	--

---

## Description

Fill out a bracket based on some criteria

## Usage

```
find.bracket(bracket.empty, probability.matrix, pool.size = 30,
  num.candidates = 100, num.sims = 1000, criterion = c("percentile",
  "score", "win"), bonus.round = c(1, 2, 4, 8, 16, 32), bonus.seed = rep(0,
  16), bonus.combine = c("add", "multiply"))
```

## Arguments

bracket.empty	a length-64 character vector giving the field of 64 teams in the tournament, in order of initial overall seeding
probability.matrix	a matrix of probabilities, with rows and columns corresponding to teams, matching the output of <code>bradley.terry()</code>
pool.size	number of brackets in your pool, matters only if <code>criterion == "win"</code> (default is 30)
num.candidates	number of random brackets to try, taking the best one (default is 100)
num.sims	number of simulations over which to evaluate the candidate brackets (default is 1000)
criterion	how to choose among candidate brackets: "percentile" (default, maximize expected percentile within pool), "score" (maximize expected number of points) or "win" (maximize probability of winning pool).
bonus.round	a length-6 vector giving the number of points awarded in your pool's scoring rules for correct picks in each round (default is $2^{\text{round}}$ )
bonus.seed	a length-16 vector giving the bonus awarded for correctly picking winner based on winner's seed (default is zero)
bonus.combine	how to combine the round bonus with the seed bonus to get the number of points awarded for each correct pick: "add" (default) or multiply

## Author(s)

sspowers

---

fold	<i>Fold a vector onto itself</i>
------	----------------------------------

---

### Description

Fold a vector onto itself

### Usage

```
fold(x, block.size = 1)
```

### Arguments

x	a vector
block.size	the size of groups in which to block the data

### Value

a new vector in the following order: first block, last block, second block, second-to-last block, ...

### Author(s)

sspowers

---

score.bracket	<i>Compute score for bracket given actual result</i>
---------------	--

---

### Description

Compute score for bracket given actual result

### Usage

```
score.bracket(bracket.empty, bracket.picks, bracket.outcome,
  bonus.round = c(1, 2, 4, 8, 16, 32), bonus.seed = rep(0, 16),
  bonus.combine = c("add", "multiply"))
```

### Arguments

bracket.empty	a length-64 character vector giving the field of 64 teams in the tournament, in order of initial overall seeding
bracket.picks	an length-63 character vector encoding the picks (this is the bracket to be evaluated)
bracket.outcome	a 63-row matrix encoding the outcome of multiple simulations of the tournament. bracket.picks will be scored against each outcome
bonus.round	a length-6 vector giving the number of points awarded in your pool's scoring rules for correct picks in each round (default is 2^round)

bonus.seed	a length-16 vector giving the bonus awarded for correctly picking winner based on winner's seed (default is zero)
bonus.combine	how to combine the round bonus with the seed bonus to get the number of points awarded for each correct pick: "add" (default) or multiply

**Author(s)**

sspowers

---

scrape.game.results	<i>Scrape the game-by-game results of the NCAA MBB season</i>
---------------------	---

---

**Description**

Scrape the game-by-game results of the NCAA MBB season

**Usage**

scrape.game.results(year)

**Author(s)**

eshayer

---

scrape.population.distribution	<i>Scrape the average rate of teams being picked to win across all ESPN brackets</i>
--------------------------------	--

---

**Description**

Scrape the average rate of teams being picked to win across all ESPN brackets

**Usage**

scrape.population.distribution(year)

**Author(s)**

eshayer

**Examples**

populationDistribution = scrape.population.distribution(2017)

---

```
scrape.team.game.results
```

*Scrape game results for a single team-year combination*

---

### Description

Scrape game results for a single team-year combination

### Usage

```
scrape.team.game.results(year, id)
```

---

```
scrape.teams
```

*Scrape the team names and ids from the ESPN NCAA MBB index*

---

### Description

Scrape the team names and ids from the ESPN NCAA MBB index

### Usage

```
scrape.teams()
```

---

```
sim.bracket
```

*Simulate the full bracket starting with an empty bracket*

---

### Description

Simulate the full bracket starting with an empty bracket

### Usage

```
sim.bracket(bracket.empty, probability.matrix, num.reps = 1)
```

### Arguments

`bracket.empty` a length-64 character vector giving the field of 64 teams in the tournament, in order of initial overall seeding

`probability.matrix` a matrix of probabilities, with rows and columns corresponding to teams, matching the output of `bradley.terry()`

`num.reps` number of simulations to perform (default is 1)

### Author(s)

sspowers

---

test.bracket	<i>Test a bracket</i>
--------------	-----------------------

---

**Description**

Test a bracket

**Usage**

```
test.bracket(bracket.empty, probability.matrix, bracket.picks, pool.size = 30,
  num.sims = 1000, bonus.round = c(1, 2, 4, 8, 16, 32),
  bonus.seed = rep(0, 16), bonus.combine = c("add", "multiply"))
```

**Arguments**

bracket.empty	a length-64 character vector giving the field of 64 teams in the tournament, in order of initial overall seeding
probability.matrix	a matrix of probabilities, with rows and columns corresponding to teams, matching the output of <code>bradley.terry()</code>
bracket.picks	an length-63 character vector encoding your picks (this is the bracket to be evaluated)
pool.size	number of brackets in your pool, matters only if <code>criterion == "win"</code> (default is 30)
num.sims	number of simulations over which to evaluate the candidate brackets (default is 1000)
bonus.round	a length-6 vector giving the number of points awarded in your pool's scoring rules for correct picks in each round (default is $2^{\text{round}}$ )
bonus.seed	a length-16 vector giving the bonus awarded for correctly picking winner based on winner's seed (default is zero)
bonus.combine	how to combine the round bonus with the seed bonus to get the number of points awarded for each correct pick: "add" (default) or multiply

**Author(s)**

sspowers

---

unfold	<i>Unfold a vector (the inverse of the fold function)</i>
--------	---

---

**Description**

Unfold a vector (the inverse of the fold function)

**Usage**

```
unfold(x, block.size = 1)
```

**Arguments**

`x`                      a vector  
`block.size`            the size of groups in which to block the data

**Value**

a vector in the following order: block 1, block 3, ..., block n-1, block n, block n-2, ..., block 2.

**Author(s)**

sspowers



# Index

`bradley.terry`, [2](#)

`draw.bracket`, [2](#)

`find.bracket`, [3](#)

`fold`, [4](#)

`score.bracket`, [4](#)

`scrape.game.results`, [5](#)

`scrape.population.distribution`, [5](#)

`scrape.team.game.results`, [6](#)

`scrape.teams`, [6](#)

`sim.bracket`, [6](#)

`test.bracket`, [7](#)

`unfold`, [7](#)