

Package ‘babsim.hospital’

March 19, 2021

License GPL (>= 2)

Title Bartz & Bartz Simulation Hospital

Type Package

LazyLoad yes

LazyData true

Encoding UTF-8

Description Implements a discrete-event simulation model for a hospital resource planning problem. The project is motivated by the challenges faced by health care institutions in the current COVID-19 pandemic.

It can be used by health departments to forecast demand for intensive care beds, ventilators, and staff resources.

Our modelling approach is inspired by “A novel modelling technique to predict resource requirements in critical care - a case study” (Lawton and McCooe 2019) and combines two powerful technologies:

(i) discrete event simulation using the 'simmer' package and

(ii) model-based optimization using 'SPOT'.

Ucar I, Smeets B, Azcorra A (2019) <doi:10.18637/jss.v090.i02>.

Bartz-Beielstein T, Lasarczyk C W G, Preuss M (2005) <doi:10.1109/CEC.2005.1554761>.

Lawton T, McCooe M (2019) <doi:10.7861/futurehosp.6-1-17>.

Copyright file inst/COPYRIGHTS

URL <https://www.th-koeln.de/babsimhospital>

Version 11.7.14

Date 2021-03-18

Depends R (>= 4.0.0)

Imports SPOT, checkmate, curl, data.table, dplyr, ggplot2, golem, igraph, lubridate, markovchain, methods, padr, parallel, rvest, scales, simmer, slider, testthat, plyr, xml2

Suggests batchtools, knitr, rpart, rpart.plot, simmer.plot, stringr, tidyverse, usethis, vctrs

VignetteBuilder knitr

RoxygenNote 7.1.1

Config/testthat/parallel true

Config/testthat/edition 3

NeedsCompilation no

Author Thomas Bartz-Beielstein [aut, cre]

(<<https://orcid.org/0000-0002-5938-5158>>),

Frederik Rehbach [aut] (<<https://orcid.org/0000-0003-0922-8629>>),

Eva Bartz [aut],

Olaf Mersmann [aut] (<<https://orcid.org/0000-0002-7720-4939>>),

Martin Zaefferer [ctb] (<<https://orcid.org/0000-0003-2372-2092>>)

Maintainer Thomas Bartz-Beielstein <tbb@bartzundbartz.de>

Repository CRAN

Date/Publication 2021-03-18 23:40:02 UTC

R topics documented:

aggMax	4
aggregateSimulationReplications	5
autoplot.BabsimDailyCases	5
babsimHospital	6
babsimHospitalPara	7
babsimToolsConf	10
checkSimPara	12
dataCovidBeds20200624	12
dataICUBeds20200821	13
ensureRangeOpen	14
envToTibble	14
ex1InfectedDf	17
extendRki	18
extractSimulationResults	19
fetchRkiCases	19
fitExponential	20
funOptimizeSim	21
funWrapOptimizeSim	22
GABeds220200624	23
GermanCounties	23
GermanStates	24
getArrivalTimes	25
getBestParameter	25
getBounds	26
getConfFromData	27
getDailyMaxResults	28
getDecision	31
getDiviLinks	31
getError	32
getIcuBeds	33
getInfectedPerDay	33

getMatrixD	34
getMatrixP	34
getObkData	35
getParameterDataFrame	35
getParameterName	36
getParameterNameList	37
getParaSet	37
getPeakVec	38
getRealBeds	38
getRegionIcu	40
getRegionRki	40
getRkiData	41
getRkiRisk	41
getStartParameter	42
getSyntheticData	42
getTrainTestObjFun	44
ggVisualizeIcu	45
ggVisualizeRki	46
ggVisualizeRkiAge	47
ggVisualizeRkiEvents	48
ggVisualizeRkiExtended	48
ggVisualizeRkiExtendedDataCalculation	49
icudata	50
koelnarchive	51
mapAgeGroupToAge	51
mapAgeToAgeGroup	52
mapPToPara	52
mapXToPara	53
messageDateRange	54
messagef	54
modelResultHospital	55
nrwarchive	57
obkarchive	57
paras	58
plotDailyMaxResults	59
plotPostprocessedEnvs	60
postprocessEnvs	61
printConf	62
RiskScore	64
rkidata	65
rkiGeschlechtToSex	66
rkiToBabsimArrivals	66
rkiToBabsimData	67
rtgamma	68
runoptDirect	68
runOptLocal	70
skip_if_quicktest	71
smoothParameter	72

softmax	72
switchRkiRefdatumToMeldedatum	73
synthpara	73
updateIcudataFile	74
updateMatrixP	74
updateParaSet	75
updateRkidataFile	75
vaccineCounts	77
visualizeGraph	77
visualizeIcu	78
visualizeRki	79
visualizeRkiEvents	80
weighted_rmse	80

Index	82
--------------	-----------

aggMax	<i>Sane max for aggregation</i>
--------	---------------------------------

Description

Sane max for aggregation

Usage

```
aggMax(x)
```

Arguments

x	[vector(n)] numerical or character argument
---	--

Details

R's max returns '-Inf' for empty lists. This is undesirable for aggregation where we would rather have 'NA'.

Value

Maximum value in 'x' or 'NA' if 'x' is empty.

aggregateSimulationReplications
Summarize replications of a simulation

Description

Summarize replications of a simulation

Usage

```
aggregateSimulationReplications(results)
```

Arguments

results Raw simulation results. The result of [extractSimulationResults](#).

autoplot.BabsimDailyCases
Visualize new daily cases

Description

Visualize new daily cases

Usage

```
autoplot.BabsimDailyCases(  
  object,  
  xlab = "Date",  
  ylab = "Cases",  
  clab = "Age",  
  droplevels = TRUE,  
  drop.NA = TRUE,  
  ...  
)
```

Arguments

object	[BabsimCases] new daily cases.
xlab	[character(1)] a label for the x axis.
ylab	[character(1)] a label for the y axis.

clab	[character(1)] a label for the fill legend.
droplevels	[logical(1)] drop unused levels from ‘ageGroup’ and ‘sex’ before plotting.
drop.NA	[logical(1)] drop rows with missing or ‘NA’ values before plotting.
...	[] additional arguments. Ignored.

babsimHospital	<i>babsimHospital</i>
----------------	-----------------------

Description

Simulate resource allocation in hospitals.

Usage

```
babsimHospital(arrivalTimes = NULL, conf = list(), para = list(), ...)
```

Arguments

arrivalTimes	Arrival times as generated using <code>getArrivalTimes</code> .
conf	list with the following entries: seed seed. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log level (0 or 1). Default: 0 (no output). Values larger than 1 are mapped to 1.
para	List with parameter settings. Can be generated with babsimHospitalPara .
...	additional parameters passed to fun.

Value

This function returns an env list with:

xbest Parameters of the best found solution (matrix).

Examples

```

require("simmer")
require("dplyr")
# Generate simulation data based on number of infected persons per day:
x <- dataCovidBeds20200624
arrivalTimes <- getArrivalTimes(x$Infected)
conf <- babsimToolsConf()
y <- babsimHospital(
  arrivalTimes = arrivalTimes,
  conf = conf,
  para = babsimHospitalPara()
)
resources <- get_mon_resources(y)
# resources <- resources %>% filter(resource != "nurse")
mean(resources$server)

## 2nd example (shows details):
# Generate simulation data based on number of infected persons per day:
x <- dataCovidBeds20200624
arrivalTimes <- getArrivalTimes(x$Infected)
para <- babsimHospitalPara()
conf <- babsimToolsConf()
conf$logLevel <- 1
conf$simRepeats <- 1
para$GammaShapeParameter <- 0.8
y <- babsimHospital(
  arrivalTimes = arrivalTimes,
  conf = conf,
  para = para
)

```

babsimHospitalPara *babsimHospitalPara*

Description

Default Control list for `babsimHospital` This function returns the default controls for the functions `babsimHospital`. Control is a list of the following settings. Note: dependent parameters that are based on other parameters (e.g., probabilities that add to 1.0) are marked with an asterisk (*). Note: parameters that are currently not used, are marked with a double asterisk (**).

`logLevel` if larger than 10, shown detailed simmer output. simmer `log_level`, default is 0. 1

`FactorPatientsInfectedToHealthy*` Z1: Infected -> Healthy: percentage of patients that move from state infected to healthy, default is 0.831. Note: not used. Value is internally calculated as: `1 - FactorPatientsInfectedToHospital`

`AmntDaysInfectedToHealthy**` Z1: Infected -> Healthy: duration (in days) if patients move from state infected to healthy, default is 20.5. Note: not used, because not modeled. 2

FactorPatientsInfectedToHospital Z2: Infected -> Hospital: percentage of patients that move from state infected to hospital, default is 0.169.

AmntDaysInfectedToHospital Z2: Infected -> Hospital: duration (in days) if patients move from state infected to hospital, default is 8.4.

3

FactorPatientsHospitalToNormal* Z3: Hospital -> Normal: percentage of patients that move from state hospital to normal, default is 0. Note: not used. Value is internally calculated as: $1 - \text{FactorPatientsHospitalToIntensive} - \text{FactorPatientsHospitalToVentilation}$

AmntDaysHospitalToNormal* Z3: Hospital -> Normal: duration (in days) if patients move from state hospital to normal, default is 1e6. Note: not used. Patients move from hospital to normal immediately 4

FactorPatientsHospitalToIntensive Z4: Hospital -> Intensive: percentage of patients that move from state hospital to intensive, default is 0.012.

AmntDaysHospitalToIntensiv* Z4: Hospital -> Intensive: duration (in days) if patients move from state hospital to intensive, default is 1e6. Note: not used. Patients move from hospital to intensive immediately 5

FactorPatientsHospitalToVentilation Z5: Hospital -> Ventilation: percentage of patients that move from state hospital to ventilation, default is 0.036.

AmntDaysHospitalToVentilation* Z5: Hospital -> Ventilation: duration (in days) if patients move from state hospitel to ventilation, default is 1e6. Note: not used. Patients move from hospital to intensive immediately 6

FactorPatientsNormalToHealthy* Z6: Normal -> Healthy: percentage of patients that move from state normal to healthy, default is 0. Note: not used. Value is internally calculated as: $1 - \text{FactorPatientsNormalToIntensive} - \text{FactorPatientsNormalToVentilation} - \text{FactorPatientsNormalToDeath}$

AmntDaysNormalToHealthy Z6: Normal -> Healthy: duration (in days) if patients move from state normal to healthy, default is 11.6. 7

FactorPatientsNormalToIntensive Z7: Normal -> Intensive: percentage of patients that move from state normal to intensive, default is 0.0506.

AmntDaysNormalToIntensive Z7: Normal -> Intensive: duration (in days) if patients move from state normal to intensive, default is 1.25. 8

FactorPatientsNormalToVentilation Z8: Normal -> Ventilation: percentage of patients that move from state normal to ventilation, default is 0.1013.

AmntDaysNormalToVentilation Z8: Normal -> Ventilation: duration (in days) if patients move from state normal to ventilation, default is 3.63. 9

FactorPatientsNormalToDeath Z9: Normal -> Death: percentage of patients that move from state normal to death, default is 0.139.

AmntDaysNormalToDeath Z9: Normal -> Death: duration (in days) if patients move from state normal to death, default is 11.4. 10

FactorPatientsIntensiveToAftercare* Z10: Intensive -> Aftercare: percentage of patients that move from state intensive to aftercare, default is 0.25. Note: not used. Value is internally calculated as: $1 - \text{FactorPatientsIntensiveToVentilation} - \text{FactorPatientsIntensiveToDeath} - \text{FactorPatientsIntensiveToHealthy}$

AmntDaysIntensiveToAftercare Z10: Intensive -> Aftercare: duration (in days) if patients move from state intensive to aftercare, default is 7.0. 11

FactorPatientsIntensiveToVentilation Z11: Intensive > Ventilation: percentage of patients that move from state intensive to ventilation, default is 0.25.

AmntDaysIntensiveToVentilation Z11: Intensive > Ventilation: duration (in days) if patients move from state intensive to ventilation, default is 2.0. 12

FactorPatientsIntensiveToDeath Z12: Intensive -> Death: percentage of patients that move from state intensive to death, default is 0.25.

AmntDaysIntensiveToDeath Z12: Intensive -> Death: duration (in days) if patients move from state intensive to death, default is 2.0. 12a

Removed in v11: FactorPatientsIntensiveToHealthy Z12a: Intensive -> Healthy: percentage of patients that move from state intensive to healthy, default is 0.25.

AmntDaysIntensiveToHealthy Z12a: Intensive -> Healthy: duration (in days) if patients move from state intensive to healthy, default is 13.0. 13

Removed in v11: FactorPatientsVentilationToAftercare* Z13: Ventilation -> Aftercare: percentage of patients that move from state ventilation to aftercare, default is 0.08. Note: not used. Value is internally calculated as: $1 - \text{FactorPatientsVentilationToIntensiveAfter} - \text{FactorPatientsVentilationToDeath}$

Removed in v11: AmntDaysVentilationToAftercare Z13: Ventilation -> Aftercare: duration (in days) if patients move from state ventilation to aftercare, default is 9.0. 14

FactorPatientsVentilationToIntensiveAfter Z14: Ventilation -> IntensiveAfter: percentage of patients that move from state ventilation to intensiveAfter, default is 0.42.

AmntDaysVentilationToIntensiveAfter Z14: Ventilation -> IntensiveAfter: duration (in days) if patients move from state ventilation to intensiveAfter, default is 23.0. 15

Removed in v11: FactorPatientsVentilationToDeath Z15: Ventilation -> Death: percentage of patients that move from state ventilation to death, default is 0.5.

AmntDaysVentilationToDeath Z15: Ventilation -> Death: duration (in days) if patients move from state ventilation to death, default is 16.0. 16

FactorPatientsAftercareToHealthy* Z16: Aftercare -> Healthy: percentage of patients that move from state aftercare to healthy, default is 1.0. Note: not used. Value is 1. No branching required, because there is no alternative.

