Package ‘spc’

June 19, 2019

Version 0.6.1
Date 2019-06-18
Title Statistical Process Control -- Calculation of ARL and Other Control Chart Performance Measures
Author Sven Knoth
Maintainer Sven Knoth <Sven.Knoth@gmx.de>
Depends R (>= 1.8.0)
Description Evaluation of control charts by means of
the zero-state, steady-state ARL (Average Run Length) and RL quantiles.
Setting up control charts for given in-control ARL. The control charts
under consideration are one- and two-sided EWMA, CUSUM, and
Shiryaev-Roberts schemes for monitoring the mean or variance of normally
distributed independent data. ARL calculation of the same set of schemes un-
der drift (in the mean) are added.
Eventually, all ARL measures for the multivariate EWMA (MEWMA) are provided.
License GPL (>= 2)
URL https://www.r-project.org
NeedsCompilation yes
Repository CRAN
Date/Publication 2019-06-19 17:50:03 UTC

R topics documented:

dphat ................................................................. 3
euklid.ewma.arl ...................................................... 5
lns2ewma.arl ........................................................... 6
lns2ewma.crit .......................................................... 8
mewma.arl ............................................................. 10
mewma.crit ............................................................ 14
mewma.psi ............................................................. 15
p.ewma.arl ............................................................. 17
phat.ewma.arl ........................................................ 19
R topics documented:

- pois.ewma.arl
- pois.ewma.crit
- quadrature.nodes.weights
- scusum.arl
- scusum.crit
- scusums.arl
- sewma.arl
- sewma.arl.prerun
- sewma.crit
- sewma.crit.prerun
- sewma.q
- sewma.q.prerun
- sewma.sf
- sewma.sf.prerun
- tewma.arl
- tol.lim.fac
- x.res.ewma.arl
- xcusum.ad
- xcusum.arl
- xcusum.crit
- xcusum.crit.L0h
- xcusum.crit.L0L1
- xcusum.q
- xDcusum.sf
- xDewma.arl
- xDewma.arl.f
- xDewma.arl.prerun
- xewma.arl
- xewma.arl.prerun
- xewma.crit
- xewma.q
- xewma.q.prerun
- xewma.sf
- xewma.sf.prerun
- xgrsr.ad
- xgrsr.arl
- xgrsr.crit
- xsewma.arl
- xsewma.arl.crit
- xsewma.q
- xsewma.sf
- xshewhart.arl.arl
- xshewhartrunsrules.arl
- xtcusum.arl
- xtewma.ad
**dphat**

Percent defective for normal samples

**Description**

Density, distribution function and quantile function for the sample percent defective calculated on normal samples with mean equal to \( \mu \) and standard deviation equal to \( \sigma \).

**Usage**

```r
dphat(x, n, mu=0, sigma=1, type="known", LSL=-3, USL=3, nodes=30)
pophat(q, n, mu=0, sigma=1, type="known", LSL=-3, USL=3, nodes=30)
qphat(p, n, mu=0, sigma=1, type="known", LSL=-3, USL=3, nodes=30)
```

**Arguments**

- **x, q** vector of quantiles.
- **p** vector of probabilities.
- **n** sample size.
- **mu, sigma** parameters of the underlying normal distribution.
- **type** choose whether the standard deviation is given and fixed ("known") or estimated and potentially monitored ("estimated").
- **LSL, USL** lower and upper specification limit, respectively.
- **nodes** number of quadrature nodes needed for type="estimated".

**Details**

Bruhn-Suhr/Krumbholz (1990) derived the cumulative distribution function of the sample percent defective calculated on normal samples to applying them for a new variables sampling plan. These results were heavily used in Krumbholz/Zöller (1995) for Shewhart and in Knoth/Steinmetz (2013) for EWMA control charts. For algorithmic details see, essentially, Bruhn-Suhr/Krumbholz (1990). Two design variants are treated: The simple case, type="known", with known normal variance and the presumably much more relevant and considerably intricate case, type="estimated", where both parameters of the normal distribution are unknown. Basically, given lower and upper specification limits and the normal distribution, one estimates the expected yield based on a normal sample of size \( n \).
**dphat**

**Value**

Returns vector of pdf, cdf or qf values for the statistic phat.

**Author(s)**

Sven Knoth

**References**


**See Also**

`phat.eewma.ar1` for routines using the herewith considered phat statistic.

**Examples**

```r
# Figures 1 (c) and (d) from Knoth/Steinmetz (2013)
n <- 5
LSL <- -3
USL <- 3
par(mar=c(5, 5, 1, 1) + 0.1)

p.star <- 2*pnorm((LSL-USL)/2) # for p <= p.star pdf and cdf vanish

p_ <- seq(p.star+1e-10, 0.07, 0.0001) # define support of Figure 1

# Figure 1 (c)
pp_ <- pphat(p_, n)
plot(pp_, type="l", xlab="p", ylab=expression(P( hat(p) <= p ))),
     xlim=c(0, 0.06), ylim=c(0,1), lwd=2)
abline(h=0:1, v=p.star, col="grey")

# Figure 1 (d)
DP_ <- dphat(p_, n)
plot(p_, DP_, type="l", xlab="p", ylab="f(p)", xlim=c(0, 0.50),
     ylim=c(0,1), lwd=2)
abline(h=0, v=p.star, col="grey")
```
**Description**

Computation of the (zero-state) Average Run Length (ARL) at given Poisson mean \( \mu \).

**Usage**

\[
\text{euklid.ewma.arl}(gX, gY, kL, kU, \mu, y0, r0=0)
\]

**Arguments**

- \( gX \): first and
- \( gY \): second integer forming the rational \( \lambda = gX/(gX+gY) \), \( \lambda \) mimics the usual EWMA smoothing constant.
- \( kL \): lower control limit of the NCS-EWMA control chart, integer.
- \( kU \): upper control limit of the NCS-EWMA control chart, integer.
- \( \mu \): mean value of Poisson distribution.
- \( y0 \): headstart like value – it is proposed to use the in-control mean.
- \( r0 \): further element of the headstart – deviating from the default should be done only in case of full understanding of the scheme.

**Details**

A new idea of applying EWMA smoothing to count data based on integer division with remainders. It is highly recommended to read the corresponding paper (see below).

**Value**

Return single value which resemble the ARL.

**Author(s)**

Sven Knoth

**References**


**See Also**

later.
Examples

```r
# RCM (2015), Table 12, page 243, first NCS column
gX <- 5
gY <- 24
kL <- 16
kU <- 24
mu0 <- 20
#L0 <- euklid.ewma.arl(gX, gY, kL, kU, mu0, mu0)
# should be 1219.2
```

`lns2ewma.arl` *Compute ARLs of EWMA ln S^2 control charts (variance charts)*

Description

Computation of the (zero-state) Average Run Length (ARL) for different types of EWMA control charts (based on the log of the sample variance $S^2$) monitoring normal variance.

Usage

```r
lns2ewma.arl(l, cl, cu, sigma, df, hs=NULL, sided="upper", r=40)
```

Arguments

- `l` smoothing parameter lambda of the EWMA control chart.
- `cl` lower control limit of the EWMA control chart.
- `cu` upper control limit of the EWMA control chart.
- `sigma` true standard deviation.
- `df` actual degrees of freedom, corresponds to subsample size (for known mean it is equal to the subsample size, for unknown mean it is equal to subsample size minus one.
- `hs` so-called headstart (enables fast initial response) – the default value (hs=NULL) corresponds to the in-control mean of ln S^2.
- `sided` distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart with reflection at cl), "lower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.
- `r` dimension of the resulting linear equation system: the larger the better.

Details

`lns2ewma.arl` determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature.
**Value**

Returns a single value which resembles the ARL.

**Author(s)**

Sven Knoth

**References**


**See Also**

`xewma.ar1` for zero-state ARL computation of EWMA control charts for monitoring normal mean.

**Examples**

```r
lns2ewma.ARL <- Vectorize("lns2ewma.arl", "sigma")

## Crowder/Hamilton (1992)
## moments of ln S^2
E_log.gamma <- function(df) log(2/df) + digamma(df/2)
V_log.gamma <- function(df) trigamma(df/2)
E_log.gamma.approx <- function(df) -1/df - 1/3/df^2 + 2/15/df^4
V_log.gamma.approx <- function(df) 2/df + 2/df^2 + 4/3/df^3 - 16/15/df^5

## results from Table 3 (upper chart with reflection at 0 = log(sigma0=1) )
## original entries are (lambda = 0.05, K = 1.06, df=n-1=4)
# sigma ARL
# 1  200
# 1.1 43
# 1.2 18
# 1.3 11
# 1.4 7.6
# 1.5 6.0
# 2  3.2

df <- 4
lambda <- .05
K <- 1.06
cu <- K * sqrt( lambda/(2-lambda) * V_log.gamma.approx(df) )
sigmas <- c(1 + (0:5)/10, 2)
arl.s <- round(lns2ewma.ARL(lambda, 0, cu, sigmas, df, hs=0, sided="upper"), digits=1)
data.frame(sigmas, arls)

## Knoth (2005)
## compare with Table 3 (p. 351)
```
\begin{verbatim}
lambda <- .05
df <- 4
K <- 1.05521
cu <- 1.05521 * sqrt(lambda/(2-lambda) * V_log_gamma_approx(df))

## upper chart with reflection at sigma0=1 in Table 4
## original entries are
# sigma  ARL_0  ARL_-.267
# 1      200.0  200.0
# 1.1    43.04  41.55
# 1.2    18.10  19.92
# 1.3    10.75  13.11
# 1.4    7.63   9.93
# 1.5    5.97   8.11
# 2      3.17   4.67

M <- -0.267
cuM <- lns2ewma.crit(lambda, 200, df, cl=M, hs=M, r=60)[2]
arsl1 <- round(lns2ewma.ARL(lambda, 0, cu, sigma0, df, hs=0, sided="upper"), digits=2)
arsl2 <- round(lns2ewma.ARL(lambda, M, cuM, sigma0, df, hs=M, sided="upper", r=60), digits=2)
data.frame(sigma0, arsl1, arsl2)
\end{verbatim}

\begin{verbatim}
lns2ewma.crit  Compute critical values of EWMA ln S^2 control charts (variance charts)

Description

Computation of the critical values (similar to alarm limits) for different types of EWMA control charts (based on the log of the sample variance $S^2$) monitoring normal variance.

Usage

lns2ewma.crit(l1,L0,df,sigma0=1,cl=NULL,cu=NULL,hs=NULL,sided="upper",mode="fixed",r=40)

Arguments

l1  smoothing parameter lambda of the EWMA control chart.
L0  in-control ARL.
df  actual degrees of freedom, corresponds to subsample size (for known mean it is equal to the subsample size, for unknown mean it is equal to subsample size minus one.

sigma0  in-control standard deviation.
cl  deployed for sided="upper", that is, upper variance control chart with lower reflecting barrier cl.
cu  for two-sided (sided="two") and fixed upper control limit (mode="fixed"), for all other cases cu is ignored.
\end{verbatim}
hs

do-called head start (enables fast initial response) – the default value (hs=NULL) corresponds to the in-control mean of ln $S^2$.

sided

distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart with reflection at cl), "lower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.

mode

only deployed for sided="two" – with "fixed" an upper control limit (see cu) is set and only the lower is calculated to obtain the in-control ARL $l_P$, while with "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated). With "vanilla" limits symmetric around the in-control mean of ln $S^2$ are determined, while for "eq. tails" the in-control ARL values of two single EWMA variance charts (decompose the two-sided scheme into one lower and one upper scheme) are matched.

r
dimension of the resulting linear equation system: the larger the more accurate.

Details

lns2ewma.crit determines the critical values (similar to alarm limits) for given in-control ARL $l_P$ by applying secant rule and using lns2ewma.arl(). In case of sided="two" and mode="unbiased" a two-dimensional secant rule is applied that also ensures that the maximum of the ARL function for given standard deviation is attained at sigma0. See Knoth (2010) and the related example.

Value

Returns the lower and upper control limit cl and cu.

Author(s)

Sven Knoth

References


See Also

lns2ewma.arl for calculation of ARL of EWMA ln $S^2$ control charts.
Examples

```r
## Knoth (2005)
## compare with 1.05521 mentioned on page 350 third line from below
L0 <- 200
lambda <- .05
df <- 4
limits <- lns2ewma.crit(lambda, L0, df, cl=0, hs=0)
limits["cu"]/sqrt(limits/(2-lambda)*(2/df+2/df^2+4/3/df^3-16/15/df^5))
```

Description

Computation of the (zero-state) Average Run Length (ARL) for multivariate exponentially weighted moving average (MEWMA) charts monitoring multivariate normal mean.

Usage

```r
mewma.arl(l, cE, p, delta=0, hs=0, r=20, ntype=NULL, qm0=20, qm1=qm0)
mewma.arl.f(l, cE, p, delta=0, r=20, ntype=NULL, qm0=20, qm1=qm0)
mewma.ad(l, cE, p, delta=0, r=20, n=20, type="cond", hs=0, ntype=NULL, qm0=20, qm1=qm0)
```

Arguments

- `l` smoothing parameter lambda of the MEWMA control chart.
- `cE` alarm threshold of the MEWMA control chart.
- `p` dimension of multivariate normal distribution.
- `delta` magnitude of the potential change, `delta=0` refers to the in-control state.
- `hs` so-called headstart (enables fast initial response) – must be non-negative.
- `r` number of quadrature nodes – dimension of the resulting linear equation system for `delta=0`. For non-zero `delta` this dimension is mostly `r^2` (Markov chain approximation leads to some larger values). Caution: If `ntype` is set to "co" (collocation), then values of `r` larger than 20 lead to large computing times. For the other selections this would happen for values larger than 40.
- `ntype` choose the numerical algorithm to solve the ARL integral equation. For `delta=0`: Possible values are "gl", "gl2" (gauss-legendre, classic and with variables change: square), "co" (collocation, for `delta > 0` with sin transformation), "ra" (radau), "cc" (clenshaw-curtis), "mc" (markov chain), and "sr" (simpson rule). For `delta` larger than 0, some more values besides the others are possible: "gl3", "gl4", "gl5" (gauss-legendre with a further change in variables: sin, tan, sinh), "co2", "co3" (collocation with some trimming and tan as quadrature stabilizing transformations, respectively). If it is set to NULL (the
default), then for \( \delta = 0 \) then "gl2" is chosen. If \( \delta > 0 \), then for \( p = 2 \) or 4 "gl3" and for all other values "gl5" is taken. "ra" denotes the method used in Rigdon (1995a). "nc" denotes the Markov chain approximation. type switch between "cond" and "cyc1" for differentiating between the conditional (no false alarm) and the cyclical (after false alarm re-start in \( hs \)), respectively.

\( n \) number of quadrature nodes for calculating the steady-state ARL integral(s).

\( qm0, qm1 \) number of collocation quadrature nodes for the out-of-control case (\( qm0 \) for the inner integral, \( qm1 \) for the outer one), that is, for positive \( \delta \), and for the in-control case (now only \( qm0 \) is deployed) if via \( ntype \) the collocation procedure is requested.

Details

Basically, this is the implementation of different numerical algorithms for solving the integral equation for the MEWMA in-control (\( \delta = 0 \)) ARL introduced in Rigdon (1995a) and out-of-control (\( \delta \neq 0 \)) ARL in Rigdon (1995b). Most of them are nothing else than the Nyström approach – the integral is replaced by a suitable quadrature. Here, the Gauss-Legendre (more powerful), Radau (used by Rigdon, 1995a), Clenshaw-Curtis, and Simpson rule (which is really bad) are provided. Additionally, the collocation approach is offered as well, because it is much better for small odd values for \( p \). FORTRAN code for the Radau quadrature based Nyström of Rigdon (1995a) was published in Bodden and Rigdon (1999) – see also http://lib.stat.cmu.edu/jqt/31-1. Furthermore, FORTRAN code for the Markov chain approximation (in- and out-of-control) could be found at http://lib.stat.cmu.edu/jqt/33-4. The related papers are Runger and Prabhu (1996) and Molnau et al. (2001). The idea of the Clenshaw-Curtis quadrature was taken from Capizzi and Masarotto (2010), who successfully deployed a modified Clenshaw-Curtis quadrature to calculate the ARL of combined (univariate) Shewhart-EWMA charts. It turns out that it works also nicely for the MEWMA ARL. The version mewma.arl.f() without the argument \( hs \) provides the ARL as function of one (in-control) or two (out-of-control) arguments.

Value

Returns a single value which is simply the zero-state ARL.

Author(s)

Sven Knoth

References


**See Also**

`mewma.crit` for getting the alarm threshold to attain a certain in-control ARL.

**Examples**

```r
# Rigdon (1995a), p. 357, Tab. 1
p <- 2
r <- 0.25
h4 <- c(8.37, 9.90, 11.89, 13.36, 14.82, 16.72)
for (i in 1:length(h4)) cat(paste(h4[i], "t", round(mewma.arl(r, h4[i], p, ntype="ra")), "n"))

r <- 0.1
h4 <- c(6.98, 8.63, 10.77, 12.37, 13.88, 15.88)
for (i in 1:length(h4)) cat(paste(h4[i], "t", round(mewma.arl(r, h4[i], p, ntype="ra")), "n"))

# Rigdon (1995b), p. 372, Tab. 1
## Not run:
r <- 0.1
p <- 4
h <- 12.73
for (sdelta in c(0, 0.125, 0.25, .5, 1, 2, 3))
cat(paste("sdt",
round(mewma.arl(r, h, p, delta=sdelta*2, ntype="ra", r=25), digits=2), "n"))

p <- 5
h <- 14.56
for (sdelta in c(0, 0.125, 0.25, .5, 1, 2, 3))
cat(paste("sdt",
round(mewma.arl(r, h, p, delta=sdelta*2, ntype="ra", r=25), digits=2), "n"))

p <- 10
h <- 22.67
for (sdelta in c(0, 0.125, 0.25, .5, 1, 2, 3))
cat(paste("sdt",
round(mewma.arl(r, h, p, delta=sdelta*2, ntype="ra", r=25), digits=2), "n"))
```
## End (Not run)

# Runger/Prabhu (1996), p. 1704, Tab. 1
# Not run:

```r
r <- 0.1
p <- 4
H <- 12.73
cat(paste(0, "\t", round(mewma.arl(r, H, p, delta=0, ntype="mc", r=50), digits=2), "\n"))
for ( delta in c(.5, 1, 1.5, 2, 3) )
cat(paste(delta, "\t",
    round(mewma.arl(r, H, p, delta=delta, ntype="mc", r=25), digits=2), "\n"))
# compare with fortran program (MEWMA-ARLs.f90) from Molnau et al. (2001) with m1 = m2 = 25
# H4  P    R    DEL ARL
# 12.73 4. 0.10 0.00 199.78
# 12.73 4. 0.10 0.50 35.05
# 12.73 4. 0.10 1.00 12.17
# 12.73 4. 0.10 1.50 7.22
# 12.73 4. 0.10 2.00 5.19
# 12.73 4. 0.10 3.00 3.42
```

p <- 20
H <- 37.01
cat(paste(0, "\t",
    round(mewma.arl(r, H, p, delta=0, ntype="mc", r=50), digits=2), "\n"))
for ( delta in c(.5, 1, 1.5, 2, 3) )
cat(paste(delta, "\t",
    round(mewma.arl(r, H, p, delta=delta, ntype="mc", r=25), digits=2), "\n"))
# compare with Fortran program (MEWMA-ARLs.f90) from Molnau et al. (2001) with m1 = m2 = 25
# H4  P    R    DEL ARL
# 37.01 20. 0.10 0.00 199.09
# 37.01 20. 0.10 0.50 61.62
# 37.01 20. 0.10 1.00 20.17
# 37.01 20. 0.10 1.50 11.40
# 37.01 20. 0.10 2.00 8.03
# 37.01 20. 0.10 3.00 5.18

## End (Not run)

# Knoth (2017), p. 85, Tab. 3, rows with p=3
# Not run:

```r
p <- 3
lambda <- 0.05
h4 <- mewma.crit(lambda, 200, p)
benchmark <- mewma.arl(lambda, h4, p, delta=1, r=50)
mc.arl <- mewma.arl(lambda, h4, p, delta=1, r=25, ntype="mc")
ra.arl <- mewma.arl(lambda, h4, p, delta=1, r=27, ntype="ra")
co.arl <- mewma.arl(lambda, h4, p, delta=1, r=12, ntype="co2")
gl3.arl <- mewma.arl(lambda, h4, p, delta=1, r=30, ntype="gl3")
gl5.arl <- mewma.arl(lambda, h4, p, delta=1, r=25, ntype="gl5")
abs( benchmark - data.frame(mc.arl, ra.arl, co.arl, gl3.arl, gl5.arl) )
```
## mewma.crit

Compute alarm threshold of MEWMA control charts

### Description

Computation of the alarm threshold for multivariate exponentially weighted moving average (MEWMA) charts monitoring multivariate normal mean.

### Usage

mewma.crit(l, L0, p, hs=0, r=20)

### Arguments

- **l**: smoothing parameter lambda of the MEWMA control chart.
- **L0**: in-control ARL.
- **p**: dimension of multivariate normal distribution.
- **hs**: so-called headstart (enables fast initial response) – must be non-negative.
- **r**: number of quadrature nodes – dimension of the resulting linear equation system.

### Details

mewma.crit determines the alarm threshold of for given in-control ARL L0 by applying secant rule and using mewma.arl() with ntype="gl2".

### Value

Returns a single value which resembles the critical value c.
Compute steady-state density of the MEWMA statistic

Description

Computation of the (zero-state) steady-state density function of the statistic deployed in multivariate exponentially weighted moving average (MEWMA) charts monitoring multivariate normal mean.

Usage

```r
mewma.psi(l, cE, p, type="cond", hs=0, r=20)
```
 Arguments

  l  smoothing parameter lambda of the MEWMA control chart.
  ce alarm threshold of the MEWMA control chart.
  p  dimension of multivariate normal distribution.
  type  switch between "cond" and "cyc1" for differentiating between the conditional
        (no false alarm) and the cyclical (after false alarm re-start in hs), respectively.
  hs  the re-starting point for the cyclical steady-state framework.
  r  number of quadrature nodes.

 Details

 Basically, ideas from Knoth (2017, MEWMA numerics) and Knoth (2016, steady-state ARL concepts) are merged. More details will follow.

 Value

 Returns a function.

 Author(s)

 Sven Knoth

 References

 Sven Knoth (2018), The Steady-State Behavior of Multivariate Exponentially Weighted Moving Average Control Charts, Sequential Analysis 37(4), 511-529.

 See Also

 mewma.arl for calculating the in-control ARL of MEWMA.

 Examples

 lambda <- 0.1
 L0 <- 200
 p <- 3
 h4 <- mewma.crit(lambda, L0, p)
 x_ <- seq(0, h4+lambda/(2-lambda), by=0.002)
 psi <- mewma.psi(lambda, h4, p)
 psi_ <- psi(x_)
  # plot(x_, psi_, type="l", xlab="x", ylab=expression(psi(x)), xlim=c(0,1.2))
  # cf. to Figure 1 in Knoth (2018), p. 514, p=3
Compute ARLs of binomial EWMA $p$ control charts

**Description**

Computation of the (zero-state) Average Run Length (ARL) at given rate $p$.

