CS 504 – Programming Languages for Data Analysis

Assignment 2: Normal PDF

Lin, Junjia 002268506

Nie, Luyun 002268087

I. Problem: Normal PDF

1. Python

CS 504 Progamming Languages for Data Analysis

Assignment 2

Lin, Junjia 002268506

Nie,Luyun 002268087

Method 1: Python

In [1]:
```
from scipy.misc import derivative
from scipy.stats import norm
import matplotlib.pyplot as plt
import math
import numpy as np
```

In [2]:
```
# Define the function of finding the error method of CDF
def finderrorcdf(nmd):
    cdf2 = []
    for i in nmd:
        cdf2.append(0.5*(1+math.erf(i/math.sqrt(2))))
    return cdf2
```

In [3]:
```
# Define the function of comparing 2 CDF Methods
def comparecdf(nmd):
    difference = []
    cdf1,cdf2 = norm.cdf(nmd).tolist(),finderrorcdf(nmd)
    for i in range(len(cdf1)):
        difference.append(cdf2[i]-cdf1[i])
    maxdifference = round(max(difference),5)
    return maxdifference
```

In [4]:
```
# Define a normal Guass distribution with μ = 0,σ**2 = 1 and has 1000 variabl
nmd = np.linspace(norm.ppf(0.001),norm.ppf(0.999), 1000)
```

Question 1: Finding the differences between 2 CDF methods

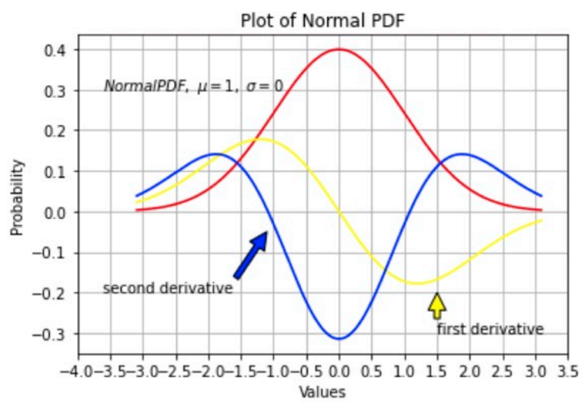
In [5]:
```
comparecdf(nmd)
```

Out[5]:
```
0.0
```

Answer: By retrieving the function of 'comparecdf', we easily find that there is no difference between two methods.

Question 2: Plot the nornal PDF and its first derivative and the second derivative

In [6]:
```
plt.plot(nmd,norm.pdf(nmd),color = 'red')
plt.plot(nmd,derivative(norm.pdf,nmd,n = 1),color = 'yellow')
plt.plot(nmd,derivative(norm.pdf,nmd,n = 2),color = 'blue')
plt.xlabel('Values')
plt.ylabel('Probability')
plt.title('Plot of Normal PDF')
plt.text(-3.6, 0.3, r'$NormalPDF,\ \mu=1,\ \sigma=0$')
plt.annotate('first derivative', xy=(1.5, -0.2), xytext=(1.5, -0.3),
            arrowprops=dict(facecolor='yellow', shrink=0.001))
plt.annotate('second derivative', xy=(-1.1, -0.05), xytext=(-3.6, -0.2),
            arrowprops=dict(facecolor='blue', shrink=0.001))
plt.xticks(np.arange(-4,4.5, step=0.5))
plt.grid(True)
plt.show()
```



Answer: as you can see above, when x = 0, the first derivative of the normal PDF is also 0. Similarily, when x = -1 or +1, the second derivative is 0 as well.

2. Julia

CS 504 Programming Language for Data Analysis

Assignment 2

Lin, Junjia 002268506

Nie, Luyun 002268087

Method 2:Julia

In [1]:
```
using Distributions, Calculus, SpecialFunctions, Plots;pyplot()
```

Out[1]:
```
Plots.PyPlotBackend()
```

In [2]:
```
# Define the normal Guass distribution range
xGrid = -5:0.01:5
```

Out[2]:
```
-5.0:0.01:5.0
```

Question 1: Finding the differences between 2 CDF methods

In [3]:
```
PhiA(x) = 0.5*(1+erf(x/sqrt(2)))
PhiB(x) = cdf(Normal(),x)
println("Maximum difference between two CDF implementations: ", maximum(PhiA.
```