AmntDaysAftercareToHealthy Z16: Aftercare -> Healthy: duration (in days) if patients move from state aftercare to healthy, default is 21.0. 17 I

FactorPatientsIntensiveAfterToAftercare* Z17I: IntensiveAfter -> Aftercare: percentage of patients that move from state intensiveAfter to aftercare, default is 0.5. Note: not used. Value is internally calculated as: $1 - \text{FactorPatientsIntensiveAfterToHealthy} - \text{FactorPatientsIntensiveAfterToDeath}$

AmntDaysIntensiveAfterToAftercare Z17I: IntensiveAfter -> Aftercare: duration (in days) if patients move from state intensiveAfter to aftercare, default is 7.0. 17 II

Removed in v11: AmntDaysIntensiveAfterToHealthy Z17II: IntensiveAfter -> Healthy: duration (in days) if patients move from state intensiveAfter to healthy, default is 18.0. 18

FactorPatientsIntensiveAfterToDeath IntensiveAfter -> Death: percentage of patients that move from state intensiveAfter to death, default is 0.0.

AmntDaysIntensiveAfterToDeath IntensiveAfter -> Death: duration (in days) if patients move from state intensiveAfter to death, default is 1e-6.

GammaShapeParameter Gamma shape parameter, default is 1 (exponential distribution).

RiskFactorA Parameter a in the exponential function $r(x) = a \exp(b x)$ that models the risk r as a function of the age x, default is 0.02048948.

RiskFactorB Parameter b in the exponential function $r(x) = a \exp(b x)$ that models the risk r as a function of the age x, default is 0.07138200.

RiskMale Death risk of male patients compared to female , default is 2.

AmntDaysIntensiveAfterToHealthy IntensiveAfter -> Healthy: duration (in days) if patients move from state intensiveAfter to death, default is 3.

FactorPatientsIntensiveAfterToHealthy IntensiveAfter -> Healthy: percentage of patients that move from state intensiveAfter to healthy, default is 0.67.

Usage

```
babsimHospitalPara()
```

Value

a list

Examples

```
# change Gamma parameter
x <- babsimHospitalPara()
x$GammaShapeParameter <- 1.0
```

babsimToolsConf

babsimToolsConf

Description

Default configuration list for babsimTools. This function returns the default configuration settings of the babsimTools functions used to run `babsimHospital`. Configuration conf is a list of the following settings.

seed (int) Initial seed. Default: 123

simRepeats (int) Number of `simmer` simulation runs. Default: 1

parallel (logical) Use parallel simulations based on `mclapply`. Default: FALSE

perCores (num) Percentage of cores used, if parallel == TRUE. Default: 0.5

ICU (logical) Use ICU (RKI) data. Default: FALSE.

logLevel (int) 0 = no logging, >= 1 logging. Default: 0. If larger than 10, shown detailed simmer output.

maxCapacity (num) Maximum capacity used for `babsimHospital` resources. Default: 1e6.

dataset (chr) "GA" or "ICU". Default: "GA".

simulationDates List with the following entries:

StartDate (chr) Start date of the simulation data (infection data used to generate arrival times), first day. Default: "2020-03-03"

EndDate (chr) End date of the simulation data, last day. Default: "2020-06-24"

fieldDates List with the following entries:

StartDate (chr) Start date of the field (resources) data, first day. Default: "2020-03-03"

EndDate (chr) End date of the field data, last day. Default: "2020-06-24"

simulationData (data frame) Data used for the simulation. Default [dataCovidBeds20200624](#)

bed int 2 2 3 3 3 3 3 3 4 4 ...

intensiveBed int 0 0 0 0 0 0 0 0 0 0 ...

intensiveBedVentilation int 0 0 0 0 0 0 0 0 0 0 ...

Day Date, format: "2020-03-03" "2020-03-04" "2020-03-05" "2020-03-06" ...

Infected num 5 0 0 0 0 0 0 7 2 5 ...

Sick num 5 5 5 5 5 5 5 12 14 19 ...

fieldEvents (data frame) Data used for evaluation of the simulation. Default [GABeds220200624](#)

resource chr "bed" "bed" "bed" "bed" ...

time int 1 2 3 4 5 6 7 8 9 10 ...

med int 2 2 3 3 3 3 3 3 4 4 ...

source int 2 2 3 3 3 3 3 3 4 4 ...

date Date, format: "2020-03-03" "2020-03-04" "2020-03-05" "2020-03-06" ...

resource (vector) Resources used in the simulation. Default: c("bed", "intensiveBed", "intensiveBedVentilation").
For ICU data use: c("bed", "intensiveBedVentilation")

resource chr "bed" "bed" "bed" "bed" ...

time int 1 2 3 4 5 6 7 8 9 10 ...

med int 2 2 3 3 3 3 3 3 4 4 ...

source int 2 2 3 3 3 3 3 3 4 4 ...

date Date, format: "2020-03-03" "2020-03-04" "2020-03-05" "2020-03-06" ...

Usage

```
babsimToolsConf()
```

Value

a list

Examples

```
# turn on parallel simulation:
conf <- babsimToolsConf()
conf$parallel <- TRUE
conf$simulationDates$StartDate <- "2020-01-01"
```

checkSimPara	<i>checkSimPara</i>
--------------	---------------------

Description

check (and correct) parameter list

Usage

```
checkSimPara(para = babsimHospitalPara())
```

Arguments

para list: optimization parameters, e.g., generated via [babsimHospitalPara](#)

Value

corrected parameter list

Examples

```
x0 <- babsimHospitalPara()
x <- checkSimPara(x0)
```

dataCovidBeds20200624	<i>dataCovidBeds20200624</i>
-----------------------	------------------------------

Description

A data set of COVID-19 cases with 99 obs. of 11 variables.

Usage

```
dataCovidBeds20200624
```

Format

A data frame with 32239 rows and 6 columns:

bed int 2 2 3 3 3 3 3 3 4 4 ...

intensiveBed int 0 0 0 0 0 0 0 0 0 ...

intensiveBedVentilation int 0 0 0 0 0 0 0 0 0 ...

Day Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

Infected num 5 0 0 0 0 0 0 7 2 5 ...

Sick num 5 5 5 5 5 5 5 12 14 19 ...

The variable `obk$Sick` was generated from the `Infected` data as follows: `slide_dbl(obk$Infected, ~sum(.x), .before = (amntDaysSickness - 1))` `amntDaysSickness` was set to 20.

Examples

```
x <- dataCovidBeds20200624
# first look
str(x)

# plot
x$InfCum <- cumsum(x$Infected)
plot(x$Day, x$InfCum, type = "l", log = "y", ylim = c(1, 500))
lines(x$Day, x$Infected + 1e-6)
```

```
dataICUBeds20200821  dataICUBeds20200821
```

Description

A data set of COVID-19 ICU beds with 113 obs. of 3 variables.

Usage

```
dataICUBeds20200821
```

Format

A data frame with the following entries:

bed 640 597 538 553 591 573 593 527 529 508 ...

intensiveBedVentilation int 1549 1508 1441 1396 1346 1311 1230 1185 1121 1073 ...

Day Date, format: '2020-05-01' '2020-05-02' '2020-05-03' '2020-05-04' ...

The data frame was generated as follows: `icu <- icudata icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu)) icuCov$daten_stand <- as.Date(icuCov$daten_stand) icuCovBeatm <- as.data.frame(xtabs(~ daten_stand, icu)) icuCovBeatm$daten_stand <- as.Date(icuCovBeatm$daten_stand) dataICUBeds20200821 <- data.frame(bed=(icuCov$Freq - icuCovBeatm$Freq), intensiveBedVentilation=icuCovBeatm$Freq, Day = as.Date(icuCovBeatm$daten_stand))`

Examples

```
x <- dataICUBeds20200821
# first look
str(x)

# plot
x <- dataICUBeds20200821
plot(x$Day, x$bed, type = "o")
lines(x$Day, x$intensiveBedVentilation, type = "o", col = "red")
```

ensureRangeOpen	<i>ensureRangeOpen</i>
-----------------	------------------------

Description

Ensure that value belongs to the open interval]a,b[

Usage

```
ensureRangeOpen(x, a, b)
```

Arguments

x	value
a	lower limit
b	upper limit

Value

corrected value

Examples

```
# return 1:
ensureRangeOpen(x = 10, a = 0, b = 1)
# return 0:
ensureRangeOpen(x = 0, a = 0, b = 1)
# return 0.5:
ensureRangeOpen(x = 0.5, a = 0, b = 1)
```

envToTibble	<i>envToTibble</i>
-------------	--------------------

Description

Convert babsim simulation results to tibble data. Input: [simmer](#) simulation environment and field data formatted using [getRealBeds](#). The formatted field data has dim (nxm, 5), the output data has dimension (nxm, 15). The method [get_mon_resources](#) function is used to extract information from the babsim.hospital simulation. The function is used by [modelResultHospital](#) to prepare the calculation of the error.

Usage

```
envToTibble(envs, fieldEvents, conf, dontFilter = FALSE)
```

Arguments

envs	simmer simulation environment. Result from <code>babsim.hospital</code> simulation runs, e.g., output from <code>babsimHospital</code> .
fieldEvents	Real values. Output from <code>getRealBeds</code> , i.e., a (nxm, 5)-dim data.frame with the following variables: resource chr: 'bed' 'bed' 'bed' 'bed' ... time int: 1 2 3 4 5 6 7 8 9 10 ... med int: 2 2 3 3 3 3 3 3 4 4 ... source chr: 'GA' 'GA' 'GA' 'GA' ... date POSIXct, format: '2020-03-03 01:00:00' '2020-03-04 01:00:00' '2020-03-05 01:00:00' '2020-03-06 01:00:00' ...
conf	list with the following entries (generated with <code>babsimToolsConf</code>): seed seed. Change the seed value to get different output for the same input parameters. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log leveled (0 or 1). Default: 0 (no output) maxCapacity max capacity of resources. Default: 1e6 dataset char name of the data set. Default: 'GA' simulationDates list with StartDate and EndDate. Period that is used for the simulation (babsim, simmer). Default: <code>list(StartDate = '2020-03-03', EndDate = '2020-06-24')</code> fieldDates list with StartDate and EndDate. Period when real data is available (resource usage). Default: <code>list(StartDate = '2020-03-03', EndDate = '2020-06-24')</code> simulationData data frame. Data used for the simulation. Default: <code>dataCovidBeds20200624</code> fieldEvents data frame. Data used for the evaluation (error). Default: <code>GABeds220200624</code> resource vector with resource names. Default: <code>c('bed', 'intensiveBed', 'intensiveBedVentilation')</code>
dontFilter	do not filter by date (used in subsimulations)

Details

`get_mon_resources` returns state changes in resources:

- 'resource': resources name
- 'time': time instant of the event that triggered the state change
- 'server': server count
- 'queue': queue count
- 'capacity': capacity

- `'queue_size'`: queue size
- `'system'`: system count (server + queue). If no queues are used, system values equal server values.
- `'system_limit'`: system limit (capacity + queue_size)

Value

This function returns an env data frame (tibble [nxm, 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

resource (**chr**) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...

time (**num**) time step: 3 10 12 13 14 15 15 15 15 16 ...

server (**int**) server: 1 2 3 2 3 4 3 4 5 6 ...

limit (**num**) limit: Inf Inf Inf Inf Inf ...

replication (**int**) replication: 1 1 1 1 1 1 1 1 1 1 ...

upper (**int**) upper: 1 2 3 2 3 5 5 5 5 7 ...

lower (**int**) lower: 1 2 3 2 3 3 3 3 3 5 ...

med (**num**) med: 1 2 3 2 3 4 4 4 4 6 ...

date (**POSIXct**) time, format: yyyy-mm-dd hh:mm:ss

rwdate (**POSIXct**) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...

source (**chr**) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

See Also

[modelResultHospital](#)

Examples

```
data <- getSyntheticData()
para <- babsimHospitalPara()
conf <- babsimToolsConf()
conf <- getConfFromData(
  conf = conf,
  simData = data$simData,
  fieldData = data$fieldData
)
arrivalTimes <- getArrivalTimes(data$simData$Infected)
fieldEvents <- getRealBeds(
  data = data$fieldData,
  resource = c("bed", "intensiveBed", "intensiveBedVentilation")
)
envs <- babsimHospital(arrivalTimes = arrivalTimes, conf = conf, para = para)
res <- envToTibble(envs = envs, conf = conf, fieldEvents = fieldEvents)
```

ex1InfectedDf	<i>ex1InfectedDf</i>
---------------	----------------------

Description

Data used in example 1 and for testing A synthetic data set of COVID-19 cases with 99 obs. of 8 variables:

Usage

```
ex1InfectedDf
```

Format

A data frame with 99 obs. of 8 variables:

index int 1 2 3 4 5 6 7 8 9 10 ...

Day Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

Infected num 6 0 1 3 3 1 5 1 6 104 ...

Sick num 6 6 7 10 13 14 19 20 26 130 ...

InfectedCum num 6 6 7 10 13 14 19 20 26 130 ...

normalStation num 1 1 1 1 2 2 3 3 4 18 ...

intensive num 0 0 0 0 0 0 0 0 1 ...

ventilation num 0 0 0 0 0 0 0 0 2 ...

