Package ‘arqas’

September 16, 2015

Type Package

Title Application in R for Queueing Analysis and Simulation

Version 1.3

Date 2015-09-06

Author Borja Varela

Maintainer Borja Varela <borja.varela.brea@gmail.com>

Depends R (>= 1.8.0), distr, ggplot2

Imports methods, reshape, iterators, doParallel, foreach, fitdistrplus, grid, gridExtra

Description Provides functions to compute the main characteristics of the following queueing models: M/M/1, M/M/s, M/M/1/k, M/M/s/k, M/M/1/Inf/H, M/M/s/Inf/H, M/M/s/Inf/H with Y replacements, M/M/Inf, Open Jackson Networks and Closed Jackson Networks. Moreover, it is also possible to simulate similar queueing models with any type of arrival or service distribution: G/G/1, G/G/s, G/G/1/k, G/G/s/k, G/G/1/Inf/H, G/G/s/Inf/H, G/G/s/Inf/H with Y replacements, Open Networks and Closed Networks. Finally, contains functions for fit data to a statistic distribution.

License GPL (>= 2)

LazyData yes

R topics documented:

`cdfcompggplot2` *Cumulative Density plot using the package ggplot2*

Description

Cumulative Density plot using the package ggplot2

Usage

```
cdfcompggplot2(lfitdata)
```

Arguments

`lfitdata`	a list of fitted data

See Also

Other DistributionAnalysis: `denscompggplot2`; `fitData`; `goodnessFit`; `qqcompggplot2`; `summaryFit`

ClosedJacksonNetwork

Obtains the main characteristics of a Closed Jackson Network model

Description

Obtains the main characteristics of a Closed Jackson Network model

Usage

```
ClosedJacksonNetwork(mu = c(5, 5, 10, 15), s = c(2, 2, 1, 1),
  p = matrix(c(0.25, 0.15, 0.2, 0.4, 0.15, 0.35, 0.2, 0.3, 0.5, 0.25, 0.15,
  0.1, 0.4, 0.3, 0.25, 0.05), 4, byrow = TRUE), n = 3)
```

Arguments

mu	Vector of mean service rates
s	Vector of servers at each node
p	Routing matrix, where p_{ij} is the routing probability from node i to node j
n	Number of customers in the network

Value

Returns the next information of a Closed Jackson Network model:

rho	Traffic intensity: ρ
l	Number of customers in the system: L
lq	Number of customers in the queue: L_q
w	Waiting time in the system: W
wq	Waiting time in the queue: W_q
eff	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: M_M_1_INF_H; M_M_1_K; M_M_1; M_M_INF; M_M_S_INF_H_Y; M_M_S_INF_H; M_M_S_K; M_M_S; OpenJacksonNetwork

Examples

```
# A system is composed of 4 workstations
#interconnected. The monitoring of the
#system is carried out by three programs
#in continuous execution in some of the
#workstations. Once each program ends,
#it creates a copy of itself and sends this
#copy to be executed in any of the
#workstations, taking into account the
#following probabilities:

# Origin-destiny     1      2     3     4
```

```
#      1           0.25   0.15  0.20  0.40
#      2           0.15   0.35  0.20  0.30
#      3           0.50   0.25  0.15  0.10
#      4           0.40   0.30  0.25  0.05

#The servers 1 and 2 have two processors and
#each of one has a processing time following an
#exponential distribution and capacity of 5
#tasks per minute.
#The servers 3 and 4 have a single processor
#and they can serve 10 and 15 task per minute
#respectively.

ClosedJacksonNetwork(mu=c(5,5,10,15),
                     s=c(2,2,1,1),
                     p=matrix(c(0.25, 0.15, 0.20, 0.40,
                                0.15, 0.35, 0.20, 0.30,
                                0.50, 0.25, 0.15, 0.10,
                                0.40, 0.30, 0.25, 0.05), 4, byrow = TRUE),
                     n = 3)
```

`ClosedNetwork` *Obtains the main characteristics of a Closed Network model by simulation*

Description

Obtains the main characteristics of a Closed Network model by simulation

Usage

```
ClosedNetwork(serviceDistribution = c(Exp(5), Exp(5), Exp(10), Exp(15)),
  s = c(2, 2, 1, 1), p = array(c(0.25, 0.15, 0.5, 0.4, 0.15, 0.35, 0.25,
  0.3, 0.2, 0.2, 0.15, 0.25, 0.4, 0.3, 0.1, 0.05), dim = c(4, 4)),
  staClients = 100, nClients = 3, transitions = 1000, historic = FALSE,
  nsim = 10, nproc = 1)
```

Arguments

`serviceDistribution`	Service distributions for the nodes of the network (Each element must be an object of S4-class `distr` defined in **distr** package)
`s`	Vector of servers at each node
`p`	Routing matrix, where p_{ij} is the routing probability from node i to node j
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers in the system
`transitions`	Number of transitions between nodes used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a Closed Network model:

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Vector of empirical number of customers in the nodes: L
`lq`	Vector of empirical number of customers in the queues of the nodes: L_q
`lqt`	Empirical number of customers in the all the queues: L_{qTotal}
`w`	Vector of empirical waiting times in the nodes: W
`wq`	Vector of empirical waiting times in the queues of the nodes: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation.

