Package ‘StochFracPoisson’

July 30, 2026

Title Stochastic Poisson Processes and Fractional Counting Models

Version 0.1.0

Description Implementation of advanced stochastic counting processes.
Main models include the Fractional Counting Process at Levy times
(Garg et al. (2025) <doi:10.1007/s10955-025-03515-9>), the Generalized
Iterated Poisson Process (Soni & Pathak (2024) <doi:10.1007/s10959-024-01362-0>),
the Generalized Fractional Risk Process (Soni & Pathak (2024) <doi:10.1007/s11009-024-10111-z>),
and the Tempered Space-Time Fractional Negative Binomial Process
(Garg et al. (2025) <doi:10.1007/s11009-025-10179-1>).

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports stats

Suggests testthat (>= 3.0.0)

NeedsCompilation no

Author Shikhar Tyagi [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1606-0844>),
Vrijesh Tripathi [aut]

Maintainer Shikhar Tyagi <shikhar1093tyagi@gmail.com>

Repository CRAN

Date/Publication 2026-07-30 17:40:14 UTC

Contents

alternative_gfrp 3
bell_polynomial 4
check_lrd_gipp 5
check_martingale_gfrp 5
compound_gfcp 6

cov_gipp . . . 7
cov_inverse_stable . . . 7
cov_tstfnbp . . . 8
create_fcp . . . 9
create_gipp . . . 9
create_tcfcp . . . 10
dgamma_subordinator . . . 11
dtempered_mittag_leffler_levy . . . 12
egf_fractional_bell . . . 12
fde_cgfcp . . . 13
first_passage_density_gipp . . . 14
first_passage_time_fcp . . . 15
first_passage_time_gipp . . . 16
first_passage_time_tcfcp . . . 16
first_waiting_time_fcp . . . 17
first_waiting_time_tcfcp . . . 18
fractional_bell_polynomial . . . 19
fractional_counting_process . . . 20
fractional_stirling2 . . . 21
gamma_subordinator . . . 21
generalized_fractional_risk_process . . . 22
generalized_fractional_stirling2 . . . 23
generalized_iterated_poisson . . . 23
gfcp . . . 24
incomplete_beta . . . 25
inverse_stable_subordinator . . . 25
invert_pgf . . . 26
lt_fcp . . . 26
lt_tcfcp . . . 27
lt_tempered_mittag_leffler_levy . . . 28
lt_tstfnbp . . . 29
mean_cgfcp . . . 30
mean_fcp . . . 30
mean_gfrp . . . 31
mean_gipp . . . 32
mean_inverse_stable . . . 32
mean_tcfcp . . . 33
mean_tstfnbp . . . 34
mgamma_subordinator . . . 34
mgf_fcp . . . 35
mgf_gipp . . . 36
mittag_leffler . . . 36
mittag_leffler_prabhakar . . . 37
net_profit_condition_gfrp . . . 38
partial_ordinary_bell . . . 38
pgamma_subordinator . . . 39
pgf_cgfcp . . . 40
pgf_fcp . . . 41

pgf_gipp 41
pgf_tstfnbp 42
pochhammer 43
print.fcp 43
print.gipp 44
print.tcfcp 44
ruin_probability_gfrp 45
simulate_cgfcp 46
simulate_fcp 47
simulate_gfrp 47
simulate_gipp 48
simulate_tcfcp 49
simulate_tstfnbp 50
stirling2 51
tempered_mittag_leffler_levy 52
tempered_mittag_leffler_moment 52
tempered_stable_subordinator 53
tempered_st_fnbp 54
time_changed_fcp 54
var_cgfcp 56
var_fcp 56
var_gfrp 57
var_gipp 58
var_tcfcp 58
var_tstfnbp 59
wright_function 60

Index **61**

`alternative_gfrp` *Alternative Generalized Fractional Risk Process (AGFRP)*

Description

Computes the alternative GFRP with premium function epsilon $\hat{R}^{\beta}(t) = \nu + \eta(1+\rho)\epsilon(W_{\beta}(t)) - \sum_{j=1}^{N^{\beta}(t)} X_j$

Usage

```
alternative_gfrp(
  nu,
  eta,
  rho,
  beta,
  lambda,
  t,
  epsilon = function(x) x,
  claim_dist = "gamma",
```

```
  claim_params = list(),
  n_sim = 1000
)
```

Arguments

nu	Initial capital (> 0)
eta	Mean claim size (> 0)
rho	Safety loading parameter (>= 0)
beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t	Time (> 0)
epsilon	Premium function (default = identity)
claim_dist	Distribution of claim sizes ("gamma", "exponential", "lognormal")
claim_params	Parameters for claim distribution
n_sim	Number of simulations (default = 1000)

Value

Simulated values of the alternative risk process at time t

Examples

```
alternative_gfrp(100, 10, 0.1, 0.7, c(1.0, 0.5), 1.0,
                 function(x) x^0.5, "gamma", list(shape = 2, rate = 0.2))
```

---

bell_polynomial *Classical Bell Polynomials*

Description

Computes the classical Bell polynomials B_n(x)

Usage

```
bell_polynomial(n, x, n_terms = 50)
```

Arguments

n	Polynomial order (non-negative integer)
x	Argument
n_terms	Number of terms in series expansion (default = 50)

Value

Value of the Bell polynomial

Examples

```
bell_polynomial(5, 1.0)
```

`check_lrd_gipp` *Check Long-Range Dependence of GIPP*

Description

Checks if the GIPP exhibits long-range dependence

Usage

```
check_lrd_gipp(beta, lambda, lambda_beta, s, t_max = 100)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Parameter of outer Poisson process (> 0)
`lambda_beta`	Parameter of inner fractional Poisson process (> 0)
`s`	Fixed time for correlation analysis
`t_max`	Maximum time for asymptotic analysis

Value

Logical indicating LRD property

Examples

```
check_lrd_gipp(0.7, 1.0, 0.5, 0.5, 100)
```

`check_martingale_gfrp` *Check Martingale Property of GFRP*

Description

Checks if the GFRP is a martingale, submartingale, or supermartingale

Usage

```
check_martingale_gfrp(rho)
```

Arguments

`rho`	Safety loading parameter (>= 0)

Value

String indicating martingale property

Examples

```
check_martingale_gfrp(0.1)
```

---

compound_gfcp *Compound Generalized Fractional Counting Process (CGFCP)*

Description

Computes the probability mass function of the CGFCP $C^{\beta}(t) = \sum_{j=1}^{N^{\beta}(t)} X_j$ From Soni and Pathak (2024)

Usage