Examples

```
x <- ex1InfectedDf
# first look
str(x)

# plot
x$InfCum <- cumsum(x$Infected)
plot(x$Day, x$InfCum, type = "l", log = "y", ylim = c(1, 500))
lines(x$Day, x$Infected + 1e-6)
```

 extendRki

extendRki Erweiterung der RKI Daten

Description

Combine existing data with synthetic data

Usage

```
extendRki(
  data = getRkiData(babsim.hospital::rkidata),
  EndDate = max(data$Day) + 28,
  R0 = c(1, 1),
  tau = 5
)
```

Arguments

data	rki data, e.g., <code>getRkiData(babsim.hospital::rkidata)</code>
EndDate	Ende (Tag), e.g., <code>'2020-05-04'</code>
R0	Basisreproduktionszahl. Constant, if a scalar value is given. If a vector of two values are given, they will be interpreted as a start and an end value, respectively. <code>c(1,2)</code> defines an increasing R0 value from 1 to 2. Default: 1, i.e., constant 1. Note: This is NOT exactly the same R0 value presented by the Robert-Koch Institute, please refer to https://en.wikipedia.org/wiki/Basic_reproduction_number for our implementation.
tau	Ansteckungszeitraum in Tagen

See Also

[getRkiData](#)

Examples

```
# take 10,000 data points only:
data <- getRkiData(babsim.hospital::rkidata[1:10000, ])
# check whether enough data are provided
if (dim(data)[1] > 1e6){
  n <- as.integer(max(data$Day) - min(data$Day))
  StartDay <- min(data$Day) + round(n * 0.995)
  data <- data[which(data$Day >= StartDay), ]
  EndDate <- max(data$Day) + 2
  dataExt <- extendRki(
    data = data,
    EndDate = EndDate,
    R0 = c(0.1, 0.2)
  )
}
```

 extractSimulationResults

Extract and summarize BaBSim results

Description

Given a list of result environments returned by `babsimHospital`, extract and summarize resource usage for each day. If the resource usage fluctuates during the day, the maximum usage is reported.

Usage

```
extractSimulationResults(envs, conf)
```

Arguments

<code>envs</code>	List of simulation environments
<code>conf</code>	Configuration for simulation runs in <code>envs</code>

Value

A data table with columns

`date` Day on which the resource is required.

`resource` The resource required.

`replication` From which replication the result stems.

`count` Number of units of resource in use.

 fetchRkiCases

Download German daily case summary from RKI

Description

Download German daily case summary from RKI

Usage

```
fetchRkiCases(dir = tempdir(), force = FALSE)
```

Arguments

<code>dir</code>	[character(1)] directory where downloaded data is stored. Defaults to <code>tempdir()</code> .
<code>force</code>	[bool(1)] if true, force download of data even if a previous download is found in <code>dir</code> .

Value

An object of class 'BabsimCasesRki'.

See Also

- [GermanStates](#) for a list of German states and their state id.
- [GermanCounties](#) for a list of German counties and their county id.

fitExponential *Fit a exponential function to data.*

Description

Fit the model $y = a e^{bx}$ to the provided data.

Usage

```
fitExponential(x, y, a0 = 0, b0 = 0)
```

Arguments

x	x data
y	y data
a0	start value (default: 1)
b0	start value (default: 1)

Value

Named vector of coefficients ('a', 'b')

Examples

```
age <- c(2, 10, 25, 47, 70, 90)
risk <- c(0.01, 0.07, 0.15, 0.65, 3, 12.64)
plot(age, risk)
ab <- fitExponential(x = age, y = risk, a0 = 1, b0 = 0)
y <- ab[1] * exp(ab[2] * age)
lines(age, y)
```

funOptimizeSim	<i>funOptimizeSim</i>
----------------	-----------------------

Description

Interface function to evaluate one parameter configuration from [babsimHospitalPara](#)

Usage

```
funOptimizeSim(x, conf, data, ...)
```

Arguments

x	num: real values. Will be interpreted as parameter values for babsimHospital . Names of these parameters can be obtained via getParameterName .
conf	list with the following entries: seed seed. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log leved (0 or 1). Default: 0 (no output)
data	list with simData and fieldData
...	additional variables

Value

This function returns a real value, that represents the combined rmse from the three beds types.

Examples

```
x <- getStartParameter()
conf <- babsimToolsConf()
data <- getObkData()
err <- funOptimizeSim(x = x, conf = conf, data = data)
```

funWrapOptimizeSim *funWrapOptimizeSim*

Description

Wrapper function for funOptimizeSim

Usage

```
funWrapOptimizeSim(x, conf, data)
```

Arguments

x	num: real values. Will be interpreted as parameter values from babsimHospital. Names of these parameters can be obtained via getParameterName .
conf	list with the following entries: seed seed. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log leved (0 or 1). Default: 0 (no output) For example: conf <-babsimToolsConf()
data	list with simData and fieldData, e.g. data <-getObkData().

Value

This function returns a num [1, 1] matrix, that represents the combined rmse from the three beds.

Examples

```
para <- getStartParameter()
conf <- babsimToolsConf()
data <- getObkData()
funWrapOptimizeSim(x = para, conf = conf, data = data)
```

`GABeds220200624`*GABeds220200624 Intensivbetten: Daten vom 24.6.2020*

Description

Betten Daten aggregiert
Ein data.frame mit 5 Variablen

Usage`GABeds220200624`**Format**

data.frame with 342 obs. of 5 variables:

ressource chr 'bed' 'bed' 'bed' 'bed' ...

time int 1 2 3 4 5 6 7 8 9 10 ...

med int 2 2 3 3 3 3 3 3 4 4 ...

source int 2 2 3 3 3 3 3 3 4 4 ...

date Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

`GermanCounties`*List of German administrative counties*

Description

A dataset containing the names and AGS of all 401 German counties.

Usage`GermanCounties`**Format**

A [data.table](#) with 401 rows and 3 variables:

stateId id of state containing this county.

countyId unique id, first five digits of the German AGS.

county name of county in German.

Source

Gemeindeverzeichnis-Informationssystem (GV-ISys) of the German Federal Statistics Office

See Also

[GermanStates](#) for German states.

Examples

```
x <- merge(GermanCounties, GermanStates, by="stateId")
subset(x, countyId == "05135") # Cologne, North Rhine-Westphalia
```

GermanStates	<i>List of German states</i>
--------------	------------------------------

Description

A dataset containing all 16 German states.

Usage

```
GermanStates
```

Format

A [data.table](#) with 16 rows and 2 variables:

stateId unique id, first two digits of the German AGS.

state name of state in German.

Source

Gemeindeverzeichnis-Informationssystem (GV-ISys) of the German Federal Statistics Office

See Also

[GermanCounties](#) for German counties.

Examples

```
head(GermanStates)
```

getArrivalTimes	<i>getArrivalTimes</i>
-----------------	------------------------

Description

Generate arrival times.

Usage

```
getArrivalTimes(xDaily)
```

Arguments

xDaily Vector that contains the number of arrivals for each day.

Value

This function returns a data frame of arrival times with the following entries:

time (**num**) name of the seized resource

@seealso [rkiToBabsimArrivals](#)

Examples

```
x <- dataCovidBeds20200624
arrivalTimes <- getArrivalTimes(xDaily = x$Infected)
# For RKI data, use rkiToBabsimArrivals as follows:
arrivalTimes <- rkiToBabsimArrivals(rki = babsim.hospital::rkidata)
```

getBestParameter	<i>Return the best parameter set found</i>
------------------	--

Description

Extract the best result from a data frame of optimization runs and return it as a valid Para list.

Usage

```
getBestParameter(para)
```

Arguments

para A data frame with columns 'y' and 'x.1' to 'x.27'.

Value

This function returns an env data frame (tibble [560 × 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

resource (**chr**) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...
 time (**num**) time step: 3 10 12 13 14 15 15 15 15 16 ...
 server (**int**) server: 1 2 3 2 3 4 3 4 5 6 ...
 limit (**num**) limit: Inf Inf Inf Inf Inf ...
 replication (**int**) replication: 1 1 1 1 1 1 1 1 1 1 ...
 upper (**int**) upper: 1 2 3 2 3 5 5 5 5 7 ...
 lower (**int**) lower: 1 2 3 2 3 3 3 3 3 5 ...
 med (**num**) med: 1 2 3 2 3 4 4 4 4 6 ...
 date (**POSIXct**) time, format: yyyy-mm-dd hh:mm:ss
 rwdate (**POSIXct**) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...
 source (**chr**) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

See Also

[mapXToPara](#)

Examples

```
getBestParameter(getParaSet(5374))
```

getBounds

getBounds

Description

Returns parameter bounds for babsim runs (settings version > v10.4.8)

Usage

```
getBounds()
```

Value

This function returns a list of two vectors

Examples

```
bounds <- getBounds()
lower <- bounds$lower
upper <- bounds$upper
```

```
getConfFromData      getConfFromData
```

Description

Generate a configuration list for `babsimTools` based on the data file and the vector of resources. This function returns the default configuration settings of the `babsimTools` functions used to run `babsimHospital`.

Usage

```
getConfFromData(conf, simData, fieldData)
```

Arguments

<code>conf</code>	Configuration. Default: <code>babsimToolsConf</code>
<code>simData</code>	Simulation data. Default: <code>dataCovidBeds20200624</code>
<code>fieldData</code>	Field (real) data. Default: <code>dataCovidBeds20200624</code>

Details

Configuration `conf` is a list of the following settings.

`seed` (int) Initial seed. Default: 123

`simRepeats` (int) Number of `simmer` simulation runs. Default: 1

`parallel` (logical) Use parallel simulations based on `mclapply`. Default: FALSE

`perCores` (num) Percentage of cores used, if `parallel == TRUE`. Default: 0.5

`ICU` (logical) Use ICU (RKI) data. Default: FALSE.

`logLevel` (int) 0 = no logging, >= 1 logging. Default: 0. If larger than 10, shown detailed `simmer` output.

`maxCapacity` (num) Maximum capacity used for `babsimHospital` resources. Default: 1e6.

`dataset` (chr) 'GA' or 'ICU'. Default: 'GA'.

`simulationDates` List with the following entries:

`StartDate` (chr) Start date of the simulation data (infection data used to generate arrival times), first day. Default: '2020-03-03'

`EndDate` (chr) End date of the simulation data, last day. Default: '2020-06-24'

`fieldDates` List with the following entries:

`StartDate` (chr) Start date of the field (resources) data, first day. Default: '2020-03-03'

`EndDate` (chr) End date of the field data, last day. Default: '2020-06-24'

`simulationData` (data frame) Data used for the simulation. Default `dataCovidBeds20200624`

bed int 2 2 3 3 3 3 3 4 4 ...

intensiveBed int 0 0 0 0 0 0 0 0 0 ...

```

intensiveBedVentilation int 0 0 0 0 0 0 0 0 0 0 ...
Day Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...
Infected num 5 0 0 0 0 0 0 7 2 5 ...
Sick num 5 5 5 5 5 5 5 12 14 19 ...
fieldEvents (data frame) Data used for evaluation of the simulation. Default GABeds220200624
  resource chr 'bed' 'bed' 'bed' 'bed' ...
  time int 1 2 3 4 5 6 7 8 9 10 ...
  med int 2 2 3 3 3 3 3 3 4 4 ...
  source int 2 2 3 3 3 3 3 3 4 4 ...
  date Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...
resource (vector) Resources used in the simulation. Default: c('bed', 'intensiveBed', 'intensiveBedVentilation').
For ICU data use: c('bed', 'intensiveBedVentilation')
  resource chr 'bed' 'bed' 'bed' 'bed' ...
  time int 1 2 3 4 5 6 7 8 9 10 ...
  med int 2 2 3 3 3 3 3 3 4 4 ...
  source int 2 2 3 3 3 3 3 3 4 4 ...
  date Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

```

Value

a list

Examples

```

conf <- babsimToolsConf()
simData <- babsim.hospital::dataCovidBeds20200624
fieldData <- babsim.hospital::dataCovidBeds20200624
conf <- getConfFromData(
  conf = conf,
  simData = simData,
  fieldData = fieldData
)
# turn on parallel simulation:
conf$parallel <- TRUE
# Change the start date of the simulations
conf$simulationDates$StartDate <- "2020-01-01"

```

getDailyMaxResults *getDailyMaxResults*

Description

Combine babsim simulation results with real (field) data. Use daily max resources. Input: [simmer](#) simulation environment and field data formatted using [getRealBeds](#). The formatted field data has dim (nxm, 5), the output data has dimension (nxm, 15). The method [get_mon_resources](#) function is used to extract information from the babsim.hospital simulation. The function is used by [modelResultHospital](#) to prepare the calculation of the error.