See Also

Other SimulatedModels: `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
ClosedNetwork(serviceDistribution = c(Exp(5), Exp(5), Exp(10), Exp(15)),
              s                   = c(2,2,1,1),
              p                   = matrix(c(0.25, 0.15, 0.2,  0.4,
                                             0.15, 0.35, 0.2,  0.3,
                                             0.5,  0.25, 0.15, 0.1,
                                             0.4,  0.3,  0.25, 0.05), 4, byrow=TRUE),
              nClient             = 3,
              staClients          = 10,
              transitions         = 100,
              nsim                = 10)
```

`combineSimulations` *Combines a list of independent simulations of a queueing model, computing the mean and variance of each characteristic of interest*

Description

Combines a list of independent simulations of a queueing model, computing the mean and variance of each characteristic of interest

Usage

```
combineSimulations(listsims)
```

Arguments

`listsims`	A list of independent simulations

Value

an object with the mean and estimated precision of estimated parameters L, Lq, W, Wq, Rho and Eff.

Examples

```
combineSimulations(G_G_1(nsim=5))
```

---

denscompggplot2 *Density histogram Plot using the package ggplot2*

Description

Density histogram Plot using the package ggplot2

Usage

```
denscompggplot2(lfitdata)
```

Arguments

`lfitdata`	a list of fitted data

See Also

Other DistributionAnalysis: `cdfcompggplot2`; `fitData`; `goodnessFit`; `qqcompggplot2`; `summaryFit`

---

fitData *Computes the estimated parameters of the distributions for the input data*

Description

Computes the estimated parameters of the distributions for the input data

Usage

```
fitData(data, ldistr = c("exp", "norm", "weibull", "unif", "lnorm", "gamma",
  "beta"))
```

Arguments

`data`	data to estimate parameters
`ldistr`	A list of distributions

Value

A list of estimated parameters for each distribution

See Also

Other DistributionAnalysis: `cdfcompggplot2`; `denscompggplot2`; `goodnessFit`; `qqcompggplot2`; `summaryFit`

Examples

```
mydata <- rnorm(100, 10, 0.5)

fitData(mydata)
```

`FW` *Distribution function of the waiting time in the system*

Description

Returns the value of the cumulative distribution function of the waiting time in the system for a queueing model

$$W(x) = P(W \leq x)$$

Usage

```
FW(qm, x)
```

Arguments

qm	Queueing model
x	Time

Value

$$W(x)$$

Methods (by class)

- `MarkovianModel`: Implements the default method (generates a message)
- `M_M_1`: Implements the method for a M/M/1 queueing model
- `M_M_S`: Implements the method for a M/M/S queueing model
- `M_M_1_K`: Implements the method for a M/M/1/K queueing model
- `M_M_S_K`: Implements the method for a M/M/S/K queueing model
- `M_M_1_INF_H`: Implements the method for a M/M/1/∞/H queueing model
- `M_M_S_INF_H`: Implements the method for a M/M/s/∞/H queueing model
- `M_M_S_INF_H_Y`: Implements the method for a M/M/s/∞/H with Y replacements queueing model
- `M_M_INF`: Implements the method for a M/M/∞ queueing model

Examples

```
#Cumulative probability of waiting 1 units
#of time in the system
FW(M_M_1(), 1)
FW(M_M_S_K(), 1)

#You can also get multiple probabilities
#at once
FW(M_M_1_INF_H(), c(0, 0.25,0.8))
FW(M_M_INF(), c(0, 0.25, 0.8))
```

`FWq` *Distribution function of the waiting time in the queue*

Description

Returns the value of the cumulative distribution function of the waiting time in the queue

$$W_q = P(W_q \leq x)$$

Usage

```
FWq(qm, x)
```

Arguments

`qm`	Queueing model
`x`	Time

Value

$$W_q(x)$$

Methods (by class)

- `MarkovianModel`: Implements the default method (generates a message)
- `M_M_1`: Implements the method for a M/M/1 queueing model
- `M_M_S`: Implements the method for a M/M/S queueing model
- `M_M_1_K`: Implements the method for a M/M/1/K queueing model
- `M_M_S_K`: Implements the method for a M/M/S/K queueing model
- `M_M_1_INF_H`: Implements the method for a M/M/1/∞/H queueing model
- `M_M_S_INF_H`: Implements the method for a M/M/s/∞/H queueing model
- `M_M_S_INF_H_Y`: Implements the method for a M/M/s/∞/H with Y replacements queueing model
- `M_M_INF`: Implements the method for a M/M/∞ queueing model

Examples

```
#Cumulative probability of waiting 1 unit
 #of time in the system
 FWq(M_M_1(), 1)
 FWq(M_M_S_K(), 1)

 #You can also get multiple probabilities
 #at once
 FWq(M_M_1_INF_H(), c(0, 0.25,0.8))
 FWq(M_M_INF(), c(0, 0.25, 0.8))
```

`goodnessFit` *Computes the p-value of the chi-square test and Kolmogorov-Smirnov test*

Description

Computes the p-value of the chi-square test and Kolmogorov-Smirnov test

Usage

```
goodnessFit(lfitdata)
```

Arguments

`lfitdata`	a list of fitted data

Value

the p-values and the values of the statistics in the chi-square test and Kolmogorov-Smirnov test

See Also

Other DistributionAnalysis: `cdfcompggplot2`; `denscompggplot2`; `fitData`; `qqcompggplot2`; `summaryFit`

Examples

