

Package ‘qAnalyst’

August 23, 2009

Type Package

Title Control Charts, Capability and Distribution Identification

Version 0.6.0

Date 2009-08-19

URL www.quantide.com

Author Andrea Spano’

Maintainer Andrea Spano’ <andrea.spano@quantide.com>

Description qAnalyst performs: control charts for variables and attributes according to Douglas C. Montgomery Introduction to Statistical Quality Control book, Capability analysis for normal and non - normal distributions and Distributions Identification

License GPL (>= 2)

Depends lattice, MASS, alr3, SuppDists

Repository CRAN

Date/Publication 2009-08-23 07:06:37

R topics documented:

qAnalyst-package	2
andersonDarlingFun	3
blemish	4
boxcoxFun	5
brakeCap	6
capabilityNormal	7
capabilityNotNormal	9
cranks	10
faults	11
funInfoFun	12
invCpFun	13

johnsonFun	14
paretoChart	15
plot.capability	16
plot.spc	17
plot.transformation	18
print.capability	19
print.spc	20
print.transformation	21
probplot	22
rapidFitFun	24
rawWeight	25
spc	26
summary.capability	28
summary.infoFun	29
summary.spc	30
toyCarsDefects	31
toys	32
tubes	33
warpTiles	34

Index 35

qAnalyst-package *Package to create and analyze control charts and to perform capability analysis*

Description

Creates xbar, s, r, i, mr, p, np, c, u chart. Perform capability analysis for both normal and non normal data.

Details

Package: qAnalyst
 Type: Package
 Version: 0.5.1
 Date: 2008-02-12
 License: GPL -2

Function `spc` and `capability` create objects of class `spc` and `capability` respectively. Charts of type `xbar`, `r`, `s`, `i`, `mr`, `p`, `np`, `c` and may be handled by `spc` objects. Generic methods: `plot`, `print` and `summary` exist for `spc` and `capability` objects.

Author(s)

Andrea Spano' and Giorgio Spedicato Maintainer: andrea.spano@quantide.com

References

Montgomery, Statistical Quality Control

Examples

```
#a simple x bar chart
#data cranks

data(cranksh)
x=cranksh$crankshaft
sg=cranksh$workingDay

go<-spc(x=x,sg=sg,type="xbar")
plot(go)
print(go)
summary(go)
rm(go)

#capability example

data(cranksh)
x=cranksh$crankshaft
sg=cranksh$workingDay

go<-capabilityNormal(x=x,sg=sg,lsl=-10, usl=10, target=0)
plot(go)
print(go)
summary(go)
rm(go)
```

andersonDarlingFun *Function to obtain Anderson Darling statistics and p-value*

Description

andersonDarlingFun returns a vector containing AD test statistics and corresponding p-value, from given data and distribution specification.

Usage

```
andersonDarlingFun(x, distribution, theta)
```

Arguments

x	a vector of data
distribution	character string specifying hypothesized distribution
theta	vector of parameter, coherent with distribution specification

Details

andersonDarlingFun currently supports following distributions: normal, lognormal, gamma, weibull, logistic.

Value

A vector of two elements: statistics and p-value

Note

No notes

Author(s)

Andrea Spano', Enrico Pegoraro, Giorgio Spedicato

References

Stephens, M.A., eds.: Goodness-of-Fit Techniques. Marcel Dekker, New York.

See Also

[funInfoFun](#)

Examples

```
data(warpTiles)
x=warpTiles$warping
infoX=funInfoFun(x, "weibull")
andersonDarlingFun(x=x, infoX$densfun, infoX$theta)
```

blemish

Blemishes on sampled squares of fabric

Description

Blemish dataframe comes from a linen manufacturing. 100 square yards of fabric are sampled and blemishes are counted.

Usage

```
data(blemish)
```

Format

A data frame with 40 observations on the following 2 variables.

blemish number of blemishes counted

linenSampled sample id

Details

Data from blemish example are suitable to be projected onto a C control chart.

Source

Industrial data.

References

This type of data can be used within attribute chart presentations.

Examples

```
data(blemish)
cchart=spc(x=blemish$blemish, type="c", name="blemish")
plot(cchart)
```

 boxcoxFun

Function to obtain Box Cox transformations

Description

boxcoxFun performs box cox transformation, finding optimal lambda for the given data vector. boxcoxFun returns a transformation class object.

