UNIVERSITÄT BERN

Applied Optimization
Exercise 7 - L-BFGS

Heng Liu   Nicolas Gallego

Handout date: 14.11.2019

Submission deadline: 20.11.2019, 23:59 h

Hand-in instructions:

Please hand-in **only one** compressed file named after the following convention: `Exercise`n`-GroupMemberNames.zip`, where n is the number of the current exercise sheet. This file should contain:

- **Only** the files you changed (headers and source). It is up to you to make sure that all files that you have changed are in the zip.
- A `readme.txt` file containing a description on how you solved each exercise (use the same numbers and titles) and the encountered problems.
- Other files that are required by your `readme.txt` file. For example, if you mention some screenshot images in `readme.txt`, these images need to be submitted too.
- Submit your solutions to ILIAS before the submission deadline.

Secant Equation (1 pts)

Secant equation is:

$$B_{k+1}s_k = y_k,$$

where $s_k = x_{k+1} - x_k, y_k = \nabla f_{k+1} - \nabla f_k$, and B_{k+1} is an approximation of the Hessian $\nabla^2 f_k$. Derive the Secant equation.

Curvature Condition (1 pts)

Show that the curvature condition $s_k^T y_k > 0$ is required for the Quasi-Newton method. Prove that the curvature condition is satisfied when the line search algorithm for the Wolfe conditions is used.

Programming (8 pts)

L-BFGS (6 pts)

Implement the L-BFGS algorithm that is described in the lecture slides in the `solve(...)` function given in the `LBFGS.hh` file. For the step length, you can use the `backtracking_line_search` in the `LineSearch.hh`. Hint: whenever the back tracking line search fails to find the step length that meet the curvature condition, you need to skip the update.

Comparison of different methods (2 pts)

With the provided fixing node condition and the random initialization, use different algorithms, including gradient descent, newton, projected newton, L-BGFS, to optimize the mass spring system with the SpringElement2DWithLength of different dimensions ($10 \times 10, 30 \times 30, 50 \times 50$). Compare the results from different algorithms with the default parameters. Make a table with the statistics, including the number of iterations, run time, convergence precision. If your are allowed to tune the parameters, redo the experiment and identify the fastest algorithm.

A line search algorithm for the Wolfe conditions (Bonus (6 pts))

On page 60 of the book "Numerical Optimization", there is a description of a line search algorithm that satisfies the Wolfe conditions. Implement the algorithm in the function `wolfe_line_search(...)` in the `LineSearch.hh` file. Run the L-BFGS algorithm with this line search algorithm and compare with the back tracking line search.