Package: midi (via r-universe)

August 29, 2024

Title Microstructure Information from Diffusion Imaging

Version 0.1.0.9000

Description An implementation of a taxonomy of models of restricted diffusion in biological tissues parametrized by the tissue geometry (axis, diameter, density, etc.). This is primarily used in the context of diffusion magnetic resonance (MR) imaging to model the MR signal attenuation in the presence of diffusion gradients. The goal is to provide tools to simulate the MR signal attenuation predicted by these models under different experimental conditions. The package feeds a companion 'shiny' app available at <`https://midi-pastrami.apps.math.cnrs.fr`> that serves as a graphical interface to the models and tools provided by the package. Models currently available are the ones in Neuman (1974) <doi:10.1063/1.1680931>, Van Gelderen et al. (1994) <doi:10.1006/jmrb.1994.1038>, Stanisz et al. (1997) <doi:10.1002/mrm.1910370115>, Soderman & Jonsson (1995) <doi:10.1006/jmra.1995.0014> and Callaghan (1995) <doi:10.1006/jmra.1995.1055>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 3.5)

URL `https://github.com/lmjl-alea/midi`, `https://lmjl-alea.github.io/midi/`

BugReports `https://github.com/lmjl-alea/midi/issues`

Imports cli, ggplot2, plotly, purrr, R6, rlang, withr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://lmjl-alea.r-universe.dev

RemoteUrl https://github.com/lmjl-alea/midi

RemoteRef HEAD

RemoteSha e6a4f13e78e6eff2d3e0e6253d0b681f0a856fca

Contents

`autoplot.bundle` *Plots a cross section of a cylinder bundle using* **ggplot2**

Description

Plots a cross section of a cylinder bundle from an object of class `bundle` as generated by `simulate_bundle()` using **ggplot2**.

Usage

```
## S3 method for class 'bundle'
autoplot(object, grid_size = 100L, ...)
```

Arguments

`object`	An object of class `bundle` as generated by `simulate_bundle()`.
`grid_size`	An integer value specifying the number of points on which the unit circle should be discretized to draw the spheres. Defaults to `100L`.
`...`	Additional arguments to be passed to the `ggplot2::autoplot()` method.

Value

A ggplot2::ggplot() object.

Examples

```
density <- 0.5
voxel_size <- 5 # micrometers
withr::with_seed(1234, {
  out <- simulate_bundle(density, voxel_size)
})
ggplot2::autoplot(out)
```

---

bvalue *B-Value Calculation*

Description

A function to calculate the b-value for a given set of experimental parameters.

Usage

```
bvalue(small_delta, big_delta, G)
```

Arguments

small_delta	A numeric value specifying the duration of the gradient pulse in ms.
big_delta	A numeric value specifying the duration between the gradient pulses in ms.
G	A numeric value specifying the strength of the gradient in μT/μm.

Value

A numeric value storing the predicted b-value in ms/μm^2.

Examples

```
bvalue(small_delta = 30, big_delta = 30, G = 0.040)
```

`CallaghanCompartment` *Callaghan's model for restricted diffusion in a cylinder*

Description

A class to model restricted diffusion in a cylinder using the Callaghan's model.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `CallaghanCompartment`

Methods

Public methods:

- `CallaghanCompartment$clone()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
CallaghanCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

References

Callaghan, P. T. (1995). Pulsed-gradient spin-echo NMR for planar, cylindrical, and spherical pores under conditions of wall relaxation. Journal of magnetic resonance, Series A, 113(1), 53-59.

`CylinderBundleCompartment` *Cylinder bundle compartment class*

Description

A class to model restricted diffusion in a bundle of cylinders.

Super class

`midi::BaseCompartment` -> `CylinderBundleCompartment`

Methods

Public methods:

- `CylinderBundleCompartment$new()`
- `CylinderBundleCompartment$clone()`

Method `new()`**:** Instantiates a new cylinder bundle compartment.

Usage:

```
CylinderBundleCompartment$new(
  cylinder_density,
  cylinder_compartments,
  axial_diffusivity = NULL,
  radial_diffusivity = NULL
)
```

Arguments:

`cylinder_density` A numeric value specifying the density of the cylinders in the voxel. Must be between 0 and 1.

`cylinder_compartments` A list of instances of the `CylinderCompartment` class specifying the compartments within the bundle.

`axial_diffusivity` A numeric value specifying the axial diffusivity in the space outside the cylinders in $m^2.s^{-1}$. If not provided, defaults to a tortuosity model reducing the intrinsic diffusivity depending on orientation dispersion. Defaults to `NULL` in which case the extracellular axial diffusivity is set via a tortuosity model based on the dispersion in orientation.

`radial_diffusivity` A numeric value specifying the radial diffusivity in the space outside the cylinders in $m^2.s^{-1}$. Defaults to `NULL` in which case the extracellular radial diffusivity is set via a tortuosity model based on the intracellular density.

