Package ‘spc’

February 15, 2012

Title Statistical Process Control

Version 0.4.1

Date 2011-04-05

Author Sven Knoth, ...

Maintainer Sven Knoth <Sven.Knoth@gmx.de>

Description Evaluation of control charts by means of the zero-state,steady-state ARL (Average Run Length). Setting up control charts for given in-control ARL and plotting of the related figures. The control charts under consideration are one- and two-sided EWMA, CUSUM, and Shiryaev-Roberts schemes for monitoring the mean of normally distributed independent data. Now, the ARL calculation of the same set of schemes under drift are added. Other charts and parameters are in preparation. Further SPC areas will be covered as well (sampling plans,capability indices ...).

Depends R (>= 1.4.0)

License GPL

Repository CRAN

Date/Publication 2011-09-02 12:40:07

R topics documented:

`phat.ewma.arl` *Compute 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

```
phat.ewma.arl(lambda, ucl, mu, n, z0, sigma=1, LSL=-3, USL=3, N=15, qm=15)

phat.ewma.crit(lambda, L0, mu, n, z0, sigma=1, LSL=-3, USL=3, N=15, qm=15)

phat.ewma.lambda(L0, mu, n, z0, sigma=1, max_l=1, min_l=.001, LSL=-3, USL=3, qm=15)
```

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`	Batch size.
`z0`	so-called headstart (give fast initial response).
`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.

Details

More details will follow.

Value

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

Author(s)

Sven Knoth

References

S. Knoth, S. Steinmetz (2011), EWMA *p* charts under Sampling by Variables – Ideas, Numerics and Properties, *Submitted to Technometrics*.

See Also

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

Examples

```
## S. Knoth, S. Steinmetz (2011),

# Table 1

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")
arls <- NULL
for ( i in 1:length(lambdas) ) {
```

```
 arls <- cbind(arls, round(phat.ewma.ARL(lambdas[i], ucls[i], mus, n, p.star, LSL=LSL, USL=USL), digits=2))
}
arls <- data.frame(arls, row.names=NULL)
names(arls) <- lambdas
print(arls)

# Table 3

mus <- c(.25, .5, 1, 2)
phat.ewma.LAMBDA <- Vectorize("phat.ewma.lambda", "mu")
lambdas <- phat.ewma.LAMBDA(L0, mus, n, p.star, LSL=LSL, USL=USL)
print(cbind(mus, lambdas))
```

---

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

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

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 batch size.
`s2.on`	distinguish between S^2 and S chart.
`hs`	so-called headstart (give fast initial response).
`sided`	distinguish 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.
`qm`	number of quadrature nodes.

Details

`sewma.arl` 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

S. Knoth (2005), Accurate ARL computation for EWMA-S^2 control charts, *Statistics and Computing 15*, 341-352.

S. Knoth (2006), Computation of the ARL for CUSUM-S^2 schemes, *Computational Statistics & Data Analysis 51*, 499-512.

See Also

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

Examples

```
## Knoth (2005)
## compare with Table 1 (p. 347): 249.9997
## Monte Carlo with 10^9 replicates: 249.9892 +/- 0.008
l <- .025
df <- 1
cu <- 1 + 1.661865*sqrt(l/(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("sewma.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

l <- 0.15
```

```
df <- 4
cu <- 1 + 2.4831*sqrt(l/(2-l))*sqrt(2/df)
sigmas <- c(1 + (0:5)/100, 1 + (1:5)/10, 2)
arls <- round(Ssewma.arl(l, 1, cu, sigmas, df, sided="Rupper", r=100), digits=2)
data.frame(sigmas, arls)


## 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

l <- 0.115
df <- 5
cl <- 1 - 2.0613*sqrt(l/(2-l))*sqrt(2/df)
sigmas <- c((10:6)/10)
arls <- round(Ssewma.arl(l, cl, 1, sigmas, df, sided="Rlower", r=100), digits=2)
data.frame(sigmas, arls)
```

`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=1,cl=NULL,cu=NULL,hs=1,s2.on=TRUE,sided="upper",mode="fixed",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 batch size.
`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 (give fast initial response).
`s2.on`	distinguish between S^2 and S chart.

`sided`	distinguish 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).
`r`	dimension of the resulting linear equation system.
`qm`	number of quadrature nodes.

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 ? and the related example.

Value

Returns the lower and upper control limit `cl` and `cu`.

