PyQt4 PySide differences

This document is currently **not exaustive**, more differences may exist...

imports

PyQt4 / PySide main module
Submodules are mainly similar
Exception: uic (see below UI handling)

QVariant, QString etc.

PySide makes direct conversions between such Qt classes and equivalent Python classes, as PyQt does with sip API version 2.
Thus PyQt needs to be switched to use this API version first:

```
API_VERSION = 2
qt_api = ["QDate", "QDateTime", "QString", "QTextStream", "QTime", "QUrl", "QVariant"]
for qt_module in qt_api:
    sip.setapi(qt_module, API_VERSION)
```

This has to be done **before any PyQt module is imported**.

Signal/slots:

in PyQt4: **QtCore.pyqtSlot** and **QtCore.pyqtSignal**
in PySide: **QtCore.Slot**, **QtCore.Signal**

UI handling

PyQt

PyQt has a PyQt4.uic module which re-parses .ui files and produces a UI class with can instantiate several times the same widget, and provides extra variables allowing an easy access to named internal widgets.

```
from PyQt4 import uic
ui = uic.loadUiType("uifile.ui")
```

ui is a tuple containing 2 classes types: a UI class and a widget class. The UI class can be instantiated one or several times, and offers the setupUi() method.

```
widget = ui[1]() # builds an empty widget of the right main type
ui_instance = ui[0]()
ui_instance.setupUi(widget) # fills the widget with the ui contents
ui_instance.sub_widget # points to the sub-widget named "sub_widget" in the UI file
```

PySide

PySide only offers bindings to the C++ functions.
Loading a .ui

```
from PySide import QtUiTools
ui = QtUiTools.QUiLoader().load("uifile.ui")
```

ui is a QWidget instance.
To allow re-instantiating widgets through a class with constructor, .ui files must be "compiled" to generate a python file, using the pyside-uic tool (equivalent of pyuic4 in PyQt). This solution obviously needs a "pre-compiling" step.

Matplotlib

To integrate Matplotlib and Qt in widgets, Matplotlib has to be switched to use the appropriate backend:

```
import matplotlib
matplotlib.use('Qt4Agg')
if QT_BACKEND == PYSIDE:
    if 'backend.qt4' in matplotlib.rcParams.keys():
        matplotlib.rcParams['backend.qt4'] = 'PySide'
    else:
        raise RuntimeError("Could not use Matplotlib, the backend using PySide is missing.")
else:
    if 'backend.qt4' in matplotlib.rcParams.keys():
        matplotlib.rcParams['backend.qt4'] = 'PyQt4'
    else:
        raise RuntimeError("Could not use Matplotlib, the backend using PyQt4 is missing.")
from matplotlib.backends.backend_qt4agg import FigureCanvasQTAgg as FigureCanvas
from matplotlib.figure import Figure
```

QFileDialog

QFileDialog.getOpenFileName() and **QFileDialog.getSaveFileName()** do not return the same:

- a file name (unicode) in PyQt
- a tuple with 2 elements in PySide: the file name (unicode) is the 1st element