Numerical Optimization
Lecture Notes #8 — Trust-Region Methods
2-D Subspace Minimization & Iterative “Nearly Exact” Methods

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://terminus.sdsu.edu/

Fall 2018
Outline

1. Trust Region Methods
   - Recap
   - 2-D Subspace Minimization

2. Iterative "Nearly Exact" Solution to the Subproblem
   - Fundamentals...
   - Building a Scheme to Solve the Problem...
Quick Recap: Last Time — Trust-Region / Cauchy Point

- Introduction to Trust-region (TR) methods.
  - Use a quadratic model $m_k(\bar{p})$ around the current point $\bar{x}_k$; we trust the model in some neighborhood (⇒ name of this class of methods).
  - Optimize the model $m_k(\bar{p})$ to find the appropriate step $\bar{p}_k$.
  - We developed a macro algorithm for expanding and contracting the size of the TR, depending on how well the model approximates the objective locally.
  - We spend quite some effort in finding the Cauchy point, which if selected in each iteration guarantees global convergence to a stationary point.
  - We experienced a "rotten tomato moment" when we realized that the Cauchy point was just a rediscovery of the steepest descent algorithm, with a particular (possibly sub-optimal) step size.
  - We ducked the rotten tomatoes by introducing the first improvement over the Cauchy point, the Dogleg Method.
Method: Dogleg for Trust-region

Use When: The model Hessian $B_k$ is positive definite.

Figure: The dogleg path tries to approximate the optimal path, which describes the optimal solution to the trust-region sub-problem as a function of the trust-region radius. The dogleg step is simply the exit point of the dogleg path, or — in the case when it is feasible — the full step.
The trust region sub-problem is given by

\[
\tilde{p}_k = \arg \min_{\|\tilde{p}\| \leq \Delta_k} \left[ f(\bar{x}_k) + \tilde{p}^T \nabla f(\bar{x}_k) + \frac{1}{2} \tilde{p}^T B_k \tilde{p} \right].
\]

The full step is the unconstrained minimum of the quadratic model

\[
\tilde{p}_k^{FS} = -B_k^{-1} \nabla f(\bar{x}_k).
\]

The step in the steepest descent direction is given by the unconstrained minimum of the quadratic model along the steepest descent direction

\[
\tilde{p}_k^U = -\frac{\nabla f(\bar{x}_k)^T \nabla f(\bar{x}_k)}{\nabla f(\bar{x}_k)^T B_k \nabla f(\bar{x}_k)} \nabla f(\bar{x}_k).
\]
Algorithm: The Dogleg Step

If( $\|\bar{p}_k^U\| \geq \Delta_k$ ), then

$$\bar{p}_{DL}^k = \Delta_k \cdot \bar{p}_k^U / \|\bar{p}_k^U\|,$$

elseif( $\|\bar{p}_k^{FS}\| \leq \Delta_k$ ), then

$$\bar{p}_{DL}^k = \bar{p}_k^{FS},$$

else

$$\bar{p}_{DL}^k = \bar{p}_k^U + (\tau^* - 1)(\bar{p}_k^{FS} - \bar{p}_k^U)$$

where $\tau^* \in [1, 2]$ so that $\|\bar{p}_k^U + (\tau^* - 1)(\bar{p}_k^{FS} - \bar{p}_k^U)\|^2 = \Delta_k^2$

end

Next we look at improvements of the dogleg method.
Lookahead: Improvements to the Cauchy Point / Dogleg Method

We seek other improvements to the Cauchy point, with the goal of developing trust region methods with better convergence properties.

In this lecture, we look at:

2-D Subspace Method:
Another simplified model for the optimal solution to the model problem in the trust-region. Here, the search for an optimal solution of the model is restricted to a plane ($\mathbb{R}^n \rightarrow \mathbb{R}^2$).

Iterative “Nearly Exact” Solutions to the Subproblem:
Useful only for small problems; costlier per iteration, but needs fewer iterations. (*) Definition of “small” may vary.
Method: 2-D Subspace Minimization for Trust-region

Use When: The model Hessian $B_k$ is positive definite, and (modified version) when $B_k$ is indefinite.