Author(s)

Sven Knoth

References

H.-J. Mittag and D. Stemann and B. Tewes (1998), EWMA-Karten zur \"Uberwachung der Streuung von Qualit\"atsmerkmalen, *Allgemeines Statistisches Archiv 82*, 327-338,

C. A. Acosta-Mej\'ia and J. J. Pignatiello Jr. and B. V. Rao (1999), A comparison of control charting procedures for monitoring process dispersion, *IIE Transactions 31*, 569-579.

S. Knoth (2005), Accurate ARL computation for EWMA-S^2 control charts, *Statistics and Computing 15*, 341-352.

S. Knoth (2006), Computation of the ARL for CUSUM-S^2 schemes, *Computational Statistics & Data Analysis 51*, 499-512.

See Also

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

Examples

```
## 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 \eqn{S^2}{S^2})
```

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

limits.cu.mittag_et_al <- 1 + sqrt(l/(2-l))*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 \eqn{S^2}{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-l))*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.

l  <- 0.1
L0 <- 500
df <- 4
limits <- sewma.crit(l, L0, df, sided="two", mode="unbiased", r=80)
SEWMA.arl <- Vectorize(sewma.arl, "sigma")
SEWMA.ARL <- function(sigma)
  SEWMA.arl(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")

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

## upper chart with reflection at sigma0=1 in Table 4: c = 2.4831
l <- 0.15
L0 <- 100
df <- 4
limits <- sewma.crit(l, L0, df, cl=1, sided="Rupper", r=100)
cv.Tab4 <- (limits["cu"]-1)/( sqrt(l/(2-l))*sqrt(2/df) )
cv.Tab4

## lower chart with reflection at sigma0=1 in Table 5: c = 2.0613
l <- 0.115
```

```
L0 <- 200
df <- 5
limits <- sewma.crit(l, L0, df, cu=1, sided="Rlower", r=100)
cv.Tab5 <- -(limits["cl"]-1)/( sqrt(l/(2-l))*sqrt(2/df) )
cv.Tab5
```

`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 the so-called tolerance limit factors k. These values are computed for a given sample size n.

Usage

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

Arguments

`n`	sample size.
`p`	coverage.
`a`	error probability α, 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.
`m`	number of abscissas for the quadrature (needed only for `method="exact"`), of course, the larger the more accurate.

Details

`tol.lim.fac` 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

A. Wald, J. Wolfowitz (1946), Tolerance limits for a normal distribution, *Annals of Mathematical Statistics 17*, 208-215.

R. E. Odeh, D. B. Owen (1980), *Tables for Normal Tolerance Limits*, Sampling Plans, and Screening, Marcel Dekker, New York.

See Also

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

Examples

```
n <- 2:10
p <- .95
a <- .05
kWW <- sapply(n,p=p,a=a,tol.lim.fac)
kEX <- sapply(n,p=p,a=a,mode="exact",tol.lim.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^{1:7}
p <- .95
a <- .05
kEX <- round(sapply(n,p=p,a=a,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)
```

`x.res.ewma.arl` *Compute ARLs of EWMA residual control charts*

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

```
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)

xs.res.ewma.arl(lx, cx, ls, csu, mu, sigma, alpha=0, n=5, hsx=0, rx=40, hss=1, rs=40, qm=30)

xs.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 (give 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

S. Knoth, M. C. Morais, A. Pacheco, W. Schmid (2009), Misleading Signals in Simultaneous Residual Schemes for the Mean and Variance of a Stationary Process, *Commun. Stat., Theory Methods 38*, 2923-2943.

S. Knoth, W. Schmid, A. Schoene (2001), Simultaneous Shewhart-Type Charts for the Mean and the Variance of a Time Series, *Frontiers of Statistical Quality Control 6, A. Lenz, H.-J. & Wilrich, P.-T. (Eds.)*, 6, 61-79.

S. Knoth, W. Schmid (2002) Monitoring the mean and the variance of a stationary process, *Statistica Neerlandica 56*, 77-100.

See Also

`xewma.arl`, `sewma.arl`, and `xsewma.arl` as more elaborated functions in the iid case.

Examples

```
## S. Knoth, W. Schmid (2002)

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)
  names(s.e) <- NULL
  crit <- rbind(crit, data.frame(lambda, x.e, s.e))
}


## orinal 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

## orinal 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)

## S. Knoth, M. C. Morais, A. Pacheco, W. Schmid (2009)

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))
}

## orinal 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

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 0.014580 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 0.000703 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

PMS.4
```

`xcusum.ad` *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

```
xcusum.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

xcusum.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^2. 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

R. B. Crosier (1986), A new two-sided cumulative quality control scheme, *Technometrics 28*, 187-194.

See Also

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

Examples