```
compound_gfcp(
  beta,
  lambda,
  t,
  n,
  jump_dist = "poisson",
  jump_params = list(),
  n_terms = 50
)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t	Time (> 0)
n	Vector of non-negative integers
jump_dist	Distribution of jump sizes ("poisson", "geometric")
jump_params	Parameters for jump distribution
n_terms	Number of terms in series expansion (default = 50)

Value

Vector of probabilities P(C(t) = n)

Examples

```
compound_gfcp(0.7, c(1.0, 0.5), 1.0, 0:10, "poisson", list(lambda = 2))
```

`cov_gipp` *Covariance of GIPP*

Description

Computes the covariance of the GIPP

Usage

```
cov_gipp(beta, lambda, lambda_beta, s, t)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Parameter of outer Poisson process (> 0)
`lambda_beta`	Parameter of inner fractional Poisson process (> 0)
`s`	First time (0 < s <= t)
`t`	Second time

Value

Covariance value

Examples

```
cov_gipp(0.7, 1.0, 0.5, 0.5, 1.0)
```

`cov_inverse_stable` *Covariance of Inverse Stable Subordinator*

Description

Computes the covariance of the inverse beta-stable subordinator Based on Leonenko et al. (2014)

Usage

```
cov_inverse_stable(s, t, beta)
```

Arguments

`s`	First time
`t`	Second time (s <= t)
`beta`	Stability parameter (0 < beta < 1)

Value

Covariance value

Examples

```
cov_inverse_stable(0.5, 1, 0.8)
```

`cov_tstfnbp` *Covariance of TSTFNBP*

Description

Computes the covariance of the TSTFNBP

Usage

```
cov_tstfnbp(beta, alpha, mu, lambda1, beta1, lambda, s, t)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`alpha`	Stability parameter (0 < alpha < 1)
`mu`	Tempering parameter (> 0)
`lambda1`	Gamma subordinator rate (> mu*alpha)
`beta1`	Gamma subordinator shape (> 0)
`lambda`	Poisson parameter (> 0)
`s`	First time (0 < s <= t)
`t`	Second time

Value

Covariance value

Examples

```
cov_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 0.5, 1.0)
```

create_fcp *Create FCP Object*

Description

Creates an object representing the Fractional Counting Process

Usage

```
create_fcp(mu, zeta, theta, lambda, t, n, vartheta = NULL, n_terms = 50)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`n`	Vector of non-negative integers
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)
`n_terms`	Number of terms in series expansion (default = 50)

Value

An object of class `fcp` containing parameters, probabilities, mean, and variance.

Examples

```
fcp_obj <- create_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 0:10)
```

create_gipp *Create GIPP Object*

Description

Creates an object representing the GIPP

Usage

```
create_gipp(beta, lambda, lambda_beta, t, k, n_terms = 50)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Parameter of outer Poisson process (> 0)
`lambda_beta`	Parameter of inner fractional Poisson process (> 0)
`t`	Time (> 0)
`k`	Vector of non-negative integers
`n_terms`	Number of terms in series expansion (default = 50)

Value

An object of class `gipp` containing parameters, probabilities, mean, and variance.

Examples

```
gipp_obj <- create_gipp(0.7, 1.0, 0.5, 1.0, 0:10)
```

`create_tcfcp` *Create TCFCP Object*

Description

Creates an object representing the Time-Changed Fractional Counting Process

Usage

```
create_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  n,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)

t	Time (> 0)
n	Vector of non-negative integers
subordinator_type	Type of subordinator
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)
n_terms	Number of terms in series expansion (default = 50)

Value

An object of class `tcfcp` containing parameters, probabilities, mean, and variance.

Examples

```
tcfcp_obj <- create_tcfcp(0.8, 1.0, 0.5, 1.0, 1.0, 0:10, "gamma", list(shape = 2, rate = 1))
```

`dgamma_subordinator` *PDF of Gamma Subordinator*

Description

Computes the probability density function of the gamma subordinator

Usage

```
dgamma_subordinator(x, t, shape, rate)
```

Arguments

x	Value
t	Time
shape	Shape parameter (beta1)
rate	Rate parameter (lambda1)

Value

PDF value

Examples

```
dgamma_subordinator(1, 1, 2, 1)
```

dtempered_mittag_leffler_levy
PDF of Tempered Mittag-Leffler Lévy Process

Description

Computes the probability density function of the tempered Mittag-Leffler Lévy process Based on Kumar et al. (2019b)

Usage

```
dtempered_mittag_leffler_levy(x, t, alpha, mu, lambda1, beta1, n_terms = 50)
```

Arguments

x	Value
t	Time
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> 0)
beta1	Gamma subordinator shape (> 0)
n_terms	Number of terms in series expansion (default = 50)

Value

PDF value

Examples

```
dtempered_mittag_leffler_levy(1, 1, 0.8, 0.5, 2, 1)
```

egf_fractional_bell *Exponential Generating Function of Fractional Bell Polynomials*

Description

Computes the exponential generating function $\sum_{m=0}^{\infty}(u^m/m!)B_m^{\beta}(x) = L_{\beta}(x(e^u - 1))$

Usage

```
egf_fractional_bell(beta, x, u, n_terms = 50)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`x`	Argument
`u`	Generating function variable
`n_terms`	Number of terms in series expansion (default = 50)

Value

Value of the exponential generating function

Examples

```
egf_fractional_bell(0.7, 1.0, 0.5)
```

`fde_cgfcp` *Fractional Differential Equation for CGFCP*

Description

Computes the fractional differential equation governing the CGFCP

Usage

```
fde_cgfcp(
  beta,
  lambda,
  t,
  n,
  jump_dist = "poisson",
  jump_params = list(),
  h = 1e-06
)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k for the GFCP
`t`	Time (> 0)
`n`	Value at which to evaluate the derivative
`jump_dist`	Distribution of jump sizes
`jump_params`	Parameters for jump distribution
`h`	Step size for numerical differentiation (default = 1e-6)

Value

Value of the fractional derivative

Examples

```
fde_cgfcp(0.7, c(1.0, 0.5), 1.0, 5, "poisson", list(lambda = 2))
```

`first_passage_density_gipp` *First Passage Time Density of GIPP*

Description

Computes the density of the first passage time for GIPP

Usage

```
first_passage_density_gipp(beta, lambda, lambda_beta, t, k, n_terms = 50)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)
k	Threshold level (positive integer)
n_terms	Number of terms in series expansion (default = 50)

Value

Density value at t

Examples

```
first_passage_density_gipp(0.7, 1.0, 0.5, 1.0, 3)
```

first_passage_time_fcp

First Passage Time Distribution of FCP

Description

Computes the survival function of the first passage time P(T_w > t) where T_w = inf{t >= 0: N(t) >= w}