**Usage**

```r
p.ewma.arl(lambda, ucl, n, p, z0, sided="upper", lcl=NULL, d.res=1, 
  r.mode="ieee.round", i.mode="integer")
```

**Arguments**

- `lambda`: smoothing parameter of the EWMA $p$ control chart.
- `ucl`: upper control limit of the EWMA $p$ control chart.
- `n`: subgroup size.
- `p`: (failure/success) rate.
- `z0`: so-called headstart (give fast initial response).
- `sided`: distinguishes between one- and two-sided EWMA control chart by choosing "upper", "lower", and "two", respectively.
- `lcl`: lower control limit of the EWMA $p$ control chart; needed for two-sided design.
- `d.res`: resolution (see details).
- `r.mode`: round mode – allowed modes are "gan.floor", "floor", "ceil", "ieee.round", "round", "mix".
- `i.mode`: type of interval center – "integer" or "half" integer.

**Details**

The monitored data follow a binomial distribution with size $n$ and failure/success probability $p$. The ARL values of the resulting EWMA control chart are determined by Markov chain approximation. Here, the original EWMA values are approximated by multiples of one over $d\text{.res}$. Different ways of rounding (see `r.mode`) to the next multiple are implemented. Besides Gan’s paper nothing is published about the numerical subtleties.

**Value**

Return single value which resemble the ARL.

**Author(s)**

Sven Knoth
References


See Also

later.

Examples

```r
## Gan (1990)

# Table 1

n <- 150
p0 <- .1
z0 <- n*p0

lambda <- c(1, .51, .165)
hu <- c(27, 22, 18)

p.value <- .1 + (0:20)/200

p.EWMA.arl <- Vectorize(p.ewma.arl, "p")

arl1.value <- round(p.EWMA.arl(lambda[1], hu[1], n, p.value, z0, r.mode="round"), digits=2)
arl2.value <- round(p.EWMA.arl(lambda[2], hu[2], n, p.value, z0, r.mode="round"), digits=2)
arl3.value <- round(p.EWMA.arl(lambda[3], hu[3], n, p.value, z0, r.mode="round"), digits=2)

arls <- matrix(c(arl1.value, arl2.value, arl3.value), ncol=length(lambda))
rownames(arls) <- p.value
colnames(arls) <- paste("lambda =", lambda)
arls

## Knoth/Steinmetz (2013)

n <- 5
p0 <- 0.02
z0 <- n*p0

lambda <- 0.3
ucl <- 0.649169922 # in-control ARL 370.4 (determined with d.res = Z^14 = 16384)

res.list <- 2^(1:11)
arl.list <- NULL
for (res in res.list) {
  arl <- p.ewma.arl(lambda, ucl, n, p0, z0, d.res=res)
arl.list <- c(arl.list, arl)
}
cbind(res.list, arl.list)
```
**Computes ARLs of EWMA phat control charts**

**Description**

Computation of the (zero-state) Average Run Length (ARL), upper control limit (ucl) for given in-control ARL, and lambda for minimal out-of control ARL at given shift.

**Usage**

```r
phat.ewma.arl(lambda, ucl, mu, n, z0, sigma="known", LSL=-3, USL=3, N=15, qm=25, ntype="coll")
```

```r
phat.ewma.crit(lambda, L0, mu, n, z0, sigma="known", LSL=-3, USL=3, N=15, qm=25)
```

```r
phat.ewma.lambda(L0, mu, n, z0, sigma="known", max_l=1, min_l=.001, LSL=-3, USL=3, qm=25)
```

**Arguments**

- `lambda`: smoothing parameter of the EWMA control chart.
- `ucl`: upper control limit of the EWMA phat control chart.
- `L0`: pre-defined in-control ARL (Average Run Length).
- `mu`: true mean or mean where the ARL should be minimized (then the in-control mean is simply 0).
- `n`: subgroup size.
- `z0`: so-called headstart (gives fast initial response).
- `type`: choose whether the standard deviation is given and fixed ("known") or estimated and potentially monitored ("estimated").
- `sigma`: actual standard deviation of the data – the in-control value is 1.
- `max_l, min_l`: maximal and minimal value for optimal lambda search.
- `LSL, USL`: lower and upper specification limit, respectively.
- `N`: size of collocation base, dimension of the resulting linear equation system is equal to N.
- `qm`: number of nodes for collocation quadratures.
- `ntype`: switch between the default method `coll` (collocation) and the classic one `markov` (Markov chain approximation) for calculating the ARL numerically.

**Details**

The three implemented functions allow to apply a new type control chart. Basically, lower and upper specification limits are given. The monitoring vehicle then is the empirical probability that an item will not follow these specification given the sequence of sample means. If the related EWMA sequence violates the control limits, then the alarm indicates a significant process deterioration. For details see the paper mentioned in the references. To be able to construct the control charts, see the first example.
Value

Return single values which resemble the ARL, the critical value, and the optimal lambda, respectively.

Author(s)

Sven Knoth

References


See Also

`sewma.arl` for a further collocation based ARL calculation routine.

Examples

```r
## Simple example to demonstrate the chart.

# some functions
h.mu <- function(mu) pnorm(LSL-mu) + pnorm(mu-USL)
ewma <- function(x, lambda=0.1, z0=0) filter(lambda*x, 1-lambda, m="r", init=z0)

# parameters
LSL <- -3        # lower specification limit
USL <- 3         # upper specification limit
n <- 5           # batch size
lambda <- 0.1    # EWMA smoothing parameter
L0 <- 1000       # in-control Average Run Length (ARL)
z0 <- h.mu(0)    # start at minimal defect level
ucl <- phat.ewma.crit(lambda, L0, 0, n, z0, LSL=LSL, USL=USL)

# data
x0 <- matrix(rnorm(50*n), ncol=5) # in-control data
x1 <- matrix(rnorm(50*n, mean=0.5), ncol=5)# out-of-control data
x <- rbind(x0,x1) # all data

# create chart
xbar <- apply(x, 1, mean)
phat <- h.mu(xbar)
z <- ewma(phat, lambda=lambda, z0=z0)
plot(1:length(z), z, type="l", xlab="batch", ylim=c(0,.02))
abline(h=z0, col="grey", lwd=.7)
abline(h=ucl, col="red")

## S. Knoth, S. Steinmetz (2013)

# Table 1
pois.ewma.arl

```r
lambdas <- c(.5, .25, .2, .1)
L0 <- 370.4
n <- 5
LSL <- -3
USL <- 3

phat.ewma.CRIT <- Vectorize("phat.ewma.crit", "lambda")
p.star <- pnorm(LSL) + pnorm(-USL) # lower bound of the chart
ucls <- phat.ewma.CRIT(lambdas, L0, 0, n, p.star, LSL=LSL, USL=USL)
print(cbind(lambdas, ucls))

# Table 2

mus <- c((0:4)/4, 1.5, 2, 3)
phat.ewma.ARL <- Vectorize("phat.ewma.arl", "mu")
arsls <- NULL
for (i in 1:length(lambdas)) {
  arls <- cbind(arsls, round(phat.ewma.ARL(lambdas[i], ucls[i], mus,
    n, p.star, LSL=LSL, USL=USL), digits=2))
}
arsls <- data.frame(arsls, row.names=NULL)
names(arsls) <- lambdas
print(arsls)

# Table 3

## Not run:
mus <- c(.25, .5, 1, 2)
phat.ewma.LAMBDA <- Vectorize("phat.ewma.lambda", "mu")
lambda <- phat.ewma.LAMBDA(L0, mus, n, p.star, LSL=LSL, USL=USL)
print(cbind(mus, lambda))
## End(Not run)
```

---

**pois.ewma.arl  Compute ARLs of Poisson EWMA control charts**

**Description**

Computation of the (zero-state) Average Run Length (ARL) at given mean \( \mu \).

**Usage**

```r
pois.ewma.arl(lambda, AL, AU, mu0, z0, mu, sided="two", rando=FALSE, gL=0, gU=0, mcdesign="transfer", N=101)
```

**Arguments**

- `lambda` smoothing parameter of the EWMA p control chart.
- `AL, AU` factors to build the lower and upper control limit, respectively, of the Poisson EWMA control chart.
mu0  in-control mean.
z0   so-called headstart (give fast initial response).
mu   actual mean.
sided distinguishes between one- and two-sided EWMA control chart by choosing "upper", "lower", and "two", and "zwei", respectively.
rando Switch between the standard limit treatment, FALSE, and an additional randomisation (to allow 'perfect' ARL calibration) by setting TRUE. If randomisation is used, then set the corresponding probabilities, gl and gu, appropriately.
gl, gu If the EWMA statistic is at the limit (approximately), then an alarm is triggered with probability gl and gu for the lower and upper limit, respectively.
mcdesign choose either "classic" which follows Borror, Champ and Rigdon (1998), or the more sophisticated "transfer" which improves the accuracy heavily.
N    number of states of the approximating Markov chain; is equal to the dimension of the resulting linear equation system.

Details

The monitored data follow a Poisson distribution with mu. The ARL values of the resulting EWMA control chart are determined by Markov chain approximation. We follow the algorithm given in Borror, Champ and Rigdon (1998).

Value

Return single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

later.

Examples

```r
## Borror, Champ and Rigdon (1998), Table 2, PEWMA column
mu0 <- 20
lambda <- 0.27
A <- 3.319
mul <- c(2*3:15, 35)
ARL1 <- rep(NA, length(mul))
for ( i in 1:length(mul) )
  ARL1[i] <- pois.eewma.arl(lambda, A, A, mu0, mu0, mul[i], mcdesign="classic")
print(cbind(mul, round(ARL1, digits=1)))
```
Description

Computation of the (zero-state) Average Run Length (ARL) at given mean $\mu$.

Usage

```r
pois.ewma.crit(lambda, L0, mu0, z0, AU=3, sided="two", design="sym", rando=FALSE, mcdesign="transfer", N=101, jmax=4)
```

Arguments

- `lambda`: smoothing parameter of the EWMA $p$ control chart.
- `L0`: value of the so-called in-control Average Run Length (ARL) for the Poisson EWMA control chart.
- `mu0`: in-control mean.
- `z0`: so-called headstart (give fast initial response).
- `AU`: in case of the lower chart deployed as reflecting upper barrier – might be increased step by step until the resulting lower limit does not change anymore.
- `sided`: distinguishes between one- and two-sided EWMA control chart by choosing "upper", "lower", and "two", respectively.
- `design`: distinguishes between limits symmetric to the in-control mean $\mu0$ and an ARL-unbiased design (ARL maximum at $\mu0$).
- `rando`: Switch between the standard limit treatment, FALSE, and an additional randomisation (to allow 'perfect' ARL calibration) by setting TRUE. If randomisation is used, then the corresponding probailities, $gl$ and $gu$ are determined, appropriately.
- `mcdesign`: choose either "classic" which follows Borror, Champ and Rigdon (1998), or the more sophisticated "transfer" which improves the accuracy heavily.
- `N`: number of states of the approximating Markov chain; is equal to the dimension of the resulting linear equation system.
- `jmax`: number of digits for the to be calculated factors $A$ (sort of accuracy).

Details

The monitored data follow a Poisson distribution with $\mu$. Here we solve the inverse task to the usual ARL calculation. Hence, determine the control limit factors so that the in-control ARL is (roughly) equal to $L0$. The ARL values underneath the routine are determined by Markov chain approximation. The algorithm is just a grid search that takes care of the discrete ARL behavior.

Value

Return one or two values being he control limit factors.
quadrature.nodes.weights

Author(s)
Sven Knoth

References

See Also
later.

Examples
```r
## Borror, Champ and Rigdon (1998), page 30, original value is A = 2.8275
mu0 <- 4
lambda <- 0.2
L0 <- 351
A <- pois.ewma.crit(lambda, L0, mu0, mu0, mcdesign="classic")
print(round(A, digits=4))
```

quadrature.nodes.weights

*Calculate quadrature nodes and weights*

Description
Computation of the nodes and weights to enable numerical quadrature.

Usage
```r
quadrature.nodes.weights(n, type="GL", x1=-1, x2=1)
```

Arguments
- `n` number of nodes (and weights).
- `type` quadrature type – currently Gauss-Legendre, "GL", and Radau, "Ra", are supported.
- `x1` lower limit of the integration interval.
- `x2` upper limit of the integration interval.

Details
A more detailed description will follow soon. The algorithm for the Gauss-Legendre quadrature was delivered by Knut Petras to me, while the one for the Radau quadrature was taken from John Burkardt.
Value

Returns two vectors which hold the needed quadrature nodes and weights.

Author(s)

Sven Knoth

References


See Also

Many of the ARL routines use the Gauss-Legendre nodes.

Examples

```r
# GL
n <- 10
qnw <- quadrature.nodes.weights(n, type="GL")
qnw

# Radau
n <- 10
qnw <- quadrature.nodes.weights(n, type="Ra")
qnw
```

---

**scusum.arl**  
Compute ARLs of CUSUM control charts (variance charts)

Description

Computation of the (zero-state) Average Run Length (ARL) for different types of CUSUM control charts (based on the sample variance $S^2$) monitoring normal variance.

Usage

```r
scusum.arl(k, h, sigma, df, hs=0, sided="upper", k2=NULL, h2=NULL, hs2=0, r=40, qm=30, version=2)
```
scusum.arl

Arguments

- **k**: reference value of the CUSUM control chart.
- **h**: decision interval (alarm limit, threshold) of the CUSUM control chart.
- **sigma**: true standard deviation.
- **df**: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one).
- **hs**: so-called headstart (enables fast initial response).
- **sided**: distinguishes between one- and two-sided two-sided CUSUM-S$^2$ control charts by choosing "upper" (upper chart), "lower" (lower chart), and "two" (two-sided chart), respectively. Note that for the two-sided chart the parameters "k2" and "h2" have to be set too.
- **k2**: In case of a two-sided CUSUM chart for variance the reference value of the lower chart.
- **h2**: In case of a two-sided CUSUM chart for variance the decision interval of the lower chart.
- **hs2**: In case of a two-sided CUSUM chart for variance the headstart of the lower chart.
- **r**: Dimension of the resulting linear equation system (highest order of the collocation polynomials times number of intervals – see Knoth 2006).
- **qm**: Number of quadrature nodes for calculating the collocation definite integrals.
- **version**: Distinguish version numbers (1,2,...). For internal use only.

Details

`scusum.arl` determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of collocation (piecewise Chebyshev polynomials).

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


### Description

Computation of the decision intervals (alarm limits) for different types of CUSUM control charts (based on the sample variance $S^2$) monitoring normal variance.

#### Usage

```r
scusum.crit(k, L0, sigma, df, hs=0, sided="upper", mode="eq.tails", k2=NULL, hs2=0, r=40, qm=30)
```

#### Arguments

- `k` reference value of the CUSUM control chart.
- `L0` in-control ARL.
- `sigma` true standard deviation.
actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.

so-called headstart (enables fast initial response).

distinguishes between one- and two-sided two-sided two-sided CUSUM-\(S^2\) control charts by choosing "upper" (upper chart), "lower" (lower chart), and "two" (two-sided chart), respectively. Note that for the two-sided chart the parameters "k2" and "h2" have to be set too.

only deployed for sided="two" – with "eq.tails" two one-sided CUSUM charts (lower and upper) with the same in-control ARL are coupled. With "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated).

in case of a two-sided CUSUM chart for variance the reference value of the lower chart.

in case of a two-sided CUSUM chart for variance the headstart of the lower chart.

Dimension of the resulting linear equation system (highest order of the collocation polynomials times number of intervals – see Knoth 2006).

Number of quadrature nodes for calculating the collocation definite integrals.

scusum.crit determines the decision interval (alarm limit) for given in-control ARL \(L0\) by applying secant rule and using scusum.arl().

Returns a single value which resembles the decision interval \(h\).

Sven Knoth


xcusum.arl for zero-state ARL computation of CUSUM control charts monitoring normal mean.
Examples

```r
## Knoth (2006)
## compare with Table 1 (p. 507)
k <- 1.46 # sigma1 = 1.5
df <- 1
L0 <- 260.74
h <- scusum.crit(k, L0, 1, df)
h
# original value is 10
```

### Description

Computation of the (zero-state) Average Run Length (ARL) for different types of CUSUM-Shewhart combo control charts (based on the sample variance $S^2$) monitoring normal variance.

### Usage

```r
scusums.arl(k, h, cS, sigma, df, hs=0, sided="upper", k2=NULL,
h2=NULL, hs2=0, r=40, qm=30, version=2)
```

### Arguments

- **k**: reference value of the CUSUM control chart.
- **h**: decision interval (alarm limit, threshold) of the CUSUM control chart.
- **cS**: Shewhart limit.
- **sigma**: true standard deviation.
- **df**: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one).
- **hs**: so-called headstart (enables fast initial response).
- **sided**: distinguishes between one- and two-sided two-sided CUSUM- $S^2$ control charts by choosing "upper" (upper chart), "lower" (lower chart), and "two" (two-sided chart), respectively. Note that for the two-sided chart the parameters "k2" and "h2" have to be set too.
- **k2**: In case of a two-sided CUSUM chart for variance the reference value of the lower chart.
- **h2**: In case of a two-sided CUSUM chart for variance the decision interval of the lower chart.
- **hs2**: In case of a two-sided CUSUM chart for variance the headstart of the lower chart.
- **r**: Dimension of the resulting linear equation system (highest order of the collocation polynomials times number of intervals – see Knoth 2006).
Number of quadrature nodes for calculating the collocation definite integrals.

Distinguish version numbers (1,2,...). For internal use only.

Details

`scusums.arl` determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of collocation (piecewise Chebyshev polynomials).

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

`scusum.arl` for zero-state ARL computation of standalone CUSUM control charts for monitoring normal variance.

Examples

```r
## will follow
```

**sewma.arl**

*Compute ARLs of EWMA control charts (variance charts)*

Description

Computation of the (zero-state) Average Run Length (ARL) for different types of EWMA control charts (based on the sample variance $S^2$) monitoring normal variance.

Usage

```r
sewma.arl(1,cl,cu,sigma,df,s2.on=TRUE,hs=NULL,sided="upper",r=40,qm=30)
```
Arguments

\[
\begin{align*}
1 & \quad \text{smoothing parameter lambda of the EWMA control chart.} \\
\text{c1} & \quad \text{lower control limit of the EWMA control chart.} \\
\text{cu} & \quad \text{upper control limit of the EWMA control chart.} \\
\text{sigma} & \quad \text{true standard deviation.} \\
\text{df} & \quad \text{actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one).} \\
\text{s2.on} & \quad \text{distinguishes between } S^2 \text{ and } S \text{ chart.} \\
\text{hs} & \quad \text{so-called headstart (enables fast initial response); the default (NULL) yields the expected in-control value of } S^2 (1) \text{ and } S (c_4), \text{ respectively.} \\
\text{sided} & \quad \text{distinguishes between one- and two-sided two-sided EWMA- } S^2 \text{ control charts by choosing "upper" (upper chart without reflection at c1 – the actual value of c1 is not used), "Rupper" (upper chart with reflection at c1), "Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.} \\
r & \quad \text{dimension of the resulting linear equation system (highest order of the collocation polynomials).} \\
\text{qm} & \quad \text{number of quadrature nodes for calculating the collocation definite integrals.}
\end{align*}
\]

Details

\text{sewma.arl} \text{ determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of collocation (Chebyshev polynomials).}

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

\text{xeewma.arl} \text{ for zero-state ARL computation of EWMA control charts for monitoring normal mean.}
Examples

```r
## Knoth (2005)
## compare with Table 1 (p. 347): 249.9997
## Monte Carlo with 10^8 replicates: 249.9892 +/- 0.008
l <- .025
df <- 1
cu <- 1 + 1.661865*sqrt(1/(2-l))*sqrt(2/df)
sewma.arl(l,0,cu,1,df)

## ARL values for upper and lower EWMA charts with reflecting barriers
## (reflection at in-control level sigma0 = 1)
## examples from Knoth (2006), Tables 4 and 5
Ssewma.arl <- Vectorize("ssewma.arl", "sigma")

## upper chart with reflection at sigma0=1 in Table 4
## original entries are
# sigma  ARL
# 1   100.0
# 1.01 85.3
# 1.02 73.4
# 1.03 63.5
# 1.04 55.4
# 1.05 48.7
# 1.1 27.9
# 1.2 12.9
# 1.3 7.86
# 1.4 5.57
# 1.5 4.30
# 2 2.11

## Not run:
l <- 0.15
df <- 4
cu <- 1 + 2.4831*sqrt(1/(2-l))*sqrt(2/df)
sigmas <- c(1 + (0:5)/100, 1 + (1:5)/10, 2)
arlsv <- round(Ssewma.arl(l, 1, cu, sigmas, df, sided="Rupper", r=100), digits=2)
data.frame(sigmas, arlsv)
## End(Not run)

## lower chart with reflection at sigma0=1 in Table 5
## original entries are
# sigma  ARL
# 1   200.04
# 0.9  38.47
# 0.8  14.63
# 0.7  8.65
# 0.6  6.31

## Not run:
l <- 0.115
df <- 5
```
Compute ARLs of EWMA control charts (variance charts) in case of estimated parameters

Description

Computation of the (zero-state) Average Run Length (ARL) for EWMA control charts (based on the sample variance $S^2$) monitoring normal variance with estimated parameters.

Usage

sewma.arl.prerun(l, cl, cu, sigma, df1, df2, hs=1, sided="upper", r=40, qm=30, qm.sigma=30, truncate=1e-10)

Arguments

1  smoothing parameter lambda of the EWMA control chart.
cl  lower control limit of the EWMA control chart.
cu  upper control limit of the EWMA control chart.
sigma  true standard deviation.
df1  actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
df2  degrees of freedom of the pre-run variance estimator.
hs  so-called headstart (enables fast initial response).
sided  distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at cl – the actual value of cl is not used), "Rupper" (upper chart with reflection at cl),"Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.
r  dimension of the resulting linear equation system (highest order of the collocation polynomials).
qm  number of quadrature nodes for calculating the collocation definite integrals.
qm.sigma  number of quadrature nodes for convoluting the standard deviation uncertainty.
truncate  size of truncated tail.

Details

Essentially, the ARL function sewma.arl is convoluted with the distribution of the sample standard deviation. For details see Jones/Champ/Rigdon (2001) and Knoth (2014?).
Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

`sewma.arl` for zero-state ARL function of EWMA control charts w/o pre run uncertainty.

Examples

```r
## will follow
```

---

**sewma.crit**  
*Compute critical values of EWMA control charts (variance charts)*

**Description**

Computation of the critical values (similar to alarm limits) for different types of EWMA control charts (based on the sample variance $S^2$) monitoring normal variance.

**Usage**

`sewma.crit(l, L0, df, sigma0, cl=NULL, cu=NULL, hs=NULL, s2.on=TRUE, sided="upper", mode="fixed", ur=4, r=40, qm=30)`

**Arguments**

- `l`  
  smoothing parameter lambda of the EWMA control chart.

- `L0`  
  in-control ARL.

- `df`  
  actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.

- `sigma0`  
  in-control standard deviation.
cl deployed for sided="Rupper", that is, upper variance control chart with lower reflecting barrier cl.

cu for two-sided (sided="two") and fixed upper control limit (mode="fixed") a value larger than sigma0 has to been given, for all other cases cu is ignored.

hs so-called headstart (enables fast initial response); the default (NULL) yields the expected in-control value of \( S^2 \) (1) and \( \bar{S} \) (c4), respectively.

s2.on distinguishes between \( S^2 \) and \( S \) chart.

sided distinguishes between one- and two-sided two-sided EWMA-S\(^2\) control charts by choosing "upper" (upper chart without reflection at cl – the actual value of cl is not used), "Rupper" (upper chart with reflection at cl), "Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.

mode only deployed for sided="two" – with "fixed" an upper control limit (see cu) is set and only the lower is calculated to obtain the in-control ARL L0, while with "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated). With "vanilla" limits symmetric around 1 (the in-control value of the variance) are determined, while for "eq.tails" the in-control ARL values of two single EWMA variance charts (decompose the two-sided scheme into one lower and one upper scheme) are matched.

ur truncation of lower chart for eq.tails mode.

r dimension of the resulting linear equation system (highest order of the collocation polynomials).

qm number of quadrature nodes for calculating the collocation definite integrals.

Details

sewma.crit determines the critical values (similar to alarm limits) for given in-control ARL L0 by applying secant rule and using sewma.arl(). In case of sided="two" and mode="unbiased" a two-dimensional secant rule is applied that also ensures that the maximum of the ARL function for given standard deviation is attained at sigma0. See Knoth (2010) and the related example.

Value

Returns the lower and upper control limit cl and cu.

Author(s)

Sven Knoth

References


**See Also**

`sewma.arl` for calculation of ARL of variance charts.

**Examples**

```r
## Mittag et al. (1998)
## compare their upper critical value 2.91 that
## leads to the upper control limit via the formula shown below
## (for the usual upper EWMA $sR/s^2$).
## See Knoth (2006b) for a discussion of this EWMA setup and it's evaluation.

l <- 0.18
L0 <- 250
df <- 4
limits <- sewma.crit(l, L0, df)
limits['cu']

limits.cu.mittag.et.al <- 1 + sqrt(l/(2-1))*sqrt(2/df)*2.91
limits.cu.mittag.et.al

## Knoth (2005)
## reproduce the critical value given in Figure 2 (c=1.661865) for
## upper EWMA $sR/s^2$ with df=1

l <- 0.025
L0 <- 250
df <- 1
limits <- sewma.crit(l, L0, df)
cv.Fig2 <- (limits['cu']-1)/(sqrt(l/(2-1))*sqrt(2/df))
cv.Fig2

## the small difference (sixth digit after decimal point) stems from
## tighter criterion in the secant rule implemented in the R package.

