Newton’s Method for Constrained Norm Minimization and Its Application to Weighted Graph Problems

Mahmoud El Chamie and Giovanni Neglia
INRIA - Sophia Antipolis
France

American Control Conference 2014 (Portland, OR, USA, June 4-6)
Acknowledgement

Konstantin Avrachenkov
k.avrachenkov@inria.fr

Giovanni Neglia
giovanni.neglia@inria.fr
Talk Outline

• The General Optimization Framework
• Contribution of the Paper
• Newton’s Method for the General Framework
• Newton’s Method for Weighted Graph Optimization (a case study)
• Performance Evaluation
The General Optimization Framework

- \( \text{Argmin}_X f(X) = E(X, \hat{y}) + \gamma R(X) \)
  
  - loss function
  
  - regularization

- \( X \in \mathbb{R}^{n_1, n_2} \)
  
  - \( \gamma \) : scalar for the trade-off between the two terms

- For underdetermined systems (more variables than the equations): the \textit{minimal norm interpolation problem} [A. Argyriou et al., 2010],
  
  \[ \text{Argmin}_X R(X) \]
  
  subject to \( L(X) = \hat{y} \)
  
  - \( L(X) = A \text{vext}(X) \) is a linear function (\( A \) has \( r \times n_1 n_2 \) dimensions, \( r \text{ rank of } A \))
Schatten-$p$ Norm

\[ R(X) = \| X \|_{\sigma p} = \left( \sum_{i} \sigma_i^p \right)^{1/p} \]

- $\sigma_i$ is the $i$-th singular value of $X$
- $p=1$ is the Nuclear norm
- $p=2$ is the Frobenius norm
- $p=\infty$ is the Spectral norm

- Differentiable for even $p=2q$, \[ \| X \|_{\sigma p}^p = \text{Tr}((XX^T)^q) \]
- The problem is then convex
Contribution

minimize $h(X) = \text{Tr}((XX^T)^q)$

subject to $A \text{vect}(X) = \hat{y}$

• Transform into unconstrained optimization
• Find close form solutions for the gradient and the Hessian
• Apply Newton’s method
• Show that the optimization as applications in graph optimization problems
  – Multi-agents consensus problems
  – Hessian can be sparse and exact step-size can be easily calculated
• Study the convergence time of different optimization methods for this problem
Weighted Graph Optimization: Weight Selection in Consensus Protocols

- Consensus algorithm
  - Given a network $G=(V, E)$
  - Each node has a local variable $x_i(k)$, where $k$ is the iteration number and $x_i(0)$ are initial values
  - Each node performs **weighted sum** of its value and its neighbors’ values:
    \[
    x_i(k + 1) = w_{ii} \times x_i(k) + \sum_{\text{neighbors } j} w_{ij} \times x_j(k)
    \]
    - In matrix form:
      \[
      x(k) = W^k x(0)
      \]
Weighted Graph Optimization: Weight Selection in Consensus Protocols

- **Consensus algorithm**
  
  \[ x_i(k+1) = w_{ii} \times x_i(k) + \sum_{\text{neighbors } j} w_{ij} \times x_j(k) \]

  - For some (easy to satisfy) condition on the weights, the algorithm is guaranteed to converge to the average of initial values
  - An approximation to the best weights (that achieve fastest convergence independent of initial values) is

  \[
  \text{minimize} \quad Tr(W^p) \\
  \text{subject to } W = W^T, \quad W1_n = 1_n, \quad W \in \mathcal{C}_G,
  \]

  - \( p \) identifies the error from optimal weights [M. El Chamie et al., 2012]
Weighted Graph Optimization: Weight Selection in Consensus Protocols

- **Consensus algorithm**

  \[ x_i(k + 1) = w_{ii} \times x_i(k) + \sum_{\text{neighbors } j} w_{ij} \times x_j(k) \]

  - For some (easy to satisfy) condition on the weights, the algorithm is guaranteed to converge to the average of initial values.