Returns: An instance of the `CylinderBundleCompartment` class.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
CylinderBundleCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
withr::with_seed(1234, {
  cyls <- rcylinders(
    n = 100,
    axis_mean = c(0, 0, 1),
    radius_mean = 5,
    diffusivity_mean = 3,
    axis_concentration = 10,
    radius_sd = 1,
    diffusivity_sd = 0
  )
})
```

```
comp <- CylinderBundleCompartment$new(
  cylinder_density = 0.5,
  cylinder_compartments = cyls
)

comp$get_signal(
  small_delta = 30,
  big_delta = 30,
  G = 0.040,
  direction = c(0, 0, 1)
)

comp$get_parameters()
```

`CylinderCompartment` *Cylinder compartment class*

Description

A class to model restricted diffusion in a cylinder.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `CylinderCompartment`

Methods

Public methods:

- `CylinderCompartment$new()`
- `CylinderCompartment$clone()`

Method `new()`**:** Instantiates a new cylinder compartment.

Usage:

```
CylinderCompartment$new(
  axis = c(0, 0, 1),
  restricted_compartment = VanGelderenCompartment$new()
)
```

Arguments:

`axis` A length-3 numeric vector specifying the axis of the cylinder.

`restricted_compartment` An instance of the `CircularlyShapedCompartment` class specifying the restricted compartment within the sphere. Defaults to a Van Gelderen compartment.

Returns: An instance of the `CylinderCompartment` class.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
CylinderCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
cylComp <- CylinderCompartment$new()
cylComp$get_signal(small_delta = 30, big_delta = 30, G = 0.040)
cylComp$get_parameter_names()
cylComp$get_parameters()
```

`FreeCompartment` *Free compartment class*

Description

A class to model free unconstrained diffusion.

Super class

`midi::BaseCompartment` -> `FreeCompartment`

Methods

Public methods:

- `FreeCompartment$new()`
- `FreeCompartment$clone()`

Method `new()`**:** Instantiates a new free compartment.

Usage:

```
FreeCompartment$new(diffusivity = NULL)
```

Arguments:

`diffusivity` A numeric value specifying the diffusivity within the sphere in μm^2.ms^{-1}. Defaults to `NULL`, in which case the default free diffusivity of 3 μm^2.ms^{-1} is used.

Returns: An instance of the `FreeCompartment` class.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
FreeCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`GammaDistribution` *Gamma distribution class*

Description

This class defines the gamma distribution. It provides methods for fitting the distribution to data and generating random samples.

Super class

`midi::BaseDistribution` -> `GammaDistribution`

Methods

Public methods:

- `GammaDistribution$new()`
- `GammaDistribution$get_shape()`
- `GammaDistribution$get_scale()`
- `GammaDistribution$clone()`

Method `new()`**:** Creates a new gamma distribution.

Usage:

```
GammaDistribution$new(shape = 1, scale = 1)
```

Arguments:

`shape` A numeric value specifying the shape parameter of the gamma distribution. Defaults to `1.0`.

`scale` A numeric value specifying the scale parameter of the gamma distribution. Defaults to `1.0`.

Returns: An instance of the gamma distribution as an object of class `GammaDistribution`.

Method `get_shape()`**:** Retrieves the shape parameter of the gamma distribution.

Usage:

```
GammaDistribution$get_shape()
```

Returns: A numeric value storing the shape parameter of the gamma distribution.

Method `get_scale()`**:** Retrieves the scale parameter of the gamma distribution.

Usage:

```
GammaDistribution$get_scale()
```

Returns: A numeric value storing the scale parameter of the gamma distribution.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GammaDistribution$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
gd <- GammaDistribution$new(
  shape = 1,
  scale = 5
)
gd$get_shape()
gd$get_scale()
gd$get_mean()
gd$get_variance()
gd$random(10)
gd$fit(gd$random(100))
gd$get_shape()
gd$get_scale()
```

`NeumanCompartment` *Neuman's model for restricted diffusion in a cylinder*

Description

A class to model restricted diffusion in a cylinder using the Neuman's model.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `NeumanCompartment`

Methods

Public methods:

- `NeumanCompartment$clone()`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
NeumanCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

References

Neuman, C. H. (1974). Spin echo of spins diffusing in a bounded medium. The Journal of Chemical Physics, 60(11), 4508-4511.

plot.bundle — *Plots a cross section of a cylinder bundle*

Description

Plots a cross section of a cylinder bundle

Usage

```
## S3 method for class 'bundle'
plot(x, grid_size = 100L, ...)
```

Arguments

x	An object of class bundle as generated by `simulate_bundle()`.
grid_size	An integer value specifying the number of points on which the unit circle should be discretized to draw the spheres. Defaults to `100L`.
...	Additional arguments to be passed to the `ggplot2::autoplot()` method.

Value

Nothing.

Examples