## demo of unbiased ARL curves
## Deploy, please, not matrix dimensions smaller than 50 -- for the
## sake of accuracy, the value 80 was used.
## Additionally, this example needs between 1 and 2 minutes on a 1.6 Ghz box.

## Not run:
l <- 0.1
```
sewma.crit.prerun

Compute critical values of of EWMA (variance charts) control charts under pre-run uncertainty

Description

Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal variance.

Usage

sewma.crit.prerun(L0, L0, df, cl=1, cu=1, sigma0=1, cl=0, cu=0, hs=1, sided="two", mode="unbiased", r=80, layout(matrix(1:2, nrow=1)), curve(SEWMA.ARL, .75, 1.25, log="y") curve(SEWMA.ARL, .95, 1.05, log="y")

## End(Not run)

# the above stuff needs about 1 minute

## control limits for upper and lower EWMA charts with reflecting barriers
## (reflection at in-control level sigma0 = 1)
## examples from Knoth (2006a), Tables 4 and 5

## Not run:
## upper chart with reflection at sigma0=1 in Table 4: c = 2.4831
L0 <- 500
df <- 4
limits <- sewma.crit(L0, L0, df, sided="two", mode="unbiased", r=80)
SEWMAarl <- Vectorize(SEWMAarl, "sigma")
SEWMA.ARL <- function(sigma)
  SEWMAarl(l, limits[1], limits[2], sigma, df, sided="two", r=80)
layout(matrix(1:2, nrow=1))
curve(SEWMA.ARL, .75, 1.25, log="y")
curve(SEWMA.ARL, .95, 1.05, log="y")

## lower chart with reflection at sigma0=1 in Table 5: c = 2.0613
L0 <- 100
df <- 4
limits <- sewma.crit(L0, L0, df, cl=1, sided="Rupper", r=100)
cv.Tab4 <- (limits["cu"]-1)/( sqrt(l/(2-l)) * sqrt(2/df) )
cv.Tab4

cv.Tab5 <- -(limits["cl"]-1)/( sqrt(l/(2-l)) * sqrt(2/df) )
cv.Tab5

## End(Not run)
Arguments

1
L
deconvolution parameter lambda of the EWMA control chart.

L0
in-control quantile value.

df1
actual degrees of freedom, corresponds to subgroup size (for known mean it is
equal to the subgroup size, for unknown mean it is equal to subgroup size minus
one.

df2
degrees of freedom of the pre-run variance estimator.

sigma,sigma0
true and in-control standard deviation, respectively.

c1
deployed for sided="Rupper", that is, upper variance control chart with lower
reflecting barrier c1.

cu
for two-sided (sided="two") and fixed upper control limit (mode="fixed") a
value larger than sigma0 has to been given, for all other cases cu is ignored.

hs
so-called headstart (enables fast initial response).

sided
distinguishes between one- and two-sided two-sided EWMA-S² control charts
by choosing "upper" (upper chart without reflection at c1 – the actual value of
c1 is not used), "Rupper" (upper chart with reflection at c1), "Rlower" (lower
chart with reflection at cu), and "two" (two-sided chart), respectively.

mode
only deployed for sided="two" – with "fixed" an upper control limit (see cu)
is set and only the lower is calculated to obtain the in-control ARL L0, while with
"unbiased" a certain unbiasedness of the ARL function is guaranteed (here,
both the lower and the upper control limit are calculated).

r
dimension of the resulting linear equation system (highest order of the collocation
polynomials).

qnm
number of quadrature nodes for calculating the collocation definite integrals.

qnm.sigma
number of quadrature nodes for convoluting the standard deviation uncertainty.

truncate
size of truncated tail.

tail_approx
controls whether the geometric tail approximation is used (is faster) or not.

c.error
error bound for two succeeding values of the critical value during applying the
secant rule.

a.error
error bound for the quantile level alpha during applying the secant rule.

Details

sewma.crit.prerun determines the critical values (similar to alarm limits) for given in-control
ARL L0 by applying secant rule and using sewma.arl.prerun(). In case of sided="two" and
mode="unbiased" a two-dimensional secant rule is applied that also ensures that the maximum of
the ARL function for given standard deviation is attained at sigma0. See Knoth (2010) for some
details of the algorithm involved.

Value

Returns the lower and upper control limit c1 and cu.
Author(s)
Sven Knoth

References

See Also
sewma.arl.prerun for calculation of ARL of variance charts under pre-run uncertainty and sewma.crit for the algorithm w/o pre-run uncertainty.

Examples
## will follow

sewma.q Compute RL quantiles of EWMA (variance charts) control charts

Description
Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal variance.

Usage
sewma.q(l, cl, cu, sigma, df, alpha, hs=1, sided="upper", r=40, qm=30)
sewma.q.crit(l, L0, alpha, df, sigma0=1, cl=NULL, cu=NULL, hs=1, sided="upper", mode="fixed", ur=4, r=40, qm=30, c.error=1e-12, a.error=1e-9)

Arguments

- **l**: smoothing parameter lambda of the EWMA control chart.
- **cl**: deployed for sided="Rupper", that is, upper variance control chart with lower reflecting barrier cl.
- **cu**: for two-sided (sided="two") and fixed upper control limit (mode="fixed") a value larger than sigma0 has to be given, for all other cases cu is ignored.
- **sigma, sigma0**: true and in-control standard deviation, respectively.
df
actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.

alpha
quantile level.

hs
so-called headstart (enables fast initial response).

sided
distinguishes between one- and two-sided two-sided EWMA-\(S^2\) control charts by choosing "upper" (upper chart without reflection at \(c_l\) – the actual value of \(c_l\) is not used), "Rupper" (upper chart with reflection at \(c_l\)), "Rlower" (lower chart with reflection at \(c_u\)), and "two" (two-sided chart), respectively.

mode
only deployed for \(\text{sided} = \text{"two"} \) – with "fixed" an upper control limit (see \(c_u\)) is set and only the lower is calculated to obtain the in-control ARL \(L_0\), while with "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated).

ur
truncation of lower chart for classic mode.

r
dimension of the resulting linear equation system (highest order of the collocation polynomials).

qm
number of quadrature nodes for calculating the collocation definite integrals.

l_0
in-control quantile value.

c.error
error bound for two succeeding values of the critical value during applying the secant rule.

a.error
error bound for the quantile level \(\alpha\) during applying the secant rule.

Details
Instead of the popular ARL (Average Run Length) quantiles of the EWMA stopping time (Run Length) are determined. The algorithm is based on Waldmann’s survival function iteration procedure. Thereby the ideas presented in Knoth (2007) are used. \texttt{sewma.q.crit} determines the critical values (similar to alarm limits) for given in-control RL quantile \(L_0\) at level \(\alpha\) by applying secant rule and using \texttt{sewma.sf()}. In case of \(\text{sided} = \text{"two"} \) and \(\text{mode} = \text{"unbiased"} \) a two-dimensional secant rule is applied that also ensures that the minimum of the cdf for given standard deviation is attained at \(\sigma_0\).

Value

Returns a single value which resembles the RL quantile of order \(\alpha\) and the lower and upper control limit \(c_l\) and \(c_u\), respectively.

Author(s)

Sven Knoth

References

H.-J. Mittag and D. Stemmann and B. Tewes (1998), EWMA-Karten zur Überwachung der Streuung von Qualitätsmerkmalen, Allgemeines Statistisches Archiv 82, 327-338,


**See Also**

`sewma.ar1` for calculation of ARL of variance charts and `sewma.sf` for the RL survival function.

**Examples**

```r
## will follow
```

---

**sewma.q.prerun**

*Compute RL quantiles of EWMA (variance charts) control charts under pre-run uncertainty*

**Description**

Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal variance.

**Usage**

```r
sewma.q.prerun(l, cl, cu, sigma, df1, df2, alpha, hs=1, sided="upper", r=40, qm=30, qm.sigma=30, truncate=1e-10)

sewma.q.crit.prerun(l, L0, alpha, df1, df2, sigma0=1, cl=NULL, cu=NULL, hs=1, sided="upper", mode="fixed", r=40, qm=30, qm.sigma=30, truncate=1e-10, tail_approx=TRUE, c.error=1e-10, a.error=1e-9)
```

**Arguments**

- `l` smoothing parameter lambda of the EWMA control chart.
- `cl` deployed for `sided="upper"`, that is, upper variance control chart with lower reflecting barrier `cl`.
- `cu` for two-sided (`sided="two"`) and fixed upper control limit (`mode="fixed"`) a value larger than `sigma0` has to been given, for all other cases `cu` is ignored.
sigma, sigma0: true and in-control standard deviation, respectively.
L0: in-control quantile value.
alpha: quantile level.
df1: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one).
df2: degrees of freedom of the pre-run variance estimator.
hs: so-called headstart (enables fast initial response).
sided: distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at c1 – the actual value of c1 is not used), "Rupper" (upper chart with reflection at c1), "Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.
mode: only deployed for sided="two" – with "fixed" an upper control limit (see cu) is set and only the lower is calculated to obtain the in-control ARL $l_P$, while with "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated).
r: dimension of the resulting linear equation system (highest order of the collocation polynomials).
qm: number of quadrature nodes for calculating the collocation definite integrals.
qm.sigma: number of quadrature nodes for convoluting the standard deviation uncertainty.
truncate: size of truncated tail.
tail_approx: controls whether the geometric tail approximation is used (is faster) or not.
c.error: error bound for two succeeding values of the critical value during applying the secant rule.
a.error: error bound for the quantile level alpha during applying the secant rule.

Details

Instead of the popular ARL (Average Run Length) quantiles of the EWMA stopping time (Run Length) are determined. The algorithm is based on Waldmann’s survival function iteration procedure. Thereby the ideas presented in Knoth (2007) are used. sewma.q.crit.prerun determines the critical values (similar to alarm limits) for given in-control RL quantile $l_0$ at level alpha by applying secant rule and using sewma.sf(). In case of sided="two" and mode="unbiased" a two-dimensional secant rule is applied that also ensures that the minimum of the cdf for given standard deviation is attained at sigma0.

Value

Returns a single value which resembles the RL quantile of order alpha and the lower and upper control limit c1 and cu, respectively.

Author(s)

Sven Knoth
References


See Also

`sewma.q` and `sewma.q.crit` for the version w/o pre-run uncertainty.

Examples

```r
# will follow
```

---

`sewma.sf`  
*Compute the survival function of EWMA run length*

Description

Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring normal variance.

Usage

```r
sewma.sf(n, l, cl, cu, sigma, df, hs=1, sided="upper", r=40, qm=30)
```

Arguments

- `n`: calculate sf up to value `n`.
- `l`: smoothing parameter lambda of the EWMA control chart.
- `cl`: lower control limit of the EWMA control chart.
- `cu`: upper control limit of the EWMA control chart.
- `sigma`: true standard deviation.
- `df`: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
- `hs`: so-called headstart (enables fast initial response).
- `sided`: distinguishes between one- and two-sided two-sided EWMA-\(S^2\) control charts by choosing "upper" (upper chart without reflection at `cl` – the actual value of `cl` is not used), "Rupper" (upper chart with reflection at `cl`), "Rlower" (lower chart with reflection at `cu`), and "two" (two-sided chart), respectively.
- `r`: dimension of the resulting linear equation system (highest order of the collocation polynomials).
- `qm`: number of quadrature nodes for calculating the collocation definite integrals.
Details

The survival function \( P(L>n) \) and derived from it also the cdf \( P(L\leq n) \) and the pmf \( P(L=n) \) illustrate the distribution of the EWMA run length. For large \( n \) the geometric tail could be exploited. That is, with reasonable large \( n \) the complete distribution is characterized. The algorithm is based on Waldmann’s survival function iteration procedure and on results in Knoth (2007).

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References

S. Knoth (2007), Accurate ARL calculation for EWMA control charts monitoring simultaneously normal mean and variance, Sequential Analysis 26, 251-264.


See Also

sewma.ar1 for zero-state ARL computation of variance EWMA control charts.

Examples

### will follow

---

**sewma.sf.prerun**

*Compute the survival function of EWMA run length*

---

**Description**

Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring normal variance.

**Usage**

\[
\text{sewma.sf.prerun}(n, l, cl, cu, sigma, df1, df2, hs=1, sided="upper", 
qm=30, qm.sigma=30, truncate=1e-10, tail_approx=TRUE)
\]
Arguments

- **n**: calculate sf up to value n.
- **l**: smoothing parameter lambda of the EWMA control chart.
- **cl**: lower control limit of the EWMA control chart.
- **cu**: upper control limit of the EWMA control chart.
- **sigma**: true standard deviation.
- **df1**: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one).
- **df2**: degrees of freedom of the pre-run variance estimator.
- **hs**: so-called headstart (enables fast initial response).
- **sided**: distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at cl – the actual value of cl is not used), "Rupper" (upper chart with reflection at cl), "Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.
- **qm**: number of quadrature nodes for calculating the collocation definite integrals.
- **qm.sigma**: number of quadrature nodes for convoluting the standard deviation uncertainty.
- **truncate**: size of truncated tail.
- **tail_approx**: Controls whether the geometric tail approximation is used (is faster) or not.

Details

The survival function $P(L>n)$ and derived from it also the cdf $P(L<=n)$ and the pmf $P(L=n)$ illustrate the distribution of the EWMA run length. For large n the geometric tail could be exploited. That is, with reasonable large n the complete distribution is characterized. The algorithm is based on Waldmann’s survival function iteration procedure and on results in Knoth (2007)...

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also

- `sewma.sf` for the RL survival function of EWMA control charts w/o pre-run uncertainty.
tewma.arl  

*Compute ARLs of Poisson TEWMA control charts*

**Description**

Computation of the (zero-state) Average Run Length (ARL) at given Poisson mean $\mu$.

**Usage**

tewma.arl(lambda, k, lk, uk, mu, z0, rando=FALSE, gl=0, gu=0)

**Arguments**

- `lambda`: smoothing parameter of the EWMA $p$ control chart.
- `k`: resolution of grid (natural number).
- `lk`: lower control limit of the TEWMA control chart, integer.
- `uk`: upper control limit of the TEWMA control chart, integer.
- `mu`: mean value of Poisson distribution.
- `z0`: so-called headstart (give fast initial response) – it is proposed to use the in-control mean.
- `rando`: Distinguish between control chart design without or with randomisation. In the latter case some meaningful values for $gl$ and $gu$ should be provided.
- `gl`: randomisation probability at the lower limit.
- `gu`: randomisation probability at the upper limit.

**Details**

A new idea of applying EWMA smoothing to count data. Here, the thinning operation is applied to independent Poisson variates is performed. Moreover, the original thinning principle is expanded to multiples of one over $k$ to allow finer grids and finally better detection performance. It is highly recommended to read the corresponding paper (see below).

**Value**

Return single value which resemble the ARL.

**Author(s)**

Sven Knoth
## References


## See Also

later.

## Examples

```r
# M9K (2018)
lambda <- 0.1 # (T)EWMA smoothing constant
mu0 <- 5 # in-control mean
k <- 10 # resolution
z0 <- round(k*mu0) # starting value of (T)EWMA sequence
# (i) without randomisation
lk <- 28
uk <- 75
L0 <- tewma.arl(lambda, k, lk, uk, mu0, z0)
# should be 501.9703
# (ii) with randomisation
uk <- 76 # lk is not changed
gl <- 0.5446310
gu <- 0.1375617
L1 <- tewma.arl(lambda, k, lk, uk, mu0, z0, rando=TRUE, gl=gl, gu=gu)
# should be 500
```

### tol.lim.fac

**Two-sided tolerance limit factors**

### Description

For constructing tolerance intervals, which cover a given proportion \( p \) of a normal distribution with unknown mean and variance with confidence \( 1 - \alpha \), one needs to calculate the so-called tolerance limit factors \( k \). These values are computed for a given sample size \( n \).

### Usage

```r
tol.lim.fac(n, p, a, mode="WW", m=30)
```

### Arguments

- **n**: sample size.
- **p**: coverage.
- **a**: error probability \( \alpha \), resulting interval covers at least proportion \( p \) with confidence of at least \( 1 - \alpha \).
- **mode**: distinguish between Wald/Wolfowitz’ approximation method ("WW") and the more accurate approach ("exact") based on Gauss-Legendre quadrature.
number of abscissas for the quadrature (needed only for method="exact"), of course, the larger the more accurate.

Details
tolлимfac determines tolerance limits factors k by means of the fast and simple approximation due to Wald/Wolfowitz (1946) and of Gauss-Legendre quadrature like Odeh/Owen (1980), respectively, who used in fact the Simpson Rule. Then, by \( \bar{x} \pm k \cdot s \) one can build the tolerance intervals which cover at least proportion \( p \) of a normal distribution for given confidence level of \( 1 - \alpha \). \( \bar{x} \) and \( s \) stand for the sample mean and the sample standard deviation, respectively.

Value

Returns a single value which resembles the tolerance limit factor.

Author(s)

Sven Knoth

References


See Also

qnorm for the "asymptotic" case – cf. second example.

Examples

```r
n <- 2:10
p <- .95
a <- .05
kWW <- sapply(n,p,a=0,tol.lin.fac)
kEX <- sapply(n,p,a=0,mode="exact",tol.lin.fac)
print(cbind(n,kWW,kEX),digits=4)
## Odeh/Owen (1980), page 98, in Table 3.4.1
## n factor k
## 2 36.519
## 3 9.789
## 4 6.341
## 5 5.077
## 6 4.422
## 7 4.020
## 8 3.746
## 9 3.546
## 10 3.393
## n -> infty
```
n <- 10^c(1:7)
p <- .95
a <- .05
kEX <- round(sapply(n,p=a,a=mu,mode="exact",tol=lim.fac),digits=4)
kEXinf <- round(qnorm(1-a/2),digits=4)
print(rbind(cbind(n,kEX),c("infinity",kEXinf)),quote=FALSE)

---

### Description

Computation of the (zero-state) Average Run Length (ARL) for EWMA residual control charts monitoring normal mean, variance, or mean and variance simultaneously. Additionally, the probability of misleading signals (PMS) is calculated.

### Usage

```r
x.res.ewma.arl(l, c, mu, alpha=0, n=5, hs=0, r=40)
s.res.ewma.arl(l, cu, sigma, mu=0, alpha=0, n=5, hs=1, r=40, qm=30)
x.s.res.ewma.arl(lx, cx, ls, csu, mu, sigma, alpha=0, n=5, hsx=0, rx=40, hss=1, rs=40, qm=30)
x.s.res.ewma.pms(lx, cx, ls, csu, mu, sigma, type="3", alpha=0, n=5, hsx=0, rx=40, hss=1, rs=40, qm=30)
```

### Arguments

- `l, lx, ls`: smoothing parameter(s) lambda of the EWMA control chart.
- `c, cu, cx, csu`: critical value (similar to alarm limit) of the EWMA control charts.
- `mu`: true mean.
- `sigma`: true standard deviation.
- `alpha`: the AR(1) coefficient – first order autocorrelation of the original data.
- `n`: batch size.
- `hs, hsx, hss`: so-called headstart (enables fast initial response).
- `r, rx, rs`: number of quadrature nodes or size of collocation base, dimension of the resulting linear equation system is equal to `r` (two-sided).
- `qm`: number of nodes for collocation quadratures.
- `type`: PMS type, for `PMS="3"` (the default) the probability of getting a mean signal despite the variance changed, and for `PMS="4"` the opposite case is dealt with.
Details

The above list of functions provides the application of algorithms developed for iid data to the residual case. To be more precise, the underlying model is a sequence of normally distributed batches with size \( n \) with autocorrelation within the batch and independence between the batches (see also the references below). It is restricted to the classical EWMA chart types, that is two-sided for the mean, upper charts for the variance, and all equipped with fixed limits. The autocorrelation is modeled by an AR(1) process with parameter \( \alpha \). Additionally, with \( xs.res.ewma.pms \) the probability of misleading signals (PMS) of type is calculated. This is offered exclusively in this small collection so that for iid data this function has to be used too (with \( \alpha = 0 \)).

Value

Return single values which resemble the ARL and the PMS, respectively.

Author(s)

Sven Knoth

References


See Also

\( xsewma.\text{arl}, sewma.\text{arl}, \) and \( xsewma.\text{arl} \) as more elaborated functions in the iid case.

Examples

```r
## Not run:

cat("\nFragments of Table 2 (n=5, lambda.1=lambda.2)\n")

lambdas <- c(.5, .25, .1, .05)
L0 <- 500
n <- 5

crit <- NULL
for (lambda in lambdas ) {
  cs <- xsewma.crit(lambda, lambda, L0, n-1)
  x.e <- round(cs[1], digits=4)
  names(x.e) <- NULL
  s.e <- round((cs[3]-1) * sqrt((2-lambda)/lambda)*sqrt((n-1)/2), digits=4)
```
```r
names(s.e) <- NULL
crit <- rbind(crit, data.frame(lambda, x.e, s.e))
}