Usage

```
first_passage_time_fcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  w,
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`w`	Threshold level (positive integer)
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)
`n_terms`	Number of terms in series expansion (default = 50)

Value

Survival probability P(T_w > t)

Examples

```
first_passage_time_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 3)
```

first_passage_time_gipp

First Passage Time Distribution of GIPP

Description

Computes the survival function of the first passage time for GIPP

Usage

```
first_passage_time_gipp(beta, lambda, lambda_beta, t, k, n_terms = 50)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)
k	Threshold level (positive integer)
n_terms	Number of terms in series expansion (default = 50)

Value

Survival probability P(T_k > t)

Examples

```
first_passage_time_gipp(0.7, 1.0, 0.5, 1.0, 3)
```

first_passage_time_tcfcp

First Passage Time Distribution of TCFCP

Description

Computes the survival function of the first passage time for TCFCP

Usage

```
first_passage_time_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  w,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
w	Threshold level (positive integer)
subordinator_type	Type of subordinator
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)
n_terms	Number of terms in series expansion (default = 50)

Value

Survival probability P(T_w > t)

Examples

```
first_passage_time_tcfcp(0.8, 1.0, 0.5, 1.0, 1.0, 3, "gamma", list(shape = 2, rate = 1))
```

`first_waiting_time_fcp` *First Waiting Time Distribution of FCP*

Description

Computes the first waiting time distribution of the FCP

Usage

```
first_waiting_time_fcp(mu, zeta, theta, lambda, tau, vartheta = NULL)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
tau	Time (> 0)
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)

Value

First waiting time density at tau

Examples

```
first_waiting_time_fcp(0.8, 1.0, 0.5, 1.0, 0.5)
```

---

first_waiting_time_tcfcp
First Waiting Time Distribution of TCFCP

Description

Computes the first waiting time distribution of the TCFCP

Usage

```
first_waiting_time_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  tau,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
tau	Time (> 0)
subordinator_type	Type of subordinator
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)
n_terms	Number of terms in series expansion (default = 50)

Value

First waiting time density at tau

Examples

```
first_waiting_time_tcfcp(0.8, 1.0, 0.5, 1.0, 0.5, "gamma", list(shape = 2, rate = 1))
```

---

fractional_bell_polynomial

Fractional Bell Polynomials

Description

Computes the fractional Bell polynomials $B_n^\beta(x)$ Uses log-space computations to prevent numerical overflow for large orders. From Soni and Pathak (2024)

Usage

```
fractional_bell_polynomial(n, beta, x, n_terms = 50)
```

Arguments

n	Polynomial order (non-negative integer)
beta	Fractional parameter (0 < beta < 1)
x	Argument
n_terms	Number of terms in series expansion (default = 50)

Value

Value of the fractional Bell polynomial

Examples

```
fractional_bell_polynomial(5, 0.7, 1.0)
```

`fractional_counting_process` *Fractional Counting Process (FCP)*

Description

Computes the probability mass function of the Fractional Counting Process based on the three-parameter Mittag-Leffler function (Prabhakar function) From Garg et al. (2025)

Usage

```
fractional_counting_process(
  mu,
  zeta,
  theta,
  lambda,
  t,
  n,
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`n`	Vector of non-negative integers
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)
`n_terms`	Number of terms in series expansion (default = 50)

Value

Vector of probabilities P(N(t) = n)

Examples

```
fractional_counting_process(0.8, 1.0, 0.5, 1.0, 1.0, 0:10)
```

fractional_stirling2 *Fractional Stirling Numbers of the Second Kind*

Description

Computes the fractional Stirling numbers of the second kind $S_\beta(m, k)$ Uses log-space computation to prevent numerical overflow. From Soni and Pathak (2024)

Usage

```
fractional_stirling2(m, k, beta)
```

Arguments

m	Upper index (non-negative integer)
k	Lower index (non-negative integer)
beta	Fractional parameter (0 < beta < 1)

Value

Value of the fractional Stirling number

Examples

```
fractional_stirling2(5, 3, 0.7)
```

gamma_subordinator *Gamma Subordinator*

Description

Simulates a gamma subordinator G(t) with shape parameter beta1*t and rate lambda1

Usage

```
gamma_subordinator(t, shape, rate, n = 1)
```

Arguments

t	Time
shape	Shape parameter (beta1)
rate	Rate parameter (lambda1)
n	Number of simulations

Value

Simulated values of the gamma subordinator

Examples

```
gamma_subordinator(1, 2, 1, 10)
```

---

generalized_fractional_risk_process
Generalized Fractional Risk Process (GFRP)

Description

Computes the Generalized Fractional Risk Process for insurance applications $R^{\beta}(t) = \nu + \eta(1 + \rho)\sum_{i=1}^{k} i\lambda_i W_{\beta}(t) - \sum_{j=1}^{N^{\beta}(t)} X_j$ From Soni and Pathak (2024)

Usage

```
generalized_fractional_risk_process(
  nu,
  eta,
  rho,
  beta,
  lambda,
  t,
  claim_dist = "gamma",
  claim_params = list(),
  n_sim = 1000
)
```

Arguments

nu	Initial capital (> 0)
eta	Mean claim size (> 0)
rho	Safety loading parameter (>= 0)
beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t	Time (> 0)
claim_dist	Distribution of claim sizes ("gamma", "exponential", "lognormal")
claim_params	Parameters for claim distribution
n_sim	Number of simulations (default = 1000)

Value

Simulated values of the risk process at time t

Examples

```
generalized_fractional_risk_process(100, 10, 0.1, 0.7, c(1.0, 0.5), 1.0,
                                     "gamma", list(shape = 2, rate = 0.2))
```

`generalized_fractional_stirling2`
Generalized Fractional Stirling Numbers of the Second Kind

Description

Computes the generalized fractional Stirling numbers $S_{\mu,\vartheta}(p,i)$ From Laskin (2024)

Usage

```
generalized_fractional_stirling2(p, i, mu, vartheta)
```

Arguments

`p`	Upper index (non-negative integer)
`i`	Lower index (non-negative integer)
`mu`	Parameter mu (0 < mu <= 1)
`vartheta`	Parameter vartheta (> 0)

Value

Value of the generalized fractional Stirling number

Examples

```
generalized_fractional_stirling2(5, 3, 0.8, 1.0)
```

`generalized_iterated_poisson`
Generalized Iterated Poisson Process (GIPP)

Description

Computes the probability mass function of the Generalized Iterated Poisson Process Q(t) = N(N_beta(t), lambda) - composition of HPP with TFPP. From Soni and Pathak (2024)

