Package ‘CancerInSilico’

December 21, 2016

Type Package
Title An R interface for computational modeling of tumor progression
Version 1.0.0
Date 2016-06-01
Author Thomas D. Sherman, Raymond Cheng, Elana J. Fertig
Maintainer Thomas D. Sherman &lt;tsherma4@jhu.edu&gt;, Elana J. Fertig &lt;ejfertig@jhmi.edu&gt;
Description The CancerInSilico package provides an R interface for running mathematical models of tumor progression. This package has the underlying models implemented in C++ and the output and analysis features implemented in R.
License GPL (&gt;= 2)
Imports methods, grDevices, graphics, stats
Depends Rcpp
LinkingTo Rcpp, testthat, BH
Suggests testthat, knitr, rmarkdown, BiocStyle
VignetteBuilder knitr
biocViews MathematicalBiology, SystemsBiology, CellBiology, BiomedicalInformatics
RoxygenNote 5.0.1
NeedsCompilation yes

R topics documented:

CancerInSilico .................................................. 2
CellModel-class ................................................ 2
getAxisAngle .................................................... 3
getAxisLength .................................................. 3
getCoordinates ............................................... 4
getCycleLengths ............................................. 4
getDensity ..................................................... 5
getGrowthRates .............................................. 5
getNumberOfCells ......................................... 6
getParameters ............................................... 6
**CellModel-class**

getRadii .......................................................... 7
interactivePlot .................................................. 7
plotCells .......................................................... 8
runCancerSim ...................................................... 8
runDrasdoHohme .................................................. 9
show.CellModel-method ....................................... 10
timeToRow ......................................................... 10

**Index**

---

**CancerInSilico**

---

**Description**

Package:         CellModel
Type:            Package
Version:         0.99.0
Date:            2016-06-24
License:         LGPL

**Author(s)**

Maintainer: Elana J. Fertig <ejfertig@jhmi.edu>, Thomas D. Sherman <tsherma4@jhu.edu>

---

**CellModel-class**

---

**Description**

An S4 class to represent the output of a cell-based model

**Slots**


mInitialNumCells the initial number of cells in the model

mRunTime the total run time (hours) of the model

mInitialDensity the density the cells were seeded at

mInheritGrowth whether or not cells inherit growth rates from their parent

mOutputIncrement the frequency of print statements during the run

mRandSeed the random seed
**getAxisAngle**

getAxisAngle get the axis angle of each cell

**Description**
getAxisAngle get the axis angle of each cell

**Usage**
getAxisAngle(model, time)

**Arguments**
- model : A CellModel
- time : time in model hours

**Value**
vector containing the axis angle of each cell at time

**getAxisLength**

getAxisLength get the axis length of each cell

**Description**
getAxisLength get the axis length of each cell

**Usage**
getAxisLength(model, time)

**Arguments**
- model : A CellModel
- time : time in model hours

**Value**
vector containing the axis length of each cell at time
**getCoordinates**

getCoordinates get a two dimensional matrix of all the cell coordinates

**Usage**

getCoordinates(model, time)

**Arguments**

model A CellModel

time time in model hours

**Value**

an N X 2 matrix of cell coordinates at time

---

**getCycleLengths**

getCycleLengths return the cycle lengths of each cells at time

**Usage**

getCycleLengths(model, time)

**Arguments**

model a CellModel object

time time in model hours

**Value**

the cycle lengths of each cell at time

**Examples**

getchCycleLengths(runCancerSim(1,1), 1)
**getDensity**

getDensity gets the density of cells at a given time

**Usage**

getDensity(model, time)

**Arguments**

- **model**: A Cell Model
- **time**: time in model hours

**Value**

The density of cells at that time (not quite the same as confluency)

**Examples**

getDensity(runCancerSim(1,1),1)

---

**getGrowthRates**

getGrowthRates get the model growth rates of each cell

**Description**

getGrowthRates get the model growth rates of each cell

**Usage**

getGrowthRates(model, time)

**Arguments**

- **model**: A CellModel
- **time**: time in model hours

**Value**

vector containing the growth rate of each cell at time
**getNumberOfCells**

*get the number of cells alive*

**Usage**

```r
getNumberOfCells(model, time)
```

**Arguments**

- `model` A CellModel
- `time` time in model hours

**Value**

the number of cells at this time

**Examples**

```r
getNumberOfCells(runCancerSim(1,1), 1)
```

---

**getParameters**

*get a named list of parameters in the model*

**Usage**

```r
getParameters(model, fullDist = FALSE)
```

**Arguments**

- `model` A CellModel
- `fullDist` [bool] return full distribution of cycle length