Usage

```
boxcoxFun(x)
```

Arguments

x numeric data vector

Details

boxcoxFun calls internally powtran and bctrans1 functions from package alr3 to build output transformation object.

Value

A transformation class object is returned. A transformation class objects consists in a list composed by the following items:

type	type of transformation, in such case: "boxcox"
parameters	parameter lambda is stored in this item
original	original data vector
parameters	trasformed data vector

Note

Box cox transformations requires data to be positive. X vector positivity is internally checked before performing transformation.

Author(s)

Giorgio Spedicato

References

Venables, Ripley Modern Applied Statistics with S-PLUS

See Also

`print.transformation`, `plot.transformation`, `johnsonFun`

Examples

```
#warpTiles example
data(warpTiles)
boxInfo=boxcoxFun(x=warpTiles$warping)
print(boxInfo)
```

brakeCap

Brake capability dataset

Description

BrakeCap dataframes contains shoes brakes physical proprieties: hardness, centering, quencing. 50 batches of 5 elements each have been sampled.

Usage

```
data(brakeCap)
```

Format

A data frame with 250 observations on the following 4 variables:

hardness brake hardness
centering brake centering
quencing brake quencing
subgroup brake subgroup

Details

This dataset contains both measurement data both subgroups information

Source

Industrial data

References

Industrial data

Examples

```
data(brakeCap)
#x-bar control chart
hadnessSPCxb=spc(x=brakeCap$hardness, sg=brakeCap$subgroup, type="xbar", name="hardness")
plot(hadnessSPCxb)
summary(hadnessSPCxb)
#r control chart
hadnessSPCr=spc(x=brakeCap$hardness, sg=brakeCap$subgroup, type="r", name="hardness")
plot(hadnessSPCr)
summary(hadnessSPCr)
#process in control
#ok for
#capability analysis
hadnessCap=capabilityNormal(x=brakeCap$hardness, sg=brakeCap$subgroup, lsl=39, usl=41, target=40)
summary(hadnessCap)
plot(hadnessCap)
```

capabilityNormal *Computing Capability Analysis for Normal data*

Description

capabilityNormal is used to compute capability analysis for normal distributed data. This function creates a capability class object for normal data. plot, print and summary methods are available for capability class objects.

Usage

```
capabilityNormal(x, sg=length(x), lsl = NULL, usl = NULL, target = NULL, name = dep
```

Arguments

x	Numeric data vector
sg	Subgroup specification. It represents rational sub-group of observations. It can be specified either as a vector, of the same length of x of ordered subgroup id, or as a constant indicating subgroup size. Default is sg=length(x).
lsl	Lower specification limit.
usl	Upper specification limit.
target	Process target.

name	Variable label to be used within graphics, default taken by "x" name.
toler	Width of tolerance specification. Default is 6
historicalMean	Mean of the population distribution if a known process parameter is available or an estimate obtained from past data. If mean value is not specified, it is estimated from data
historicalSd	Standard Deviation of the population distribution if a known process parameter is available or an estimate obtained from past data. If value of standard deviation is not specified, it is estimated from data

Details

Either lsl or usl must be specified.

Value

An object of class capability

Note

No notes

Author(s)

Andrea Spano'

References

Bothe (1997), Measuring Process Capability, McGraw Hill

See Also

[plot.capability](#), [print.capability](#), [summary.capability](#)

Examples

```
data (brakeCap)
capObj=capabilityNormal(x=brakeCap$hardness,sg=brakeCap$subgroup, lsl=39, usl=41, target=40,
summary (capObj)
```

`capabilityNotNormal`*Capability object creator for non - normal data*

Description

`capabilityNotNormal` function creates capability class object for non - normal data given specification limits. `plot`, `print`, `summary` methods are available for capability class objects.

Usage

```
capabilityNotNormal(x, lsl = NULL, usl = NULL, target = NULL, distribution, name =
```

Arguments

<code>x</code>	a numeric data vector.
<code>lsl</code>	lower specification limit.
<code>usl</code>	upper specification limit.
<code>target</code>	process target.
<code>distribution</code>	distribution of the variable.
<code>name</code>	name of the variable, default taken by "x" name.
<code>toler</code>	width of tolerance specification. Default is 6, that is six sigma.
<code>histPars</code>	a vector of two elements that allows user to specify historical mean and standard deviation respectively.