## original values are (Markov chain approximation with 50 states)
# lambda x.e s.e
# 0.50 3.2765 4.6439
# 0.25 3.2168 4.0149
# 0.10 3.0578 3.3376
# 0.05 2.8817 2.9103

print(crit)

cat("\nFragments of Table 4 (n=5, lambda.1=lambda.2=0.1)\\n\\n")

lambda <- .1
# the algorithm used in Knoth/Schmid is less accurate -- proceed with their values
cx <- x.e <- 3.0578
s.e <- 3.3376
csu <- 1 + s.e * sqrt(lambda/(2-lambda))*sqrt(2/(n-1))
alpha <- .3
a.values <- c((0:6)/4, 2)
d.values <- c(1 + (0:5)/10, 1.75 , 2)

arls <- NULL
for ( delta in d.values ) {
  row <- NULL
  for ( mu in a.values ) {
    arl <- round(xs.res.ewma.arl(lambda, cx, lambda, csu, mu*sqrt(n), delta, alpha=alpha, n=n),
                  digits=2)
    names(arl) <- NULL
    row <- c(row, arl)
  }
  arls <- rbind(arls, data.frame(t(row)))

names(arls) <- a.values
rownames(arls) <- d.values

## original values are (now Monte-Carlo with 10^6 replicates)
#  0  0.25  0.5  0.75  1  1.25  1.5  2
#1 502.44  49.50 14.21  7.93  5.53  4.28  3.53  2.65
#1.1 73.19  32.91 13.33  7.82  5.52  4.29  3.54  2.66
#1.2 24.42  18.88 11.37  7.44  5.42  4.27  3.54  2.67
#1.3 13.11  11.83  9.09  6.74  5.18  4.17  3.50  2.66
#1.4  8.74  8.31  7.19  5.89  4.81  4.00  3.41  2.64
#1.5  6.50  6.31  5.80  5.08  4.37  3.76  3.28  2.59
#1.75  3.94  3.90  3.78  3.59  3.35  3.09  2.83  2.40
#2  2.85  2.84  2.80  2.73  2.63  2.51  2.39  2.14
```
print(arls)


cat("\nFragments of Table 5 (n=5, lambda=0.1)\n\n")

d.values <- c(1.02, 1 + (1:5)/10, 1.75, 2)
arl.x <- arl.s <- arl.xs <- PMS.3 <- NULL
for (delta in d.values ) {
arl.x <- c(arl.x, round(x.res.ewma.arl(lambda, cx/delta, 0, n=n),
digits=3))
arl.s <- c(arl.s, round(s.res.ewma.arl(lambda, csu, delta, n=n),
digits=3))
arl.xs <- c(arl.xs, round(xs.res.ewma.arl(lambda, cx, lambda, csu, 0, delta, n=n),
digits=3))
PMS.3 <- c(PMS.3, round(xs.res.ewma.pms(lambda, cx, lambda, csu, 0, delta, n=n),
digits=6))
}

## Original values are (Markov chain approximation)
# delta  arl.x  arl.s  arl.xs  PMS.3
# 1.02 833.086 518.935 323.324 0.381118
# 1.10 454.101 84.208 73.029 0.145005
# 1.20 250.665 25.871 24.432 0.071024
# 1.30 157.343 13.567 13.125 0.047193
# 1.40 108.112 8.941 8.734 0.035945
# 1.50 79.308 6.614 6.493 0.029499
# 1.75 44.128 3.995 3.942 0.021579
# 2.00 28.974 2.887 2.853 0.018220

print(cbind(delta=d.values, arl.x, arl.s, arl.xs, PMS.3))

cat("\nFragments of Table 6 (n=5, lambda=0.1)\n\n")

alphas <- c(-0.9, -0.5, -0.3, 0, 0.3, 0.5, 0.9)
deltas <- c(0.05, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2)
PMS.4 <- NULL
for ( ir in 1:length(deltas) ) {
  mu <- deltas[ir]*sqrt(n)
pms <- NULL
  for ( alpha in alphas ) {
    pms <- c(pms, round(xs.res.ewma.pms(lambda, cx, lambda, csu, mu, 1, type="4", alpha=alpha, n=n),
digits=6))
  }
PMS.4 <- rbind(PMS.4, data.frame(delta=deltas[ir], t(pms)))
}
names(PMS.4) <- c("delta", alphas)
rownames(PMS.4) <- NULL
## orinal values are (Markov chain approximation)
# delta -0.9 -0.5 -0.3 0 0.3 0.5 0.9
# 0.05 0.055789 0.224521 0.279842 0.342805 0.391299 0.418915 0.471386
# 0.25 0.003566 0.009522 0.014588 0.025786 0.044892 0.066584 0.192023
# 0.50 0.002994 0.001816 0.002596 0.004774 0.009259 0.015303 0.072945
# 0.75 0.006967 0.000783 0.000837 0.001529 0.003400 0.006424 0.046602
# 1.00 0.005098 0.000402 0.000370 0.000625 0.001589 0.003490 0.039978
# 1.25 0.000084 0.000266 0.000202 0.000300 0.000867 0.002220 0.039773
# 1.50 0.000000 0.000256 0.000120 0.000163 0.000531 0.001584 0.042734
# 2.00 0.000000 0.000311 0.000091 0.000056 0.000259 0.001029 0.054543

print(PMS.4)

## End(Not run)

---

### Compute steady-state ARLs of CUSUM control charts

**Description**

Computation of the steady-state Average Run Length (ARL) for different types of CUSUM control charts monitoring normal mean.

**Usage**

```r
cusum.ad(k, h, mu1, mu0 = 0, sided = "one", r = 30)
```

**Arguments**

- `k` reference value of the CUSUM control chart.
- `h` decision interval (alarm limit, threshold) of the CUSUM control chart.
- `mu1` out-of-control mean.
- `mu0` in-control mean.
- `sided` distinguish between one-, two-sided and Crosier's modified two-sided CUSUM scheme by choosing "one", "two", and "Crosier", respectively.
- `r` number of quadrature nodes, dimension of the resulting linear equation system is equal to `r+1` (one-, two-sided) or `2r+1` (Crosier).

**Details**

 cusum.ad determines the steady-state Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature and using the power method for deriving the largest in magnitude eigenvalue and the related left eigenfunction.

**Value**

Returns a single value which resembles the steady-state ARL.
Note

Be cautious in increasing the dimension parameter \( r \) for two-sided CUSUM schemes. The resulting matrix dimension is \( r^2 \) times \( r \). Thus, go beyond 30 only on fast machines. This is the only case, were the package routines are based on the Markov chain approach. Moreover, the two-sided CUSUM scheme needs a two-dimensional Markov chain.

Author(s)

Sven Knoth

References


See Also

`xcusum.ad` for zero-state ARL computation and `xewma.ad` for the steady-state ARL of EWMA control charts.

Examples

```r
## comparison of zero-state (= worst case ) and steady-state performance
## for one-sided CUSUM control charts

k <- .5
h <- xcusum.crit(k, 500)
mu <- c(0, .5, 1, 1.5, 2)
arl <- sapply(mu, k = k, h = h, xcusum.arl)
ad <- sapply(mu, k = k, h = h, xcusum.ad)
round(cbind(mu, arl, ad), digits = 2)

## Crosier (1986). Crosier's modified two-sided CUSUM
## He introduced the modification and evaluated it by means of
## Markov chain approximation

k <- .5
h2 <- 4
hc <- 3.73
mu <- c(0, .25, .5, .75, 1, 1.5, 2, 2.5, 3, 4.5)
ad2 <- sapply(mu, k = k, h = h2, sided = "two", r = 20, xcusum.ad)
adC <- sapply(mu, k = k, h = hc, sided = "Crosier", xcusum.ad)
round(cbind(mu, ad2, adC), digits = 2)

## results in the original paper are (in Table 5)
## 0.00 163. 164.
## 0.25 71.6 69.0
## 0.50 25.2 24.3
## 0.75 12.3 12.1
## 1.00 7.68 7.69
## 1.50 4.21 4.39
```
xcusum.arl

## 2.00 3.03 3.12
## 2.50 2.38 2.46
## 3.00 2.00 2.07
## 4.00 1.55 1.60
## 5.00 1.22 1.29

---

**xcusum.arl**  
*Compute ARLs of CUSUM control charts*

### Description

Computation of the (zero-state) Average Run Length (ARL) for different types of CUSUM control charts monitoring normal mean.

### Usage

```r
xcusum.arl(k, h, mu, hs = 0, sided = "one", method = "ig1", q = 1, r = 30)
```

### Arguments

- **k**: reference value of the CUSUM control chart.
- **h**: decision interval (alarm limit, threshold) of the CUSUM control chart.
- **mu**: true mean.
- **hs**: so-called headstart (give fast initial response).
- **sided**: distinguish between one-, two-sided and Crosier's modified two-sided CUSUM scheme by choosing "one", "two", and "Crosier", respectively.
- **method**: deploy the integral equation ("ig1") or Markov chain approximation ("mc") method to calculate the ARL (currently only for two-sided CUSUM implemented).
- **q**: change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu_l = 0 \) conditional delays, that is, \( E_q(L - q + 1 | L \geq q) \), will be determined. Note that \( mu0=0 \) is implicitly fixed.
- **r**: number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-, two-sided) or \( 2r+1 \) (Crosier).

### Details

*xcusum.arl* determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature.

### Value

Returns a vector of length \( q \) which resembles the ARL and the sequence of conditional expected delays for \( q=1 \) and \( q>1 \), respectively.
Author(s)
Sven Knoth

References


See Also

`xewma.ar1` for zero-state ARL computation of EWMA control charts and `xcusum.ad` for the steady-state ARL.

Examples

```r
## Brook/Evans (1972), one-sided CUSUM
## Their results are based on the less accurate Markov chain approach.

k <- .5
h <- 3
round(c( xcusum.arl(k,h,0), xcusum.arl(k,h,1.5) ),digits=2)

## results in the original paper are L0 = 117.59, L1 = 3.75 (in Subsection 4.3).

## Lucas, Crosier (1982)
## (one- and) two-sided CUSUM with possible headstarts

k <- .5
h <- 4
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,4,5)
arl1 <- sapply(mu,k=k,h=h,sided="two",xcusum.arl)
arl2 <- sapply(mu,k=k,h=h,hs=h/2,sided="two",xcusum.arl)
round(cbind(mu,arl1,arl2),digits=2)

## results in the original paper are (in Table 1)
## 0.00 168. 149.
## 0.25 74.2 62.7
## 0.50 26.6 20.1
## 0.75 13.3 8.97
```
## Vance (1986), one-sided CUSUM
## The first paper on using Nyström method and Gauss-Legendre quadrature
## for solving the ARL integral equation (see as well Goel/Wu, 1971)

```r
k <- 0
h <- 10
mu <- c(-.25,-.125,0,.125,.25,.5,.75,1)
round(cbind(mu,sapply(mu,k=k,h=h,xcusum.arl)),digits=2)
```

## results in the original paper are (in Table 1 incl. Goel/Wu (1971) results)
```
# -0.25  2071.51
# -0.125 400.28
#  0.0   124.66
#  0.125  59.30
#  0.25  36.71
#  0.50  20.37
#  0.75  14.06
#  1.00  10.75
```

## Waldmann (1986),
## one- and two-sided CUSUM

## one-sided case

```r
k <- .5
h <- 3
mu <- c(-.5,0,.5)
round(sapply(mu,k=k,h=h,xcusum.arl),digits=2)
```

## results in the original paper are 1963, 117.4, and 17.35, resp.
## (in Tables 3, 1, and 5, resp.).

## two-sided case

```r
k <- .6
h <- 3
round(xcusum.arl(k,h,-.2,sided="two"),digits=1) # fits to Waldmann's setup
```

## result in the original paper is 65.4 (in Table 6).

## Crosier (1986), Crosier's modified two-sided CUSUM
## He introduced the modification and evaluated it by means of
## Markov chain approximation

```r
k <- .5
```
xcusum.crit

Compute decision intervals of CUSUM control charts

Description

Computation of the decision intervals (alarm limits) for different types of CUSUM control charts monitoring normal mean.

Usage

xcusum.crit(k, L0, mu0 = 0, hs = 0, sided = "one", r = 30)
xcusum.crit.L0h

Arguments

k  
L0  
mu0  
hs  
sided  
r  

reference value of the CUSUM control chart.
in-control ARL.
in-control mean.
so-called headstart (enables fast initial response).
distinguishes between one-, two-sided and Crosier's modified two-sided CUSUM scheme by choosing "one", "two", and "Crosier", respectively.
number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-, two-sided) or 2r+1 (Crosier).

Details

xcusum.crit determines the decision interval (alarm limit) for given in-control ARL L0 by applying secant rule and using xcusum.arl().

Value

Returns a single value which resembles the decision interval h.

Author(s)

Sven Knoth

See Also

xcusum.arl for zero-state ARL computation.

Examples

k <- .5
incontrolARL <- c(500,5000,50000)
sapply(incontrolARL,k=k,xcusum.crit,r=10) # accuracy with 10 nodes
sapply(incontrolARL,k=k,xcusum.crit,r=20) # accuracy with 20 nodes
sapply(incontrolARL,k=k,xcusum.crit)      # accuracy with 30 nodes

xcusum.crit.L0h

Compute the CUSUM reference value k for given in-control ARL and threshold h

Description

Computation of the reference value k for one-sided CUSUM control charts monitoring normal mean, if the in-control ARL L0 and the alarm threshold h are given.

Usage

xcusum.crit.L0h(L0, h, hs=0, sided="one", r=30, L0.eps=1e-6, k.eps=1e-8)
Arguments

L0 in-control ARL.
h alarm level of the CUSUM control chart.
hs so-called headstart (enables fast initial response).
sided distinguishes between one-, two-sided and Crosier’s modified two-sided CUSUM scheme choosing "one", "two", and "Crosier", respectively.
r number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-, two-sided) or 2r+1 (Crosier).
L0.epsp error bound for the L0 error.
k.epsp bound for the difference of two successive values of k.

details

xcusum.crit.L0h determines the reference value k for given in-control ARL L0 and alarm level h by applying secant rule and using xcusum.arl(). Note that not for any combination of L0 and h a solution exists – for given L0 there is a maximal value for h to get a valid result k.

Value

Returns a single value which resembles the reference value k.

Author(s)

Sven Knoth

See Also

xcusum.arl for zero-state ARL computation.

Examples

L0 <- 100
h.max <- xcusum.crit(0, L0, 0)
hs <- (300:1)/100
hs <- hs[hs < h.max]
ks <- NULL
for ( h in hs ) ks <- c(ks, xcusum.crit.L0h(L0, h))
k.max <- qnorm( 1 - 1/L0 )
plot(hs, ks, type="l", ylim=c(0, max(k.max, ks)), xlab="h", ylab="k")
abline(h=c(0, k.max), col="red")
xcusum.crit.L0L1

Compute the CUSUM k and h for given in-control ARL L0 and out-of-control L1

Description

Computation of the reference value k and the alarm threshold h for one-sided CUSUM control charts monitoring normal mean, if the in-control ARL L0 and the out-of-control L1 are given.

Usage

xcusum.crit.L0L1(L0, L1, hs=0, sided="one", r=30, L1.eps=1e-6, k.eps=1e-8)

Arguments

L0    in-control ARL.
L1    out-of-control ARL.
hs    so-called headstart (enables fast initial response).
sided distinguishes between one-, two-sided and Crosier’s modified two-sided CUSUM schemes as “one”, “two”, and “Crosier”, respectively.
r    number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-, two-sided) or 2r+1 (Crosier).
L1.eps error bound for the L1 error.
k.eps bound for the difference of two successive values of k.

Details

xcusum.crit.L0L1 determines the reference value k and the alarm threshold h for given in-control ARL L0 and out-of-control ARL L1 by applying secant rule and using xcusum.arl() and xcusum.crit(). These CUSUM design rules were firstly (and quite rarely afterwards) used by Ewan and Kemp.

Value

Returns two values which resemble the reference value k and the threshold h.

Author(s)

Sven Knoth

References

W. D. Ewan and K. W. Kemp (1960), Sampling inspection of continuous processes with no autocorrelation between successive results, *Biometrika* 47, 363-380.

See Also

xcusum.arl for zero-state ARL and xcusum.crit for threshold h computation.

Examples

```r
## Table 2 from Ewan/Kemp (1960) -- one-sided CUSUM
#
# A.R.L. at A.Q.L. A.R.L. at A.Q.L. k  h
# 1000 3 1.12  2.40
# 1000 7 0.65  4.06
# 500  3 1.04  2.26
# 500  7 0.60  3.00
# 250  3 0.94  2.11
# 250  7 0.54  3.51
#
L0.set <- c(1000, 500, 250)
L1.set <- c(3, 7)
cat("\nL0\tL1\tk\n")
for ( L0 in L0.set ) {
  for ( L1 in L1.set ) {
    result <- round(xcusum.crit.L0L1(L0, L1), digits=2)
    cat(paste(L0, L1, result[1], result[2], sep="\t"), "\n")
  }
}
#
# two confirmation runs
xcusum.arl(0.54, 3.51, 0) # Ewan/Kemp
xcusum.arl(result[1], result[2], 0) # here
xcusum.arl(0.54, 3.51, 2*0.54) # Ewan/Kemp
xcusum.arl(result[1], result[2], 2*result[1]) # here
#
## Table II from Kemp (1962) -- two-sided CUSUM
#
# Lr k
# La=250 La=500 La=1000
# 2.5  1.05  1.17  1.27
# 3.0  0.94  1.035 1.13
# 4.0  0.78  0.85  0.92
# 5.0  0.68  0.74  0.80
# 6.0  0.60  0.655 0.71
# 7.5  0.52  0.57  0.62
# 10.0 0.43  0.48  0.52
#
L0.set <- c(250, 500, 1000)
L1.set <- c(2.5, 3.6, 7.5, 10)
cat("\nL0\nLa=250\tLa=500\tLa=1000\n")
for ( L1 in L1.set ) {
  cat(L1)
  for ( L0 in L0.set ) {
    result <- round(xcusum.crit.L0L1(L0, L1, sided="two"), digits=2)
    cat("\t", result[1])
  }
}
```
xcusum.q

Compute RL quantiles of CUSUM control charts

Description
Computation of quantiles of the Run Length (RL) for CUSUM control charts monitoring normal mean.

Usage
xcusum.q(k, h, mu, alpha, hs=0, sided="one", r=40)

Arguments
- k: reference value of the CUSUM control chart.
- h: decision interval (alarm limit, threshold) of the CUSUM control chart.
- mu: true mean.
- alpha: quantile level.
- hs: so-called headstart (enables fast initial response).
- sided: distinguishes between one- and two-sided CUSUM control chart by choosing "one" and "two", respectively.
- r: number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1.

Details
Instead of the popular ARL (Average Run Length) quantiles of the CUSUM stopping time (Run Length) are determined. The algorithm is based on Waldmann’s survival function iteration procedure.

Value
Returns a single value which resembles the RL quantile of order q.

Author(s)
Sven Knoth

References
xcusum.sf

See Also

xcusum.arl for zero-state ARL computation of CUSUM control charts.

Examples

### Waldmann (1986), one-sided CUSUM, Table 2
### original values are 345, 82, 9

```r
Xcusum.Q <- Vectorize("xcusum.q", "alpha")
k <- .5
h <- 3
mu <- 0 # corresponds to Waldmann's -0.5
a.list <- c(.95, .5, .05)
rl.quantiles <- ceiling(Xcusum.Q(k, h, mu, a.list))
cbind(a.list, rl.quantiles)
```

### Original values are 345, 82, 9

```r
cusum.sf(k, h, mu, n, hs=0, sided="one", r=40)
```

Arguments

- `k` reference value of the CUSUM control chart.
- `h` decision interval (alarm limit, threshold) of the CUSUM control chart.
- `mu` true mean.
- `n` calculate sf up to value n.
- `hs` so-called headstart (enables fast initial response).
- `sided` distinguishes between one- and two-sided CUSUM control chart by choosing "one" and "two", respectively.
- `r` number of quadrature nodes, dimension of the resulting linear equation system is equal to $r+1$.

Details

The survival function $P(L>n)$ and derived from it also the cdf $P(L<=n)$ and the pmf $P(L=n)$ illustrate the distribution of the CUSUM run length. For large n the geometric tail could be exploited. That is, with reasonable large n the complete distribution is characterized. The algorithm is based on Waldmann’s survival function iteration procedure.
Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also

xcusum.q for computation of CUSUM run length quantiles.

Examples

```r
## Waldmann (1986), one-sided CUSUM, Table 2

k <- .5
h <- 3
mu <- 0 # corresponds to Waldmann's -0.5
SF <- xcusum.sf(k, h, 0, 1000)
plot(1:length(SF), SF, type="l", xlab="n", ylab="P(L>n)", ylim=c(0,1))
```

---

Description

Computation of the (zero-state and other) Average Run Length (ARL) under drift for one-sided CUSUM control charts monitoring normal mean.

Usage

```r
xdcusum.arl(k, h, delta, hs = 0, sided = "one",
            mode = "Gan", m = NULL, q = 1, r = 30, with0 = FALSE)
```

Arguments

- `k` reference value of the CUSUM control chart.
- `h` decision interval (alarm limit, threshold) of the CUSUM control chart.
- `delta` true drift parameter.
- `hs` so-called headstart (enables fast initial response).
sided distinguishes between one- and two-sided CUSUM control chart by choosing "one" and "two", respectively. Currently, the two-sided scheme is not implemented.

type decide whether Gan’s or Knoth’s approach is used. Use "Gan" and "Knoth", respectively.

m parameter used if type="Gan". m is design parameter of Gan’s approach. If m=NULL, then m will increased until the resulting ARL does not change anymore.

q change point position. For q = 1 and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq L) \), will be determined. Note that mu0=0 is implicitly fixed. Deploy large q to mimic steady-state. It works only for type="Knoth".

r number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-sided) or r (two-sided).

with0 defines whether the first observation used for the RL calculation follows already 1*delta or still 0*delta. With q additional flexibility is given.

Details

Based on Gan (1991) or Knoth (2003), the ARL is calculated for one-sided CUSUM control charts under drift. In case of Gan’s framework, the usual ARL function with mu=m*delta is determined and recursively via m-1, m-2, ... 1 (or 0) the drift ARL determined. The framework of Knoth allows to calculate ARLs for varying parameters, such as control limits and distributional parameters. For details see the cited papers. Note that two-sided CUSUM charts under drift are difficult to treat.

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References

F. F. Gan (1992), CUSUM control charts under linear drift, Statistician 41, 71-84.


S. Knoth (2003), EWMA schemes with non-homogeneous transition kernels, Sequential Analysis 22, 241-255.


C. Zou, Y. Liu and Z. Wang (2009), Comparisons of control schemes for monitoring the means of processes subject to drifts, Metrika 70, 141-163.
See Also

xcusum.ar1 and xcusum.ad for zero-state and steady-state ARL computation of CUSUM control charts for the classical step change model.

Examples

```r
## Gan (1992)
## Table 1
## original values are
# deltas arl
#  0.0001  475
#  0.0005  261
#  0.0010  187
#  0.0020  129
#  0.0050   76.3
#  0.0100   52.0
#  0.0200   35.2
#  0.0500   21.4
#  0.1000   15.0
#  0.5000   6.95
#  1.0000   5.16
#  3.0000   3.30
## Not run: k <- .25
h <- 8
r <- 50
DxCusum.ar1 <- Vectorize(xDcusum.ar1, "delta")
deltas <- c(0.0001, 0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.5, 1, 3)
arl.like.Gan <-
  round(DxCusum.ar1(k, h, deltas, r=r, with0=TRUE), digits=2)
arl.like.Knoth <-
  round(DxCusum.ar1(k, h, deltas, r=r, mode="Knoth", with0=TRUE), digits=2)
data.frame(deltas, arl.like.Gan, arl.like.Knoth)
## End(Not run)