```
mydata <- rnorm(100, 10, 0.5)

goodnessFit(fitData(mydata))
```

G_G_1 *Obtains the main characteristics of a G/G/1 model by simulation*

Description

Obtains the main characteristics of a G/G/1 model by simulation

Usage

```
G_G_1(arrivalDistribution = Exp(3), serviceDistribution = Exp(6),
  staClients = 100, nClients = 1000, historic = FALSE, nsim = 10,
  nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter used to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/1 model:

`pn`	Stores all the empirical steady-state probabilities of having n customers, with n from 0 to staClients+nClients: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q,*Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_1(Norm(10, 0.5), Unif(5,6), staClients=10, nClients=100, nsim=10)
```

`G_G_1_INF_H` *Obtains the main characteristics of a G/G/1/∞/H model by simulation*

Description

Obtains the main characteristics of a G/G/1/∞/H model by simulation

Usage

```
G_G_1_INF_H(arrivalDistribution = Exp(3), serviceDistribution = Exp(6),
  H = 5, staClients = 100, nClients = 1000, historic = FALSE,
  nsim = 10, nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`H`	Population size
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/1/∞/H model:

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_1_INF_H(Norm(10, 0.5), Unif(5,6), 10, staClients=10, nClients=100, nsim=10)
```

G_G_1_K *Obtains the main characteristics of a G/G/1/K model by simulation*

Description

Obtains the main characteristics of a G/G/1/K model by simulation

Usage

```
G_G_1_K(arrivalDistribution = Exp(3), serviceDistribution = Exp(6), K = 2,
  staClients = 100, nClients = 1000, historic = FALSE, nsim = 10,
  nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`K`	Maximun size of the queue
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/1/K model:

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation.

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_1_K(Norm(10, 0.5), Unif(5,6), 5, staClients=10, nClients=100, nsim=10)
```

G_G_INF — *Obtains the main characteristics of a G/G/∞ model by simulation*

Description

Obtains the main characteristics of a G/G/∞ model by simulation

Usage

```
G_G_INF(arrivalDistribution = Exp(3), serviceDistribution = Exp(6),
  staClients = 100, nClients = 1000, historic = FALSE, nsim = 10,
  nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`staClients`	Number of customers used in stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/∞ model:

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_INF(Norm(10, 0.5), Unif(2,4), staClients=50, nClients=100, nsim=10)
```

G_G_S *Obtains the main characteristics of a G/G/s model by simulation*

Description

Obtains the main characteristics of a G/G/s model by simulation

Usage

```
G_G_S(arrivalDistribution = Exp(3), serviceDistribution = Exp(6), s = 2,
  staClients = 100, nClients = 1000, historic = FALSE, nsim = 10,
  nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`s`	Number of servers
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter used to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/S model:

`pn`	vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical Traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `OpenNetwork`

Examples

```
G_G_S(Norm(10, 0.5), Unif(5,6), 2, staClients=10, nClients=100, nsim=10)
```

`G_G_S_INF_H` *Obtains the main characteristics of a G/G/S/∞/H model by simulation*

Description

Obtains the main characteristics of a G/G/S/∞/H model by simulation

Usage

```
G_G_S_INF_H(arrivalDistribution = Exp(3), serviceDistribution = Exp(6),
  s = 3, H = 5, staClients = 100, nClients = 1000, historic = FALSE,
  nsim = 10, nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`s`	Number of servers
`H`	Population size
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/S/∞/H model

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system:L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_S_INF_H(Norm(10, 0.5), Unif(5,6), 3, 10, staClients=10, nClients=100, nsim=10)
```

G_G_S_INF_H_Y *Obtains the main characteristics of a G/G/S/∞/H with Y replacements model by simulation*

Description

Obtains the main characteristics of a G/G/S/∞/H with Y replacements model by simulation

Usage

```
G_G_S_INF_H_Y(arrivalDistribution = Exp(3), serviceDistribution = Exp(6),
  s = 3, H = 5, Y = 3, staClients = 100, nClients = 1000,
  historic = FALSE, nsim = 10, nproc = 1)
```

Arguments

`arrivalDistribution`	Arrival distribution (object of S4-class `distr` defined in **distr** package)
`serviceDistribution`	Service distribution (object of S4-class `distr` defined in **distr** package)
`s`	Number of servers
`H`	Population size
`Y`	Number of replacements
`staClients`	Number of customers used in the stabilization stage
`nClients`	Number of customers used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of a G/G/1/S/∞/H/Y model:

`pn`	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
`l`	Empirical number of customers in the system: L
`lq`	Empirical number of customers in the queue: L_q
`w`	Empirical waiting time in the system: W
`wq`	Empirical waiting time in the queue: W_q
`eff`	Empirical system efficiency: $Eff = W/(W - W_q)$
`rho`	Empirical traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_S_INF_H_Y(Norm(10, 0.5), Unif(5,6), 3, 10, 2, staClients=10, nClients=100, nsim=10)
```

G_G_S_K *Obtains the main characteristics of a G/G/s/K model by simulation*

Description

Obtains the main characteristics of a G/G/s/K model by simulation

Usage

