```python
#PSP_TTplot.py
from __future__ import print_function, division

import PSP_travelTime as tt
import numpy as np
import matplotlib.pyplot as plt
```

PSP_travelTime を tt として
呼び出せるようにする

```python
def cleanDisplay():
    plt.close()
    plt.figure(figsize=(10,8))

def showDisplay():
    plt.title("")
    plt.xlabel("Time [ns]",fontsize=20,labelpad=8)
    plt.ylabel("Reflection coefficient [-]",fontsize=20,labelpad=8)
    plt.tick_params(axis='both', which='major', labelsize=20,pad=8)
    plt.tick_params(axis='both', which='minor', labelsize=20,pad=8)
    plt.ylim(ymin=-1, ymax=1)
#    plt.ylim(-0.85,-0.05)
#    plt.xlim(2.1,13.0)

    plt.show()

def drawWaveForm():
    lastIndex = len(tt.reflecCoeff)-2
    t = np.zeros(lastIndex, float)
    for i in range(lastIndex):
        t[i] = tt.timeVector[i] * 1E09
    y = tt.reflecCoeff[0:lastIndex]
    dy = tt.dy[0:lastIndex]
    plt.plot(t, y, 'k.')
    plt.plot(t, dy, 'k--')
```

実験データの波形をグラフ
にするための関数群

```python
def drawRegressionLines():
    nrPoints = len(tt.timeVector)
    step = int(16. * (nrPoints / 256.0))

    t = np.zeros(nrPoints, float)
    curve1 = np.zeros(nrPoints, float)
    curve2 = np.zeros(nrPoints, float)
    curve3 = np.zeros(nrPoints, float)
    curve4 = np.zeros(nrPoints, float)
    for i in range(nrPoints):
        t[i] = tt.timeVector[i] * 1E09
        curve1[i] = tt.flatLine.b
        curve2[i] = tt.line1.a * tt.timeVector[i] + tt.line1.b
        curve3[i] = tt.line2.a * tt.timeVector[i] + tt.line2.b
        curve4[i] = tt.line3.a * tt.timeVector[i] + tt.line3.b

    index = int(round(tt.p0.x / tt.deltaTime))
    first = max(0, index - step)
    last = min(nrPoints, index + step)
#    print(first, last, nrPoints, index, step)

    plt.plot(t[first:last], curve1[first:last], 'k')
    plt.plot(t[first:last], curve2[first:last], 'k')

    index = int(round(tt.p2.x / tt.deltaTime))
    first = max(0, index - step)
    last = min(nrPoints, index + step)
```

```python
    plt.plot(t[first:last], curve3[first:last], 'k')
    plt.plot(t[first:last], curve4[first:last], 'k')

    plt.plot(tt.p0.x* 1E09, tt.p0.y, 'ks')
    plt.plot(tt.p1.x* 1E09, tt.p1.y, 'ks')
    plt.plot(tt.p2.x* 1E09, tt.p2.y, 'ks')
```