Without too much extra effort, we can expand the minimization from the dogleg path to the entire plane spanned by the steepest descent vector $\bar{p}_k^U$ and the full step vector $\bar{p}_k^{FS}$,

$$\arg\min_{\bar{p}} \left[ f(\bar{x}_k) + \bar{p}^T \nabla f(\bar{x}_k) + \frac{1}{2} \bar{p}^T B_k \bar{p} \right]_{m_k(\bar{p})}, \text{ s.t. } \|\bar{p}\| \leq \Delta_k, \bar{p} \in \text{span}[\bar{p}_k^U, \bar{p}_k^{FS}]$$

**Figure:** The two-dimensional subspace spanned by $\bar{p}_k^U$ and $\bar{p}_k^{FS}$, with the dogleg path indicated.
The plane is parameterized by \((\eta_1, \eta_2)\):

\[
\bar{p}(\eta_1, \eta_2) = \eta_1 \bar{p}_k^U + \eta_2 \bar{p}_k^{FS}.
\]

Hence we are reduced to solving the problem

\[
\tilde{\eta}_k = \arg \min_{\eta_1, \eta_2} \left[ f(\bar{x}_k) + (\eta_1 \bar{p}_k^U + \eta_2 \bar{p}_k^{FS})^T \nabla f(\bar{x}_k) + \frac{1}{2} (\eta_1 \bar{p}_k^U + \eta_2 \bar{p}_k^{FS})^T B_k (\eta_1 \bar{p}_k^U + \eta_2 \bar{p}_k^{FS}) \right],
\]

subject to \(\| \bar{p}(\eta_1, \eta_2) \| \leq \Delta_k\).

This is a minimization of a quadratic model in two variables — \(m_k(\eta_1, \eta_2)\); to find the minimum, we set

\[
\left[ \frac{\partial}{\partial \eta_1} m_k(\eta_1, \eta_2), \frac{\partial}{\partial \eta_2} m_k(\eta_1, \eta_2) \right]^T = \bar{0}.
\]


\[
\begin{bmatrix}
\frac{\partial}{\partial \eta_1} m_k(\eta_1, \eta_2), \\
\frac{\partial}{\partial \eta_2} m_k(\eta_1, \eta_2)
\end{bmatrix}^T = \bar{0}
\]

is a set (2-by-2) of linear equations, and with “some” algebraic manipulation we can explicitly write down the solution \((\eta_1, \eta_2)\).

The optimal parameters \((\eta_1, \eta_2)\) are given by

\[
\begin{bmatrix}
\eta_1 \\
\eta_2
\end{bmatrix} = - \begin{bmatrix}
(\bar{p}_k^U)^T B_k \bar{p}_k^U & (\bar{p}_k^U)^T B_k \bar{p}_k^{FS} \\
(\bar{p}_k^{FS})^T B_k \bar{p}_k^U & (\bar{p}_k^{FS})^T B_k \bar{p}_k^{FS}
\end{bmatrix}^{-1} \begin{bmatrix}
(\bar{p}_k^U)^T \nabla f(\bar{x}_k) \\
(\bar{p}_k^{FS})^T \nabla f(\bar{x}_k)
\end{bmatrix},
\]

as long as this gives \(\bar{p}(\eta_1, \eta_2) \in T_k\); otherwise we perform a one-parameter search \(\eta_1 \in [0, \Delta_k/\|\bar{p}_k^U\|] \iff \eta_2 \in [0, \Delta_k/\|\bar{p}_k^{FS}\|]\) so that

\[
\eta_1^2\|\bar{p}_k^U\|^2 + \eta_2^2\|\bar{p}_k^{FS}\|^2 + 2\eta_1\eta_2\langle \bar{p}_k^U, \bar{p}_k^{FS}\rangle = \Delta_k^2.
\]
Some basic linear algebra can greatly simplify the one-parameter search \( \eta_1 \in [0, \Delta_k / \| \bar{p}_k^U \|] \Leftrightarrow \eta_2 \in [0, \Delta_k / \| \bar{p}_k^{FS} \|] \) so that
\[
\eta_1^2 \| \bar{p}_k^U \|^2 + \eta_2^2 \| \bar{p}_k^{FS} \|^2 + 2 \eta_1 \eta_2 \langle \bar{p}_k^U, \bar{p}_k^{FS} \rangle = \Delta_k^2.
\]

