R-package “spcadjust”
A short introduction
Version 0.1-1

Axel Gandy and Jan Terje Kvaløy

June 26, 2015

This packages is still in relatively early stages of development.

This document describes briefly how to use the R-package which implements the algorithm for “Guaranteed Conditional Performance of Control Charts via Bootstrap Methods.” based on Gandy and Kvaløy [2013].

Some information on how to install the package is in Appendix A. Information on how to access help can be found in Appendix B. The package is loaded by

```
library(spcadjust)
```

1 Some simple standard usage

1.1 CUSUM charts with normality assumption

The following is a simple application for CUSUM charts, assuming that all observations are normally distributed.

Based on n past in-control observations $X_{-n}, \dots, X_{-1}$, the in-control mean is estimated by $\hat{\mu} = \frac{1}{n}\sum_{i=-n}^{-1} X_i$ and the in-control variance by $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=-n}^{-1}(X_i - \hat{\mu})^2$. Based on new observations $X_1, X_2, \dots$, the CUSUM chart is then defined by

$$S_0 = 0, \quad S_t = \max(0, \frac{S_{t-1} + X_t - \hat{\mu} - \Delta/2}{\hat{\sigma}}).$$

The following defines the control chart.

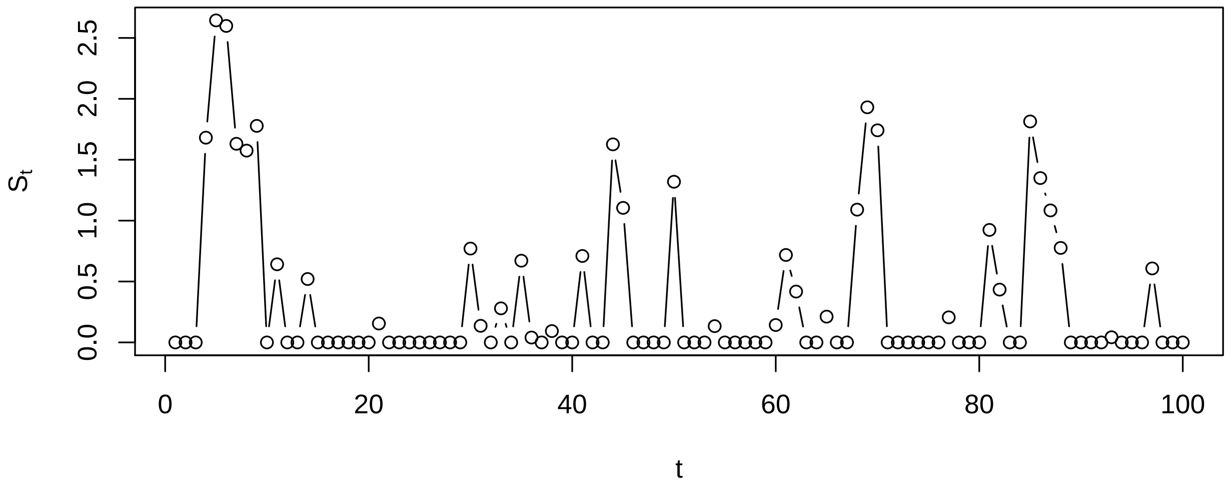
```
chart <- new("SPCCUSUMNormal",Delta=1);
```

Then we generate a data set of past observations and compute the resulting estimates for running the chart.

```
X <-  rnorm(250)
xihat <- xiofdata(chart,X)
str(xihat)

## List of 3
##  $ mu: num 0.0749
##  $ sd: num 0.977
##  $ m : int 250
```

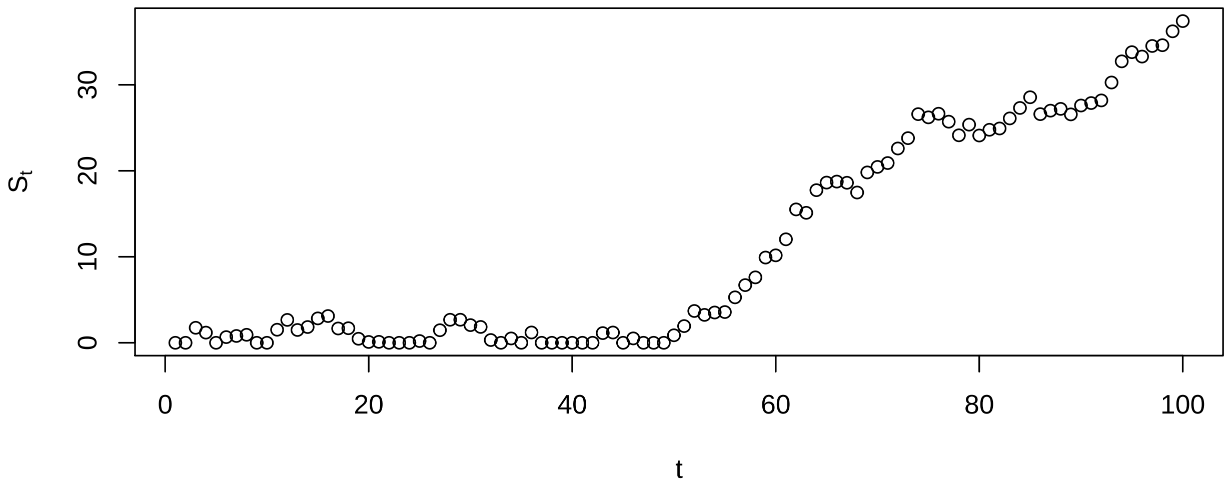
Next, we run the chart with new observations (that happen to be in-control).

```
plot(runchart(chart, newdata=rnorm(100),xi=xihat),ylab=expression(S[t]),xlab="t",type="b")
```