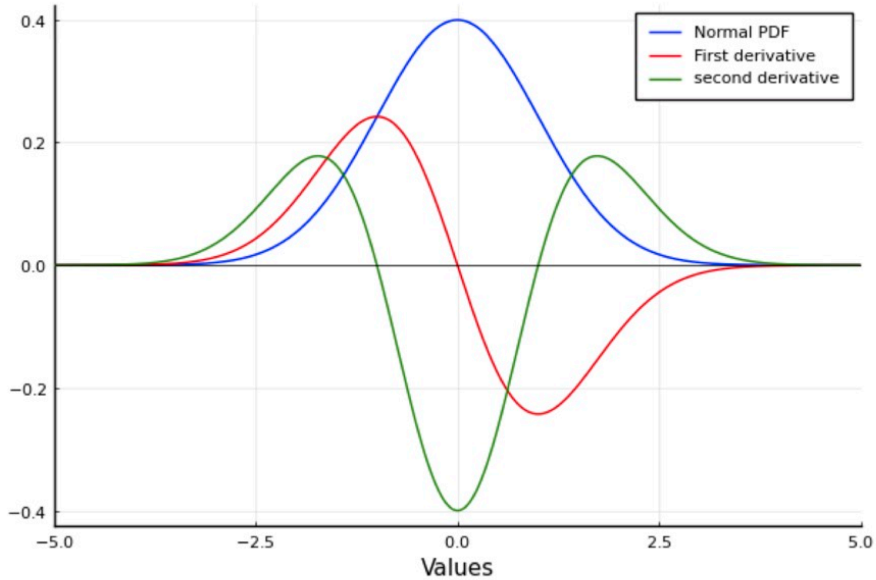
```
Maximum difference between two CDF implementations: 1.1102230246251565e-16
```

Answer: We are able to find the difference between the 2 methods is very few,nearly 0. Generally, these 2 methods can be treated equal.

Question 2: Plot the normal PDF distribution with the first derivative and second derivative

In [4]:
```
normalDensity(z) = pdf(Normal(),z)
d0 = normalDensity.(xGrid)
d1 = derivative.(normalDensity,xGrid)
d2 = second_derivative.(normalDensity, xGrid)
plot(xGrid, [d0 d1 d2], c=[:blue :red :green],label=["Normal PDF" "First deri
plot!([-5,5],[0,0], color=:black, lw=0.5, xlabel="Values", xlims=(-5,5), labe
```

Out[4]:



Answer: As we can see above, when x = 0, the first derivative of the normal PDF is 0. On the other hand, the second derivative os the normal PDF is 0 by x = -1 or x = +1