Usage

```
getDailyMaxResults(envs, fieldEvents, conf)
```

Arguments

envs	simmer simulation environment. Result from <code>babsim.hospital</code> simulation runs, e.g., output from <code>babsimHospital</code> .
fieldEvents	Real values. Output from <code>getRealBeds</code> , i.e., a (nxm, 5)-dim data.frame with the following variables: resource chr: 'bed' 'bed' 'bed' 'bed' ... time int: 1 2 3 4 5 6 7 8 9 10 ... med int: 2 2 3 3 3 3 3 3 4 4 ... source chr: 'GA' 'GA' 'GA' 'GA' ... date POSIXct, format: '2020-03-03 01:00:00' '2020-03-04 01:00:00' '2020-03-05 01:00:00' '2020-03-06 01:00:00' ...
conf	list with the following entries (generated with <code>babsimToolsConf</code>): seed seed. Change the seed value to get different output for the same input parameters. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log leveled (0 or 1). Default: 0 (no output) maxCapacity max capacity of resources. Default: 1e6 dataset char name of the data set. Default: 'GA' simulationDates list with StartDate and EndDate. Period that is used for the simulation (babsim, simmer). Default: list(StartDate = '2020-03-03', EndDate = '2020-06-24') fieldDates list with StartDate and EndDate. Period when real data is available (resource usage). Default: list(StartDate = '2020-03-03', EndDate = '2020-06-24') simulationData data frame. Data used for the simulation. Default: dataCovidBeds20200624 fieldEvents data frame. Data used for the evaluation (error). Default: GABeds220200624 resource vector with resource names. Default: c('bed', 'intensiveBed', 'intensiveBedVentilation')

Details

`get_mon_resources` returns state changes in resources:

- 'resource': resources name
- 'time': time instant of the event that triggered the state change
- 'server': server count

- 'queue': queue count
- 'capacity': capacity
- 'queue_size': queue size
- 'system': system count (server + queue). If no queues are used, system values equal server values.
- 'system_limit': system limit (capacity + queue_size)

Value

This function returns an env data frame (tibble [nxm, 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

resource (**chr**) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...
 time (**num**) time step: 3 10 12 13 14 15 15 15 15 16 ...
 server (**int**) server: 1 2 3 2 3 4 3 4 5 6 ...
 limit (**num**) limit: Inf Inf Inf Inf Inf ...
 replication (**int**) replication: 1 1 1 1 1 1 1 1 1 ...
 upper (**int**) upper: 1 2 3 2 3 5 5 5 5 7 ...
 lower (**int**) lower: 1 2 3 2 3 3 3 3 5 ...
 med (**num**) med: 1 2 3 2 3 4 4 4 4 6 ...
 date (**POSIXct**) time, format: yyyy-mm-dd hh:mm:ss
 rwdate (**POSIXct**) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...
 source (**chr**) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

See Also

[modelResultHospital](#)

Examples

```
data <- getSyntheticData()
para <- babsimHospitalPara()
conf <- babsimToolsConf()
conf <- getConfFromData(
  conf = conf,
  simData = data$simData,
  fieldData = data$fieldData
)
arrivalTimes <- getArrivalTimes(data$simData$Infected)
envs <- babsimHospital(arrivalTimes = arrivalTimes, conf = conf, para = para)
fieldEvents <- getRealBeds(
  data = data$fieldData,
  resource = c("bed", "intensiveBed", "intensiveBedVentilation")
)
res <- getDailyMaxResults(envs = envs, fieldEvents = fieldEvents, conf = conf)
```

getDecision	<i>getDecision</i>
-------------	--------------------

Description

For given n probabilities $0 \leq p_i \leq 1$ with $\sum(p_i)=1$, return 0,1,2,3,...,n with probability $p_0, p_1, p_2, \dots, p_n$.

Usage

```
getDecision(p)
```

Arguments

p vector of probabilities

Value

int decision in the range from 0 to n

Examples

```
p <- c(0.6, 0.3, 0.1)
getDecision(p)
```

getDiviLinks	<i>Retrieve links to all DIVI day reports.</i>
--------------	--

Description

Retrieve links to all DIVI day reports.

Usage

```
getDiviLinks()
```

Value

List of download URLs to the daily reports.

<code>getError</code>	<i>getError</i>
-----------------------	-----------------

Description

Determine error from babsim runs. This error is the sum of the RMSE values for bed, intensiveBed, and intensiveBedVentilation.

Usage

```
getError(res, conf)
```

Arguments

<code>res</code>	Results from <code>getDailyMaxResults</code> .
<code>conf</code>	configuration

Value

This function returns a num value, that represents the combined rmse from the beds.

Examples

```
para <- babsimHospitalPara()
conf <- babsimToolsConf()
data <- getObkData()
set.seed(conf$seed)
para <- checkSimPara(para)
arrivalTimes <- getArrivalTimes(data$simData$Infected)
envs <- babsimHospital(
  arrivalTimes = arrivalTimes,
  conf = conf,
  para = para
)
fieldEvents <- getRealBeds(
  data = data$fieldData,
  resource = conf$ResourceNames
)
res <- getDailyMaxResults(
  envs = envs,
  fieldEvents = fieldEvents,
  conf = conf
)
err <- getError(res, conf = conf)
```

```
getIcuBeds          getIcuBeds
```

Description

Convert the 9 dim DIVI ICU data (bundesland,gemeindeschluessel,..., daten_stand) into a data.frame with bed, intensiveBedVentilation, and Day

Usage

```
getIcuBeds(data = babsim.hospital::icudata)
```

Arguments

data data.frame with obs. of 9 variables

Value

data frame with observations of 3 variables:

intensiveBed int COVID-19 ICU beds without ventilation

intensiveBedVentilation int COVID-19 ICU beds with ventilation

Day Date, format: '2020-05-01' '2020-05-02' '2020-05-03' '2020-05-04' ...

Examples

```
IcuBeds <- getIcuBeds(data = icudata)
```

```
getInfectedPerDay  getInfectedPerDay
```

Description

Generate Poisson distributed infections. This function calculates n, the number of days between StartDate and EndDate, and returns a vector with n realizations of a Poisson(lambda) distributed random variable.

Usage

```
getInfectedPerDay(lambda = 4, StartDate = "2020-03-03", EndDate = "2020-06-24")
```

Arguments

lambda Expected number of infections/day.
 StartDate Day, simulation starts
 EndDate Day, simulation ends

Value

This function returns a vector that lists the number of infections.

Examples

```
StartDate <- "2020-03-03"
EndDate <- "2020-06-24"
getInfectedPerDay(lambda = 4, StartDate = StartDate, EndDate = EndDate)
```

`getMatrixD`

getMatrixD

Description

Builds the duration matrix

Usage

```
getMatrixD(para = babsimHospitalPara())
```

Arguments

`para` parameter vector, e.g., generated via [babsimHospitalPara](#). Default: [babsimHospitalPara](#)

Value

a matrix with transition durations

Examples

```
getMatrixD()
```

`getMatrixP`

getMatrixP

Description

Builds the probability matrix

Usage

```
getMatrixP(para = babsimHospitalPara())
```

Arguments

`para` parameter vector, e.g., generated via [babsimHospitalPara](#). Default: [babsimHospitalPara](#)

Value

a matrix with transition probabilities

Examples

```
getMatrixP()
```

`getObkData` *getObkData*

Description

Generate simData and fieldData from OBK data

Usage

```
getObkData(data = babsim.hospital::dataCovidBeds20200624)
```

Arguments

data OBK data. Default: [dataCovidBeds20200624](#)

Value

list with simData and fieldData

Examples

```
data <- getObkData()
```

`getParameterDataFrame` *getParameterDataFrame*

Description

Get parameters (probabilities and durations) of the babsim.hospital simulator

Usage

```
getParameterDataFrame(  
  paraList = list(obk = getParaSet(5374), koeln = getParaSet(5315), nrw =  
  getParaSet(5))  
)
```

Arguments

`paraList` list of parameter values. Each list element has the form `obk=getParaSet(5374)`.

Value

data.frame with parameters in each column.

Examples

```
df <- getParameterDataFrame()
```

<code>getParameterName</code>	<i>getParameterName</i>
-------------------------------	-------------------------

Description

Returns the name (chr) of the babsim x parameter vector.

Usage

```
getParameterName(n)
```

Arguments

`n` int: position

Value

This function returns a character value, which represents the name of the n-th x variable.

Examples

```
getParameterName(16)
```

`getParameterNameList` *getParameterNameList*

Description

Returns the names (chr) of the babsim x parameter vector.

Usage

```
getParameterNameList(x)
```

Arguments

x vector of int: positions

Value

This function returns a vector. Its elements represent the names of the n-th x variables.

Examples

```
getParameterNameList(c(16, 18))
```

`getParaSet` *getParaSet*

Description

Load a specific parameter set by region.

Usage

```
getParaSet(region)
```

Arguments

region integer, the region id

Value

data.frame parameters of that specific region

Examples

```
getParaSet(5315)
```

getPeakVec	<i>getPeakVec</i>
------------	-------------------

Description

Generate peak values.

Usage

```
getPeakVec(
  peakData = c(10, 100),
  startDate = "2020-03-03",
  endDate = "2020-06-24"
)
```

Arguments

peakData	Vector of time steps and peak heights.
startDate	Day, simulation starts.
endDate	Day, simulation ends.

Value

This function returns a vector of peaks data.

Examples

```
getPeakVec()
```

getRealBeds	<i>getRealBeds</i>
-------------	--------------------

Description

Convert daily data, e.g., a data.frame with the columns `bed`, `intensiveBedVentilation`, `Day` into event data, e.g., a data.frame with the following columns `resource`, `time`, `med`, `source`, `date`.

Usage

```
getRealBeds(data, resource)
```

Arguments

data	(n, m) data frame with daily bed data, e.g., from icudata
resource	vector of resource names, e.g., 'bed'. Default: <code>resource=c('bed', 'intensiveBedVentilation')</code> . For GA data use: <code>resource=c('bed', 'intensiveBed', 'intensiveBedVentilation')</code>

Details

Prepares data for combination with output (env) from [simmer](#). Extracts and formats real data from the real data sets, e.g., `icudata`. The resulting data frame can be combined with the output from the simulation run. Can be used to add the true data (ground truth) to the simulated data.

Value

This function returns a (n x m, 5) data frame with:

resource (**chr**) name of the seized resource

time (**int**) time step, starts with 1

med (**int**) amount of the seized resource

source (**chr**) name of the simulation that was used

date (**Date**) time, format: yyyy-mm-dd

See Also

[getIcuBeds](#).

Examples

```
# First example shows how to process the GA data
GABeds <- getRealBeds(
  data = babsim.hospital::dataCovidBeds20200624,
  resource = c("bed", "intensiveBed", "intensiveBedVentilation")
)

# Second example shows how to process the DIVI ICU data.
icu <- babsim.hospital::icudata
icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu))
icuCov$daten_stand <- as.Date(icuCov$daten_stand)
icuCovBeatm <- as.data.frame(xtabs(faelle_covid_aktuell_beatmet ~ daten_stand, icu))
icuCovBeatm$daten_stand <- as.Date(icuCovBeatm$daten_stand)
Day <- as.Date(icuCovBeatm$daten_stand)
dataICUBeds20200821 <- data.frame(
  bed = (icuCov$Freq - icuCovBeatm$Freq),
  intensiveBedVentilation = icuCovBeatm$Freq,
  Day = as.Date(icuCovBeatm$daten_stand)
)
ICUBeds <- getRealBeds(
  data = dataICUBeds20200821,
  resource = c("bed", "intensiveBedVentilation")
)
```

getRegionIcu *getRegionIcu Auswahl der ICU Daten fuer eine Region*

Description

Auswahl anhand der Bundeslaender, Landkreis IDs

Usage

```
getRegionIcu(data = babsim.hospital::icudata, region = 5315)
```

Arguments

data	Daten, z.B. icudata
region	Id der Region, 0 Deutschland

Examples

```
data <- getRegionIcu(babsim.hospital::icudata, 5315)
```

getRegionRki *getRegionRki Auswahl der RKI Daten fuer eine Region*

Description

Auswahl anhand der Bundeslaender, Landkreis IDs

Usage

```
getRegionRki(data = babsim.hospital::rkidata, region)
```

Arguments

data	Daten, z.B. rkidata
region	Id der Region, 0 Deutschland

Examples

```
data <- getRegionRki(  
  data = babsim.hospital::rkidata[1:1000, ],  
  region = 0  
)
```

getRkiData	<i>getRkiData</i>
------------	-------------------

Description

Transforms the freshly downloaded rki data into the babsim data frame. Also imputes missing dates as zero frequency in the data.

Usage

```
getRkiData(rki)
```

Arguments

rki data.frame of downloaded rki data before preprocessing

Value

a data.frame of aggregated data

Day Date, format: '2020-01-01' '2020-01-02' '2020-01-03' '2020-01-04' ...
... ..

Examples

```
data <- getRkiData(rkidata[1:100, ])
```

getRkiRisk	<i>getRkiRisk</i>
------------	-------------------

Description

Calculate risk for RKI data

Usage

```
getRkiRisk(rki, para)
```

Arguments

rki data.frame of downloaded rki data
para parameter

Value

a data.frame

Examples

```
rki <- getRkiData(rkidata[1:10, ])
para <- babsimHospitalPara()
# get risk for the first 10 entries:
rkiWithRisk <- getRkiRisk(rki = rki, para)
```

getStartParameter *getStartParameter*

Description

Returns parameter for babsim runs

Usage

```
getStartParameter(para = babsimHospitalPara(), region = -1)
```

Arguments

para	parameter vector, e.g., generated via babsimHospitalPara . Default: babsimHospitalPara
region	(int) use region specific start parameter, e.g., 5374 for OBK. If region is negative (default), a generic start parameter is chosen. The selection is based on the obkpara, koelnpara, and nrwpara parameter values, which are the best known values found so far.

Value

This function returns a (1,n) dim matrix

Examples

```
para <- getStartParameter(region = 5374)
```

getSyntheticData *getSyntheticData*

Description

Generate synthetic data

Usage

```
getSyntheticData(
  StartDate = "2020-09-01",
  EndDate = "2020-11-30",
  lambda = 4,
  peakData = c(21, 50, 28, 40, 42, 50),
  amntDaysSickness = 20,
  hospitalizationRates = list(rBed = 0.1347, rIntensiveBed = 0.004,
    rIntensiveBedVentilation = 0.0171)
)
```

Arguments

StartDate	Start date. Default: '2020-09-01'
EndDate	Start date. Default: '2020-11-30'
lambda	Average number of daily infections. Default: 4
peakData	Vector to define peak events. Odd entries represent days, even entries the number of infections. Default: c(21, 50, 28, 40, 42, 50), i.e., after 21 days 50 additional infections, after 28 days 40 additional infections, and after 42 days 50 additional infections to the base infection rate per day, which is defined by lambda.
amntDaysSickness	Length (in days) of the interval that is used to determine the number of infected individuals that have to go to the hospital. Based on this interval, the number of sick individuals is determined. The number of sick individuals is multiplied by the hospitalization rate to determine the number of bed. Default: 20
hospitalizationRates	list of hospitalization rates, i.e., percentage of sick individuals that need a bed, or an intensiveBed, or an intensiveBedVentilation. Default: list(rBed = 0.1347, rIntensiveBed = 0.004, rIntensiveBedVentilation = 0.0171).