```
## 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   4.31   4.39
## 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

```
xcusum.arl(k, h, mu, hs = 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.
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.
`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 single value which resembles the ARL.

Author(s)

Sven Knoth

References

A. L. Goel, S. M. Wu (1971), Determination of A.R.L. and a contour nomogram for CUSUM charts to control normal mean, *Technometrics 13*, 221-230.

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

J. M. Lucas, R. B. Crosier (1982), Fast initial response for cusum quality-control schemes: Give your cusum a headstart, *Technometrics 24*, 199-205.

L. C. Vance (1986), Average run lengths of cumulative sum control charts for controlling normal means, *Journal of Quality Technology 18*, 189-193.

K.-H. Waldmann (1986), Bounds for the distribution of the run length of one-sided and two-sided CUSUM quality control schemes, *Technometrics 28*, 61-67.

R. B. Crosier (1986), A new two-sided cumulative quality control scheme, *Technometrics 28*, 187-194.

See Also

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

Examples

```
## 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
## 1.00   8.38   5.29
## 1.50   4.75   2.86
## 2.00   3.34   2.01
## 2.50   2.62   1.59
## 3.00   2.19   1.32
## 4.00   1.71   1.07
## 5.00   1.31   1.01.

## Vance (1986), one-sided CUSUM
## The first paper on using Nystroem method and Gauss-Legendre quadrature
## for solving the ARL integral equation (see as well Goel/Wu, 1971)

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

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

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

k <- .5
h <- 3.73
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,4,5)
round(cbind(mu,sapply(mu,k=k,h=h,sided="Crosier",xcusum.arl)),digits=2)

## results in the original paper are (in Table 3)
## 0.00 168.
## 0.25  70.7
## 0.50  25.1
## 0.75  12.5
## 1.00   7.92
## 1.50   4.49
## 2.00   3.17
## 2.50   2.49
## 3.00   2.09
## 4.00   1.60
## 5.00   1.22.

## SAS/QC manual 1999
## one- and two-sided CUSUM schemes

## one-sided

k <- .25
h <- 8
mu <- 2.5
print(xcusum.arl(k,h,mu),digits=12)
print(xcusum.arl(k,h,mu,hs=.1),digits=12)

## original results are 4.1500836225 and 4.1061588131.

## two-sided

print(xcusum.arl(k,h,mu,sided="two"),digits=12)

## original result is 4.1500826715.
```

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)
```

Arguments

`k`	reference value of the CUSUM control chart.
`L0`	in-control ARL.
`mu0`	in-control mean.
`hs`	so-called headstart (give fast initial response).
`sided`	distinguish between one-, two-sided and Crosier's modified two-sided CUSUM schemoosing `"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

`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 give 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 (give fast initial response).
`sided`	distinguish between one-, two-sided and Crosier's modified two-sided CUSUM schemoosing `"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.eps`	error bound for the L0 error.
`k.eps`	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 (give fast initial response).
`sided`	distinguish between one-, two-sided and Crosier's modified two-sided CUSUM schemoosing `"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, K. W. Kemp (1960), Sampling inspection of continuous processes with no autocorrelation between successive results, *Biometrika 47*, 363-380.

K. W. Kemp (1962), The Use of Cumulative Sums for Sampling Inspection Schemes, *Journal of the Royal Statistical Sociecty C, Applied Statistics, 10*, 16-31.

See Also

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

Examples

```
## 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   3.80
#        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\th\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
# one confirmation run
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("\nL1\tL0=250\tL0=500\tL0=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])
  }
  cat("\n")
}
#
# two confirmation runs
xcusum.arl(0.52, 5, 0, sided="two") # Kemp
xcusum.arl(result[1], result[2], 0, sided="two") # here
xcusum.arl(0.52, 5, 2*0.52, sided="two") # Kemp
xcusum.arl(result[1], result[2], 2*result[1], sided="two") # here
#
```

---

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, p, 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.
p	quantile level.
hs	so-called headstart (give fast initial response).
sided	distinguish 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

K.-H. Waldmann (1986), Bounds for the distribution of the run length of one-sided and two-sided CUSUM quality control schemes, *Technometrics 28*, 61-67.

