Numerical Optimization

Lecture Notes #15

Practical Newton Methods — Trust-Region Newton 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

Hessian Modifications

We discussed strategies for modifying the Hessian in order to make it positive definite:

If we use the Frobenius matrix norm, the smallest change is of the type **“change negative eigenvalues to small positive ones:”**

$$B = A + \Delta A, \text{ where } \Delta A = Q\,\mathrm{diag}(\tau_i)\,Q^T, \; \tau_i = \begin{cases} 0 & \lambda_i \geq \delta \\ \delta - \lambda_i & \lambda_i < \delta. \end{cases}$$

If, on the other hand, we use the Euclidean norm the smallest change is a multiple of the identity matrix, *i.e.* **“shift the eigenvalue spectrum, so all eigenvalues are positive:”**

$$B = A + \Delta A, \quad \text{where} \quad \Delta A = \tau I, \quad \tau = \max(0, \delta - \lambda_{\min}(A)).$$

Recall: The Trust Region Algorithm

Algorithm: Trust Region

Set $k = 1$, $\widehat{\Delta} > 0$, $\Delta_0 \in (0, \widehat{\Delta})$, and $\eta \in [0, \frac{1}{4}]$
While optimality condition not satisfied
  Get $\bar{\mathbf{p}}_k$ (approximate solution, Today's Discussion)
  Evaluate ρ_k
  if $\rho_k < \frac{1}{4}$
    $\Delta_{k+1} = \frac{1}{4}\Delta_k$
  else
    if $\rho_k > \frac{3}{4}$ and $\|\bar{\mathbf{p}}_k\| = \Delta_k$
      $\Delta_{k+1} = \min(2\Delta_k, \widehat{\Delta})$
    else
      $\Delta_{k+1} = \Delta_k$
    endif
  endif
  if $\rho_k > \eta$
    $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k + \bar{\mathbf{p}}_k$
  else
    $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k$
  endif
  $k = k + 1$
End-While

Trust-Region Methods: B_k not Positive Definite is OK(?)

The Trust-region framework **does not require that the model Hessian is positive definite.**

It is possible to use the exact Hessian $B_k = \nabla^2 f(\bar{\mathbf{x}}_k)$ directly and find the search direction $\bar{\mathbf{p}}_k$ by solving the trust-region subproblem

$$\min_{\bar{\mathbf{p}} \in \mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \nabla f(\bar{\mathbf{x}}_k)^T \bar{\mathbf{p}} + \frac{1}{2} \bar{\mathbf{p}}^T B_k \bar{\mathbf{p}}, \quad \|\bar{\mathbf{p}}\| \leq \Delta_k.$$

Some of the **techniques** we discussed, *e.g.* dogleg, **require that B_k is positive definite.**

Review + Add Hessian Modifications and/or CG-solvers

We have seen quite few ideas floating around, lets review what we have seen in the context of our methods:

(i) the dogleg method,
(ii) 2D-subspace minimization,
(iii) nearly exact solution, and
(iv) the CG method.

The goal is to improve the methods and remove as many restrictions as possible.

Newton-Dogleg "Newton" $\Rightarrow B_k = \nabla^2 f(x_k)$ 1 of 2

When B_k is positive definite the dogleg method — minimizing the model over the **dogleg path**

$$\tilde{\bar{p}}(\tau) = \begin{cases} \tau \bar{\mathbf{p}}_k^U & 0 \leq \tau \leq 1 \\ \bar{\mathbf{p}}_k^U + (\tau - 1)(\bar{\mathbf{p}}_k^B - \bar{\mathbf{p}}_k^U) & 1 \leq \tau \leq 2 \end{cases}$$

where

$$\underbrace{\bar{\mathbf{p}}_k^B = -B_k^{-1} \nabla f(\bar{\mathbf{x}}_k),}_{\text{The Full Step}} \quad \underbrace{\bar{\mathbf{p}}_k^U = -\frac{\nabla f(\bar{\mathbf{x}}_k)^T \nabla f(\bar{\mathbf{x}}_k)}{\nabla f(\bar{\mathbf{x}}_k)^T B_k \nabla f(\bar{\mathbf{x}}_k)} \nabla f(\bar{\mathbf{x}}_k)}_{\text{The unconstrained minimum of the quadratic model along the steepest descent direction}}$$

gives good approximate solutions to the trust-region subproblems which can be computed efficiently.