Value

data frame with the following entries: bed=bed, intensiveBed = intensiveBed, intensiveBedVentilation = intensiveBedVentilation, Day = Day, Infected=Infected, Sick = Sick)

bed int: COVID-19 beds

intensiveBed int: COVID-19 ICU beds

intensiveBedVentilation int COVID-19 ICU beds with ventilation

Day Date, format: '2020-05-01' '2020-05-02' '2020-05-03' '2020-05-04' ...

Infected int: number of infected individuals (daily)

Sick int: number of sick individuals (daily)

Examples

```
dataSynth <- getSyntheticData()
```

```
getTrainTestObjFun    getTrainTestObjFun
```

Description

Generate objective functions (one for train, optionally one for test)

Usage

```
getTrainTestObjFun(
  rkiwerte = babsim.hospital::rkidata,
  icuwerte = babsim.hospital::icudata,
  region = 5374,
  TrainSimStartDate = Sys.Date() - 12 * 7,
  TrainFieldStartDate = Sys.Date() - 8 * 7,
  TestSimStartDate = Sys.Date() - 8 * 7,
  TestFieldStartDate = Sys.Date() - 4 * 7,
  verbosity = 0,
  parallel = FALSE,
  percCores = NULL,
  icu = TRUE,
  icuWeights = c(1, 1),
  resourceNames = c("intensiveBed", "intensiveBedVentilation"),
  resourceEval = c("intensiveBed", "intensiveBedVentilation"),
  tryOnTestSet = TRUE
)
```

Arguments

rkiwerte	RKI Daten
icuwerte	ICU Daten
region	Landkreis Id, e.g., 5374 fuer OBK, 5315 fuer Koeln, 0 fuer Deutschland, oder Bundesland ID, e.g., 5 fuer NRW.
TrainSimStartDate	Start (Tag), e.g., '2020-05-01'
TrainFieldStartDate	Start (Tag), e.g., '2020-06-01'
TestSimStartDate	Start (Day), e.g., '2020-05-01' for test simulation data, TestSimStartDate is usually before TestFieldStartDate
TestFieldStartDate	Start (Day), e.g., '2020-06-01' for test field data
verbosity	verbosity (int). Default: 0
parallel	logical
percCores	percentage

icu	ICU Daten
icuWeights	Gewichtung der ICU Betten
resourceNames	Name der Ressourcen
resourceEval	Name der zu evaluierenden Ressourcen
tryOnTestSet	Should results be tested on a separate test set?

ggVisualizeIcu *ggVisualizeIcu Visualisierung der ICU Daten*

Description

Quelle: ICU Daten bundesweit

Usage

```
ggVisualizeIcu(data = babsim.hospital::icudata, region = 5315)
```

Arguments

data	icu data, e.g., icudata
region	Region: Gemeindegemeinschaft, int 05374 fuer OBK oder lcode05315 fuer Koeln oder 05911 fuer Bochum.

Format

data.frame of 9 variables

bundesland int 1 1 1 1 1 1 1 1 1 ...

gemeindegemeinschaft int 1001 1002 1003 1004 1051 1053 1054 1055 1056 1057 ...

anzahl_meldebereiche int 2 3 2 1 1 2 1 3 2 1 ...

faelle_covid_aktuell int 0 3 5 1 3 1 0 0 5 1 ...

faelle_covid_aktuell_beatmet int 0 2 5 1 1 1 0 0 4 0 ...

anzahl_standorte int 2 3 2 1 1 2 1 3 2 1 ...

betten_frei int 44 113 115 19 54 7 7 18 10 7 ...

betten_belegt int 38 110 108 19 26 17 3 34 27 5 ...

daten_stand Date, format: "2020-05-01" "2020-05-01" "2020-05-01" "2020-05-01" ... "2020-08-21"

See Also

[icudata](#)

Examples

```

require("stats")
icu <- babsim.hospital::icudata
icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu))
icuCov$daten_stand <- as.Date(icuCov$daten_stand)
icuCovBeatm <- as.data.frame(xtabs(faelle_covid_aktuell_beatmet ~ daten_stand, icu))
icuCovBeatm$daten_stand <- as.Date(icuCovBeatm$daten_stand)
dataICUBeds <- data.frame(
  bed = (icuCov$Freq - icuCovBeatm$Freq),
  intensiveBedVentilation = icuCovBeatm$Freq,
  Day = as.Date(icuCovBeatm$daten_stand)
)

require("padr")
require("stats")
require("slider")

icu <- babsim.hospital::icudata
icu <- padr(icu, interval = "day")
icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu))
icuCovBeatm <- as.data.frame(xtabs(faelle_covid_aktuell_beatmet ~ daten_stand, icu))
icuCovBett <- as.data.frame(xtabs(betten_belegt ~ daten_stand, icu))
plot(icuCov$daten_stand, icuCov$Freq,
     type = "p", xlab = "Tag", ylab = "COVID ",
     main = "COVID-Faelle in Behandlung im KHaus"
)
plot(icuCovBeatm$daten_stand, icuCovBeatm$Freq,
     type = "p", xlab = "Tag",
     ylab = "Patienten", main = "Beatmete COVID-19-Pat. nur invasive Beatmung und ECMO"
)
plot(icuCovBett$daten_stand, icuCovBett$Freq, type = "l", xlab = "Tag", ylab = "ICU Betten belegt")

# Nur Daten des OBK:
icu <- babsim.hospital::icudata
icu <- icu[icu$gemeindeschluessel == 05374, ]

```

ggVisualizeRki

ggVisualizeRki ggPlot Visualisierung der RKI Daten

Description

Quelle: RKI Daten bundesweit

Usage

```

ggVisualizeRki(
  data = babsim.hospital::rkidata,
  region = 5374,
  StartDate = "2020-05-01"
)

```

Arguments

data	rki data, e.g., rkidata
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'

See Also[rkidata](#)

ggVisualizeRkiAge	<i>ggVisualizeRkiAge Visualisation of the pre-processed RKI Data with respect to age and gender</i>
-------------------	---

Description

ggplot RKI data as result from `getRkiData(babsim.hospital::rkidata)`

Usage

```
ggVisualizeRkiAge(  
  data = babsim.hospital::rkidata,  
  region = 5374,  
  StartDate = "2020-10-01",  
  simplify = TRUE  
)
```

Arguments

data	rki data as preprocessed by rkiToBabsimData
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'
simplify	logical. Simplify presentation, Default: TRUE.

See Also[rkiToBabsimData](#)

ggVisualizeRkiEvents *ggVisualizeRkiEvents Visualisation of the pre-processed RKI Data*

Description

ggplot RKI data as result from `getRkiData(babsim.hospital::rkidata)`

Usage

```
ggVisualizeRkiEvents(  
  data = getRkiData(babsim.hospital::rkidata),  
  region = 5374,  
  StartDate = "2020-10-01"  
)
```

Arguments

<code>data</code>	rki data as preprocessed by getRkiData
<code>region</code>	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
<code>StartDate</code>	Start (Tag), e.g., '2020-05-01'

See Also

[getRkiData](#)

Examples

```
# use 10000 data points only:  
data <- getRkiData(babsim.hospital::rkidata[1:10000, ])  
p <- ggVisualizeRkiEvents(data = data, region = 0, StartDate = "2020-10-01")
```

ggVisualizeRkiExtended
ggVisualizeRkiExtended Visualisation of the extended RKI Data

Description

ggplot RKI data as result from [extendRki](#)

Usage

```
ggVisualizeRkiExtended(  
  data = extendRki(),  
  region = 5374,  
  StartDate = "2020-10-01",  
  simplify = TRUE,  
  preloadedData = NULL  
)
```


Arguments

data	rki data as preprocessed by extendRki
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'
simplify	logical. Simplify presentation, Default: TRUE.
preloadedData	optional way to pass the result of preloading.

See Also[rkidata](#)

`ggVisualizeRkiExtendedDataCalculation`*Data precalculation for ggVisualizeRkiExtended*

Description

Data for ggplot RKI data as result from [extendRki](#)

Usage

```
ggVisualizeRkiExtendedDataCalculation(  
  data = extendRki(),  
  region = 5374,  
  StartDate = "2020-10-01",  
  simplify = TRUE  
)
```

Arguments

data	rki data as preprocessed by extendRki
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'
simplify	logical. Simplify presentation, Default: TRUE.

See Also[rkidata](#)

Examples

```
data <- getRkiData(babsim.hospital::rkidata[1:10000, ])
# data size sufficient?:
if (dim(data)[1]> 1e6){
  p <- ggVisualizeRkiExtended(
    data = extendRki(data),
    region = 5374, StartDate = "2020-10-01"
  )
}
```

icudata

icudata IntensivbettenDaten (Beispieldatensatz)

Description

ICU Beispiel-Datensatz (nur für Demonstrationszwecke). Der Beispieldatensatz `icudata` dient nur zu Demonstrationszwecken. Er besitzt das gleiche Format wie Tagesreports des DIVI Intensivregisters, siehe <https://www.divi.de/register/tagesreport>. Ziel des DIVI-Intensivregisters ist, die Verfügbarkeiten von Beatmungsbetten und von erweiterten Therapiemaßnahmen bei akutem Lungenversagen in Deutschland sichtbar zu machen. Eine weitere wissenschaftliche Nutzung der Daten ist nur mit Zustimmung der DIVI gestattet. Bitte kontaktieren Sie die wissenschaftliche Leitung des DIVI-Intensivregisters. Please contact DIVI e.V. if you are interested in the full data: <https://www.divi.de/register/anprechpartner-register>

Usage

```
icudata
```

Format

data.frame of 9 variables

bundesland int 1 1 1 1 1 1 1 1 1 ...

gemeindeschluessel int 1001 1002 1003 1004 1051 1053 1054 1055 1056 1057 ...

anzahl_meldebereiche int 2 3 2 1 1 2 1 3 2 1 ...

faelle_covid_aktuell int 0 3 5 1 3 1 0 0 5 1 ...

faelle_covid_aktuell_beatmet int 0 2 5 1 1 1 0 0 4 0 ...

anzahl_standorte int 2 3 2 1 1 2 1 3 2 1 ...

betten_frei int 44 113 115 19 54 7 7 18 10 7 ...

betten_belegt int 38 110 108 19 26 17 3 34 27 5 ...

daten_stand Date, format: "2020-05-01" "2020-05-01" "2020-05-01" "2020-05-01" ... "2020-08-21"

Details

A sample of the ICU data

koelnarchive	<i>koelnarchive archived koeln data</i>
--------------	---

Description

Data: koeln data generated with SPOT.

Usage

```
koelnarchive
```

Format

'data.frame': obs. of 28 variables:

y num 311 158 180 232 297 ...

x.1 num 10.55 17.91 3.19 9.5 17.71 ...

... ..

x.33 num 1.072 1.015 1.044 1.057 0.556 ...

Details

Result from the [runoptDirect](#) run.

mapAgeGroupToAge	<i>mapAgeGroupToAge</i>
------------------	-------------------------

Description

Calculate real valued age based on RKI age classes

Usage

```
mapAgeGroupToAge(x)
```

Arguments

x age vector

Value

age as a real value

mapAgeToAgeGroup	<i>mapAgeToAgeGroup</i>
------------------	-------------------------

Description

Calculate age based on RKI age classes

Usage

```
mapAgeToAgeGroup(x)
```

Arguments

x	age vector
---	------------

Value

age class

mapPToPara	<i>mapPToPara</i>
------------	-------------------

Description

mapPToPara accepts a nxn matrix. Its values will be mapped onto the probability entries of a [babsimHospitalPara](#) list.

Usage

```
mapPToPara(P = getMatrixP(), para)
```

Arguments

P	(num) nxn-dim matrix. Values will be mapped onto the probabilities in babsimHospitalPara . Names of these parameters can be obtained via getParameterName .
para	Parameter list, e.g., generated via babsimHospitalPara

Value

This function returns a parameter list.

Examples

```
para <- babsimHospitalPara()
P <- getMatrixP()
para <- mapPToPara(
  P = P,
  para = para
)
```

mapXToPara

*mapXToPara***Description**

mapXToPara accepts a n-dim vector. Its values will be mapped onto a [babsimHospitalPara](#) list.

Usage

```
mapXToPara(x)
```

Arguments

x (num) n-dim vector. Values will be mapped onto [babsimHospitalPara](#). Names of these parameters can be obtained via [getParameterName](#).

Details

This function will be replaced by the function `simulateHospital` in versions $\geq 1.2.8$.

Value

This function returns an env data frame (tibble [560 × 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

resource (chr) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...

time (num) time step: 3 10 12 13 14 15 15 15 15 16 ...

server (int) server: 1 2 3 2 3 4 3 4 5 6 ...

limit (num) limit: Inf Inf Inf Inf Inf ...

replication (int) replication: 1 1 1 1 1 1 1 1 1 ...

upper (int) upper: 1 2 3 2 3 5 5 5 7 ...

lower (int) lower: 1 2 3 2 3 3 3 3 5 ...

med (num) med: 1 2 3 2 3 4 4 4 6 ...

date (POSIXct) time, format: yyyy-mm-dd hh:mm:ss

startdate (POSIXct) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...

source (chr) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

Examples

```
x <- rep(0.2, 29)
para <- mapXToPara(x)
conf <- babsimToolsConf()
data <- getObkData()
res <- modelResultHospital(para = para, conf = conf, data = data)
getError(res = res, conf = conf)
p <- plotDailyMaxResults(res)
```

messageDateRange	<i>Display date range of a vector</i>
------------------	---------------------------------------

Description

Utility function to display the range of dates in a vector.

Usage

```
messageDateRange(prefix, d)
```

Arguments

prefix	string to print before outputting range.
d	vector of values.

Value

Nothing, called for the side effect.

messagef	<i>Output a formatted message</i>
----------	-----------------------------------

Description

Output a formatted message

Usage

```
messagef(fmt, ...)
```

Arguments

fmt	format string (see sprintf for details)
...	values passed into fmt. Only logical, integer, real and character vectors are supported, but some coercion will be done.

