Package 'genSEIR'

October 13, 2022

Type Package

Title Predict Epidemic Curves with Generalized SEIR Modeling

Version 0.1.1

Date 2021-07-12

Maintainer Selcuk Korkmaz <selcukorkmaz@gmail.com>

Depends R (>= 3.5.0)

Imports pracma, minpack.lm, nlsr, ggplot2

Description

Performs generalized Susceptible-Exposed-Infected-Recovered (SEIR) modeling to predict epidemic curves. The method is described in Peng et al. (2020) <doi:10.1101/2020.02.16.20023465>.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Selcuk Korkmaz [aut, cre] (<https://orcid.org/0000-0003-4632-6850>)

Repository CRAN

Date/Publication 2021-07-12 14:20:02 UTC

R topics documented:

`checkRates` *Check Rates*

Description

This function compares the fitted and calculated death and recovered ratios. The idea is to check whether the approximation of these ratios is appropriate.

Usage

```
checkRates(time, Q, R, D, kappaFun, lambdaFun, kappa, lambda, dt = 1)
```

Arguments

`time`	time vector
`Q`	time histories of the quarantined/active cases
`R`	time histories of the recovered cases
`D`	time histories of the deceased cases
`kappaFun`	anonymous function approximating the death rate
`lambdaFun`	anonymous function approximating the recovery rate
`kappa`	mortality rate
`lambda`	cure rate
`dt`	a time step, default is 1/24. This oversample time to ensure that the algorithm converges.

Value

plots for death rate and recovery rate

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

`SEIQRDP` `fit_SEIQRDP`

`fit_SEIQRDP` *Fit SEIQRDP function*

Description

Fit SEIQRDP function parameters used in the SEIQRDP function, used to model the time-evolution of an epidemic outbreak.

Usage

```
fit_SEIQRDP(
  Q,
  R,
  D,
  Npop,
  E0,
  I0,
  time,
  alpha = 0.05,
  dt = 1/24,
  guess,
  ftol = sqrt(.Machine$double.eps),
  ptol = sqrt(.Machine$double.eps),
  gtol = 0,
  diag = list(),
  epsfcn = 0,
  factor = 100,
  maxfev = integer(),
  maxiter = 1000,
  nprint = 1,
  trace = TRUE,
  ...
)
```

Arguments

`Q`	time histories of the active cases
`R`	time histories of the recovered cases
`D`	time histories of the deceased cases
`Npop`	total population of the country
`E0`	initial number of exposed cases
`I0`	initial number of predicted infectious cases
`time`	a time vector
`alpha`	type I error rate, default is 0.05
`dt`	the time step. This oversamples time to ensure that the algorithm converges

guess	initial guess parameters
ftol	nls.lm.control object. non-negative numeric. Default is 1e-6
ptol	nls.lm.control object. non-negative numeric. Default is 1e-6
gtol	nls.lm.control object. non-negative numeric. Default is 1e-6
diag	nls.lm.control object. a list or numeric vector containing positive entries that serve as multiplicative scale factors for the parameters.
epsfcn	nls.lm.control object. Default is 0.001
factor	nls.lm.control object. Default is 100
maxfev	nls.lm.control object. Default is 1000
maxiter	nls.lm.control object. Default is 100
nprint	nls.lm.control object. Default is 1
trace	set TRUE to trace iteration results
...	further arguments

