Non-linear least squares (Pose graph) optimization using conjugate gradient algorithm

Subramanian Krishnan (subramak)

Non-linear least square minimization is extremely common in SLAM and computer vision - Bundle adjustment, photometric error etc. In most of these systems, the objective is to estimate a state vector given a set of non-linear constraints. The standard approach is to linearize the system at the current estimate and apply iterative gauss-newton or factorization to find the minima. This process is repeated until the solution converges for the non-linear constraints/equations. What I'd like to try is to use conjugate gradient learned in class to see how well it solves the linearized system. In SLAM, the state vector tends to grow really fast and I want to investigate if conjugate gradient helps converge to the solution faster than factorization or Gauss Newton. Generally two metrics are tested: accuracy and real-time factor/computation time. I'd like to test the computation time specifically (although this may change based on how the project moves). I'm thinking of using the Manhattan dataset to test the solver. I'm looking to write as many methods from scratch as possible in mostly python (could be C++ - not sure right now) to get a taste of implementation difficulty. An extension I'm thinking of - is to see whether the conjugate gradient algorithm can be used directly to solve for the non-linear system instead of post linearization. The cost functions to be tested will be convex. Some libraries I'd use as a reference are g2o, Toro etc.