In the next example, the chart is run with data that is out-of-control from time 51 onwards.

```
plot(runchart(chart, newdata=rnorm(100,mean=c(rep(0,50),rep(1,50))),xi=xihat),ylab=expression(S[t]),xla
```



The following computes confidence intervals for various properties of the chart. This is based on parametric resampling assuming normality of the observations. You should increase the number of bootstrap replications (the argument nrep) for real applications.

The first computes the threshold with a method that with roughly 90% probability results in an average run length of at least 100. The second computes a threshold such that with probability 90% this gives a false alarm probability of 5%.

```
SPCproperty(data=X,nrep=50,
            property=new("calARLCUSUM",chart=chart,target=100))

## 90 % CI: A threshold of 3.284 gives an in-control ARL of at least
##   100.
## Unadjusted result:  2.79
## Based on  50 bootstrap repetitions.

SPCproperty(data=X,nrep=50,
            property=new("calhitprobCUSUM",chart=chart,target=0.05,nsteps=1000))

## 90 % CI: A threshold of 9.63 gives an in-control false alarm
##   probability of at most 0.05 within 1000 steps.
## Unadjusted result:  7.853
## Based on  50 bootstrap repetitions.
```

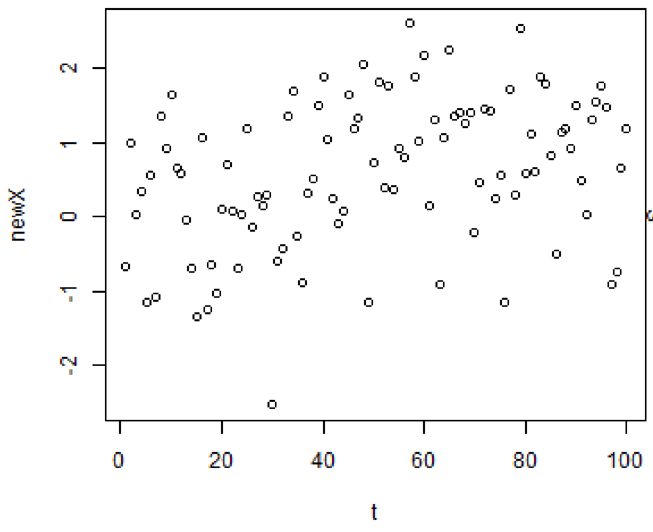
The next two examples compute confidence intervals for ARL and hitting probabilities for certain thresholds.

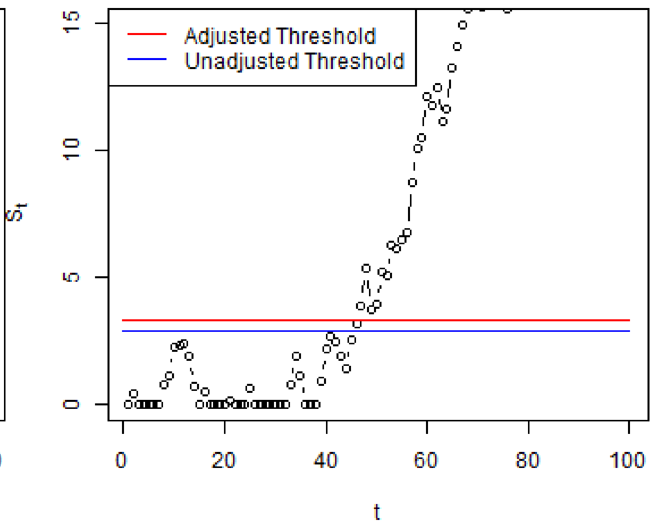
```
SPCproperty(dat=X,nrep=50,
            property=new("ARLCUSUM",chart=chart,threshold=3),
            covprob=c(0.8,0.9))

## 80 % CI: A threshold of 3 gives an in-control ARL of at least
##   103.6.
## 90 % CI: A threshold of 3 gives an in-control ARL of at least
##   80.71.
## Unadjusted result:  125.7
## Based on  50 bootstrap repetitions.

SPCproperty(dat=X,nrep=50,
            property=new("hitprobCUSUM",chart=chart,threshold=5,nsteps=100),
            covprob=c(0.8,0.9))

## 80 % CI: A threshold of 5 gives an in-control false alarm
##   probability of at most 0.1424 within 100 steps.
## 90 % CI: A threshold of 5 gives an in-control false alarm
##   probability of at most 0.1721 within 100 steps.
## Unadjusted result:  0.08784
## Based on  50 bootstrap repetitions.
```

