

DRPpdfCdf 概率密度和分布用法

DRPpdfCdf 概率密度和分布控件提供一个直方图分析模块，可以对收到的采样信号进行概率密度和分布曲线分析。该控件为后台控件，在 GUI 界面上无显示。

在 Python 程序 Tkinter GUI 界面中创建一个 DRPpdfCdf 概率密度和分布控件的命令和语法格式如下：

```
hl=DRPpdfCdf(option=value, ...)
```

该控件创建时只有 3 个可选的输入参数：

M: 幅值等分数，默认值=100

ylow: 幅值分割计算的下限，默认值=0

yhigh: 幅值分割计算的上限，默认值=0

返回值 hl 则是该控件创建后的句柄变量名，可以通过 hl 对创建后的控件进行进一步的设定。如果 ylow==yhigh，幅值分割范围采用自动量程。

DRPpdfCdf 概率密度和分布控件有 3 个属性函数，其功能定义如下：

hl.setValue2DX(dx,yarray): 设定信号的采样间隔和采样数组

hl.setPar(M,ylow,yhigh): 设定分析参数

abin,pdf,cdf=hl.getPdfCdf(): 获取当前的概率密度和分布曲线数据

下面是一个 DRPpdfCdf 概率密度和分布控件创建实例：

```
hl=dsp.DRPpdfCdf()
```

通常，可以选择设定部分选择参数的形式进行设定，其余可选参数采用默认值。创建实例如下：

```
hl=dsp.DRPpdfCdf(ylow=-1,yhigh=1)
```

创建后可以通过返回的控件句柄对其属性函数进行操作。

DRPpdfCdf 概率密度和分布控件可以通过绑定的方式将信号的概率密度和分布曲线数据传递给绑定的回调函数或其他控件。获取概率密度曲线的绑定方法为：

```
hl.addCallBack2Dpdf(fun)
```

获取功率谱的计算数据绑定方法为：

```
hl.addCallBack2Dcdf(fun)
```

被绑定的回调函数或另一控件的 set 函数是一个二维数组函数，收到的是信号曲线的 x 和 y 数组值。回调函数形式为：

```
def setArrayTwo(xArray,yArray):
```

```
...
```

当采用 DRPython 进行图形化编程时，控件的创建代码和直连绑定代码的生成均可以由机器实现；控件创建只需要将所需控件拖拽到 GUI 布局区，而直连绑定则需要在 GUI 布局区的控件间进行连线操作。DRPpdfCdf 概率密度和分布控件可以做为字符串或单数值输出控件，在连线状态下在控件上点击鼠标，则弹出端选择；选择完成后，点击另一控件，则弹出输入端选择；若输出端和输入端类型一致，则完成连线。输出参数为 2 个一维数组构成的概率密度曲线数据(addCallBack2Dpdf)，和 2 个一维数组构成的概率分布曲线数据(addCallBack2Dcdf)。



DRPpdfCdf 概率密度和分布控件也可以做为单值/字符串输入控件，在连线状态下当上一控件已选输出函数后，在控件上点击鼠标，则弹出连线选择菜单；选择完成后，若输出端和输入端类型一致，则完成连线。输入参数为单值+一维数组的采样信号(setValue2DX)。



下面是 DRPpdfCdf 概率密度和分布控件的创建和应用示例小程序，功能是在 GUI 界面上插入 1 个按钮控件，1 个信号发生器控件、1 个 DRPpdfCdf 概率密度和分布计算控件，一个 3 幅曲线的 SubPlot 控件，然后分别通过直连绑定的方式，用一个按钮控件控制信号发生器的启/停和信号波形曲线与概率密度和分布曲线的显示。DRPpdfCdf 概率密度和分布控件应用示例小程序代码如下：

```

#==引用的包
import tkinter as tk
import numpy as np
import drvi.drviControlls as dr
import drvi.drviDSP as dsp

#用户自定义的回调函数
def muserSingle(v):
    global F
    Fs=5000
    N=512
    F=F+10
    t=np.arange(N)*(1/Fs)
    y=0.5*np.sin(2*np.pi*F*t)
    abc.setValue2DX(1/Fs,y)
    abin,pdf,cdf=abc.getPdfCdf()
    mSub.setValue2D1(t,y)
    mSub.setValue2D2(abin,pdf)
    mSub.setValue2D3(abin,cdf)

#==创建 Tkinter GUI 窗口
win= tk.Tk()
win.geometry('1100x600')
win.config(bg="#ddeeee")
#==插入按钮控件
mButton1=dr.DRButton(win,20,10,100,30,txt='Run/Stop')
#==插入概率密度和分布控件

```

```
abc=dsp.DRPdfCdf(ylow=-1, yhigh=1)
#==插入曲线显示控件
mSub=dr.DRSubPlot(win, 20, 50, 950, 500, N=3, M=1)
#==信号的直连绑定
mButton1.addCallbackSingle(muserSingle)
#==全局变量==
F=0
#==主程序循环
win.mainloop()
```

pyqt 版本中的用法说明:

DRVI 有继承自 Tkinter 和继承自 pyqt 两个不同库,除引用的 GUI 库(Tkinter/pyqt)和框架程序略有不同外,DRVI 部分控件的创建、布局和连线是完全相同的。因均继承自 Numpy 和 Scipy,该函数在两个版本中用法无区别。