AE588

Assignment 3: Unconstrained Optimization

umid: akshatdy

3.1

Implemented in uncon_optimizer.py

3.2

The algorithm uses

- search direction: BFGS
- line search: bracketing and pinpointing

First, the most basic methods for determining search direction (steepest descent) and line search (backtracking) were used to establish a baseline and make sure that the optimization loop was functioning correctly.

Then, as per the book's recommendation, the best algorithms taught till now that do not require a Hessian were implemented

Here is a comparison between the 2 approaches

Problem	Steepest Descent + Backtracking	BFGS + Bracketing and Pinpointing
Slanted Quadratic	82	8
Bean	86	18
2D Rosenbrock	DNF, exceeded fcalls	54

3.3

I chose to compare scipy's BFGS implementation for the most apples-to-apples comparison

Problem	Steepest Descent + Backtracking	BFGS + Bracketing and Pinpointing	Scipy
Slanted Quadratic	82	8	7
Bean	86	18	11
2D Rosenbrock	DNF, exceeded fcalls	54	24

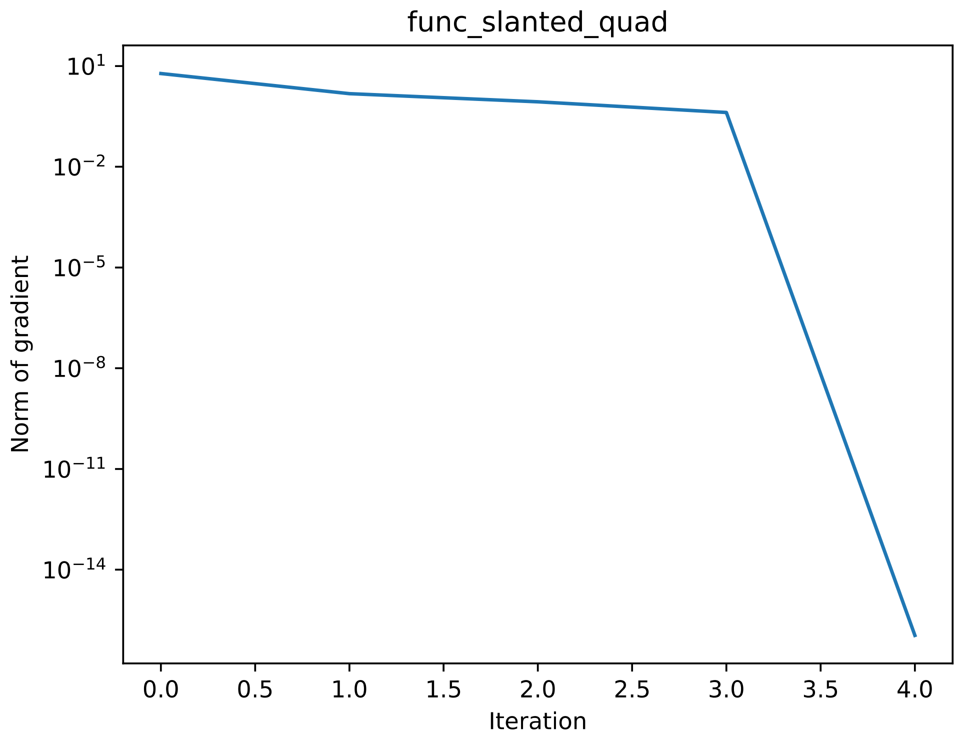
My optimizer is around 50% slower for harder problems like rosenbrock, but it seems like the basic implementation of steepest descent and backtracking is not performing well. There might be some bugs in it that make it so slow, or there might be a need to tune the optimization parameters like sufficient decrease to make it converge faster.