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

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

`xDcusum.arl` *Compute ARLs of CUSUM control charts under drift*

Description

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

Usage

```
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 (give fast initial response).
- `sided` distinguish between one- and two-sided CUSUM control chart by choosing `"one"` and `"two"`, respectively. Currentlly, the two-sided scheme is not implemented.
- `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 $\mu = \mu_0$ 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 implicitely fixed. Deploy large `q` to mimic steady-state. It works only for `mode="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` define whether the first observation used for the ARL calculatio follows already 1*delta or 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.

F. F. Gan (1996), Average Run Lengths for Cumulative Sum control chart under linear trend, *Applied Statistics 45*, 505-512.

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, *to appear in Metrika*.

See Also

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

Examples

```
## 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
k <- .25
h <- 8
r <- 50
DxDcusum.arl <- Vectorize(xDcusum.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)
arl.like.Gan   <-
  round(DxDcusum.arl(k, h, deltas, r=r, with0=TRUE), digits=2)
arl.like.Knoth <-
  round(DxDcusum.arl(k, h, deltas, r=r, mode="Knoth", with0=TRUE), digits=2)
data.frame(deltas, arl.like.Gan, arl.like.Knoth)


## Zou et al. (2009)
## Table 1
## original values are
#  delta   arl1  arl2  arl3
#  0           ~ 1730
#  0.0005  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
k1 <- 0.25
k2 <- 0.5
```

```
k3 <- 0.75
h1 <- 9.660
h2 <- 5.620
h3 <- 3.904
deltas <- c(0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1:4)
arl1 <- c(round(xcusum.arl(k1, h1, 0, r=r), digits=2),
          round(DxDcusum.arl(k1, h1, deltas, r=r), digits=2))
arl2 <- c(round(xcusum.arl(k2, h2, 0), digits=2),
          round(DxDcusum.arl(k2, h2, deltas, r=r), digits=2))
arl3 <- c(round(xcusum.arl(k3, h3, 0, r=r), digits=2),
          round(DxDcusum.arl(k3, h3, deltas, r=r), digits=2))
data.frame(delta=c(0, deltas), arl1, arl2, arl3)
```

`xDewma.arl` *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 (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.
`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 $\mu = \mu_0$ 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 implicitely fixed. Deploy large `q` to mimic steady-state. It works only for `mode="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` define whether the first observation used for the ARL calculatio follows already 1*delta or 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, ... 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

F. F. Gan (1991), EWMA control chart under linear drift, *J. Stat. Comput. Simulation 38*, 181-200.

L. A. Aerne, C. W. Champ and S. E. Rigdon (1991), Evaluation of control charts under linear trend, *Commun. Stat., Theory Methods 20*, 3341-3349.

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

H. M. Fahmy and E. A. Elsayed (2006), Detection of linear trends in process mean, *International Journal of Production Research 44*, 487-504.

C. Zou, Y. Liu and Z. Wang (2009), Comparisons of control schemes for monitoring the means of processes subject to drifts, *to appear in Metrika*.

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

```
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
#
lambda1 <- 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", with0=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", with0=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", with0=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", with0=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", with0=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", with0=TRUE),
                         digits=2))
data.frame(delta=c(0, deltas), arlE1, arlE2, arlE3)

## Aerne et al. (1991) -- two-sided EWMA
## Table I (continued)
## original numbers are
#     delta  arlE1  arlE2  arlE3
#  0.000000  465.0  465.0  465.0
#  0.005623  77.01  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
cE1 <- 2.856
cE2 <- 2.974
cE3 <- 3.049
deltas <- 10^(-(18:0)/8)
arlE10 <- round(xewma.arl(lambda1, cE1, 0, sided="two"), digits=2)
arlE1 <- c(arlE10, round(DxDewma.arl(lambda1, cE1, deltas, sided="two"), digits=2))
arlE20 <- round(xewma.arl(lambda2, cE2, 0, sided="two"), digits=2)
arlE2 <- c(arlE20, round(DxDewma.arl(lambda2, cE2, deltas, sided="two"), digits=2))
arlE30 <- round(xewma.arl(lambda3, cE3, 0, sided="two"), digits=2)
arlE3 <- c(arlE30, round(DxDewma.arl(lambda3, cE3, 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
#
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(DxDewma.arl(lambda, cE, deltas, sided="two"), digits=3))
arlE51 <- c(round(xewma.arl(lambda, cE, 0, sided="two", q=51), digits=3),
     round(DxDewma.arl(lambda, cE, deltas, sided="two", mode="Knoth", q=51),
           digits=3))
data.frame(delta=c(0, deltas), arlE1, arlE51)

## additional Monte Carlo results with 10^8 replicates
#   delta   arl.q=1   s.e.    arl.q=51  s.e.
#   0.00    368.910   0.036   361.714   0.038
#   0.10     12.986   0.000    12.781   0.000
#   0.25      7.758   0.000     7.637   0.000
#   0.50      5.318   0.000     5.235   0.000
#   0.75      4.285   0.000     4.218   0.000
#   1.00      3.688   0.000     3.628   0.000
#   1.25      3.274   0.000     3.233   0.000
#   1.50      2.993   0.000     2.942   0.000
#   1.75      2.808   0.000     2.723   0.000
#   2.00      2.616   0.000     2.554   0.000