```
density <- 0.5
voxel_size <- 5 # micrometers
withr::with_seed(1234, {
  out <- simulate_bundle(density, voxel_size)
})
plot(out)
```

plot3d — *Plots a 3D representation of a cylinder bundle using* **plotly**

Description

Plots a 3D representation of a cylinder bundle from an object of class bundle as generated by `simulate_bundle()` using **plotly**.

Usage

```
plot3d(b, show_linear_mesh = FALSE)
```

Arguments

`b`	An object of class `bundle` as generated by `simulate_bundle()`.
`show_linear_mesh`	A logical value indicating whether the linear mesh of each cylinder should be displayed. Defaults to `FALSE` for computational efficiency.

Value

An HTML widget of class `plotly::plotly` storing the 3D visualization of the cylinder bundle.

Examples

```
density <- 0.5
voxel_size <- 5 # micrometers
withr::with_seed(1234, {
  out <- simulate_bundle(density, voxel_size)
})
plot3d(out)
```

`rcylinders` *Sample cylinder compartments*

Description

This function samples n cylinder compartments with given axis, radius and diffusivity distributions.

Usage

```
rcylinders(
  n,
  axis_mean,
  radius_mean,
  diffusivity_mean,
  axis_concentration = Inf,
  radius_sd = 0,
  diffusivity_sd = 0,
 restricted_model = c("Callaghan", "Neuman", "Soderman", "Stanisz", "Van Gelderen")
)
```

Arguments

`n`	An integer value specifying the number of compartments to sample.
`axis_mean`	A numeric value specifying the mean of the axis distribution.
`radius_mean`	A numeric value specifying the mean of the radius distribution.
`diffusivity_mean`	A numeric value specifying the mean of the diffusivity distribution.

`axis_concentration`
: A numeric value specifying the concentration of the axis distribution. Defaults to `Inf` in which case the axis is fixed to the mean value.

`radius_sd`
: A numeric value specifying the standard deviation of the radius distribution. Defaults to `0` in which case the radius is fixed to the mean value.

`diffusivity_sd`
: A numeric value specifying the standard deviation of the diffusivity distribution. Defaults to `0` in which case the diffusivity is fixed to the mean value.

`restricted_model`
: A character vector specifying the restricted diffusion model to use. Defaults to `callaghan`. Possible values are `callaghan`, `neuman`, `soderman`, `stanisz`, and `vangelderen`.

Details

The axis distribution is given by a mean and a concentration parameter and the Dimroth-Watson distribution is used to sample values. The radius and diffusivity distributions are given by a mean and a standard deviation and the Gamma distribution is used to sample values. If the concentration parameter is set to `Inf` the axis is fixed to the mean value. If the standard deviation of the radius and diffusivity distributions are set to `0` the radius and diffusivity are fixed to the mean values. If all parameters are fixed, only one compartment is sampled.

Value

A list of n cylinder compartments of class `CylinderCompartment`.

Examples

```
# Sample 10 cylinder compartments with fixed axis, radius and diffusivity
# set.seed(42)
cyl_distr <- rcylinders(
  n = 10L,
  axis_mean = c(0, 0, 1),
  radius_mean = 5,
  diffusivity_mean = 3
)
```

`run_app` *Runs the MIDI Shiny web application*

Description

This is a helper function to run the MIDI Shiny web application in the default web browser.

Usage

```
run_app()
```

Value

Nothing, but launches the Shiny app in the default web browser.

Examples

```
run_app()
```

---

simulate_bundle *Generates a cross section of a cylinder bundle*

Description

Generates a cross section of a cylinder bundle with a given density and voxel size. The cross section is generated by simulating a random distribution of cylinders and computing the intersection of the cylinders with a plane. The radii of the cylinders are drawn from a Gamma distribution fitted from data retrieved by histology in the genu of the corpus callosum (Aboitiz et al., 1992). The number of cylinders is determined by the density and the voxel size.

Usage

```
simulate_bundle(
  density = 0.5,
  voxel_size = 10,
  rel_tol = 0.001,
  verbose = FALSE
)
```

Arguments

`density`	A numeric value between 0 and 1 specifying the density of the cylinders in the voxel. Defaults to `0.5`.
`voxel_size`	A numeric value specifying the size of the voxel in micro- meters. Defaults to `10`.
`rel_tol`	A numeric value specifying the relative tolerance to reach the target volume defined as `density * voxel_size^2`. Defaults to `1e-3`.
`verbose`	A logical value specifying whether to print messages. Defaults to `FALSE`.

Value

A list with the following components:

- `sections`: A numeric matrix with 3 columns:
  - `x`: The x-coordinates of the centers of the cylinders;
  - `y`: The y-coordinates of the centers of the cylinders;
  - `r`: The radii of the cylinders in micrometers.
- `voxel_size`: The size of the voxel in micrometers

References

Aboitiz, F., Scheibel, A. B., Fisher, R. S., & Zaidel, E. (1992). Fiber composition of the human corpus callosum. Brain research, 598(1-2), 143-153.

Examples

