Modern Optimization Techniques

2. Unconstrained Optimization / 2.4. Quasi-Newton Methods

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

Mon.	1.11.	(1)	0. Overview
			1. Theory
Mon.	8.11.	(2)	1. Convex Sets and Functions
			2. Unconstrained Optimization
Mon.	15.11.	(3)	2.1 Gradient Descent
Mon.	22.11.	(4)	2.2 Stochastic Gradient Descent
Mon.	29.11.	(5)	2.3 Newton's Method
Mon.	6.12.	(6)	2.4 Quasi-Newton Methods
Mon.	13.12.	(7)	2.5 Subgradient Methods
Mon.	20.12.	(8)	2.6 Coordinate Descent
		—	— *Christmas Break* —
			3. Equality Constrained Optimization
Mon.	3.1.	(9)	3.1 Duality
Mon.	10.1.	(10)	3.2 Methods
			4. Inequality Constrained Optimization
Mon.	17.1.	(11)	4.1 Primal Methods
Mon.	24.1.	(12)	4.2 Barrier and Penalty Methods
Mon.	31.1.	(13)	4.3 Cutting Plane Methods
Mon.	7.2.	(14)	Q & A

Outline

1. Excursion: Inverting Matrices
2. The Idea of Quasi-Newton Methods
3. BFGS and L-BFGS

Outline

1. Excursion: Inverting Matrices

2. The Idea of Quasi-Newton Methods

3. BFGS and L-BFGS

Matrix Inversion

Given a square matrix $A \in \mathbb{R}^{N \times N}$,
its **inverse** A^{-1} is a matrix such that:

$$AA^{-1} = \mathbf{I}$$

where

- $\mathbf{I}$ is the identity matrix.
- if such a matrix A^{-1} exists, A is called **regular** (aka **invertible**).
- if no such matrix A^{-1} exists, A is called **singular** (aka **non-invertible**).

Matrix Inversion — Easy Cases

1. **small matrices:**
   - for $A \in \mathbb{R}^{2\times 2}$ the inverse can be computed analytically:

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \quad A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

   - slightly more complex closed formula for $A \in \mathbb{R}^{3\times 3}$

2. **orthogonal matrices:**
   - $A \in \mathbb{R}^{N\times N}$ is **orthogonal** if $A^T A = \mathbf{I}$
   - thus $A^{-1} = A^T$
   - example:

$$A := \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$$

Matrix Inversion — Easy Cases

3. **diagonal matrices:**
   - $A \in \mathbb{R}^{N \times N}$ is **diagonal** if $A_{n,m} = 0$ for all $n \neq m$
   - thus $A = \text{diag}(a_1, a_2, \ldots, a_N)$ with

$$\text{diag}(a_1, \ldots, a_N) := \begin{pmatrix} a_1 & 0 & \ldots & 0 \\ 0 & a_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & a_N \end{pmatrix}$$

   - Q: What is its inverse A^{-1}?

Matrix Inversion — Easy Cases

3. **diagonal matrices:**
   - $A \in \mathbb{R}^{N\times N}$ is **diagonal** if $A_{n,m} = 0$ for all $n \neq m$
   - thus $A = \text{diag}(a_1, a_2, \ldots, a_N)$ with

$$\text{diag}(a_1, \ldots, a_N) := \begin{pmatrix} a_1 & 0 & \ldots & 0 \\ 0 & a_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & a_N \end{pmatrix}$$

   - $A^{-1} = \text{diag}(\frac{1}{a_1}, \frac{1}{a_2}, \ldots, \frac{1}{a_N})$

4. **upper (or lower) triangular matrices:**
   - $A \in \mathbb{R}^{N\times N}$ is **upper triangular** if $A_{n,m} = 0$ for all $n > m$
   - A^{-1} can be computed in $O(N^2)$ by back substitution.

General Matrix Inversion

Generally, inverting a matrix $A \in \mathbb{R}^{N\times N}$ is equivalent to solving a linear system of equations with n different right sides:

$$AA^{-1} = I \iff Ax^n = e^n, \quad e^n := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{matrix} \\ \\ \\ \leftarrow n\text{-th position} \\ \\ \\ \\ \end{matrix}, \quad n = 1, \ldots,$$

via $A^{-1} = (x^1, x^2, \ldots, x^N)$

If an inverse is used only once to compute $x := A^{-1}b$ for a vector $b \in \mathbb{R}^N$, it usually is faster to solve the linear system of equations $Ax = b$ instead.