  - An approximation to the best weights (that achieve fastest convergence independent of initial values) is

    \[
    \begin{align*}
    \text{minimize} & \quad Tr(W^p) & \text{Schatten p-norm of } W \\
    \text{subject to} & \quad W = W^T, \; W1_n = 1_n, \; W \in C_G, & \text{Linear constraints} \\
    \end{align*}
    \]

    - \( p \) identifies the error from optimal weights [M. El Chamie et al., 2012]
Newton's Method

• Form the unconstrained problem:

\[ f(w) = Tr(W^p) \big|_{W=I_n - Q\text{diag}(w)Q^T} \]

- \( w \): \( m \) by 1 weight vector (each link is given a variable)
- \( Q \): \( n \) by \( m \) incidence matrix of the graph \( G \)

• Form the gradient \( g = \nabla_w f \in \mathbb{R}^m \), \( l \leftrightarrow (a,b) \):

\[
(\nabla_w f)_l = \frac{\partial f}{\partial w_l} = p((W^{p-1})_{a,b} + (W^{p-1})_{b,a} - (W^{p-1})_{a,a} - (W^{p-1})_{b,b})
\]
Newton’s Method

• Form the unconstrained problem:

\[ f(w) = Tr(W^p) \mid W = I_n - Q \text{diag}(w)Q^T \]

- \( w \): \( m \)-by-1 weight vector (each link is given a variable)
- \( Q \): \( n \)-by-\( m \) incidence matrix of the graph \( G \)

• Form the Hessian \( H = \nabla^2_w f \in \mathbb{R}^{m \times m} : \)

\[
(\nabla^2_w f)_{l,k} = p \sum_{z=0}^{p-2} \psi(z)\psi(p - 2 - z),
\]

\[
\psi(z) = (W^z)_{a,c} + (W^z)_{b,d} - (W^z)_{a,d} - (W^z)_{b,c}.
\]
Newton's Direction

- We calculated the gradient $g$, and the Hessian $H$
  - Notice that $H$ is positive semi-definite since $f$ is convex
- Newton's Direction $\Delta w$ is simply the solution of:

$$H \Delta w = g$$
Line Search

• For choosing a stepsize that guarantees sufficient decrease in the function
  \[ \phi(t) = f(w - t\Delta w), \quad t > 0 \]

• Exact line search is usually complex and other simple (non-optimal) choice are usually used
  - Pure Newton \( t=1 \) (for all iterations)
  - Backtracking line search starts from \( t=1 \), and multiplicatively decrease \( t \) till sufficient decrease in the function

• However, the special structure of our problem allows for a simple exact line search
Exact Line Search

- For choosing a stepsize that guarantees sufficient decrease in the function

$$\phi(t) = f(w - t\Delta w) = h(W + tU)$$

where $$U = Q \text{diag}(\Delta w)Q^T$$ and let $$Y = W + tU$$

- Since $$h$$ is convex, then $$\phi(t)$$ is also convex, and the first and second derivatives:

$$\phi'(t) = p \text{Tr}(Y^{p-1}U)$$ and $$\phi''(t) = p \text{Tr}(\sum_r Y^{p-2-r}UYrU)$$

- With Newton-Raphson (exact stepsize)

$$t_n \leftarrow t_{n-1} - \frac{\phi'(t_{n-1})}{\phi''(t_{n-1})}$$
Summary for Newton's Method

- **Step 0**: Initial start $W^{(0)} = I_n$, precision $\varepsilon$, $k = 0$
- **Step 1**: Calculate gradient $g = \nabla_w f \in \mathbb{R}^m$
- **Step 2**: Calculate Hessian $H = (\nabla_w^2 f + \gamma I_m) \in \mathbb{R}^{m \times m}$
- **Step 3**: Calculate Newton’s direction $\Delta w^{(k)} = H^{-1} g$

