Nonlinear Optimization
Trust-region methods

Niclas Börlin
Department of Computing Science
Umeå University
niclas.borlin@cs.umu.se

December 3, 2007

Line search and trust-region

- Line search and trust-region and two examples of *global strategies* that modify a (usually) locally convergent algorithm, e.g. Newton, to become globally convergent.
- At every iteration \( k \), both global strategies enforce the descent condition

\[ f(x_{k+1}) < f(x_k) \]

by controlling the length and direction of the step.

The trust-region model

- Trust-region methods use the quadratic model

\[ m_k(p) = f_k + p^T g_k + \frac{1}{2} p^T B_k p. \]

\[ f_k = f(x_k), \quad g_k = \nabla f(x_k). \]

- Newton-type trust-region methods have \( B_k = \nabla^2 f(x_k) \).
- The model is “trusted” within a limited region around the current point \( x_k \) defined by

\[ ||p|| \leq \Delta_k. \]

This will limit the length of the step from \( x_k \) to \( x_{k+1} \).
- The value of \( \Delta_k \) will be adjusted up if the model is found to be in “good” agreement with the objective function, and down if the model is a “poor” approximation.
The trust-region subproblem

- At iteration $k$ of a trust-region method, the following subproblem must be solved:

$$
\min_{\rho} m_k(\rho) = f_k + p^T g_k + \frac{1}{2} p^T B_k p,
\text{ s.t. } ||p|| \leq \Delta_k
$$

- It can be shown that the solution $p^*$ of this constrained problem is the solution of the linear equation system

$$(B_k + \lambda I)p^* = -g_k$$

for some $\lambda \geq 0$ such that the matrix $(B_k + \lambda I)$ is positive semidefinite.

- Furthermore,

$$
\lambda(\Delta_k - ||p^*||) = 0.
$$

Note that if $B_k = \nabla^2 f(x_k)$ is positive definite and $\Delta_k$ big enough, the solution of the trust-region subproblem is the solution of

$$
\nabla^2 f(x_k) p = -\nabla f(x_k),
$$

i.e. $p$ is a Newton-direction.

- Otherwise,

$$
\Delta_k \geq ||p_k|| = ||(\nabla^2 f(x_k) + \lambda I)^{-1}\nabla f(x_k)||,
$$

so if $\Delta_k \to 0$, then $\lambda \to \infty$ and

$$
p_k \to -\frac{1}{\lambda} \nabla f(x_k).
$$

- When $\lambda$ varies between 0 and $\infty$, the corresponding search direction $p_k(\lambda)$ will vary between the Newton direction and a multiple of the negative gradient.

The reduction ratio

- To enable adaption of the trust-region size $\Delta_k$, the reduction ratio

$$
\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} = \frac{\text{actual reduction}}{\text{predicted reduction}}
$$

is defined.

- If the reduction ratio is large, e.g. $\rho_k > \frac{3}{4}$, the trust-region size is increased in the next iteration.

- If the reduction ratio is small, e.g. $\rho_k < \frac{1}{4}$, the trust-region size is decreased in the next iteration.

- Furthermore, a step $p_k$ will only be accepted if the reduction ratio is not too small.
The trust-region algorithm

- Specify starting approximation $x_0$, maximum step length $\hat{\Delta}$, initial trust-region size $\Delta_0 \in (0, \hat{\Delta})$ and acceptance constant $\eta \in [0, \frac{1}{4})$.
- For $k = 0, 1, \ldots$ until $x_k$ is optimal
  - Solve
    $\min_{\mathbf{p}} m_k(\mathbf{p}) = f_k + \mathbf{p}^T \mathbf{g}_k + \frac{1}{2} \mathbf{p}^T \mathbf{B}_k \mathbf{p},$
    s.t. $\|\mathbf{p}\| \leq \Delta_k$
  - For a trial step $\mathbf{p}_k$
    - Calculate the reduction ratio
      $\rho_k = \frac{f(x_k) - f(x_k + \mathbf{p}_k)}{m_k(0) - m_k(\mathbf{p}_k)}$
      for $\mathbf{p}_k$.