## Zou et al. (2009) -- one-sided EWMA
## Table 1
## original values are
#  delta   arl1  arl2  arl3
#  0           ~ 1730
#  0.0005  317   377   440
#  0.001   215   253   297
#  0.005   83.6  92.6  106
#  0.01    55.6  58.8  66.1
#  0.05    22.6  21.1  22.0
#  0.1     15.5  13.9  13.8
#  0.5     6.65  5.56  5.09
#  1.0     4.67  3.83  3.43
#  2.0     3.21  2.74  2.32
#  3.0     2.86  2.06  1.98
#  4.0     2.14  2.00  1.83
l1 <- 0.03479
l2 <- 0.11125
l3 <- 0.23052
c1 <- 2.711
c2 <- 3.033
c3 <- 3.161
zr <- -6
r  <- 50
deltas <- c(0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1:4)
arl1 <- c(round(xewma.arl(l1, c1, 0, zr=zr, r=r), digits=2),
          round(DxDewma.arl(l1, c1, deltas, zr=zr, r=r), digits=2))
arl2 <- c(round(xewma.arl(l2, c2, 0, zr=zr), digits=2),
          round(DxDewma.arl(l2, c2, deltas, zr=zr, r=r), digits=2))
arl3 <- c(round(xewma.arl(l3, c3, 0, zr=zr, r=r), digits=2),
          round(DxDewma.arl(l3, c3, deltas, zr=zr, r=r), digits=2))
data.frame(delta=c(0, deltas), arl1, arl2, arl3)
```

xDgrsr.arl *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

```
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 (give fast initial response).
sided	distinguish between one- and two-sided Shiryaev-Roberts schemes by choosing "one" and "two", respectively. Currentlly, 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 implicitely 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	define whether the first observation used for the ARL calculatio follows already 1*delta or 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

F. F. Gan (1991), EWMA control chart under linear drift, *J. Stat. Comput. Simulation 38*, 181-200.

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

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

```
## 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=zr, 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)
arls <- round(DxDgrsr.arl(k, g, deltas, zr=zr, r=r), digits=4)
data.frame(deltas, arls)
```

`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

F. F. Gan (1991), EWMA control chart under linear drift, *J. Stat. Comput. Simulation 38*, 181-200.

L. A. Aerne, C. W. Champ and S. E. Rigdon (1991), Evaluation of control charts under linear trend, *Commun. Stat., Theory Methods 20*, 3341-3349.

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

```
## Aerne et al. (1991)
## Table I (continued)
## original numbers are
#     delta arl1of1 arl2of3 arl4of5  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 <- 3.069/3
c2of3 <- 2.1494/2
c4of5 <- 1.14
c10   <- 3.2425/3
DxDshewhartrunsrules.arl <- Vectorize(xDshewhartrunsrules.arl, "delta")
deltas <- 10^(-(18:0)/8)
arl1of1 <-
round(DxDshewhartrunsrules.arl(deltas, c=c1of1, type="1"), digits=2)
arl2of3 <-
round(DxDshewhartrunsrules.arl(deltas, c=c2of3, type="12"), digits=2)
arl4of5 <-
round(DxDshewhartrunsrules.arl(deltas, c=c4of5, type="13"), digits=2)
arl10 <-
round(DxDshewhartrunsrules.arl(deltas, c=c10, type="SameSide10"), digits=2)
data.frame(delta=round(deltas, digits=6), arl1of1, arl2of3, arl4of5, arl10)
```

`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,sided="one",limits="fix",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`	in-control mean.
`mu0`	out-of-control mean.
`zr`	reflection border for the one-sided chart.
`sided`	distinguish between one- and two-sided two-sided EWMA control chart by choosing `"one"` and `"two"`, respectively.
`limits`	distinguish 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

`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

S. V. Crowder (1987), A simple method for studying run-length distributions of exponentially weighted moving average charts, *Technometrics 29*, 401-407.

J. M. Lucas, M. S. Saccucci (1990), Exponentially weighted moving average control schemes: Properties and enhancements, *Technometrics 32*, 1-12.