Usage

```
generalized_iterated_poisson(beta, lambda, lambda_beta, t, k, n_terms = 50)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Parameter of outer Poisson process (> 0)
`lambda_beta`	Parameter of inner fractional Poisson process (> 0)
`t`	Time (> 0)
`k`	Vector of non-negative integers
`n_terms`	Number of terms in series expansion (default = 50)

Value

Vector of probabilities P(Q(t) = k)

Examples

```
generalized_iterated_poisson(0.7, 1.0, 0.5, 1.0, 0:10)
```

---

gfcp *Generalized Fractional Counting Process (GFCP)*

Description

Computes the probability mass function of the GFCP (with jumps of size 1, ..., k) using PGF inversion via FFT.

Usage

```
gfcp(beta, lambda, t, n)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k
`t`	Time (> 0)
`n`	Vector of non-negative integers

Value

Vector of probabilities P(N(t) = n)

Examples

```
gfcp(0.7, c(1.0, 0.5), 1.0, 0:10)
```

incomplete_beta *Incomplete Beta Function*

Description

Computes the incomplete beta function $B(a, b; x) = \int_0^x t^{a-1}(1-t)^{b-1}dt$

Usage

```
incomplete_beta(a, b, x)
```

Arguments

a	Parameter a > 0
b	Parameter b > 0
x	Upper limit of integration (0 <= x <= 1)

Value

Value of the incomplete beta function

Examples

```
incomplete_beta(0.5, 0.5, 0.5)
```

inverse_stable_subordinator *Inverse Stable Subordinator*

Description

Simulates an inverse beta-stable subordinator E_beta(t) = inf{r >= 0: S_beta(r) > t} Using the exact relationship E_beta(t) = t^beta * S_beta(1)^(-beta).

Usage

```
inverse_stable_subordinator(t, beta, n = 1)
```

Arguments

t	Time
beta	Stability parameter (0 < beta < 1)
n	Number of simulations

Value

Simulated values of the inverse stable subordinator

Examples

```
inverse_stable_subordinator(1, 0.8, 10)
```

---

invert_pgf — *Numerical Inversion of Probability Generating Function*

Description

Computes the probability mass function (PMF) of a discrete random variable from its PGF using Fast Fourier Transform (FFT).

Usage

```
invert_pgf(pgf_func, max_n, r = 0.85, M = NULL)
```

Arguments

`pgf_func`	A function that evaluates the PGF at complex arguments
`max_n`	Maximum value of the random variable to compute the probability for
`r`	Radius of the contour circle ($0 < r < 1$, default = 0.85)
`M`	Number of points on the circle (must be a power of 2, default = NULL)

Value

Vector of probabilities P(X = 0), ..., P(X = max_n)

---

lt_fcp — *Laplace Transform of FCP*

Description

Computes the Laplace transform of the FCP (expectation of exp(-s * N(t)))

Usage

```
lt_fcp(mu, zeta, theta, lambda, t, s, vartheta = NULL)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`s`	Laplace parameter
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)

Value

Laplace transform value

Examples

```
lt_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 0.5)
```

`lt_tcfcp` *Laplace Transform of TCFCP*

Description

Computes the Laplace transform of the TCFCP (expectation of exp(-s * Z(t)))

Usage

```
lt_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  s,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
s	Laplace parameter
subordinator_type	Type of subordinator
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)
n_terms	Number of terms in series expansion (default = 50)

Value

Laplace transform value

Examples

```
lt_tcfcp(0.8, 1.0, 0.5, 1.0, 1.0, 0.5, "gamma", list(shape = 2, rate = 1))
```

---

lt_tempered_mittag_leffler_levy

Laplace Transform of Tempered Mittag-Leffler Lévy Process

Description

Computes the Laplace transform E(exp(-u * M(t)))

Usage

```
lt_tempered_mittag_leffler_levy(u, t, alpha, mu, lambda1, beta1)
```

Arguments

u	Laplace transform parameter
t	Time
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> 0)
beta1	Gamma subordinator shape (> 0)

Value

Laplace transform value

Examples

```
lt_tempered_mittag_leffler_levy(1, 1, 0.8, 0.5, 2, 1)
```

`lt_tstfnbp` *Laplace Transform of TSTFNBP*

Description

Computes the Laplace transform of the TSTFNBP (expectation of exp(-u * Q(t)))

Usage

```
lt_tstfnbp(beta, alpha, mu, lambda1, beta1, lambda, t, u, n_terms = 50)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`alpha`	Stability parameter (0 < alpha < 1)
`mu`	Tempering parameter (> 0)
`lambda1`	Gamma subordinator rate (> mu*alpha)
`beta1`	Gamma subordinator shape (> 0)
`lambda`	Poisson parameter (> 0)
`t`	Time (> 0)
`u`	Laplace parameter
`n_terms`	Number of terms in series expansion (default = 50)

Value

Laplace transform value

Examples

```
lt_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 1.0, 0.5)
```

mean_cgfcp *Mean of CGFCP*

Description

Computes the mean of the CGFCP

Usage

```
mean_cgfcp(beta, lambda, t, jump_dist = "poisson", jump_params = list())
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t	Time (> 0)
jump_dist	Distribution of jump sizes ("poisson", "geometric")
jump_params	Parameters for jump distribution

Value

Mean value

Examples

```
mean_cgfcp(0.7, c(1.0, 0.5), 1.0, "poisson", list(lambda = 2))
```

mean_fcp *Mean of Fractional Counting Process*

Description

Computes the mean of the FCP

Usage

```
mean_fcp(mu, zeta, theta, lambda, t, vartheta = NULL)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)

Value

Mean value

Examples

```
mean_fcp(0.8, 1.0, 0.5, 1.0, 1.0)
```

`mean_gfrp` *Mean of GFRP*

Description

Computes the mean of the GFRP

Usage

```
mean_gfrp(nu, eta, rho, beta, lambda, t)
```

Arguments

`nu`	Initial capital (> 0)
`eta`	Mean claim size (> 0)
`rho`	Safety loading parameter (>= 0)
`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k for the GFCP
`t`	Time (> 0)

Value

Mean value

Examples

```
mean_gfrp(100, 10, 0.1, 0.7, c(1.0, 0.5), 1.0)
```

mean_gipp *Mean of GIPP*

Description

Computes the mean of the GIPP

Usage

```
mean_gipp(beta, lambda, lambda_beta, t)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)