Value

a list of optimized parameters

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP predict_SEIQRDP

Examples

```
start = "01/01/21"
finish = "04/01/21"
country = "Italy"
dt = 1
f=30

covidData = getDataCOVID(start = start, finish = finish, country = country)
Recovered = covidData$tableRecovered
Deaths = covidData$tableDeaths
Confirmed = covidData$tableConfirmed

if(nrow(Recovered) == 1){
```

```
  name = Recovered$CountryRegion
}else{
   name = paste0(Recovered$ProvinceState, " (",Recovered$CountryRegion,")")
}

  recovered = Recovered[ ,5:ncol(covidData$tableRecovered)]
  deaths = Deaths[ ,5:ncol(covidData$tableDeaths)]
  confirmed = Confirmed[ ,5:ncol(covidData$tableConfirmed)]

  Npop = 60000000

  alpha_guess = 0.05
  beta_guess = 0.8
  LT_guess = 7
  Q_guess = 0.8
  lambda_guess = c(0.01,0.001,10)
  kappa_guess = c(0.001,0.001,10)

guess = list(alpha_guess,
             beta_guess,
             1/LT_guess,
             Q_guess,
             lambda_guess[1],
             lambda_guess[2],
             lambda_guess[3],
             kappa_guess[1],
             kappa_guess[2],
             kappa_guess[3])

 Q0 = confirmed[1]-recovered[1]-deaths[1]
 I0 = 0.3*Q0
 E0 = 0.3*Q0
 R0 = recovered[1]
 D0 = deaths[1]

 Active = confirmed-recovered-deaths
 Active[Active<0] <- 0

 Q=Active
 R=recovered
 D = deaths

 time = seq(as.Date(start, format = "%m/%d/%y"), as.Date(finish, format = "%m/%d/%y"), by = "1 day")

 params = fit_SEIQRDP(Q = Active, R = recovered, D = deaths, Npop = Npop, E0 = E0, I0 = I0,
                         time = time, alpha = 0.05, dt = dt, guess = guess, ftol = 1e-6,
                     ptol = 1e-6, gtol = 1e-6, epsfcn = 0.001, factor = 100, maxfev = 1000,
                         maxiter = 100, nprint = 1, trace = TRUE)
```

---

getA *Compute the matrix A*

Description

This function computes the matrix A that is found in: dY/dt = A*Y + F

Usage

```
getA(alpha, gamma, delta, lambda, kappa)
```

Arguments

alpha	protection rate
gamma	inverse of the average latent time
delta	rate of people entering in quarantine
lambda	cure rate
kappa	mortality rate

Value

The matrix A that is found in: dY/dt = A*Y + F

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

---

getDataCOVID *Get COVID-19 Data*

Description

The function collects the updated COVID-19 data from the John Hopkins University.

Usage

```
getDataCOVID(country, start = NULL, finish = NULL)
```

Arguments

country	name of the country. It should be a character string.
start	a start date in mm/dd/yy format. Start date can not be earlier than 01/22/20. Start date can not be later than finish date. If start date is NULL then start date will be 01/22/20.
finish	a finish date in mm/dd/yy format. Finish date can not be earlier than start date. If finish date is NULL then finish date will be the latest date at John-Hopkins CSSE system.

Value

a list of COVID-19 historical data including confirmed, death and recovered cases in desired time ranges.

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

Examples

```
covidData = getDataCOVID(country = "Italy",
                         start = "05/01/20",
                         finish = "12/31/20")
recovered = covidData$tableRecovered
deaths = covidData$tableDeaths
confirmed = covidData$tableConfirmed
```

---

getKappaFun *Estimate Death Rate*

Description

This function provides a first estimate of the death rate, to faciliate convergence of the main algorithm.

Usage

```
getKappaFun(
  tTarget,
  Q,
  D,
  guess,
  ftol,
  ptol,
  gtol,
  epsfcn,
  factor,
  maxfev,
  maxiter,
  nprint,
  trace
)
```

Arguments

tTarget	time vector
Q	target time-histories of the quarantined cases
D	target time-histories of the dead cases
guess	initial guess parameters for kappa
ftol	nls.lm.control object. non-negative numeric. Default is 1e-6
ptol	nls.lm.control object. non-negative numeric. Default is 1e-6
gtol	nls.lm.control object. non-negative numeric. Default is 1e-6
epsfcn	nls.lm.control object. Default is 0.001
factor	nls.lm.control object. Default is 100
maxfev	nls.lm.control object. Default is 1000
maxiter	nls.lm.control object. Default is 100
nprint	nls.lm.control object. Default is 1
trace	Set TRUE to trace iteration results

Value

vector of estimation and optimization function for the death rate

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

---

getLambdaFun *Estimate Recovery Rate*

Description

This function provides a first estimate of the recovery rate, to faciliate convergence of the main algorithm.