See Also

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

Examples

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

l <- .1
c <- xewma.crit(l,500,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)
## Lucas/Saccucci (1990)
## two-sided EWMA

## with fixed limits
l1 <- .5
l2 <- .03
c1 <- 3.071
c2 <- 2.437
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,3.5,4,5)
ad1 <- sapply(mu,l=l1,c=c1,sided="two",xewma.ad)
ad2 <- sapply(mu,l=l2,c=c2,sided="two",xewma.ad)
round(cbind(mu,ad1,ad2),digits=2)

## original results are (in Table 3)
## 0.00 499.   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
## 3.00   1.91   4.02
## 3.50   1.58   3.49
## 4.00   1.36   3.09
## 5.00   1.10   2.55.
```

`xewma.arl` *Compute ARLs of EWMA control charts*

Description

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

Usage

```
xewma.arl(l,c,mu,zr=0,hs=0,sided="one",limits="fix",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.
- `mu` — true mean.
- `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 implicitely 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 two-sided chart with fixed control limits and of the one-sided chart, `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

Returns a single value which resembles the ARL.

Author(s)

Sven Knoth

References

K.-H. Waldmann (1986), Bounds for the distribution of the run length of geometric moving average charts, *Appl. Statist. 35*, 151-158.

S. V. Crowder (1987), A simple method for studying run-length distributions of exponentially weighted moving average charts, *Technometrics 29*, 401-407.

J. M. Lucas, M. S. Saccucci (1990), Exponentially weighted moving average control schemes: Properties and enhancements, *Technometrics 32*, 1-12.

S. Chandrasekaran, J. R. English, R. L. Disney (1995), Modeling and analysis of EWMA control schemes with variance-adjusted control limits, *IIE Transactions 277*, 282-290.

T. R. Rhoads, D. C. Montgomery, C. M. Mastrangelo (1996), Fast initial response scheme for exponentially weighted moving average control chart, *Quality Engineering 9*, 317-327.

S. H. Steiner (1999), EWMA control charts with time-varying control limits and fast initial response, *Journal of Quality Technology 31*, 75-86.

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

S. Knoth (2004), Fast initial response features for EWMA Control Charts, *Statistical Papers 46*, 47-64.

See Also

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

Examples

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

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

## Crowder (1987), two-sided EWMA with fixed control limits
l1 <- .5
l2 <- .05
c <- 2
mu <- (0:16)/4
arl1 <- sapply(mu,l=l1,c=c,sided="two",xewma.arl)
arl2 <- sapply(mu,l=l2,c=c,sided="two",xewma.arl)
round(cbind(mu,arl1,arl2),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
l1 <- .5
l2 <- .03
c1 <- 3.071
c2 <- 2.437
mu <- c(0,.25,.5,.75,1,1.5,2,2.5,3,3.5,4,5)
arl1 <- sapply(mu,l=l1,c=c1,sided="two",xewma.arl)
arl2 <- sapply(mu,l=l2,c=c2,sided="two",xewma.arl)
round(cbind(mu,arl1,arl2),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.

## 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)
## 0.00 500.   500.   500.   500.
## 0.25 170.09 167.54 105.90  96.6
## 0.50  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)
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)

## 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*(2-l))
arlfix  <- sapply(mu,l=l,c=c,sided="two",xewma.arl)
arlvacl <- sapply(mu,l=l,c=c,sided="two",limits="vacl",xewma.arl)
arlfir  <- sapply(mu,l=l,c=c,hs=c/2,sided="two",limits="fir",xewma.arl)
```

```
arlboth <- sapply(mu,l=l,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)
## 0.0 477.3* 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=l,c=c,sided="two",xewma.arl)
arlvacl <- sapply(mu,l=l,c=c,sided="two",limits="vacl",xewma.arl)
round(cbind(mu,arlfix,arlvacl),digits=1)

## original results are (in Table 2)
## 0.00 1379.0   1353.0
## 0.25  135.0    127.0
## 0.50   37.4     32.5
## 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 <- .03
cfir <- 2.44
cboth <- 2.54
cstein <- 2.55
hsfir <- cfir/2
hsboth <- cboth/2*sqrt(l*(2-l))
mu <- c(0,.5,1,1.5,2,3,4)
arlfir <- sapply(mu,l=l,c=cfir,hs=hsfir,sided="two",limits="fir",xewma.arl)
arlboth <- sapply(mu,l=l,c=cboth,hs=hsboth,sided="two",limits="both",xewma.arl)
arlstein <- sapply(mu,l=l,c=cstein,sided="two",limits="Steiner",xewma.arl)
round(cbind(mu,arlfir,arlboth,arlstein),digits=1)

## 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 <- .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
# 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)
```

`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 (give fast initial response).
`sided`	distinguish between one- and two-sided two-sided EWMA control chart by choosing `"one"` and `"two"`, respectively.
`limits`	distinguish 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

S. V. Crowder (1989), Design of exponentially weighted moving average schemes, *Journal of Quality Technology 21*, 155-162.

See Also

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

Examples

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

```
xewma.q(l, c, mu, p, zr=0, hs=0, sided="one", limits="fix", 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.
`mu`	true mean.
`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, $E_q(L - q + 1 \vert L \geq)$, will be determined. Note that mu0=0 is implicitely 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

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 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

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

Author(s)

Sven Knoth

References

F. F. Gan (1993), An optimal design of EWMA control charts based on the median run length, *J. Stat. Comput. Simulation 45*, 169-184.