However, when B_k is not positive definite we cannot safely compute $\bar{\mathbf{p}}_k^B$, further the denominator $\nabla f(\bar{\mathbf{x}}_k)^T B_k \nabla f(\bar{\mathbf{x}}_k)$ could be zero...

Newton-Dogleg Convexification 2 of 2

In order to make the dogleg method work for non-positive definite B_ks we can use the **Hessian modification** from last time to replace

$$B_k \rightarrow \underbrace{(B_k + E_k)}_{\textbf{Pos.Def}}$$

and use this matrix in the dogleg solution.

There is a price to pay. When the matrix B_k is modified, the importance of different directions are potentially changed in different ways, and the 1D-path (approximating the optimal path) is moved in nD-space. This may negatively impact the benefits of the trust-region approach.

Modifications of the type $E_k = \tau I$ behave somewhat more predictably than modifications of the type $E_k = \text{diag}(\tau_1, \tau_2, \ldots, \tau_n)$.

Usage of the dogleg method for non-convex problems is somewhat dicey, and even though it may work it is not the preferred method.

Newton-2D-Subspace-Minimization

In much the same way we modified the dogleg method, we can adapt the 2D-subspace minimization subproblem to work in the case of indefinite B_k

$$\min_{\bar{\mathbf{p}}\in\mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \nabla f(\bar{\mathbf{x}}_k)^T\bar{\mathbf{p}} + \frac{1}{2}\bar{\mathbf{p}}^T B_k \bar{\mathbf{p}}, \quad \|\bar{\mathbf{p}}\| \le \Delta_k, \quad \bar{\mathbf{p}} \in \text{span}(\nabla f(\bar{\mathbf{x}}_k), \bar{\mathbf{p}}^B)$$

can be applied when B_k is positive definite, and with a modified $\tilde{B}_k = (B_k + E_k)$ which is positive definite in the case when B_k is not positive definite:

$$\min_{\bar{\mathbf{p}}\in\mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \nabla f(\bar{\mathbf{x}}_k)^T\bar{\mathbf{p}} + \frac{1}{2}\bar{\mathbf{p}}^T \tilde{B}_k \bar{\mathbf{p}}, \quad \|\bar{\mathbf{p}}\| \le \Delta_k, \quad \bar{\mathbf{p}} \in \text{span}(\nabla f(\bar{\mathbf{x}}_k), \bar{\mathbf{p}}^{\tilde{B}})$$

The 2D-subspace method is only marginally more "expensive" (per iteration) than the dogleg approach; it is however more robust with respect to Hessian modification.

Iterative "Nearly Exact" Solution of the Trust-Region Subproblem

Recall the characterization of the exact solution, from lecture #9:

Theorem

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

$$\min_{\|\bar{\mathbf{p}}\|\le\Delta_k} f(\bar{\mathbf{x}}_k) + \bar{\mathbf{p}}^T\nabla f(\bar{\mathbf{x}}_k) + \frac{1}{2}\bar{\mathbf{p}}^T B_k \bar{\mathbf{p}}$$

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

$$\begin{array}{rrcl} 1. & (B_k + \lambda I)\bar{\mathbf{p}}^* & = & -\nabla f(\bar{\mathbf{x}}_k) \\ 2. & \lambda(\Delta_k - \|\bar{\mathbf{p}}^*\|) & = & 0 \\ 3. & (B_k + \lambda I) & & \textit{is positive semi-definite} \end{array}$$

This approach is already using the Hessian modification in the "Euclidian" form $E_k = \lambda I$, good for "small problems."

Trust-Region Newton-CG

The trust-region subproblem

$$\min_{\bar{\mathbf{p}}\in\mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \nabla f(\bar{\mathbf{x}}_k)^T\bar{\mathbf{p}} + \frac{1}{2}\bar{\mathbf{p}}^T B_k \bar{\mathbf{p}}, \quad \|\bar{\mathbf{p}}\| \le \Delta_k,$$

can be solved using the [Preconditioned] Conjugate Gradient ([P]CG) method, with two additional termination criteria (one of which we have seen already).

For each subproblem we must solve

$$B_k \bar{\mathbf{p}}_k = -\nabla f(\bar{\mathbf{x}}_k).$$

We apply CG with the following stopping criteria

(standard) The system has been solved to desired accuracy.
(previous) Negative curvature encountered.
(new) Size of the approximate solution exceeds the trust-region radius.

