Package ‘HMMRel’

November 15, 2024

Title Hidden Markov Models for Reliability and Maintenance

Version 0.1.0

Author M.L. Gamiz [aut, cre, cph],
N. Limnios [aut, cph],
M.C. Segovia-Garcia [aut, cph]

Maintainer M.L. Gamiz <mgamiz@ugr.es>

Description Reliability Analysis and Maintenance Optimization using Hidden Markov Models (HMM). The use of HMMs to model the state of a system which is not directly observable and instead certain indicators (signals) of the true situation are provided via a control system. A hidden model can provide key information about the system dependability, such as the reliability of the system and related measures. An estimation procedure is implemented based on the Baum-Welch algorithm. Classical structures such as K-out-of-N systems and Shock models are illustrated. Finally, the maintenance of the system is considered in the HMM context and two functions for new preventive maintenance strategies are considered. Maintenance efficiency is measured in terms of expected cost. Maintenance efficiency is measured in terms of expected cost. Methods are described in Gamiz, Limnios, and Segovia-Garcia (2023) <doi:10.1016/j.ejor.2022.05.006>.

License GPL-2

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-11-15 09:00:17 UTC

Contents

`HMMRel-package` *HMMRel - Hidden Markov Models for Reliablity and Maintenance*

Description

Reliability Analysis and Maintenance Optimization using Hidden Markov Models (HMM).

Details

Package:	HMMRel
Type:	Package
Version:	1.0
Date:	2024-11-11
License:	GPL version 2 or later
Maintainer:	M.L. Gamiz <mgamiz@ugr.es>
URL:	`http://wpd.ugr.es/~reliastat`

Author(s)

M.L. Gamiz, N.Limnios, and M.C. Segovia-Garcia

References

Gamiz, M.L., Limnios, N., Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

`cost.cspc` *Maintenance Policy based on Critical State Probability Critera.*

Description

Preventive maintenance based on Critical State Probability Criteria (CSPC).

Usage

```
cost.cspc(prob,hmmR,n.up1,cost.C,cost.P,t.max)
```

Arguments

`prob`	A real number in the interval (0,1).
`hmmR`	A hidden Markov Model.
`n.up1`	An integer value for indicating the total number of optimal performance states of the hidden MC.
`cost.C`	A positive real number denoting the cost value in monetary units incurred by a corrective maintenance action.
`cost.P`	A positive real number denoting the cost value in monetary units incurred by a preventive maintenance action.
`t.max`	A time value for the maximum time the system will be in use. After that time the system will not operate anymore.

Details

Preventive maintenance policies based on critical states probability critera (CSPC) is considered. Roughly speaking, a preventive maintenance action is carried out once the system enters a subset of operational states that are considered critical in some sense. The subset of operative states `up` is in turn split into two subsets: optimal states or `up1` and operative but critical states or `up2`, where `up=up1` $\cup$ `up2`. For a given probability value (`prob`) this function first calculates the optimal inspection time

$$\texttt{t.insp} = \min\{t > 0 : \Pr(X(t) \in \texttt{up2}, X(u) \in \texttt{up1}, \forall u \leq t)\}.$$

The system is inspected every `t.insp` time-units. At the time of inspection, any of three situations can be found:

1. the system is in failure, then the system is returned to operational conditions (`up1`), and a cost of `cost.C` monetary-units is implied;
2. the system is in a state of `up2`, then a preventive maintenance action is carried out, returning the system to a state in `up1` and implying a cost of `cost.P` monetary-units; and
3. the system is found in a state of `up1`, then no maintenance action is carried out and there is no associated cost.

Value

`time.insp`	The time at which preventive maintenance is carried out.
`t.max`	The maximum time that the system is being used.
`n.insp`	The total number of inspections that are carried out during the system lifetime, i.e. in the interval (0, `t.max`).
`total.cost`	The total cost incurred by all maintenance actions (corrective and preventive) developed in the system.

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M.L., Limnios, N., and Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See `cost.wspc` for the implementation of the WSPC algorithm for maintenance policy.