```
G_G_S_K(arrivalDistribution = Exp(3), serviceDistribution = Exp(6), s = 2,
  K = 3, staClients = 100, nClients = 1000, historic = FALSE,
  nsim = 10, nproc = 1)
```

Arguments

arrivalDistribution	Arrival distribution (object of S4-class distr defined in **distr** package)
serviceDistribution	Service distribution (object of S4-class distr defined in **distr** package)
s	Number of servers
K	Maximun size of the queue
staClients	Number of customers used in the stabilization stage
nClients	Number of customers used in the simulation stage
historic	Parameter to activate/deactivate the historic information
nsim	Number of simulations
nproc	Processors used in the simulation.

Value

Returns the next information of a G/G/S/K model:

pn	Vector of empirical steady-state probabilities of having n customers in the system: P_n (Only the probabilities bigger than 0 are included)
l	Empirical number of customers in the system: L
lq	Empirical number of customers in the queue: L_q
w	Empirical waiting time in the system: W
wq	Empirical waiting time in the queue: W_q
eff	Empirical system efficiency: $Eff = W/(W - W_q)$
rho	Empirical traffic intensity: ρ
historic	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S`; `OpenNetwork`

Examples

```
G_G_S_K(Norm(10, 0.5), Unif(5,6), 2, 5, staClients=10, nClients=100, nsim=10)
```

`MarkovianModel` *Defines a queueing model*

Description

Constructor for `MarkovianModel` class.

Usage

```
MarkovianModel(arrivalDistribution = Exp(1), serviceDistribution = Exp(1))
```

Arguments

`arrivalDistribution`
: Arrival distribution (object of S4-class `distr` defined in **distr** package)

`serviceDistribution`
: Service distribution (object of S4-class `distr` defined in **distr** package)

Value

An object of class `MarkovianModel`, a list with the following components:

`arrivalDistribution`
: Arrival distribution (object of S4-class `distr` defined in **distr** package)

`serviceDistribution`
: Service distribution (object of S4-class `distr` defined in **distr** package)

`maxCustomers` *Returns the maximun value of n that satisfies the condition P_n > 0*

Description

Returns the maximun value of n that satisfies the condition $P_n > 0$

Usage

```
maxCustomers(qm)

## S3 method for class 'M_M_S_INF_H'
maxCustomers(qm)
```

Arguments

`qm`	object MarkovianModel

Details

`maxCustomers.M_M_S_INF_H` implements the method for a M/M/s/∞/H queueing model

Methods (by class)

- `MarkovianModel`: implements the default method. Returns infinite.
- `M_M_1_K`: Implements the method for a M/M/1/K queueing model
- `M_M_S_K`: Implements the method for a M/M/S/K queueing model
- `M_M_1_INF_H`: Implements the method for a M/M/1/∞/H queueing model
- `M_M_S_INF_H_Y`: Implements the method for a M/M/s/∞/H/Y queueing model

Examples

```
maxCustomers(M_M_1_K())

maxCustomers(M_M_S_INF_H_Y())
maxCustomers(M_M_S_INF_H(lambda=1/2, mu=60/5, s=2, h=5))
```

`M_M_1` *Obtains the main characteristics of a M/M/1 queueing model*

Description

Obtains the main characteristics of a M/M/1 queueing model

Usage

```
M_M_1(lambda = 3, mu = 6)
```

Arguments

`lambda`	Mean arrival rate
`mu`	Mean service rate

Value

Returns the next information of a M/M/1 model:

`rho`	Traffic intensity: ρ
`cn`	Coefficients used in the computation of P_n: C_n
`p0`	Probability of empty system: P_0
`l`	Number of customers in the system: L
`lq`	Number of customers in the queue: L_q
`w`	Waiting time in the system: W
`wq`	Waiting time in the queue: W_q
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: ClosedJacksonNetwork; M_M_1_INF_H; M_M_1_K; M_M_INF; M_M_S_INF_H_Y; M_M_S_INF_H; M_M_S_K; M_M_S; OpenJacksonNetwork

Examples

```
#A workstation with a single processor
#runs programs with CPU time following
#an exponential distribution with mean 3 minutes.
#The programs arrive to the workstation following
#a Poisson process with an intensity of 15
#programs per hour.

M_M_1(lambda=15, mu=60/3)
```

M_M_1_INF_H *Obtains the main characteristics of a M/M/1/∞/H queueing model*

Description

Obtains the main characteristics of a M/M/1/∞/H queueing model

Usage

```
M_M_1_INF_H(lambda = 1/2, mu = 60/5, h = 5)
```

Arguments

lambda	Mean arrival rate
mu	Mean service rate
h	Population size

Value

Returns the next information of a M/M/1/∞/H model :

rho	Constant: λ/μ
barrho	Traffic intensity: $\bar{\rho}$
barlambda	Mean effective arrival rate: $\bar{\rho}$
cn	Coefficients used in the computation of P_n: C_n
p0	Probability of empty system: P_0
l	Number of customers in the system: L
lq	Number of customers in the queue: L_q
w	Waiting time in the system: W
wq	Waiting time in the queue: W_q
eff	System Efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: ClosedJacksonNetwork; M_M_1_K; M_M_1; M_M_INF; M_M_S_INF_H_Y; M_M_S_INF_H; M_M_S_K; M_M_S; OpenJacksonNetwork

Examples

```
# A computer system with five workstations must
# make a back- up from time to time, preventing
# users from using the system.
# The time between ending a back -up until the
# next begins is random and follows an
# exponential distribution with mean 2 hours.
# The duration of the backup is also random and
# follows an exponential distribution with mean
# 5 minutes.
# There exists a single tape drive to perform the
# back-up process and the station will wait if it is
# busy.

