**Advice for testing numerical algorithms**

Before we begin this week's assignment, there are some advice that we would like to give for writing functions that work with numerical data. They are useful for finding bugs in your implementation.

Testing machine learning algorithms (or numerical algorithms in general) is sometimes really hard as it depends on the dataset to produce an answer, and you will never be able to test your algorithm on all the datasets we have in the world. Nevertheless, we have some tips for you to help you identify bugs in your implementations.

**1. Test on small dataset**

Test your algorithms on small dataset: datasets of size 1 or 2 sometimes will suffice. This is useful because you can (if necessary) compute the answers by hand and compare them with the answers produced by the computer program you wrote. In fact, these small datasets can even have special numbers, which will allow you to compute the answers by hand easily.

**2. Find invariants**

Invariants refer to properties of your algorithm and functions that are maintained regardless of the input. We will highlight this point later in this notebook where you will see functions, which will check invariants for some of the answers you produce.

Invariants you may want to look for:

1. Does your algorithm always produce a positive/negative answer, or a positive definite matrix?
2. If the algorithm is iterative, do the intermediate results increase/decrease monotonically?
3. Does your solution relate with your input in some interesting way, e.g. orthogonality?

Finding invariants is hard, and sometimes there simply isn't any invariant. However, DO take advantage of them if you can find them. They are the most powerful checks when you have them.