

# SBMLR

April 20, 2011

---

Ops.SBML

*Check the equality of the species and reactions of two SBML models*

---

## Description

This function tests the equivalence of two models with respect to the species and reaction data frames generated by summary.

## Usage

```
## S3 method for class 'SBML':  
Ops(e1, e2)
```

## Arguments

e1            The first of the two model objects of class SBML which are to be compared.  
e2            The second model object.

## Value

A list containing the following two boolean dataframes

species        The equality of species information tabularized as a data frame.  
reactions     The equality of reaction information tabularized as a dataframe.

## Author(s)

Tom Radivoyevitch

## See Also

[summary.SBML](#)

## Examples

```
library(SBMLR)  
curto1=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))  
curto2=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.xml"))  
curto1==curto2
```

---

`readSBML`*Convert an SBML file into an R model object of class SBML*

---

## Description

This function converts an SBML level 2 file into a corresponding R model structure of class SBML.

## Usage

```
readSBML(filename)
```

## Arguments

`filename` An SBML level 2 model input file.

## Details

A limited subset of SBML level 2 models is currently supported, e.g. events and function definitions are not covered.

## Value

A corresponding SBML model object in R.

## Note

This function replaces `read.SBML` of older versions.

## Author(s)

Tom Radivoyevitch

## See Also

[readSBMLR](#)

## Examples

```
library(SBMLR)
library(odesolve)
curtoX=readSBML(file.path(system.file(package="SBMLR"), "models/curto.xml"))
curtoR=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
curtoX==curtoR
```

---

`readSBMLR`*Convert an SBMLR file into an R model object of class SBML*

---

**Description**

This function converts an SBMLR model definition in filename into a corresponding returned SBML model structure.

**Usage**

```
readSBMLR(filename)
```

**Arguments**

`filename` An SBMLR model definition file.

**Details**

A limited subset of SBML level 2 models is currently supported, e.g. events and function definitions are not covered.

**Value**

A corresponding SBML model object in R.

**Note**

This function replaces the use of `source` in older versions of SBMLR. It includes rate law and rule string to function, expression and MathML mappings.

**Author(s)**

Tom Radivoyevitch

**See Also**

[readSBML](#)

**Examples**

```
library(SBMLR)
library(odesolve)
curtoX=readSBML(file.path(system.file(package="SBMLR"), "models/curto.xml"))
curtoR=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
curtoX==curtoR
```

---

`saveSBML`*Saves an R model object as an SBML file*

---

**Description**

This function converts a class `SBML` model object in R into an SBML level 2 file.

**Usage**

```
saveSBML(model, filename)
```

**Arguments**

<code>model</code>	The model object in R.
<code>filename</code>	The name of the SBML file

**Details**

The output file is SBML level 2.

**Value**

No value returned.

**Warning**

SBML events and function definitions are NOT implemented.

**Note**

For speed, the SBML file is written incrementally, rather than first built as a DOM in R and then saved using `xmlSave`.

**Author(s)**

Tom Radivoyevitch

**References**

Radivoyevitch, T. A two-way interface between limited Systems Biology Markup Language and R. *BMC Bioinformatics* 5, 190 (2004).

**See Also**

[saveSBMLR](#)

**Examples**

```
library(SBMLR)
library(odesolve)
curtoR=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
saveSBML(curtoR, "curtoR.xml")
curtoX=readSBML("curtoR.xml")
curtoX==curtoR
summary(curtoR)
unlink("curtoR.xml")
```

---

saveSBMLR

*Save an R model object of class SBML as an SBMLR file*

---

**Description**

This function converts SBML model object in R into an SBMLR model definition file.

**Usage**

```
saveSBMLR(model, filename)
```

**Arguments**

model	The SBML model object to be mapped into the SBMLR model definition file.
filename	The file name of the destination SBMLR model definition file.

**Value**

No value returned.

**Warning**

SBML events and function definitions are NOT implemented.

**Note**

Similar to saveSBML, the file is written incrementally.

**Author(s)**

Tom Radivoyevitch

**References**

Radivoyevitch, T. A two-way interface between limited Systems Biology Markup Language and R. BMC Bioinformatics 5, 190 (2004).

**See Also**

[saveSBML](#)

**Examples**

```

library(SBMLR)
library(odesolve)
curto=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
saveSBMLR(curto, "curtoR.r")
curtoR=readSBMLR("curtoR.r")
curto==curtoR
summary(curtoR)
unlink("curtoR.r")

```

simulate

*Simulate a model of class SBML***Description**

This function simulates a model given the report times and optional modulators. It uses `lsoda` of the `odesolve` package.

**Usage**

```
simulate(model, times, modulator=NULL, X0=NULL, ...)
```

**Arguments**

<code>model</code>	The model object to be simulated. Initial conditions are passed through this object.
<code>times</code>	The sequence of time points to be sampled and provided as rows of the output matrix.
<code>modulator</code>	Null if there are no modulators (default), a vector of numbers if there are steady state Vmax modulators, and a list of interpolating functions if there are time course Vmax modulators.
<code>X0</code>	Override model initial conditions in simulations, particularly piece-wise perturbation simulations.
<code>...</code>	For compatibility with <code>simulate</code> of the <code>stats</code> package.

