Package ‘YangHuiMagic’

March 23, 2026

Type Package

Title A Generalization of Yang Hui's Magic Squares

Version 1.1

Description The generalized construction methods for magic squares, inspired by the ancient Chinese mathematician Yang Hui's classical work ``Xu Gu Zhai Qi Suan Fa''. These methods can construct 4n-order magic squares and 2(2n+1)-order magic squares.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Chun-Xiao Nie [aut, cre] (ORCID: <https://orcid.org/0000-0002-7790-0803>)

Maintainer Chun-Xiao Nie <niechunxiao2009@163.com>

Repository CRAN

Date/Publication 2026-03-23 10:00:22 UTC

Contents

`construct_M` *Construct the initial matrix M (satisfying the form of Matrix (8))*

Description

Construct the initial matrix M (satisfying the form of Matrix (8))

Usage

```
construct_M(n, a11, d1, d2, d12, d21, d22)
```

Arguments

`n`	Parameter, final order is 4n
`a11`	Parameters from the lemma, must satisfy d12 + d21 = d22
`d1`	Parameters from the lemma, must satisfy d12 + d21 = d22
`d2`	Parameters from the lemma, must satisfy d12 + d21 = d22
`d12`	Parameters from the lemma, must satisfy d12 + d21 = d22
`d21`	Parameters from the lemma, must satisfy d12 + d21 = d22
`d22`	Parameters from the lemma, must satisfy d12 + d21 = d22

Value

4n x 4n matrix

`generate_A11` *Generate the 2n x 2n A11 matrix (arithmetic progression form)*

Description

Generate the 2n x 2n A11 matrix (arithmetic progression form)

Usage

```
generate_A11(n, a11, d1, d2)
```

Arguments

`n`	Parameter, final order is 4n
`a11`	Upper-left corner starting value
`d1`	Column direction common difference (horizontal increment)
`d2`	Row direction common difference (vertical increment)

Value

2n x 2n matrix

`generate_symmetric_L` *Generate a symmetric L set satisfying the condition (first n numbers and their complements)*

Description

Generate a symmetric L set satisfying the condition (first n numbers and their complements)

Usage

```
generate_symmetric_L(n)
```

Arguments

`n`	Parameter, final order 4n

Value

Integer vector of length 2n

`is_magic_square` *Verify magic square properties*

Description

Verify magic square properties

Usage

```
is_magic_square(M)
```

Arguments

`M`	Square matrix

Value

TRUE if M satisfies the magic square condition

Examples

```
M=YangConway(m = 2, d_type = "unit", template_set = 1)
is_magic_square(M)
```

odd_magic_square — *Construct odd order magic square (Siamese method)*

Description

Construct odd order magic square (Siamese method)

Usage

```
odd_magic_square(k)
```

Arguments

k	Odd order number

Value

k x k magic square with numbers 1 to k^2

Examples

```
odd_magic_square(7)
```

transform_M — *Apply row left-right flips and column up-down flips to a matrix*

Description

Apply row left-right flips and column up-down flips to a matrix

Usage

```
transform_M(M, L1, L2)
```

Arguments

M	Initial matrix
L1	Vector of row indices (1-based) to be left-right flipped
L2	Vector of column indices (1-based) to be up-down flipped

Value

Transformed matrix

YangConway *Construct singly even order magic square (Yang-Hui Conway generalization)*

Description

Construct singly even order magic square (Yang-Hui Conway generalization)

Usage

```
YangConway(m, d_type = "square", template_set = 2)
```

Arguments

m	Positive integer, final order n = 2*(2*m+1)
d_type	Type of common difference: "unit" (d=1) or "square" (d=(2m+1)^2)
template_set	Template series: 1~4 corresponding to matrices 4,7,9,10 in the paper