Use **Gram-Schmidt** to build an orthonormal basis for the 2D subspace —

\[
\bar{q}_k^1 = \frac{\bar{p}_k^U}{\| \bar{p}_k^U \|}, \quad \bar{q}_k^2 = \frac{\bar{p}_k^{FS} - \langle \bar{p}_k^{FS}, \bar{q}_k^1 \rangle \bar{q}_k^1}{\| \bar{p}_k^{FS} - \langle \bar{p}_k^{FS}, \bar{q}_k^1 \rangle \bar{q}_k^1 \|}
\]

now, \( \text{span}(\bar{q}_k^1, \bar{q}_k^2) = \text{span}(\bar{p}_k^U, \bar{p}_k^{FS}) \) and we can use to search for \( \bar{p}(\theta) = \Delta \left( \cos(\theta) \bar{q}_k^1 + \sin(\theta) \bar{q}_k^2 \right) \)

\[
\theta^* = \arg \min_{\theta \in [0, 2\pi]} m_k(\bar{p}(\theta)) \quad \Rightarrow \quad \bar{p}(\theta^*) : \| \bar{p}(\theta^*) \| = \Delta.
\]
The main advantage of this approach is that we can extend it to the case where $B_k$ is indefinite.

When $B_k$ has negative eigenvalues, the 2-D subspace minimization is performed over a different subspace: —

Since we no longer safely can compute the full step $\bar{p}_k^{FS} = -B_k^{-1}\nabla f(\bar{x}_k)$, we define a new (modified) step $\bar{p}_k^*$ to replace the full step

$$\text{span}(\bar{p}_k^U, \bar{p}_k^{FS}) \rightarrow \text{span}(\bar{p}_k^U, \bar{p}_k^*)$$

The goal is that $\bar{p}_k^*$ should be “close” to $\bar{p}_k^{FS}$, but guaranteed safe to compute.
How do we select $\bar{p}_k^*$?

Let $\lambda_1$ denote the most negative eigenvalue of $B_k$, and select a parameter $\beta \in (-\lambda_1, -2\lambda_1]$. Then the matrix $(B_k + \beta I)$ must be positive definite.

We now define

$$\bar{p}_k^* = -(B_k + \beta I)^{-1} \nabla f(\bar{x}_k)$$

**Note:** In the case $\|\bar{p}_k^*\| \leq \Delta_k$ the modified full step is feasible, so we do not have to perform the subspace optimization. **However**, we do not use the step $\bar{p}_k^*$...

Instead, the step is defined to be $\bar{p}_k = \bar{p}_k^* + \bar{v}$ where the vector $\bar{v}$ satisfies $\bar{v}^T \bar{p}_k^* \geq 0$ (so that $\|\bar{p}_k\| \geq \|\bar{p}_k^*\|$). This is a safeguard which makes sure that $\bar{p}_k$ does not approach 0, in which case the algorithm is not making any progress. [Details Follow...]
Note: The computation of $\beta$ for $(B_k + \beta I)$ can be computed by e.g. the Lanczos method(*)


Note: For descriptions on the selection of $\bar{v}$, see e.g.
Lanczos method is an iterative method for computing the eigenvalues of a Hermitian (in the real case, symmetric) matrix.

**Stage #1:** It is easy (Math 543), to find an orthonormal similarity transformation $QB_kQ^T = T_k$ so that $T_k$ is triangular.