Value

Nothing, called for the side effect of outputting a message to the console.

modelResultHospital *modelResultHospital*

Description

Simulate one parameter configuration from [babsimHospitalPara](#). The simulation is by default deterministic, because `conf$seed` is used for [set.seed](#).

Usage

```
modelResultHospital(para, conf, data)
```

Arguments

para	Simulation model parameters. The function babsimHospitalPara can be used to generate these parameters.
conf	list with the following entries: seed seed. Change the seed value to get different output for the same input parameters. Default: 123 simRepeats simmer repeats parallel simmer parallel runs. Default: FALSE perCores percentage of cores used for parallel simmer simulations. Default: 0.5 (=50 percent) ICU use ICU infection data. Default: FALSE logLevel log leved (0 or 1). Default: 0 (no output) maxCapacity max capacity of resources. Default: 1e6 dataset char name of the data set. Default: 'GA' simulationDates list with StartDate and EndDate. Period that is used for the simulation (babsim, simmer). Default: list(StartDate = '2020-03-03', EndDate = '2020-06-24') fieldDates list with StartDate and EndDate. Period when real data is available (resource usage). Default: list(StartDate = '2020-03-03', EndDate = '2020-06-24') simulationData data frame. Data used for the simulation. Default: dataCovidBeds20200624 fieldEvents data frame. Data used for the evaluation (error). Default: GABeds220200624 resource vector with resource names. Default: c('bed', 'intensiveBed', 'intensiveBedVentilation')
data	list with simData and fieldData

Value

This function returns an env data frame (tibble [560 × 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

resource (**chr**) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...
 time (**num**) time step: 3 10 12 13 14 15 15 15 16 ...
 server (**int**) server: 1 2 3 2 3 4 3 4 5 6 ...
 queue (**int**) 1 1 2 3 4 3 4 1 0 2 ...
 capacity (**num**) 10000 10000 10000 10000 10000 10000 10000 10000 10000 10000 ...
 queue_size (**num**) Inf Inf Inf Inf Inf ...
 system (**int**) 1 1 2 3 4 3 4 1 0 2 ...
 limit (**num**) limit: Inf Inf Inf Inf Inf ...
 replication (**int**) replication: 1 1 1 1 1 1 1 1 1 1 ...
 upper (**int**) upper: 1 2 3 2 3 5 5 5 5 7 ...
 lower (**int**) lower: 1 2 3 2 3 3 3 3 3 5 ...
 med (**num**) med: 1 2 3 2 3 4 4 4 4 6 ...
 date (**POSIXct**) time, format: yyyy-mm-dd hh:mm:ss
 rwdate (**POSIXct**) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...
 source (**chr**) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

Examples

```
# First example: OBK data
data <- getObkData()
para <- babsimHospitalPara()
para$GammaShapeParameter <- 0.8
# turn off parallelized simulation:
conf <- babsimToolsConf()
conf <- getConfFromData(
  conf = conf,
  simData = data$simData,
  fieldData = data$fieldData
)
# no logging (default)
conf$logLevel <- 0
res <- modelResultHospital(
  para = para,
  conf = conf,
  data = data
)

# Second example: synthetic data
data <- getSyntheticData()
para <- babsimHospitalPara()
conf <- babsimToolsConf()
conf <- getConfFromData(
```



```
    conf = conf,  
    simData = data$simData,  
    fieldData = data$fieldData  
  )  
  res <- modelResultHospital(para = para, conf = conf, data = data)
```

nrwarchive

nrwarchive archived nrw data

Description

Data: nrw data generated with SPOT.

Usage

```
nrwarchive
```

Format

'data.frame': obs. of 28 variables:

y num 311 158 180 232 297 ...

x.1 num 10.55 17.91 3.19 9.5 17.71 ...

... ..

x.33 num 1.072 1.015 1.044 1.057 0.556 ...

Details

Result from the [runoptDirect](#) run.

obkarchive

obkarchive archived obk data

Description

Data: OBK data generated with SPOT.

Usage

```
obkarchive
```

Format

'data.frame': obs. of 28 variables:

y num 311 158 180 232 297 ...

x.1 num 10.55 17.91 3.19 9.5 17.71 ...

... ..

x.33 num 1.072 1.015 1.044 1.057 0.556 ...

Details

Result from the [runoptDirect](#) run. to extract the best parameter set x.

paras

paras data

Description

Data: Parameters for all regions, generated through SPOT optimization

Usage

paras

Format

'data.frame' obs. of 31 variables:

y num 311 158 180 232 297 ...

x.1 num 10.55 17.91 3.19 9.5 17.71 ...

... ..

x.29 num 1.072 1.015 1.044 1.057 0.556 ...

region num 5315

Details

Result from the [runoptDirect](#) run.

plotDailyMaxResults *plotDailyMaxResults*

Description

Plot output from `getDailyMax()`.

Usage

```
plotDailyMaxResults(  
  results,  
  labels = c("babsim", "DIVI"),  
  title = "Betten: Tuerkis = Readfssldaten, Rot = Simulation",  
  showBeds = FALSE,  
  icuDataRegion = NULL  
)
```

Arguments

<code>results</code>	Results from <code>getDailyMax</code> .
<code>labels</code>	Axes labels (vector). Default: <code>c('babsim', 'DIVI')</code>
<code>title</code>	Title. Default: <code>'Betten: Tuerkis = Realdaten, Rot = Simulation'</code>
<code>showBeds</code>	should normal beds be shown in the plot?
<code>icuDataRegion</code>	regional icudata

Value

This function returns a `ggplot` object.

Examples

```
set.seed(123)  
# 1. Generate simulation data based on number of infected persons per day:  
x <- dataCovidBeds20200624  
StartDate <- x$Day[1]  
EndDate <- x$Day[length(x$Day)]  
arrivalTimes <- getArrivalTimes(x$Infected)  
para <- babsimHospitalPara()  
conf <- babsimToolsConf()  
y <- babsimHospital(  
  arrivalTimes = arrivalTimes,  
  conf = conf,  
  para = para  
)  
  
# 2. Extract real data:  
fieldEvents <- getRealBeds(  

```

```

    data = babsim.hospital::dataCovidBeds20200624,
    resource = c("bed", "intensiveBed", "intensiveBedVentilation")
  )
  conf <- babsimToolsConf()
  # 3. Combine simulated and real data:
  res <- getDailyMaxResults(
    envs = y,
    fieldEvents = fieldEvents,
    conf = conf
  )
  # 4. Plot results
  p <- plotDailyMaxResults(res)
  # print(p)

```

`plotPostprocessedEnvs` *plotPostprocessedEnvs*

Description

Plot output from [postprocessEnvs](#).

Usage

```
plotPostprocessedEnvs(results)
```

Arguments

`results` Results from [postprocessEnvs](#).

Value

This function returns a `ggplot` object.

Examples

```

set.seed(123)
# 1. Generate simulation data based on number of infected persons per day:
x <- dataCovidBeds20200624
StartDate <- x$Day[1]
EndDate <- x$Day[length(x$Day)]
arrivalTimes <- getArrivalTimes(x$Infected)
para <- babsimHospitalPara()
conf <- babsimToolsConf()
y <- babsimHospital(
  arrivalTimes = arrivalTimes,
  conf = conf,
  para = para
)

# 2. Postprocess simmer environment:

```

```

res <- postprocessEnvs(envs = y, StartDate = "2020-03-03")
# 4. Plot results
p <- plotPostprocessedEnvs(res)
# print(p)

```

postprocessEnvs	<i>postprocessEnvs</i>
-----------------	------------------------

Description

Postprocess results from several `simmer` results. Input: `simmer` simulation environment. The method `get_mon_resources` function is used to extract information from the `babsim.hospital` simulation.

Usage

```
postprocessEnvs(envs, StartDate = "2020-03-03")
```

Arguments

<code>envs</code>	<code>simmer</code> simulation environment. Result from <code>babsim.hospital</code> simulation runs, e.g., output from <code>babsimHospital</code> .
<code>StartDate</code>	<code>Date[1:1]</code> , format: 'YYYY-MM-DD' First day of the simulation. Default: '2020-03-03'.

Details

`get_mon_resources` returns state changes in resources:

- 'resource': resources name
- 'time': time instant of the event that triggered the state change
- 'server': server count
- 'queue': queue count
- 'capacity': capacity
- 'queue_size': queue size
- 'system': system count (server + queue). If no queues are used, system values equal server values.
- 'system_limit': system limit (capacity + queue_size)

Value

This function returns an env data frame (tibble [nxm, 15] (S3: grouped_df/tbl_df/tbl/data.frame)) with the following entries:

```

resource (chr) name of the seized resource: 'bed' 'bed' 'bed' 'bed' ...
time (num) time step: 3 10 12 13 14 15 15 15 15 16 ...

```

```

server (int) server: 1 2 3 2 3 4 3 4 5 6 ...
limit (num) limit: Inf Inf Inf Inf Inf ...
replication (int) replication: 1 1 1 1 1 1 1 1 1 ...
upper (int) upper: 1 2 3 2 3 5 5 5 7 ...
lower (int) lower: 1 2 3 2 3 3 3 3 5 ...
med (num) med: 1 2 3 2 3 4 4 4 6 ...
date (POSIXct) time, format: yyyy-mm-dd hh:mm:ss
rwdate (POSIXct) format: '2020-03-01' '2020-03-08' '2020-03-15' '2020-03-15' ...
source (chr) name of the simulation that was used: 'babsim' 'babsim' 'babsim' 'babsim' ...

```

plotPostprocessedEnvs

```
NA
```

Examples

```

set.seed(123)
# 1. Generate simulation data based on number of infected persons per day:
x <- dataCovidBeds20200624
StartDate <- x$Day[1]
EndDate <- x$Day[length(x$Day)]
arrivalTimes <- getArrivalTimes(x$Infected)
para <- babsimHospitalPara()
conf <- babsimToolsConf()
y <- babsimHospital(
  arrivalTimes = arrivalTimes,
  conf = conf,
  para = para
)

# 2. Postprocess simulation results:
res <- postprocessEnvs(envs = y)
# 3. Plot results
p <- plotPostprocessedEnvs(res)

```

```
printConf
```

```
printConf
```

Description

Print configuration information, e.g., for debugging. This function returns the configuration settings of the `babsim.hospital` functions used to run `babsimHospital`. Currently, only `str` is used.

Usage

```
printConf(conf)
```

Arguments

conf Configuration, e.g., generated with [babsimToolsConf](#).

Details

Configuration conf is a list of the following settings.

seed (int) Initial seed. Default: 123

simRepeats (int) Number of [simmer](#) simulation runs. Default: 1

parallel (logical) Use parallel simulations based on [mclapply](#). Default: FALSE

perCores (num) Percentage of cores used, if parallel == TRUE. Default: 0.5

ICU (logical) Use ICU (RKI) data. Default: FALSE.

logLevel (int) 0 = no logging, >= 1 logging. Default: 0. If larger than 10, shown detailed simmer output.

maxCapacity (num) Maximum capacity used for [babsimHospital](#) resources. Default: 1e6.

dataset (chr) 'GA' or 'ICU'. Default: 'GA'.

simulationDates List with the following entries:

StartDate (chr) Start date of the simulation data (infection data used to generate arrival times), first day. Default: '2020-03-03'

EndDate (chr) End date of the simulation data, last day. Default: '2020-06-24'

fieldDates List with the following entries:

StartDate (chr) Start date of the field (resources) data, first day. Default: '2020-03-03'

EndDate (chr) End date of the field data, last day. Default: '2020-06-24'

simulationData (data frame) Data used for the simulation. Default [dataCovidBeds20200624](#)

bed int 2 2 3 3 3 3 3 3 4 4 ...

intensiveBed int 0 0 0 0 0 0 0 0 0 ...

intensiveBedVentilation int 0 0 0 0 0 0 0 0 0 ...

Day Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

Infected num 5 0 0 0 0 0 0 7 2 5 ...

Sick num 5 5 5 5 5 5 5 12 14 19 ...

fieldEvents (data frame) Data used for evaluation of the simulation. Default [GABeds220200624](#)

resource chr 'bed' 'bed' 'bed' 'bed' ...

time int 1 2 3 4 5 6 7 8 9 10 ...

med int 2 2 3 3 3 3 3 3 4 4 ...

source int 2 2 3 3 3 3 3 3 4 4 ...

date Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

resource (vector) Resources used in the simulation. Default: c('bed', 'intensiveBed', 'intensiveBedVentilation').
For ICU data use: c('bed', 'intensiveBedVentilation')

resource chr 'bed' 'bed' 'bed' 'bed' ...

time int 1 2 3 4 5 6 7 8 9 10 ...

med int 2 2 3 3 3 3 3 3 4 4 ...

source int 2 2 3 3 3 3 3 3 4 4 ...

date Date, format: '2020-03-03' '2020-03-04' '2020-03-05' '2020-03-06' ...

Value

conf elements

Examples

```
conf <- babsimToolsConf()
# turn on parallel simulation:
conf$parallel <- TRUE
# Change the start date of the simulations
printConf(conf = conf)
```

RiskScore

Calculate risk scores

Description

Calculate a risk score based on age and sex of patient for a set of cases.

Usage

```
RiskScore(cases, par)
```

Arguments

cases	[object] table of cases, one per row.
par	[list()] parameters of risk model.

Details

The risk score is calculated according to the following formula:

$$\text{RiskScore} = \text{RiskFactorA} \exp(\text{RiskFactorBage})(1 + I_{\text{male}}(\text{RiskMale} - 1))$$

Here the RiskFactorA, RiskFactorB and RiskMale are taken from the parameter list par and the age and sex are taken from the cases .