Examples

```
model<-'other'
rate<-p<-NA
P<-matrix(c(8,2,1,0,6,4,6,2,2)/10,3,3,byrow=TRUE)
M<-matrix(c(7,3,0,4,3,3,0,4,6)/10,3,3,byrow=TRUE)
Nx<-3; Ny<-3
n.up<-2; n.green<-2
alpha<-c(1,0,0)
hmm1<-def.hmmR(model=model,rate=NA,p=NA,alpha=alpha,P=P,M=M,Nx=Nx,Ny=Ny,n.up=n.up,n.green=n.green)
prob<-0.8;
n.up1<-n.green1<-1;cost.C<-10;cost.P<-1;t.max<-50
cost1<-cost.cspc(prob=prob,hmmR=hmm1,n.up1=n.up1,cost.C=cost.C,cost.P=cost.P,t.max=t.max)
cost1
#
v.prob<-seq(0.1,0.99,length=100)
v.cost1<-inspection.time<-double(100)
for(i in 1:100)
{cost<-cost.cspc(prob=v.prob[i],hmmR=hmm1,n.up1=n.up1,
   cost.C=cost.C,cost.P=cost.P,t.max=t.max)
v.cost1[i]<-cost$total.cost
inspection.time[i]<-cost$time.insp
}
oldpar <- par(mar = c(5, 5, 10, 10))
plot(v.prob,v.cost1,type='s',main='CSPC Algorithm for Maintenance Policy',
xlab='Probability of critical state',
ylab='Cost of maintenance')
grid()
par(oldpar)
```

`cost.wcspc` *Maintenance Policy based on Warning Signal Probability Criteria.*

Description

Preventive maintenance based on Warning Signal Probability Criteria (CSPC).

Usage

```
cost.wspc(prob,hmmR,n.up1,n.green1,cost.C,cost.P,t.max)
```

Arguments

`prob`	A real number in the interval (0,1).
`hmmR`	A hidden Markov Model.
`n.up1`	An integer value for indicating the total number of optimal performance states of the hidden MC.
`n.green1`	An integer value for indicating the total number of safe signals. A safe signal indicates an optimal system performance.
`cost.C`	A positive real number denoting the cost value in monetary units incurred by a corrective maintenance action.
`cost.P`	A positive real number denoting the cost value in monetary units incurred by a preventive maintenance action.
`t.max`	A time value for the maximum time the system will be in use. After that time the system will not operate anymore.

Details

Preventive maintenance policies based on Warning Signal Probability criteria (CSPC) is considered. Roughly speaking, a preventive maintenance action is carried out at a time when the probability that system is not occupying an optimal operational state or a warning signal is received is above a prespecified value `prob`. The subset of operative states `up` is in turn split into two subsets: optimal states or `up1` and operative but critical states or `up2`, where `up=up1`∪`up2`. Similarly, the set of `green` signals is split into two subsets: `safe` signals and `warning` signals. `n.green1` is the size of subset `safe`. For a given probability value (`prob`) this function first calculates the optimal inspection time

$$\texttt{t.insp} = \min\{t > 0 : \Pr(Y(t) \in \text{warning}, Y(u) \in \text{safe}, \forall u \leq t)\}.$$

The system is inspected every `t.insp` units.of time. At the time of inspection, any of three situations can be found:

1. the system is in failure, then the system is returned to operational conditions (`up1`), and a cost of `cost.C` monetary-units is implied;
2. the system is in a state of `up2`, then a preventive maintenance action is carried out, returning the system to a state in `up1` and implying a cost of `cost.P` monetary-units; and
3. the system is found in a state of `up1`, then no maintenance action is carried out and there is no associated cost.

Value

`time.insp`	The time at which preventive maintenance is carried out.
`t.max`	The maximum time that the system is being used.
`n.insp`	The total number of inspections that are carried out during the system lifetime, i.e. in the interval (0, `t.max`).
`total.cost`	The total cost incurred by all maintenance actions (corrective and preventive) developed in the system.

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M.L., Limnios, N., and Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See `cost.cspc` for the implementation of the CSPC algorithm for maintenance policy.

Examples

```
model<-'other'
rate<-p<-NA
P<-matrix(c(8,2,1,0,6,4,6,2,2)/10,3,3,byrow=TRUE)
M<-matrix(c(7,3,0,4,3,3,0,4,6)/10,3,3,byrow=TRUE)
Nx<-3; Ny<-3
n.up<-2; n.green<-2
alpha<-c(1,0,0)
hmm1<-def.hmmR(model=model,rate=NA,p=NA,alpha=alpha,P=P,M=M,Nx=Nx,Ny=Ny,n.up=n.up,n.green=n.green)
prob<-0.8;
n.up1<-n.green1<-1;cost.C<-10;cost.P<-5;t.max<-50
cost1<-cost.wspc(prob=prob,hmmR=hmm1,n.up1=n.up1,n.green1=n.green1,
         cost.C=cost.C,cost.P=cost.P,t.max=t.max)
cost1
#
v.prob<-seq(0.1,0.99,length=100)
v.cost1<-inspection.time<-double(100)
for(i in 1:100)
{cost<-cost.wspc(prob=v.prob[i],hmmR=hmm1,n.up1=n.up1,n.green1=n.green1,
cost.C=cost.C,cost.P=cost.P,t.max=t.max)
v.cost1[i]<-cost$total.cost
#inspection.time[i]<-cost$time.insp
}
oldpar<-par(mar=c(5,5,10,10))
plot(v.prob,v.cost1,type='s',main='WSPC Algorithm for Maintenance Policy',
      xlab='Probability of critical state',
     ylab='Cost of maintenance')
grid()
par(oldpar)
```