Details

Either `lsl` or `usl` must be specified. Hypothized distribution must be specified. `capabilityNotNormal` is a wrapper of many internal functions, that check user specifications consistency and perform calculation of statistics. Results are stored in a capability object.

Value

A capability object list is returned

Note

Creates the core capability objects

Author(s)

Giorgio Spedicato

References

Bothe (1997), *Measuring Process Capability*, McGraw Hill

See Also

[plot.capability](#), [print.capability](#), [summary.capability](#)

Examples

```
data(warpTiles)
capObj=capabilityNotNormal(x=warpTiles$warping, usl=8, target=40, distribution="weibull", na
summary(capObj)
```

cranks

Cranks data set

Description

Cranks dataframe comes from an engines assembly department, where the deviation from an ideal position of a parts of the crankshaft are measured daily.

Usage

```
data(cranks)
```

Format

A data frame with 125 observations on the following 2 variables.

crankshaft Distance in millimeters from the actual point of the crankshaft to the baseline position.

workingDay Working day of the relevation. This is the subgroup id.

Details

Subgroups dimensions are not equal.

Source

Industrial data.

References

Cranks dataset can be used to show subgroups charts capabilities

Examples

```
data(cranks)
xbarchart=spc(x=cranks$crankshaft, sg=cranks$workingDay, type="xbar", name="crankshaft")
rbarchart=spc(x=cranks$crankshaft, sg=cranks$workingDay, type="r", name="crankshaft")
plot(xbarchart)
plot(rbarchart)
```

`faults`*Variability of faults during week shift*

Description

Faults dataframe contains measured faults during weekly shifts. There are three shifts per week.

Usage

```
data(faults)
```

Format

A data frame with 100 observations on the following 2 variables.

faults fault measures.

shift shift id.

Details

Each shift can be used as subgroups id.

Source

Industrial data.

References

Faults dataframe can be used to show an application of s-bar chart.

Examples

```
data(faults)
schart=spc(x=faults$faults, sg=faults$shift, type="s", name="faults")
plot(schart)
```

funInfoFun

Function to estimate distribution quantities

Description

funInfoFun receives a vector of data and a hyphotized distribution. It estimates hyphotized distribution parameters. Anderson Darling statistics is provided, if desired.

Usage

```
funInfoFun(x, fun, adStats = TRUE)
```

Arguments

x	a vector of data
fun	name of function density. This name is processed internally by switchFun.
adStats	boolean value to specify if Anderson Darling statistics shall be calculated.

Details

funInfoFun uses switchFun to parse given fun argument into R internal distribution nomenclature. funInfoFun calls MASS function fitdistr to estimate parameters (via maximum likelihood). If required andersonDarlingFun function is called to obtain goodnes of fit statistic and corresponding p-value. The returned list contains items that are the corresponding distribution functions to obtain random numbers, quantiles and density.

Value

A list containing the following items:

densfun	distribution name as specified by user
theta	named vector containing user specification
theta	named vector containing parameter specification
qfun	corresponding internal r function to obtain quantiles
dfun	corresponding internal r function to obtain density
rfun	corresponding internal r function to obtain random values
pfun	corresponding internal r function to obtain CDF

Warning

funInfoFun estimates parameters by maximum likelihood. Numerical MLE may not always converge. Warning message are thrown in such case. Moreover, user shall check that x vector values lie in the natural support of the distribution specified in fun

Note

funInfoFun returns a vector of class "infoFun".

Author(s)

Giorgio Spedicato

References

Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth edition. Springer.

See Also

[andersonDarlingFun](#), [probplot](#)

Examples

```
data(warpTiles)
infoX=funInfoFun(x=warpTiles$warping, fun="weibull")
str(infoX)
```

invCpFun

Function to get inverse capability tolerance limits

Description

invCpFun returns tolerance limits for a given data vector, provided a desired cp.

Usage

```
invCpFun(x, cp = 1, fun = "normal")
```

Arguments

x	a vector of data
cp	desired cp
fun	specified distribution, default normal

Details

Cp is supposed to be equal to cp, that is $cp=cpk$.

