Input parameters to use the software Borehole-PyopenCL (version 0.11)

Ursula Iturrarán-Viveros and Miguel Molero, 2013
ursula @ciencias.unam.mx, miguel.molero@csic.es

The following parameters are in the `main.py` file and these were used to simulate the fast formation in Fig. 3. This first instruction corresponds to the size of the model: size along the *r*-axis (`rmax`) and size along the *z*-axis (`zmin` and `zmax`) the units are meters.

```
#define scenario dimension limits
scenario  = [3.0, 0, 8]  # rmax, zmin, zmax
```

In principle one can choose to model a monopolar or a dipolar logging tool. However we only tested monopolar so we always consider n=0. The next parameter corresponds to the maximum frequency `fmax` (kHz). The following two parameters are the source location (*r*,*z*) in meters.

```
#define source parameters and source position: n=0 =>
Monopolar, n=1 =>Dipolar
source   = [0, 8000.0, 0.005, 4]    #n, fmax, r, z
```

The radii for the fluid and in the case that one wants to consider invaded zones or a casket we should set the sizes (in meters) of the various rings within the `rx` array. The last value in `rx=[0.1, 3.0]` needs to be the same as rmax previously defined. The following arrays `vpx` and `vsx` are the *P* and *S*-wave velocities that correspond to each one of the rings. The densities for each ring are in the array `rho`.

```
#define media parameters
rx  = [0.1,3.0]         #radius borehole and for the rings
vpx = [1500.0, 4000.0]  #longitudinal velocity
vsx = [1e-30,  2300.0]  #shear velocity
rho = [1000.0, 2500.0]  #density
```

Number of time steps for the simulation `TimeIter` and size (number of grid points) of the absorbing boundaries in the computational domain Tap.

```
# define Time Iteration
TimeIter  = 5500
#absorbing layer size (absorbing boundaries)
Tap       = 20
```

The position and number of receivers are given in the three parameters: `pz = np.linspace(6.75,7.8,8,endpoint=True)`, where the first and last receivers are placed (in meters) and how many receivers do we have. The position of the receivers on the *r*-axis is given by `pr` (in meters).

```
# Define Position of Receivers
pz = np.linspace(6.75,7.8,8,endpoint=True)
```

```
pr = [0.01]*np.size(pz,0)
########
FD = FD25D(scenario, source, rx, vpx, vsx, rho)
```

Set the device you want to use, choose between “GPU” or “CPU”. Note that NVIDIA cards do not let to use the “CPU” option using its drivers. In this case one needs to use the OpenCL Intel Drivers.

```
FD.InitCL("GPU")  # change to "CPU" when running on a CPU
FD.Setup(Tap)
FD.MediumSetup()  #
```

In the case one wants to consider a layer intercepting the borehole the last parameter in the following instruction has to be True, otherwise the default value is set to False. The properties of the intercepting layer are `LayerInterProps=[2200, 1800, 967]`= LayerInterProps=[Density(kg/m^3), Vp (m/s), Vs(m/s)], the position of the layer is set by `LayerInterDim=[3.0, 3.5]`, (in meters) z1<z2.

```
FD.MediumSetup(LayerInterProps=[2200, 1800, 967],
               LayerInterDim=[3.0, 3.5],
               LayerIntercepting=True)
```

In addition, the program allows one to include rings of different materials that surround the fluid-filled borehole. This will help when studying the effect of invasion of the drilling mud in the formation that is in contact with the borehole. We use these parameters to produce the example in Fig. 5.

```
rx  = [0.1,  0.5,  3.0]       #radius borehole
vpx = [1500.0, 3500, 4000.0]  #longitudinal velocity
vsx = [1e-30,  2000, 2300.0]  #shear velocity
rho = [1000.0, 2300, 2500.0]  #density
```