**Stage #2:** The Lanczos Iteration —
Using a 3-term recurrence relation it produces the coefficients for another sequence of symmetric tri-diagonal matrices, whose eigenvalues converge to the eigenvalues of the original matrix; in particular we get convergence to the outliers, i.e. largest and smallest eigenvalues, first.
According to Byrd-Schnabel-Schultz (1988), we let

\[
\bar{p}_k^* = -(B_k + \beta I)^{-1}\nabla f(\bar{x}_k),
\]

and when

\[
\|\bar{p}_k^*\| \leq \Delta_k,
\]

we let \(\bar{v} = \xi \bar{v}_-\), where \(\bar{v}_-\) is a vector in the negative curvature direction of \(B_k\), and \(\xi\) is chosen so that

\[
\|\bar{p}_k\| = \|\bar{p}_k^* + \xi \bar{v}_-\| = \Delta_k,
\]

in particular, the Rayleigh Coefficient must indicate sufficient negative curvature in the \(\bar{v}_-\)-direction

\[
\frac{\bar{v}_-^T B_k \bar{v}_-}{\bar{v}_-^T \bar{v}_-} \in \left[ \lambda_1(B_k), \frac{1}{1 + \rho \lambda_1(B_k)} \right], \quad \text{where } \rho \in (0, 1).
\]
Figure: Visualization of $\bar{p}_k^U$, $\bar{p}_k^*$, and $\bar{v}_-$: [LEFT] The modified full step $\bar{p}_k^*$ falls inside the trust region; [CENTER] From, $\bar{p}_k^*$ we follow a direction of negative curvature (of the original un-shifted $B_k$) $\bar{v}_-$ out to the trust-region boundary; [CENTER+RIGHT] the direction $\bar{v}_-$ does not necessarily fall in the plane spanned by $\bar{p}_k^U$ and $\bar{p}_k^*$, so we end up exploring a 3rd dimension.
Interpretation: $+\beta I$

Figure: The addition of $\beta I$ to $B_k$ can be seen as adding positive curvature $\beta$ in all directions. In the left panel we see a saddle with eigenvalues $\{-1, 2\}$; in the center panel, we have added $\beta = 0.75$ and we can see that the convex direction has become more convex, and the concave direction less concave; finally, in the right panel we have added $\beta = 1.5$ so that both directions in the “modified saddle” now are convex.

We note that the absolute distances between the eigenvalues are the same in all three cases, but that the condition numbers change — in this example they are 2, 11, and 7.
Figure: We see the condition number $\kappa(B_k + \beta I)$ has a spike (to $\infty$) at $\beta = 1$, so somehow we need to make sure that we make the modified matrix positive definite “enough” that the resulting condition number does not destroy our computational scheme. (More details on conditioning and its effects in Math 543.)
In the dogleg as well as in the 2-D subspace minimization algorithms, we must compute the solution of a linear system, either $B_k^{-1}\nabla f(\bar{x}_k)$ or $(B_k + \beta I)^{-1}\nabla f(\bar{x}_k)$.

When $B_k$ is large, i.e. we have a large number of unknowns in our minimization problem, solving these linear systems may be quite costly. — In fact, this is probably where we are likely to spend most of our computational resources!

Much research has been done on this issue — instead of solving the linear system exactly, we will look at approximate solutions. However, we need approximate solutions which yield improvements to the Cauchy point when used inside our optimization algorithms.
Computing \( B_k^{-1} \nabla f(\overline{x}_k) \) or \( (B_k + \beta I)^{-1} \nabla f(\overline{x}_k) \)

Steinhaug’s Approach (NW\(^1\)st pp.75-77, NW\(^2\)nd pp.171-173) is an adaptation of such an approximate solution scheme (the conjugate gradient algorithm).

For now, we will sweep these issues under the rug. — Soon we will take a close look at conjugate gradient (CG) methods. Once we have added the CG “tool” to our bag-of-tricks we will re-visit both line-search and trust-region methods and integrate the CG-tools into the discussion.