Finally to plot the chart with the threshold that guarantees a certain ARL.

```
cal <- SPCproperty(data=X,nrep=50,
            property=new("calARLCUSUM",chart=chart,target=100))
newX <- rnorm(100)
S <- runchart(chart, newdata=newX,xi=xihat)
```

```
plot(S,ylab=expression(S[t]),xlab="t",type="b",ylim=range(S,cal@res+1,cal@raw))
lines(c(0,100),rep(cal@res,2),col="red")
lines(c(0,100),rep(cal@raw,2),col="blue")
legend("topleft",c("Adjusted Threshold","Unadjusted Threshold"),col=c("red","blue"),lty=1)
```





1.2 CUSUM chart using nonparametric resampling

The following example now calibrates using nonparametric resampling (resampling the observations with replacement), which would be robust against model misspecifications.

```
chartnp <- new("SPCCUSUMNonparCenterScale",Delta=1)
SPCproperty(data=X,
            nrep=100,property=new("calARLCUSUM",chart=chartnp,target=100))

## 90 % CI: A threshold of 3.251 gives an in-control ARL of at least
##   100.
## Unadjusted result:  2.678
## Based on  100 bootstrap repetitions.
```

1.3 Shewhart charts with normality assumptions

Based on n past in-control observations $X_{-n}, \dots, X_{-1}$, the in-control mean is estimated by $\hat{\mu} = \frac{1}{n}\sum_{i=-n}^{-1} X_i$ and the in-control variance by $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=-n}^{-1}(X_i - \hat{\mu})^2$. Based on new observations $X_1, X_2, \dots,$ a two-sided chart is then defined by

$$S_t = \frac{X_t - \hat{\mu}}{\hat{\sigma}}.$$

and signals when $|S_t| > c$ for some threshold c.

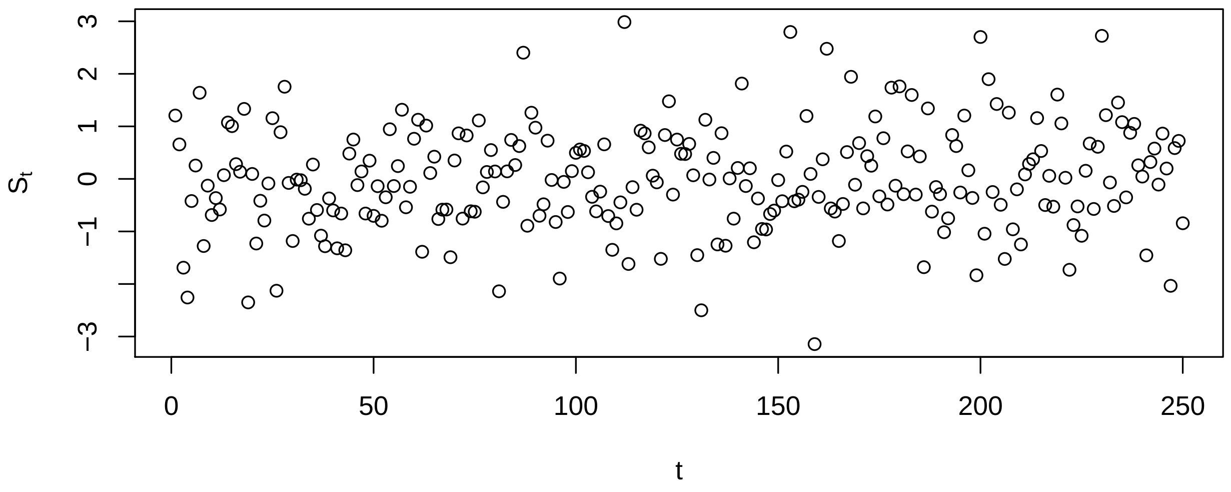
```
chartShew <- new("SPCShewNormalCenterScale",twosided=TRUE)
```

Generate some new data and estimate the parameters to run the chart.

```
X <-  rnorm(250)
xihat <- xiofdata(chartShew,X)
```

Plotting the chart based on the new data.

```
plot(runchart(chartShew, newdata=X,xi=xihat),ylab=expression(S[t]),xlab="t")
```