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

S. Knoth (2004), Fast initial response features for EWMA Control Charts, *Statistical Papers 46*, 47-64.

K.-H. Waldmann (1986), Bounds for the distribution of the run length of geometric moving average charts, *Appl. Statist. 35*, 151-158.

See Also

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

Examples

```
## 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
#     deplyoing secant rule
xewma.c.of.quantile <- function(l, L0, mu, p, zr=0, hs=0, sided="one", mode="integer", r=40) {
  c2 <- 0
  a2 <- 0
  while ( a2 < L0 ) {
    c2 <- c2 + .5
    a2 <- xewma.q(l, c2, mu, p, zr=zr, hs=hs, sided=sided, r=r)
    if ( mode=="integer" ) a2 <- ceiling(a2)
  }
  c1 <- c2 - .5
  a1 <- xewma.q(l, c1, mu, p, zr=zr, hs=hs, sided=sided, r=r)
  a.error <- 1; c.error <- 1
  while ( a.error>1e-6 && c.error>1e-12 ) {
    c3 <- c1 + (L0 - a1)/(a2 - a1)*(c2 - c1)
    a3 <- xewma.q(l, c3, mu, p, zr=zr, hs=hs, sided=sided, r=r)
    if ( mode=="integer" ) a3 <- ceiling(a3)
    c1 <- c2; c2 <- c3
    a1 <- a2; a2 <- a3
    a.error <- abs(a2 - L0); c.error <- abs(c2 - c1)
  }
  c3
}
XEWMA.c.of.quantile <- Vectorize("xewma.c.of.quantile", "l")

# (ii) re-calculate the thresholds and remove the standardization step
L0 <- 500
G.h.new <- XEWMA.c.of.quantile(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))
```

---

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, mu1, mu0 = 0, zr = 0, sided = "one", r = 30)
```

Arguments

k	reference value of the Shiryaev-Roberts scheme.
g	control limit (alarm threshold) of Shiryaev-Roberts scheme.
mu1	out-of-control mean.
mu0	in-control mean.
zr	reflection border to enable the numerical algorithms used here.
sided	distinguish between one- and two-sided schemes by choosing "one" and"two", respectively. Currently only one-sided schemes are implemented.
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

S. Knoth (2006), The art of evaluating monitoring schemes – how to measure the performance of control charts? S. Lenz, H. & Wilrich, P. (ed.), *Frontiers in Statistical Quality Control 8*, Physica Verlag, Heidelberg, Germany, 74-99.

G. Moustakides, A. Polunchenko, A. Tartakovsky (2009), Numerical comparison of CUSUM and Shiryaev-Roberts procedures for detectin changes in distributions, *Communications in Statistics: Theory and Methods*, to appear.

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

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

```
data.frame(zrs, ads)

## Table 2 from Knoth (2006)
## original values are
#  mu   arl
#  0    689
#  0.5  30
#  1    8.9
#  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)
## original values are
# gamma  A        STADD/steady-state ARL
# 50     28.02    4.37
# 100    56.04    5.46
# 500    280.19   8.33
# 1000   560.37   9.64
# 5000   2801.75  12.79
# 10000  5603.7   14.17
Gxgrsr.ad  <- Vectorize("xgrsr.ad", "g")
As <- c(28.02, 56.04, 280.19, 560.37, 2801.75, 5603.7)
gs <- log(As)
theta <- 1
zr <- -6
ads <- round(Gxgrsr.ad(theta/2, gs, theta, zr=zr, r=100), digits=2)
data.frame(As, ads)
```

---

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", 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 (give fast initial response). If `hs=NULL`, then the classical headstart -Inf is used (corresponds to 0 for the non-log scheme).
`sided`	distinguish between one- and two-sided schemes by choosing `"one"` and `"two"`, respectively. Currently only one-sided schemes are implemented.
`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 single value which resembles the ARL.

Author(s)

Sven Knoth

References

S. Knoth (2006), The art of evaluating monitoring schemes – how to measure the performance of control charts? S. Lenz, H. & Wilrich, P. (ed.), *Frontiers in Statistical Quality Control 8*, Physica Verlag, Heidelberg, Germany, 74-99.

G. Moustakides, A. Polunchenko, A. Tartakovsky (2009), Numerical comparison of CUSUM and Shiryaev-Roberts procedures for detectin changes in distributions, *Communications in Statistics: Theory and Methods*, to appear.

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

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