The CG-integration will do two things for us — (i) it will stabilize the solution of the linear systems, and (ii) at the same time allow us to solve much larger problems, essentially taking us from “toy-problem” to large (realistic) problem size.
# Roadmap to Minimization (So Far)

## The Global Minimization Problem

Iterative Solution — From $x(k)$ to a better $x(k+1)$

### LINESEARCH

- **Search Direction $p(k)$**
  - Coordinate Descent
  - Steepest Descent
  - [Quasi-Newton]
  - Newton
- **Step Length**
  - Initial Step
  - Methods
    - Backtracking
    - Interpolation
      - Quadratic
      - Cubic
    - Sufficient Decrease Condition
      - Wolfe Conditions
      - Strong Wolfe Conditions
      - [Goldstein Conditions]
- **Convergence**
  - Global
  - Rates

### TRUSTREGION

- **Cauchy Point**
- **Steepest Descent in TR framework**
- **Dogleg**
  - 1D subproblem
- **2D Subspace Minimization**
  - 2D subproblem
  - Model Hessian can be indefinite
  - [Steihaug’s Method]
- **Nearly Exact Subproblem**
  - For low-dimensional problems
  - [Convergence]

---

Peter Blomgren, {blomgren.peter@gmail.com}
Problem NW-4.1

Let 

\[ f(\vec{x}) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2 \]

At \( \vec{x}_0 = (0, -1) \) draw the contour lines of the quadratic model assuming that \( B = \nabla^2 f(x) \). Draw the family of solutions* of trust-region subproblem as the radius varies from \( \Delta = 0 \) to \( \Delta = 2 \). Repeat at \( \vec{x}_1 = (0, 0.5) \).

* Compute the solutions “almost exactly” using some method we have discussed (or will discuss), or some *ad hoc* brute-force method. The question is really “find, as best as you can, the optimal path.”

**Note:** The homework is due in Peter’s mailbox in GMCS-411 or in Peter’s office GMCS-587 (slide under the door if I’m not there).
Iterative “Nearly Exact” Methods for “Small” Problems

The methods we have looked at: Cauchy Point, Dogleg, and 2-D Subspace Minimization do not try to solve the subproblem

$$\bar{p}_k = \arg\min_{\bar{p} \in T_k} m_k(\bar{p}) = \min_{\bar{p} \in T_k} f(\bar{x}_k) + \bar{p}^T \nabla f(\bar{x}_k) + \frac{1}{2} \bar{p}^T B_k \bar{p}$$

exactly. — They use some of the information in the model Hessian $B_k$, and guarantee global convergence to a stationary point at a relatively low cost.

For small problems, when the number of unknowns $n$ is not too large, it is sometimes worth the effort to solve the subproblem more accurately.

The cost (per iteration) of the method we describe here is about three factorizations of the (model) Hessian Matrix. The hope is that, by working harder in each step, significantly fewer iterations will be necessary.

Peter Blomgren, ⟨blomgren.peter@gmail.com⟩
The basis of the nearly exact solution of the subproblem is:

— A good characterization of the exact solution.
— Application of Newton’s method in 1-D.

Essentially we are looking for a solution of

\[(B_k + \lambda I)\bar{p} = -\nabla f(\bar{x}_k),\quad \text{(The Convexified Problem)}\]

for an appropriate \(\lambda \geq 0\).
Trust Region Methods
Iterative “Nearly Exact” Solution to the Subproblem

Iterative “Nearly Exact” Methods: Key Result

Theorem

The vector $\bar{p}^*$ is a global solution of the trust-region problem

$$
\bar{p}^* = \arg \min_{\|\bar{p}\| \leq \Delta_k} \left[ f(\bar{x}_k) + \bar{p}^T \nabla f(\bar{x}_k) + \frac{1}{2} \bar{p}^T B_k \bar{p} \right]
$$

if and only if $\bar{p}^*$ is feasible, and there is a scalar $\lambda \geq 0$ such that the following conditions are satisfied:

1. $$(B_k + \lambda I)\bar{p}^* = -\nabla f(\bar{x}_k),$$
2. $$\lambda (\Delta_k - \|\bar{p}^*\|) = 0,$$
3. $$(B_k + \lambda I)$$ is positive semi-definite.

