Exercise 7_solutions

November 14, 2023

[2]:
```python
import numpy as np
import matplotlib.pyplot as plt
import math
import sys
from scipy.stats import norm
from scipy.stats import chi2
# This is some custom code that I wrote
# You can pick it up from the website. (A link is given)
# http://hep.ucsb.edu/people/claudio/ph29-f23/ccHistStuff.py
sys.path.insert(0,'/Users/claudio/Dropbox/Claudio/ph29-f23/')  # macOS
# sys.path.insert(0,'C:\Users\Claud\Dropbox\Claudio\ph29-f23\')  # Windows
from ccHistStuff import statBox
```

0.1 Exercise 1

[3]:
```python
np.random.seed(12345)  # Set the seed
n = 100000  # number to pick

x = np.random.normal(loc=1, scale=0.02, size=n)
y = np.random.normal(loc=1, scale=0.3, size=n)
z = np.random.normal(loc=1, scale=0.3, size=n)
a = x/y
b = y/x
c = x*y
d = y*z
e = y/z
results = [a, b, c, d, e]
```

[4]:
```python
fig = plt.figure(figsize=(10,3))

ipl = 0
bins = np.linspace(0,3,500)
for blah,nn in zip([x,y,z],[x,y,z]):
    ipl = ipl+1
    ax = plt.subplot(130+ipl)
    ax.hist(blah,bins,histtype='step',label=nn)
```
ax.grid()
ax.legend(fontsize='large', loc='lower right')
statBox(ax, blah, bins, greek=False)
_ = ax.set_xlim(0, 3)

[5]:
    # Plot a, b, c, d, e
    # We will use the statBox function to get
    # means and standard deviations.
    fig = plt.figure(figsize=(10, 10))

    ipl = 0
    bins = np.linspace(0, 3, 100)
    for r, n in zip(results, names):
        ipl = ipl + 1
        ax = plt.subplot(320 + ipl)
        ax.hist(r, bins, histtype='step', label=n)
        ax.grid()
        ax.legend(fontsize='xx-large', loc='center right')
        statBox(ax, r, bins, greek=False)
        _ = ax.set_xlim(0, 3)
0.2 Exercise 2

[6]:
```python
np.random.seed(12345) # Set the seed
n = 100000 # number to pick
means = [2.7, 8.2, 29.5] # The means to use
picks = [] # a list to save the picks
for m in means:
    temp = np.random.poisson(m, size=n)
picks.append(temp)
```

[7]:
```python
# Reasonable histogram binning
# (Put the center of the bins at integer value)
b1 = np.linspace(-0.5,10.5,12) # for mean = 2.7
b2 = np.linspace(-0.5,22.5,24) # for mean = 8.2
```
b3 = np.linspace(10.5, 50.5, 41)  # for mean = 29.5
bins = [b1, b2, b3]

# Plot them
fig = plt.figure(figsize=(15, 5))
ipl = 0
for n, b, m in zip(picks, bins, means):
    ipl = ipl + 1
    ax = plt.subplot(130 + ipl)
    ax.hist(n, b, histtype='step', label='mean = '+str(round(m, 1)))
    ax.legend(fontsize='large')
    ax.grid()
    ax.set_xlim(b[0], b[-1])

0.3 Exercise 3

0.3.1 \[ I = \int_{1}^{\infty} \frac{e^{-x^2}}{x+1} dx \]

0.3.2 \[ t = \frac{1}{x} \rightarrow dx = \frac{-1}{t^2} dt \]

0.3.3 \[ x = 1, \infty \rightarrow t = 1, 0 \]

0.3.4 \[ x + 1 = \frac{1+t}{t} \]

0.3.5 \[ I = \int_{0}^{1} e^{-1/t^2} \frac{-1}{t(1+t)} dt = \int_{0}^{1} e^{-1/t^2} \frac{dt}{t(1+t)} \]

[9]: # The function to integrate
    def f(t):
        return np.exp(-1/(t*t)) / (t*(1+t))

[10]: # plot the function to see what it looks like
    # Start the plot away from 0 otherwise we get a divide by zero
    ax = plt.subplot(111)
xpl = np.linspace(0.0001, 1, 500)
```python
ax.plot(xpl, f(xpl))
ax.set_xlabel('t', fontsize='xx-large')
ax.set_ylabel('f(t)', rotation=0, fontsize='xx-large')
ax.grid()
ax.set_xlim(0, 1)
ax.set_ylim(bottom=0)
ax.yaxis.set_label_coords(-.15, .5)

# Use midpoint with 100 bins
n = 100
bins = np.linspace(0, 1, n+1)
binsC = 0.5 * (bins[1:] + bins[:-1])  # bins centers
dt = bins[1] - bins[0]  # bins width
print("Integral = ", (dt*f(binsC)).sum())

Integral = 0.060877076783674336
```

Integral = 0.060877076783674336