Stopping Condition: $\| \Delta w^{(k)} \| < \varepsilon$

- **Step 4**: Use exact line search for stepsize $t^{(k)}$
- **Step 5**: Update the weight matrix

$$W^{(k+1)} = W^{(k)} + t^{(k)} Q \text{diag}(\Delta w^{(k)}) Q^T$$

- **Step 6**: $k = k + 1$ and go back to Step 1
Closed form solution for $p=2$

\[
\begin{align*}
\text{minimize} & \quad h(W) = Tr(W^2) = \sum_{i,j} w_{ij}^2 \\
\text{subject to} & \quad W = W^T, W1_n = 1_n, W \in \mathcal{C}_G.
\end{align*}
\]

\begin{itemize}
\item Since objective function is quadratic, pure newton converges in **one iteration**, starting from any feasible initial value
\item Let $W^{(0)}=I_n$, then $g = -4 \times 1_m$ and $H = 2(2I_m + Q^TQ)$
\item Substitute in equation
\end{itemize}

\[
W^{(k+1)} = W^{(k)} + t^{(k)} Q \text{diag}(\Delta w^{(k)}) Q^T
\]

\[
= I_n - Q \text{diag}((I_m + 0.5Q^TQ)^{-1}1_m)Q^T
\]
Closed form solution for p=2

\[
\begin{align*}
\text{minimize } & h(W) = Tr(W^2) = \sum_{i,j} w_{ij}^2 \\
\text{subject to } & W = W^T, W1_n = 1_n, W \in \mathcal{C}_G.
\end{align*}
\]

- Since objective function is quadratic, pure newton converges in one iteration, starting from any feasible initial value.
- Let \(W^{(0)} = I_n\), then \(g = -4 \times 1_m\) and \(H = 2(2I_m + Q^TQ)\).
- Substitute in equation

\[
W^{(k+1)} = W^{(k)} + t^{(k)}Q \text{diag}(\Delta w^{(k)})Q^T
\]

\[
= I_n - Q \text{diag}((I_m + 0.5Q^TQ)^{-1}1_m)Q^T
\]

Closed Form
Simulation

<table>
<thead>
<tr>
<th>$T_{conv}$ (number of iterations)</th>
<th>$ER(n = 100, Pr = 0.07)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p = 2$</td>
</tr>
<tr>
<td>Exact-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Pure-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Exact-GD</td>
<td>72.3</td>
</tr>
<tr>
<td>Exact-Nesterov</td>
<td>130.2</td>
</tr>
<tr>
<td>BT-GD or BT-Nesterov</td>
<td>&gt; 5000</td>
</tr>
</tbody>
</table>

**TABLE I**

Convergence time using different optimization methods for problem (12).

- GD: Decent Gradient
- BT-: Backtracking line search
- Stopping Condition $\|g\| < 10^{-10}$
Simulation

<table>
<thead>
<tr>
<th>$T_{conv}$ (number of iterations)</th>
<th>$ER(n = 100, Pr = 0.07)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p = 2$</td>
</tr>
<tr>
<td>Exact-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Pure-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Exact-GD</td>
<td>72.3</td>
</tr>
<tr>
<td>Exact-Nesterov</td>
<td>130.2</td>
</tr>
<tr>
<td>BT-GD or BT-Nesterov</td>
<td>&gt; 5000</td>
</tr>
</tbody>
</table>

**TABLE I**

Convergence time using different optimization methods for problem (12).

- **Observation 1** (intuitive)

On average, the number of iterations to converge of Newton much less than first order methods.

- GD: Decent Gradient
- BT-: Backtracking line search
- Stopping Condition $\|g\| < 10^{-10}$
Simulation

<table>
<thead>
<tr>
<th>$T_{conv}$ (number of iterations)</th>
<th>$ER(n = 100, Pr = 0.07)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p = 2$</td>
</tr>
<tr>
<td>Exact-Newton Pure-Newton</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Exact-GD</td>
<td>72.3</td>
</tr>
<tr>
<td>Exact-Nesterov</td>
<td>130.2</td>
</tr>
<tr>
<td>BT-GD or BT-Nesterov</td>
<td>$&gt; 5000$</td>
</tr>
</tbody>
</table>

**TABLE I**

Convergence time using different optimization methods for problem (12).