Trust-Region Newton-CG

Steihaug's Method

In the case of *negative curvature* we follow the direction to the boundary of the trust region; we get **Steihaug's Method**

Algorithm: CG-Steihaug

```
Given ε > 0; set p̄_0 = 0, r̄_0 = ∇f(x̄_k), d̄_0 = −r̄_0
if( ||r̄_0|| < ε ) return(p̄_0)
while( TRUE )
  if( d̄_j^T B d̄_j ≤ 0 )  % Negative Curvature
    Find τ ≥ 0 such that p̄ = p̄_j + τ d̄_j satisfies ||p̄|| = Δ
    return(p̄)
  endif
  α_j = r̄_j^T r̄_j / d̄_j^T B d̄_j,  p̄_{j+1} = p̄_j + α_j d̄_j
  if( ||p̄_{j+1}|| ≥ Δ )  % Step outside trust region
    Find τ ≥ 0 such that p̄ = p̄_j + τ d̄_j satisfies ||p̄|| = Δ
    return(p̄)
  endif
  r̄_{j+1} = r̄_j + α_j B d̄_j
  if( ||r̄_{j+1}|| ≤ ε||r̄_0|| ) return(p̄_{j+1})
  β_{j+1} = r̄_{j+1}^T r̄_{j+1} / r̄_j^T r̄_j,  d̄_{j+1} = −r̄_{j+1} + β_{j+1} d̄_j
end-while
```

Trust-Region Newton-CG

When we get close to the optimum, the trust-region constraint becomes **inactive** (the model becomes a good approximation of the objective, and the radius of the trust-region grows).

At this juncture, we need to pay particular attention to how the ϵ in CG-Steihaug is selected. It should be given by the forcing sequence $\{\eta_k\}$ which gives us quadratic convergence, *i.e.* $\epsilon \sim \|\nabla f(\bar{\mathbf{x}}_k)\|$.

Good properties of TR-Newton-CG: **Globally convergent**, the first step in the $-\nabla f(\bar{\mathbf{x}}_k)$ direction identifies the Cauchy point, the subsequent steps improve on $\bar{\mathbf{p}}^c$. **No matrix factorizations** are necessary.

Advantages over LS-Newton-CG: Step lengths are **controlled** by the trust region. Directions of negative curvature are **explored**.

Trust-Region Newton-CG

Room for Improvement: Any direction of negative curvature is accepted — the accepted direction can give an insignificant reduction in the model.

There is an extension of CG known as **Lanczos method**, and it is possible to build a TR-Newton-Lanczos algorithm which does not terminate when encountering the *first* direction of curvature, but continues to search for a direction of *sufficient negative curvature*.

TR-Newton-Lanczos is more robust, but comes at a cost of a more expensive solution of the subproblem.

We leave the discussion of the *Lanczos algorithm* to `Math 643` (to be offered in ∼Spring 2049).

Trust-Region Newton-PCG(M)

As we have seen in other (very similar) settings, adding preconditioning to the CG-solver can cut the number of iterations quite drastically.

It would seem like a good (and natural) idea to add preconditioning to the Trust-Region Newton-CG scheme.

We have to be a little careful... For the standard CG-Steihaug, the following is true

Theorem

The sequence of vectors generated by CG-Steihaug satisfies

$$0 = \|\bar{\mathbf{p}}_0\|_2 < \|\bar{\mathbf{p}}_1\|_2 < \cdots < \|\bar{\mathbf{p}}_j\|_2 < \|\bar{\mathbf{p}}_{j+1}\|_2 < \ldots \|\bar{\mathbf{p}}\|_2 \leq \Delta$$

This does not hold for preconditioned PCG(M)-Steihaug. This means that the sequence can leave the trust region, and then come back!

Trust-Region Newton-PCG(M)

It is possible to define a weighed norm in which the PCG(M) iterates grow monotonically — this weighted norm depends on the preconditioner.

