Statistics library

Alan Kantserov

November 5, 2015

Figure 1: Function in Python

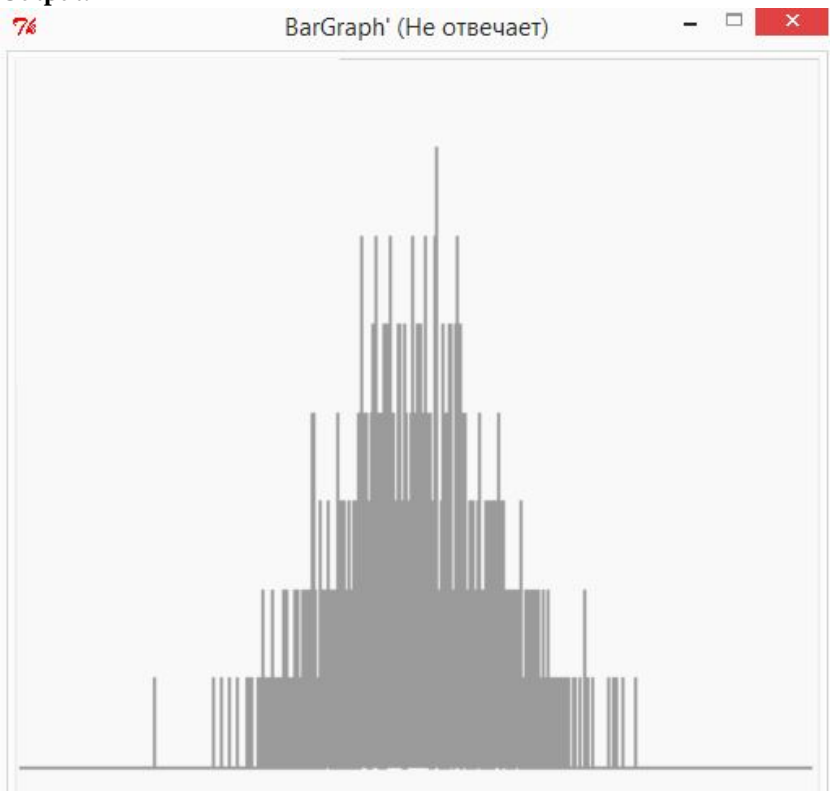
In this report I am going to show the statistics library that we created in class. The program consists of functions and all of them are related to each other. By the end the output is nice and neat diagram with Gauss distribution which is nowadays widely used in different test results.

PYTHON CODE

```
import random #import library that generates random numbers
from graphics import* #import all the graphics from graphics.py program in order to draw.

def zeros(n): #function with parameter n
    a=[] #empty array
    for i in range(n): #creates for in loop which goes until function's parameter value is reached
        a=a+[0] #creates an array till for in loop runs
    return a #returns an array that is fullfiled with empty arrays

def sum_array(a): #function that will calculate sum of array
        s=0 #initial sum value is equal to 0
        for i in range(len(a)): #creates for in loop which loops the same amount of time as length of array - a
            s=s+a[i] #each time it loops, program adds a number to the sum until the length of array is reached.
        return s #returns the sum value

def rand_array(n,mini,maxi): #function that fulfills array with random values
        a=[] #empty array
        for i in range(n): #for in loop that runs until reaches n
            a=a+[random.uniform(mini,maxi)] #append random number between minimum and maximum values
        return a #returns array

def avg(a): #function that calculates and average of an array
        return sum_array(a)/len(a) #program takes sum of an array a and divides it by the amount of numbers stored in that array.

def var(a): #function that calculates variance
        s=0 #initial value is 0
        for i in range(len(a)): #loop until reaches the last value of array a
            s=s+s[a[i]**2] #
        m=avg(a) #m is equal to an average number of an array a
        return (s/len(a)-m**2) #return sum divided by the squared average value of an array a

def maximum(a): #function look for the maximum value in the array
        m=0 #initially, maximum value is 0
        for i in range(len(a)): #we want to run this program till we reach the last number of an array a
            if a[i]>m: #checks each value in the array whether it is bigger than number m
                m=a[i] #if it is bigger, then it changes current m value on a new, bigger value from an array a
        return m #by the end, it returns maximum value

def histogram(mini,maxi,bins,a): #histogram function includes min, max values, number of bins, and array a
        h=zeros(bins) #run function zeros with parameter bins
        w=(maxi-mini)/bins #create new variable which is equal to max value minus min value dived by number of bins
        for i in range(len(a)): #loop in range of length of array a
            for j in range(bins): #loop in range of number of bins
                if a[i]>(mini+j*w) and a[i]<(mini+(j+1)*w): #if statement that checks whether we should add more
                    h[j]=h[j]+1 #if it is true add 1 to it
        return h #return h
```

```
def bargraph(a, window): #function that draws
        win=GraphWin("BarGraph", 500, 500) #Graph with name "BarGraph", width 500, height 500
        window.setCoords(-1,-1,len(a)+1, maximum(a)+1) #set coordinates of window bigger than current values of a.
        for i in range(len(a)): #for in loops runs until the last number of array a is reached.
            rec=Rectangle(Point(i,0), Point(i+1,a[i])) #Draw rectangle that is equal to our array's values
            rec.draw(window) #draw window

def main(): main function that includes all other functions
        gauss=zeros(1000) #amount of numbers we want to check
        for i in range(len(gauss)): #run loop with the amount we
            gauss[i]=sum_array(rand_array(10, 0, 1)) #gauss number is equal to sum of 10 random arrays
        histo=histogram(0., 10., 1700, gauss) #show 1700 values between 0-10
        win=GraphWin("BarGraph'",500,500) #creates a windows
        bargraph(histo,win) #draws histogram inside the window

main() #calls main function in runs everything we wrote down.
```

Output:



The window with this output appeared. As you clearly see, it is a Gauss distribution