- **Observation 2 (less intuitive)**
  
  Newton’s method is less sensitive to stepsize (exact stepsize does not change much convergence)

In gradient methods, highly sensitive to stepsize

- GD: Decent Gradient
- BT- : Backtracking line search
- Stopping Condition $\|g\|<10^{-10}$
Simulation

<table>
<thead>
<tr>
<th>$T_{conv}$ (number of iterations)</th>
<th>$ER(n = 100, Pr = 0.07)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p = 2$</td>
</tr>
<tr>
<td>Exact-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Pure-Newton</td>
<td>1</td>
</tr>
<tr>
<td>Exact-GD</td>
<td>72.3</td>
</tr>
<tr>
<td>Exact-Nesterov</td>
<td>130.2</td>
</tr>
<tr>
<td>BT-GD or BT-Nesterov</td>
<td>$&gt; 5000$</td>
</tr>
</tbody>
</table>

**TABLE I**

Convergence time using different optimization methods for problem (12).

- **Observation 3 (not intuitive)**
  
  Decent Gradient method is faster than Nesterov

- **GD: Decent Gradient**
- **BT- : Backtracking line search**
- **Stopping Condition** $\|g\| < 10^{-10}$
Conclusion

• Newton’s method for Schatten $p$-norm minimization
• Its application to weighted graph problems
  – Sparse Hessian
  – Fast convergence
  – Robustness to stepsize
• Future work
  – General values of $p$ (not just for even values)
  – Other graph optimization
Conclusion

• Newton’s method for Schatten p-norm minimization
• Its application to weighted graph problems
  – Sparse Hessian
  – Fast convergence
  – Robustness to stepsize
• Future work
  – General values of p (not just for even values)
  – Other graph optimization

Thank you! Questions?
Possible Approaches

Argmin_x R(X)
subject to L(X) = \hat{y}

• First Order Methods
  – Gradient Method
  – Fast Gradient Method (Nestrov)
  – Drawbacks
    • Slow convergence
    • Step size selection (for unbounded gradient)

• Second Order Methods (for differentiable R(.) )
  – Newton’s Method
  – Drawbacks
    • Difficult to have closed form solutions
    • Complexity (forming and inverting the Hessian)
Newton's Method for Schatten-p Norm

\[
\begin{align*}
\text{minimize} & \quad h(X) = Tr \left( (XX^T)^q \right) \\
\text{subject to} & \quad [ I_r \quad B ] P \ vect(X) = \hat{y} \\
1. & \quad \text{Substitute the constraints in the objective function,} \\
x = [ 0^{n_1n_2-r,r} \quad I_{n_1n_2-r} ] P \ vect(X), \;
X = vect^{-1} \left( P^{-1} \left[ \begin{array}{c} \hat{y} - Bx \\ x \end{array} \right] \right), \\
2. & \quad \text{Reformulate into an unconstrained problem:} \\
& \quad \text{minimize } f(x), \quad f(x) = Tr \left( (XX^T)^q \right) \\
3. & \quad \text{Solve for the gradient } g = \nabla_x f \quad \text{and Hessian } H = \nabla^2_x f \\
& \quad (\text{closed form formulas are given in the paper, we only give the results here for a case study}) \\
4. & \quad \text{Find the newton's direction, iteratively update variable using stepsize} (\text{details in the upcoming case study})
\end{align*}
\]
D-regular graphs

- D-regular graphs are graphs where all vertices have the same degree D (cycles, complete graphs, ...)
- Optimal values for p=2 is

\[ w_l = \frac{1}{1 + D} \quad \forall l = 1, \ldots, m. \]

- Which gives the same weights as well known weight selection heuristics as Metropolis weight selection or the maximum degree.