Package: serieshaz (via r-universe)

May 24, 2026

Title Series System Distributions from Dynamic Failure Rate Components

Version 0.2.0

Description Compose multiple dynamic failure rate distributions into series system distributions where the system hazard equals the sum of component hazards. Supports hazard, survival, cumulative distribution function, density, sampling, and maximum likelihood estimation fitting via the dfr_dist() class from 'flexhaz'. Series distributions implement the 'dist.structure' protocol so structural queries (phi, min_paths, min_cuts, system_signature, structural importance, reliability, dual) and the importance measures from 'dist.structure' work directly on serieshaz objects. Methods for series system reliability follow Barlow and Proschan (1975, ISBN:0898713692).

License GPL (>= 3)

URL https://github.com/queelius/serieshaz, https://queelius.github.io/serieshaz/

BugReports https://github.com/queelius/serieshaz/issues

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 3.5.0)

Imports flexhaz, algebraic.dist, likelihood.model, generics, numDeriv, dist.structure

Suggests testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

Repository https://queelius.r-universe.dev

Date/Publication 2026-05-18 05:28:53 UTC

RemoteUrl https://github.com/queelius/serieshaz

RemoteRef HEAD

RemoteSha bed2c400da919d900827c05e5b50e80bc755723c

Contents

`assumptions.dfr_dist_series`

Assumptions for series system distributions

Description

Returns the statistical and structural assumptions underlying a series system model, which are important for the validity of MLE-based inference.

Usage

```
## S3 method for class 'dfr_dist_series'
assumptions(model, ...)
```

Arguments

`model`	A `dfr_dist_series` object.
`...`	Additional arguments (unused).

Details

The assumptions returned are:

- **Series structure**: The system fails when any component fails (weakest-link model)
- **Component independence**: Component lifetimes are statistically independent
- **Non-negative hazard**: Each component hazard satisfies $h_j(t) \geq 0$ for all $t > 0$
- **Proper distribution**: The cumulative hazard diverges, ensuring $S_{sys}(t) \to 0$ as $t \to \infty$
- **Positive support**: The time domain is $(0, \infty)$
- **Independent observations**: The observed lifetimes are independent
- **Censoring convention**: `delta` = `1` for exact, `0` for right-censored, `-1` for left-censored

- **Non-informative censoring**: The censoring mechanism carries no information about the failure process

These assumptions are required for the MLE fitting procedure (`fit`) to produce valid estimates. Violation of component independence, in particular, invalidates the hazard-sum property that defines series systems.

Value

Character vector of model assumptions.

See Also

`assumptions` for the generic, `dfr_dist_series` for the constructor, `vignette("series-fitting")` for how assumptions affect inference

Other series system: `dfr_dist_series()`, `is_dfr_dist_series()`, `print.dfr_dist_series()`

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_weibull(shape = 2, scale = 100)
))
assumptions(sys)
```

---

component_hazard *Get the hazard function for a specific component*

Description

Returns a closure that computes the hazard rate for component `j` of a series system. Useful for plotting hazard decompositions and understanding each component's contribution to system risk.

Usage

```
component_hazard(x, j, ...)

## S3 method for class 'dfr_dist_series'
component_hazard(x, j, ...)
```

Arguments

`x`	A system object (e.g., `dfr_dist_series`).
`j`	Component index (integer, 1 <= `j` <= `ncomponents(x)`).
`...`	Additional arguments passed to methods.

Details

The returned closure evaluates $h_j(t, \theta_j)$ for component `j`. The `par` argument accepts *component-local* parameters (not the full system parameter vector). This is useful for:

- Plotting individual hazard contributions
- Verifying that $\sum_j h_j(t) = h_{sys}(t)$
- Sensitivity analysis on a single component

Value

A closure `function(t, par = NULL, ...)` that evaluates component `j`'s hazard rate. If `par` is `NULL`, the component's default parameters (from the system) are used.

Methods (by class)

- `component_hazard(dfr_dist_series)`: Hazard closure for component j of a series system. Returns `function(t, par_j = NULL, ...)` where `par_j` are the component-local parameters.

See Also

`component` to extract the full component object, `hazard` for the system-level hazard, `dfr_dist_series` for the constructor

Other system introspection: `param_layout()`, `sample_components()`

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_exponential(0.2)
))

h1 <- component_hazard(sys, 1)
h2 <- component_hazard(sys, 2)
h_sys <- hazard(sys)

# Verify hazard sum property
t <- 10
h1(t) + h2(t)  # 0.3
h_sys(t)        # 0.3 (same!)
```

dfr_dist_series *Series System Distribution from DFR Components*

Description

Composes m dfr_dist component distributions into a series system distribution. A series system fails when any component fails, so the system hazard is the sum of component hazards: $h_{sys}(t) = \sum_j h_j(t)$.