Value

Mean value

Examples

```
mean_gipp(0.7, 1.0, 0.5, 1.0)
```

mean_inverse_stable *Mean of Inverse Stable Subordinator*

Description

Computes the mean of the inverse beta-stable subordinator

Usage

```
mean_inverse_stable(t, beta)
```

Arguments

t	Time
beta	Stability parameter (0 < beta < 1)

Value

Mean value

Examples

```
mean_inverse_stable(1, 0.8)
```

`mean_tcfcp` *Mean of Time-Changed FCP*

Description

Computes the mean of the TCFCP

Usage

```
mean_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL
)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`subordinator_type`	Type of subordinator
`subordinator_params`	List of parameters for the subordinator
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)

Value

Mean value

Examples

```
mean_tcfcp(0.8, 1.0, 0.5, 1.0, 1.0, "gamma", list(shape = 2, rate = 1))
```

mean_tstfnbp *Mean of TSTFNBP*

Description

Computes the mean of the TSTFNBP

Usage

```
mean_tstfnbp(beta, alpha, mu, lambda1, beta1, lambda, t)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> mu*alpha)
beta1	Gamma subordinator shape (> 0)
lambda	Poisson parameter (> 0)
t	Time (> 0)

Value

Mean value

Examples

```
mean_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 1.0)
```

mgamma_subordinator *Moment of Gamma Subordinator*

Description

Computes the q-th moment of the gamma subordinator

Usage

```
mgamma_subordinator(q, t, shape, rate)
```

Arguments

`q`	Order of moment
`t`	Time
`shape`	Shape parameter (beta1)
`rate`	Rate parameter (lambda1)

Value

q-th moment value

Examples

```
mgamma_subordinator(2, 1, 2, 1)
```

`mgf_fcp` *Moment Generating Function of FCP*

Description

Computes the moment generating function of the Fractional Counting Process

Usage

```
mgf_fcp(mu, zeta, theta, lambda, t, s, vartheta = NULL)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`s`	MGF argument
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)

Value

MGF value

Examples

```
mgf_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 0.5)
```

mgf_gipp *Moment Generating Function of GIPP*

Description

Computes the moment generating function of the GIPP

Usage

```
mgf_gipp(beta, lambda, lambda_beta, t, u)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)
u	MGF argument

Value

MGF value

Examples

```
mgf_gipp(0.7, 1.0, 0.5, 1.0, 0.5)
```

mittag_leffler *Mittag-Leffler Function*

Description

Computes the Mittag-Leffler function $E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} z^k/\Gamma(\alpha k + \beta)$ Supports complex arguments and uses log-space computation for numerical stability.

Usage

```
mittag_leffler(alpha, beta = 1, z, n_terms = 100, tolerance = 1e-10)
```

Arguments

alpha	Parameter alpha > 0
beta	Parameter beta > 0 (default = 1)
z	Complex argument (can be a vector)
n_terms	Number of terms in series expansion (default = 100)
tolerance	Convergence tolerance (default = 1e-10)

Value

Value of the Mittag-Leffler function

Examples

```
mittag_leffler(0.8, 1, -1)
mittag_leffler(1, 1, 1)  # Should equal exp(1)
```

---

mittag_leffler_prabhakar
Prabhakar (Three-Parameter) Mittag-Leffler Function

Description

Computes the generalized three-parameter Mittag-Leffler function $E_{\alpha,\beta}^{\gamma}(z) = \sum_{k=0}^{\infty} (\gamma)_k z^k / (k!\Gamma(\alpha k + \beta))$ Supports complex arguments and uses log-space computation for numerical stability.

Usage

```
mittag_leffler_prabhakar(
  alpha,
  beta,
  gamma,
  z,
  n_terms = 100,
  tolerance = 1e-10
)
```

Arguments

alpha	Parameter alpha > 0
beta	Parameter beta > 0
gamma	Parameter gamma > 0
z	Complex argument (can be a vector)
n_terms	Number of terms in series expansion (default = 100)
tolerance	Convergence tolerance (default = 1e-10)

Value

Value of the Prabhakar Mittag-Leffler function

Examples

```
mittag_leffler_prabhakar(0.8, 1, 1, -1)
```

net_profit_condition_gfrp *Net Profit Condition for GFRP*

Description

Checks if the net profit condition is satisfied

Usage

```
net_profit_condition_gfrp(eta, rho, beta, lambda)
```

Arguments

eta	Mean claim size (> 0)
rho	Safety loading parameter (>= 0)
beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP

Value

Logical indicating if net profit condition is satisfied

Examples

```
net_profit_condition_gfrp(10, 0.1, 0.7, c(1.0, 0.5))
```

partial_ordinary_bell *Partial Ordinary Bell Polynomials*

Description

Computes the partial ordinary Bell polynomials $\hat{B}_{j,i}(a)$ Used in the asymptotic expansion of tempered Mittag-Leffler moments.

Usage

```
partial_ordinary_bell(j, i, a)
```

Arguments

j	Index (non-negative integer)
i	Index (non-negative integer)
a	Coefficients vector

Value

Value of the partial ordinary Bell polynomial

Examples

```
partial_ordinary_bell(3, 2, c(0.5, 0.3, 0.2))
```

---

pgamma_subordinator   *CDF of Gamma Subordinator*

Description

Computes the cumulative distribution function of the gamma subordinator

Usage

```
pgamma_subordinator(x, t, shape, rate)
```

Arguments

`x`	Value
`t`	Time
`shape`	Shape parameter (beta1)
`rate`	Rate parameter (lambda1)

Value

CDF value

Examples

```
pgamma_subordinator(1, 1, 2, 1)
```

pgf_cgfcp *Probability Generating Function of CGFCP*

Description

Computes the probability generating function of the CGFCP

Usage

```
pgf_cgfcp(
  beta,
  lambda,
  t,
  u,
  jump_dist = "poisson",
  jump_params = list(),
  n_terms = 50
)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t	Time (> 0)
u	PGF argument (|u| <= 1)
jump_dist	Distribution of jump sizes
jump_params	Parameters for jump distribution
n_terms	Number of terms in series expansion (default = 50)

Value

PGF value

Examples

```
pgf_cgfcp(0.7, c(1.0, 0.5), 1.0, 0.5, "poisson", list(lambda = 2))
```

pgf_fcp — *Probability Generating Function of FCP*

Description

Computes the probability generating function of the Fractional Counting Process

Usage

```
pgf_fcp(mu, zeta, theta, lambda, t, s, vartheta = NULL)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
s	PGF argument (|s| <= 1)
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)

Value

PGF value

Examples

```
pgf_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 0.5)
```

pgf_gipp — *Probability Generating Function of GIPP*

Description

Computes the probability generating function of the GIPP

Usage

```
pgf_gipp(beta, lambda, lambda_beta, t, u)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)
u	PGF argument (|u| <= 1)

Value

PGF value

Examples