For the proof, see NW$^{1st}$ pp.84–87, or NW$^{2nd}$ pp.89–91.
Iterative “Nearly Exact” Methods: Key Result — Discussion

The Conditions

1. \((B_k + \lambda I)\bar{p}^* = -\nabla f(\bar{x}_k)\)
2. \(\lambda(\Delta_k - \|\bar{p}^*\|) = 0\)
3. \((B_k + \lambda I)\) is positive semi-definite

Condition 1

implies that \(\lambda\bar{p}^* = -B_k\bar{p}^* - \nabla f(\bar{x}_k) = -\nabla m_k(\bar{p}^*),\) i.e. \(\bar{p}^*\) is collinear with the negative gradient of the model \(m_k,\) and hence normal to its contour lines. (Steepest Descent for the model, in \(\bar{p}^*\).)

Condition 2

says that either \(\lambda = 0\) (in which case \(B_k\) is positive semidefinite and \(B_k\bar{p}^* = -\nabla f(\bar{x}_k)\)), or \(\bar{p}^*\) lies on the boundary of the trust region.
Iterative “Nearly Exact” Methods: The Scheme

Based on the theorem, we construct an algorithm for finding the solution of the local subproblem:

Case #1: \( \lambda = 0 \) works, i.e. \( B_k \) is positive semi-definite, and
\[
\|\bar{p}^*\| = \|B_k^\dagger \nabla f(\bar{x}_k)\| \leq \Delta_k.
\]

[pseudo-inverse, \( B_k^\dagger \): next slide].

Case #2: Otherwise, we define a function
\[
\bar{p}(\lambda) = -(B_k + \lambda I)^{-1} \nabla f(\bar{x}_k)
\]
which is defined for \( \lambda \) sufficiently large that \((B_k + \lambda I)\) is positive semi-definite. Now, we seek the value of \( \lambda \) so that \( \|\bar{p}(\lambda)\| = \Delta_k \).

Notice: this is a 1-D root-finding problem in \( \lambda \).
Definition (The Pseudo-Inverse)

Given \( A \in \mathbb{C}^{m\times n} \), the pseudo-inverse is the unique matrix \( A^\dagger \in \mathbb{C}^{n\times m} \) that satisfies the Moore-Penrose conditions:

(i) \( AA^\dagger A = A \)

(ii) \( A^\dagger AA^\dagger = A^\dagger \)

(iii) \( (AA^\dagger)^* = AA^\dagger \)

(iv) \( (A^\dagger A)^* = A^\dagger A \)

Computationally, the Pseudo-Inverse is typically implemented using the \( QR \)-decomposition \( [A^\dagger \sim R^{-1}Q^*] \), or the Singular Value Decomposition (SVD) \( [A^\dagger \sim V\Sigma^{-1}U^*] \) (see Math 543).
We take a closer look at the properties of $\|\bar{p}(\lambda)\|$ —

Since $B_k$ is symmetric, it is **unitarily diagonalizable** (Math 543), i.e. there exists an orthonormal matrix $Q$ and a diagonal matrix $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ where $\lambda_i$ are the eigenvalues of $B_k$ listed in increasing order, such that $B_k = Q\Lambda Q^T$. Now,

$$B_k + \lambda I = Q(\Lambda + \lambda I)Q^T,$$

and we can write

$$\bar{p}(\lambda) = -Q(\Lambda + \lambda I)^{-1}Q^T \nabla f(\bar{x}_k) = -\sum_{j=1}^{n} \frac{\bar{q}_j^T \nabla f(\bar{x}_k)}{\lambda_j + \lambda} \bar{q}_j,$$

where $\bar{q}_j$ is the $j$th column of $Q$. 
The "definition" of a small problem is a problem for which we can compute the unitary diagonalization $B_k = Q\Lambda Q^T$ in reasonable amount of time.

If $B_k$ is of size $n \times n$, then the number of operations required to find the decomposition is $\sim O(n^3)$.