Value

A vector containing lsl and usl for given data.

Note

No notes

Author(s)

Giorgio Spedicato and Andrea Spano'

References

Bothe (1997), Measuring Process Capability, McGraw Hill

See Also

[capability](#), [funInfoFun](#)

Examples

```
#brakeCap example
data(brakeCap)
invCpFun(x=brakeCap$hardness, cp=1, fun="normal")
```

johnsonFun

Function to perform Johnson transformations

Description

johnsonFun performs Johnson transformation and returns a corresponding transformation object for given data.

Usage

```
johnsonFun(x)
```

Arguments

x a vector of numeric data

Details

johnsonFun wraps JohnsonFit function from SuppDists package.

Value

A transformation class object is returned. transformation class objects are lists composed by following items:

type	type of transformation, in such case: "johnson"
parameters	a vector or a list of parameters
original	original data vector
parameters	trasformed data vector

Note

`print` and `plot` methods are available for such class

Author(s)

Giorgio Spedicato

References

Johnson, N.L. (1949). Systems of frequency curves generated by methods of translation. *Biometrika*

See Also

[print.transformation](#), [plot.transformation](#), [boxcoxFun](#)

Examples

```
data(warpTiles)
warpJohnson=johnsonFun(warpTiles$warping)
print(warpJohnson)
```

paretoChart

function to plot Pareto charts

Description

`paretoChart` plots Pareto charts for categorical variables. Plot is performed by lattice graphics.

Usage

```
paretoChart(x, mergeThr = 0.9, addLine = TRUE, abbrev = FALSE)
```

Arguments

<code>x</code>	data vector treated as character variable
<code>mergeThr</code>	cumulated probability after which categories are merged into generic category "Other".
<code>addLine</code>	boolean. If true, cumulated frequency line is plot out
<code>abbrev</code>	boolean. If true, abbreviation of categories names written on x axis is done to obtain a better print.

Details

Different categories are sorted by absolute frequency in decreasing order. Categories with same absolute frequency are sorted according to their appearing order in the given vector.

Value

paretoChart returns no value

Note

A great part of this code comes from pareto.chart function of Luca Scrucca's qcc package.

Author(s)

Giorgio Spedicato

References

Montgomery, statistical quality control

See Also

No see also

Examples

```
#MASS data set example
require(MASS)
data(Cars93)
paretoData=Cars93$Manufacturer[1:45]
paretoChart(x=paretoData,mergeThr=.8)
```

plot.capability *Plot method for class capability*

Description

plot.capability implements generic plot method for capability class objects. Plot of capability objects is performed by trellis graphics.

Usage

```
## S3 method for class 'capability':
plot(x, ...)
```

Arguments

x an object of class capability.
... further arguments to be added

Details

plot.capability makes use of trellis function histogram.

Value

This function return no value.

Note

Generic S3 function method

Author(s)

Giorgio Spedicato

References

Lattice package help manual

See Also

[capability](#)

Examples

```
#creates capability obj
data(brakeCap)
capObj=capabilityNormal(x=brakeCap$hardness, sg=brakeCap$subgroup, usl=41.5)
plot.capability(capObj)
```

plot.spc

Plot method for spc class objects

Description

plot.spc implements generic plot method for spc class objects. Plot of spc object is performed by trellis graphics.

Usage

```
## S3 method for class 'spc':
plot(x, cex = list(cexStrip=2, cexAxes=2, cexScales=2, cexPoints=1.5), ...)
```

Arguments

x	An object of class spc
cex	A list containing cex parameter for Strip, Axes, Scales and Points
...	further arguments to be added

Details

plot.spc requires lattice library.

Value

No value is returned

Note

An spc object is required

Author(s)

Andrea Spano'

References

Lattice package help manual

See Also

[spc](#), [print](#)

Examples

```
#brakeCap
data(brakeCap)
x=brakeCap$hardness
sg=brakeCap$subgroup
go=spc(x=x, sg=sg, type="xbar")
plot(go)
```

plot.transformation

plot method for class transformation

Description

plot.transformation implements generic plot method for transformation class objects. Plot of transformation objects is performed by trellis graphics.