Value

Vector of risk scores.

rkidata	<i>rkidata: RKI COVID-19 Daten (complete, not included in CRAN version)</i>
---------	---

Description

RKI Daten komplett

Usage

rkidata

Format

a data.frame of of 18 variables

FID int 40613780 40613781 40613782 40613783 40613784 40613785 40613786 40613787 40613788
40613789 ...

IdBundesland 1 1 1 1 1 1 1 1 1 1 ...

Bundesland chr "Schleswig-Holstein" "Schleswig-Holstein" "Schleswig-Holstein" "Schleswig-Holstein"
...

Landkreis chr "SK Flensburg" "SK Flensburg" "SK Flensburg" "SK Flensburg" ...

Altersgruppe chr "A00-A04" "A00-A04" "A00-A04" "A05-A14" ...

Geschlecht chr "M" "W" "W" "M" ...

AnzahlFall int 1 1 1 1 1 1 1 1 1 1 ...

AnzahlTodesfall ...

Melddatum chr "2020/09/30 00:00:00" "2020/08/24 00:00:00" "2020/09/26 00:00:00" "2020/09/25
00:00:00" ...

IdLandkreis int 1001 1001 1001 1001 1001 1001 1001 1001 1001 1001 ...

Datenstand "04.10.2020, 00:00 Uhr" "04.10.2020, 00:00 Uhr" "04.10.2020, 00:00 Uhr" "04.10.2020,
00:00 Uhr" ...

NeuerFall int 0 0 0 0 0 0 0 0 0 0 ...

NeuerTodesfall int -9 -9 -9 -9 -9 -9 -9 -9 -9 -9 ...

Refdatum chr "2020/09/30 00:00:00" "2020/08/24 00:00:00" "2020/09/26 00:00:00" "2020/09/21
00:00:00" ...

NeuGenesen int -9 0 -9 -9 -9 0 -9 -9 0 ...

AnzahlGenesen int 0 1 0 0 0 0 1 0 0 1 ...

IstErkrankungsbeginn int 0 0 0 1 1 0 0 1 0 1 ...

Altergruppe2 chr "Nicht übermittelt" "Nicht übermittelt" "Nicht übermittelt" "Nicht übermittelt"
...

Details

Heruntergeladen mittels `wget https://www.arcgis.com/sharing/rest/content/items/f10774f1c63e40168479a1feb`
 Eingelesen mittels `rkiData <-read.csv("data", header=TRUE, encoding="UTF-8")` Eingepflegt
 mittel `useThis::use_data(rkiData)`

Weiterverarbeitung z.B. mit: `rkiDataObk <-rkiData[rkiData$Landkreis == "LK Oberbergischer Kreis",]`
`rkiDataObk$Meldedatum <-as.Date(rkiDataObk$Meldedatum)` `sum(rkiDataObk$AnzahlFall)`
`sum(rkiDataObk$AnzahlTodesfall)` `rkiDataObkAgg <-as.data.frame(xtabs(AnzahlFall ~`
`Meldedatum, rkiDataObk))` `plot(rkiDataObkAgg$Meldedatum, rkiDataObkAgg$Freq)`

Ein `data.frame` mit 18 Variablen

`rkiGeschlechtToSex` *Map Geschlecht to biological sex*

Description

Map Geschlecht to biological sex

Usage

`rkiGeschlechtToSex(geschlecht)`

Arguments

`geschlecht` [character(n)]
 character vector with values in ‘M’, ‘W’, and ‘unbekannt’.

Value

A factor vector with `n` elements and levels ‘male’ or ‘female’. ‘unbekannt’ or other values are mapped to NA.

`rkiToBabsimArrivals` *rkiToBabsimArrivals*

Description

Transforms the freshly downloaded rki data into the babsim fitting format of arrival times. Also imputes missing dates as zero frequency in the data.

Usage

`rkiToBabsimArrivals(rki)`

Arguments

`rki` data.frame of downloaded rki data before preprocessing

Value

a data.frame of arrival times suited for babsimHospital

Examples

```
arrivals <- rkiToBabsimArrivals(rkidata[1:100, ])
min(as.Date(rkidata$Refdatum)) + max(arrivals)
```

<code>rkiToBabsimData</code>	<i>rkiToBabsimData</i>
------------------------------	------------------------

Description

Transforms the freshly downloaded rki data into the babsim data frame. Also imputes missing dates as zero frequency in the data.

Usage

```
rkiToBabsimData(rki = babsim.hospital::rkidata)
```

Arguments

`rki` data.frame of downloaded rki data before preprocessing

Value

a data.frame of aggregated data

Day Date, format: '2020-01-01' '2020-01-02' '2020-01-03' '2020-01-04' ...

Infected Infiziert: num 1 0 0 0 0 0 0 0 0 ...

Weiblich Geschlecht weiblich: int 0 0 0 0 0 0 0 0 0 ...

Maennlich Geschlecht maennlich: int 1 0 0 0 0 0 0 0 0 ...

GUnbekannt Geschlecht unbekannt: int 0 0 0 0 0 0 0 0 0 ...

Examples

```
data <- rkiToBabsimData(rkidata[1:100, ])
plot(data$Day, data$Infected, type = "o")
#
max(data$Day) - min(data$Day)
```

rtgamma	<i>rtgamma</i>
---------	----------------

Description

Random generation for the shifted and truncated Gamma distribution with parameters shape and scale.

Usage

```
rtgamma(n = 1, shape = 1, rate = 1, shift = 0, alpha = 0.95)
```

Arguments

n	number of observations
shape	Gamma shape parameter
rate	Gamma rate parameter
shift	shift parameter.
alpha	upper quantile of gamma distribution. All values above alpha are truncated.

Value

rtgamma generates random deviates. The length of the result is determined by n.

Examples

```
rtgamma(n = 1, shape = 1, rate = 1, shift = 1, alpha = 0.95)
```

runoptDirect	<i>runoptDirect Optimierung der babsim.hospital Parameter</i>
--------------	---

Description

SPOT Aufruf zur Optimierung der babsim Parameter mit Lasso

Usage

```
runoptDirect(
  expName = "obkpara20201017",
  rkiwerte = babsim.hospital::rkidata,
  icuwerte = babsim.hospital::icudata,
  region = 5374,
  TrainFieldStartDate = NULL,
  TrainSimStartDate = NULL,
```

```

    TestFieldStartDate = NULL,
    TestSimStartDate = NULL,
    Overlap = 7,
    verbosity = 0,
    seed = 123,
    direct = FALSE,
    repeats = 1,
    funEvals = 35,
    funEvalsFactor = 0,
    size = 30,
    simrepeats = 2,
    subset = 32,
    parallel = FALSE,
    percCores = NULL,
    icu = TRUE,
    icuWeights = 1,
    testRepeats = 3,
    resourceNames = c("intensiveBed", "intensiveBedVentilation"),
    resourceEval = c("intensiveBed", "intensiveBedVentilation"),
    spotEvalsParallel = FALSE,
    tryOnTestSet = TRUE
)

```

Arguments

expName	Experiment Name
rkiwerte	RKI Daten
icuwerte	ICU Daten
region	Landkreis Id, e.g., 5374 fuer OBK, 5315 fuer Koeln, 0 fuer Deutschland, oder Bundesland ID, e.g., 5 fuer NRW.
TrainFieldStartDate	Start (Tag), e.g., "2020-06-01"
TrainSimStartDate	Start (Tag), e.g., "2020-05-01"
TestFieldStartDate	Start (Day), e.g., "2020-06-01" for test field data
TestSimStartDate	Start (Day), e.g., "2020-05-01" for test simulation data, TestSimStartDate is usually before TestFieldStartDate
Overlap	integer. Days, train data will be extended (overlap with test data). Default: 7
verbosity	verbosity (int). Default: 0
seed	Seed
direct	use model-free optimization. Default: FALSE
repeats	Wiederholungen fuer SPOT (Optimierungslaeufe mit unterschiedlichem Seed)
funEvals	Auswertungen fuer SPOT (Simulationen, die fuer einen SPOT Lauf zur Verfuegung stehen)

funEvalsFactor	factor to increase function evaluations. Default: 0
size	Groesse des initialen Desings
simrepeats	Sim Wdhlg
subset	Subset (SPOT)
parallel	logical
percCores	percentage
icu	ICU Daten
icuWeights	Gewichtung der ICU Betten
testRepeats	number of final evaluations on the test data
resourceNames	Name der Ressourcen
resourceEval	Name der zu evaluierenden Ressourcen
spotEvalsParallel	Should the spot repeats be evaluated in parallel?
tryOnTestSet	Should results be tested on a separate test set?

runOptLocal

runOptLocal

Description

Run a local optimization on a set of parameters

Usage

```
runOptLocal(
  X,
  rkiwerte = babsim.hospital::rkidata,
  icuwerte = babsim.hospital::icudata,
  region = 5374,
  TrainFieldStartDate = NULL,
  TrainSimStartDate = NULL,
  verbosity = 0,
  seed = 123,
  funEvals = 35,
  simrepeats = 2,
  parallel = FALSE,
  percCores = NULL,
  icu = TRUE,
  icuWeights = 1,
  resourceNames = c("intensiveBed", "intensiveBedVentilation"),
  resourceEval = c("intensiveBed", "intensiveBedVentilation"),
  spotEvalsParallel = FALSE
)
```

Arguments

X	matrix of parameters. Each row should contain one parameterset to which a local optimization is applied
rkiwerte	RKI Daten
icuwerte	ICU Daten
region	Landkreis Id, e.g., 5374 fuer OBK, 5315 fuer Koeln, 0 fuer Deutschland, oder Bundesland ID, e.g., 5 fuer NRW.
TrainFieldStartDate	Start (Tag), e.g., "2020-06-01"
TrainSimStartDate	Start (Tag), e.g., "2020-05-01"
verbosity	verbosity (int). Default: 0
seed	Seed
funEvals	Auswertungen fuer SPOT (Simulationen, die fuer einen SPOT Lauf zur Verfuegung stehen)
simrepeats	Sim Wdhlg
parallel	logical
percCores	percentage
icu	ICU Daten
icuWeights	Gewichtung der ICU Betten
resourceNames	Name der Ressourcen
resourceEval	Name der zu evaluierenden Ressourcen
spotEvalsParallel	Should the spot repeats be evaluated in parallel?

skip_if_quicktest *Skip test if in quicktest mode*

Description

Skip test if option 'babsim.hospital.quicktest' is TRUE or if the environment variable 'BABSIM_HOSPITAL_QUICKTEST' is set.

Usage

```
skip_if_quicktest()
```

smoothParameter	<i>Smooth a parameter set using another parameter set</i>
-----------------	---

Description

Calculate the average of two parameter sets to smooth out any local anomalies. Mostly useful to smooth out a local (say OBK) parameter set using a global one (say NRW).

Technically this function calculates $(1-\text{weight}) * \text{para} + \text{weight} * \text{other}$ ensuring that the names etc. of para are preserved.

Usage

```
smoothParameter(para, other, weight = 0.2)
```

Arguments

para	Parameter set to smooth
other	Other parameters to average in
weight	Weight of other parameters

Value

Weighted parameter set

softmax	<i>softmax</i>
---------	----------------

Description

softmax function

Usage

```
softmax(par)
```

Arguments

par	vector
-----	--------

Value

num vector with components ≥ 0 and sum = 1

Examples

```
p <- c(0.6, 0.3, 0.1)
softmax(p)
```

 switchRkiRefdatumToMeldedatum

switchRkiRefdatumToMeldedatum Change RKI data File

Description

Use Meldedatum instead of Refdatum in [rkidata](#). It is recommended to run install and restart the package afterwards.

Usage

```
switchRkiRefdatumToMeldedatum(rkidata = rkidata, overwrite = TRUE)
```

Arguments

rkidata	RKI Data
overwrite	logical Overwrite existing file. Default TRUE.

 synthpara

synthpara data

Description

Data: Synthetic data generated with SPOT.

Usage

```
synthpara
```

Format

'data.frame': obs. of 34 variables:

y num 311 158 180 232 297 ...

x.1 num 10.55 17.91 3.19 9.5 17.71 ...

... ..

x.33 num 1.072 1.015 1.044 1.057 0.556 ...

Details

Result from the [runoptDirect](#) run. Use `yx <- synthpara[synthpara$y == min(synthpara$y),]`
`x <- yx[1, 2:34]` to extract the best parameter set x.

updateIcudataFile	<i>updateIcudataFile</i> Update ICU data File
-------------------	---

Description

Update icudata (divi)

Usage

```
updateIcudataFile(OldData = babsim.hospital::icudataFull, overwrite = TRUE)
```

Arguments

OldData	Old icu (DIVI) data, default babsim.hospital::icudata.
overwrite	logical Overwrite existing file. Default TRUE.

Value

True if new data was downloaded, otherwise false

updateMatrixP	<i>updateMatrixP</i>
---------------	----------------------

Description

Updates the probability matrix

Usage

```
updateMatrixP(P = getMatrixP(), u)
```

Arguments

P	matrix P. Default getMatrixP
u	list of factors used for the update

Value

a matrix with updated transition probabilities

Examples

```
u <- list(k = 2)
R <- updateMatrixP(u = u)
```

updateParaSet	<i>updateParaSet</i>
---------------	----------------------

Description

Delete old parameters from paras and replace them with new ones.

Usage

```
updateParaSet(paramDF, region, path = NULL)
```

Arguments

paramDF	the data frame with the new parameters that should replace the old ones
region	integer, the region id
path	path to the old para file.

Value

parameter set

Examples

```
getParaSet(5315)
```

updateRkidataFile	<i>updateRkidataFile Update RKI data File</i>
-------------------	---

Description

Update rkidata. Download RKI data from the RKI Server.

Usage

```
updateRkidataFile(fileName, overwrite = TRUE)
```

Arguments

fileName	RKI Filename, e.g., 'https://www.arcgis.com/sharing/rest/content/items/f10774f1c63e40168'
overwrite	logical Overwrite existing file. Default TRUE.