```
density <- 0.5
voxel_size <- 5 # micrometers
withr::with_seed(1234, {
  out <- simulate_bundle(density, voxel_size)
})

# Actual density in the simulated substrate
sum(out$sections[, "r"]^2 * pi) / voxel_size^2
```

`SodermanCompartment` *Soderman's model for restricted diffusion in a cylinder*

Description

A class to model restricted diffusion in a cylinder using the Soderman's model.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `SodermanCompartment`

Methods

Public methods:

- `SodermanCompartment$clone()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`SodermanCompartment$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

References

Söderman, O., & Jönsson, B. (1995). Restricted diffusion in cylindrical geometry. Journal of Magnetic Resonance, Series A, 117(1), 94-97.

`SphereCompartment` *Sphere compartment class*

Description

A class to model restricted diffusion in a sphere.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `SphereCompartment`

Methods

Public methods:

- `SphereCompartment$new()`
- `SphereCompartment$clone()`

Method `new()`**:** Instantiates a new sphere compartment.

Usage:

```
SphereCompartment$new(restricted_compartment = VanGelderenCompartment$new())
```

Arguments:

`restricted_compartment` An instance of the `CircularlyShapedCompartment` class specifying the restricted compartment within the sphere. Defaults to a Van Gelderen compartment.

Returns: An instance of the `SphereCompartment` class.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
SphereCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
sphComp <- SphereCompartment$new()
sphComp$get_signal(small_delta = 30, big_delta = 30, G = 0.040)
sphComp$get_parameter_names()
sphComp$get_parameters()
```

`StaniszCompartment` *Stanisz's model for restricted diffusion in a cylinder*

Description

A class to model restricted diffusion in a cylinder using the Stanisz's model.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `StaniszCompartment`

Methods

Public methods:

- `StaniszCompartment$clone()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
StaniszCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

References

Stanisz, G. J., Wright, G. A., Henkelman, R. M., & Szafer, A. (1997). An analytical model of restricted diffusion in bovine optic nerve. Magnetic Resonance in Medicine, 37(1), 103-111.

`VanGelderenCompartment` *Van Gelderen's model for restricted diffusion in a cylinder*

Description

A class to model restricted diffusion in a cylinder using the Van Gelderen's model.

Super classes

`midi::BaseCompartment` -> `midi::CircularlyShapedCompartment` -> `VanGelderenCompartment`

Methods

Public methods:

- `VanGelderenCompartment$clone()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
VanGelderenCompartment$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

References

Vangelderen, P., DesPres, D., Vanzijl, P. C. M., & Moonen, C. T. W. (1994). Evaluation of restricted diffusion in cylinders. Phosphocreatine in rabbit leg muscle. Journal of Magnetic Resonance, Series B, 103(3), 255-260.

`WatsonDistribution` *Watson distribution class*

Description

This class defines the Watson distribution. It provides methods for fitting the distribution to data and generating random samples.

Super class

`midi::BaseDistribution` -> `WatsonDistribution`

Methods

Public methods:

- `WatsonDistribution$new()`
- `WatsonDistribution$get_axis()`
- `WatsonDistribution$get_concentration()`
- `WatsonDistribution$clone()`

Method `new()`**:** Creates a new Watson distribution.

Usage:

```
WatsonDistribution$new(mu = c(0, 0, 1), kappa = 10)
```

Arguments:

`mu` A numeric vector of length 3 specifying the mean direction of the Watson distribution. Defaults to `(0, 0, 1)`.

`kappa` A numeric value specifying the concentration parameter of the Watson distribution. Defaults to `10.0`.

Returns: An instance of the Watson distribution as an object of class `WatsonDistribution`.

Method `get_axis()`**:** Retrieves the mean axis of the Watson distribution.

Usage:

```
WatsonDistribution$get_axis()
```

Returns: A numeric vector of length 3 storing the mean axis of the Watson distribution.

Method `get_concentration()`**:** Retrieves the concentration parameter of the Watson distribution.

Usage:

```
WatsonDistribution$get_concentration()
```

Returns: A numeric value storing the concentration parameter of the Watson distribution.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
WatsonDistribution$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
wd <- WatsonDistribution$new(
  mu = c(0, 0, 1),
  kappa = 10
)
wd$get_axis()
wd$get_concentration()
wd$random(10)
wd$fit(wd$random(100))
wd$get_axis()
wd$get_concentration()
```

Index