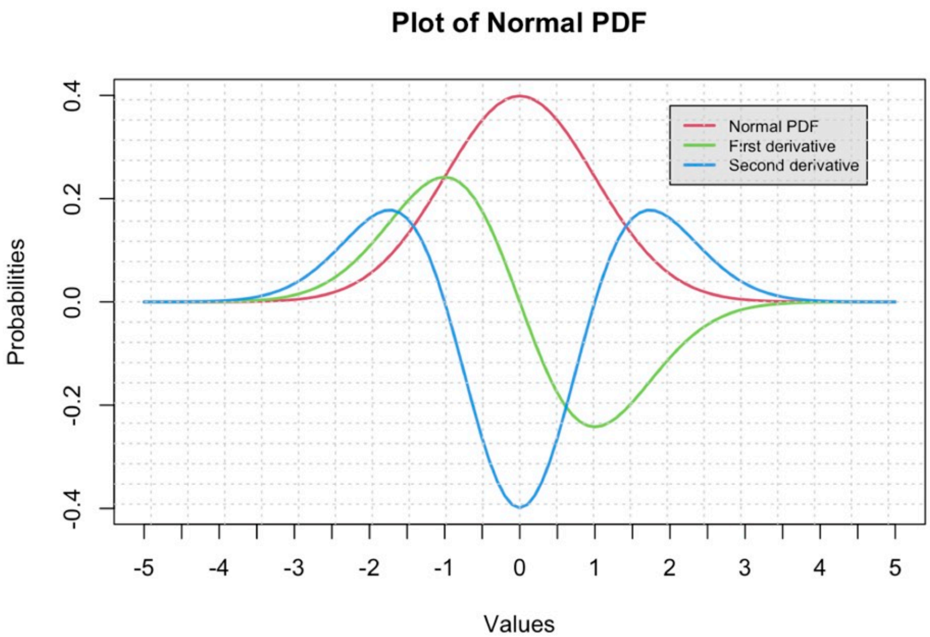
3. R

CS504-Assignment-2-R.R

tinan

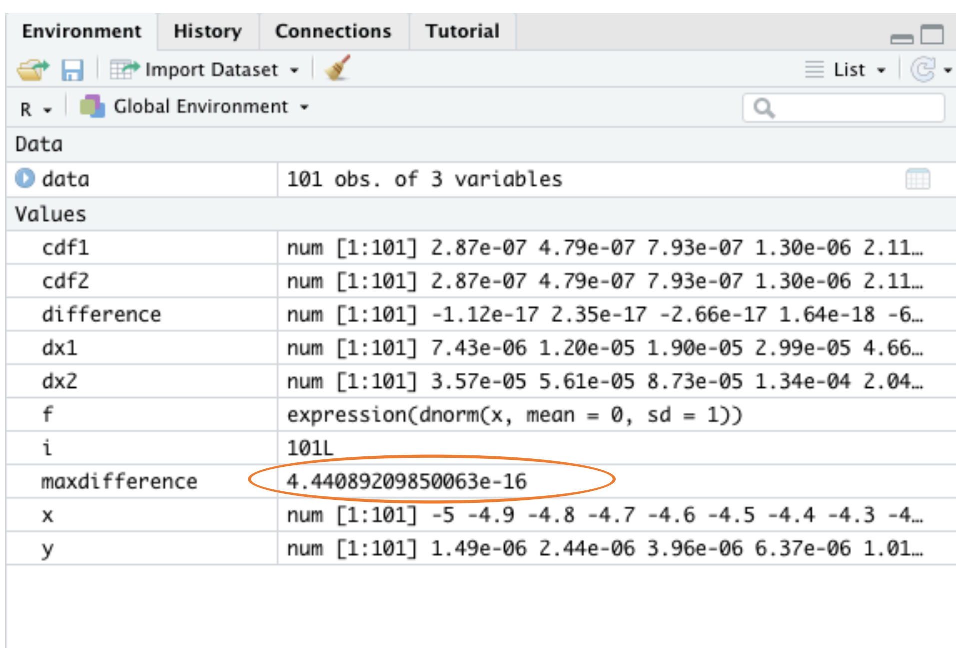
2021-03-03

```
#CS 504 Programming Language for Data Analysis
#Assignment 2
#Lin, Junjia 002268506
#Nie, Luyun 002268087
# Method 3: R

# Define the normal Guass distribution range
x<- seq(-5,5,0.1)
y<- dnorm(x, mean = 0, sd = 1)

#Question 1: Finding the differences between 2 CDF methods
difference <- c()

#To appy the drf function, we introduce the 'pracma' library
library(pracma)
cdf1 <- 0.5*(1+erf(x/sqrt(2)))
cdf2 <- pnorm(x)
for (i in seq(1,length(x))) {
  difference[i] <- cdf1[i]-cdf2[i]
}
maxdifference = max(difference)
#Answer: We are able to find the difference between the 2 methods is nearly 0.
#Generally, these 2 methods can be treated equal.

#Question 2: Plot the normal PDF with the first derivative and second derivative
f = expression(dnorm(x, mean = 0, sd = 1))
dx1 <- eval(D(f,'x'))
dx2 <- eval(D(D(f,'x'),'x'))
data <- data.frame(y,dx1,dx2)
matplot(x, data, type = "l", lty = 1, lwd = 2,col = 2:5,axes = TRUE,
        xlab="Values",ylab = "Probabilities",xaxt="n")
legend(2, 0.38, c("Normal PDF", "First derivative", "Second derivative"),
       col = 2:5,lwd = 2, merge = FALSE, bg='gray90', cex= 0.7,)
title("Plot of Normal PDF")
axis(side=1,at=seq(-5,5,0.5),labels=seq(-5,5,0.5))
grid(nx = 22, ny = 22)
```




```
#Answer: when x = 0, the first derivative of the normal PDF is 0 and
#the second derivative os the normal PDF is 0 by x = -1 or x = +1
```

The running result of the difference is shown below:



4. Octave

CS504 Programming Language in Data Analysis

Assignment 2

Lin, Junjia 002268506

Nie,Luyun 002268087

Method 4 Octave

In [1]:
```
pkg load statistics
pkg load symbolic
```

In [2]:
```
%Define the sequence
x = [-5:0.01:5];
```

Question 1: Find the difference between two cdf implementations

In [3]:
```
x1=0.5*(1+erf(x/sqrt(2)));
x2=normcdf(x,0,1);
x3=max(x1-x2);
disp(x3);
%maximum difference between two cdf implementations
```

```
1.1102e-16
```

Answer: the difference between two cdf implementations are extremly small that we can consider these two methods are the same.