Details

Because the downloaded RKI data include data from January 2020 (first COVID-19 wave in Germany) until today (or the day before today), the original data are stored as `rkidataFull`. `babsim.hospital` simulations require data starting from September 2020 (second COVID-19 wave in Germany). Therefore, a limited data set starting from Sep 1st is stored as `rkidata` in the `babsim.hospital` package. Furthermore, because no nowcasting is implemented in the current version of the `babsim.hospital` simulator, the `Meldedatum` information will be used instead of the `Refdatum` information: the corresponding columns in `rkidata` are renamed, because all functions in `babsim.hospital` use the `Refdatum` information.

Creates two *.rda files in the folder `data`: 1. `babsim.hospital::rkidata` and 2. `babsim.hospital::rkidataFull`. Both data frames contain the following 18 variables

```
FID int 40613780 40613781 40613782 40613783 40613784 40613785 40613786 40613787 40613788
      40613789 ...
IdBundesland int 1 1 1 1 1 1 1 1 1 1 ...
Bundesland chr 'Schleswig-Holstein' 'Schleswig-Holstein' 'Schleswig-Holstein' 'Schleswig-Holstein'
      ...
Landkreis chr 'SK Flensburg' 'SK Flensburg' 'SK Flensburg' 'SK Flensburg' ...
Altersgruppe chr 'A00-A04' 'A00-A04' 'A00-A04' 'A05-A14' ...
Geschlecht chr 'M' 'W' 'W' 'M' ...
AnzahlFall int 1 1 1 1 1 1 1 1 1 1 ...
AnzahlTodesfall int ...
Meldedatum chr '2020/09/30 00:00:00' '2020/08/24 00:00:00' '2020/09/26 00:00:00' '2020/09/25
      00:00:00' ...
IdLandkreis int 1001 1001 1001 1001 1001 1001 1001 1001 1001 1001 ...
Datenstand chr '04.10.2020, 00:00 Uhr' '04.10.2020, 00:00 Uhr' '04.10.2020, 00:00 Uhr' '04.10.2020,
      00:00 Uhr' ...
NeuerFall int 0 0 0 0 0 0 0 0 0 0 ...
NeuerTodesfall int -9 -9 -9 -9 -9 -9 -9 -9 -9 -9 ...
Refdatum chr '2020/09/30 00:00:00' '2020/08/24 00:00:00' '2020/09/26 00:00:00' '2020/09/21
      00:00:00' ...
NeuGenesen int -9 0 -9 -9 -9 0 -9 -9 0 ...
AnzahlGenesen int 0 1 0 0 0 0 1 0 0 1 ...
IstErkrankungsbeginn int 0 0 0 1 1 0 0 1 0 1 ...
Altergruppe2 chr 'Nicht übermittelt' 'Nicht übermittelt' 'Nicht übermittelt' 'Nicht übermittelt'
      ...
```

Value

True if new data was downloaded, otherwise false

vaccineCounts	<i>Amount of vaccinated people in germany</i>
---------------	---

Description

Data: Amount of vaccinated people in germany. Collected by day and state. Additionally the vaccine-quota (vaccinated/capita) is stored

Usage

```
vaccineCounts
```

Format

'data.frame': obs. of 4 variables:

vaccinated num 311 158 180 232 297 ...

quote num 10.55 17.91 3.19 9.5 17.71 ...

state char Berlin Bayern ...

date date 2020-12-29 2020-12-30 ...

visualizeGraph	<i>visualizeGraph Visualisierung der Wahrscheinlichkeiten und Dauern</i>
----------------	--

Description

Ergebnisse

Usage

```
visualizeGraph(para = babsimHospitalPara(), option = "P")
```

Arguments

para parameter

option Option: plot probabilities ('P') or durations ('D')

Examples

```
para <- babsimHospitalPara()
visualizeGraph(para = para, option = "P")
```

visualizeIcu	<i>visualizeIcu Visualisierung der ICU Daten</i>
--------------	--

Description

Quelle: ICU Daten bundesweit

Usage

```
visualizeIcu(data = babsim.hospital::icudata, region = 5315)
```

Arguments

data	icu data, e.g., icudata
region	Region: Gemeindegchlüssel, int 05374 fuer OBK oder lcode05315 fuer Koeln oder 05911 fuer Bochum.

Format

data.frame of 9 variables

```
bundesland int 1 1 1 1 1 1 1 1 1 1 ...
gemeindegchlüssel int 1001 1002 1003 1004 1051 1053 1054 1055 1056 1057 ...
anzahl_meldebereiche int 2 3 2 1 1 2 1 3 2 1 ...
faelle_covid_aktuell int 0 3 5 1 3 1 0 0 5 1 ...
faelle_covid_aktuell_beatmet int 0 2 5 1 1 1 0 0 4 0 ...
anzahl_standorte int 2 3 2 1 1 2 1 3 2 1 ...
betten_frei int 44 113 115 19 54 7 7 18 10 7 ...
betten_belegt int 38 110 108 19 26 17 3 34 27 5 ...
daten_stand Date, format: '2020-05-01' '2020-05-01' '2020-05-01' '2020-05-01' ... '2020-08-21'
```

See Also

[icudata](#)

Examples

```
require("stats")
icu <- babsim.hospital::icudata
icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu))
icuCov$daten_stand <- as.Date(icuCov$daten_stand)
icuCovBeatm <- as.data.frame(xtabs(faelle_covid_aktuell_beatmet ~ daten_stand, icu))
icuCovBeatm$daten_stand <- as.Date(icuCovBeatm$daten_stand)
dataICUBeds <- data.frame(
  bed = (icuCov$Freq - icuCovBeatm$Freq),
```

```

    intensiveBedVentilation = icuCovBeatm$Freq,
    Day = as.Date(icuCovBeatm$daten_stand)
  )

  require("padr")
  require("stats")
  icu <- babsim.hospital::icudata
  icu <- pad(icu, interval = "day")
  icuCov <- as.data.frame(xtabs(faelle_covid_aktuell ~ daten_stand, icu))
  icuCovBeatm <- as.data.frame(xtabs(faelle_covid_aktuell_beatmet ~ daten_stand, icu))
  icuCovBett <- as.data.frame(xtabs(betten_belegt ~ daten_stand, icu))
  plot(icuCov$daten_stand, icuCov$Freq,
       type = "p", xlab = "Tag", ylab = "COVID ",
       main = "COVID-Faelle in Behandlung im KHaus"
  )
  plot(icuCovBeatm$daten_stand, icuCovBeatm$Freq,
       type = "p", xlab = "Tag",
       ylab = "Patienten", main = "Beatmete COVID-19-Pat. nur invasive Beatmung und ECMO"
  )
  plot(icuCovBett$daten_stand, icuCovBett$Freq, type = "l", xlab = "Tag", ylab = "ICU Betten belegt")

  # Nur Daten des OBK:
  icu <- babsim.hospital::icudata
  icu <- icu[icu$gemeindeschluessel == 05374, ]

```

 visualizeRki

visualizeRki Visualisierung der RKI Daten

Description

Quelle: RKI Daten bundesweit

Usage

```

visualizeRki(
  data = babsim.hospital::rkidata,
  region = 5374,
  StartDate = "2020-05-01"
)

```

Arguments

data	rki data, e.g., rkidata
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'

See Also

[rkidata](#)

visualizeRkiEvents	<i>visualizeRkiEvents</i> Visualisation of the pre-processed RKI Data
--------------------	---

Description

RKI data as result from `getRkiData(babsim.hospital::rkidata)`

Usage

```
visualizeRkiEvents(
  data = getRkiData(babsim.hospital::rkidata),
  region = 0,
  StartDate = "2020-05-01"
)
```

Arguments

data	rki data as preprocessed by getRkiData
region	Landkreis Id, e.g., 5374 oder Bundesland ID, e.g., 5.
StartDate	Start (Tag), e.g., '2020-05-01'

See Also

[getRkiData](#)

Examples

```
p <- visualizeRkiEvents(getRkiData(babsim.hospital::rkidata[1:1000, ]))
```

weighted_rmse	<i>weighted_rmse</i>
---------------	----------------------

Description

Calculate a weighted RMSE. Weights are based on 'time' in the case of the weights variable. (E.g. older errors weigh less). And also based on the 'direction' e.g. predicting to few used resources is worse than predicting a few resources used too much.

Usage

```
weighted_rmse(
  actual,
  predicted,
  weights = exp(-(length(actual):1)/14),
  worsenGoodExpectations = 1.5
)
```


Arguments

actual	Real Data, vector of observations
predicted	Predicted Data, vector of observations
weights	Time based decay. Default is an exponential decay
worsenGoodExpectations	Factor by how much predicting too few used ressources should be punished more.

Value

weighted RMSE

Index

* datasets

- dataCovidBeds20200624, [12](#)
 - dataICUBeds20200821, [13](#)
 - ex1InfectedDf, [17](#)
 - GABeds220200624, [23](#)
 - GermanCounties, [23](#)
 - GermanStates, [24](#)
 - icudata, [50](#)
 - koelnarchive, [51](#)
 - nrwarchive, [57](#)
 - obkarchive, [57](#)
 - paras, [58](#)
 - rkidata, [65](#)
 - synthpara, [73](#)
 - vaccineCounts, [77](#)
- aggMax, [4](#)
- aggregateSimulationReplications, [5](#)
- autoplot.BabsimDailyCases, [5](#)
- babsimHospital, [6](#), [7](#), [10](#), [15](#), [19](#), [21](#), [27](#), [29](#),
[61–63](#)
- babsimHospitalPara, [6](#), [7](#), [12](#), [21](#), [34](#), [42](#), [52](#),
[53](#), [55](#)
- babsimToolsConf, [10](#), [15](#), [27](#), [29](#), [63](#)
- checkSimPara, [12](#)
- data.table, [23](#), [24](#)
- dataCovidBeds20200624, [11](#), [12](#), [15](#), [27](#), [29](#),
[35](#), [55](#), [63](#)
- dataICUBeds20200821, [13](#)
- ensureRangeOpen, [14](#)
- envToTibble, [14](#)
- ex1InfectedDf, [17](#)
- extendRki, [18](#), [48](#), [49](#)
- extractSimulationResults, [5](#), [19](#)
- fetchRkiCases, [19](#)
- fitExponential, [20](#)
- funOptimizeSim, [21](#)
- funWrapOptimizeSim, [22](#)
- GABeds220200624, [11](#), [15](#), [23](#), [28](#), [29](#), [55](#), [63](#)
- GermanCounties, [20](#), [23](#), [24](#)
- GermanStates, [20](#), [24](#), [24](#)
- get_mon_resources, [14](#), [15](#), [28](#), [29](#), [61](#)
- getArrivalTimes, [25](#)
- getBestParameter, [25](#)
- getBounds, [26](#)
- getConfFromData, [27](#)
- getDailyMaxResults, [28](#)
- getDecision, [31](#)
- getDiviLinks, [31](#)
- getError, [32](#)
- getIcuBeds, [33](#), [39](#)
- getInfectedPerDay, [33](#)
- getMatrixD, [34](#)
- getMatrixP, [34](#), [74](#)
- getObkData, [35](#)
- getParameterDataFrame, [35](#)
- getParameterName, [21](#), [22](#), [36](#), [52](#), [53](#)
- getParameterNameList, [37](#)
- getParaSet, [37](#)
- getPeakVec, [38](#)
- getRealBeds, [14](#), [15](#), [28](#), [29](#), [38](#)
- getRegionIcu, [40](#)
- getRegionRki, [40](#)
- getRkiData, [18](#), [41](#), [48](#), [80](#)
- getRkiRisk, [41](#)
- getStartParameter, [42](#)
- getSyntheticData, [42](#)
- getTrainTestObjFun, [44](#)
- ggVisualizeIcu, [45](#)
- ggVisualizeRki, [46](#)
- ggVisualizeRkiAge, [47](#)
- ggVisualizeRkiEvents, [48](#)
- ggVisualizeRkiExtended, [48](#)
- ggVisualizeRkiExtendedDataCalculation,
[49](#)

icudata, [38](#), [45](#), [50](#), [78](#)

koelnarchive, [51](#)

mapAgeGroupToAge, [51](#)
mapAgeToAgeGroup, [52](#)
mapPToPara, [52](#)
mapXToPara, [26](#), [53](#)
mclapply, [10](#), [27](#), [63](#)
messageDateRange, [54](#)
messagef, [54](#)
modelResultHospital, [14](#), [16](#), [28](#), [30](#), [55](#)

nrwarchive, [57](#)

obkarchive, [57](#)

paras, [58](#)
plotDailyMaxResults, [59](#)
plotPostprocessedEnvs, [60](#), [62](#)
postprocessEnvs, [60](#), [61](#)
printConf, [62](#)

RiskScore, [64](#)
rkidata, [47](#), [49](#), [65](#), [73](#), [76](#), [79](#)
rkiGeschlechtToSex, [66](#)
rkiToBabsimArrivals, [25](#), [66](#)
rkiToBabsimData, [47](#), [67](#)
rtgamma, [68](#)
runoptDirect, [51](#), [57](#), [58](#), [68](#), [73](#)
runOptLocal, [70](#)

set.seed, [55](#)
simmer, [10](#), [14](#), [15](#), [27–29](#), [39](#), [61](#), [63](#)
skip_if_quicktest, [71](#)
smoothParameter, [72](#)
softmax, [72](#)
sprintf, [54](#)
str, [62](#)
switchRkiRefdatumToMeldedatum, [73](#)
synthpara, [73](#)

updateIcudataFile, [74](#)
updateMatrixP, [74](#)
updateParaSet, [75](#)
updateRkidataFile, [75](#)

vaccineCounts, [77](#)
visualizeGraph, [77](#)
visualizeIcu, [78](#)
visualizeRki, [79](#)
visualizeRkiEvents, [80](#)
weighted_rmse, [80](#)