

Introduction to quantitative data analysis and MATLAB

Neurophysiology invariably results in the collection of a great deal of neural data that must be analyzed to make interpretations about its meaning. There is essentially an infinite number of ways that neuronal data can be analyzed and a substantial fraction of the neuroscience community is involved in developing and improving such methods. Given this, a major part of learning to do neurophysiological research involves learning about how to analyze the neuronal data that you collect. Indeed, most scientific research involves first thinking about and planning the analyses that will be applied to the collected data to get at the major questions of interest. This is part of good experimental design and it touches on many fields, but especially applied mathematics and statistics.

Thoughtful experimental design is especially important in neurophysiological research because the data often take a great deal of effort to collect and one would not want to invest such effort only to later realize that he or she had not collected the type or amount of data that was most appropriate for the question(s) of interest. Thus, most neurophysiological data is not collected until the experimenter has in mind the kinds of analysis that will be applied to the data once it is obtained. This does not mean that those analyses are the only ones that will ultimately be applied to that data, but they are typically the first analyses that are applied.

Because neurophysiological data can be analyzed in many ways and those methods are often tailored to the details of each experiment, it turns out to be extremely useful to have very flexible analysis tools at your disposal. Basic, off-the-shelf statistical packages allow you to put data in a spreadsheet and run functions like t-tests and make basic plots (e.g. Microsoft EXCEL). These programs work very well for data that has a relatively simple form. For example, is the number of sick animals that improved with drug injection X greater than the number of sick animals that improved with a control saline injection? However, as mentioned above, neurophysiological data is often much more complex and thus cannot be easily handled by such packages. For example, each recorded neuron produces a voltage signal that varies in time as a function of a potentially large number of conditions that the subject was exposed to (e.g. many visual patterns presented to the eye). What aspects of that voltage signal should be analyzed? How can the data be processed to make those aspects explicit? Because this one neuron is part of a large population of neurons, how can we analyze the combined output of the entire population?, etc...

Because all modern neurophysiological data is converted to digital format (using analog to digital converters), the most flexible analysis tools are programming languages that can read that digital data and manipulate it in any possible manner. Examples of such programming languages are FORTRAN (old), C and C++. Although these programming

languages have essentially infinite flexibility, and, once compiled into applications, are very fast, these advantages come at the price of having to write many basic routines from scratch (e.g. plotting routines, statistical routines, etc.). Moreover, these languages are very unforgiving in that programming errors can easily crash the application and can be difficult to detect.

It turns out that the best environments to analyze neuophysiological data are those that provide many basic, well-tested routines, are relatively robust to programming errors, can be easily transferred to different platforms, and still allow an essentially unlimited ability to analyze and view data in different ways. One example of such an environment in very widespread use today is MATLAB (Mathworks, Natick, MA). The main downside of such programs is that they tend to be less efficient than (e.g. C++) programs in terms of speed and computer memory use. Nevertheless, these disadvantages are typically not critical for most analyses (especially for during the creative phase of developing new analyses) and, even if they do prove to be limitations, there are ways to transfer MATLAB analysis code into compiled software that can run fast and efficiently.

Because of MATLAB's continued widespread use in engineering and science and its usefulness in analyzing neurophysiologic data, part of the goal of this course is to teach you the basics of MATLAB and to then use MATLAB to perform basic analyses on the data you collect. The flexibility of MATLAB will thus allow you to perform experiments and analyses that go beyond the basics provided in the lab. That is, it will free you to be creative and possibly discover things that are not built into the course.