The details of the decomposition (which is a 2-phase algorithm, first converting $B_k$ into tridiagonal form $T_k$, and then iteratively diagonalizing $T_k$ using the QR-algorithm) is given in Math 543.
Now, we can write

$$\|\bar{p}(\lambda)\|^2 = \sum_{j=1}^{n} \frac{(\bar{q}_j^T \nabla f(\bar{x}_k))^2}{(\lambda_j + \lambda)^2}.$$  

Note that the numerators are all constants independent of $\lambda$. If $\lambda > -\lambda_1$, then $(\lambda_j + \lambda) > 0$, $\forall j$ and $\|\bar{p}(\lambda)\|$ is a decreasing function of $\lambda$ in the interval $(-\lambda_1, \infty)$. We have the following

$$\lim_{\lambda \to \infty} \|\bar{p}(\lambda)\| = 0,$$

and when $\bar{q}_j^T \nabla f(\bar{x}_k) \neq 0$

$$\lim_{\lambda \to -\lambda_j} \|\bar{p}(\lambda)\| = \infty.$$
**Illustration:** $\|\bar{p}(\lambda)\|$

**Figure:** Illustration, here $\lambda_1 = -3$, $\lambda_2 = -1$, and all other $\lambda_j > 0$. We see the blow-ups of $\|\bar{p}(\lambda)\|$ at $\lambda = 1$, and $\lambda = 3$. The dash-dotted line illustrates the trust-region bound $\Delta_k = 0.05$. 
Case #2a: If $B_k$ is positive definite, but $\|B_k^{-1}\nabla f(\bar{x}_k)\| > \Delta_k$, there is a strictly positive $\lambda$ for which $\|\bar{p}(\lambda)\| = \Delta_k$, so we search for a $\lambda \in (0, \infty)$.

Case #2b: If $B_k$ is indefinite, and $\bar{q}_1^T \nabla f(\bar{x}_k) \neq 0$ we must search in the interval $(-\lambda_1, \infty)$...

Case #2c: If $B_k$ is indefinite, and $\bar{q}_1^T \nabla f(\bar{x}_k) = 0 \Rightarrow "The Hard Case."

We would like to apply Newton’s method to

$$\Phi_1(\lambda) = \|\bar{p}(\lambda)\| - \Delta_k = 0,$$

but since $\|\bar{p}(\lambda)\|$ (and therefore $\Phi_1(\lambda)$) blows up around $\lambda_1$, Newton’s method may be slow and unreliable.
Instead, we apply Newton’s method to the root finding problem

\[
\Phi_2(\lambda) = \frac{1}{\Delta_k} - \frac{1}{\|\bar{p}(\lambda)\|} = 0.
\]

\(\Phi_2(\lambda)\) behaves nicely (linearly!) around \(\lambda_1\):

\[
\Phi_2(\lambda) \approx \frac{1}{\Delta_k} - \frac{\lambda + \lambda_1}{C}, \quad C > 0, \quad \lambda \approx \lambda_1
\]

Hence, root-finding with Newton’s method will work.
Illustration: Searching for $1/\|\bar{p}(\lambda)\| = 1/\Delta_k$

**Figure:** Illustration, here $\lambda_1 = -3$, $\lambda_2 = -1$, and all other $\lambda_j > 0$. We see that the blow-ups of $\|\bar{p}(\lambda)\|$ at $\lambda = 1$, and $\lambda = 3$ have been “converted” into zeros. The dashed line illustrates the converted trust-region bound $1/\Delta_k = 1/0.05 = 20.0$. 
We are now ready to write down the Newton iteration

\[ \lambda^{(n+1)} = \lambda^{(n)} - \frac{\Phi_2(\lambda^{(n)})}{\Phi'_2(\lambda^{(n)})} \]

**Algorithm: Exact Trust Region**

1. Given \( \lambda^{(0)} \), \( \Delta_k > 0 \)
2. LOOP until convergence
3. TRY to factor \( B + \lambda^{(n)} I = L^T L \)
4. IF factorization failed
5. increase \( \lambda^{(n)} \) and RETRY (3)
6. ELSE
7. Solve \( L^T L \bar{p}_n = -\nabla f(\bar{x}_k), \ L^T \bar{q}_n = \bar{p}_n \)
8. Update

\[ \lambda^{(n+1)} = \lambda^{(n)} + \left[ \frac{\| \bar{p}_n \|}{\| \bar{q}_n \|} \right]^2 \left[ \frac{\| \bar{p}_n \| - \Delta}{\Delta} \right] \]