3.4

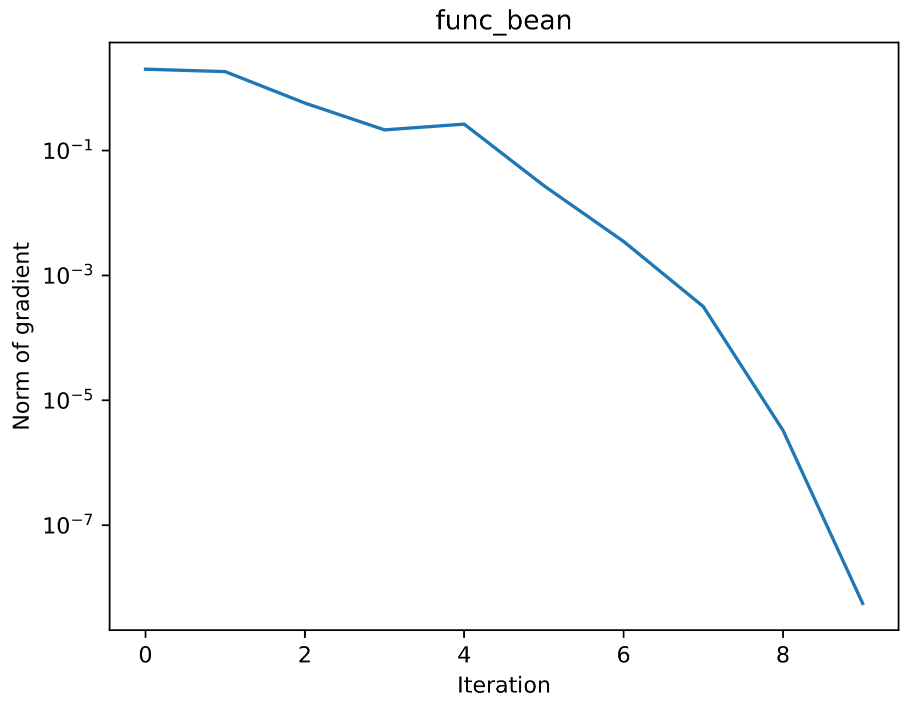
Convergence plots for BFGS + Bracketing and Pinpointing

