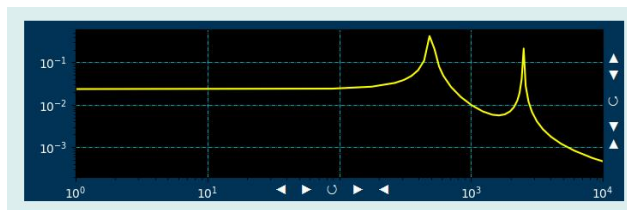


DRLoglog 二维曲线控件用法

DRLoglog 二维曲线控件采用双对数坐标形式，也就是 $\log_{10}(x)$, $\log_{10}(y)$ ，主要用于以双对数形式显示频谱等二维曲线。当有曲线动态刷新速度要求时，建议用 DRPlot 控件，将 x 和 y 坐标类型均设为对数， $xtype=1$, $ytype=1$ ，二者等价，但 DRPlot 控件做了动态显示优化，显示速度快一些。



DRLoglog 二维曲线控件图形显示效果图

在 Python 程序 Tkinter GUI 界面中创建一个 DRLoglog 二维曲线控件的命令和语法格式如下：

```
hl=DRLoglog(win,x,y,w,h,option=value, ...)
```

其中输入参数 win 为 Tkinter 库创建的 GUI 窗口句柄，[x,y,w,h]四个参数为该控件插入 GUI 窗口 win 时的左上角坐标[x,y]和该控件的宽、高[w,h]。另外是一组可选的输入参数，包括：

- title: 曲线标题，默认值=""
- cb1: 背景色，默认值='#003355'
- cb2: 背景色 1，默认值='#000000'
- cf1: 前景色，默认值='#ffffff'
- cf2: 前景色，默认值='#00eeff'
- cf3: 前景色，默认值='#ffff00'
- xlow: 横坐标最小值，默认值=0
- xhigh: 横坐标最大值，默认值=0
- ylow: 纵坐标最小值，默认值=0
- yhigh: 纵坐标最大值，默认值=0
- xarray: 曲线横坐标数据数组，默认值=0
- yarray: 曲线纵坐标数据数组，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。

控件创建时，当 xlow 和 xhigh 均设为零时，横坐标轴采用自动量程；当 ylow 和 yhigh 均设为零时，纵坐标采用自动量程；当 xarray 和 yarray 设为零，则初始曲线为空。另外，在自动量程时，DRLoglog 二维曲线控件中的曲线放大、缩小、滚动等操作符无效。

DRLoglog 二维曲线控件有 6 个属性函数，其功能定义如下：

- hl.setXlim(xmin,xmax): 设定 X 轴显示范围
- hl.setYlim(ymin,ymax): 设定 Y 轴显示范围
- hl.setValue2D(xarray,yarray): 设定二维曲线数据，并以覆盖形式重新绘制曲线
- hl.setValue2DX(dx,yarray): 设定等离散采样间隔二维曲线数据，并以覆盖形式重新绘制曲线
- hl.addValue2D(xarray,yarray): 设定二维曲线数据，并以添加形式绘制新曲线。
- hl.addValue2DX(dx,yarray): 设定等离散采样间隔二维曲线数据，并以添加形式绘制新曲线

下面是一个全参数的 DRLoglog 二维曲线控件创建实例：

```
hl=dr.DRLoglog(win,61,103,700,210,"'#003355'','#000000'','#ffff'','#00eeff'','#ffff00',1,10000,0,0,0,0)
```

通常，可以选择设定部分选择参数的形式进行设定，除基本参数 win、[x,y,w,h]和用变量名设定的可选参数外，其余可选参数采用默认值。创建实例如下：

```
hl=dr.DRLoglog(win,61,103,700,210)
```

创建后可以通过返回的控件句柄对其属性函数进行操作。

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。DRLogLog 二维曲线控件为输入控件，可接收 2 种输入形式，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。输入端口(setValue2D)和端口(setValue2DX)收到的是控件的曲线显示数据。



下面是 DRLoglog 二维曲线控件的创建和应用示例小程序，功能是在 GUI 界面上插入一个按钮控件，一个自定义函数和一个 DRLoglog 二维曲线控件，然后通过直连绑定方式，用按钮控件来驱动自定义函数生成信号和信号频谱，然后用 DRLoglog 二维曲线控件来显示信号频谱。DRLoglog 二维曲线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
from scipy.fft import rfft,irfft
import drvi.drviControlls as dr

#用户自定义的回调函数
def muserSingle(v):
    global F
    Fs=5000
    N=512
    F=F+100
    t=np.arange(N)*(1/Fs)
    y=0.5*np.sin(2*np.pi*F*t)
    N2=int(N/2)
    A=np.abs(np.fft.rfft(y))/(N2)
    df=Fs/N;
    mP.setValue2DX(df,A)

#==创建 Tkinter GUI 窗口
win= tk.Tk()
win.geometry('900x500')
win.config(bg="#ddeeee")
#==插入按钮控件
mB1=dr.DRButton(win,78,39,100,40,'#00aacc','#eeee00','Button',12,101)
```

```
#==插入 LogLog 控件
mP=dr.DRLoglog(win, 95, 122, 700, 210)
#==控件直连绑定
mBl.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 Matplotlib,该函数在两个版本中用法无区别。