`def.hmmR` *Define a HMM object for Reliability Analysis.*

Description

This function creates a list with all the elements that describe a HMM in the context of Reliability and Maintenance.

Usage

```
def.hmmR(model,rate,p,alpha,P,M,Nx,Ny,n.up,n.green)
```

Arguments

`model`	A character string to choose which HMM model is considered. Possible values are "KooN", "shock", "other"
`rate`	A positive real number indicating the failure rate of one unit of the system.
`p`	A real number in the interval (0,1) indicating the probability that the system receives one shock during two consecutive inspections.
`alpha`	A vector of size `Nx` with the initial law of the hidden Markov chain.
`P`	A square matrix of dimension `Nx` with the transition probabilities between the hidden states.
`M`	A matrix of dimension `Nx` $\times$ `Ny` with the emission probabilities.
`Nx`	An integer indicating the total number of states in the system. By default the states are labelled: 1,...,`Nx`.
`Ny`	An integer indicating the total number of signals received. By default the signals are labelled: 1,...,`Ny`.
`n.up`	An integer lower than `Nx` indicating the number of (hidden) operative states in the system. The first `n.up` values in the state set denote operative states.
`n.green`	An integer lower then `Ny` indicating the number of signals of good performance. The first `n.green` signals are read as good performance of the system.

Details

- When `model="KooN"` the argument `Nx` is the maximum number of units in the system. There must be `K=n.up` operative units for the system to function. If `K=`1 a parallel system is built. If `K=`N a series system is built.
- When `model="shock"` the argument `Nx` is the maximum number of shocks that the system can accumulate before breakdown.

Value

A list with the elements of the HMM.

- `states` A set of `Nx` characters or integers decribing the hidden states of the system.
- `signals` A set of `Ny` characters or integers decribing the possible signals observed.
- `P` A square matrix with `Nx` rows with the transition probabilities between the hidden states.
- `M` A matrix of dimension `Nx` $\times$ `Ny` with the emission probabilities.

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M.L., Limnios, N., and Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See also `sim.hmmR` to simulate data from a given HMM.

Examples

```
## Define a HMM object describing a repairable system
## The system can be in one of 3 states: 2 up states and 1 down state.
## 3 different signals can be received: 2 good performance signals (green)
##  and 1 signal of failure (red)
P<-matrix(c(8,2,1,0,6,4,6,2,2)/10,3,3,byrow=TRUE)
M<-matrix(c(7,3,0,4,3,3,0,4,6)/10,3,3,byrow=TRUE)
Nx<-3; Ny<-3
n.up<-2; n.green<-2
alpha<-c(1,0,0)
hmm1<-def.hmmR(model='other',rate=NA,p=NA,alpha=alpha,P=P,M=M,Nx=Nx,Ny=Ny,
               n.up=n.up,n.green=n.green)
hmm1
```

---

fit.hmmR *Non-parametric fitting of a HMM using the Baum-Welch algorithm.*

Description

This function adapts the EM algorithm to fit the transition matrix of the hidden Markov chain as well as the emission probability matrix of a HMM.

Usage

```
fit.hmmR(Y,P0,M0,alpha0,max.iter=50,epsilon=1e-9,Nx,Ny)
```

Arguments

- `Y` — A sequence of observations consisting of signals of system performance.
- `P0` — A square matrix of dimension `Nx` with the transition probabilities of the hidden MC at the first iteration of the algorithm.
- `M0` — A matrix of dimension `Nx` × `Ny` with the emission probabilities at the first step of the algorithm.
- `alpha0` — A vector of size `Nx` with the initial distribution of the hidden Markov chain at the first iteration of the algorithm.
- `max.iter` — A integer value with the maximum number of iterations in the iterative algorithm. Default value is `max.ite`=50.
- `epsilon` — A numeric value with the tolerance in the iterative algorithm. Default value is `epsilon`=1e-9.
- `Nx` — An integer value with the maximum number of states of the hidden Markov chain.
- `Ny` — An integer value with the maximum number of signals emitted by the system.