```
pgf_gipp(0.7, 1.0, 0.5, 1.0, 0.5)
```

---

pgf_tstfnbp *Probability Generating Function of TSTFNBP*

Description

Computes the probability generating function of the TSTFNBP

Usage

```
pgf_tstfnbp(beta, alpha, mu, lambda1, beta1, lambda, t, u, n_terms = 50)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> mu*alpha)
beta1	Gamma subordinator shape (> 0)
lambda	Poisson parameter (> 0)
t	Time (> 0)
u	PGF argument (|u| <= 1)
n_terms	Number of terms in series expansion (default = 50)

Value

PGF value

Examples

```
pgf_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 1.0, 0.5)
```

pochhammer — *Pochhammer Symbol*

Description

Computes the Pochhammer symbol $(a)_n = \Gamma(a+n)/\Gamma(a)$ Uses log-gamma for numerical stability and preventing overflow.

Usage

```
pochhammer(a, n)
```

Arguments

`a`	Real number
`n`	Non-negative integer

Value

Value of the Pochhammer symbol

Examples

```
pochhammer(1, 5)  # Should equal 120 = 5!
```

print.fcp — *S3 Print Method for FCP*

Description

S3 Print Method for FCP

Usage

```
## S3 method for class 'fcp'
print(x, ...)
```

Arguments

`x`	FCP object
`...`	Additional arguments

Value

No return value, called for side effects.

print.gipp — *S3 Print Method for GIPP*

Description

S3 Print Method for GIPP

Usage

```
## S3 method for class 'gipp'
print(x, ...)
```

Arguments

`x`	GIPP object
`...`	Additional arguments

Value

No return value, called for side effects.

print.tcfcp — *S3 Print Method for TCFCP*

Description

S3 Print Method for TCFCP

Usage

```
## S3 method for class 'tcfcp'
print(x, ...)
```

Arguments

`x`	TCFCP object
`...`	Additional arguments

Value

No return value, called for side effects.

ruin_probability_gfrp *Ruin Probability for GFRP*

Description

Estimates the ruin probability using coherent path simulation

Usage

```
ruin_probability_gfrp(
  nu,
  eta,
  rho,
  beta,
  lambda,
  t_max,
  claim_dist = "gamma",
  claim_params = list(),
  n_grid = 100,
  n_sim = 1000
)
```

Arguments

nu	Initial capital (> 0)
eta	Mean claim size (> 0)
rho	Safety loading parameter (>= 0)
beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k for the GFCP
t_max	Maximum time horizon
claim_dist	Distribution of claim sizes
claim_params	Parameters for claim distribution
n_grid	Number of grid points for path discretization (default = 100)
n_sim	Number of simulations (default = 1000)

Value

Estimated ruin probability

Examples

```
ruin_probability_gfrp(100, 10, 0.1, 0.7, c(1.0, 0.5), 10,
                      "gamma", list(shape = 2, rate = 0.2), n_sim = 100)
```

simulate_cgfcp *Simulate Sample Paths of CGFCP*

Description

Simulates sample paths of the Compound Generalized Fractional Counting Process.

Usage

```
simulate_cgfcp(
  beta,
  lambda,
  t_max,
  jump_dist = "poisson",
  jump_params = list(),
  n_paths = 1,
  n_grid = 100
)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Vector of rates lambda_1, ..., lambda_k
t_max	Maximum simulation time
jump_dist	Distribution of jump sizes ("poisson", "geometric")
jump_params	Parameters for jump distribution
n_paths	Number of paths to simulate (default = 1)
n_grid	Number of grid points for path discretization (default = 100)

Value

A list of paths, each path is a list containing `times`, `states`, and the `subordinator` path

Examples

```
simulate_cgfcp(0.7, c(1.0, 0.5), 10, "poisson", list(lambda = 2), n_paths = 2)
```

`simulate_fcp` *Simulate Sample Paths of FCP*

Description

Simulates sample paths of the Fractional Counting Process. Uses exact Chambers-Mallows-Stuck simulation for FPP (zeta = vartheta = 1) and numerical CDF inversion for general FCP.

Usage

```
simulate_fcp(mu, zeta, theta, lambda, t_max, n_paths = 1, vartheta = NULL)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t_max`	Maximum simulation time
`n_paths`	Number of paths to simulate (default = 1)
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)

Value

A list of paths, each path is a list containing `times` and `states`

Examples

```
simulate_fcp(0.8, 1.0, 0.5, 1.0, 10, n_paths = 2)
```

`simulate_gfrp` *Simulate Sample Paths of GFRP*

Description

Simulates sample paths of the Generalized Fractional Risk Process.

Usage

```
simulate_gfrp(
  nu,
  eta,
  rho,
  beta,
  lambda,
  t_max,
  claim_dist = "gamma",
  claim_params = list(),
  n_paths = 1,
  n_grid = 100
)
```

Arguments

`nu`	Initial capital (> 0)
`eta`	Mean claim size (> 0)
`rho`	Safety loading parameter (>= 0)
`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k for the GFCP
`t_max`	Maximum simulation time
`claim_dist`	Distribution of claim sizes ("gamma", "exponential", "lognormal")
`claim_params`	Parameters for claim distribution
`n_paths`	Number of paths to simulate (default = 1)
`n_grid`	Number of grid points for path discretization (default = 100)

Value

A list of paths, each path is a list containing `times`, `states` (capital), `subordinator` (W_beta), and `claims` (accumulated claims)

Examples

```
simulate_gfrp(100, 10, 0.1, 0.7, c(1.0, 0.5), 10, "gamma", list(shape = 2, rate = 0.2), n_paths = 2)
```

---

simulate_gipp *Simulate Sample Paths of GIPP*

Description

Simulates sample paths of the Generalized Iterated Poisson Process. Q(t) = N(N_beta(t), lambda).