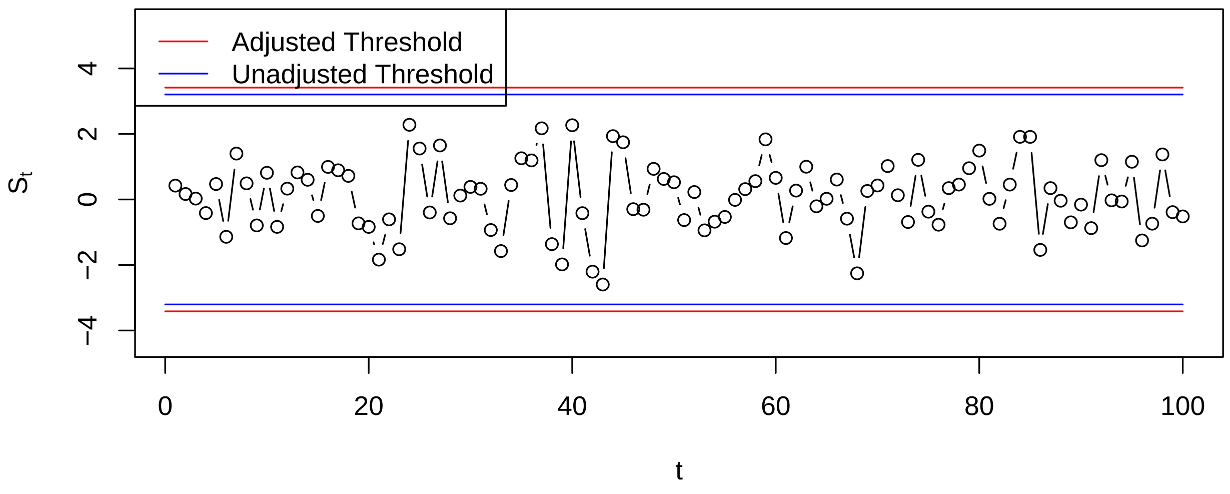
Computing various properties of the chart, adjusted for estimation error.

```
SPCproperty(data=X,nrep=100,
            property=new("calARLShew",chart=chartShew,target=741))

## 90 % CI: A threshold of 3.439 gives an in-control ARL of at least
##   741.
## Unadjusted result:  3.205
## Based on  100 bootstrap repetitions.

SPCproperty(data=X,nrep=100,
            property=new("ARLShew",chart=chartShew,threshold=3))

## 90 % CI: A threshold of 3 gives an in-control ARL of at least
##   222.8.
## Unadjusted result:  370.4
## Based on  100 bootstrap repetitions.

SPCproperty(data=X,nrep=100,
            property=new("hitprobShew",chart=chartShew,nsteps=100,threshold=3))

## 90 % CI: A threshold of 3 gives an in-control false alarm
##   probability of at most 0.3566 within 100 steps.
## Unadjusted result:  0.2369
## Based on  100 bootstrap repetitions.

SPCproperty(data=X,nrep=100,
            property=new("calhitprobShew",chart=chartShew,target=0.01,nsteps=100))

## 90 % CI: A threshold of 4.113 gives an in-control false alarm
##   probability of at most 0.01 within 100 steps.
## Unadjusted result:  3.889
## Based on  100 bootstrap repetitions.
```

To plot a chart with adjusted thresholds added.

```
cal <- SPCproperty(data=X,nrep=100,
            property=new("calARLShew",chart=chartShew,target=741))
S <- runchart(chartShew, newdata=newX,xi=xihat)
```

```
plot(S,ylab=expression(S[t]),xlab="t",type="b",ylim=range(S,cal@res+2,cal@raw,-cal@res-1,-cal@raw))
lines(c(0,100),rep(cal@res,2),col="red")
lines(c(0,100),rep(cal@raw,2),col="blue")
lines(c(0,100),-rep(cal@res,2),col="red")
lines(c(0,100),-rep(cal@raw,2),col="blue")
legend("topleft",c("Adjusted Threshold","Unadjusted Threshold"),col=c("red","blue"),lty=1)
```



2 Linear regression example

Suppose one observes past data $(Y_{-n}, X_{-n}), \ldots, (Y_{-1}, X_{-1})$, where Y_i is a response of interest and X_i is a corresponding vector of covariates. The parameter β of the linear model $\mathrm{E}\,Y = X\beta$ is estimated via the function lm (which does least squares estimation) and gives an estimator $\hat{\beta}$. The corresponding risk adjusted CUSUM chart to detect a shift of $\Delta > 0$ in the mean of the response for new observations $(Y_1, X_1), \ldots, (Y_n, X_n)$ is then defined by

$$S_0 = 0, \quad S_t = \max\left(0, S_{t-1} + Y_t - X_t\hat{\beta} - \Delta/2\right).$$

where Δ would be the minimal mean-shift of interest.

First create some past data.

```
n <- 1000
Xlinreg <- data.frame(x1= rbinom(n,1,0.4), x2= runif(n,0,1), x3= rnorm(n))
Xlinreg$y <- 2 + Xlinreg$x1 + Xlinreg$x2 + Xlinreg$x3 + rnorm(n)
```

Defining a chart for a specific linear model and finding the threshold that would give an ARL of 100.

```
chartlinreg <- new("SPCCUSUMlm",Delta=1,formula="y~x1+x2+x3")
SPCproperty(data=Xlinreg,
            nrep=100,
            property=new("calARLCUSUM",chart=chartlinreg,target=100))

## 90 % CI: A threshold of 3.517 gives an in-control ARL of at least
##   100.
## Unadjusted result:  3.179
## Based on  100 bootstrap repetitions.
```

