Visit QuantNet

NoiseDifference

```
Name of QuantLet: NoiseDifference

Published in: Statistic Financial Market

Description: Drawing figures for White, Pink and Blue
noise in time-domain and frequency-domain, ACF, PCF
and applied Fourier transform

Keywords: White nose, pink noise, blue noise, time-
domain, frequency-domain, Fourier transform

Author: Junjie Hu

Submitted: Mon, January 08 2017 by Junjie Hu
```

```
Input: blue_noise, hbo_opening, pink_noise,
white_noise

Output: ACF_BN, ACF_PN, ACF_WN, FD_BN, FD_HBO, FD_PN,
FD_SinFunc, FD_WN, HBO_TD, TD_BN, TD_Decomp, TD_PN,
TD_WN
```

Python Code:

```
import wave
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np
import statsmodels.tsa.api as smt
from scipy.signal import periodogram
from statsmodels.tsa.stattools import adfuller
from scipy.stats.mstats import normaltest


def extract_signal(file_name, num_frames=-1,
dtype='int16'):
    sound_file = wave.open(file_name, 'rb')
    # Open signal file, decoding as int16
    signal =
np.fromstring(sound_file.readframes(num_frames),
dtype=dtype)
    print(len(signal))
    return signal


def plot_signal(signal, color='b', name=None):
    plt.figure(figsize=(20, 5))
    plt.plot(signal, c=color, linewidth=1.0)
    plt.xlabel('Time, t', fontsize=16)
    # plt.title(name)
    plt.savefig(name + '.pdf', dpi=300)
    plt.show()


def plot_acf_pacf(signal, lags=None, name=None):
    plt.figure(figsize=(20, 5))
    acf = plt.subplot(1, 2, 1)
    smt.graphics.plot_acf(signal, lags=lags, ax=acf,
marker='.')
    plt.xlabel('Time Lag', fontsize=16)
    pacf = plt.subplot(1, 2, 2)
    smt.graphics.plot_pacf(signal, lags=lags,
ax=pacf)
    plt.xlabel('Time Lag', fontsize=16)
    plt.savefig(name + '.pdf', dpi=300)
    plt.show()


def plot_periodgram(signal, color=None, name=None):
    plt.figure(figsize=(20, 5))
    plt.subplot(2, 1, 1)
    plt.plot(signal, c=color, linewidth=1.0)
    plt.xlabel('Time, t', fontsize=16)
    plt.ylabel('intensity', fontsize=16)
    plt.subplot(2, 1, 2)
    spectrum_signal = periodogram(signal)
    plt.plot(spectrum_signal[0], spectrum_signal[1],
'red')
    plt.xlabel('Freq', fontsize=16)
    plt.ylabel('spectrum', fontsize=16)
    plt.tight_layout()
    plt.savefig(name + '.pdf', dpi=300)
    plt.show()


def plot_decomposition():
    time = np.linspace(0, 100, 10000)
    plt.figure(figsize=(20, 5))
    plt.subplot(2, 1, 1)
    sin_func_1 = 0.5 * np.sin(2 * np.pi * 0.1 * time)
# f=0.1
    sin_func_2 = 0.7 * np.sin(2 * np.pi * 0.6 * time)
# f=0.6
    sin_func_3 = 1 * np.sin(2 * np.pi * 1.2 * time)
# f=1.2
    sin_func = sin_func_1 + sin_func_2 + sin_func_3
    plt.plot(time, sin_func)
    plt.xlim(-1, 101)
    plt.subplot(2, 1, 2)
    plt.plot(time, sin_func_1, linewidth=1)
    plt.plot(time, sin_func_2, c='g', linewidth=1)
    plt.plot(time, sin_func_3, c='r', linewidth=1)
    plt.xlabel('Time, t', fontsize=16)
    plt.xlim(-1, 101)
    plt.tight_layout()
    plt.savefig('TD_Decomp.pdf', dpi=300)
    plt.show()
    return sin_func


mpl.rcParams['agg.path.chunksize'] = 10000

# sampling from the audio
hbo_signal =
extract_signal(file_name='hbo_opening.wav')
white_noise_signal =
extract_signal(file_name='white_noise.wav',
num_frames=500)
blue_noise_signal =
extract_signal(file_name='blue_noise.wav',
num_frames=500)
pink_noise_signal =
extract_signal(file_name='pink_noise.wav',
num_frames=500)

# plot time series of signals
hbo_td = plot_signal(hbo_signal, color='k',
name='HBO_TD')
white_td = plot_signal(white_noise_signal, color='k',
name='TD_WN')
blue_td = plot_signal(blue_noise_signal, color='b',
name='TD_BN')
pink_td = plot_signal(pink_noise_signal, color='m',
name='TD_PN')

# plot acf and pacf of signals
white_acf_pacf =
plot_acf_pacf(signal=white_noise_signal,
name='ACF_WN')
blue_acf_pacf =
plot_acf_pacf(signal=blue_noise_signal,
name='ACF_BN')
pink_acf_pacf =
plot_acf_pacf(signal=pink_noise_signal,
name='ACF_PN')

# plot periodogram of signals
hbo_specgram = plot_periodgram(signal=hbo_signal,
name='FD_HBO', color='k')
white_specgram =
plot_periodgram(signal=extract_signal(file_name='whit
e_noise.wav', num_frames=441000), name='FD_WN',
                                 color='k')
blue_specgram =
plot_periodgram(signal=extract_signal(file_name='blue
_noise.wav', num_frames=441000), name='FD_BN',
                                color='b')
pink_specgram =
plot_periodgram(signal=extract_signal(file_name='pink
_noise.wav', num_frames=441000), name='FD_PN',
                                color='m')
plot_periodgram(plot_decomposition(),
name='FD_SinFunc')

# adf test
hbo_adf = adfuller(hbo_signal)
```

```
white_noise_adf = adfuller(white_noise_signal)
blue_noise_adf = adfuller(blue_noise_signal)
pink_noise_adf = adfuller(pink_noise_signal)
# normality test
hbo_norm = normaltest(hbo_signal)
white_noise_norm = normaltest(white_noise_signal)
blue_noise_norm = normaltest(blue_noise_signal)
pink_noise_norm = normaltest(pink_noise_signal)
# histogram plot
plt.hist(white_noise_signal, bins=50)
plt.hist(blue_noise_signal, bins=50)
plt.hist(pink_noise_signal, bins=50)
```