Usage

```
## S3 method for class 'transformation':
plot(x, ...)
```

Arguments

x an object of class transformation.
... further arguments to be added.

Details

`plot.transformation` makes use of trellis function histograms. Two histograms are plotted onto the same graph. Leftward the original data distribution is plotted. Rightward the trasformed data distribution is plotted.

Value

No values are returned

Note

Generic S3 function method

Author(s)

Giorgio Spedicato

References

Lattice package help manual

See Also

[johnsonFun](#), [boxcoxFun](#)

Examples

```
#warpTiles data set
data(warpTiles)
johnsonWarp=johnsonFun(warpTiles$warping)
plot(johnsonWarp)
```

`print.capability` *Print method for class capability*

Description

`print.capability` implements generic print method for class `capability`. Summary variable statistics and overall capability indexes are printed out.

Usage

```
## S3 method for class 'capability':
print(x, ...)
```

Arguments

`x` an object of class `capability`
`...` further arguments to be added

Details

NA value is returned for any index not computed.

Value

No values are returned

Note

Output returned is a concise version of [summary.capability](#) output.

Author(s)

Giorgio Spedicato

References

Bothe (1997), Measuring Process Capability, McGraw Hill

See Also

[summary.capability](#)

Examples

```
data(brakeCap)
x=brakeCap$quencing
sg=brakeCap$subgroup
capObj=capabilityNormal(x=x,sg=sg, lsl=41,usl=42, target=41.5, name="example data")
print.capability(capObj)
```

print.spc

Print method for spc object

Description

print.spc implements generic print function to spc object.

Usage

```
## S3 method for class 'spc':
print(x, ...)
```

Arguments

x an object of class spc
... further arguments to be added

Details

`prints.spc` prints out brief informations on `spc` object.

Value

This function does not return any value.

Note

Printed informations are general statistics.

Author(s)

Andrea Spano'

References

No references

See Also

[spc](#), [plot.spc](#)

Examples

```
#data brakeCap

data(brakeCap)
x=brakeCap$quencing
sg=brakeCap$subgroup

go=spc(x=x,sg=sg,type="xbar", testType=2, k=0,p=0,nSigma=0)
plot.spc(go)
print.spc(go)
rm(go)
```

```
print.transformation
```

Print method for class capability

Description

`print.transformation` implements generic print method for class `transformation`. Type of performed transformation and estimated parameters are printed out.

Usage

```
## S3 method for class 'transformation':
print(x, ...)
```

Arguments

x an object of class transformation
... further arguments to be added

Details

No details

Value

No values are returned

Note

No notes

Author(s)

Andrea Spano'

References

No references

See Also

[boxcoxFun](#), [johnsonFun](#), [plot.transformation](#)

Examples

```
data(warpTiles)
boxcoxWarp=boxcoxFun(warpTiles$warping)
print(boxcoxWarp)
```

probplot

Function to plot probability plots

Description

probplot plots probability plot for a vector of values, given distribution specifications. Confidence intervals and a legend can be plot too.

Usage

```
probplot(x, distribution, theta, confintervals = FALSE, confidence = 0.95, name =
```

Arguments

<code>x</code>	a vector of values.
<code>distribution</code>	character specifying distribution to plot
<code>theta</code>	vector of parameter. Estimated by the function if left missing
<code>confintervals</code>	logical item, indicating whether confidence interval have to be plotted or not.
<code>confidence</code>	confidence $1 - \alpha$ level.
<code>name</code>	name of variable plotted.

Details

(x,y) values represent respectively original data and corresponding quantiles of theoretical estimated distribution. If the theoretical distribution fits data well, values should lie around the straight line plotted.

Value

probplot returns no value

Note

No notes

Author(s)

Giorgio Spedicato

References

No references

See Also

[funInfoFun](#)

Examples

```
data(warpTiles)
```

```
probplot(x=warpTiles$warping,distribution="weibull", confintervals=TRUE, confidence=.95, na
```

`rapidFitFun`*Function to obtain rapid fitting of multiple distributions*

Description

`rapidFitFun` fits parameter on data vector `x` for some of most important continuous distributions. Fitted parameters values and names are printed out in a table together. Anderson Darling goodness of fit statistics p-value is moreover provided.