Value

n x n magic square

Examples

```
# Example 1: Reproduce Yang-Hui 6th order magic square (Yang diagram),
# m=1, d_type="square", template_set=2
cat("===== Yang-Hui 6th order magic square (Yang diagram) =====\n")
yanghui6 <- YangConway(m = 1, d_type = "square", template_set = 2)
print(yanghui6)
cat("Is it a magic square: ", is_magic_square(yanghui6), "\n\n")
# Example 2: Construct 10th order magic square (matrix 8 in the paper),
# m=2, d_type="square", template_set=2
cat("===== 10th order magic square (Yang-Hui method, template 7) =====\n")
yanghui10 <- YangConway(m = 2, d_type = "square", template_set = 2)
print(yanghui10)
cat("Is it a magic square: ", is_magic_square(yanghui10), "\n\n")
# Example 3: Use common difference 1 (LUX method) to construct 10th order magic square, template 4
cat("===== 10th order magic square (LUX method, template 4) =====\n")
lux10 <- YangConway(m = 2, d_type = "unit", template_set = 1)
print(lux10)
cat("Is it a magic square: ", is_magic_square(lux10), "\n\n")
```

YangRamanujan *Construct doubly even order magic square (Yang-Hui · Dürer · Ramanujan inspired)*

Description

Construct doubly even order magic square (Yang-Hui · Dürer · Ramanujan inspired)

Usage

```
YangRamanujan(n, params = "natural", L1 = NULL, L2 = NULL)
```

Arguments

n	Positive integer, order = 4n
params	Parameter setting. Can be a list containing a11, d1, d2, d12, d21 (d22 is optional, automatically set to d12+d21). Predefined strings: - "natural": natural square parameters (a11=1, d1=1, d2=4n, d12=2n, d21=2n*4n) - "type1" : second row of Table 3 (a11=1, d1=1, d2=2n, d12=(2n)^2, d21=2*(2n)^2) - "type2" : third row of Table 3 (a11=1, d1=4n, d2=1, d12=?, d21=?, d22=?) [values incomplete, for illustration only]
L1	Sets of rows/columns to flip (1-based), default automatically generated symmetric sets
L2	Sets of rows/columns to flip (1-based), default automatically generated symmetric sets

Value

4n x 4n magic square matrix

Examples

```
# Example 1: Ramanujan's 8th order magic square (matrix 4 in the paper)
# Using natural square parameters, L1 = L2 = {2,3,6,7}
cat("\n===== Ramanujan's 8th order magic square =====\n")
ramanujan8 <- YangRamanujan(n = 2,params = "natural",L1 = c(2, 3, 6, 7),L2 = c(2, 3, 6, 7))
print(ramanujan8)
cat("Is it a magic square: ", is_magic_square(ramanujan8), "\n")
# Example 2: One of Yang-Hui's 4th order magic squares (matrix 2 in the paper)
# For n=1, use natural square, L1 = L2 = {2,3} (rows/columns 1..4 for order 4)
cat("\n===== Yang-Hui 4th order magic square (matrix 2) =====\n")
yanghui4 <- YangRamanujan(n = 1, params = "natural",L1 = c(2, 3),L2 = c(2, 3))
print(yanghui4)
cat("Is it a magic square: ", is_magic_square(yanghui4), "\n")
# Example 3: Dürer's 4th order magic square (matrix 3 in the paper)
# According to Table 1, use specific parameters (n=1) and L1 = L2 = {2,3}
# Parameters from row (22) in Table 1: a11=1, d1=-1, d2=-8, d12=-2, d21=-4, d22=-6
cat("\n===== Dürer's 4th order magic square (matrix 3) =====\n")
```

```
duerer_params <- list(a11 = 1, d1 = 1, d2 = 8, d12 = 2, d21 = 4)
duerer4 <- YangRamanujan(n = 1,params = duerer_params,L1 = c(2, 3),L2 = c(2, 3))
print(duerer4)
cat("Is it a magic square: ", is_magic_square(duerer4), "\n")
# Example 4: Custom 12th order magic square (n=3), using natural square, default symmetric L
cat("\n===== Custom 12th order magic square (natural square, default L) =====\n")
magic12 <- YangRamanujan(n = 3, params = "natural")
# Print only first 6 rows to save space
print(magic12)
cat("Is it a magic square: ", is_magic_square(magic12), "\n")
```

Index