**Value**

a named list of parameters in the model

**Examples**

```r
getParameters(runCancerSim(1,1))
```
getRadii

getRadii get the radius of each cell

Description
getRadii get the radius of each cell

Usage
getRadii(model, time)

Arguments
model A CellModel
time time in model hours

Value
vector containing the radius of each cell at time

interactivePlot

interactivePlot plots a CellModel and allows the user to control the plot with various commands

Description
interactivePlot plots a CellModel and allows the user to control the plot with various commands

Usage
interactivePlot(model, time = 0)

Arguments
model A CellModel
time time in model hours

Value
plot a visual representation of cells that takes in command-line-like inputs, type 'h' for help and a list of all available commands
**Description**

`plotCell` plots a `CellModel` at a given time

**Usage**

```r
plotCells(model, time)
```

**Arguments**

- `model`: A `CellModel`
- `time`: time in model hours

**Value**

plot a visual representation of cells

**Examples**

```r
plotCells(runCancerSim(10, 1), 1)
```

---

**Description**

`runCancerSim` runs a cell-based model of cancer

**Usage**

```r
runCancerSim(initialNum, runTime, density = 0.01, cycleLengthDist = 12,
              inheritGrowth = FALSE, outputIncrement = 6, randSeed = 0,
              modelType = "DrasdoHohme2003", ...)
```

**Arguments**

- `initialNum`: how many cells initially (integer)
- `runTime`: how long the simulation runs (model hours)
- `density`: the density the cells are seeded at, must be in (0, 0.1]
- `cycleLengthDist`: cycle time distribution
- `inheritGrowth`: whether or not daughter cells have the same cycle-length as parents
- `outputIncrement`: time increment to print status at
- `randSeed`: seed for the model
- `modelType`: the name of the cell-based model to use
- `...`: model specific parameters (depends on `modelType`)
This function provides a centralized R interface to run C++ code for cell-based models implemented in this package. Standard parameters, as well as model-specific parameters, are passed in to this function along with a model name. This function then runs the model and returns a CellModel object containing all the information from the model. This object can then be accessed with various functions designed to interact with the class. To see a list of available functions, there is a `show()` command implemented for CellModel objects.

**Value**

A CellModel containing all info from the model run

**Examples**

```r
runCancerSim(1, 4)
```

---

**Description**

`runDrasdoHohme` runs the model based on Drasdo and Hohme (2003)

**Usage**

```r
runDrasdoHohme(initialNum, runTime, density, cycleLengthDist, inheritGrowth, outputIncrement, randSeed, ...)
```

**Arguments**

- `initialNum`: how many cells initially
- `runTime`: how long the simulation represents in realtime
- `density`: the density the cells are seeded at
- `cycleLengthDist`: cycle time distribution
- `inheritGrowth`: whether or not daughter cells have the same cycle-length as parents
- `outputIncrement`: time increment to print status at
- `randSeed`: seed for the model
- `...`: nG, epsilon parameters (specific to this model)

**Details**

This function calls the C++ implementation of the Drasdo and Hohme (2003) model.

**Value**

A CellModel containing all info from the model run
show, CellModel-method  
show display summary of CellModel class

Description
show display summary of CellModel class

Usage
## S4 method for signature 'CellModel'
show(object)

Arguments
object A CellModel Object

Value
shows all available functions and parameters of model

Examples
show(runCancerSim(1,1))

timeToRow

timeToRow return the correct row in the mCells list corresponding to a given time

Description
timeToRow return the correct row in the mCells list corresponding to a given time

Usage
timeToRow(model, time)

Arguments
model A CellModel
time time in model hours

Value
corresponding row in mCells list
Index

CancerInSilico, 2
CancerInSilico-package
   (CancerInSilico), 2
CellModel-class, 2

getAxisAngle, 3
getAxisLength, 3
getCoordinates, 4
getCycleLengths, 4
getDensity, 5
getGrowthRates, 5
getNumberOfCells, 6
getParameters, 6
getRadii, 7

interactivePlot, 7

plotCells, 8

runCancerSim, 8
runDrasdoHohme, 9

show, CellModel-method, 10

timeToRow, 10