Details

- The argument `alpha0` representing the initial distribution of the hidden MC is fixed, and defined by the user. The argument `Nx` is the size of the state space of the hidden MC. As default, the set of numbers 1,...,`Nx` is the state space of the hidden MC.
- `Ny` is the size of the alphabet of signals emitted. As default, the set of numbers 1,...,`Ny` is the signal-alphabet.
- The successive iterations of the algorithm can be traced and information is accessible from the outcome of this function.

Value

Among other information, this function provides the following values:

- `P` — A square matrix with `Nx` rows with the estimated transition probabilities.
- `M` — A matrix of `Nx` rows and `Ny` columns, with the estimated emission probabilities.
- `n.iter` — An integer indicating the number of iterations performed in the algorithm.
- `tol` — A numeric value with the achieved tolerance value in the algorithm.
- `Fm` — A matrix of dimension `n` × `Nx` with the estimated forward probability values at the last iteration of the algorithm.
- `Bm` — A matrix of dimension `Nx` × `n` with the estimated forward probability values at the last iteration of the algorithm.
- `AIC` — The estimated value of the Akaike statistics. The number of parameters to be estimated is `nparam`=`Nx`*(`Nx`-1)+`Nx`*(`Ny`-1).

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M.L., Limnios, N., and Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See `def.hmmR` to define an object HMM, and `sim.hmmR` to simulate a random path from a given HMM object.

Examples

```
model<-'other'
rate<-NA
p<-NA
P<-matrix(c(0.7,0.3,1,0),2,2,byrow=TRUE)
M<-matrix(c(0.6,0.4,0,0,0,1),2,3,byrow=TRUE)
alpha<-c(1,0)
Nx<-2
Ny<-3
n.up<-1
n.green<-2
hmm0<-def.hmmR(model,rate,p,alpha,P,M,Nx,Ny,n.up,n.green)
set.seed(1969)
datos<-sim.hmmR(hmmR=hmm0,n=10)
estim<-fit.hmmR(Y=datos$Yn,P0=P,M0=M,alpha0=alpha,max.iter=50,epsilon=1e-9,Ny=3,Nx=2)
estim$P;P
estim$M;M
```

`Rcalc.hmmR` *Calculate the reliability of a system based on HMM.*

Description

For a given time `t` this function returns the value of the probability that the system does not fail in the interval `(0,t]`. It gives the probability that the system survives and is still working beyond time `t`.

Usage

```
Rcalc.hmmR(hmmR,t)
```

Arguments

`hmmR`	A Hidden Markov Model.
`t`	A value of time, it must be an integer equal or greater than 0.

Details

The state space is split into two subsets, i.e. `states=up` ∪ `down`. The subset up contains the states of good functioning, while the subset down contains the failure states. The signals aphabet is split into two subsets, i.e. `signals= green` ∪ `red`. A `green`-signal indicates good performance of the system, while a `red`-signal alerts of something wrong in the system. This function returns the probability that the system has not entered the set of down states or any signal from the `red` subset of signals has been emitted at any time before `t`.

Value

This function returns the probability that the system is working through a state in the up subset, and a `green` signal is being received. If `t`=0, then the returned value is 1.

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M. L., Limnios, N., and Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See `def.hmmR` to define a HMM object.

Examples

```
model<-'other'
rate<-NA
p<-NA
P<-matrix(c(0.7,0.3,1,0),2,2,byrow=TRUE)
M<-matrix(c(0.6,0.4,0,0,0,1),2,3,byrow=TRUE)
alpha<-c(1,0)
Nx<-2
Ny<-3
n.up<-1
n.green<-2
hmm0<-def.hmmR(model=model,rate=NA,p=NA,alpha=alpha,P=P,M=M,Nx=Nx,Ny=Ny,n.up=n.up,n.green=n.green)
times<-0:30
Rt<-Rcalc.hmmR(hmmR=hmm0,t=times)
oldpar <- par(mar = c(5, 5, 10, 10))
plot(times,Rt,type='s',ylim=c(0,1),ylab='',xlab='time',main='Reliability based on HMM')
grid()
par(oldpar)
```

`sim.hmmR` *Simulate sequence of states and signals of functioning of a system modelled by a HMM.*

Description

This function simulates a sample path from a 2-dimensional HMM. It returns the hidden sequence of states and signals. At each time, the hidden state of the system is simulated from the HMM as well as the associated signal that informs on the system performance at that time.