**Details**

This is a wrapper for `lsoda`.

**Value**

The data frame output that comes out of `lsoda`.

**Note**

Rules are implemented through time varying boundary conditions updated at each time point as a side effect within the (now internal) function `fderiv`.

**Author(s)**

Tom Radivoyevitch

## References

For the folate cycle example given below: Morrison PF, Allegra CJ: Folate cycle kinetics in human breast cancer cells. *JBiolChem* 1989, 264(18):10552-10566.

## Examples

```
##---- The following example performs a perturbation in PRPP from 5 to 50 uM in Curto et
library(SBMLR)
library(odesolve)
curto=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.xml"))
out1=simulate(curto,seq(-20,0,1))
curto$species$PRPP$ic=50
out2=simulate(curto,0:70)
outs=data.frame(rbind(out1,out2))
attach(outs)
par(mfrow=c(2,1))
plot(time,IMP,type="l")
plot(time,HX,type="l")
par(mfrow=c(1,1))
detach(outs)

# which should be the same plots as
curto=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
out1=simulate(curto,seq(-20,0,1))
curto$species$PRPP$ic=50
out2=simulate(curto,0:70)
outs=data.frame(rbind(out1,out2))
attach(outs)
par(mfrow=c(2,1))
plot(time,IMP,type="l")
plot(time,HX,type="l")
par(mfrow=c(1,1))
detach(outs)

##---- The following example uses fderiv to generate Morrison's folate system response to

morr=readSBMLR(file.path(system.file(package="SBMLR"), "models/morrison.r"))
out1=simulate(morr,seq(-20,0,1))
morr$species$EMTX$ic=1
out2=simulate(morr,0:30)
outs=data.frame(rbind(out1,out2))
attach(outs)
par(mfrow=c(3,4))
plot(time,FH2b,type="l",xlab="Hours")
plot(time,FH2f,type="l",xlab="Hours")
plot(time,DHFRf,type="l",xlab="Hours")
plot(time,DHFRtot,type="l",xlab="Hours")
plot(time,CHOFH4,type="l",xlab="Hours")
plot(time,FH4,type="l",xlab="Hours")
plot(time,CH2FH4,type="l",xlab="Hours")
plot(time,CH3FH4,type="l",xlab="Hours")
plot(time,AICARsyn,type="l",xlab="Hours")
plot(time,MTR,type="l",xlab="Hours")
plot(time,TYMS,type="l",xlab="Hours")
#plot(time,EMTX,type="l",xlab="Hours")
plot(time,DHFReductase,type="l",xlab="Hours")
```

```
par(mfrow=c(1,1))
detach(outs)
morr$species$EMTX$ic=0
```

---

summary.SBML

*Get summary information from an SBML model*


---

### Description

This function extracts information from a model of class SBML and returns it as a list. The list includes species and reaction information tabularized as data frames.

### Usage

```
## S3 method for class 'SBML':
summary(object, ...)
```

### Arguments

`object` A model object of class SBML from which information is to be extracted.  
`...` For compatibility with `summary` of the base package.

### Details

no details

### Value

A list containing the following elements

<code>BC</code>	A logical vector indicating which species are not state variables, i.e. which species are boundary conditions or auxillary variables.
<code>y0</code>	The initial state (boundary conditions excluded!).
<code>nStates</code>	The length of the state vector, i.e. the number of system states.
<code>S0</code>	The full set of species initial values.
<code>nReactions</code>	The number of reactions.
<code>nSpecies</code>	The number of species, including states, boundary conditions and possibly auxillary variables such as the total concentration of dihydrofolate reductase in the morrison.r model.
<code>incid</code>	The incidence/stoichiometry matrix. This usually contains ones and minus ones corresponding to fluxes either synthesizing or degrading (respectively) a state variable chemical species. This matrix multiplied by the flux vector on its right yields the corresponding concentration state variable time derivatives.
<code>species</code>	Species information (i.e. names, ICs, BCs, and compartments) as a data frame.
<code>reactions</code>	Reaction information tabularized as a dataframe, including string laws and initial fluxes.

### Note

The list output can be attached to immediately define many model variables of interest.

**Author(s)**

Tom Radivoyevitch

**Examples**

```
library(SBMLR)
curto=readSBMLR(file.path(system.file(package="SBMLR"), "models/curto.r"))
summary(curto)
```

# Index

## \*Topic **arith**

- readSBML, 2
- readSBMLR, 3
- saveSBML, 4
- saveSBMLR, 5

## \*Topic **math**

- Ops.SBML, 1
- readSBML, 2
- readSBMLR, 3
- saveSBML, 4
- saveSBMLR, 5
- simulate, 6
- summary.SBML, 8

Ops.SBML, 1

readSBML, 2, 3  
readSBMLR, 2, 3

saveSBML, 4, 5  
saveSBMLR, 4, 5  
simulate, 6  
summary (*summary.SBML*), 8  
summary.SBML, 1, 8