PIC 16, Winter 2018 – Preparation 8W

Assigned 2/23/2018. To be completed by class 2/28/2018.

Intended Learning Outcomes

By the end of this preparatory assignment, students should be able to:

- solve systems of nonlinear algebraic equations,
- fit curves,
- and optimize functions (subject to constraints)

using `scipy.optimize`.

Tasks

- [ ] In Assignment 1F, you used Newton's method to find a root of a function. SciPy's `fsolve` and `root` functions are efficient implementations of Newton's method (with improvements) that work in multiple dimensions, that is, they can find the root of systems of nonlinear equations in several variables simultaneously. All you have to do is provide the function that you want to zero and a guess for the independent variables. SciPy automatically approximates derivatives/gradients as necessary and returns a set of values for the independent variables that zero the function (if it can find one).
- [ ] Assignment 1W links to some references if you need them, and you can also check out the first 30 minutes or so of the Track A Math Review (Part I) if this is really unfamiliar.
- [ ] Newton's method can be modified slightly in order to do optimization, that is, finding the minimum/maximum of an equation. If you're unfamiliar with Newton's method for optimization, watch this. You can also check out the followup video if you're interested.
- [ ] SciPy's `minimize` function contains efficient implementations of such a method (with many improvements) that work in multiple dimensions; they can find the values of several variables that minimize a (single-valued) function of those variables. All you have to do is provide the function and a guess; SciPy automatically approximates derivatives/gradients and second derivatives/hessians as necessary and returns the solution if possible.
- [ ] Follow SciPy Lectures 1.5.5 – Optimization and Fit.
- [ ] Track A Math Review (Part II) covers minimization with constraints (nonlinear programming) in detail. It's probably not needed to finish the preparation, but might be helpful for the assignment.
- [ ] For the "Finding the minimum of a scalar function" try using the `SLSQP` method rather than limited memory BFGS. (Find documentation for `scipy.optimize.minimize`.)
- [ ] For the "Finding the roots of a scalar function", use the `root` function with the `hybr` method. (Find documentation for `scipy.optimize.root`.)
- [ ] In addition to doing the "Curve fitting" example using the `curve_fit` function, improve your understanding of both the `curve_fit` function and the `minimize` function by trying to fit the same curve using the more general `minimize` function. That is, you are trying to find the values of `a` and `b` that minimize the L2-norm (square root of the sum of the squares) of the error (difference) between the function `f2` (evaluated at the points `xdata`) and the `ydata`. So:
  - write a function (or lambda function) that:
    - accepts *only* one argument, an array `z`;
    - sets `a` to the zeroth element of `z`;
    - sets `b` to the first element of `z`;
    - for these `a` and `b`, calculates `f2` at the `xdata`;
    - calculates the error, the difference between these values and the `ydata`;

        - takes the L2-norm (see `scipy.linalg.norm`, which you should have found in the last preparation) of these differences; and
        - returns this value, which is a single number that represents how well the function `f2` fits the data. We are trying to minimize this value, because that means that there is little error between `ydata` and `f2` evaluated at the `xdata`.
    - `minimize` this function. Your guess is an array with two values (guesses for `a` and `b`).
    - NumPy will use a variant of Newton's method for optimization to modify the values in the array (the values of `a` and `b`) until your function, and thus the error between `ydata` and `f2` and at the `xdata` points, is minimized. The object returned by `minimize` includes the array (with values for `a` and `b`) in a field called `x`. These values should be nearly identical to those returned by `curve_fit`.
    - As you can see, `minimize` can do the same job as `curve_fit`, but it's far more general. Modify your program such that the L1-norm (sum of absolute values) rather than the L2 norm (root of sum of squares) of errors is minimized. The values returned will be a little different because we've re-defined what it means to "best fit" the data.
- (Optional) Follow 1.5.4 and 1.5.6 if you're interested.