Package ‘RQEntangle’

May 7, 2026

Type Package

Title Quantum Entanglement of Bipartite System

Version 0.1.3

Description

It computes the Schmidt decomposition of bipartite quantum systems, discrete or continuous, and their respective entanglement metrics. See Artur Ekert, Peter L. Knight (1995) <doi:10.1119/1.17904> for more details.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R(>= 2.15.1), itertools(>= 0.1-3), iterators

Suggests knitr, rmarkdown, dplyr, ggplot2, roxygen2

RoxygenNote 6.1.0

URL https://github.com/stephenhky/RQEntangle

BugReports https://github.com/stephenhky/RQEntangle/issues

VignetteBuilder knitr

NeedsCompilation no

Author Kwan-Yuet Ho [aut, cre]

Maintainer Kwan-Yuet Ho <stephenhky@yahoo.com.hk>

Repository CRAN

Date/Publication 2019-01-04 10:40:08 UTC

Contents

continuous.function.interpolate 2
continuous.schmidt.decompose 2
discretize.continuous.bipartitefunc 3
entanglement.entropy 4

interpolated.continuous.function 4
negativity 5
participation.ratio 5
reduced.denmat 6
schmidt.decompose 6

Index **8**

`continuous.function.interpolate`
Interpolate values of functions.

Description

Interpolate values of functions.

Usage

```
continuous.function.interpolate(xarr, yarr, x)
```

Arguments

`xarr`	a vector of x (sorted)
`yarr`	a vector of y
`x`	given value of x

Value

interpolated value of y

`continuous.schmidt.decompose`
Perform a continuous Schmidt decomposition

Description

Perform a continuous Schmidt decomposition

Usage

```
continuous.schmidt.decompose(bifunc, x1lo, x1hi, x2lo, x2hi, nbx1 = 100,
  nbx2 = 100, keep = min(10, nbx1, nbx2))
```

Arguments

bifunc	bipartitite continuous wavefunction
x1lo	lower limit of x1
x1hi	upper limit of x1
x2lo	lower limit of x2
x2hi	upper limit of x2
nbx1	number of discretized x1 (default: 100)
nbx2	number of discretized x2 (default: 100)
keep	number of Schmidt modes to keep (default: minimum of 10, nbx1, and nbx2)

Value

Schmidt modes, including the eigenvalues, and the lambda interpolated function of the Schmidt modes

Examples

```
coupled.harm.fcn<- function(x1,x2) exp(-((0.5*(x1+x2))**2))*exp(-(x1-x2)**2)*sqrt(2./pi)
continuous.schmidt.decompose(coupled.harm.fcn, -10, 10, -10, 10)
```

---

discretize.continuous.bipartitefunc

Making a discretized tensor for a continuous function

Description

Making a discretized tensor for a continuous function

Usage

```
discretize.continuous.bipartitefunc(bifunc, x1lo, x1hi, x2lo, x2hi,
  nbx1 = 100, nbx2 = 100)
```

Arguments

bifunc	bipartitite continuous wavefunction
x1lo	lower limit of x1
x1hi	upper limit of x1
x2lo	lower limit of x2
x2hi	upper limit of x2
nbx1	number of discretized x1 (default: 100)
nbx2	number of discretized x2 (default: 100)

Value

discretized tensor for Schmidt decomposition

---

entanglement.entropy *Calculate the entanglement entropy given the calculate Schmidt modes.*

Description

Calculate the entanglement entropy given the calculate Schmidt modes.

Usage

```
entanglement.entropy(modes)
```

Arguments

modes	Schmidt modes

Value

entanglement entropy

Examples

```
singlet<- matrix(c(0, sqrt(0.7), sqrt(0.3), 0), byrow = TRUE, nrow = 2)
modes<- schmidt.decompose(singlet)
entanglement.entropy(modes)
```

---

interpolated.continuous.function *Lambda function of the interpolated continous function.*

Description

Lambda function of the interpolated continous function.

Usage

```
interpolated.continuous.function(xarr, yarr)
```

Arguments

xarr	a vector of x (sorted)
yarr	a vector of y

Value

interpolated lambda function

---

negativity — *Calculate the negativity given the calculate Schmidt modes.*

Description

Calculate the negativity given the calculate Schmidt modes.

Usage

```
negativity(modes)
```

Arguments

modes	Schmidt modes

Value

negativity

Examples

```
singlet<- matrix(c(0, sqrt(0.7), sqrt(0.3), 0), byrow = TRUE, nrow = 2)
modes<- schmidt.decompose(singlet)
negativity(modes)
```

---

participation.ratio — *Calculate the participation ratio given the calculate Schmidt modes.*

Description

Calculate the participation ratio given the calculate Schmidt modes.

Usage

```
participation.ratio(modes)
```

Arguments

modes	Schmidt modes

Value

participation ratio

Examples

```
singlet<- matrix(c(0, sqrt(0.7), sqrt(0.3), 0), byrow = TRUE, nrow = 2)
modes<- schmidt.decompose(singlet)
participation.ratio(modes)
```

---

reduced.denmat *Get reduced density matrix*

Description

Get reduced density matrix

Usage

```
reduced.denmat(bipartite.qubits, keep.dim = 1)
```

Arguments

`bipartite.qubits`	tensor of bipartite systems
`keep.dim`	dimension to keep (default: 1)

Value

reduced density matrix

Examples

```
singlet<- matrix(c(0, sqrt(0.7), sqrt(0.3), 0), byrow = TRUE, nrow = 2)
reduced.denmat(singlet)
```

---

schmidt.decompose *Perform Schmidt decomposition*

Description

Perform Schmidt decomposition

Usage

```
schmidt.decompose(bipartite.qubits)
```

Arguments

`bipartite.qubits`
: tensor of bipartite systems

Value

Schmidt modes, including the eigenvalues, and eigenvectors of both subsystems of the modes

Examples

```
singlet<- matrix(c(0, sqrt(0.7), sqrt(0.3), 0), byrow = TRUE, nrow = 2)
schmidt.decompose(singlet)
```

Index

continuous.function.interpolate, 2
continuous.schmidt.decompose, 2

discretize.continuous.bipartitefunc, 3

entanglement.entropy, 4

interpolated.continuous.function, 4

negativity, 5

participation.ratio, 5

reduced.denmat, 6

schmidt.decompose, 6