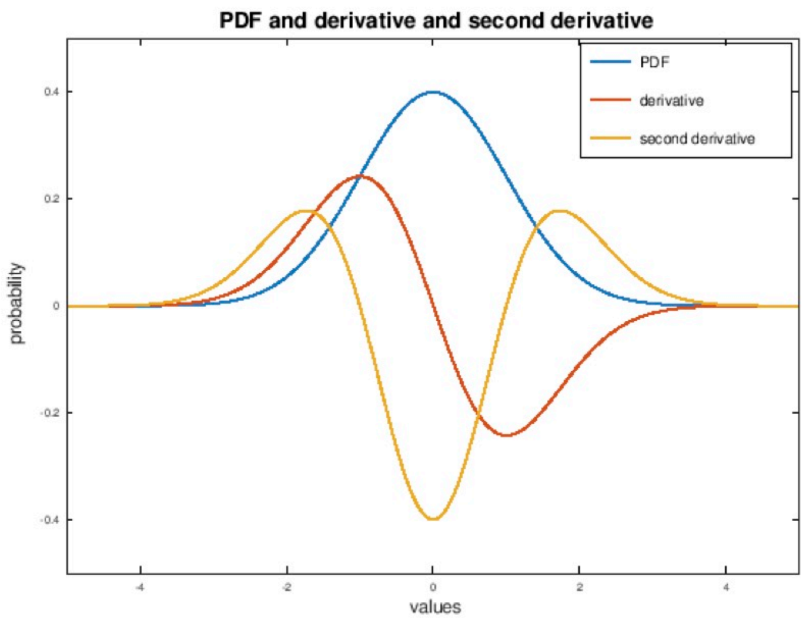
Question 2: Plot the first and the second derivative

In [4]:
```
%define PDF, derivative and second derivative
f=normpdf(x,0,1);
df= diff (f)/0.01;
dff= diff(df)/0.01;
```

In [5]:
```
% check the length of f df and dff
n=2*length(f);
a1=n
n=2*length(df);
a2=n
n=2*length(dff);
a3=n
```

```
a1 = 2002
a2 = 2000
a3 = 1998
```

In [7]:
```
graphics_toolkit ("gnuplot");
x1=[-4.99:0.01:5];
x2=[-4.99:0.01:4.99];
plot(x,f,'LineWidth',4,x1,df,'LineWidth',4,x2,dff,'LineWidth',4);
h = legend({'PDF','derivative','second derivative'});
set (h, "fontsize", 14);
title('{\fontsize{30} PDF and derivative and second derivative}')
axis([-5 5 -0.5 0.5])
xlabel('{\fontsize{25} values}');
ylabel('{\fontsize{25} probability}');
hold off;
%plot the curves
```



Answer: Answer: as you can see above, when x = 0, the first derivative of the normal PDF is also 0. Similarily, when x = -1 or +1, the second derivative is 0 as well.

II. Problem: Answer Problems

Which programming language provided a relevant solution for this assignment? Which difficulties will you face if you want to solve this problem using Scala programming language?

Answer: After writing all these four language codes, we found that they all contain the normal pdf functions as well as the CDF functions. Comparing all these programming languages, Julia is the most concise and fastest language that we only need few lines to proceed with the PDF. Python is the most structured that we can specifically customize any functions and retrieve them directly. In terms of this feature, Python is easy to read and understand. By executing codes line by line, R gives running results correspondently and it is convenient to calculate equations or expressions. However, in this case, we have to import the "pracma" library to solve the erf expression. At last, as we wrote in the former assignment, Octave is more suitable for calculating matrix. Regarding its matrix properties, we should be careful about implementation different variables before plotting by comparing each length of them. Besides, there are many warnings in the process of installing related packages costing a longer time than other languages. Undoubtedly, the Octave still wons the most dislike data analyzing language prize.