9. ENDIF, ENDLOOP (\( n = n + 1 \))
Algorithm: Cholesky Factorization

for i = 1, 2, ..., n
    \( L_{ii} = \sqrt{A_{ii}} \), this **Fails** if \( A_{ii} < 0 \), (really if \( A_{ii} < \epsilon_* \))
    for j = i+1, i+2, ..., n
        \( L_{ji} = A_{ji} / L_{ii} \)
        for k = i+1, i+2, ..., j
            \( A_{jk} = A_{jk} - L_{ji} L_{ki} \)
        end-for-k
    end-for-j
end-for-i

(*) André-Louis Cholesky (15 October 1875 – 31 August 1918). He served the French military as artillery officer, and was killed in battle a few months before the end of World War I.
This algorithm will work nicely for Case #2a and Case #2b, unfortunately the story does not end there. (In Case #2a we can apply Newton’s Method directly to the un-converted $\Phi_1(\lambda)$, as long as we enforce $\lambda > 0$ in the Newton iteration).

Recall that in Case #2b we assumed $\bar{q}_1^T \nabla f(\bar{x}_k) \neq 0$. If that is not true there may not be a value $\lambda \in (-\lambda_1, \infty)$ for which $\|\bar{p}(\lambda)\| = \Delta_k$. (See illustration on next slide.)

Since root finding on $(-\lambda_1, \infty)$ fails, and the theorem guarantees that $\lambda \in [-\lambda_1, \infty)$, we must have $\lambda = -\lambda_1$.

Finding $\bar{p}$ in this case requires a little more work...
Illustration: The Hard Case

**Figure:** Illustration, here \( \lambda_1 = -3, \lambda_2 = -1, \) and all other \( \lambda_j > 0. \) Here \( \bar{q}_1^T \nabla f(\bar{x}_k) = 0, \) and we end up with the hard case where \( \|\bar{p}(\lambda)\| = \Delta \) is not solvable in the region \( \lambda \in (-\lambda_1, \infty). \) (We see the blow-up at \(-\lambda_2,\) and the location of \(-\lambda_1\) has been marked with a vertical dash-dotted line.)
The Hard Case: Computing \( \tilde{p} \)

The matrix \((B - \lambda_1 I)\) is singular, therefore there exists a vector \( \tilde{z} \) such that \( \|\tilde{z}\| = 1 \) and \((B - \lambda_1 I)\tilde{z} = 0\).

Hence \( \tilde{z} \) is the eigenvector of \( B \) corresponding to \( \lambda_1 \), by the orthogonality of \( Q \), we have \( \tilde{q}_j^T \tilde{z} = 0 \) for \( \lambda_j \neq \lambda_1 \). If we set

\[
\tilde{p}(\tau, \lambda) = \tau \tilde{z} + \sum_{\{j: \lambda_j \neq \lambda_1\}} \frac{\tilde{q}_j^T \nabla f(\bar{x}_k)}{\lambda_j + \lambda} \tilde{q}_j,
\]

for any \( \tau \in \mathbb{R} \), then

\[
\|\tilde{p}(\tau, \lambda)\|^2 = \tau^2 + \sum_{\{j: \lambda_j \neq \lambda_1\}} \frac{(\tilde{q}_j^T \nabla f(\bar{x}_k))^2}{(\lambda_j + \lambda)^2}.
\]

Since \( \|\tilde{p}(0, \lambda_1)\| < \Delta_k \) and \( \|\tilde{p}(\tau, \lambda_1)\| \) is monotonically increasing in \( \tau \), we can find the unique \( \tau^* \) for which \( \|\tilde{p}(\tau^*, \lambda_1)\| = \Delta_k \).
2D subspace minimization, 8
  indefinite Hessian, 12
Cholesky factorization, 38
minimization road map, 22
trust region subproblem
  iterative solution, 24

Reference(s):