## Zou et al. (2009)
## Table 1
## original values are
# delta arl1 arl2 arl3
#  0   - 1730
# 0.005  345  412  470
# 0.001  231  275  317
# 0.005  86.6  98.6 112
# 0.01  56.9  61.8  69.3
# 0.05  22.6  21.6  22.7
# 0.1   15.4  14.7  14.2
# 0.5   6.60  5.54  5.17
# 1.0   4.63  3.80  3.45
# 2.0   3.17  2.67  2.32
# 3.0   2.79  2.04  1.96
# 4.0   2.10  1.98  1.74
## Not run:
k1 <- 0.25
```
Compute ARLs of EWMA control charts under drift

Description

Computation of the (zero-state and other) Average Run Length (ARL) under drift for different types of EWMA control charts monitoring normal mean.

Usage

xdewma.arl(l, c, delta, zr = 0, hs = 0, sided = "one", limits = "fix", mode = "Gan", m = NULL, q = 1, r = 40, with0 = FALSE)

Arguments

- **l**: smoothing parameter lambda of the EWMA control chart.
- **c**: critical value (similar to alarm limit) of the EWMA control chart.
- **delta**: true drift parameter.
- **zr**: reflection border for the one-sided chart.
- **hs**: so-called headstart (enables fast initial response).
- **sided**: distinguish between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
- **limits**: distinguishes between different control limits behavior.
- **mode**: decide whether Gan's or Knoth's approach is used. Use "Gan" and "Knoth", respectively.
- **m**: parameter used if mode="Gan". m is design parameter of Gan’s approach. If m=NULL, then m will increased until the resulting ARL does not change anymore.
- **q**: change point position. For q = 1 and μ = μ₀ and μ = μ₁, the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For q > 1 and μ₁ = 0 conditional delays, that is, E_q(L - q + 1/L ≥), will be determined. Note that μ0=0 is implicitly fixed. Deploy large q to mimic steady-state. It works only for mode="Knoth".
number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-sided) or \( r \) (two-sided).

\[
\text{with} \theta \quad \text{defines whether the first observation used for the RL calculation follows already 1*delta or still 0*delta. With q additional flexibility is given.}
\]

Details

Based on Gan (1991) or Knoth (2003), the ARL is calculated for EWMA control charts under drift. In case of Gan's framework, the usual ARL function with \( \mu=m*\delta \) is determined and recursively via \( m-1, m-2, \ldots, 1 \) (or 0) the drift ARL determined. The framework of Knoth allows to calculate ARLs for varying parameters, such as control limits and distributional parameters. For details see the cited papers.

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

`xewma.arl` and `xewma.ad` for zero-state and steady-state ARL computation of EWMA control charts for the classical step change model.

Examples

```r
## Not run:
DxDewma.arl <- Vectorize(xDewma.arl, "delta")
## Gan (1991)
## Table 1
## original values are
# delta  arlE1  arlE2  arlE3
```
# 0  500  500  500
# 0.0001  482  460  424
# 0.0010  289  231  185
# 0.0020  210  162  129
# 0.0050  126  94.6  77.9
# 0.0100  81.7  61.3  52.7
# 0.0500  27.5  21.8  21.9
# 0.1000  17.0  14.2  15.3
# 1.0000   4.09  4.28  5.25
# 3.0000  2.60  2.90  3.43

# lambdaQ < 0.676
lambda2 <- 0.242
lambda3 <- 0.047
h1 <- 2.204/sqrt(lambda1/(2-lambda1))
h2 <- 1.111/sqrt(lambda2/(2-lambda2))
h3 <- 0.403/sqrt(lambda3/(2-lambda3))
deltas <- c(.0001, .001, .002, .005, .01, .05, .1, 1, 3)
arlE10 <- round(xewma.arl(lambda1, h1, 0, sided="two"), digits=2)
arlE1 <- c(arlE10, round(DxDewma.arl(lambda1, h1, deltas, sided="two", with=TRUE),
digits=2))
arlE20 <- round(xewma.arl(lambda2, h2, 0, sided="two"), digits=2)
arlE2 <- c(arlE20, round(DxDewma.arl(lambda2, h2, deltas, sided="two", with=TRUE),
digits=2))
arlE30 <- round(xewma.arl(lambda3, h3, 0, sided="two"), digits=2)
arlE3 <- c(arlE30, round(DxDewma.arl(lambda3, h3, deltas, sided="two", with=TRUE),
digits=2))
data.frame(delta=c(0, deltas), arlE1, arlE2, arlE3)

## do the same with more digits for the alarm threshold
L0 <- 500
h1 <- xewma.crit(lambda1, L0, sided="two")
h2 <- xewma.crit(lambda2, L0, sided="two")
h3 <- xewma.crit(lambda3, L0, sided="two")
lambdas <- c(lambda1, lambda2, lambda3)
hs <- c(h1, h2, h3) * sqrt(lambdas/(2-lambdas))
hs
# compare with Gan's values 2.204, 1.111, 0.403
round(hs, digits=3)
arlE10 <- round(xewma.arl(lambda1, h1, 0, sided="two"), digits=2)
arlE1 <- c(arlE10, round(DxDewma.arl(lambda1, h1, deltas, sided="two", with=TRUE),
digits=2))
arlE20 <- round(xewma.arl(lambda2, h2, 0, sided="two"), digits=2)
arlE2 <- c(arlE20, round(DxDewma.arl(lambda2, h2, deltas, sided="two", with=TRUE),
digits=2))
arlE30 <- round(xewma.arl(lambda3, h3, 0, sided="two"), digits=2)
arlE3 <- c(arlE30, round(DxDewma.arl(lambda3, h3, deltas, sided="two", with=TRUE),
digits=2))
data.frame(delta=c(0, deltas), arlE1, arlE2, arlE3)

## Aerne et al. (1991) -- two-sided EWMA
## Table I (continued)
```r
## original numbers are
#     delta  arle1  arle2  arle3
# 0.000000 465.0  465.0  465.0
# 0.005623  77.0  85.93 102.68
# 0.007499  64.61  71.78  85.74
# 0.010000  54.20  59.74  71.22
# 0.013335  45.20  49.58  58.90
# 0.017783  37.76  41.06  48.54
# 0.023714  31.54  33.95  39.87
# 0.031623  26.36  28.06  32.68
# 0.042170  22.06  23.19  26.73
# 0.056234  18.49  19.17  21.84
# 0.074989  15.53  15.87  17.83
# 0.100000  13.07  13.16  14.55
# 0.133352  11.03  10.94  11.88
# 0.177828   9.33   9.12   9.71
# 0.237137   7.91   7.62   7.95
# 0.316228   6.72   6.39   6.52
# 0.421697   5.72   5.38   5.37
# 0.562341   4.88   4.54   4.44
# 0.749894   4.18   3.84   3.68
# 1.000000   3.58   3.27   3.07

# lambda1 <- .133
# lambda2 <- .25
# lambda3 <- .5
c1 <- 2.856
c2 <- 2.974
c3 <- 3.049
deltas <- 10^(--(18:0)/8)
arle1 <- round(xewma.arl(lambda1, c1, 0, sided="two"), digits=2)
arle1 <- c(arle10, round(Dxewma.arl(lambda1, c1, deltas, sided="two"), digits=2))
arle2 <- round(xewma.arl(lambda2, c2, 0, sided="two"), digits=2)
arle2 <- c(arle20, round(Dxewma.arl(lambda2, c2, deltas, sided="two"), digits=2))
arle3 <- round(xewma.arl(lambda3, c3, 0, sided="two"), digits=2)
arle3 <- c(arle30, round(Dxewma.arl(lambda3, c3, deltas, sided="two"), digits=2))
data.frame(delta=c(0, round(deltas, digits=6)), arle1, arle2, arle3)

## Fahmy/Elsayed (2006) -- two-sided EWMA
## Table 4 (Monte Carlo results, 10^4 replicates, change point at t=51!)
## original numbers are
#     delta  arl  s.e.
# 0.00  365.749 3.598
# 0.10  12.971  0.029
# 0.25   7.738  0.015
# 0.50   5.312  0.009
# 0.75   4.279  0.007
# 1.00   3.680  0.006
# 1.25   3.271  0.006
# 1.50   2.979  0.005
# 1.75   2.782  0.004
# 2.00   2.598  0.005
```
# Additional Monte Carlo results with 10^8 replicates
#
# Zou et al. (2009) -- one-sided EWMA
## Table 1
## Original values are
#
# delta arl.c arl.R arl.S
# 0 368.910 368.010 361.714
# 0.1 12.986 12.781 12.000
# 0.25 7.758 7.637 7.000
# 0.5 5.318 5.235 5.000
# 0.75 4.285 4.218 4.000
# 1.0 3.688 3.628 3.000
# 1.25 3.274 3.233 3.000
# 1.5 2.993 2.942 2.000
# 1.75 2.808 2.723 2.000
# 2.0 2.616 2.554 2.000

```r
# lambda <- 0.1
cE <- 2.7
deltas <- c(-1, (1:8)/4)
arlE1 <- c(round(xewma.arl(lambda, cE, 0, sided="two"), digits=3),
round(Dxewma.arl(lambda, cE, deltas, sided="two"), digits=3))
arlE51 <- c(round(xewma.arl(lambda, cE, 0, sided="two", q=51)[51], digits=3),
round(Dxewma.arl(lambda, cE, deltas, sided="two", mode="Knot", q=51),
digits=3))
data.frame(delta=c(0, deltas), arlE1, arlE51)
```
Compute ARLs of Shiryaev-Roberts schemes under drift

Description

Computation of the (zero-state and other) Average Run Length (ARL) under drift for Shiryaev-Roberts schemes monitoring normal mean.

Usage

```r
xDgrsr.arl(k, g, delta, zr = 0, hs = NULL, sided = "one", m = NULL,
mode = "Gan", q = 1, r = 30, with0 = FALSE)
```

Arguments

- `k` reference value of the Shiryaev-Roberts scheme.
- `g` control limit (alarm threshold) of Shiryaev-Roberts scheme.
- `delta` true drift parameter.
- `zr` reflection border for the one-sided chart.
- `hs` so-called headstart (enables fast initial response).
- `sided` distinguishes between one- and two-sided Shiryaev-Roberts schemes by choosing "one" and "two", respectively. Currently, the two-sided scheme is not implemented.
- `m` parameter used if `mode="Gan"`. `m` is design parameter of Gan’s approach. If `m=NULL`, then `m` will increased until the resulting ARL does not change anymore.
- `q` change point position. For `q = 1` and `\mu = \mu_0` and `\mu = \mu_1`, the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For `q > 1` and `\mu! = 0` conditional delays, that is, `E_q(L - q + 1|L \geq)`, will be determined. Note that `mu0=0` is implicitly fixed. Deploy large `q` to mimic steady-state. It works only for `mode="Knoth"`.
- `mode` decide whether Gan’s or Knoth’s approach is used. Use "Gan" and "Knoth", respectively. "Knoth" is not implemented yet.
- `r` number of quadrature nodes, dimension of the resulting linear equation system is equal to `r+1` (one-sided) or `r` (two-sided).
- `with0` defines whether the first observation used for the RL calculation follows already `1*delta` or still `0*delta`. With `q` additional flexibility is given.
Details

Based on Gan (1991) or Knoth (2003), the ARL is calculated for Shiryaev-Roberts schemes under drift. In case of Gan’s framework, the usual ARL function with mu=m*delta is determined and recursively via m-1, m-2, ..., 1 (or 0) the drift ARL determined. The framework of Knoth allows to calculate ARLs for varying parameters, such as control limits and distributional parameters. For details see the cited papers.

Value

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References


See Also

xewma.arl and xewma.ad for zero-state and steady-state ARL computation of EWMA control charts for the classical step change model.

Examples

```r
## Not run:
## Monte Carlo example with 10^8 replicates
# delta arl s.e.
# 0.0001 381.8240 0.0304
# 0.0005 238.4630 0.0148
# 0.001 177.4061 0.0097
# 0.002 125.9055 0.0061
# 0.005 75.7574 0.0031
# 0.01 50.2203 0.0018
# 0.02 32.9458 0.0011
# 0.05 18.9213 0.0005
# 0.1 12.6054 0.0003
# 0.5 5.2157 0.0001
# 1 3.6537 0.0001
# 3 2.0289 0.0000
k <- .5
l0 <- 500
zr <- -7
```
r <- 50
g <- xgrsr.crit(k, L0, zr=zh, r=r)
DxDgrsr.arl <- Vectorize(xDgrsr.arl, "delta")
deltas <- c(0.0001, 0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.5, 1, 3)
arsl <- round(DxDgrsr.arl(k, g, deltas, zr=zh, r=r), digits=4)
data.frame(deltas, arsl)

## End(Not run)

---

**xDshewhartrunsrules.arl**

*Compute ARLs of Shewhart control charts with and without runs rules under drift*

**Description**

Computation of the zero-state Average Run Length (ARL) under drift for Shewhart control charts with and without runs rules monitoring normal mean.

**Usage**

`xDshewhartrunsrules.arl(delta, c = 1, m = NULL, type = "12")`

`xDshewhartrunsrulesFixedm.arl(delta, c = 1, m = 100, type = "12")`

**Arguments**

- `delta`: true drift parameter.
- `c`: normalizing constant to ensure specific alarming behavior.
- `type`: controls the type of Shewhart chart used, seed details section.
- `m`: parameter of Gan’s approach. If m=NULL, then m will increased until the resulting ARL does not change anymore.

**Details**

Based on Gan (1991), the ARL is calculated for Shewhart control charts with and without runs rules under drift. The usual ARL function with mu=m*delta is determined and recursively via m-1, m-2, ... 1 (or 0) the drift ARL determined. `xDshewhartrunsrulesFixedm.arl` is the actual work horse, while `xDshewhartrunsrules.arl` provides a convenience wrapper. Note that Aerne et al. (1991) deployed a method that is quite similar to Gan’s algorithm. For type see the help page of `xshewhartrunsrules.arl`.

**Value**

Returns a single value which resembles the ARL.
Author(s)
Sven Knoth

References

See Also
xshewhartrunsrules.arl for zero-state ARL computation of Shewhart control charts with and without runs rules for the classical step change model.

Examples
```r
## Aerne et al. (1991)
## Table I (continued)
## original numbers are
# delta arl1of1 arl2of3 arl14of5 arl10
# 0.005623 136.67 120.90 105.34 107.08
# 0.007499 114.98 101.23 88.09 89.94
# 0.010000 96.03 84.22 73.31 75.23
# 0.013335 79.69 69.68 60.75 62.73
# 0.017783 65.75 57.38 50.18 52.18
# 0.023714 53.99 47.06 41.33 43.35
# 0.031623 44.15 38.47 33.99 36.00
# 0.042170 35.97 31.36 27.91 29.90
# 0.056234 29.21 25.51 22.91 24.86
# 0.074989 23.65 20.71 18.81 20.70
# 0.100000 19.11 16.79 15.45 17.29
# 0.133352 15.41 13.61 12.72 14.47
# 0.177828 12.41 11.03 10.50 12.14
# 0.237137 9.98 8.94 8.71 10.18
# 0.316228 8.02 7.25 7.26 8.45
# 0.421697 6.44 5.89 6.09 6.84
# 0.562341 5.17 4.80 5.15 5.48
# 0.749894 4.16 3.92 4.36 4.39
# 1.000000 3.35 3.22 3.63 3.52
c1of1 <- 3.0693
c2of3 <- 2.1494/2
c4of5 <- 1.14
c10 <- 3.2425/3
DxDs hysteratriunrules.arl <- Vectorize(xDshewhartrunsrules.arl, "delta")
deltas <- 10^(-18:0)/8
arl1of1 <- round(DxDs hysteratriunrules.arl(deltas, c=c1of1, type="1"), digits=2)
arl2of3 <- round(DxDs hysteratriunrules.arl(deltas, c=c2of3, type="12"), digits=2)
arl14of5 <- round(DxDs hysteratriunrules.arl(deltas, c=c4of5, type="13"), digits=2)
```
xewma.ad

Compute steady-state ARLs of EWMA control charts

Description

Computation of the steady-state Average Run Length (ARL) for different types of EWMA control charts monitoring normal mean.

Usage

xewma.ad(l, c, mu1, mu0=0, zr=0, z0=0, sided="one", limits="fix", steady.state.mode="conditional", r=40)

Arguments

l
  smoothing parameter lambda of the EWMA control chart.
  c
  critical value (similar to alarm limit) of the EWMA control chart.
  mu1
  out-of-control mean.
  mu0
  in-control mean.
  zr
  reflection border for the one-sided chart.
  z0
  restarting value of the EWMA sequence in case of a false alarm in steady.state.mode="cyclical".
  sided
distinguishes between one- and two-sided two-sided EWMA control chart by choosing "one" and "two", respectively.
  limits
distinguishes between different control limits behavior.
  steady.state.mode
  distinguishes between two steady-state modes – conditional and cyclical.
  r
  number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-sided) or r (two-sided).

Details

xewma.ad determines the steady-state Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature and using the power method for deriving the largest in magnitude eigenvalue and the related left eigenfunction.

Value

Returns a single value which resembles the steady-state ARL.
Author(s)

Sven Knoth

References


See Also

`xewma.arl` for zero-state ARL computation and `xcusum.ad` for the steady-state ARL of CUSUM control charts.

Examples

```r
## comparison of zero-state (= worst case) and steady-state performance
## for two-sided EWMA control charts

l <- .1
l <- xewma.crit(l, 1500, sided="two")
mu <- c(0, .5, 1, 1.5, 2)
arl <- sapply(mu, l=l, c=c, sided="two", xewma.arl)
ad <- sapply(mu, l=l, c=c, sided="two", xewma.ad)
round(cbind(mu, arl, ad), digits=2)

## Lucas/Saccucci (1990)
## two-sided EWMA

## with fixed limits
l1 <- .5
l2 <- .03
c1 <- 3.071
w <- xewma.crit(l1, 1500, sided="two")
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,3.5,4,5)
am1 <- sapply(mu, l=l1, c=c1, sided="two", xewma.ad)
am2 <- sapply(mu, l=l2, c=c2, sided="two", xewma.ad)
round(cbind(mu, a1, a2), digits=2)

## original results are (in Table 3)
##  0.80 499. 480.
##  0.25 254.  74.1
##  0.50  88.4 28.6
##  0.75  35.7 17.3
##  1.00 17.3 12.5
##  1.50  6.44  8.00
##  2.00  3.58  5.95
##  2.50  2.47  4.78
```
## Description

Computation of the (zero-state) Average Run Length (ARL) for different types of EWMA control charts monitoring normal mean.

## Usage

```r
xewma.arl(l, c, mu, zr=0, hs=0, sided="one", limits="fix", q=1,
steady.state.mode="conditional", r=40)
```

## Arguments

- `l`: smoothing parameter lambda of the EWMA control chart.
- `c`: critical value (similar to alarm limit) of the EWMA control chart.
- `mu`: true mean.
- `zr`: reflection border for the one-sided chart.
- `hs`: so-called headstart (enables fast initial response).
- `sided`: distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
- `limits`: distinguishes between different control limits behavior.
- `q`: change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq q) \), will be determined. Note that \( \mu_0=0 \) is implicitly fixed.
- `steady.state.mode`: distinguishes between two steady-state modes – conditional and cyclical (needed for \( q>1 \)).
- `r`: number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-sided) or \( r \) (two-sided).

## Details

In case of the EWMA chart with fixed control limits, `xewma.arl` determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature. If `limits` is not "fix", then the method presented in Knoth (2003) is utilized. Note that for one-sided EWMA charts (sided="one"), only "vacl" and "stat" are deployed, while for two-sided ones (sided="two") also "fir", "both" (combination of "fir" and "vacl"), and "Steiner" are implemented. For details see Knoth (2004).
Value

Except for the fixed limits EWMA charts it returns a single value which resembles the ARL. For fixed limits charts, it returns a vector of length $q$ which resembles the ARL and the sequence of conditional expected delays for $q=1$ and $q>1$, respectively.

Author(s)

Sven Knoth

References


See Also

*xcusum.arl* for zero-state ARL computation of CUSUM control charts and *xewma.ad* for the steady-state ARL.

Examples

```r
## Waldmann (1986), one-sided EWMA
1 <- .75
round(xewma.arl(1,2*sqrt((2-1)/1),0,zr=-4*sqrt((2-1)/1)),digits=1)
1 <- .5
round(xewma.arl(1,2*sqrt((2-1)/1),0,zr=-4*sqrt((2-1)/1)),digits=1)
## original values are 209.3 and 3907.5 (in Table 2).