Usage

```
getLambdaFun(
  tTarget,
  Q,
  R,
  guess,
  ftol,
  ptol,
  gtol,
  epsfcn,
  factor,
  maxfev,
  maxiter,
  nprint,
  trace
)
```

Arguments

tTarget	target time vector
Q	target time-histories of the quarantined cases
R	target time-histories of the recovered cases
guess	initial guess parameters for kappa
ftol	nls.lm.control object. non-negative numeric. Default is 1e-6
ptol	nls.lm.control object. non-negative numeric. Default is 1e-6
gtol	nls.lm.control object. non-negative numeric. Default is 1e-6
epsfcn	nls.lm.control object. Default is 0.001
factor	nls.lm.control object. Default is 100
maxfev	nls.lm.control object. Default is 1000
maxiter	nls.lm.control object. Default is 100
nprint	nls.lm.control object. Default is 1
trace	set TRUE to trace iteration results

Value

vector of estimation and optimization function for the recovery rate

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

---

kappaFun *Anonymous function approximating the death rate*

Description

Anonymous function approximating the death rate

Usage

```
kappaFun(a, t)
```

Arguments

a	parameter vector
t	time vector

Value

No return value, called for side effects

lambdaFun *Anonymous function approximating the recovery rate*

Description

Anonymous function approximating the recovery rate

Usage

```
lambdaFun(a, t)
```

Arguments

a	parameter vector
t	time vector

Value

No return value, called for side effects

modelFun *Model function*

Description

Model function

Usage

```
modelFun(Y, A, K)
```

Arguments

Y	time vector
A	the matrix A that is found in: dY/dt = A*Y + F
K	the zero matrix for the seven states

Value

No return value, called for side effects

plot_SEIQRDP *Plots for Epidemic Curves*

Description

This function creates plots for reported and predicted active, recovered and death cases.

Usage

```
plot_SEIQRDP(
  object,
  reported = TRUE,
  sep = FALSE,
  show = c("S", "E", "I", "Q", "R", "D", "P"),
  ci = FALSE,
  title = NULL,
  checkRates = FALSE,
  ...
)
```

Arguments

object	a predict_SEIQRDP result.
reported	a logical argument. If TRUE reported official cases will be added to the plot.
sep	a logical argument. If TRUE seperate plots will be plotted. If FALSE one plot with all desired states will be plotted.
show	select one or more desired state. S: Susceptible, E: Exposed, I: Infectious, Q: Quarantined, R: Recovered, D: Dead, P: Insusceptible
ci	a logical argument. If TRUE a bootstrap confidence intetval will be added to the plot.
title	an optional title for the plot.
checkRates	if TRUE compares the fitted and calcualted death and recovered ratios through plots
...	other plot options

Value

plots for epidemic curves: active cases, recovered and deaths

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

See Also

SEIQRDP fit_SEIQRDP

Examples

```
alpha_guess = 0.45
beta_guess = 1
LT_guess = 2
Q_guess = 0.55
lambda_guess = c(0.01,0.01,30)
kappa_guess = c(0.01,0.001,30)

guess = list(alpha_guess,
             beta_guess,
             1/LT_guess,
             Q_guess,
             lambda_guess[1],
             lambda_guess[2],
             lambda_guess[3],
             kappa_guess[1],
             kappa_guess[2],
             kappa_guess[3])

pred = predict_SEIQRDP(country = "Germany", start = "10/15/20", finish = "12/15/20",
                       dt = 1, f = 30, conf = 0.95, Npop = 80000000, guess, boot = TRUE,
                        seed = 123, repeatNumber = 10, bootSample = NULL, type = "norm")


plot_SEIQRDP(object = pred, sep = FALSE, ci = TRUE, show = c("Q", "R", "D"), checkRates = TRUE)
```

