# Neural Networks Applied to Signal Separation on Multi-Sensor Array Data

## Neural networks architectures developed for the Independent Research Project (ACSE 9)

Here are a set of U-shaped networks that have been developed and tested to perform seismic signal separation (a.k.a de-blending). The code is written in Python 3.6 and the neural networks are implemented with the PyTorch platform (version 1.0.1.post2).

All the experiments presented in the Report have been run using UNetRes.py, UNetSkip.py, UNetInPlus.py and UNetDeep3.py. An additional version UNetPro.py, with more input parameters, is also given. This is intended to be used in a production stage.

## Examples of how to instantiate the networks

Network in the script UNetRes.py can be instantiated as:

from UNetRes import UNet

model = UNet(input\_channels, output\_channels, [64, 96, 144, 216, 324, 486, 730, 1096, 1644], 636, 1251)

This will generate the reference model, with 9 down-blocks and 8 down-blocks.  
input\_channels is the number of channels in th input tensor.  
output\_channels is the number of channels in the output tensor.  
The list of integers gives the number of kernels in the convolutions for each block.  
The length of this list is also the number of down-blocks.  
The last two integers are the width and height of the input tensor. Its third and fourth dimensions.  
Note that these last two inputs will be removed in the code used in production. They are introduced here only for developing reasons, in particular, keep my modifications to the machine learning pipeline to a minimum. Networks in UNetSkip.py, UNetInPlus.py and UNetDeep3.py can be used in a similar way

Network in the script UNetPro.py can be instantiated as:

from UNetPro import UNet

blocks = [64, 96, 144, 216, 324, 486, 730, 1096]

input\_channels = 3

output\_channels = 1

inWidth = 636

inHeight = 1251

residual = False

iniPad = [0, 0, 0, 0]

model = UNet(input\_channels, output\_channels, blocks, inWidth, inHeight, residual, iniPad)

The first 5 inputs are as above.  
Residual: boolean, default is False. If true, residual learning is implemented in all the blocks. iniPad: list of four integers. It allows padding of the input tensor.