## Waldmann (1986), two-sided EWMA with fixed control limits
1 <- .75
round(xewma.arl(1,2*sqrt((2-1)/1),0,sided="two"),digits=1)
1 <- .5
round(xewma.arl(1,2*sqrt((2-1)/1),0,sided="two"),digits=1)
## original values are 104.0 and 1952 (in Table 1).
```
## Crowder (1987), two-sided EWMA with fixed control limits

```r
l1 <- .5
l2 <- .05
c <- 2
mu <- (0:16)/4
ar1 <- sapply(mu, l=l1, c=c, sided="two", xewma.arl)
ar2 <- sapply(mu, l=l2, c=c, sided="two", xewma.arl)
round(cbind(mu, ar1, ar2), digits=2)
```

## original results are (in Table 1)

```
#  0.00  26.45 127.53
#  0.25  20.12  43.94
#  0.50  11.89  18.97
#  0.75   7.29  11.64
#  1.00   4.91   8.38
#  1.25   3.95   6.56
#  1.50   2.80   5.41
#  1.75   2.29   4.62
#  2.00   1.94   4.04
#  2.25   1.70   3.61
#  2.50   1.51   3.26
#  2.75   1.37   2.99
#  3.00   1.26   2.76
#  3.25   1.18   2.56
#  3.50   1.12   2.39
#  3.75   1.08   2.26
#  4.00   1.05   2.15 (* -- in Crowder (1987) typo!?)
```

## Lucas/Saccucci (1990)

## two-sided EWMA

## with fixed limits

```r
l1 <- .5
l2 <- .03
c1 <- 3.071
c2 <- 2.437
mu <- c(0.25, 0.5, 0.75, 1.5, 2.5, 3.5, 4.5)
ar1 <- sapply(mu, l=l1, c=c1, sided="two", xewma.arl)
ar2 <- sapply(mu, l=l2, c=c2, sided="two", xewma.arl)
round(cbind(mu, ar1, ar2), digits=2)
```

## original results are (in Table 3)

```
#  0.00   500.   500.
#  0.25  255.    76.7
#  0.50  88.8  29.3
#  0.75  35.9  17.6
#  1.00  17.5  12.6
#  1.50   6.53  8.07
#  2.00   3.63  5.99
#  2.50   2.50  4.80
#  3.00   1.93  4.03
#  3.50   1.58  3.49
```
### 4.00 1.34 3.11
### 5.00 1.07 2.55

## Not run:
## with fir feature
l1 <- .5
l2 <- .03
c1 <- 3.071
c2 <- 2.437
hs1 <- c1/2
hs2 <- c2/2
mu <- c(0,.5,1,2,3,5)
arl1 <- sapply(mu,l=l1,c=c1,hs=hs1,sided="two",limits="fir",xewma.arl)
arl2 <- sapply(mu,l=l2,c=c2,hs=hs2,sided="two",limits="fir",xewma.arl)
round(cbind(mu,arl1,arl2),digits=2)

## original results are in Table 5
## 0.0 487. 406.
## 0.5 86.1 18.4
## 1.0 15.9 7.36
## 2.0 2.87 3.43
## 3.0 1.45 2.34
## 5.0 1.01 1.57

## Chandrasekaran, English, Disney (1995)
## two-sided EWMA with fixed and variance adjusted limits (vacl)

l1 <- .25
l2 <- .1
c1s <- 2.9985
c1n <- 3.0042
c2s <- 2.8159
c2n <- 2.8452
mu <- c(0,.25,.5,.75,1,2)
arl1s <- sapply(mu,l=l1,c=c1s,sided="two",xewma.arl)
arl1n <- sapply(mu,l=l1,c=c1n,sided="two",limits="vacl",xewma.arl)
arl2s <- sapply(mu,l=l2,c=c2s,sided="two",xewma.arl)
arl2n <- sapply(mu,l=l2,c=c2n,sided="two",limits="vacl",xewma.arl)
round(cbind(mu,arl1s,arl1n,arl2s,arl2n),digits=2)

## original results are in Table 2
## 0.0 500. 500. 500. 500.
## 0.25 179.09 167.54 105.90 96.6
## 0.5 48.14 45.65 31.08 24.35
## 0.75 20.02 19.72 15.71 10.74
## 1.00 11.07 9.37 10.23 6.35
## 2.00 3.59 2.64 4.32 2.73

## The results in Chandrasekaran, English, Disney (1995) are not
## that accurate. Let us consider the more appropriate comparison

c1s <- xewma.crit(l1,500,sided="two")
c1n <- xewma.crit(l1,500,sided="two",limits="vacl")
c2s <- xewma.crit(l2,500,sided="two")
c2n <- xewma.crit(l2,500,sided="two",limits="vacl")
mu <- c(0,.25,.5,.75,1.2)
arls1 <- sapply(mu,l=1,c=c1s,sided="two",xewma.arl)
arln1 <- sapply(mu,l=1,c=c1n,sided="two",limits="vacl",xewma.arl)
arls2 <- sapply(mu,l=1,c=c2s,sided="two",xewma.arl)
arln2 <- sapply(mu,l=1,c=c2n,sided="two",limits="vacl",xewma.arl)
round(cbind(mu,arls1,arln1,arls2,arln2),digits=2)

## which demonstrate the abilities of the variance-adjusted limits
## scheme more explicitely.

## Rhoads, Montgomery, Mastrangelo (1996)
## two-sided EWMA with fixed and variance adjusted limits (vacl),
## with fir and both features

l <- .03
c <- 2.437
mu <- c(0,.5,1,1.5,2,3,4)
sl <- sqrt(l*(1-l))
arlfix <- sapply(mu,l=1,c=c,sided="two",xewma.arl)
arlvacl <- sapply(mu,l=1,c=c,sided="two",limits="vacl",xewma.arl)
arlfir <- sapply(mu,l=1,c=c,hs=c/2,sided="two",limits="fir",xewma.arl)
arlboth <- sapply(mu,l=1,c=c,hs=c/2*sl,sided="two",limits="both",xewma.arl)
round(cbind(mu,arlfix,arlvacl,arlfir,arlboth),digits=1)

## original results are (in Table 1)
## 0 0.0 477.3 427.9 383.4 286.2
## 0.5 29.7 20.0 18.6 12.8
## 1.0 12.5 6.5 7.4 3.6
## 1.5 8.1 3.3 4.6 1.9
## 2.0 6.0 2.2 3.4 1.4
## 3.0 4.0 1.3 2.4 1.0
## 4.0 3.1 1.1 1.9 1.0
## * -- the in-control values differ sustainably from the true values!

## Steiner (1999)
## two-sided EWMA control charts with various modifications

## fixed vs. variance adjusted limits

l <- .05
c <- 3
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,3.5,4)
arlfix <- sapply(mu,l=1,c=c,sided="two",xewma.arl)
arlvacl <- sapply(mu,l=1,c=c,sided="two",limits="vacl",xewma.arl)
round(cbind(mu,arlfix,arlvacl),digits=1)

## original results are (in Table 2)
## 0.0 0.0 1379.0 1353.0
## 0.25 0.25 135.0 127.0
## 0.50 0.50 37.4 32.5
## 0.75 0.75 20.0 15.6
## 1.00  13.5  9.0
## 1.50  8.3  4.5
## 2.00  6.0  2.8
## 2.50  4.8  2.0
## 3.00  4.0  1.6
## 3.50  3.4  1.3
## 4.00  3.0  1.1

## fir, both, and Steiner's modification

`l <- 0.03`
cfir <- 2.44
cboth <- 2.54
cstein <- 2.55
hsfir <- cfir/2
hsboth <- cboth/2+sqrt(l*(l-1))
u <- c(0.5,1.5,2,3,4)
arlfir <- sapply(u,l=1,c=cfir,hs=hsfir,sided="two",limits="fir",xewma.arl)
arlboth <- sapply(u,l=1,c=cboth,hs=hsboth,sided="two",limits="both",xewma.arl)
arlstein <- sapply(u,l=1,c=cstein,sided="two",limits="Steiner",xewma.arl)
round(cbind(mu,arlfir,arlbboth,arlstein),digits=3)

## original values are (in Table 5)
## 0.0 383.0 384.0 391.0
## 0.5 18.6 14.9 13.8
## 1.0 7.4 3.9 3.6
## 1.5 4.6 2.0 1.8
## 2.0 3.4 1.4 1.3
## 3.0 2.4 1.1 1.0
## 4.0 1.9 1.0 1.0

## SAS/QC manual 1999
## two-sided EWMA control charts with fixed limits

`l <- 0.25`
c <- 3
mu <- 1
print(xewma.arl(l,c,mu,sided="two"),digits=11)

# original value is 11.154267016.

## Some recent examples for one-sided EWMA charts
## with varying limits and in the so-called stationary mode

# 1. varying limits = "vacl"

`lambda <- .1`
L0 <- 500

## Monte Carlo results (10^9 replicates)
# mu   ARL   s.e.
# 0   500.00 0.0160
# 0.5  21.637 0.0006
xewma.arl.f

```r
# 1  6.7596  0.0001
# 1.5 3.5398  0.0001
# 2  2.3038  0.0000
# 2.5 1.7004  0.0000
# 3  1.3675  0.0000

zr <- -6
r <- 50
c <- xewma.crit(lambda, L0, zr=zr, limits="vacl", r=r)
Mxewma.arl <- Vectorize(xewma.arl, "mu")
mus <- (0:6)/2
arls <- round(Mxewma.arl(lambda, c, mus, zr=zr, limits="vacl", r=r), digits=4)
data.frame(mus, arls)

# 2. stationary mode, i.e. limits = "stat"

## Monte Carlo results (10^9 replicates)
# mu ARL s.e.
# 0  500.00  0.0159
# 0.5 22.313  0.0006
# 1  7.2920  0.0001
# 1.5 3.9064  0.0001
# 2  2.5131  0.0000
# 2.5 1.7983  0.0000
# 3  1.4029  0.0000
c <- xewma.crit(lambda, L0, zr=zr, limits="stat", r=r)
arls <- round(Mxewma.arl(lambda, c, mus, zr=zr, limits="stat", r=r), digits=4)
data.frame(mus, arls)

## End(Not run)
```

---

**xewma.arl.f**

*Compute ARL function of EWMA control charts*

**Description**

Computation of the (zero-state) Average Run Length (ARL) function for different types of EWMA control charts monitoring normal mean.

**Usage**

`xewma.arl.f(1, c, mu, zr=0, sided="one", limits="fix", r=40)`

**Arguments**

- **1** smoothing parameter lambda of the EWMA control chart.
- **c** critical value (similar to alarm limit) of the EWMA control chart.
- **mu** true mean.
zr  

reflection border for the one-sided chart.

sided  
distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.

limits  
distinguishes between different control limits behavior.

r  
number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-sided) or r (two-sided).

Details

It is a convenience function to yield the ARL as function of the head start hs. For more details see xewma.arl.

Value

It returns a function of a single argument, hs=x which maps the head-start value hs to the ARL.

Author(s)

Sven Knoth

References


See Also

xewma.arl for zero-state ARL for one specific head-start hs.

Examples

# will follow
Usage

`xewma.arl.prerun(l, c, mu, zr=0, hs=0, sided="two", limits="fix", q=1, size=100, df=NULL, estimated="mu", qm.mu=30, qm.sigma=30, truncate=1e-10)`

`xewma.crit.prerun(l, L0, mu, zr=0, hs=0, sided="two", limits="fix", size=100, df=NULL, estimated="mu", qm.mu=30, qm.sigma=30, truncate=1e-10, c.error=1e-12, L.error=1e-9, OUTPUT=FALSE)`

Arguments

- **l** smoothing parameter lambda of the EWMA control chart.
- **c** critical value (similar to alarm limit) of the EWMA control chart.
- **mu** true mean shift.
- **zr** reflection border for the one-sided chart.
- **hs** so-called headstart (give fast initial response).
- **sided** distinguish between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
- **limits** distinguish between different control limits behavior.
- **q** change point position. For $q = 1$ and $\mu = \mu_0$ and $\mu = \mu_1$, the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For $q > 1$ and $\mu_1 = 0$ conditional delays, that is, $E_q(L - q + 1|L \geq q)$, will be determined. Note that $\mu_0=0$ is implicitly fixed.
- **size** pre run sample size.
- **df** Degrees of freedom of the pre run variance estimator. Typically it is simply size - 1. If the pre run is collected in batches, then also other values are needed.
- **estimated** name the parameter to be estimated within the "mu", "sigma", "both".
- **qm.mu** number of quadrature nodes for convoluting the mean uncertainty.
- **qm.sigma** number of quadrature nodes for convoluting the standard deviation uncertainty.
- **truncate** size of truncated tail.
- **L0** in-control ARL.
- **c.error** error bound for two succeeding values of the critical value during applying the secant rule.
- **L.error** error bound for the ARL level $L0$ during applying the secant rule.
- **OUTPUT** activate or deactivate additional output.

Details

Essentially, the ARL function `xewma.arl` is convoluted with the distribution of the sample mean, standard deviation or both. For details see Jones/Champ/Rigdon (2001) and Knoth (2014?).

Value

Returns a single value which resembles the ARL.
Author(s)
Sven Knoth

References
S. Knoth (2014?), tbd, tbd, tbd-tbd.

See Also
xewma.arl for the usual zero-state ARL computation.

Examples
```r
## Jones/Champ/Rigdon (2001)
c4m <- function(m, n) sqrt(2)*gamma( (m*(n-1)+1)/2 )/sqrt( m*(n-1) )/gamma( m*(n-1)/2 )

n <- 5 # sample size
m <- 20 # pre run with 20 samples of size n = 5
C4m <- c4m(m, n) # needed for bias correction

# Table 1, 3rd column
lambda <- 0.2
L <- 2.636

xewma.ARL <- Vectorize("xewma.arl", "mu")
xewma.ARL.prerun <- Vectorize("xewma.arl.prerun", "mu")

mu <- c(0, .25, .5, 1, 1.5, 2)
ARL <- round(xewma.ARL(lambda, L, mu, sided="two"), digits=2)
p.ARL <- round(xewma.ARL.prerun(lambda, L/C4m, mu, sided="two", size=m, df=m*(n-1), estimated="both", qm.mu=70), digits=2)

# Monte-Carlo with 10^8 repetitions: 200.325 (0.020) and 144.458 (0.022)
cbind(mu, ARL, p.ARL)

## Not run:
# Figure 5, subfigure r = 0.2
mu_ <- (0.85)/40
ARL_ <- round(xewma.ARL(lambda, L, mu_, sided="two"), digits=2)
p.ARL_ <- round(xewma.ARL.prerun(lambda, L/C4m, mu_, sided="two", size=m, df=m*(n-1), estimated="both"), digits=2)

plot(mu_, ARL_, type="l", xlab=expression(delta), ylab="ARL", xlim=c(0,2))
```

xewma.crit

Compute critical values of EWMA control charts

Description

Computation of the critical values (similar to alarm limits) for different types of EWMA control charts monitoring normal mean.

Usage

xewma.crit(l, L0, mu0 = 0, zr = 0, hs = 0, sided = "one", limits = "fix", r = 40, c0 = NULL)

Arguments

- `l`: smoothing parameter lambda of the EWMA control chart.
- `L0`: in-control ARL.
- `mu0`: in-control mean.
- `zr`: reflection border for the one-sided chart.
- `hs`: so-called headstart (enables fast initial response).
- `sided`: distinguishes between one- and two-sided two-sided EWMA control chart by choosing "one" and "two", respectively.
- `limits`: distinguishes between different control limits behavior.
- `r`: number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-sided) or r (two-sided).
- `c0`: starting value for iteration rule.

Details

xewma.crit determines the critical values (similar to alarm limits) for given in-control ARL L0 by applying secant rule and using xewma.arl().

Value

Returns a single value which resembles the critical value c.

Author(s)

Sven Knoth
References


See Also

*xewma.arl* for zero-state ARL computation.

Examples

```r
l <- .1
incontrolARL <- c(500,5000,50000)
sapply(incontrolARL,l=1,sided="two",xewma.crit,r=35) # accuracy with 35 nodes
sapply(incontrolARL,l=1,sided="two",xewma.crit) # accuracy with 40 nodes
sapply(incontrolARL,l=1,sided="two",xewma.crit,r=50) # accuracy with 50 nodes

## Crowder (1989)
## two-sided EWMA control charts with fixed limits

l <- c(.05,.1,.15,.2,.25)
L0 <- 250
round(sapply(l,L0=L0,sided="two",xewma.crit),digits=2)

## original values are 2.32, 2.55, 2.65, 2.72, and 2.76.
```

**xewma.q**

*Compute RL quantiles of EWMA control charts*

Description

Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal mean.

Usage

```r
xewma.q(l, c, mu, alpha, zr=0, hs=0, sided="two", limits="fix", q=1, r=40)

xewma.q.crit(l, L0, mu, alpha, zr=0, hs=0, sided="two", limits="fix", r=40, c.error=1e-12, a.error=1e-9, OUTPUT=FALSE)
```

Arguments

- `l`: smoothing parameter lambda of the EWMA control chart.
- `c`: critical value (similar to alarm limit) of the EWMA control chart.
- `mu`: true mean.
- `alpha`: quantile level.
- `zr`: reflection border for the one-sided chart.
so-called headstart (enables fast initial response).

distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.

distinguishes between different control limits behavior.

change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq) \), will be determined. Note that \( \mu_0=0 \) is implicitly fixed.

number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-sided) or \( r \) (two-sided).

in-control quantile value.

error bound for two succeeding values of the critical value during applying the secant rule.

error bound for the quantile level \( \alpha \) during applying the secant rule.

activate or deactivate additional output.

Instead of the popular ARL (Average Run Length) quantiles of the EWMA stopping time (Run Length) are determined. The algorithm is based on Waldmann’s survival function iteration procedure. If \( \text{limits} \) is not "fix", then the method presented in Knoth (2003) is utilized. Note that for one-sided EWMA charts (\( \text{sided} = \text{"one"} \)), only "vacl" and "stat" are deployed, while for two-sided ones (\( \text{sided} = \text{"two"} \)) also "fir", "both" (combination of "fir" and "vacl"), and "Steiner" are implemented. For details see Knoth (2004).

Returns a single value which resembles the RL quantile of order \( q \).

Sven Knoth


See Also

xewma.arl for zero-state ARL computation of EWMA control charts.

Examples

```r
## Gan (1993), two-sided EWMA with fixed control limits
## some values of his Table 1 -- any median RL should be 500
XEWMA.Q <- Vectorize("xewma.q", c("l", "c"))
G.lambda <- c(.05, .1, .15, .2, .25)
G.h <- c(.441, .675, .863, 1.027, 1.177)
MEDIAN <- ceiling(XEWMA.Q(G.lambda, G.h/sqrt(G.lambda/(2-G.lambda)),
0, .5, sided="two")
print(cbind(G.lambda, MEDIAN))

## increase accuracy of thresholds

# (i) calculate threshold for given in-control median value by
# deploying secant rule
XEWMA.q.crit <- Vectorize("xewma.q.crit", "l")

# (ii) re-calculate the thresholds and remove the standardization step
L0 <- 500
G.h.new <- XEWMA.q.crit(G.lambda, L0, 0, .5, sided="two")
G.h.new <- round(G.h.new * sqrt(G.lambda/(2-G.lambda)), digits=5)

# (iii) compare Gan's original values and the new ones with 5 digits
print(cbind(G.lambda, G.h.new, G.h))

# (iv) calculate the new medians
MEDIAN <- ceiling(XEWMA.Q(G.lambda, G.h.new/sqrt(G.lambda/(2-G.lambda)),
0, .5, sided="two")
print(cbind(G.lambda, MEDIAN))
```

**xewma.q.prerun**

Compute RL quantiles of EWMA control charts in case of estimated parameters

Description

Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal mean if the in-control mean, standard deviation, or both are estimated by a pre run.

Usage

```r
xewma.q.prerun(l, c, mu, p, zr=0, hs=0, sided="two", limits="fix", q=1, size=100,
df=NULL, estimated="mu", qm.mu=30, qm.sigma=30, truncate=1e-10, bound=1e-10)

xewma.q.crit.prerun(l, L0, mu, p, zr=0, hs=0, sided="two", limits="fix", size=100,
df=NULL, estimated="mu", qm.mu=30, qm.sigma=30, truncate=1e-10, bound=1e-10,
c.error=1e-10, p.error=1e-9, OUTPUT=FALSE)
```
Arguments

1  smoothing parameter lambda of the EWMA control chart.
c  critical value (similar to alarm limit) of the EWMA control chart.
mu  true mean shift.
p  quantile level.
zr  reflection border for the one-sided chart.
hs  so-called headstart (give fast initial response).
sided  distinguish between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
limits  distinguish between different control limits behavior.
q  change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq) \), will be determined. Note that mu0=0 is implicitly fixed.
size  pre run sample size.
df  Degrees of freedom of the pre run variance estimator. Typically it is simply size - 1. If the pre run is collected in batches, then also other values are needed.
estimated  name the parameter to be estimated within the "mu", "sigma", "both".
qm.mu  number of quadrature nodes for convoluting the mean uncertainty.
qm.sigma  number of quadrature nodes for convoluting the standard deviation uncertainty.
truncate  size of truncated tail.
bound  control when the geometric tail kicks in; the larger the quicker and less accurate; bound should be larger than 0 and less than 0.001.
L0  in-control quantile value.
c.error  error bound for two succeeding values of the critical value during applying the secant rule.
p.error  error bound for the quantile level p during applying the secant rule.
OUTPUT  activate or deactivate additional output.

Details

Essentially, the ARL function xewma.q is convoluted with the distribution of the sample mean, standard deviation or both. For details see Jones/Champ/Rigdon (2001) and Knoth (2014?).

Value

Returns a single value which resembles the RL quantile of order q.

Author(s)

Sven Knoth
xewma.q.prerun

References


See Also

*xewma.q* for the usual RL quantiles computation of EWMA control charts.

Examples

## Jones/Champ/Rigdon (2001)

```r
c4m <- function(m, n) sqrt(2)*gamma((m*(n-1)+1)/2)/sqrt( m*(n-1) )/gamma( m*(n-1)/2 )

n <- 5 # sample size
m <- 20 # pre run with 20 samples of size n = 5
C4m <- c4m(m, n) # needed for bias correction

# Table 1, 3rd column
lambda <- 0.2
L <- 2.636

xewma.Q <- Vectorize("xewma.q", "mu")
xewma.Q.prerun <- Vectorize("xewma.q.prerun", "mu")

mu <- c(0, .25, .5, 1, 1.5, 2)
Q1 <- ceiling(xewma.Q(lambda, L, mu, 0.1, sided="two"))
Q2 <- ceiling(xewma.Q(lambda, L, mu, 0.5, sided="two"))
Q3 <- ceiling(xewma.Q(lambda, L, mu, 0.9, sided="two"))

cbind(mu, Q1, Q2, Q3)

## Not run:
p.Q1 <- xewma.Q.prerun(lambda, L/C4m, mu, 0.1, sided="two", size=m, df=m*(n-1), estimated="both")
p.Q2 <- xewma.Q.prerun(lambda, L/C4m, mu, 0.5, sided="two", size=m, df=m*(n-1), estimated="both")
p.Q3 <- xewma.Q.prerun(lambda, L/C4m, mu, 0.9, sided="two", size=m, df=m*(n-1), estimated="both")

cbind(mu, p.Q1, p.Q2, p.Q3)
## End(Not run)
```
xewma.sf

Compute the survival function of EWMA run length

Description

Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring normal mean.

Usage

xewma.sf(1, c, mu, n, zr=0, hs=0, sided="one", limits="fix", q=1, r=40)

Arguments

1      smoothing parameter lambda of the EWMA control chart.
c      critical value (similar to alarm limit) of the EWMA control chart.
mu     true mean.
n      calculate sf up to value n.
zr     reflection border for the one-sided chart.
hs     so-called headstart (enables fast initial response).
sided distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
limits distinguishes between different control limits behavior.
q      change point position. For $q = 1$ and $\mu = \mu_0$ and $\mu = \mu_1$, the usual zero-state situation for the in-control and out-of-control case, respectively, are calculated. Note that mu0=0 is implicitly fixed.
r      number of quadrature nodes, dimension of the resulting linear equation system is equal to $r+1$ (one-sided) or $r$ (two-sided).
Details

The survival function \( P(L>n) \) and derived from it also the cdf \( P(L<=n) \) and the pmf \( P(L=n) \) illustrate the distribution of the EWMA run length. For large \( n \) the geometric tail could be exploited. That is, with reasonable large \( n \) the complete distribution is characterized. The algorithm is based on Waldmann’s survival function iteration procedure. For varying limits and for change points after 1 the algorithm from Knoth (2004) is applied. Note that for one-sided EWMA charts (\( \text{sided} = \text{“one”} \)), only "vacl" and "stat" are deployed, while for two-sided ones (\( \text{sided} = \text{“two”} \)) also "fir", "both" (combination of "fir" and "vacl"), and "Steiner" are implemented. For details see Knoth (2004).

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also

*xewma.arl* for zero-state ARL computation of EWMA control charts.

Examples

```r
## Gan (1993), two-sided EWMA with fixed control limits
## some values of his Table 1 -- any median RL should be 500

G.lambda <- c(.05, .1, .15, .2, .25)
G.h <- c(.441, .675, .863, 1.027, 1.177)/sqrt(G.lambda/(2-G.lambda))

for ( i in 1:length(G.lambda) ) {
  SF <- xewma.sf(G.lambda[i], G.h[i], 0, 1000)
  if (i==1) plot(1:length(SF), SF, type="l", xlab="n", ylab="P(L>n)")
  else lines(1:length(SF), SF, col=i)
}
```
Compute the survival function of EWMA run length in case of estimated parameters

Description

Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring normal mean if the in-control mean, standard deviation, or both are estimated by a pre run.

Usage