- Update the current point
  $x_{k+1} = \begin{cases} x_k + \mathbf{p}_k & \text{if } \rho_k > \eta, \\ x_k & \text{otherwise}. \end{cases}$

- Update the trust-region radius
  $\Delta_{k+1} = \begin{cases} \frac{1}{4} \Delta_k & \text{if } \rho_k < \frac{1}{4}, \\ \min(2 \Delta_k, \hat{\Delta}) & \text{if } \rho_k > \frac{3}{4} \text{ and } \|\mathbf{p}_k\| = \Delta_k, \\ \Delta_k & \text{otherwise}. \end{cases}$

The Levenberg-Marquardt algorithm

- The first trust-region algorithm was developed for least squares problems by Levenberg (1944) and Marquardt (1963).
- The original algorithm uses the approximation $\mathbf{B}_k = \mathbf{J}_k^T \mathbf{J}_k$ and solves
  $(\mathbf{B}_k + \lambda_k \mathbf{I}) \mathbf{p} = -\mathbf{g}_k$
  for different values of $\lambda_k$.
- The original algorithm adapts by modifying the $\lambda$ value, i.e. if the reduction produced by $\mathbf{p}$ is good enough, $\lambda_{k+1} = \frac{1}{10} \lambda_k$, otherwise $\lambda_{k+1} = 10 \lambda_k$ and the step is rejected.
- The Levenberg-Marquardt algorithm was put into the trust-region framework ($\Delta$-parameterized) in the early 80-ies (Moré, 1981).
- The $\Delta$ version of Levenberg-Marquardts has a number of advantages over the $\lambda$ version:
  - $\lambda$ is nontrivially related to the problem. $\Delta$ is related to the size of $x$. E.g. $\Delta_0 = \|x_0\|$ is often a reasonable choice.
  - The transition to $\lambda = 0$ is handled transparently.
  - The $\lambda$ algorithm need to re-solve
    $(\mathbf{B}_k + \lambda_k \mathbf{I}) \mathbf{p} = -\mathbf{g}_k$
    when a step is rejected and $\lambda$ is reduced. The $\Delta$ algorithm has ways to avoid that.
- However, many popular implementation of Levenberg-Marquardt still use the original, $\lambda$-parameterized, formulation.
The Dogleg algorithm

- The trust-region subproblem
  \[
  \min_p \, m_k(p) = f_k + p^T g_k + \frac{1}{2} p^T B_k p,
  \quad \text{s.t. } \|p\| \leq \Delta_k
  \]
  is a hard problem.

- If the unconstrained solution
  \[
  p^B = -B_k^{-1} g_k
  \]
  is too long, \(\|p^B\| > \Delta_k\), we have to find a \(\lambda\) such that
  \[
  \|p_k(\lambda)\| = \|(B_k + \lambda I)^{-1} g_k\| = \Delta_k.
  \]
  This is a non-linear equation in \(\lambda\).

- The dogleg algorithm solves this problem by approximating the function \(p_k(\lambda)\) with a piecewise linear polygon \(\bar{p}(\tau)\) and solving \(\|\bar{p}(\tau)\| = \Delta_k\).

- The polygon \(\|\bar{p}(\tau)\|\) is defined as
  \[
  \bar{p}(\tau) = \begin{cases} \tau p^U, & 0 \leq \tau \leq 1, \\ p^U + (\tau - 1)(p^B - p^U), & 1 \leq \tau \leq 2. \end{cases}
  \]

- The point \(p^U\) is the Cauchy point, i.e. the minimizer of \(m\) along the steepest descent direction
  \[
  p^U = -\frac{g^T g}{g^T B g} g.
  \]

- The dogleg algorithm works only if \(B_k\) is positive definite, e.g. for least squares problems.