Usage

```
dfr_dist_series(components, par = NULL, n_par = NULL)
```

Arguments

components	A list of dfr_dist objects representing system components.
par	Optional concatenated parameter vector $\theta = (\theta_1, \ldots, \theta_m)$. If NULL, parameters are concatenated from component objects.
n_par	Optional integer vector giving the number of parameters per component. Inferred from component par if not supplied; required when any component has NULL parameters.

Details

The resulting object inherits from dfr_dist, so all existing methods (hazard, survival, CDF, density, sampling, log-likelihood, MLE fitting) work automatically.

Parameter layout: Parameters are stored as a single concatenated vector. The $layout field maps global indices to component indices. For example, if component 1 has 2 parameters and component 2 has 1, then layout = list(1:2, 3).

Analytical cumulative hazard: If *all* components provide cum_haz_rate, the series system gets an analytical $H_{sys}(t) = \sum_j H_j(t)$. Otherwise, falls back to numerical integration.

Score and Hessian: Both use a decomposed per-component approach rather than numDeriv on the full system log-likelihood. When components provide analytical score_fn/hess_fn, the cumulative hazard derivatives are computed analytically via the all-censored trick; the hazard derivatives use numDeriv::jacobian (score) or numDeriv::hessian (Hessian) per-component. The Hessian exploits block structure: cross-component blocks use only the rate Jacobians already computed for the score.

Identifiability: Exponential series systems are *not* identifiable from system-level data alone — only the sum of rates is identifiable. When fitting to data, check sum(coef(result)) rather than individual rate parameters. Mixed-type series systems (e.g., Weibull + Gompertz) are generally identifiable because the components have different hazard shapes.

Nested series: A dfr_dist_series is itself a dfr_dist, so it can be used as a component in another series system. The resulting nested system's hazard is the sum of all leaf-component hazards.

Class hierarchy: dfr_dist_series inherits from dfr_dist -> likelihood_model -> univariate_dist -> dist. All methods from these parent classes work automatically.

Value

A `dfr_dist_series` object (inherits `dfr_dist`). Extra fields: `$components`, `$layout`, `$m`, `$n_par`.

See Also

`is_dfr_dist_series` for the type predicate, `ncomponents` and `component` for introspection, `param_layout` for parameter index mapping, `component_hazard` for per-component hazard closures, `sample_components` for sampling component lifetimes, `dfr_dist` for the parent class constructor, `hazard` for distribution generics

Other series system: `assumptions.dfr_dist_series()`, `is_dfr_dist_series()`, `print.dfr_dist_series()`

Examples

```
library(flexhaz)

# --- Basic exponential series ---
# Three exponential components -> equivalent to single exponential
sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_exponential(0.2),
    dfr_exponential(0.3)
))
# System hazard = 0.6 (constant)
h <- hazard(sys)
h(10)  # 0.6

# System survival at t = 5
S <- surv(sys)
S(5)   # exp(-0.6 * 5)

# --- Mixed Weibull + Gompertz series ---
sys2 <- dfr_dist_series(list(
    dfr_weibull(shape = 2, scale = 100),
    dfr_gompertz(a = 0.01, b = 0.1)
))
h2 <- hazard(sys2)
h2(50)  # sum of Weibull and Gompertz hazards at t=50

# --- Nested series ---
subsystem <- dfr_dist_series(list(
    dfr_exponential(0.05),
    dfr_exponential(0.10)
))
full_system <- dfr_dist_series(list(
    subsystem,
    dfr_weibull(shape = 2, scale = 200)
))

# --- Fitting workflow ---
solver <- fit(sys)
# result <- solver(df, par = c(0.1, 0.2, 0.3))
```

```
# coef(result)   # fitted parameters
# vcov(result)   # variance-covariance matrix
# logLik(result) # maximized log-likelihood
```

`is_dfr_dist_series` *Test whether an object is a dfr_dist_series*

Description

Returns `TRUE` if `x` inherits from `"dfr_dist_series"`, `FALSE` otherwise.

Usage

```
is_dfr_dist_series(x)
```

Arguments

`x`	Object to test.

Details

Since `dfr_dist_series` inherits from `dfr_dist`, an object that passes `is_dfr_dist_series()` will also pass `is_dfr_dist()`. Use this function when you need to distinguish series systems from ordinary `dfr_dist` objects.

Value

Logical scalar.

See Also

`dfr_dist_series` for the constructor, `is_dfr_dist` for the parent class predicate

Other series system: `assumptions.dfr_dist_series()`, `dfr_dist_series()`, `print.dfr_dist_series()`

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_exponential(0.2)
))
is_dfr_dist_series(sys)  # TRUE
is_dfr_dist(sys)         # also TRUE (inherits dfr_dist)

single <- dfr_exponential(0.5)
is_dfr_dist_series(single)  # FALSE
is_dfr_dist(single)         # TRUE
```

```
is_dfr_dist_series(42)  # FALSE
```

`param_layout` *Get the parameter layout for a system*

Description

Returns the mapping from global (flat) parameter indices to per-component parameter indices, enabling the series system to distribute a single parameter vector across its components.

Usage

```
param_layout(x, ...)