Usage

```
sim.hmmR(hmmR,n)
```

Arguments

`hmmR`	A Hidden Markov Model.
`n`	An integer number indicating the length of the sequence of states and signals to be simulated.

Value

The function `sim.hmmR` returns a list with the following information:

`Xn`	The sequence of simulated hidden states.
`Yn`	The sequence of observed signals.
`P`	The transition probability matrix of the hidden Markov chain (MC).
`alpha`	The initial distribution of the hidden MC.
`M`	The emission probability matrix.
`states`	The set of hidden states of the system.
`signal`	The alphabet corresponding to the observations.

Author(s)

M.L. Gamiz, N. Limnios, and M.C. Segovia-Garcia (2024)

References

Gamiz, M.L., Limnios, N., Segovia-Garcia, M.C. (2023). Hidden Markov models in reliability and maintenance. European Journal of Operational Research, 304(3), 1242-1255.

See Also

See `def.hmmR` to define a HMM object.

Examples

```
## Define a HMM object describing a repairable system
P<-matrix(c(8,2,1,0,6,4,6,2,2)/10,3,3,byrow=TRUE)
M<-matrix(c(7,3,0,4,3,3,0,4,6)/10,3,3,byrow=TRUE)
hmm1<-def.hmmR(model='other',rate=NA,p=NA,alpha=c(1,0,0),P=P,M=M,Nx=3,Ny=3,n.up=2,n.green=2)
sim.hmmR(hmmR=hmm1,n=20)
```

`Virkler25` *Fatigue crack growth in materials: Virkler dataset (tests 1 to 25)*

Description

The data consist of an aluminum alloy specimen that was tested to investigate fatigue crack propagation. Starting from an initial crack of length 9 mm for a particular item in test, the number of cycles for the size of the crack to reach a predetermined value was recorded successively. That is, it is registered the number of cycles every time an increment of size 0.2 mm in length occurs. The experiment finishes once a critical size of the crack is reached, meaning the failure of the item. The data were first published in Virkler et al. (1979) where there were 68 specimens tested to grow the initial crack of 9 mm to the final crack of 50 mm. The first 25 tests are included.

Format

A data frame with 26 variables:

CrackLength Length of the crack in the material.

CycleCount1 Cycle count for the first test.

CycleCount2 Cycle count for the second test.

CycleCount3 Cycle count for the third test.

CycleCount4 Cycle count for the fourth test.

CycleCount5 Cycle count for the fifth test.

CycleCount6 Cycle count for the sixth test.

CycleCount7 Cycle count for the seventh test.

CycleCount8 Cycle count for the eighth test.

CycleCount9 Cycle count for the ninth test.

CycleCount10 Cycle count for the tenth test.

CycleCount11 Cycle count for the eleventh test.

CycleCount12 Cycle count for the twelfth test.

CycleCount13 Cycle count for the thirteenth test.

CycleCount14 Cycle count for the fourteenth test.

CycleCount15 Cycle count for the fifteenth test.

CycleCount16 Cycle count for the sixteenth test.

CycleCount17 Cycle count for the seventeenth test.

CycleCount18 Cycle count for the eighteenth test.

CycleCount19 Cycle count for the nineteenth test.

CycleCount20 Cycle count for the twentieth test.

CycleCount21 Cycle count for the twenty-first test.

CycleCount22 Cycle count for the twenty-second test.

CycleCount23 Cycle count for the twenty-third test.

CycleCount24 Cycle count for the twenty-fourth test.

CycleCount25 Cycle count for the twenty-fifth test.

References

Virkler, D. A., Hillberry, B. M., and Goel, P. K. (1979). The statistical nature of fatigue crack propagation. Journal of Engineering Materials and Technology, 101 , 148–153 .

Examples

```
data(Virkler25)
i<-1 ## choose specimen number 1
ti<-Virkler25[,(i+1)]/10000  ## cycles at which the cracksize increases 0.2 units.
yt<-Virkler25[,1] ##
### the system is observed every 2000 cycles (0.2 unit times)
### observations:
t.obs<-seq(0,max(ti),by=0.2)
yi<-approx(x=ti,y=yt,xout=t.obs,method='linear',rule=2)$y
yi<-diff(yi)
#discretize the observations:
yi<-kmeans(yi,4)$cluster ## consider an alphabet of 4 signals
Nx<-2; # consider 2 hidden states
Ny<-4
alpha0<-c(1,0)
estim<-fit.hmmR(Y=yi,P0=NA,M0=NA,alpha0=alpha0,max.iter=50,epsilon=1e-9,Nx=Nx,Ny=Ny)
estim$P
estim$M
```

Index