```
data.frame(zrs, arls)

## 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")
arls <- round(Mxgrsr.arl(k, g, mus, zr=zr), digits=1)
data.frame(mus, arls)

## Table 4 from Moustakides et al. (2009)
## original values are
# gamma  A        ARL/E_infty(L) SADD/E_1(L)
# 50     28.02    50.79          5.46
# 100    56.04    100.79         6.71
# 500    280.19   500.8          9.78
# 1000   560.37   1000.79        11.14
# 5000   2801.75  5001.75        14.34
# 10000  5603.7   10000.78       15.73
Gxgrsr.arl  <- Vectorize("xgrsr.arl", "g")
As <- c(28.02, 56.04, 280.19, 560.37, 2801.75, 5603.7)
gs <- log(As)
theta <- 1
zr <- -6
arls0 <- round(Gxgrsr.arl(theta/2, gs, 0, zr=zr, r=100), digits=2)
arls1 <- round(Gxgrsr.arl(theta/2, gs, theta, zr=zr, r=100), digits=2)
data.frame(As, arls0, arls1)
```

`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

```
xgrsr.crit(k, L0, mu0 = 0, zr = 0, hs = NULL, sided = "one", 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 (give fast initial response). If `hs=NULL`, then the classical headstart -Inf is used (corresponds to 0 for the non-log scheme).
`sided`	distinguish between one- and two-sided schemes by choosing `"one"` and`"two"`, respectively. Currently only one-sided schemes are implemented.
`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

G. Moustakides, A. Polunchenko, A. Tartakovsky (2009), Numerical comparison of CUSUM and Shiryaev-Roberts procedures for detectin changes in distributions, *Communications in Statistics: Theory and Methods*, to appear.

See Also

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

Examples

```
## 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))
```

`xsewma.arl` *Compute ARLs of simultaneous EWMA control charts (mean and variance charts)*

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

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

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`	upper control limit of the variance EWMA control chart.
`df`	actual degrees of freedom, corresponds to batch size.
`mu`	true mean.
`sigma`	true standard deviation.
`hsx`	so-called headstart (give 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 (give fast initial response) of the variance chart.
`Ns`	dimension of the approximating matrix of the variance chart.
`s2.on`	distinguish between S^2 and S chart.
`sided`	distinguish 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

`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 chartas 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

K. Waldmann (1986), Bounds for the distribution of the run length of geometric moving average charts, *J. R. Stat. Soc., Ser. C, Appl. Stat. 35*, 151-158.

F. F. Gan (1995), Joint monitoring of process mean and variance using exponentially weighted moving average control charts, *Technometrics 37*, 446-453.

S. Knoth (2005), Accurate ARL computation for EWMA-S^2 control charts, *Statistics and Computing 15*, 341-352.

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

See Also

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

Examples

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

# process parameters
mu <- 0
sigma <- 1
# batch 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
SIDED <- "upper"

arl <- xsewma.arl(lx, cx, ls, csu, df, mu, sigma, Nx=Nx, Ns=Ns, sided=SIDED)
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

```
xsewma.crit(lx, ls, L0, df, mu0=0, sigma0=1, cu=NULL, hsx=0, hss=1, s2.on=TRUE, sided="upper", mode="fi
```

Arguments

lx	smoothing parameter lambda of the two-sided mean EWMA chart.
ls	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 (give fast initial response) of the mean chart – do not confuse with the true FIR feature considered in xewma.arl; will be updated.
hss	headstart (give fast initial response) of the variance chart.
df	actual degrees of freedom, corresponds to batch size.
s2.on	distinguish between S^2 and S chart.
sided	distinguish 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 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 `cl` and `cu` of the variance EWMA chart.

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

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

Examples

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

# batch 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
```

`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

```
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 observation 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 1, and mu1 must be given.
`c`	normalizing constant to ensure specific alarming behavior.
`type`	controls the type of Shewhart chart used, seed 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

C. W. Champ and W. H. Woodall (1987), Exact results for Shewhart control charts with supplementary runs rules, *Technometrics 29*, 393-399.

See Also

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

Examples

```
## Champ/Woodall (1987)
## Table 1
mus <- (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
Mxshewhartrunsrules.ad <- Vectorize(xshewhartrunsrules.ad, "mu1")
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)
```

Index