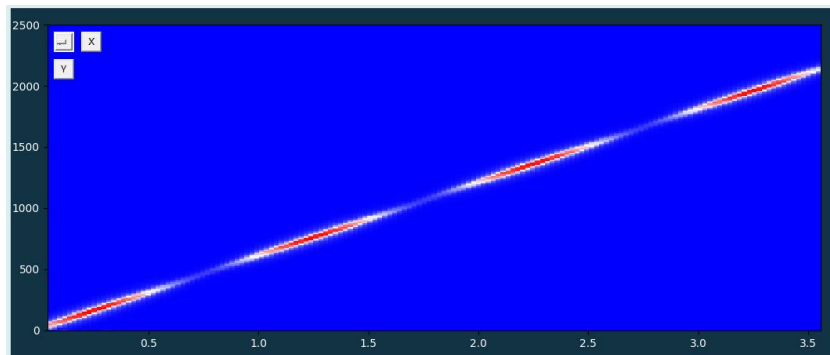


DRColormesh 语谱图控件用法

DRColormesh 语谱图显示控件主要用于显示时变信号的语谱图。一般来说，对数功率谱阵的显示效果较好。其作用与 DRPlotContour 伪彩色图控件相似，但依托的是 matplotlib 库中的 pcolormesh 伪彩色图函数，而非 plotcontour 三维曲面图函数，效率更高一些；另外直接采用 STFT 变换结果数据格式，也更方便。该控件主要用于教学目的，专业的语谱图分析和显示建议用 Librosa 库的 librosa.display.specshow+DRFigure 函数。



DRColormesh 语谱图显示控件图形显示效果图

在 Python 程序 Tkinter GUI 界面中创建一个 DRColormesh 语谱图显示控件的命令和语法格式如下：

```
hl= DRColormesh(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：

cb1: 背景色，默认值='#113344'

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'

ylog: Y 轴显示模式，默认值 0，0/1=线性、对数

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。

DRColormesh 语谱图显示控件有 4 个属性函数，其功能定义如下：

hl.setValue3D(t,f,a2D): 设定语谱图曲线数据，t 和 f 为一维数组，a2D 为二维数组，对应

f,t,a2D=signal.spectrogram(...)；但通常将 spectrogram 函数返回的功率谱阵数据转化为对数功率谱阵后再进行显示，即 $20 * \text{np.log}_{10}(\text{a2D})$ 。

hl.setXlim(x1,x2): 设定 X 轴(通常代表时间)的显示量程范围。

hl.setYlim(y1,y2): 设定 Y 轴(通常代表频率)的显示量程范围。

hl.setCmap(name): 设定伪彩色图谱名称，参略 matplotlib 的 cmap 参数介绍，常用图谱名称有：bwr、rainbow、cool、viridis、Blues、Reds、Greys。

下面是一个全参数的 DRColormesh 语谱图显示控件创建实例：

```
hl=dr.DRColormesh(win,50,92,1000,400,'#113344','#000000','#ffffff','#00eeff')
```

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.DRColormesh(win,50,92,1000,400)
```

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。DRColormesh 语谱图控件为输入控件，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。输入端口(setValue3D)收到的是控件的曲线显示数据。



下面是 DRColormesh 语谱图显示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 DRColormesh 语谱图显示控件和一个用户自定义函数，然后通过直连绑定方式，用按钮控件来驱动用户自定义回调函数产生一段 chirp 信号，然后计算 STFT 谱阵，最后用 DRColormesh 语谱图显示控件来显示。DRColormesh 语谱图显示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 import signal
import drvi.drviControls as dr

#用户自定义的回调函数
def muserSingle3(v):
    Fs=5120;
    N=50000;
    t=np.arange(N)*(1.0/Fs)
    y=np.sin(2*np.pi*(100+5*t*t)*t)
    nft=1024; ovl=int(nft/2)
    ff, tt, Sxx=signal.spectrogram(y, Fs, nperseg=nft, noverlap=ovl)
    mArray.setValue3D(tt, ff, 20*np.log10(Sxx))

##创建 Tkinter GUI 窗口
win= tk.Tk()
win.geometry('900x497')
win.config(bg="#ddeeee")
##插入按钮控件
mButton1=dr.DRButton(win, 20, 20, 100, 30, txt=' Run' )
##插入语谱图显示控件
mArray=dr.DRColormesh(win, 20, 60, 800, 400)
##绑定用户回调函数
mButton1.addCallBackSingle(muserSingle3)
##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。