M_M_1_INF_H(lambda =1/2, mu=60/5, h=5)
```

M_M_1_K *Obtains the main characteristics of a M/M/1/K queueing model*

Description

Obtains the main characteristics of a M/M/1/K queueing model

Usage

```
M_M_1_K(lambda = 3, mu = 6, k = 2)
```

Arguments

lambda	Mean arrival rate
mu	Mean service rate
k	Maximun size of the queue

Value

Returns the next information of a M/M/1/K model:

rho	Constant coefficient: λ/ρ
barrho	Traffic intensity: $\bar{\rho}$
barlambda	Effective arrival rate: $\bar{\lambda}$
l	Mean number of customers in the system: L
lq	Mean number of customers in the queue: L_q
w	Waiting time in the system: W
wq	Waiting time in the queue: W_q
eff	Efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H_Y`; `M_M_S_INF_H`; `M_M_S_K`; `M_M_S`; `OpenJacksonNetwork`

Examples

```
#A workstation with a single processor
#runs programs with CPU time following
#an exponential distribution with mean 3 minutes.
#The programs arrive to the workstation following
#a Poisson process with an intensity of 15
#programs per hour.
#The workstation has a limited memory and only
#one program is allowed to wait if the processor
#is busy.

M_M_1_K(lambda=15, mu=60/3, k=1)
```

M_M_INF — *Obtains the main characteristics of a M/M/∞ queueing model*

Description

Obtains the main characteristics of a M/M/∞ queueing model

Usage

```
M_M_INF(lambda = 3, mu = 6)
```

Arguments

`lambda`	Mean arrival rate
`mu`	Mean service rate

Value

Returns the next information of a M/M/∞ model:

`rho`	Constant coefficient: λ/μ
`barrho`	Traffic intensity: $\bar{\rho}$
`p0`	Probability of empty system: P_0
`l`	Number of customers in the system: L
`lq`	Number of customers in the queue: L_q (L_q = 0 in this model)
`w`	Waiting time in the system: W
`wq`	Waiting time in the queue: W_q (W_q = 0 in this model)
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_S_INF_H_Y`; `M_M_S_INF_H`; `M_M_S_K`; `M_M_S`; `OpenJacksonNetwork`

Examples

```
#The number of people turning on their television sets
#on Saturday evening during prime time can be described
#rather well by a Poisson distribution with a mean of
#100000/hr.
#There are five major TV stations, and a given person
#chooses among them essentially at random.
#Surveys have also shown that the average person tunes
#in for 90 min and that viewing times are approximately
#exponentially distributed.
M_M_INF(lambda=100000/5, mu=60/90)
```

`M_M_S` *Obtains the main characteristics of a M/M/s queueing model*

Description

Obtains the main characteristics of a M/M/s queueing model

Usage

```
M_M_S(lambda = 3, mu = 6, s = 2)
```

Arguments

`lambda`	Mean arrival rate
`mu`	Mean service rate
`s`	Number of servers

Value

Returns the next information of a M/M/s model:

`rho`	Traffic intensity: ρ
`cn`	Coefficients used in the computation of P_n: C_n
`p0`	Probability of empty system: P_0
`l`	Number of customers in the system: L
`lq`	Number of customers in the queue: L_q
`w`	Waiting time in the system: W
`wq`	Waiting time in the queue: W_q
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H_Y`; `M_M_S_INF_H`; `M_M_S_K`; `OpenJacksonNetwork`

Examples

```
#A workstation with three processors
#runs programs with CPU time following
#an exponential distribution with mean 3 minutes.
#The programs arrives to the workstation following
#a Poisson process with an intensity of 15
#programs per hour.

M_M_S(lambda=15, mu=60/3, s=3)
```

M_M_S_INF_H *Obtains the main characteristics of a M/M/s/∞/H queueing model*

Description

Obtains the main characteristics of a M/M/s/∞/H queueing model

Usage

```
M_M_S_INF_H(lambda = 1/2, mu = 60/5, s = 2, h = 5)
```

Arguments

lambda	Mean arrival rate
mu	Mean service rate
s	Number of servers
h	Population size

Value

Returns the next information of a M/M/s/∞/H model:

rho	Constant coefficient: λ/μ
barrho	Traffic intensity: $\bar{\rho}$
barlambda	Mean effective arrival rate: $\bar{\rho}$
cn	Coefficients used in the computation of P_n: C_n
p0	Probability of empty system: P_0
l	Number of customers in the system: L
lq	Number of customers in the queue: L_q
w	Waiting time in the system: W
wq	Waiting time in the queue: W_q
eff	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H_Y`; `M_M_S_K`; `M_M_S`; `OpenJacksonNetwork`

Examples

```
# A computer system with five workstations must
# make a back- up from time to time, preventing
# users from using the system.
# The time between ending a back -up until the
# next begins is random and follows an
# exponential distribution with mean 2 hours.
# The duration of the backup is also random and
# follows an exponential distribution with mean
# 5 minutes.
# There are 2 tape drives to perform the
# back-up process, the station remained on hold
# if both were occupied.

