URL: http://cxc.harvard.edu/ciao3.4/grpadaptive.html
Last modified: December 2006

*AHELP for CIAO 3.4*

# grpAdaptive

Context: group

***Jump to:*** Description Examples Bugs See Also

## Synopsis

Group an array by the number of counts per group using an adaptive scheme.

## Syntax

```
grpAdaptive( Array_Type countsArray, Integer_Type numCounts )
grpAdaptive( Array_Type countsArray, Integer_Type numCounts,
Integer_Type maxLength )
grpAdaptive(Array_Type countsArray, Integer_Type minCounts,
Integer_Type maxLength, Array_Type tabStops )

Returns: ( Array_Type grouping, Array_Type quality )
```

## Description

This function returns the grouping and quality arrays that represent the input data (countsArray) after it has been adaptively grouped so that each group contains at least numCounts counts. The optional parameters maxLength and tabStops represent the maximum number of elements that can be combined and an array representing those elements that should be ignored respectively.

This function provides the same functionality as the ADAPTIVE option of dmgroup.

## Example 1

```
chips> (g,q) = grpAdaptive( y, 20 )
```

This example calculates the grouping and quality arrays that represent the input data (here the contents of the y array) after it has been adaptively grouped to at least 20 counts per group.

## Example 2

```
chips> x = [0.5:6.0:0.05]
chips> y = 3 + 30 * exp( − (x−2.0)^2 / 0.1 )
```

```
chips> (g,q) = grpAdaptive( y, 15 )
chips> ysum = grpGetGroupSum( y, g )
chips> nchan = grpGetChansPerGroup( g )
chips> i = where( g == 1 )
chips> yavg = ysum[i] / nchan[i]
chips> curve( x, y )
chips> simpleline
chips> curve( x[i], yavg )
chips> symbol square
chips> symbol red
```

Here we take the function

```
y = 3 + 30 * exp( −(x−2)^2 / 0.1 )
```

and adaptively group it by 15 counts per group. The plot shows the original data (the solid line and the crosses) and the grouped data (as the red squares); the latter has been normalised by the width of each group and is displayed at the left−edge of each group.

Unlike the simple grouping done by grpNumCounts() − where only the end element(s) may have non−zero quality values − the adaptive grouping scheme can create groups with non−zero quality anywhere in the array. The code below identifies these points and marks them with a solid−yellow circle.

```
chips> i = where( g == 1 )
chips> j = where( q[i] != 0 )
chips> curve( x[i][j], yavg[j] )
chips> symbol bigpoint
chips> symbol yellow
```

# Bugs

See the bugs page for the group library on the CIAO website for an up−to−date listing of known bugs.

# See Also

*group*

grpadaptive, grpadaptivesnr, grpbin, grpbinfile, grpbinwidth, grpgetchanspergroup, grpgetgroupsum, grpgetgrpnum, grpmaxslope, grpminslope, grpnumbins, grpnumcounts, grpsnr

*modules*

group



URL: http://cxc.harvard.edu/ciao3.4/grpadaptive.html
Last modified: December 2006