Introduction to Neo for Use With Spike2

* Whenever you are lost, you can type help(*confusing method or object here*).

Getting Started: the Basics

1. Open a command prompt (Terminal on Macs) and type `python` to open the Python Interactive Shell.

2. Import Spike2IO from Neo. Type

```
from neo import Spike2IO
```

3. Instantiate (create) a Spike2IO object using your Spike2 file which should be in SMR format. Say you want to name the object ‘neo_object'. Type

```
neo_object = Spike2IO(‘filename’, ced_units=True)
```

The second argument is important in order to maintain any wavemark information. You may have to type either single or double quotes around the filename. Neo IO does not seem to be consistent in which it will accept, but they do need to be consistent with each other (i.e. both double quotes or both single quotes).

4. Read the Spike2IO object. Type

```
block_list = neo_object.read(lazy=False, cascade=True)
```

This returns one list of one Block object. The only seemingly useful information you have access to at this point is the ‘name’ and ‘segments’ attributes of the Block object. As these are attributes of a Block object, they can only be accessed by operating on block_list[0], the first element in the list of length one of Block objects. In order to get the full list of attributes for a Block object, type dir(block_list[0]). Unfortunately, many of these attributes do not actually contain any information.

The lazy and cascade parameters indicate how much information to load in. The arguments shown above are the default values and load in all information. For more details on these parameters, see https://pythonhosted.org/neo/io.html#module-neo.io.

5. In order to gather more information about the file contents, you will need to access a lower level object. To get access to the Segment object within the Block object, type

```
segment_list = block_list[0].segments
```

Similar to before, this returns one list of one Segment object. However, now you have access to more useful attributes via the Segment object — just be sure you type segment_list[0] to access the Segment object itself first.