M_M_S_INF_H(lambda=1/2, mu=60/5, s=2, h=5)
```

`M_M_S_INF_H_Y` *Obtains the main characteristics of a M/M/s/∞/H with Y replacements queueing model*

Description

Obtains the main characteristics of a M/M/s/∞/H with Y replacements queueing model

Usage

```
M_M_S_INF_H_Y(lambda = 3, mu = 6, s = 3, h = 5, y = 3)
```

Arguments

`lambda`	Mean arrival rate
`mu`	Mean service rate
`s`	Number of servers
`h`	Population size
`y`	Number of replacements

Value

Returns the next information of a M/M/s/∞/H/Y model:

`rho`	Constant coefficient: λ/ρ
`barrho`	Traffic intensity: $\bar{\rho}$
`barlambda`	Effective arrival rate: $\bar{\lambda}$
`cn`	Coefficients used in the computation of P_n: C_n
`p0`	Probability of empty system: P_0
`l`	Number of customers in the system: L
`lq`	Number of customers in the queue: L_q
`w`	Waiting time in the system: W
`wq`	Waiting time in the queue: W_q
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H`; `M_M_S_K`; `M_M_S`; `OpenJacksonNetwork`

Examples

```
#A bank has 5 ATMs. Occasionally one ot them is
#damaged until one of the two hired technicians
#repairs it. It is known that the mean time to repair
#each ATM follows an exponential distribution with mean
#10 minutes, while the distribution of time an ATM
#works is also exponential
#with mean 2 hours. The bank has an ATM extra to
#replace a damaged one.

M_M_S_INF_H_Y(lambda=1/2, mu=60/10, s=2, h=5, y=1)
```

`M_M_S_K` — *Obtains the main characteristics of a M/M/S/k queueing model*

Description

Obtains the main characteristics of a M/M/S/k queueing model

Usage

```
M_M_S_K(lambda = 3, mu = 6, s = 2, k = 3)
```

Arguments

`lambda`	Mean arrival rate
`mu`	Mean service rate
`s`	Number of servers
`k`	Maximun size of the queue

Value

Returns the next information of a M/M/S/K model:

`rho`	Constant coefficient: λ/ρ
`barrho`	Traffic intensity: $\bar{\rho}$
`barlambda`	Effective arrival rate: $\bar{\lambda}$
`cn`	Coefficients used in the computation of P_n: C_n
`pks`	Probability of having $K + s$ customers in the system: P_{K+s}
`p0`	Probability of empty system: P_0
`l`	Number of customers in the system: L
`lq`	Number of customers in the queue: L_q
`w`	Waiting time in the system: W
`wq`	Waiting time in the queue: W_q
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H_Y`; `M_M_S_INF_H`; `M_M_S`; `OpenJacksonNetwork`

Examples

```
#A workstation with three processors
#runs programs with CPU time following
#an exponential distribution with mean 3 minutes.
#The programs arrive to the workstation following
#a Poisson process with an intensity of 15
#programs per hour.
#The workstation has a limited memory and only
#one program is allowed to wait if the processor
#is busy.

M_M_S_K(lambda=15, mu=60/3, s=3, k=1)
```

`no_distr` *Defines an empty object representing the inexistence of a distribution.*

Description

Defines an empty object representing the inexistence of a distribution.

Usage

```
no_distr()
```

`OpenJacksonNetwork` *Obtains the main characteristics of an Open Jackson network model*

Description

Obtains the main characteristics of an Open Jackson network model

Usage

```
OpenJacksonNetwork(lambda = c(20, 30), mu = c(100, 25), s = c(1, 2),
  p = matrix(c(0.2, 0.25, 0.1, 0), nrow = 2, ncol = 2))
```

Arguments

`lambda`	Vector of arrival rates at each node
`mu`	Vector of mean service rates
`s`	Vector with the number of servers at each node
`p`	Routing matrix, where p_{ij} is the routing probability from node i to node j

Value

Returns the next information of an Open Jackson network model:

`rho`	Traffic intensity: ρ
`l`	Vector with the number of customers in the nodes: L
`lq`	Vector with the number of customers in the queue at each node: L_q
`w`	Vector with the waiting time in each node: W
`wq`	Vector with the waiting time in the queue at each node: W_q
`lt`	Number of customers in the network: L_{Total}
`lqt`	Number of customers in all the queues: L_{qTotal}
`wt`	Total waiting time in the network: W_{Total}
`wqt`	Total waiting time in all the queues: W_{qTotal}
`eff`	System efficiency: $Eff = W/(W - W_q)$

See Also

Other AnaliticalModels: `ClosedJacksonNetwork`; `M_M_1_INF_H`; `M_M_1_K`; `M_M_1`; `M_M_INF`; `M_M_S_INF_H_Y`; `M_M_S_INF_H`; `M_M_S_K`; `M_M_S`

Examples

```
#Two servers receive a number of tasks per minute;
#20 tasks per minute in the case of the first
#server and 30 tasks per minute in the second one.
#The unique processsor in the first server can manage
#100 tasks per minute, while the two processors in the
#second server only can manage 25 task per minute.
#When a task is finishing in the server 2, it creates
#a new task in the server 1 with a probability of 25%,
#the task ends in the other case.
#The task that ends in the server 1 creates a new one
#in the same server the 20% of the times and creates
#a new one in the server 2 the 10% of the times, ending
#in other case.