`predict_SEIQRDP` *Predict cases using generalized SEIR model*

Description

This function predicts cases of an outbreak using a generalized SEIR model

Usage

```
predict_SEIQRDP(
  country,
  start,
  finish,
  Npop = NULL,
```

```
  guess,
  dt = 1,
  f = 0,
  boot = FALSE,
  conf = 0.95,
  seed = 123,
  repeatNumber = 200,
  bootSample = NULL,
  type = "norm"
)
```

Arguments

country	name of the country. It should be a character string.
start	a start date in mm/dd/yy format. Start date can not be earlier than 01/22/20. Start date can not be later than finish date. If start date is NULL then start date will be 01/22/20.
finish	a finish date in mm/dd/yy format. Finish date can not be earlier than start date. If finish date is NULL then finish date will be the latest date at John-Hopkins CSSE system.
Npop	total population of the country
guess	initial guess parameters
dt	the time step. This oversamples time to ensure that the algorithm converges
f	number of days for future predictions
boot	if TRUE bootstrap will be performed to calculate confidence interval
conf	confidence level, default is 0.95.
seed	set a seed for reproducible results.
repeatNumber	number of iteration for bootstrap.
bootSample	number of sample for each bootstrap. if NULL then the number of sample is 80 percent of the original data.
type	a condidence interval type. If "norm" it calculates based on normal approximation, if "perc" it calculates based on percentile approximation,

Value

a list of predicted and actual cases.

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

Examples

```
alpha_guess = 0.45
beta_guess = 1
LT_guess = 2
Q_guess = 0.55
lambda_guess = c(0.01,0.01,30)
kappa_guess = c(0.01,0.001,30)

guess = list(alpha_guess,
             beta_guess,
             1/LT_guess,
             Q_guess,
             lambda_guess[1],
             lambda_guess[2],
             lambda_guess[3],
             kappa_guess[1],
             kappa_guess[2],
             kappa_guess[3])


pred = predict_SEIQRDP(country = "Germany", start = "10/15/20", finish = "12/15/20",
dt = 1, f = 30, conf = 0.95, Npop = 80000000, guess, boot = FALSE,
seed = 123, repeatNumber = 100, bootSample = NULL, type = "norm")

predict = pred$pred
actual = pred$actual
```

RK4 *Runge-Kutta 4th Order Method to Solve Differential Equation*

Description

Runge-Kutta 4th Order Method to Solve Differential Equation

Usage

```
RK4(Y, A, K, dt)
```

Arguments

Y	initial values for seven states
A	the matrix A that is found in: dY/dt = A*Y + F
K	the zero matrix for the seven states
dt	the time step. This oversamples time to ensure that the algorithm converges

Value

ordinary differential equation result for the seven states

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

SEIQRDP fit_SEIQRDP

SEIQRDP *Simulate generalized SEIR model*

Description

This function simulates the time-histories of an epidemic outbreak using a generalized SEIR model

Usage

```
SEIQRDP(
  alpha,
  beta,
  gamma,
  delta,
  lambda0,
  kappa0,
  Npop,
  E0,
  I0,
  Q0,
  R0,
  D0,
  lambdaFun,
  kappaFun,
  tstart,
  tfinish,
  dt = 1/24,
  f = 0
)
```

Arguments

alpha	fitted protection rate
beta	fitted infection rate
gamma	fitted Inverse of the average latent time
delta	fitted rate at which people enter in quarantine
lambda0	fitted cure rate
kappa0	fitted mortality rate
Npop	Total population of the sample
E0	Initial number of exposed cases
I0	Initial number of infectious cases
Q0	Initial number of quarantined cases
R0	Initial number of recovered cases
D0	Initial number of dead cases
lambdaFun	anonymous function giving the time-dependant recovery rate
kappaFun	anonymous function giving the time-dependant death rate
tstart	start date
tfinish	finish date
dt	the time step. This oversamples time to ensure that the algorithm converges
f	number of days for future predictions

Value

a list of predicted cases including susceptible, exposed, infectious, quarantined, recovered, dead and insusceptible.