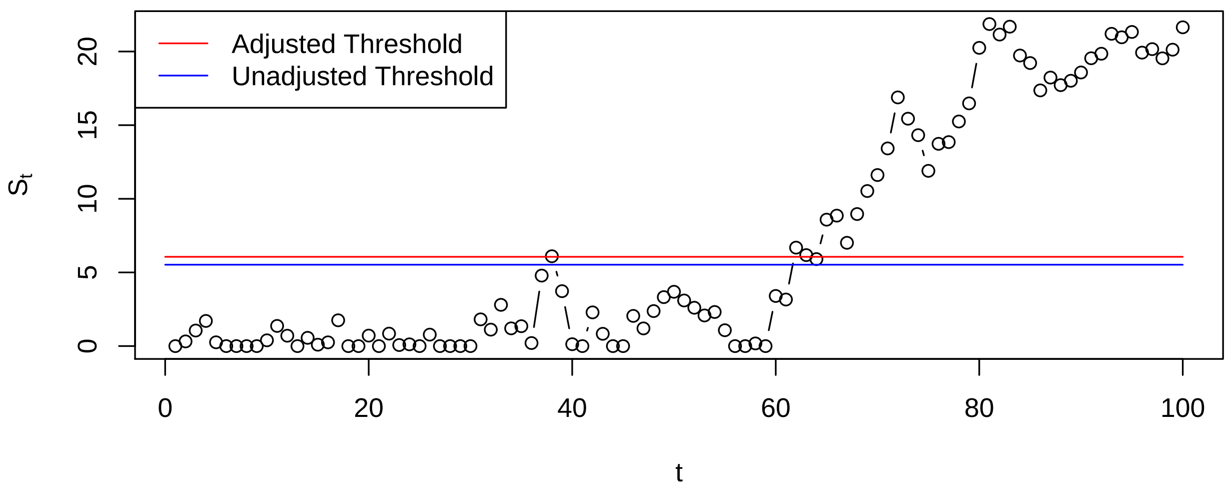
Same as before but for a different linear model.

```
chartlinreg <- new("SPCCUSUMlm",Delta=1,formula="y~x1")
SPCproperty(data=Xlinreg,
            nrep=100,
            property=new("calARLCUSUM",chart=chartlinreg,target=100))

## 90 % CI: A threshold of 6.068 gives an in-control ARL of at least
##   100.
## Unadjusted result:  5.524
## Based on  100 bootstrap repetitions.
```

Finally, running the chart with new data (that are out of control from observation 50) and plotting the chart.

```
xihat <- xiofdata(chartlinreg,Xlinreg)
cal <- SPCproperty(data=Xlinreg,
             nrep=100,
             property=new("calARLCUSUM",chart=chartlinreg,target=100))
n <- 100
newXlinreg <- data.frame(x1= rbinom(n,1,0.4), x2= runif(n,0,1), x3=rnorm(n))
newXlinreg$y <- 2 + newXlinreg$x1 + newXlinreg$x2 + newXlinreg$x3 + rnorm(n)+c(rep(0,50),rep(1,50))
S <- runchart(chartlinreg, newdata=newXlinreg,xi=xihat)
```

```
plot(S,ylab=expression(S[t]),xlab="t",type="b",ylim=range(S,cal@res+1,cal@raw))
lines(c(0,100),rep(cal@res,2),col="red")
lines(c(0,100),rep(cal@raw,2),col="blue")
legend("topleft",c("Adjusted Threshold","Unadjusted Threshold"),col=c("red","blue"),lty=1)
```



3 Logistic regression example

Suppose one observes past in-control data $(Y_{-n}, X_{-n}), \ldots, (Y_{-1}, X_{-1})$, where Y_i is a binary response variable and X_i is a corresponding vector of covariates. Suppose that in control $\text{logit}(\text{P}(Y_i = 1|X_i)) = X_i\xi$. The function glm can be used to obtain an estimate $\hat{\xi}$.

For detecting a change to $\text{logit}(\text{P}(Y_i = 1|X_i)) = \Delta + X_i\xi$, a CUSUM chart can be defined based on the cumulative sum of likelihood ratios of the in-control versus out-of-control model [Steiner et al., 2000] by

$$S_t = \max(0, S_{t-1} + R_t), \quad S_0 = 0,$$

where

$$\exp(R_t) = \frac{\exp(\Delta + X_t\xi)^{Y_t}/(1 + \exp(\Delta + X_t\xi))}{\exp(X_t\xi)^{Y_t}/(1 + \exp(X_t\xi))} = \exp(Y_t\Delta)\frac{1 + \exp(X_t\xi)}{1 + \exp(\Delta + X_t\xi)}.$$

Create an example data set of past observations.