## S3 method for class 'dfr_dist_series'
param_layout(x, ...)
```

Arguments

`x`	A system object (e.g., `dfr_dist_series`).
`...`	Additional arguments passed to methods.

Details

Parameters across all components are stored as a single concatenated vector $\theta = (\theta_1, \ldots, \theta_m)$. The layout maps global indices back to each component. For example, with:

- Component 1: Weibull (shape, scale) — 2 parameters
- Component 2: Exponential (rate) — 1 parameter
- Component 3: Gompertz (a, b) — 2 parameters

the layout is `list(1:2, 3, 4:5)`, so the global parameter vector `c(shape1, scale1, rate2, a3, b3)` gets sliced as `par[1:2]` for component 1, `par[3]` for component 2, and `par[4:5]` for component 3.

This design enables standard optimizers to work on a flat vector while the series system internally distributes parameters to the correct components.

Value

A list of integer vectors, one per component, containing global parameter indices.

Methods (by class)

- `param_layout(dfr_dist_series)`: Parameter index mapping for a series system.

See Also

component to extract a component with its parameters, params to get the full parameter vector, dfr_dist_series for the constructor

Other system introspection: component_hazard(), sample_components()

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_weibull(shape = 2, scale = 100),
    dfr_exponential(0.05),
    dfr_gompertz(a = 0.01, b = 0.1)
))
param_layout(sys)
# list(1:2, 3, 4:5)
```

---

print.dfr_dist_series *Print method for series system distributions*

Description

Displays a human-readable summary of a series system distribution, including the number of components, per-component parameter counts and values, and the system hazard/survival formulas.

Usage

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

Arguments

x	A dfr_dist_series object.
...	Additional arguments (unused).

Details

The output includes:

- Header with the number of components
- One line per component showing its parameter count and current parameter values (or "unknown" if parameters are NULL)
- The system hazard formula: $h_{sys}(t) = \sum_j h_j(t, \theta_j)$
- The system survival formula: $S_{sys}(t) = \prod_j S_j(t, \theta_j)$

Value

Invisibly returns `x`.

See Also

`dfr_dist_series` for the constructor

Other series system: `assumptions.dfr_dist_series()`, `dfr_dist_series()`, `is_dfr_dist_series()`

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_weibull(shape = 2, scale = 100)
))
print(sys)
# Series system distribution with 2 components
#   Component 1: 1 param(s) [0.1]
#   Component 2: 2 param(s) [2, 100]
# System hazard: h_sys(t) = sum_j h_j(t, theta_j)
# Survival: S_sys(t) = exp(-H_sys(t)) = prod_j S_j(t, theta_j)
```

`sample_components` *Sample component lifetimes from a system*

Description

Generates an $n \times m$ matrix where column j contains independent samples from component j's lifetime distribution. The system lifetime is the row-wise minimum.

Usage

```
sample_components(x, n, ...)

## S3 method for class 'dfr_dist_series'
sample_components(x, n, par = NULL, ...)
```

Arguments

`x`	A system object (e.g., `dfr_dist_series`).
`n`	Number of samples (rows).
`...`	Additional arguments passed to methods.
`par`	Optional parameter vector override.

Details

Each column is sampled independently using the component's own sampler. Since the series system fails when *any* component fails, the system lifetime for each observation is:

```
t_sys <- apply(mat, 1, min)
```

The failing component for each observation can be identified via:

```
failing <- apply(mat, 1, which.min)
```

This enables failure attribution analysis: what proportion of system failures are caused by each component?

Value

An $n \times m$ numeric matrix of component lifetimes, with columns named `comp1`, `comp2`, etc.

Methods (by class)

- `sample_components(dfr_dist_series)`: Sample component lifetimes from a series system. Returns an n x m matrix where column j holds samples from component j. The system lifetime is `apply(mat, 1, min)`.

See Also

`sampler` for system-level sampling, `component` to extract individual component objects, `dfr_dist_series` for the constructor

Other system introspection: `component_hazard()`, `param_layout()`

Examples

```
library(flexhaz)

sys <- dfr_dist_series(list(
    dfr_exponential(0.1),
    dfr_exponential(0.2),
    dfr_exponential(0.3)
))

set.seed(42)
mat <- sample_components(sys, n = 1000)
dim(mat)  # 1000 x 3

# System lifetimes
t_sys <- apply(mat, 1, min)

# Which component caused each failure?
failing <- apply(mat, 1, which.min)
table(failing) / 1000
# Proportions ~= c(1/6, 2/6, 3/6) for rates (0.1, 0.2, 0.3)
```

Index