Usage

```
rapidFitFun(x, rounding = 3, boxcox=FALSE, johnson=FALSE)
```

Arguments

<code>x</code>	a vector of countinuous values. All values must be positive
<code>rounding</code>	rounding digits to obtain pretty table printing
<code>boxcox</code>	Boolean. If true, boxcox normality transformation is performed
<code>johnson</code>	Boolean. If true, johnson normality transformation is performed

Details

`rapidFitFun` uses `funInfoFun` functionalities to estimate parameters and AD p-value. If either `boxcox` or `johnson` are TRUE, estimated AD statistics p-value is related to trasformed data.

Value

`rapidFitFun` returns no value

Note

`rapidFitFun` is a wrapper for `funInfoFun`. Numerical procedures to obtain parameters MLE may not always converge. In such cases warnings and errors message may be thrown.

Author(s)

Giorgio Spedicato

References

Graybill Mood, Introduction to Statistics, Mc-Graw Hill

See Also

[funInfoFun](#), [andersonDarlingFun](#)

Examples

```
##brakeCap
data(brakeCap)
rapidFitFun(brakeCap$centering)
#warpTiles
data(warpTiles)
rapidFitFun(warpTiles$warping)
```

rawWeight

Raw material bars weight

Description

rawWeight dataframe contains individual weight measurements of bars of raw material.

Usage

```
data(rawWeight)
```

Format

A data frame with 45 observations on the following 2 variables.

rawWeight weight of each bar. Measured in pounds.

id id number of measurement.

Details

Data from this example are suitable to be projected onto individual values control charts.

Source

Industrial data.

References

This dataset can be used to shown examples of individual charts.

Examples

```
data(rawWeight)
head(rawWeight)
ichart=spc(x=rawWeight$rawWeight,sg=2,type="i",name="weight")
plot(ichart)
mrchart=spc(x=rawWeight$rawWeight,sg=2,type="mr",name="weight")
plot(mrchart)
```

spc

*Function to create spc object***Description**

spc function creates spc class objects from data. Generic methods plot, print, summary are available for SPC class objects.

Usage

```
spc(x, sg, type = "xbar", name = deparse(substitute(x)), testType = 1, k = NA, p =
```

Arguments

x	data vector
sg	Subgroup variable. This argument has a different meaning according to chart type. xbar, r and s charts: sg represents rational sub-group of observations. It can be specified either as a vector, of the same length of x of ordered subgroup id, or as a constant value that will be replicated along x. Subgroup dimension must be greather than one. i and mr charts: sg represents the width of the window over which moving range are computed. Must be a scalar. Default value is set to two. p, np and u charts: sg represents sample size of observation. It can be specified either as a vector, of the same length of x or as a constant value that will be replicated along x. c chart: sg is not required
type	Chart type. String identifying chart type. Charts can be of type: "xbar", "r", "s", "i", "mr", "p", "np", "u", "c".
name	Name of x variable as it appears in the charts. By default the name of the given x variable
testType	A vector or a scalar of test codes to be performed. Eight tests are implemented corrensponding to codes 1-8. Default performed test is 1. test 1: k points out of p points beyond nSigma*sd from center line. Default values are k=1 p=1 nSigma=3 test 2: k points out of p points in a row on same side of center line. Default values are: k=9 p=9 test 3: k points out of p points in a row either all increasing or decreasing. Default values: k=6 p=6 test 4: k points out of p points in a row alternating up and down. Default values: k= 14 p=14 test 5: k points out of p points in a row beyond nSigma*sd from center line. Default values are: k=2 p=3 nSigma = 3

	test 6: k points out of p points in a row beyond $n\text{Sigma} \cdot \text{sd}$ from center line (one side of center line). Default values: $k=4$ $p = 5$ $n\text{Sigma}=2$
	test 7: k points out of p points in a row within $n\text{Sigma} \cdot \text{sd}$ (both sides of centre line). Default values are: $k = 15$ $p=15$ $n\text{Sigma}=1$
	test 8: k points out of p points in a row beyond $n\text{Sigma} \cdot \text{sd}$ (both sides of central line). Default values are $k=8$ $p = 8$ $n\text{Sigma}=1$
k	A vector or a scalar of parameters to be used by tests. If k is not specified, tests are performed with k default values. If k is specified, its length must be of the same length as <code>testType</code> .
p	A vector or a scalar of parameters to be used by tests. If p is not specified, tests are performed with p default values. If p is specified, its length must be of the same length as <code>testType</code> .
$n\text{Sigma}$	A vector or a scalar of parameters to be used by tests. If $n\text{Sigma}$ is not specified, tests are performed with $n\text{Sigma}$ default values. If $n\text{Sigma}$ is specified, its length must be of the same length as <code>testType</code> .