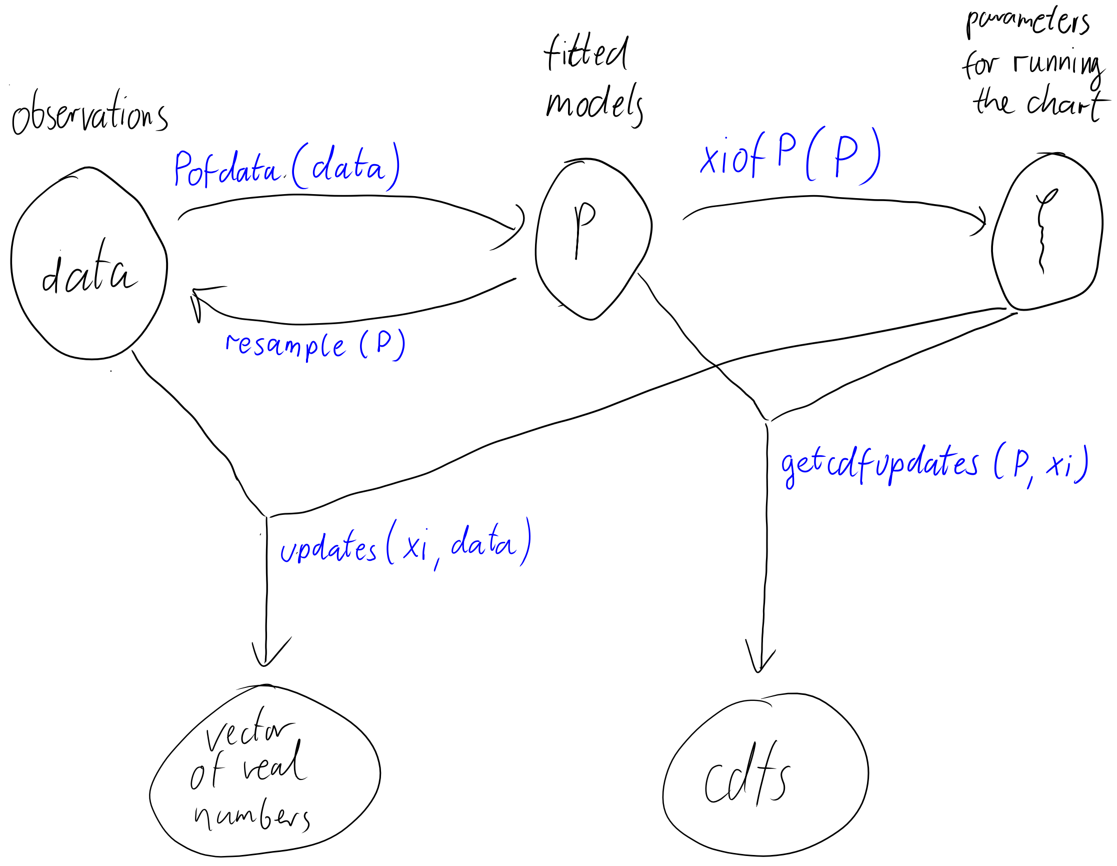
```
n <- 1000
Xlogreg <- data.frame(x1=rbinom(n,1,0.4), x2=runif(n,0,1), x3=rnorm(n))
xbeta <- -1+Xlogreg$x1*100+Xlogreg$x2+Xlogreg$x3
Xlogreg$y <- rbinom(n,1,exp(xbeta)/(1+exp(xbeta)))
```

Computing the threshold to give an in-control ARL of 100.

```
chartlogreg <- new("SPCCUSUMlogreg",Delta= 1, formula="y~x1+x2+x3")
SPCproperty(data=Xlogreg,
            nrep=100,
            property=new("calARLCUSUM",chart=chartlogreg,target=100))

## 90 % CI: A threshold of 2.017 gives an in-control ARL of at least
##   100.
## Unadjusted result:  1.781
## Based on  100 bootstrap repetitions.
```