Usage

```
simulate_gipp(beta, lambda, lambda_beta, t_max, n_paths = 1, n_grid = 100)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Parameter of outer Poisson process (> 0)
`lambda_beta`	Parameter of inner fractional Poisson process (> 0)
`t_max`	Maximum simulation time
`n_paths`	Number of paths to simulate (default = 1)
`n_grid`	Number of grid points for path discretization (default = 100)

Value

A list of paths, each path is a list containing `times`, `states`, and `inner_process` (TFPP path)

Examples

```
simulate_gipp(0.7, 1.0, 0.5, 10, n_paths = 2)
```

---

simulate_tcfcp *Simulate Sample Paths of TCFCP*

Description

Simulates sample paths of the Time-Changed Fractional Counting Process. First simulates the Lévy subordinator path on a time grid, then time-changes an independent FPP/FCP path.

Usage

```
simulate_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t_max,
  subordinator_type = "gamma",
  subordinator_params = list(),
  n_paths = 1,
  vartheta = NULL,
  n_grid = 100
)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t_max`	Maximum simulation time
`subordinator_type`	Type of subordinator ("gamma", "tempered_mittag_leffler")
`subordinator_params`	List of parameters for the subordinator
`n_paths`	Number of paths to simulate (default = 1)
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)
`n_grid`	Number of grid points for path discretization (default = 100)

Value

A list of paths, each path is a list containing `times`, `states`, and the `subordinator` path

Examples

```
simulate_tcfcp(0.8, 1.0, 0.5, 1.0, 10, "gamma", list(shape = 2, rate = 1), n_paths = 2)
```

`simulate_tstfnbp` *Simulate Sample Paths of TSTFNBP*

Description

Simulates sample paths of the Tempered Space-Time Fractional Negative Binomial Process. First simulates the TMLLP subordinator path, then time-changes an independent FPP path.

Usage

```
simulate_tstfnbp(
  beta,
  alpha,
  mu,
  lambda1,
  beta1,
  lambda,
  t_max,
  n_paths = 1,
  n_grid = 100
)
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`alpha`	Stability parameter (0 < alpha < 1)
`mu`	Tempering parameter (> 0)
`lambda1`	Gamma subordinator rate (> mu*alpha)
`beta1`	Gamma subordinator shape (> 0)
`lambda`	Poisson parameter (> 0)
`t_max`	Maximum simulation time
`n_paths`	Number of paths to simulate (default = 1)
`n_grid`	Number of grid points for path discretization (default = 100)

Value

A list of paths, each path is a list containing `times`, `states`, and the `subordinator` path

Examples

```
simulate_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 10, n_paths = 2)
```

---

stirling2 *Classical Stirling Numbers of the Second Kind*

Description

Computes the classical Stirling numbers of the second kind S(m, k)

Usage

```
stirling2(m, k)
```

Arguments

`m`	Upper index (non-negative integer)
`k`	Lower index (non-negative integer)

Value

Value of the Stirling number

Examples

```
stirling2(5, 3)
```

`tempered_mittag_leffler_levy` *Tempered Mittag-Leffler Lévy Process*

Description

Simulates the tempered Mittag-Leffler Lévy process $M_{\alpha,\mu,\lambda_1,\beta_1}(t) = S_{\alpha,\mu}(G_{\lambda_1,\beta_1}(t))$

Usage

```
tempered_mittag_leffler_levy(t, alpha, mu, lambda1, beta1, n = 1)
```

Arguments

`t`	Time
`alpha`	Stability parameter (0 < alpha < 1)
`mu`	Tempering parameter (> 0)
`lambda1`	Gamma subordinator rate (> 0)
`beta1`	Gamma subordinator shape (> 0)
`n`	Number of simulations

Value

Simulated values of the tempered Mittag-Leffler Lévy process

Examples

```
tempered_mittag_leffler_levy(1, 0.8, 0.5, 2, 1, 10)
```

`tempered_mittag_leffler_moment` *Fractional Moment of Tempered Mittag-Leffler Lévy Process*

Description

Computes the q-th order moment of the tempered Mittag-Leffler Lévy process based on asymptotic results from Kumar et al. (2019b)

Usage

```
tempered_mittag_leffler_moment(q, alpha, mu, lambda1, beta1, t)
```

Arguments

q	Order of moment (> 0)
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> 0)
beta1	Gamma subordinator shape (> 0)
t	Time (> 0)

Value

Asymptotic value of the q-th moment

Examples

```
tempered_mittag_leffler_moment(2, 0.8, 0.5, 2, 1, 1)
```

---

tempered_stable_subordinator *Tempered Stable Subordinator*

Description

Simulates a tempered alpha-stable subordinator. Uses exact rejection-sampling via tilting of stable random variables.

Usage

```
tempered_stable_subordinator(t, alpha, mu, n = 1)
```

Arguments

t	Time
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
n	Number of simulations

Value

Simulated values of the tempered stable subordinator

Examples

```
tempered_stable_subordinator(1, 0.8, 0.5, 10)
```

tempered_st_fnbp	*Tempered Space-Time Fractional Negative Binomial Process (TSTFNBP)*

Description

Computes the probability mass function of the TSTFNBP $Q^{\beta}_{\alpha,\mu,\lambda_1,\beta_1}(t,\lambda) = N_{\beta}(M_{\alpha,\mu,\lambda_1,\beta_1}(t),\lambda)$ From Garg et al. (2025)

Usage

```
tempered_st_fnbp(beta, alpha, mu, lambda1, beta1, lambda, t, n, n_terms = 50)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> mu*alpha)
beta1	Gamma subordinator shape (> 0)
lambda	Poisson parameter (> 0)
t	Time (> 0)
n	Vector of non-negative integers
n_terms	Number of terms in series expansion (default = 50)

Value

Vector of probabilities P(Q(t) = n)

Examples

```
tempered_st_fnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 1.0, 0:10)
```

time_changed_fcp	*Time-Changed Fractional Counting Process (TCFCP)*

Description

Computes the probability mass function of the Time-Changed Fractional Counting Process obtained by subordinating the FCP with a Lévy subordinator. From Garg et al. (2025)

Usage

```
time_changed_fcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  n,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL,
  n_terms = 50
)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
n	Vector of non-negative integers
subordinator_type	Type of subordinator ("gamma", "tempered_mittag_leffler")
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)
n_terms	Number of terms in series expansion (default = 50)

Value

Vector of probabilities P(Z(t) = n)

Examples

```
time_changed_fcp(0.8, 1.0, 0.5, 1.0, 1.0, 0:10,
                 "gamma", list(shape = 2, rate = 1))
```

var_cgfcp — *Variance of CGFCP*

Description

Computes the variance of the CGFCP

Usage

```
var_cgfcp(beta, lambda, t, jump_dist = "poisson", jump_params = list())
```

Arguments

`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k for the GFCP
`t`	Time (> 0)
`jump_dist`	Distribution of jump sizes ("poisson", "geometric")
`jump_params`	Parameters for jump distribution

Value

Variance value

Examples

```
var_cgfcp(0.7, c(1.0, 0.5), 1.0, "poisson", list(lambda = 2))
```

var_fcp — *Variance of Fractional Counting Process*

Description

Computes the variance of the FCP

Usage