OpenJacksonNetwork(lambda=c(20, 30),
                   mu=c(100, 25),
                   s=c(1,2),
                   p=matrix(c(0.2,0.1,
                              0.25,0), 2, byrow = TRUE))
```

`OpenNetwork` *Obtains the main characteristics of an Open Network model by simulation*

Description

Obtains the main characteristics of an Open Network model by simulation

Usage

```
OpenNetwork(arrivalDistribution = c(Exp(20), Exp(30)),
  serviceDistribution = c(Exp(100), Exp(25)), s = c(1, 2),
  p = matrix(c(0.2, 0.25, 0.1, 0), nrow = 2, ncol = 2), staClients = 100,
  transitions = 1000, historic = FALSE, nsim = 10, nproc = 1)
```

Arguments

`arrivalDistribution`	Vector indicating the arrival distribution at each node (Each element must be an object of S4-class `distr` defined in **distr** package or the no_distr() object)
`serviceDistribution`	Vector indicating the service distribution at each node (Each element must be an object of S4-class `distr` defined in **distr** package)
`s`	Vector of servers in each node
`p`	Routing matrix, where p_{ij} is the routing probability from node i to node j
`staClients`	Number of customers used in the stabilization stage
`transitions`	Number of transitions between nodes used in the simulation stage
`historic`	Parameter to activate/deactivate the historic information
`nsim`	Number of simulations
`nproc`	Processors used in the simulation.

Value

Returns the next information of an Open network model:

`pn`	Vector of steady-state probabilities of having n customers in the system: P_n
`l`	Vector of expected number of customers in the nodes: L
`lq`	Vector of expected number of customers in the queues of the nodes: L_q
`lqt`	Expected number of customers in all the queues: L_{qTotal}
`w`	Vector of expected waiting times in the nodes: W
`wq`	Vector of expected waiting time in the queues of the nodes: W_q
`eff`	System efficiency: $Eff = W/(W - W_q)$
`rho`	Traffic intensity: ρ
`historic`	Optional parameter that stores the evolution of L, L_q, W and W_q, *Customers in the system, Rho and Elapsed time* during the simulation.

See Also

Other SimulatedModels: `ClosedNetwork`; `G_G_1_INF_H`; `G_G_1_K`; `G_G_1`; `G_G_INF`; `G_G_S_INF_H_Y`; `G_G_S_INF_H`; `G_G_S_K`; `G_G_S`

Examples

```
OpenNetwork(arrivalDistribution = c(Exp(20), no_distr()),
            serviceDistribution = c(Exp(100), Exp(25)),
            s                   = c(1,2),
            p                   = matrix(c(0.2, 0.25, 0.1, 0), nrow=2, ncol=2),
            staClients          = 10,
            transitions         = 100,
            nsim                = 10)
```

P0i	*Steady-state probability of 0 customers in the system on the node i of an Open Jackson Network.*

Description

Returns the value of the probability of having 0 customers at node i of an Open Jackson Network.

Usage

```
P0i(net, i)

## S3 method for class 'OpenJackson'
P0i(net, i)
```

Arguments

`net`	Network
`i`	Node. Index starts in 1.

Details

`P0i.OpenJackson` implements the method for an Open Jackson Network model

Value

$$P_{0,i}$$

Examples

```
#Probability of having 0 customers on the node 2
P0i(OpenJacksonNetwork(), 2)
```

Pi	*Steady-state probability of n customers at node i of a network.*

Description

Returns the value $P_i(n)$ in the node i of a Closed Jackson Network

Usage

```
Pi(net, n, node)

## S3 method for class 'ClosedJackson'
Pi(net, n, node)

## S3 method for class 'SimulatedNetwork'
Pi(net, n, node)
```

Arguments

`net`	Closed Jackson Network
`n`	Customers
`node`	Node

Details

`Pi.ClosedJackson` implements the method for a Closed Jackson Network model

`Pi.SimulatedNetwork` implements the method for a SimulatedNetwork model

Value

P_n in the selected node

Examples

```
#Probability of having 0 customers on node 2
Pi(ClosedJacksonNetwork(), 0, 2)

#It is possible obtain multiple probabilities
#for a node at once.
Pi(ClosedJacksonNetwork(), 0:2, 2)
```

`plot.MarkovianModel` *Shows the main graphics of the parameters of a Markovian Model*

Description

Shows the main graphics of the parameters of a Markovian Model

Usage

```
## S3 method for class 'MarkovianModel'
plot(x, t = list(range = seq(x$out$w, x$out$w * 3,
  length.out = 100)), n = c(0:5), only = NULL, graphics = "ggplot2", ...)
```

Arguments

`x`	Markovian Model
`t`	range for drawing the waiting plots
`n`	range for drawing the probabilities plot
`only`	Allow to only show the waiting plots or the probabilites plots. Must be NULL, "t" or "n"
`graphics`	library used to draw the plots
`...`	Further arguments

Details

`plot.MarkovianModel` implements the function for an object of class MarkovianModel.

`plot.SimulatedModel` *#Shows a plot of the evolution of a variable during the simulation # #*

Description

#Shows a plot of the evolution of a variable during the simulation # #

Usage

```
## S3 method for class 'SimulatedModel'
plot(x, minrange = 1, maxrange, var = "L",
  graphics = "ggplot2", depth = maxrange - minrange, ...)

## S3 method for class 'SimulatedNetwork'
plot(x, minrange = 1, maxrange, var = "L",
  graphics = "ggplot2", depth = maxrange - minrange, nSimulation = NULL,
  ...)