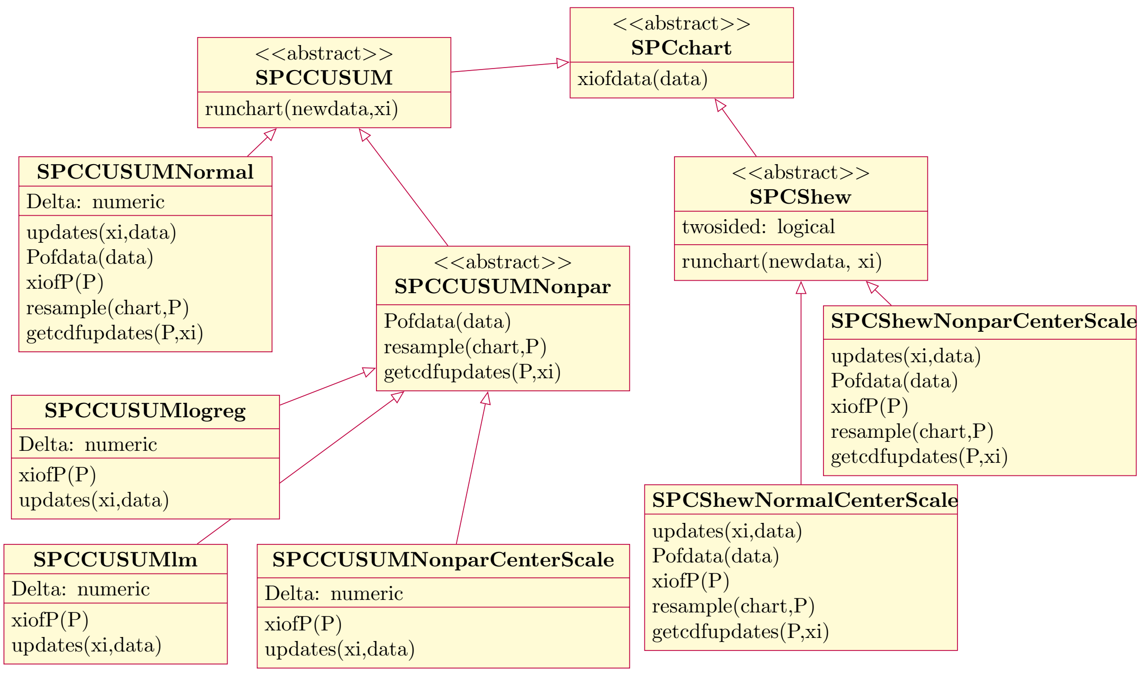
4 Methods needed for SPCproperty

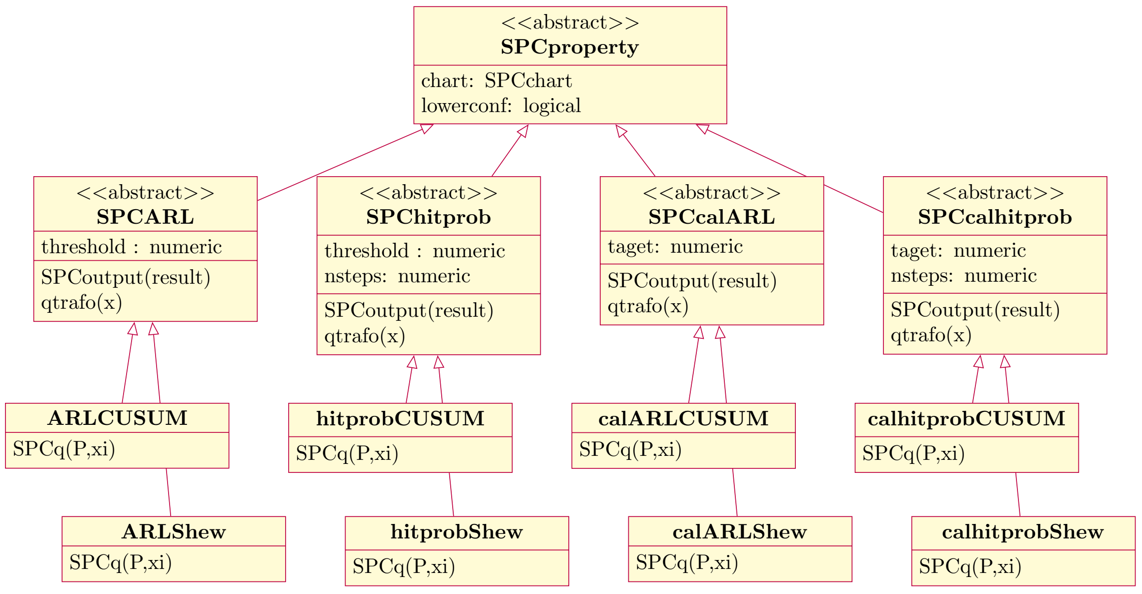
The main function SPCproperty needs a definition of the chart as an S4 class that has the methods marked in blue in the following diagram.



5 Class structure

The following slightly simplified UML diagram gives an overview over the main classes provided by the package.





6 Extensions

The framework provided in this package can be easily extended to work with different charts and/or estimation procedures. The following are some examples of extensions.

6.1 Modifying CUSUM to use robust estimation

The following defines a CUSUM chart using normality assumptions to estimate the in-control distribution using Median and MAD. Only the method Pofdata needs to be redefined, the rest is as in SPCCUSUMNormal.

```
setClass("SPCCUSUMNormalROBUST", contains="SPCCUSUMNormal")
setMethod("Pofdata", "SPCCUSUMNormalROBUST",
          function(chart,data){
              list(mu= median(data), sd= mad(data), m=length(data))
          })
```

Properties of this chart can then be computed as before:

```
X <-  rnorm(100)
chartrobust <- new("SPCCUSUMNormalROBUST",Delta=1)
SPCproperty(data=X,nrep=50,
            property=new("calARLCUSUM",chart=chartrobust,target=100))

## 90 % CI: A threshold of 4.489 gives an in-control ARL of at least
##   100.
## Unadjusted result:  2.755
## Based on  50 bootstrap repetitions.
```