Details

`spc` function performs coherence tests on subgroups dimension and chart name. Then it calculates graphical parameters, statistics values and test results and stores this values in lists.

Value

An object of class `spc`

Note

No notes

Author(s)

Andrea Spano'

References

Montgomery, Statistical Quality Control

See Also

[plot.spc](#), [print.spc](#)

Examples

```
#i-chart, moving range to estimate st. dev. is equal to 2 points with testType=1,
data(rawWeight)
ichart=spc(x=rawWeight$rawWeight, sg=2, type="i", name="weight", testType=1)
plot(ichart)
summary(ichart)
```

summary.capability *Summary method for class capability*

Description

summary.capability implements generic summary method for capability class objects. A thorough report of performed analysis results is printed out. Three sections are printed: summary variables statistics, estimated ppm, capability index.

Usage

```
## S3 method for class 'capability':  
summary(object, printPotential = TRUE, printOverall = TRUE, printPerc = FALSE, printZeta = TRUE, ...)
```

Arguments

object	an object of capability obj
printPotential	ask use if potential statistics shall be printed, default true.
printOverall	ask use if overall statistics shall be printed, default true.
printPerc	ask uses if percentages have to be used in place of ppm, default false.
printZeta	ask uses if zeta values have to be used in place of capability statistics, default false.
...	further arguments to be added

Details

summary.capability expands print.capability output.

Value

No value is returned

Note

This function is an S3 primitive

Author(s)

Giorgio Spedicato

References

Bothe (1997), Measuring Process Capability, McGraw Hill

See Also

[print.capability](#)

Examples

```
data (brakeCap)
x=brakeCap$hardness
sg=brakeCap$subgroup
capObj=capabilityNormal(x=x,sg=sg, lsl=39,usl=41, target=40, name="example data")
summary.capability(capObj)
```

summary.infoFun *Generic summary method for infoFun objects*

Description

summary.infoFun implements generic summary method on infoFun objects. Estimated parameters names and values are print out, with AD statistics if calculated

Usage

```
## S3 method for class 'infoFun':
summary(object, ...)
```

Arguments

object	An object of class infoFun
...	furter arguments to be added

Details

summary.infoFun output is build from infoFun objects internal list items.

Value

No values are returned

Note

No notes

Author(s)

Giorgio Spedicato

References

No references

See Also[funInfoFun](#)**Examples**

```
#warpTiles data
data(warpTiles)
infoX=funInfoFun(warpTiles$warping, "weibull")
summary(infoX)
```

`summary.spc`*Generic summary method for spc objects*

Description

`summary.spc` implements generic summary method on `spc` object. `summary.spc` prints general statistics, the chart elements coordinates and performed test details.

Usage

```
## S3 method for class 'spc':
summary(object, ...)
```

Arguments

<code>object</code>	an object of class <code>spc</code>
<code>...</code>	further arguments to be added

Details

`summary.spc` function extracts information from the lists embedded internally in the `SPC` object given.

Value

This function does not return any value.

Note

Default printed informations are general statistics.

Author(s)

Giorgio Spedicato

References

Final user function

See Also

[spc](#), [plot.spc](#), [print.spc](#)

Examples

```
set.seed(100)
x=c(rnorm(30,m=10,s=1), rnorm(30,m=12,s=1), rnorm(30,m=10,s=4))
sg=sort(rep(1:30,3))
#1 LA + SEMPLICE
go=spc(x=x,sg=sg,type="xbar", testType=2, k=0,nSigma=0, p=0)
plot.spc(go)
summary.spc(go)
rm(go)
```

toyCarsDefects	<i>Toys car defects</i>
----------------	-------------------------

Description

toyCarsDefects dataframe reports sample size and the total number of defectives on sampled toy cars in a toy manufacture.

