

histogram, v. 0.13: Generate Histogram

Bernhard Haubold

Max-Planck-Institute for Evolutionary Biology, Plön, Germany

November 6, 2018

1 Introduction

I often need a quick method for drawing histograms from lists of numbers. This is what the program `histogram` is for.

2 Getting Started

`histogram` was written in C on a computer running Linux and should work on any standard UNIX system. However, please contact me at haubold@evolbio.mpg.de if you have any problems with the program.

- Unpack the program

```
tar -xvzf histogram_XXX.tgz
```

where XXX indicates the version.

- Change into the newly created directory

```
cd Histogram_XXX
```

and list its contents

```
ls
```

- Generate histogram

```
make
```

- List its options

```
./histogram -h
```

3 Tutorial

- We begin with the simple command

```
histogram Data/sample1.dat
```

- If necessary, install the package `plotutils` on your computer using, for example

```
sudo apt-get install plotutils
```

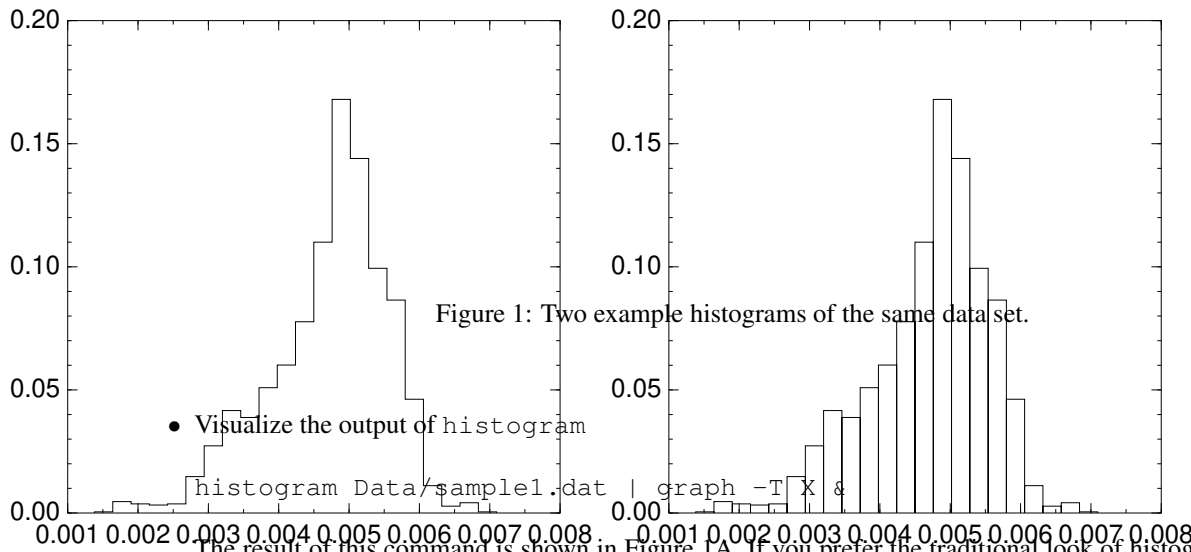
A**B**

Figure 1: Two example histograms of the same data set.

- Visualize the output of histogram

histogram Data/sample1.dat | graph -T X &

The result of this command is shown in Figure 1A. If you prefer the traditional look of histograms you can use

histogram -B Data/sample1.dat | graph -T X &

to generate Figure 1B.

4 Listing

The following listing documents the driver program for histogram.

```

1  /***** histogram.c *****/
   * Description: Compute histogram.
   * Author: Bernhard Haubold, haubold@evolbio.mpg.de
   * Date: Fri Jun 8 12:09:43 2012
   *****/
6  #include <stdio.h>
   #include <stdlib.h>
   #include <float.h>
   #include <gsl/gsl_histogram.h>
   #include "stringUtil.h"
11  #include "histogram.h"
   #include "interface.h"
   #include "eprintf.h"

   void printLinePlot(gsl_histogram *hist){
16     size_t i, n;
       double x, l, u, p;