```r
xewma.sf.prerun(l, c, mu, n, zr=0, hs=0, sided="one", limits="fix", q=1, size=100, df=NULL, estimated="mu", qm.mu=30, qm.sigma=30, truncate=1e-10, tail_approx=TRUE, bound=1e-10)
```

Arguments

- `l`: smoothing parameter lambda of the EWMA control chart.
- `c`: critical value (similar to alarm limit) of the EWMA control chart.
- `mu`: true mean.
- `n`: calculate sf up to value n.
- `zr`: reflection border for the one-sided chart.
- `hs`: so-called headstart (give fast initial response).
- `sided`: distinguish between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
- `limits`: distinguish between different control limits behavior.
- `q`: change point position. For $q = 1$ and $\mu = \mu_0$ and $\mu = \mu_1$, the usual zero-state situation for the in-control and out-of-control case, respectively, are calculated. Note that mu0=0 is implicitly fixed.
- `size`: pre run sample size.
- `df`: degrees of freedom of the pre run variance estimator. Typically it is simply size - 1. If the pre run is collected in batches, then also other values are needed.
- `estimated`: name the parameter to be estimated within the "mu", "sigma", "both".
- `qm.mu`: number of quadrature nodes for convoluting the mean uncertainty.
- `qm.sigma`: number of quadrature nodes for convoluting the standard deviation uncertainty.
- `truncate`: size of truncated tail.
- `tail_approx`: Controls whether the geometric tail approximation is used (is faster) or not.
- `bound`: control when the geometric tail kicks in; the larger the quicker and less accurate; bound should be larger than 0 and less than 0.001.
Details

The survival function $P(L > n)$ and derived from it also the cdf $P(L \leq n)$ and the pmf $P(L = n)$ illustrate the distribution of the EWMA run length...

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also

`xewma.sf.prerun` for the RL survival function of EWMA control charts w/o pre run uncertainty.

Examples

```r
# Jones/Champ/Rigdon (2001)
c4m <- function(m, n) sqrt(2)*gamma( (m*(n-1)+1)/2 )/sqrt( m*(n-1) )/gamma( m*(n-1)/2 )

n <- 5 # sample size

# Figure 6, subfigure r=0.1
lambda <- 0.1
L <- 2.454

CDF0 <- 1 - xewma.sf(lambda, L, 0, 600, sided="two")
m <- 10 # pre run size

CDF1 <- 1 - xewma.sf.prerun(lambda, L/c4m(m,n), 0, 600, sided="two", size=m, df=m*(n-1), estimated="both")
m <- 20

CDF2 <- 1 - xewma.sf.prerun(lambda, L/c4m(m,n), 0, 600, sided="two", size=m, df=m*(n-1), estimated="both")
m <- 50
```
CDF3 <- 1 - xewma.sf.prerun(lambda, L/c4m(m,n), 0, 600, sided="two", size=m, df=m*(n-1), estimated="both")

plot(CDF0, type="l", xlab="t", ylab=expression(P(T<=t)), xlim=c(0,500), ylim=c(0,1))
abline(v=0, h=c(0,1), col="grey", lwd=.7)
points((1:5)*100, CDF0[(1:5)*100], pch=18)
lines(CDF1, col="blue")
points((1:5)*100, CDF1[(1:5)*100], pch=2, col="blue")
lines(CDF2, col="red")
points((1:5)*100, CDF2[(1:5)*100], pch=16, col="red")
lines(CDF3, col="green")
points((1:5)*100, CDF3[(1:5)*100], pch=5, col="green")

legend("bottomright", c("Known", "m=10, n=5", "m=20, n=5", "m=50, n=5"),
col=c("black", "blue", "red", "green"), pch=c(18, 2, 16, 5), lty=1)

---

xgrsr.ad  
Compute steady-state ARLs of Shiryaev-Roberts schemes

Description

Computation of the steady-state Average Run Length (ARL) for Shiryaev-Roberts schemes monitoring normal mean.

Usage

xgrsr.ad(k, g, muQ, muP = 0, zr = 0, sided = "one", MPT = FALSE, r = 30)

Arguments

k  
reference value of the Shiryaev-Roberts scheme.

g  
control limit (alarm threshold) of Shiryaev-Roberts scheme.

muQ  
out-of-control mean.

muP  
in-control mean.

zr  
reflection border to enable the numerical algorithms used here.

sided  
distinguishes between one- and two-sided schemes by choosing "one" and "two", respectively. Currently only one-sided schemes are implemented.

MPT  
switch between the old implementation (FALSE) and the new one (TRUE) that considers the completed likelihood ratio. MPT contains the initials of G. Moustakides, A. Polunchenko and A. Tartakovsky.

r  
number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1.

Details

xgrsr.ad determines the steady-state Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature.
Value

Returns a single value which resembles the steady-state ARL.

Author(s)

Sven Knoth

References


See Also

*xewma.arl* and *xcusum.arl* for zero-state ARL computation of EWMA and CUSUM control charts, respectively, and *xgrsr.arl* for the zero-state ARL.

Examples

```r
## Small study to identify appropriate reflection border to mimic unreflected schemes
k <- .5
g <- log(390)
zsrs <- -(0:10)
ZRxgrsr.ad <- Vectorize(xgrsr.ad, "zr")
ads <- ZRxgrsr.ad(k, g, 0, zr=zsrs)
data.frame(zsrs, ads)

## Table 2 from Knoth (2006)
## original values are
# mu  arl
# 0   689
# 0.5 38
# 1   9
# 1.5 5.1
# 2   3.6
# 2.5 2.8
# 3   2.4
#
k <- .5
g <- log(390)
zr <- -5 # see first example
mus <- (0:6)/2
Mxgrsr.ad <- Vectorize(xgrsr.ad, "mu1")
ads <- round(Mxgrsr.ad(k, g, mus, zr=zr), digits=1)
data.frame(mus, ads)

## Table 4 from Moustakides et al. (2009)
```
## xgrsr.arl

** Compute (zero-state) ARLs of Shiryaev-Roberts schemes **

### Description

Computation of the (zero-state) Average Run Length (ARL) for Shiryaev-Roberts schemes monitoring normal mean.

### Usage

```
xgrsr.arl(k, g, mu, zr = 0, hs=NULL, sided = "one", q = 1, MPT = FALSE, r = 30)
```

### Arguments

- **k**: reference value of the Shiryaev-Roberts scheme.
- **g**: control limit (alarm threshold) of Shiryaev-Roberts scheme.
- **mu**: true mean.
- **zr**: reflection border to enable the numerical algorithms used here.
- **hs**: so-called headstart (enables fast initial response). If hs=NULL, then the classical headstart -Inf is used (corresponds to 0 for the non-log scheme).
- **sided**: distinguishes between one- and two-sided schemes by choosing "one" and "two", respectively. Currently only one-sided schemes are implemented.
- **q**: change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq q) \), will be determined. Note that mu0=0 is implicitly fixed.
- **MPT**: switch between the old implementation (FALSE) and the new one (TRUE) that considers the complete likelihood ratio. MPT stands for the initials of G. Moustakides, A. Polunchenko and A. Tartakovsky.
- **r**: number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \).
Details

xgrsr.arl determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature.

Value

Returns a vector of length q which resembles the ARL and the sequence of conditional expected delays for q=1 and q>1, respectively.

Author(s)

Sven Knoth

References


See Also

xewma.arl and xcusum.arl for zero-state ARL computation of EWMA and CUSUM control charts, respectively, and xgrsr.ad for the steady-state ARL.

Examples

```r
## Small study to identify appropriate reflection border to mimic unreflected schemes
k <- .5
g <- log(390)
zs <- -(0:10)
ZRxgrsr.arl <- Vectorize(xgrsr.arl, "zr")
arl <- ZRxgrsr.arl(k, g, zs)
data.frame(zs, arl)

## Table 2 from Knoth (2006)
## original values are
# mu arl
# 0 697
# 0.5 33
# 1 10.4
# 1.5 6.2
# 2 4.4
# 2.5 3.5
# 3 2.9
#
k <- .5
g <- log(390)
```
zr <- -5 # see first example
mus <- (0:6)/2
Mxgrsr.arl <- Vectorize(xgrsr.arl, "mu")
ars <- round(Mxgrsr.arl(k, g, mus, zr=zr), digits=1)
data.frame(mus, arls)

XGRSR.arl <- Vectorize("xgrsr.arl", "g")
zr <- -6

## Table 2 from Moustakides et al. (2009)
## original values are
# gamma A ARL/E_infty(L) SADD/E_1(L)
# 50 47.17 50.29 41.40
# 100 94.34 100.28 72.32
# 500 471.70 500.28 209.44
# 1000 943.41 1000.28 298.50
# 5000 4717.04 5000.24 557.87
#10000 9434.08 10000.17 684.17

theta <- .1
As2 <- c(47.17, 94.34, 471.7, 943.41, 4717.04, 9434.08)
gs2 <- log(As2)
ars0 <- round(XGRSR.arl(theta/2, gs2, 0, zr=-5, r=300, MPT=TRUE), digits=2)
ars1 <- round(XGRSR.arl(theta/2, gs2, theta, zr=-5, r=300, MPT=TRUE), digits=2)
data.frame(As2, arls0, arls1)

## Table 3 from Moustakides et al. (2009)
## original values are
# gamma A ARL/E_infty(L) SADD/E_1(L)
# 50 37.38 49.45 12.30
# 100 74.76 99.45 16.60
# 500 373.81 499.45 28.05
# 1000 747.62 999.45 33.33
# 5000 3738.08 4999.45 45.96
#10000 7476.15 9999.24 51.49

theta <- .5
As3 <- c(37.38, 74.76, 373.81, 747.62, 3738.08, 7476.15)
gs3 <- log(As3)
ars0 <- round(XGRSR.arl(theta/2, gs3, 0, zr=-5, r=70, MPT=TRUE), digits=2)
ars1 <- round(XGRSR.arl(theta/2, gs3, theta, zr=-5, r=70, MPT=TRUE), digits=2)
data.frame(As3, arls0, arls1)

## Table 4 from Moustakides et al. (2009)
## original values are
# gamma A ARL/E_infty(L) SADD/E_1(L)
# 50 28.02 49.78 4.98
# 100 56.04 99.79 6.22
# 500 280.19 499.79 9.30
# 1000 560.37 999.79 10.66
# 5000 2801.85 5000.93 13.86
#10000 5603.70 9999.87 15.24
theta <- 1
As4 <- c(28.02, 56.04, 280.19, 560.37, 2801.85, 5603.7)
gs4 <- log(As4)
arsl0 <- round(XGSR.arl(theta/2, gs4, 0, zr=-6, r=40, MPT=TRUE), digits=2)
arsl1 <- round(XGSR.arl(theta/2, gs4, theta, zr=-6, r=40, MPT=TRUE), digits=2)
data.frame(As4, ars0, ars1)

---

**xgrsr.crit**

*Compute alarm thresholds for Shiryaev-Roberts schemes*

**Description**

Computation of the alarm thresholds (alarm limits) for Shiryaev-Roberts schemes monitoring normal mean.

**Usage**

```r
gxsr.crit(k, L0, mu0 = 0, zr = 0, hs = NULL, sided = "one", MPT = FALSE, r = 30)
```

**Arguments**

- **k**: reference value of the Shiryaev-Roberts scheme.
- **L0**: in-control ARL.
- **mu0**: in-control mean.
- **zr**: reflection border to enable the numerical algorithms used here.
- **hs**: so-called headstart (enables fast initial response). If hs=NULL, then the classical headstart -Inf is used (corresponds to 0 for the non-log scheme).
- **sided**: distinguishes between one- and two-sided schemes by choosing "one" and "two", respectively. Currently only one-sided schemes are implemented.
- **MPT**: switch between the old implementation (FALSE) and the new one (TRUE) that considers the completed likelihood ratio. MPT contains the initials of G. Moustakides, A. Polunchenko and A. Tartakovsky.
- **r**: number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1.

**Details**

*`xgrsr.crit`* determines the alarm threshold (alarm limit) for given in-control ARL L0 by applying secant rule and using `xgrsr.arl()`.

**Value**

Returns a single value which resembles the alarm limit g.

**Author(s)**

Sven Knoth
References


See Also

`xgrsr.arl` for zero-state ARL computation.

Examples

```r
# Table 4 from Moustakides et al. (2009)
# original values are
# gamma/L0 A/exp(g)
# 50  28.02
# 100 56.04
# 500 280.19
# 1000 560.37
# 5000 2801.75
# 10000 5603.7
theta <- 1
zr <- -6
r <- 100
Lxgrsr.crit <- Vectorize("xgrsr.crit", "L0")
L0s <- c(50, 100, 500, 1000, 5000, 10000)
gs <- LXgrsr.crit(theta/2, L0s, zr=zr, r=r)
data.frame(L0s, gs, A=round(exp(gs), digits=2))
```

---

**Description**

Computation of the (zero-state) Average Run Length (ARL) for different types of simultaneous EWMA control charts (based on the sample mean and the sample variance $S^2$) monitoring normal mean and variance.

**Usage**

```r
xsewma.arl(lx, cx, ls, csu, df, mu, sigma, hsx=0, Nx=40, csl=0, hss=1, Ns=40, s2.on=TRUE, sided="upper", qm=30)
```

**Arguments**

- `lx`: smoothing parameter lambda of the two-sided mean EWMA chart.
- `cx`: control limit of the two-sided mean EWMA control chart.
- `ls`: smoothing parameter lambda of the variance EWMA chart.
upper control limit of the variance EWMA control chart.
df actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
mu true mean.
sigma true standard deviation.
hsx so-called headstart (enables fast initial response) of the mean chart – do not confuse with the true FIR feature considered in xewma.arl; will be updated.
Nx dimension of the approximating matrix of the mean chart.
csl lower control limit of the variance EWMA control chart; default value is 0; not considered if sided is "upper".
hss headstart (enables fast initial response) of the variance chart.
Ns dimension of the approximating matrix of the variance chart.
s2.on distinguishes between $S^2$ and $S$ chart.
sided distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at $c1$ – the actual value of $c1$ is not used), "Rupper" (upper chart with reflection at $c1$), "RLower" (lower chart with reflection at $cu$), and "two" (two-sided chart), respectively.
qm number of quadrature nodes used for the collocation integrals.

Details
xsewma.arl determines the Average Run Length (ARL) by an extension of Gan’s (derived from ideas already published by Waldmann) algorithm. The variance EWMA part is treated similarly to the ARL calculation method deployed for the single variance EWMA charts in Knoth (2005), that is, by means of collocation (Chebyshev polynomials). For more details see Knoth (2007).

Value
Returns a single value which resembles the ARL.

Author(s)
Sven Knoth

References
See Also

xewma.arl and sewma.arl for zero-state ARL computation of single mean and variance EWMA control charts, respectively.

Examples

```r
## Knoth (2007)
## collocation results in Table 1
## Monte Carlo with 10^9 replicates: 252.307 +/- 0.0078

# process parameters
mu <- 0
sigma <- 1
# subgroup size n=5, df=n-1
df <- 4
# lambda of mean chart
lx <-.134
# c_mu = .345476571 = cx/sqrt(n) * sqrt(lx/(2-lx)
CX <- .345476571 + sqrt(df + 1) / sqrt(lx / (2 - lx)
# lambda of variance chart
ls <- .1
# c_sigma = .477977
CSU <- 1 + .477977
# matrix dimensions for mean and variance part
NX <- 25
NS <- 25
# mode of variance chart
SIZED <- "upper"

arl <- xsewma.arl(lx, CX, ls, CSU, df, mu, sigma, NX=NX, NS=NS, sided=SIZED)
arl
```

---

**xsewma.crit**

> Compute critical values of simultaneous EWMA control charts (mean and variance charts)

**Description**

Computation of the critical values (similar to alarm limits) for different types of simultaneous EWMA control charts (based on the sample mean and the sample variance $S^2$) monitoring normal mean and variance.

**Usage**

```r
xsewma.crit(lx, ls, L0, df, mu0=0, sigma0=1, cu=NULL, hsx=0, hss=1, s2.on=TRUE, sided="upper", mode="fixed", Nx=30, Ns=40, qm=30)
```
Arguments

1x  smoothing parameter lambda of the two-sided mean EWMA chart.
1s  smoothing parameter lambda of the variance EWMA chart.
L0  in-control ARL.
mu0  in-control mean.
sigma0  in-control standard deviation.
cu  for two-sided (sided="two") and fixed upper control limit (mode="fixed") a value larger than sigma0 has to been given, for all other cases cu is ignored.
hsx  so-called headstart (enables fast initial response) of the mean chart – do not confuse with the true FIR feature considered in xewma.arl; will be updated.
hss  headstart (enables fast initial response) of the variance chart.
df  actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
s2.on  distinguishes between $S^2$ and $S$ chart.
sided  distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at $c_1$ – the actual value of $c_1$ is not used), "Rupper" (upper chart with reflection at $c_1$), "Rlower" (lower chart with reflection at $cu$), and "two" (two-sided chart), respectively.
mode  only deployed for sided="two" – with "fixed" an upper control limit (see cu) is set and only the lower is determined to obtain the in-control ARL L0, while with "unbiased" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated).
Nx  dimension of the approximating matrix of the mean chart.
Ns  dimension of the approximating matrix of the variance chart.
qm  number of quadrature nodes used for the collocation integrals.

Details

xsewma.crit determines the critical values (similar to alarm limits) for given in-control ARL L0 by applying secant rule and using xsewma.arl(). In case of sided="two" and mode="unbiased" a two-dimensional secant rule is applied that also ensures that the maximum of the ARL function for given standard deviation is attained at sigma0. See Knoth (2007) for details and application.

Value

Returns the critical value of the two-sided mean EWMA chart and the lower and upper controls limit $c_1$ and $cu$ of the variance EWMA chart.

Author(s)

Sven Knoth
References


See Also

xsewma.arl for calculation of ARL of simultaneous EWMA charts.

Examples

```r
## Knoth (2007)
## results in Table 2

# subgroup size n=5, df=n-1
df <- 4
# lambda of mean chart
lx <- .134
# lambda of variance chart
ls <- .1
# in-control ARL
L0 <- 252.3
# matrix dimensions for mean and variance part
Nx <- 25
Ns <- 25
# mode of variance chart
SIDED <- "upper"

crit <- xsewma.crit(lx, ls, L0, df, sided=SIDED, Nx=Nx, Ns=Ns)
crit

## output as used in Knoth (2007)
crit["cx"]/sqrt(df+1)*sqrt(lx/(2-lx))
crit["cu"] - 1
```

---

**xsewma.q**

*Compute critical values of simultaneous EWMA control charts (mean and variance charts) for given RL quantile*

Description

Computation of the critical values (similar to alarm limits) for different types of simultaneous EWMA control charts (based on the sample mean and the sample variance $S^2$) monitoring normal mean and variance.

Usage

```r
xsewma.q(lx, cx, ls, csu, df, alpha, mu, sigma, hsx=0, Nx=40, csl=0, hss=1, Ns=40, sided="upper", qm=30)
```
xsewma.q.crit(lx, ls, L0, alpha, df, mu0=0, sigma0=1, csu=NULL, 
hsx=0, hss=1, sided="upper", mode="fixed", Nx=20, Ns=40, qm=30, 
c.error=1e-12, a.error=1e-9)

**Arguments**

- **lx** smoothing parameter lambda of the two-sided mean EWMA chart.
- **cx** control limit of the two-sided mean EWMA control chart.
- **ls** smoothing parameter lambda of the variance EWMA chart.
- **csu** for two-sided (sided="two") and fixed upper control limit (mode="fixed", only for xsewma.q.crit) a value larger than sigma0 has to been given, for all other cases cu is ignored. It is the upper control limit of the variance EWMA control chart.
- **L0** in-control RL quantile at level alpha.
- **df** actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
- **alpha** quantile level.
- **mu** true mean.
- **sigma** true standard deviation.
- **mu0** in-control mean.
- **sigma0** in-control standard deviation.
- **hsx** so-called headstart (enables fast initial response) of the mean chart – do not confuse with the true FIR feature considered in xewma.arl; will be updated.
- **Nx** dimension of the approximating matrix of the mean chart.
- **cs1** lower control limit of the variance EWMA control chart; default value is 0; not considered if sided is "upper".
- **hss** headstart (enables fast initial response) of the variance chart.
- **Ns** dimension of the approximating matrix of the variance chart.
- **sided** distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at cl – the actual value of of cl is not used).
- **mode** only deployed for sided=\"two\" – with \"fixed\" an upper control limit (see cu) is set and only the lower is determined to obtain the in-control ARL L0, while with \"unbiased\" a certain unbiasedness of the ARL function is guaranteed (here, both the lower and the upper control limit are calculated).
- **qm** number of quadrature nodes used for the collocation integrals.
- **c.error** error bound for two succeeding values of the critical value during applying the secant rule.
- **a.error** error bound for the quantile level alpha during applying the secant rule.
Details

Instead of the popular ARL (Average Run Length) quantiles of the EWMA stopping time (Run Length) are determined. The algorithm is based on Waldmann's survival function iteration procedure and on Knoth (2007). \texttt{xsewma.q.crit} determines the critical values (similar to alarm limits) for given in-control RL quantile \( L_0 \) at level \( \alpha \) by applying secant rule and using \texttt{xsewma.sf()}.

In case of \( \texttt{side} = "\text{two}" \) and \( \texttt{mode} = "\text{unbiased}" \) a two-dimensional secant rule is applied that also ensures that the maximum of the RL cdf for given standard deviation is attained at \( \sigma_0 \).

Value

Returns a single value which resembles the RL quantile of order \( \alpha \) and the critical value of the two-sided mean EWMA chart and the lower and upper controls limit \( \texttt{csl} \) and \( \texttt{csu} \) of the variance EWMA chart, respectively.

Author(s)

Sven Knoth

References


See Also

\texttt{xsewma.arl} for calculation of ARL of simultaneous EWMA charts and \texttt{xsewma.sf} for the RL survival function.

Examples

## will follow

\begin{verbatim}
\end{verbatim}

\begin{verbatim}
xsewma.sf
\end{verbatim}

\begin{verbatim}
Compute the survival function of simultaneous EWMA control charts (mean and variance charts)
\end{verbatim}

Description

Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring simultaneously normal mean and variance.

Usage

\texttt{xsewma.sf(n, lx, cx, ls, csu, df, mu, sigma, hsx=0, Nx=40, csl=0, hss=1, Ns=40, sided="upper", qm=30)
Arguments

- **n**: calculate sf up to value n.
- **lx**: smoothing parameter lambda of the two-sided mean EWMA chart.
- **cx**: control limit of the two-sided mean EWMA control chart.
- **ls**: smoothing parameter lambda of the variance EWMA chart.
- **csu**: upper control limit of the variance EWMA control chart.
- **df**: actual degrees of freedom, corresponds to subgroup size (for known mean it is equal to the subgroup size, for unknown mean it is equal to subgroup size minus one.
- **mu**: true mean.
- **sigma**: true standard deviation.
- **hsx**: so-called headstart (enables fast initial response) of the mean chart – do not confuse with the true FIR feature considered in xewma.arl; will be updated.
- **Nx**: dimension of the approximating matrix of the mean chart.
- **csl**: lower control limit of the variance EWMA control chart; default value is 0; not considered if sided is "upper".
- **hss**: headstart (enables fast initial response) of the variance chart.
- **Ns**: dimension of the approximating matrix of the variance chart.
- **sided**: distinguishes between one- and two-sided two-sided EWMA-$S^2$ control charts by choosing "upper" (upper chart without reflection at cl – the actual value of cl is not used), "Rupper" (upper chart with reflection at cl), "Rlower" (lower chart with reflection at cu), and "two" (two-sided chart), respectively.
- **qm**: number of quadrature nodes used for the collocation integrals.

Details

The survival function $P(L>n)$ and derived from it also the cdf $P(L\leq n)$ and the pmf $P(L=n)$ illustrate the distribution of the EWMA run length. For large $n$ the geometric tail could be exploited. That is, with reasonable large $n$ the complete distribution is characterized. The algorithm is based on Waldmann’s survival function iteration procedure and on results in Knoth (2007).

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also
xsewma.arl for zero-state ARL computation of simultaneous EWMA control charts.

Examples
## will follow