Inf norm of gradient vs iterations

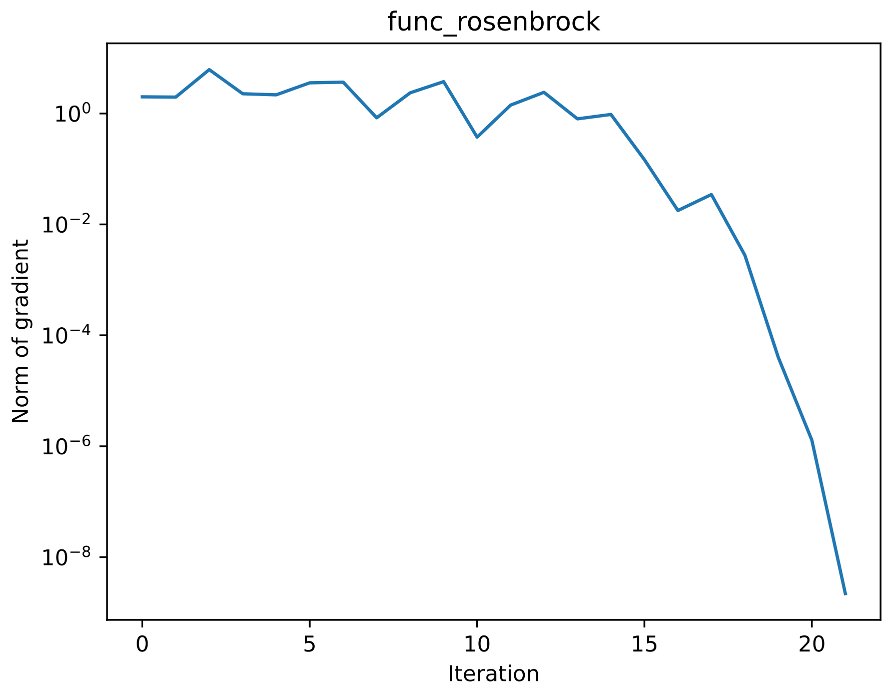
Slanted Quadratic



Bean



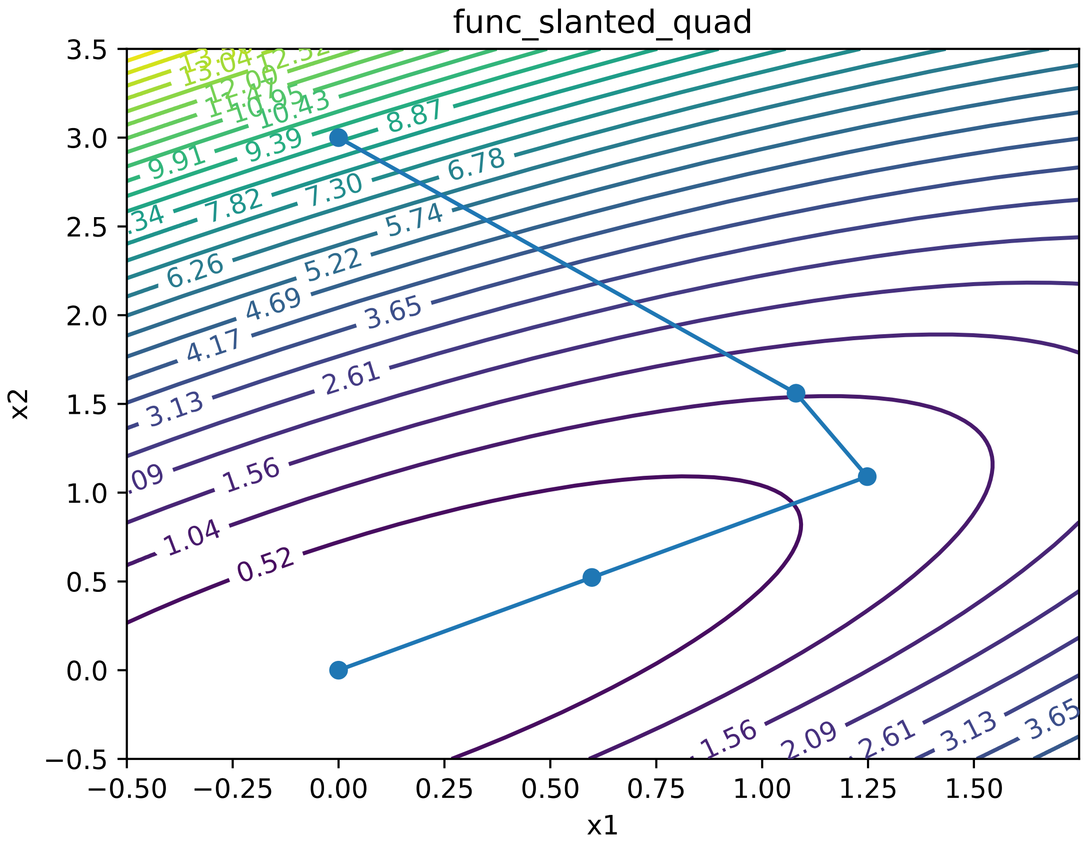
2D Rosenbrock



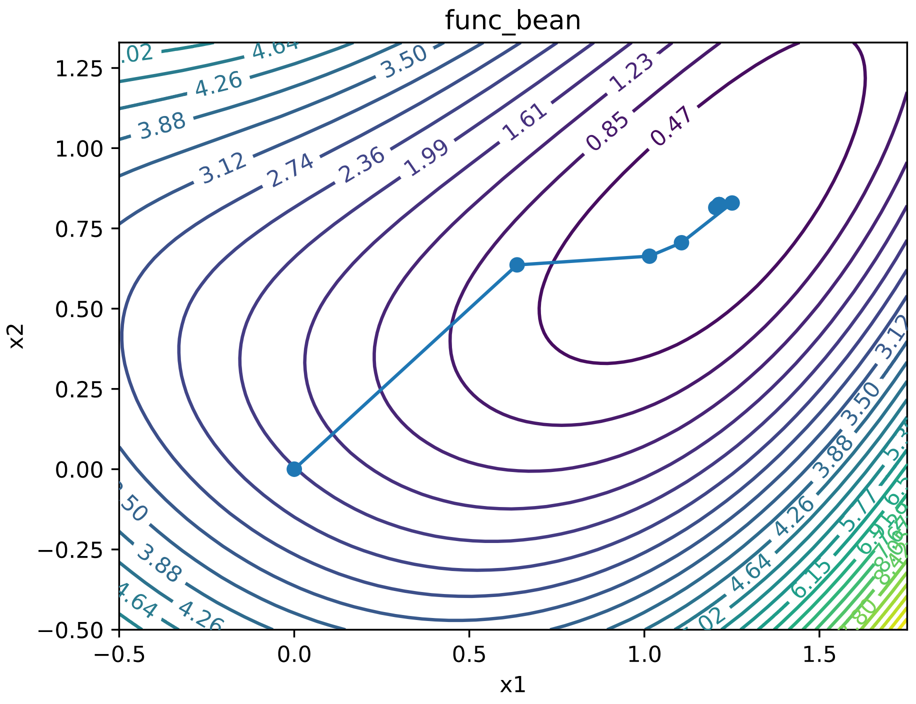
3.5

Optimization Path for BFGS + Bracketing and Pinpointing

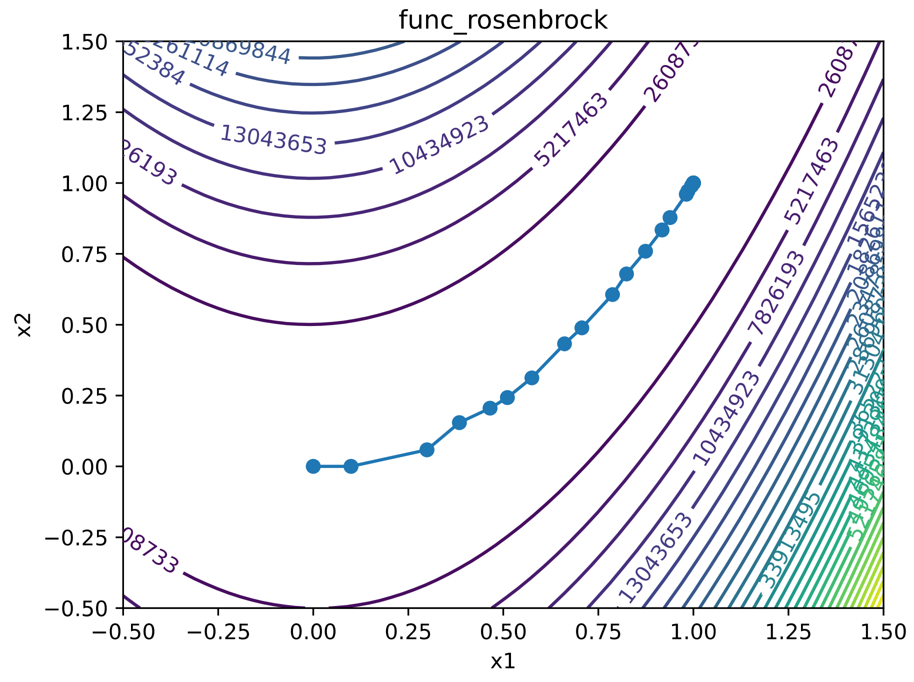
Slanted Quadratic



Bean



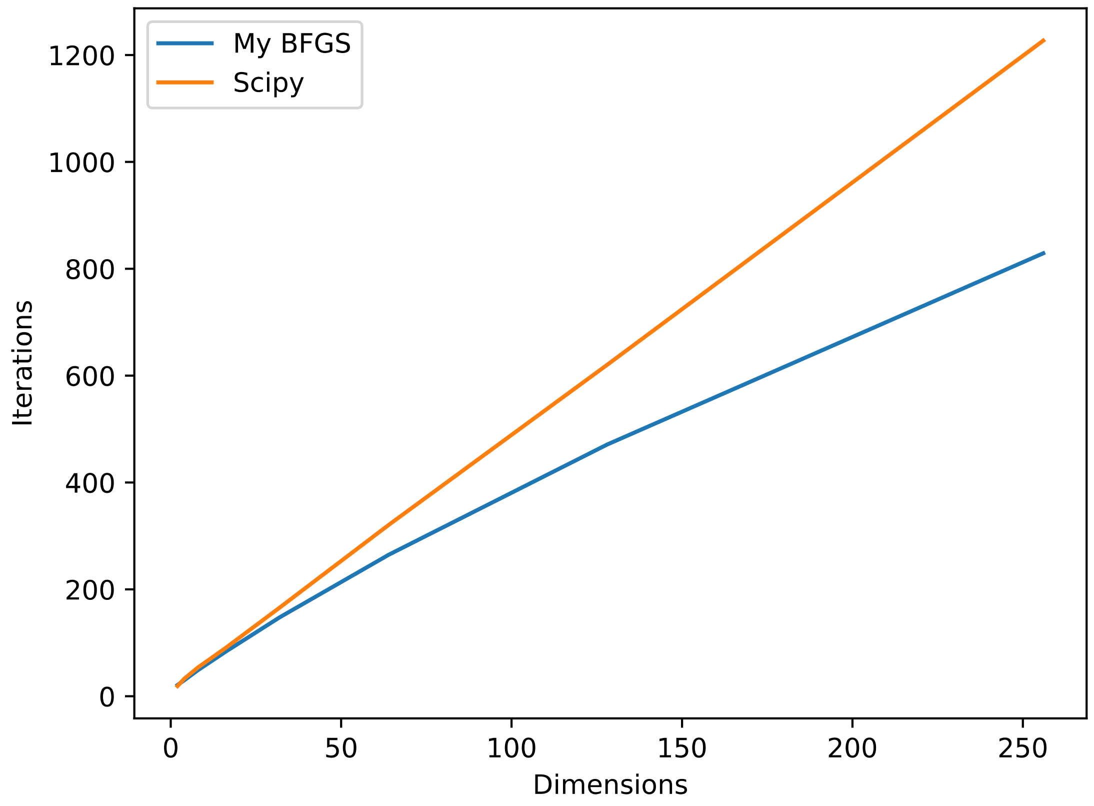
2D Rosenbrock



3.6

Comparison of scipy vs my own BFGS implementation for solving the n dimensional rosenbrock problem

My implementation seems to take fewer iterations, but as we saw before scipy actually has fewer function calls so it might ultimately be faster



3.7

Main challenge: Debugging the correctness of the algorithms

Lessons Learned: collaborate more with others to compare correctness

Areas of improvement: do a search for the optimal optimizer parameters to see if there is any scope for improvement