Tutorial 3: Newton & Quasi-Newton Methods

- Implement Newton's method with the Hessian modification.
  - Use your (or mine) steepest descend code as a starting point.
  - Use Matlab's eigenvalue functions, `eig`, to compute the eigenvalue.
  - Use Matlab's backslash operator to solve the Newton system.
  - Test the code again on our examples.
- Implement a quasi-Newton (or limited memory BFGS) method.

Theory of Newton's Method

1. Show that Newton's method oscillates for the example $\min\ f(x) = x^2 - x^4/4$.
1. Show that the quasi-Newton condition, $B\gamma = \delta$ holds for a quadratic function.
1. Show that the rank-one formula terminates for a quadratic:
   - Show by induction that $H^{(k+1)}\gamma^{(j)} = \delta^{(j)}$ for all $j = 1, \ldots, k$.
   - Hence conclude that the method terminates after $n+1$ iterations.
1. Code BFGS or limited-memory BFGS method in Matlab.
1. Apply Newton's method to nonlinear least-squares:

$$\underset{x}{\text{minimize}}\ f(x) = \sum_{i=1}^{m} r_i(x)^2 = r(x)^T r(x) = \|r(x)\|_2^2.$$

What happens, if $r_i(x)$ are linear? Can you propose a strategy for handling the case, where $\nabla^2 r_i(x)$ are bounded, and $r_i(x) \to 0$?

Tutorial 3: Newton and Conjugate Gradient Methods

1. Implement the Barzilai-Borwein family of methods.
   - Modify `SteepestDescend.m` or your own code.
   - Try the method on quadratic problem, with $G = \text{tri}(-1, 4 - 1)$ a tri-diagonal Matrix with 4 on the diagonal, and -1 on both off-diagonals. You can try several g's, e.g. $G = (1, \ldots, 1)^T$.
   - Compare it to steepest descend.
   - Try Powell's problem