```r
xshewhart.arl alpha, cS, delta=0, N1=50, N2=30)
```

Arguments

- `alpha`: lag 1 correlation of the data.
- `cS`: critical value (alias to alarm limit) of the Shewhart control chart.
- `delta`: potential shift in the data (in-control mean is zero).
- `N1`: number of quadrature nodes for solving the ARL integral equation, dimension of the resulting linear equation system is N1.
- `N2`: second number of quadrature nodes for combining the probability density function of the first observation following the margin distribution and the solution of the ARL integral equation.

Details

Following the idea of Schmid (1995), $1 - \alpha$ times the data turns out to be an EWMA smoothing of the underlying AR(1) residuals. Hence, by combining the solution of the EWMA ARL integral equation and the stationary distribution of the AR(1) data (normal distribution is assumed) one gets easily the overall ARL.

Value

It returns a single value resembling the zero-state ARL of a modified Shewhart chart.

Author(s)

Sven Knoth
References


See Also

*xewma.arl* for zero-state ARL computation of EWMA control charts.

Examples

```r
## Table 1 in Kramer/Schmid (2000)

cS <- 3.09023
a <- seq(0, 4, by=.5)
row1 <- row2 <- row3 <- NULL
for ( i in 1:length(a) ) {
  row1 <- c(row1, round(xshewhart.arl.arl( 0.4, cS, delta=a[i]), digits=2))
  row2 <- c(row2, round(xshewhart.arl.arl( 0.2, cS, delta=a[i]), digits=2))
  row3 <- c(row3, round(xshewhart.arl.arl(-0.2, cS, delta=a[i]), digits=2))
}

results <- rbind(row1, row2, row3)
results

# original values are
# row1 515.44 215.48 61.85 21.63 9.19 4.58 2.61 1.71 1.29
# row2 502.56 204.97 56.72 19.13 7.95 3.97 2.33 1.59 1.25
# row3 502.56 201.41 54.05 17.42 6.89 3.37 2.03 1.46 1.20
```

---

*xshewhartrunsrules.arl*

*Compute ARLs of Shewhart control charts with and without runs rules*

Description

Computation of the (zero-state and steady-state) Average Run Length (ARL) for Shewhart control charts with and without runs rules monitoring normal mean.

Usage

```r
xshewhartrunsrules.arl(mu, c = 1, type = "12")
xshewhartrunsrules.crit(L0, mu = 0, type = "12")
```
xshewhartrunsrules.ad(mu1, mu0 = 0, c = 1, type = "12")

xshewhartrunsrules.matrix(mu, c = 1, type = "12")

Arguments

mu  true mean.
L0  pre-defined in-control ARL, that is, determine c so that the mean number of observations until a false alarm is L0.
mu1, mu0  for the steady-state ARL two means are specified, mu0 is the in-control one and usually equal to 0, and mu1 must be given.
c  normalizing constant to ensure specific alarming behavior.
type  controls the type of Shewhart chart used, see details section.

Details

xshewhartrunsrules.arl determines the zero-state Average Run Length (ARL) based on the Markov chain approach given in Champ and Woodall (1987). xshewhartrunsrules.matrix provides the corresponding transition matrix that is also used in xDshewhartrunsrules.arl (ARL for control charting drift). xshewhartrunsrules.crit allows to find the normalization constant c to attain a fixed in-control ARL. Typically this is needed to calibrate the chart. With xshewhartrunsrules.ad the steady-state ARL is calculated. With the argument type certain runs rules could be set. The following list gives an overview.

• "1" The classical Shewhart chart with +/- 3 c sigma control limits (c is typically equal to 1 and can be changed by the argument c).
• "12" The classic and the following runs rule: 2 of 3 are beyond +/- 2 c sigma on the same side of the chart.
• "13" The classic and the following runs rule: 4 of 5 are beyond +/- 1 c sigma on the same side of the chart.
• "14" The classic and the following runs rule: 8 of 8 are on the same side of the chart (referring to the center line).

Value

Returns a single value which resembles the zero-state or steady-state ARL. xshewhartrunsrules.matrix returns a matrix.

Author(s)

Sven Knoth

References

See Also

`xDshewhartrunsrules.arl` for zero-state ARL of Shewhart control charts with or without runs rules under drift.

Examples

```r
## Champ/Woodall (1987)
## Table 1
mus <- c(0:15)/5
Mxshewhartrunsrules.arl <- Vectorize(xshewhartrunsrules.arl, "mu")

# standard (1 of 1 beyond 3 sigma) Shewhart chart without runs rules
C1 <- round(Mxshewhartrunsrules.arl(mus, type="1"), digits=2)

# standard + runs rule: 2 of 3 beyond 2 sigma on the same side
C12 <- round(Mxshewhartrunsrules.arl(mus, type="12"), digits=2)

# standard + runs rule: 4 of 5 beyond 1 sigma on the same side
C13 <- round(Mxshewhartrunsrules.arl(mus, type="13"), digits=2)

# standard + runs rule: 8 of 8 on the same side of the center line
C14 <- round(Mxshewhartrunsrules.arl(mus, type="14"), digits=2)

## original results are
# mus C1 C12 C13 C14
# 0.0 370.40 225.44 166.05 152.73
# 0.2 308.43 177.56 120.70 110.52
# 0.4 200.08 104.46 63.88 59.76
# 0.6 119.67 57.92 33.99 33.64
# 0.8 71.55 33.12 19.78 21.07
# 1.0 43.89 20.01 12.66 14.58
# 1.2 27.82 12.81 8.84 10.90
# 1.4 18.25 8.69 6.62 8.60
# 1.6 12.38 6.21 5.24 7.03
# 1.8 8.69 4.66 4.33 5.85
# 2.0 6.30 3.65 3.68 4.89
# 2.2 4.72 2.96 3.18 4.08
# 2.4 3.65 2.48 2.78 3.38
# 2.6 2.90 2.13 2.43 2.81
# 2.8 2.38 1.87 2.14 2.35
# 3.0 2.00 1.68 1.89 1.99

data.frame(mus, C1, C12, C13, C14)

## plus calibration, i. e. L0=250 (the maximal value for "14" is 255!)
L0 <- 250
c1 <- xshewhartrunsrules.crit(L0, type = "1")
c12 <- xshewhartrunsrules.crit(L0, type = "12")
c13 <- xshewhartrunsrules.crit(L0, type = "13")
c14 <- xshewhartrunsrules.crit(L0, type = "14")
C1 <- round(Mxshewhartrunsrules.arl(mus, c=c1, type="1"), digits=2)
C12 <- round(Mxshewhartrunsrules.arl(mus, c=c12, type="12"), digits=2)
C13 <- round(Mxshewhartrunsrules.arl(mus, c=c13, type="13"), digits=2)
C14 <- round(Mxshewhartrunsrules.arl(mus, c=c14, type="14"), digits=2)
data.frame(mus, C1, C12, C13, C14)
```
## and the steady-state ARL

```r
Mxshewhartrunsrules.ad <- Vectorize(xshewhartrunsrules.ad, "mu")
C1 <- round(Mxshewhartrunsrules.ad(mus, c=c1, type="1"), digits=2)
C12 <- round(Mxshewhartrunsrules.ad(mus, c=c12, type="12"), digits=2)
C13 <- round(Mxshewhartrunsrules.ad(mus, c=c13, type="13"), digits=2)
C14 <- round(Mxshewhartrunsrules.ad(mus, c=c14, type="14"), digits=2)
data.frame(mus, C1, C12, C13, C14)
```

---

### xtcusum.arl

#### Compute ARLs of CUSUM control charts

**Description**

Computation of the (zero-state) Average Run Length (ARL) for different types of CUSUM control charts monitoring normal mean.

**Usage**

```r
xtcusum.arl(k, h, df, mu, hs = 0, sided="one", mode="tan", r=30)
```

**Arguments**

- **k**: reference value of the CUSUM control chart.
- **h**: decision interval (alarm limit, threshold) of the CUSUM control chart.
- **df**: degrees of freedom – parameter of the t distribution.
- **mu**: true mean.
- **hs**: so-called headstart (give fast initial response).
- **sided**: distinguish between one- and two-sided CUSUM schemes by choosing "one" and "two", respectively.
- **r**: number of quadrature nodes, dimension of the resulting linear equation system is equal to \(r+1\).
- **mode**: Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.

**Details**

`xtcusum.arl` determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystr"om method based on Gauss-Legendre quadrature.

**Value**

Returns a single value which resembles the ARL.

**Author(s)**

Sven Knoth
References


D. Brook, D. A. Evans (1972), An approach to the probability distribution of cusum run length, Biometrika 59, 539-548.


See Also

xtewma.ar1 for zero-state ARL computation of EWMA control charts and xtcusum.ar1 for the zero-state ARL of CUSUM for normal data.

Examples

## will follow

xtewma.ad

Compute steady-state ARLs of EWMA control charts, t distributed data

Description

Computation of the steady-state Average Run Length (ARL) for different types of EWMA control charts monitoring the mean of t distributed data.

Usage

xtewma.ad(l, c, df, muL, mu0=0, zr=0, z0=0, sided="one", limits="fix", steady.state.mode="conditional", mode="tan", r=40)

Arguments

- **l**: smoothing parameter lambda of the EWMA control chart.
- **c**: critical value (similar to alarm limit) of the EWMA control chart.
- **df**: degrees of freedom – parameter of the t distribution.
- **muL**: in-control mean.
- **mu0**: out-of-control mean.
- **zr**: reflection border for the one-sided chart.
z0  restarting value of the EWMA sequence in case of a false alarm in steady.state.mode="cyclical".
sided  distinguishes between one- and two-sided two-sided EWMA control chart by choosing "one" and "two", respectively.
limits  distinguishes between different control limits behavior.
steady.state.mode  distinguishes between two steady-state modes – conditional and cyclical.
mode  Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.
r  number of quadrature nodes, dimension of the resulting linear equation system is equal to r+1 (one-sided) or r (two-sided).

Details
xtewma.ad determines the steady-state Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature and using the power method for deriving the largest in magnitude eigenvalue and the related left eigenfunction.

Value
Returns a single value which resembles the steady-state ARL.

Author(s)
Sven Knoth

References

See Also
xtewma.arl for zero-state ARL computation and xewma.ad for the steady-state ARL for normal data.

Examples
## will follow
xtewma.arl

Compute ARLs of EWMA control charts, t distributed data

Description

Computation of the (zero-state) Average Run Length (ARL) for different types of EWMA control charts monitoring the mean of t distributed data.

Usage

xtewma.arl(l, c, df, mu, zr=0, hs=0, sided="two", limits="fix", mode="tan", q=1, r=40)

Arguments

- \( l \) smoothing parameter lambda of the EWMA control chart.
- \( c \) critical value (similar to alarm limit) of the EWMA control chart.
- \( df \) degrees of freedom – parameter of the t distribution.
- \( mu \) true mean.
- \( zr \) reflection border for the one-sided chart.
- \( hs \) so-called headstart (enables fast initial response).
- \( sided \) distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
- \( limits \) distinguishes between different control limits behavior.
- \( mode \) Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.
- \( q \) change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu! = 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq q) \), will be determined. Note that \( \mu_0=0 \) is implicitly fixed.
- \( r \) number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-sided) or \( r \) (two-sided).

Details

In case of the EWMA chart with fixed control limits, xtwema.arl determines the Average Run Length (ARL) by numerically solving the related ARL integral equation by means of the Nystroem method based on Gauss-Legendre quadrature. If \( limits = "vac1" \), then the method presented in Knoth (2003) is utilized. Other values (normal case) for \( limits \) are not yet supported.

Value

Except for the fixed limits EWMA charts it returns a single value which resembles the ARL. For fixed limits charts, it returns a vector of length \( q \) which resembles the ARL and the sequence of conditional expected delays for \( q=1 \) and \( q>1 \), respectively.
Author(s)
Sven Knoth

References


See Also
xewma.arl for zero-state ARL computation of EWMA control charts in the normal case.

Examples
```r
## Borror/Montgomery/Runger (1999), Table 3
lambda <- 0.1
cE <- 2.703
df <- c(4, 6, 8, 10, 15, 20, 30, 40, 50)
L0 <- rep(NA, length(df))
for ( i in 1:length(df) ) {
  L0[i] <- round(xewma.arl(lambda, cE*sqrt(df[i]/(df[i]-2)), df[i], 0), digits=0)
}
data.frame(df, L0)
```

xewma.q  Compute RL quantiles of EWMA control charts

Description
Computation of quantiles of the Run Length (RL) for EWMA control charts monitoring normal mean.
Usage

xtewma.q(l, c, df, mu, alpha, zr=0, hs=0, sided="two", limits="fix", mode="tan", q=1, r=40)

xtewma.q.crit(l, L0, df, mu, alpha, zr=0, hs=0, sided="two", limits="fix", mode="tan", r=40, c.error=1e-12, a.error=1e-9, OUTPUT=FALSE)

Arguments

1 smoothing parameter lambda of the EWMA control chart.
c critical value (similar to alarm limit) of the EWMA control chart.
df degrees of freedom – parameter of the t distribution.
mu true mean.
alpha quantile level.
zs reflection border for the one-sided chart.
hs so-called headstart (enables fast initial response).
sided distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.
limits distinguishes between different control limits behavior.
mode Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.
q change point position. For \( q = 1 \) and \( \mu = \mu_0 \) and \( \mu = \mu_1 \), the usual zero-state ARLs for the in-control and out-of-control case, respectively, are calculated. For \( q > 1 \) and \( \mu \neq 0 \) conditional delays, that is, \( E_q(L - q + 1|L \geq) \), will be determined. Note that \( \mu_0=0 \) is implicitly fixed.
r number of quadrature nodes, dimension of the resulting linear equation system is equal to \( r+1 \) (one-sided) or \( r \) (two-sided).
L0 in-control quantile value.
c.error error bound for two succeeding values of the critical value during applying the secant rule.
a.error error bound for the quantile level alpha during applying the secant rule.
OUTPUT activate or deactivate additional output.

Details

Instead of the popular ARL (Average Run Length) quantiles of the EWMA stopping time (Run Length) are determined. The algorithm is based on Waldmann’s survival function iteration procedure. If limits is "vacl", then the method presented in Knoth (2003) is utilized. For details see Knoth (2004).

Value

Returns a single value which resembles the RL quantile of order \( q \).
xtewma.sf

Author(s)
Sven Knoth

References

See Also
xtewma.q for RL quantile computation of EWMA control charts in the normal case.

Examples
## will follow

xtewma.sf            Compute the survival function of EWMA run length

Description
Computation of the survival function of the Run Length (RL) for EWMA control charts monitoring normal mean.

Usage
xtewma.sf(l, c, df, mu, n, zr=0, hs=0, sided="two", limits="fix", mode="tan", q=1, r=40)

Arguments
- l: smoothing parameter lambda of the EWMA control chart.
- c: critical value (similar to alarm limit) of the EWMA control chart.
- df: degrees of freedom – parameter of the t distribution.
- mu: true mean.
- n: calculate sf up to value n.
- zr: reflection border for the one-sided chart.
- hs: so-called headstart (enables fast initial response).
sided distinguishes between one- and two-sided EWMA control chart by choosing "one" and "two", respectively.

limits distinguishes between different control limits behavior.

mode Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.

q change point position. For $q = 1$ and $\mu = \mu_0$ and $\mu = \mu_1$, the usual zero-state situation for the in-control and out-of-control case, respectively, are calculated. Note that $\mu_0=0$ is implicitly fixed.

r number of quadrature nodes, dimension of the resulting linear equation system is equal to $r+1$ (one-sided) or $r$ (two-sided).

Details

The survival function $P(L > n)$ and derived from it also the cdf $P(L \leq n)$ and the pmf $P(L = n)$ illustrate the distribution of the EWMA run length. For large $n$ the geometric tail could be exploited. That is, with reasonable large $n$ the complete distribution is characterized. The algorithm is based on Waldmann's survival function iteration procedure. For varying limits and for change points after 1 the algorithm from Knoth (2004) is applied. For details see Knoth (2004).

Value

Returns a vector which resembles the survival function up to a certain point.

Author(s)

Sven Knoth

References


See Also

*xewma.sf* for survival function computation of EWMA control charts in the normal case.

Examples

```r
## will follow
```
xtshewhart.ar1.arl

Description

Computation of the (zero-state) Average Run Length (ARL) for modified Shewhart charts deployed to the original AR(1) data where the residuals follow a Student t distribution.

Usage

xtshewhart.ar1.arl(alpha, cS, df, delta=0, N1=50, N2=30, N3=2*N2, INFI=10, mode="tan")

Arguments

- **alpha**: lag 1 correlation of the data.
- **cS**: critical value (alias to alarm limit) of the Shewhart control chart.
- **df**: degrees of freedom – parameter of the t distribution.
- **delta**: potential shift in the data (in-control mean is zero).
- **N1**: number of quadrature nodes for solving the ARL integral equation, dimension of the resulting linear equation system is N1.
- **N2**: second number of quadrature nodes for combining the probability density function of the first observation following the margin distribution and the solution of the ARL integral equation.
- **N3**: third number of quadrature nodes for solving the left eigenfunction integral equation to determine the margin density (see Andel/Hrach, 2000), dimension of the resulting linear equation system is N3.
- **INFI**: substitute of Inf – the left eigenfunction integral equation is defined on the whole real axis; now it is reduced to (-INFI,INFI).
- **mode**: Controls the type of variables substitution that might improve the numerical performance. Currently, "identity", "sin", "sinh", and "tan" (default) are provided.

Details

Following the idea of Schmid (1995), 1-\alpha times the data turns out to be an EWMA smoothing of the underlying AR(1) residuals. Hence, by combining the solution of the EWMA ARL integral equation and the stationary distribution of the AR(1) data (here Student t distribution is assumed) one gets easily the overall ARL.

Value

It returns a single value resembling the zero-state ARL of a modified Shewhart chart.
Author(s)
Sven Knoth

References

See Also
xtewma.arl for zero-state ARL computation of EWMA control charts in case of Student t distributed data.

Examples
## will follow
Index

*dTopic ts
   dphat, 3
euklid.ewma.arl, 5
lns2ewma.arl, 6
lns2ewma.crit, 8
mewma.arl, 10
mewma.crit, 14
mewma.psi, 15
p.ewma.arl, 17
phat.ewma.arl, 19
pois.ewma.arl, 21
pois.ewma.crit, 23
quadrature.nodes.weights, 24
scusum.arl, 25
scusum.crit, 27
scusums.arl, 29
sewma.arl, 30
sewma.arl.prerun, 33
sewma.crit, 34
sewma.crit.prerun, 37
sewma.q, 39
sewma.q.prerun, 41
sewma.sf, 43
sewma.sf.prerun, 44
tewma.arl, 46
tol.lim.fac, 47
x.res.ewma.arl, 49
xcusum.ad, 53
xcusum.arl, 55
xcusum.crit, 58
xcusum.crit.L0h, 59
xcusum.crit.L0L1, 61
xcusum.q, 63
xcusum.sf, 64
xDocusum.arl, 65
xDewma.arl, 68
xDgrsr.arl, 73
xDshewhartrunsrules.arl, 75
xewma.ad, 77
xewma.arl, 79
xewma.arl.f, 85
xewma.arl.prerun, 86
xewma.crit, 89
xewma.q, 90
xewma.q.prerun, 92
xewma.sf, 95
xewma.sf.prerun, 97
xgrsr.ad, 99
xgrsr.arl, 101
xgrsr.crit, 104
xsewma.arl, 105
xsewma.crit, 107
xsewma.q, 109
xsewma.sf, 111
xshewhart.arl.arl, 113
xshewhartrunsrules.arl, 114
xtcusum.arl, 117
xtewma.ad, 118
xtewma.arl, 120
xtewma.q, 121
xtewma.sf, 123
xshewhart.arl1.arl, 125

dphat, 3
euklid.ewma.arl, 5
lns2ewma.arl, 6
lns2ewma.crit, 8
mewma.ad (mewma.arl), 10
mewma.arl, 10
mewma.crit, 14
mewma.psi, 15
p.ewma.arl, 17
phat.ewma.arl, 19
phat.ewma.crit (phat.ewma.arl), 19
phat.ewma.lambda (phat.ewma.arl), 19
pois.ewma.arl, 21
pois.ewma.crit, 23
pphat(dphat), 3
qphat(dphat), 3
quadrature.nodes.weights, 24
s.res.ewma.arl(x.res.ewma.arl), 49
scusum.arl, 25
scusum.crit, 27
scusums.arl, 29
sewma.arl, 30
sewma.arl.prerun, 33
sewma.crit, 34
sewma.crit.prerun, 37
sewma.q, 39
sewma.q.crit.prerun(sewma.q.prerun), 41
sewma.q.prerun, 41
sewma.sf, 43
sewma.sf.prerun, 44
tewma.arl, 46
tol.lim.fac, 47
x.res.ewma.arl, 49
xcusum.ad, 53
xcusum.arl, 55
xcusum.crit, 58
xcusum.crit.L0h, 59
xcusum.crit.L0L1, 61
xcusum.q, 63
xcusum.sf, 64
xDCusum.arl, 65
xDewma.arl, 68
xDgrsr.arl, 73
xDshewhartrunsrules.arl, 75
xDshewhartrunsrulesFixedm.arl
  (xDshewhartrunsrules.arl), 75
xewma.ad, 77
xewma.arl, 79
xewma.arl.f, 85
xewma.arl.prerun, 86
xewma.crit, 89
xewma.crit.prerun(xewma.arl.prerun), 86
xewma.q, 90
xewma.q.crit.prerun(xewma.q.prerun), 92
xewma.q.prerun, 92
xewma.sf, 95
xewma.sf.prerun, 97
xgrsr.ad, 99
xgrsr.arl, 101
xgrsr.crit, 104
xs.res.ewma.arl(x.res.ewma.arl), 49
xs.res.ewma.pms(x.res.ewma.arl), 49
xsewma.arl, 105
xsewma.crit, 107
xsewma.q, 109
xsewma.sf, 111
xshewhart.arl.arl, 113
xshewhartrunsrules.ad
  (xshewhartrunsrules.arl), 114
xshewhartrunsrules.arl, 114
xshewhartrunsrules.crit
  (xshewhartrunsrules.arl), 114
xshewhartrunsrules.matrix
  (xshewhartrunsrules.arl), 114
xtcusum.arl, 117
xtewma.ad, 118
xtewma.arl, 120
xtewma.q, 121
xtewma.sf, 123
xtshewhart.arl.arl, 125