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

fit_SEIQRDP

Examples

```
start = "01/01/21"
finish = "04/01/21"
country = "Italy"
dt = 1
f=30

covidData = getDataCOVID(start = start, finish = finish, country = country)
Recovered = covidData$tableRecovered
Deaths = covidData$tableDeaths
Confirmed = covidData$tableConfirmed

if(nrow(Recovered) == 1){
  name = Recovered$CountryRegion
}else{
   name = paste0(Recovered$ProvinceState, " (",Recovered$CountryRegion,")")
}

  recovered = Recovered[ ,5:ncol(covidData$tableRecovered)]
  deaths = Deaths[ ,5:ncol(covidData$tableDeaths)]
  confirmed = Confirmed[ ,5:ncol(covidData$tableConfirmed)]

  Npop = 60000000

  alpha_guess = 0.05
  beta_guess = 0.8
  LT_guess = 7
  Q_guess = 0.8
  lambda_guess = c(0.01,0.001,10)
  kappa_guess = c(0.001,0.001,10)

guess = list(alpha_guess,
             beta_guess,
             1/LT_guess,
             Q_guess,
             lambda_guess[1],
             lambda_guess[2],
             lambda_guess[3],
             kappa_guess[1],
             kappa_guess[2],
             kappa_guess[3])

 Q0 = confirmed[1]-recovered[1]-deaths[1]
 I0 = 0.3*Q0
 E0 = 0.3*Q0
 R0 = recovered[1]
 D0 = deaths[1]

 Active = confirmed-recovered-deaths
 Active[Active<0] <- 0
```

```
Q=Active
R=recovered
D = deaths

time = seq(as.Date(start, format = "%m/%d/%y"), as.Date(finish, format = "%m/%d/%y"), by = "1 day")

params = fit_SEIQRDP(Q = Active, R = recovered, D = deaths, Npop = Npop, E0 = E0, I0 = I0,
                 time = time, dt = dt, guess = guess, ftol = 1e-6, ptol = 1e-6, gtol = 1e-6,
                   epsfcn = 0.001, factor = 100, maxfev = 1000,maxiter = 100, nprint = 1,
                       trace = TRUE)

res = SEIQRDP(alpha = params$alpha1, beta = params$beta1,
              gamma = params$gamma1, delta = params$delta1,
              lambda0 = c(params$lambda01, params$lambda02, params$lambda03),
              kappa0 = c(params$kappa01, params$kappa02, params$kappa03),
              Npop, E0, I0, Q0, R0, D0,lambdaFun = params$lambdaFun,
              kappaFun = params$kappaFun, tstart = start, tfinish = finish,
              dt = dt, f =f)
```

SEIQRDP_for_fitting *Fitted Results for SEIQRDP*

Description

Fitted Results for SEIQRDP

Usage

```
SEIQRDP_for_fitting(par, t, t0, Npop, E0, I0, Q, R, D, dt)
```

Arguments

par	initial guess parameters
t	historical time vector
t0	target time vector
Npop	total population of the country
E0	initial number of exposed cases
I0	initial number of infectious cases
Q	actual number of quarantined cases
R	actual number of recovered cases
D	actual number of dead cases
dt	the time step. This oversamples time to ensure that the algorithm converges

Value

a data frame for fitted quarantined, recovered and deaths

Author(s)

Selcuk Korkmaz, <selcukorkmaz@gmail.com>

References

Peng, L., Yang, W., Zhang, D., Zhuge, C., Hong, L. 2020. "Epidemic analysis of COVID-19 in China by dynamical modeling", arXiv preprint arXiv:2002.06563.

https://www.mathworks.com/matlabcentral/fileexchange/74545-generalized-seir-epidemic-model-fitting-

See Also

fit_SEIQRDP RK4

Index