```

```

    n = gsl_histogram_bins(hist);
    for(i=0; i<n; i++){
21     gsl_histogram_get_range(hist, i, &l, &u);
        x = gsl_histogram_get(hist, i);
        p = l + (u-l) / 2.;
        printf("%g\t%g\n",p,x);
    }
26 }

void printStepPlot(gsl_histogram *hist){
    size_t i, n;
    double x, l, u;
31
    n = gsl_histogram_bins(hist);
    if(n>1){
        gsl_histogram_get_range(hist, 0, &l, &u);
        printf("%g\t0\n", l);
36    }
    for(i=0; i<n; i++){
        gsl_histogram_get_range(hist, i, &l, &u);
        x = gsl_histogram_get(hist, i);
        printf("%g\t%g\n",l,x);
41    printf("%g\t%g\n",u,x);
    }
    if(n>1)
        printf("%g\t0\n", u);
}

46
void printBoxPlot(gsl_histogram *hist){
    size_t i, n;
    double x, l, u;

51    n = gsl_histogram_bins(hist);
    for(i=0; i<n; i++){
        gsl_histogram_get_range(hist, i, &l, &u);
        x = gsl_histogram_get(hist, i);
        printf("%g\t0\n", l);
56    printf("%g\t%g\n",l,x);
        printf("%g\t%g\n",u,x);
    }
    printf("%g\t0\n",u);
}

61
void scanFile(FILE *fp, Args *args){
    double min, max, x;
    double *array;
    gsl_histogram *hist;
    size_t arraySize, n;
66    int i;

    /* get input data */
    min = DBL_MAX;
71    max = -DBL_MAX;

```

```

arraySize = 1;
array = (double *)emalloc(arraySize*sizeof(double));
n = 0;
while(fscanf(fp, "%lg", &x) == 1){
76     if(x > max)
        max = x;
        if(x < min)
            min = x;
            array[n++] = x;
81     if(n == arraySize){
        arraySize *= 2;
        array = (double *)erealloc(array, arraySize*sizeof(double));
    }
}
86 array = (double *)erealloc(array, n*sizeof(double));
/* construct histogram */
hist = gsl_histogram_alloc(args->b);
if(!args->R){
    max += DBL_EPSILON;
91    min -= DBL_EPSILON;
    gsl_histogram_set_ranges_uniform(hist, min, max);
} else
    gsl_histogram_set_ranges_uniform(hist, args->R[0], args->R[1]);
for(i=0; i<n; i++){
96    gsl_histogram_increment(hist, array[i]);
    if(!args->r)
        gsl_histogram_scale(hist, 1./n);
    if(args->B)
        printBoxPlot(hist);
101    else if(args->L)
        printLinePlot(hist);
    else
        printStepPlot(hist);
    gsl_histogram_free(hist);
106    free(array);
}

int main(int argc, char *argv[]){
    int i;
111    char *version;
    Args *args;
    FILE *fp;

    version = "0.13";
116    setprogname2("histogram");
    args = getArgs(argc, argv);
    if(args->h || args->e)
        printUsage(version);
    if(args->v)
121    printSplash(version);
    if(args->numInputFiles == 0){
        fp = stdin;
        scanFile(fp, args);
    } else{

```

```

126     for (i=0; i<args->numInputFiles; i++) {
        fp = fopen(args->inputFiles[i], "r");
        scanFile(fp, args);
        fclose(fp);
    }
131 }
    freeArgs();
    free(progname());
    return 0;
}

```

5 Change Log

- Version 0.7 (June 8, 2012)
 - Changed interface.
 - Wrote documentation.
 - Adjusted printing along x-range.
 - Included option for box-plot, which previously was the default (`-B`).
- Version 0.8 (August 28, 2012)
 - Fixed handling of bin borders by moving the line


```
ll += step;
```

 in front of the `while` block.
- Version 0.9 (June 21, 2016)
 - Switched code to GSL library functions. This should make the program more accurate.
- Version 0.10 (July 1, 2016)
 - Fixed handling of the `-R` option.
- Version 0.11 (April 8, 2017)
 - Fixed range finding.
- Version 0.12 (December 8, 2017)
 - Added Option for generating line graph (`-L`).
- Version 0.13 (November 6, 2018)
 - Fixed bug in `interface.c`.