If we express the preconditioning of B_k in terms of a non-singular matrix D, which guarantees that the eigenvalues of $D^{-T}B_kD^{-1}$ have a favorable distribution, when the subproblem takes the form

$$\min_{\bar{\mathbf{p}}\in\mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \nabla f(\bar{\mathbf{x}}_k)^T\bar{\mathbf{p}} + \frac{1}{2}\bar{\mathbf{p}}^TB_k\bar{\mathbf{p}}, \quad \|\mathbf{D}\bar{\mathbf{p}}\| \leq \Delta_k$$

if we formally make the change of variables $\widehat{\mathbf{p}} = D\bar{\mathbf{p}}$, and set $\widehat{\mathbf{g}}_k = D^{-T}\nabla f(\bar{\mathbf{x}}_k)$, $\widehat{B}_k = D^{-T}B_kD^{-1}$, the subproblem transform into

$$\min_{\bar{\mathbf{p}}\in\mathbb{R}^n} f(\bar{\mathbf{x}}_k) + \widehat{\mathbf{g}}_k^T\widehat{\mathbf{p}} + \frac{1}{2}\widehat{\mathbf{p}}^T\widehat{B}_k\widehat{\mathbf{p}}, \quad \|\widehat{\mathbf{p}}\| \leq \Delta_k$$

to which we can apply CG-Steihaug.

Trust-Region Newton-PCG(M)

As usual, we never make this change of variables explicitly. Instead the CG-Steihaug algorithm is modified so that the wherever we have a multiplication by D^{-1} or D^{-T} we solve the appropriate linear system.

Note, if $D^{-T}B_kD^{-1} = I$ the preconditioning is perfect. Usually

$$D^{-T}B_kD^{-1} = I + E$$

and if we multiply by D^T from the left and D from the right we see

$$B_k = \underbrace{D^TD}_{M} + \underbrace{D^TED}_{R}$$

So that $M \approx B_k$, and R captures the "inexactness" of the preconditioning.

Trust-Region Newton-PCG(M)

We can get a good general-purpose preconditioner by using a variant of the Cholesky factorization, $LL^T = B_k$.

We have discussed two ideas in connection with the Cholesky factorization — last time, we talked about how to **modify** it to get an approximate factorization of an indefinite matrix, *i.e.*

$$[L, L^T] = \begin{cases} \textbf{choldecomp}(B_k) & = & \textbf{cholesky}(B_k + \textbf{diag}(\tau_1, \tau_2, \dots, \tau_n)) \\ \textbf{modelhess}(B_k) & = & \textbf{cholesky}(B_k + \lambda I) \end{cases}$$

We have also (in general terms) talked about the **incomplete Cholesky factorization**, which preserves the sparsity pattern of B_k by not allowing fill-ins.

Putting the two together we get something like the algorithm on the next slide... (do not implement this one!)

Trust-Region Newton-PCG(M)

Algorithm: Modified Incomplete Cholesky Factorization, LDL^T-form

Given $\delta > 0$, $\beta > 0$
for j = 1:n
  $c_{jj} = a_{jj} - \sum_{s=1}^{j-1} d_s l_{js}^2$
  $\theta_j = \max_{j<i\leq n} |c_{ij}|$
  $\mathbf{d_j} = \max\left(|\mathbf{c_{jj}}|,\ \delta,\ \left[\frac{\theta_j}{\beta}\right]^2\right)$
  for i = (j+1):n
    if($a_{ij} \neq 0$) % Only allow $l_{ij} \neq 0$ if $a_{ij} \neq 0$
      $c_{ij} = a_{ij} - \sum_{s=1}^{j-1} d_s l_{is} l_{js}$
      $l_{ij} = c_{ij}/d_j$
    else
      $l_{ij} = c_{ij} = 0$
    endif
  endfor(i)
endfor(j)

Comments

We have looked at **Newton methods** (with quadratic convergence, if and only if we implement and solve all the subproblems in the right way) for both the linesearch and trust-region approach, and have developed quite a powerful framework of algorithms that are suitable and quite stable for large problems.

Are we done??? — *Not quite!*

We several topics left on the menu, including:

1. **Estimation of derivatives** — how to proceed if the gradient and/or the Hessian is not available in analytic form.
2. **Quasi-Newton methods** — how to proceed if the Hessian is not available (too expensive).
3. Application to **Nonlinear Least Squares** problems.
4. Application to **Nonlinear Equations**. — If we can minimize, we can also solve $\bar{\mathbf{F}}(\bar{\mathbf{x}}) = \bar{\mathbf{0}}$.

Index

Reference(s):

- Steihaug, Trond. *The conjugate gradient method and trust regions in large scale optimization.* SIAM Journal on Numerical Analysis 20, no.3 (1983): 626–637.
- Nicholas IM Gould, Stefano Lucidi, Massimo Roma, and Philippe L. Toint. *Solving the trust-region subproblem using the Lanczos method.* SIAM Journal on Optimization 9, no. 2 (1999): 504–525.