## S3 method for class 'list'
plot(x, minrange = 1, maxrange, var = "L",
  graphics = "ggplot2", depth = maxrange - minrange + 1, nSimulation = 1,
  showMean = TRUE, showValues = TRUE, ...)
```

Arguments

`x`	Simulated Model #
`minrange`	Number of customers needed to establish the start of the plot #
`maxrange`	Number of customers needed to establish the end of the plot #
`var`	This variable indicates the parameter of the queue to show in graphic (L, Lq, W, Wq, Clients, Intensity) #
`graphics`	Type of graphics: "graphics" use the basic R plot and "ggplot2" the library ggplot2 #
`depth`	Number of points printed in the plot #
`...`	Further arguments passed to or from other methods. #
`nSimulation`	Only used when the var param is equal to "Clients". Selects one of the multiple simulations to show the evolution of the Clients. #
`showMean`	Shows the mean of all the simulations
`showValues`	Shows the values of all the simulations

Details

`plot.SimulatedModel` implements the function for an object of class SimulatedModel.

`plot.SimulatedNetwork` implements the function for an object of class SimulatedNetwork.

`plot.list` implements the function for an object of class list

Pn *Steady-state probability of having n customers in the system*

Description

Returns the probability of having n customers in the given queueing model

Usage

```
Pn(qm, n)
```

Arguments

qm	Queueing model
n	Number of customers. With `OpenJacksonNetwork` objects must be a vector with same length as nodes. With `ClosedJacksonNetwork` objects also the sum the vector must be equal to the number of customers in the network.

Value

$$P_n$$

Methods (by class)

- `MarkovianModel`: Implements the method for a Markovian model
- `M_M_1`: Implements the method for a M/M/1 queueing model
- `M_M_S`: Implements the method for a M/M/s queueing model
- `M_M_1_K`: Implements the method for a M/M/1/K queueing model
- `M_M_S_K`: Implements the method for a M/M/s/K queueing model
- `M_M_1_INF_H`: implements the method for a M/M/1/∞/H queueing model
- `M_M_S_INF_H`: Implements the method for a M/M/s/∞/H queueing model
- `M_M_S_INF_H_Y`: Implements the method for a M/M/s/∞/H/Y queueing model
- `M_M_INF`: Implements the method for a M_M_INF queueing model
- `OpenJackson`: Implements the method for a Open Jackson Network model
- `ClosedJackson`: Implements the method for a Closed Jackson Network model
- `SimulatedModel`: Implements the method for a Simulated model

Examples

```
#Probability of having one customer in the
#system
Pn(M_M_S(), 1)
Pn(M_M_INF(), 1)

#You can also get multiple probabilities
#at once
Pn(M_M_1_INF_H(), 0:5)
Pn(M_M_S_K(), 1:3)
```

```
#With networks must be a vector with
#same length as nodes

#Probability of having 0 customers in
#the node 1, and 2 customers in node 2
Pn(OpenJacksonNetwork(), c(0, 2))

#Probability of having 1,2,0, and 0
#customers in nodes 1,2,3 and 4 respectively
Pn(ClosedJacksonNetwork(), c(1,2,0,0))
```

`Qn` *Steady-state probability of finding n customers in the system when a new customer arrives*

Description

Returns the probability of having n customers in the system at the moment of the arrival of a customer.

Usage

```
Qn(qm, n)
```

Arguments

`qm`	Queueing model
`n`	Customers

Value

$$Q_n$$

Methods (by class)

- `MarkovianModel`: Implements the default method (generates a message)
- `M_M_1_K`: Implements the method for a M/M/1/K queueing model
- `M_M_S_K`: Implements the method for a M/M/S/K queueing model
- `M_M_1_INF_H`: Implements the method for a M/M/1/∞/H queueing model
- `M_M_S_INF_H`: Implements the method for a M/M/s/∞/H queueing model
- `M_M_S_INF_H_Y`: Implements the method for a M/M/s/∞/H with Y replacements queueing model

Examples

```
#Probability of having one customer in the
#queue
Qn(M_M_1_K(), 1)
Qn(M_M_S_INF_H(), 1)

#You can also get multiple probabilities
#at once
Qn(M_M_1_INF_H(), 0:5)
Qn(M_M_S_K(), 1:3)
```

`qqcompggplot2` *Q-Q Plot using the package ggplot2*

Description

Q-Q Plot using the package ggplot2

Usage

```
qqcompggplot2(lfitdata)
```

Arguments

`lfitdata`	a list of fitted data

See Also

Other DistributionAnalysis: `cdfcompggplot2`; `denscompggplot2`; `fitData`; `goodnessFit`; `summaryFit`

`summary.SimulatedModel` *Shows basic statistical information of the variable*

Description

Shows basic statistical information of the variable

Usage

```
## S3 method for class 'SimulatedModel'
summary(object, var = "l", ...)
```

Arguments

`object`	SimulatedModel
`var`	Main characteristic of an queue model
`...`	Further argumets

Details

`summary.SimulatedModel` implements the function for an object of class SimulatedModel.

`summaryFit` *Shows three plots:The histogram and theoretical densities, the empirical and theoretical CDF's and the Q-Q plot*

Description

Shows three plots:The histogram and theoretical densities, the empirical and theoretical CDF's and the Q-Q plot

Usage

```
summaryFit(lfitdata, graphics = "ggplot2", show = "all")
```

Arguments

`lfitdata`	a list of fitted data
`graphics`	Type of graphics: "graphics" uses the basic R plot and "ggplot2" the library ggplot2
`show`	Select what plots to show. Can be: "all", "dens" for the histogram, "cdf" for the CDF's or "qq" for the Q-Q-plot

See Also

Other DistributionAnalysis: `cdfcompggplot2`; `denscompggplot2`; `fitData`; `goodnessFit`; `qqcompggplot2`