Histogram_ext

Introduction to histogram extended

Package *histogram_ext* contains user contributed code based on the *draw* package and the *descriptive* package. Both packages are loaded automatically when loading histogram_ext.mac and two histogram structures (histogram_structure,) will be defined, please see chapter XXX and YYY.

The main goal is to provide more options to express statistical data, especially in form of bubble graphs. Therefor it overs a couple of functions to gain histogram distributions from data in list as well as matrix form. This distributions are covered in a structure that can be later accessed in an easy and clean way.

The graphical part offers a basic bubble function as a generic function for *draw2d* and an easier function for presenting histogram data in a bubble plot. All function have several options to influence the appearance of the bubbles as well as there position in the graph.

Histogram Extensions

By Marcus Menzel (2020)

Function for making histogram structures from data line or rows in matrices for further use.

histogram_ext (LIST , Options) [Function]

Makes a histogram structure of the form histogram_structure. See below for a detailed description of the content of the structure.

- histogram_ext (LIST)
- histogram_ext (LIST , integer)
- histogram_ext (LIST, bin-list)

Returns the histogram of LIST. LIST can be a flat list of integers, floats or rationals and matrices of rank 1 are accepted. The length of LIST must be >2 obvious otherwise an error is trigged.

Providing a LIST only sets the number of symmetric bin according to an algorithm described by Freedman and Diaconis "n the histogram as a density estimator: L_2 theory. " in Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete. Band 57, Nr. 4, 1981, S. 453.

Providing a LIST and an integer sets the number of symmetric bin between the lowest and highest values from LIST.

In case the parameter is a nested list given all numbers in the form [[low_1, high_1], [low_2,high_2],...] it will count the bins accordingly. A zero distance is ok, but the list needs to be sorted in ascending order to make sure no unwanted counting happens. If the classes don't match all list members, the total value in the resulting structure will differ.

The output is a structure with the internal lists (bounds, distance, abs_count, mass_fraction, area_fraction, min, max, total), the last three give the minimal, maximal and total sum of values from the LIST.

Example:

```
load(“histogram_ext”) $

list_test : map( lambda ( [x], (x-50) ),
                 makelist( random(10000)/100, i, 1, 5000 )
            )                                       $
remvalue( example )                                 $
example   : new( histogram_structure )              $
example   : histogram_ext( list_test )              ;
histogram_plot ( example, 'mass )                   $
example   : histogram_ext( list_test, 40 )          ;
histogram_plot ( example, 'mass )                   $

list_bins : [ [0,1],[1,2],[2,3],[4,4],[5,5],
              [51/10,55/10],[8.0,9.9]
            ]                                       $
remvalue( example )                                 $
example   : new( histogram_structure )              $
example   : histogram_ext( list_test, list_bins )   ;
histogram_plot ( example, 'area )                   $

remvalue( example )                                 $
example   : new( histogram_structure )              $
example   : histogram_ext( list_test,
                     makelist([i-1,i+1],i,0,14,2)
            )                                       ;
histogram_plot ( example )                          $
```