```
var_fcp(mu, zeta, theta, lambda, t, vartheta = NULL)
```

Arguments

`mu`	Parameter mu (0 < mu <= 1)
`zeta`	Parameter zeta (> 0)
`theta`	Parameter theta (0 < theta <= 1)
`lambda`	Parameter lambda (> 0)
`t`	Time (> 0)
`vartheta`	Parameter vartheta (>= mu*zeta, default = zeta)

Value

Variance value

Examples

```
var_fcp(0.8, 1.0, 0.5, 1.0, 1.0)
```

---

var_gfrp *Variance of GFRP*

Description

Computes the variance of the GFRP

Usage

```
var_gfrp(
  nu,
  eta,
  rho,
  beta,
  lambda,
  t,
  claim_dist = "gamma",
  claim_params = list()
)
```

Arguments

`nu`	Initial capital (> 0)
`eta`	Mean claim size (> 0)
`rho`	Safety loading parameter (>= 0)
`beta`	Fractional parameter (0 < beta < 1)
`lambda`	Vector of rates lambda_1, ..., lambda_k for the GFCP
`t`	Time (> 0)
`claim_dist`	Distribution of claim sizes
`claim_params`	Parameters for claim distribution

Value

Variance value

Examples

```
var_gfrp(100, 10, 0.1, 0.7, c(1.0, 0.5), 1.0, "gamma", list(shape = 2, rate = 0.2))
```

var_gipp *Variance of GIPP*

Description

Computes the variance of the GIPP

Usage

```
var_gipp(beta, lambda, lambda_beta, t)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
lambda	Parameter of outer Poisson process (> 0)
lambda_beta	Parameter of inner fractional Poisson process (> 0)
t	Time (> 0)

Value

Variance value

Examples

```
var_gipp(0.7, 1.0, 0.5, 1.0)
```

var_tcfcp *Variance of Time-Changed FCP*

Description

Computes the variance of the TCFCP using the law of total variance.

Usage

```
var_tcfcp(
  mu,
  zeta,
  theta,
  lambda,
  t,
  subordinator_type = "gamma",
  subordinator_params = list(),
  vartheta = NULL
)
```

Arguments

mu	Parameter mu (0 < mu <= 1)
zeta	Parameter zeta (> 0)
theta	Parameter theta (0 < theta <= 1)
lambda	Parameter lambda (> 0)
t	Time (> 0)
subordinator_type	Type of subordinator
subordinator_params	List of parameters for the subordinator
vartheta	Parameter vartheta (>= mu*zeta, default = zeta)

Value

Variance value

Examples

```
var_tcfcp(0.8, 1.0, 0.5, 1.0, 1.0, "gamma", list(shape = 2, rate = 1))
```

---

var_tstfnbp *Variance of TSTFNBP*

Description

Computes the variance of the TSTFNBP

Usage

```
var_tstfnbp(beta, alpha, mu, lambda1, beta1, lambda, t)
```

Arguments

beta	Fractional parameter (0 < beta < 1)
alpha	Stability parameter (0 < alpha < 1)
mu	Tempering parameter (> 0)
lambda1	Gamma subordinator rate (> mu*alpha)
beta1	Gamma subordinator shape (> 0)
lambda	Poisson parameter (> 0)
t	Time (> 0)

Value

Variance value

Examples

```
var_tstfnbp(0.7, 0.8, 0.5, 2, 1, 1.0, 1.0)
```

`wright_function` *Generalized Wright Function*

Description

Computes the generalized Wright function $\psi_{p,q}[(a_i, \alpha_i)_{1,p}; (b_j, \beta_j)_{1,q}](z)$ Uses log-space computation for numerical stability.

Usage

```
wright_function(a, alpha, b, beta, z, n_terms = 100, tolerance = 1e-10)
```

Arguments

a	Vector of a_i parameters
alpha	Vector of alpha_i parameters
b	Vector of b_j parameters
beta	Vector of beta_j parameters
z	Complex argument (can be a vector)
n_terms	Number of terms in series expansion (default = 100)
tolerance	Convergence tolerance (default = 1e-10)

Value

Value of the generalized Wright function

Examples

```
wright_function(c(1), c(1), c(1), c(1), 1)
```

Index

alternative_gfrp, 3

bell_polynomial, 4

check_lrd_gipp, 5
check_martingale_gfrp, 5
compound_gfcp, 6
cov_gipp, 7
cov_inverse_stable, 7
cov_tstfnbp, 8
create_fcp, 9
create_gipp, 9
create_tcfcp, 10

dgamma_subordinator, 11
dtempered_mittag_leffler_levy, 12

egf_fractional_bell, 12

fde_cgfcp, 13
first_passage_density_gipp, 14
first_passage_time_fcp, 15
first_passage_time_gipp, 16
first_passage_time_tcfcp, 16
first_waiting_time_fcp, 17
first_waiting_time_tcfcp, 18
fractional_bell_polynomial, 19
fractional_counting_process, 20
fractional_stirling2, 21

gamma_subordinator, 21
generalized_fractional_risk_process, 22
generalized_fractional_stirling2, 23
generalized_iterated_poisson, 23
gfcp, 24

incomplete_beta, 25
inverse_stable_subordinator, 25
invert_pgf, 26

lt_fcp, 26
lt_tcfcp, 27
lt_tempered_mittag_leffler_levy, 28
lt_tstfnbp, 29

mean_cgfcp, 30
mean_fcp, 30
mean_gfrp, 31
mean_gipp, 32
mean_inverse_stable, 32
mean_tcfcp, 33
mean_tstfnbp, 34
mgamma_subordinator, 34
mgf_fcp, 35
mgf_gipp, 36
mittag_leffler, 36
mittag_leffler_prabhakar, 37

net_profit_condition_gfrp, 38

partial_ordinary_bell, 38
pgamma_subordinator, 39
pgf_cgfcp, 40
pgf_fcp, 41
pgf_gipp, 41
pgf_tstfnbp, 42
pochhammer, 43
print.fcp, 43
print.gipp, 44
print.tcfcp, 44

ruin_probability_gfrp, 45

simulate_cgfcp, 46
simulate_fcp, 47
simulate_gfrp, 47
simulate_gipp, 48
simulate_tcfcp, 49
simulate_tstfnbp, 50
stirling2, 51

tempered_mittag_leffler_levy, 52
tempered_mittag_leffler_moment, 52
tempered_st_fnbp, 54
tempered_stable_subordinator, 53
time_changed_fcp, 54

var_cgfcp, 56
var_fcp, 56
var_gfrp, 57
var_gipp, 58
var_tcfcp, 58
var_tstfnbp, 59

wright_function, 60