Usage

```
data(toyCarsDefects)
```

Format

A data frame with 20 observations on the following 2 variables.

defects number of defects found.

samp1ed sample size.

Details

Data contained in toyCarsDefects dataframe can be used to show attribute charts proprieties.

Source

Industrial data.

References

This dataset can be shown by a u - chart

Examples

```
data(toyCarsDefects)
head(toyCarsDefects)
chart=spc(x=toyCarsDefects$defects, sg=toyCarsDefects$sampled, name="defects", type="u")
plot(chart)
```

toys

Toys manufacturing data

Description

toys dataframe reports control quality results from a toys factory.

Usage

```
data(toys)
```

Format

A data frame with 80 observations on the following 7 variables.

day day of inspection

weight weight of the sampled toy

length length of the sampled toy

defects defects on sampled toys

sample sampled toys

rejects number of toys rejected in inspected toys

inspected dimension of inspection sample

Details

Fields have different lengths, due coming from different inspection processes.

Source

Industrial data.

References

This dataset can be use to plot different types of control charts. Fields "rejects" and "inspected" may be used to plot np control charts.

Examples

```
data(toys)
npchart=spc(x=na.omit(toys$rejects), sg=na.omit(toys$inspected), type="np", name="Toys rejec")
plot(npchart)
```

tubes

Tubes for television

Description

Tubes dataframe contains data from control quality inspections of tubes for television.

Usage

```
data(tubes)
```

Format

A data frame with 20 observations on the following 2 variables.

rejects number of rejected devices during quality inspection.

sampled sampled devices during quality inspection.

Details

Results of twenty quality inspections are reported.

Source

Industrial data.

References

Content of this dataset can be shown by p-chart.

Examples

```
data(tubes)
pchart=spc(x=tubes$rejects, sg=tubes$sampled, type="p", name="proportion of defective tv tubes")
plot(tubes)
```

`warpTiles`*Warp tiles dataframe*

Description

WarpTiles dataframe contains measurement about warp in a tiles factory.

Usage

```
data(warpTiles)
```

Format

A data frame with 100 observations on the following variable.

warping a numeric vector

Details

Tiles distribution is well fitted by a Weibull or a Gamma distribution.

Source

Industrial data.

References

warpTiles dataframe may be used to show capability for non - normal data.

Examples

```
data(warpTiles)
plot(capabilityNotNormal(x=warpTiles$warping, usl=8, distribution="weibull", name="WARP"))
```

Index

*Topic **datasets**

blemish, 4
brakeCap, 6
cranks, 10
faults, 11
rawWeight, 25
toyCarsDefects, 31
toys, 32
tubes, 33
warpTiles, 34

*Topic **methods**

andersonDarlingFun, 3
boxcoxFun, 5
capabilityNormal, 7
capabilityNotNormal, 8
funInfoFun, 12
invCpFun, 13
johnsonFun, 14
paretoChart, 15
plot.capability, 16
plot.spc, 17
plot.transformation, 18
print.capability, 19
print.spc, 20
print.transformation, 21
probplot, 22
rapidFitFun, 24
spc, 26
summary.capability, 28
summary.infoFun, 29
summary.spc, 30

*Topic **package**

qAnalyst-package, 1

andersonDarlingFun, 3, 13, 24

blemish, 4
boxcoxFun, 5, 15, 19, 22
brakeCap, 6

capability, 14, 17
capabilityNormal, 7
capabilityNotNormal, 8
cranks, 10
faults, 11
funInfoFun, 3, 12, 14, 23, 24, 30

invCpFun, 13

johnsonFun, 5, 14, 19, 22

paretoChart, 15
plot.capability, 8, 9, 16
plot.spc, 17, 21, 27, 31
plot.transformation, 5, 15, 18, 22
print, 18
print.capability, 8, 9, 19, 29
print.spc, 20, 27, 31
print.transformation, 5, 15, 21
probplot, 13, 22

qAnalyst (*qAnalyst-package*), 1
qAnalyst-package, 1

rapidFitFun, 24
rawWeight, 25

spc, 18, 21, 26, 31
summary.capability, 8, 9, 20, 28
summary.infoFun, 29
summary.spc, 30

toyCarsDefects, 31
toys, 32
tubes, 33

warpTiles, 34