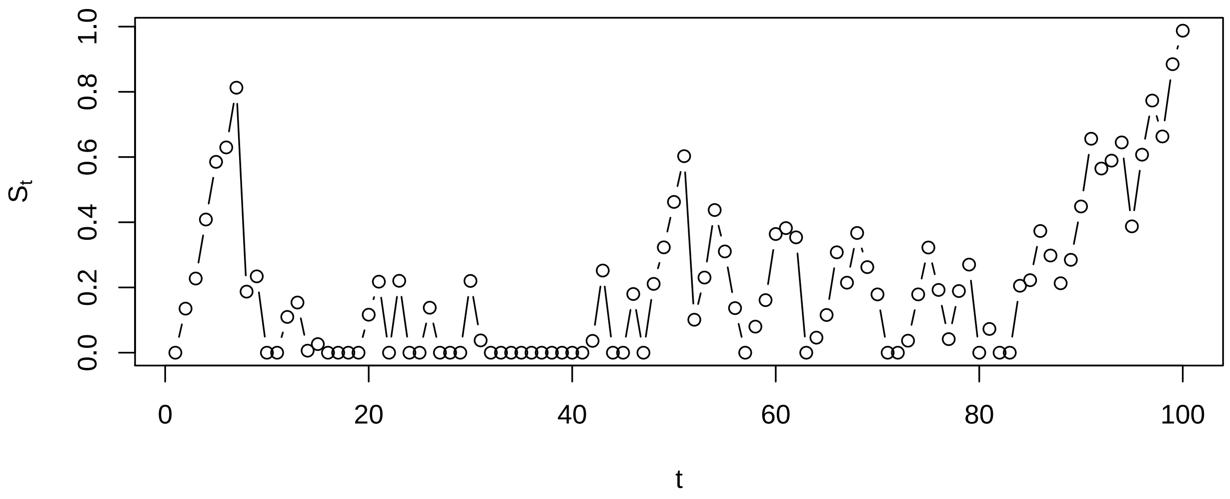
6.2 Parametric exponential CUSUM chart

The following defines a class that defines a CUSUM chart for exponentially distributed data with parametric resampling.

```
setClass("SPCCUSUMExponential",contains="SPCCUSUM",representation(Delta="numeric"))
setMethod("Pofdata", "SPCCUSUMExponential",
          function(chart,data){
              list(lambda=1/mean(data), n=length(data))
          })
setMethod("xiofP", "SPCCUSUMExponential",
          function(chart,P) P$lambda)
setMethod("resample", "SPCCUSUMExponential",
          function(chart,P) rexp(P$n,rate=P$lambda))
setMethod("getcdfupdates", "SPCCUSUMExponential",
          function(chart, P, xi) {
              ; function(x){ if(chart@Delta<1)
                pmax(0,1-exp(-P$lambda*(x-log(chart@Delta))/(xi*(1-chart@Delta))))
              else
                pmin(1,exp(-P$lambda*(log(chart@Delta)-x)/(xi*(chart@Delta-1))))
            }
          })
setMethod("updates","SPCCUSUMExponential",
          function(chart,xi,data) log(chart@Delta)-xi*(chart@Delta-1)*data
)

ExpCUSUMchart=new("SPCCUSUMExponential",Delta=1.25)
```

The following greates some past observations and then runs a chart for 100 steps with new observations.

```
X <- rexp(1000)
plot(runchart(ExpCUSUMchart, newdata=rexp(100),xi=xiofdata(ExpCUSUMchart,X)),
     ylab=expression(S[t]),xlab="t",type="b")
```



The following computes various properties of the chart.

```
SPCproperty(data=X,
            nrep=100,
            property=new("hitprobCUSUM",chart=ExpCUSUMchart,
              threshold=1,nsteps=100),covprob=c(0.5,0.9))

## 50 % CI: A threshold of 1 gives an in-control false alarm
##   probability of at most 0.9049 within 100 steps.
## 90 % CI: A threshold of 1 gives an in-control false alarm
##   probability of at most 0.9356 within 100 steps.
## Unadjusted result:  0.8922
## Based on  100 bootstrap repetitions.

SPCproperty(data=X,
            nrep=100,
            property=new("ARLCUSUM",chart=ExpCUSUMchart,
              threshold=3),covprob=c(0.5,0.9))

## 50 % CI: A threshold of 3 gives an in-control ARL of at least
##   893.9.
## 90 % CI: A threshold of 3 gives an in-control ARL of at least
##   507.6.
## Unadjusted result:  891.5
## Based on  100 bootstrap repetitions.

SPCproperty(data=X,
            nrep=100,
            property=new("calARLCUSUM",chart=ExpCUSUMchart,
              target=1000),covprob=c(0.5,0.9))
```

```
## 50 % CI: A threshold of 3.198 gives an in-control ARL of at least
##   1000.
## 90 % CI: A threshold of 3.667 gives an in-control ARL of at least
##   1000.
## Unadjusted result:  3.163
## Based on  100 bootstrap repetitions.
```

References

Axel Gandy and Jan Terje Kvaløy. Guaranteed conditional performance of control charts via bootstrap methods. *Scandinavian Journal of Statistics*, 40:647–668, 2013. doi: 10.1002/sjos.12006.

Stefan H. Steiner, Richard J. Cook, Vern T. Farewell, and Tom Treasure. Monitoring surgical performance using risk-adjusted cumulative sum charts. *Biostat*, 1(4):441–452, 2000. doi: 10.1093/biostatistics/1.4.441. URL `http://biostatistics.oxfordjournals.org/cgi/content/abstract/1/4/441`.

A Installation

The installation is as for most R-packages that do not reside in CRAN. The general procedure is described in the Section 6 on "Add-on packages" in the R Manual on Istallation and Administration:
`http://cran.r-project.org/doc/manuals/R-admin.html`.

The following is merely an adaptation of those procedures to our package.

A.1 Windows

Download the package "spcadjust_0.1-1.zip". In the graphical evironment (Rgui) use the menue option : Packets ... Install packet from local zip-file.

A.2 Linux/Unix

If you do not have write access to the package repository:

1. Download the package "simctest_0.1-1.tar.gz" and place it into your home directory.
2. If needed, create a local package directory via the following commands.

   ```
   bash
   mkdir ~/Rlibrary
   echo ".libPaths(\"$HOME/Rlibrary\")" >$HOME/.Rprofile
   ```

3. Install the package

   ```
   R CMD INSTALL  simctest_???.tar.gz
   ```

   where ??? needs to be replaced by 0.1-1.

B Accessing the help files

This document can be accessed via

```
vignette("spcadjust-intro")
```

Documentation of the most useful command can be obtained as follows:

```
> ? SPCproperty
```

(Rudimentary) documentation of the S4-classes in this package can be obtained e.g. via

```
> class ? SPCCUSUMNormal
```