General Matrix Inversion / Complexity

Inverting matrices and solving systems of linear equations can be accomplished two ways:

1. algebraic algorithms ("direct algorithms")
   - like Gaussian elimination, LU decomposition, QR decomposition
   - complexity generally $O(N^3)$
   - there exist specialized matrix inversion algorithms with lower costs
     - Strassen algorithm $O(N^{2.807})$
     - Coppersmith–Winograd algorithm $O(N^{2.376})$
     - but they are impractical and not used in implementations
2. optimization algorithms ("iterative algorithms")
   - Gauss-Seidel, Gradient-descent type of algorithms

Inverse of a Rank-One Update

Lemma (Inverse of a Rank-One Update – Sherman-Morrison formula)
For $A \in \mathbb{R}^{N \times N}$ invertible and $a, b \in \mathbb{R}^N$: *(with $b^T A^{-1} a \neq -1$)*

$$(A + ab^T)^{-1} = A^{-1} - \frac{A^{-1} a b^T A^{-1}}{1 + b^T A^{-1} a}$$

Meaning:

- the inverse of a rank-one update can be computed fast
  - in $O(N^2)$ instead of in $O(N^3)$
  - if the inverse of the original matrix is available

Inverse of a Rank-One Update / Proof

Show that the right side has the property of the inverse:

$$
\begin{aligned}
(A + ab^T)(A^{-1} - \frac{A^{-1}ab^TA^{-1}}{1 + b^TA^{-1}a})
&= I + ab^TA^{-1} - \frac{ab^TA^{-1} + ab^TA^{-1}ab^TA^{-1}}{1 + b^TA^{-1}a})\\
&= I + ab^TA^{-1} - \frac{a(1 + b^TA^{-1}a)b^TA^{-1}}{1 + b^TA^{-1}a}\\
&= I + ab^TA^{-1} - ab^TA^{-1} = I
\end{aligned}
$$

Outline

Underlying Idea

- Approximate the Hessian with a matrix H that is fast to invert.

$$H \approx \nabla^2 f(x)$$

  - Use a low-rank update

$$H^{(0)} := I$$

$$H^{\text{next}} = H + \sum_{k=1}^{K} a_k b_k^T$$

    - fast to invert using K-times inverses of rank-one updates

$$(H^{-1})^{(0)} = I$$

$$(H^{-1})^{\text{next}} = H^{-1} + \dots$$

- Compute the next direction using the inverse of the Hessian approximation:

$$\Delta x = -H^{-1} \nabla f(x)$$

Properties of the Hessian $\nabla^2 f(x)$

- it fulfills the **secant condition**

$$H(y - x) = \nabla f(y) - \nabla f(x)$$

  approximately:

$$\nabla^2 f(x)(y - x) \approx \nabla f(y) - \nabla f(x) \quad \text{for } y \approx x$$

  - due to first order Taylor expansion of ∇f:

$$\nabla f(y) \approx \nabla f(x) + \nabla^2 f(x)(y - x)$$

  - equivalent to:
    the second order approximation of f by ∇f and H around x
    has gradient $\nabla f(y)$ at y

- it is symmetric

- it is positive semidefinite
- it is positive definite
  - for a strongly convex objective function

Properties of the Hessian $\nabla^2 f(x)$

- H fulfills the secant condition $\Leftrightarrow$ the second order approximation of f by ∇f and H around x has gradient $\nabla f(y)$ at y

proof:

$$
\begin{aligned}
F(y) :=& f(x) + \nabla f(x)^T (y-x) + \frac{1}{2}(y-x)^T H (y-x) \\
\nabla F(y) =& \nabla f(x) + H(y-x) = \nabla f(y)
\end{aligned}
$$

Hessian Approximations

Idea: search for a matrix H that

- has some of the properties of the Hessian and
- is fast to compute
  - e.g., by a low-rank update from the previous approximation:

$$H^{(0)} := I$$

$$H^{\text{next}} = H + \sum_{k=1}^{K} a_k b_k^T, \quad a_k, b_k \in \mathbb{R}^N$$

Symmetric Rank-One Update

Lemma (Symmetric Rank-One Update)

There exists exactly one low-rank update of H such that

i) H^{next} *fulfils the secant condition*

$$H^{next}s = g, \quad s := x^{next} - x, \quad g := \nabla f(x^{next}) - \nabla f(x)$$

ii) H^{next} *is symmetric and*

iii) H^{next} *is a rank-one update:*

$$a_1 = b_1 := \frac{g - Hs}{((g - Hs)^T s)^{\frac{1}{2}}}$$

$$H^{next} = H + \frac{(g - Hs)(g - Hs)^T}{(g - Hs)^T s}$$

Symmetric Rank-One Update / Proof

If H and H^{next} are symmetric, then $a_1 b_1^T$ must be also symmetric.

$$a_1 b_1^T \stackrel{!}{=} (a_1 b_1^T)^T = b_1 a_1^T \quad | \cdot a_1$$

$$a_1 b_1^T a_1 \stackrel{!}{=} b_1 a_1^T a_1 \quad \rightsquigarrow b_1 = \beta a_1, \quad \beta \in \mathbb{R}, \beta \neq 0$$

$$H^{\text{next}} \underset{iii}{=} H + \beta a_1 a_1^T$$

$$H^{\text{next}} s \underset{i}{=} g$$

$$\beta a_1 a_1^T s = g - Hs \quad \rightsquigarrow \quad a_1 = \gamma(g - Hs), \quad \gamma \in \mathbb{R}$$

$$\beta\gamma(g - Hs)\gamma(g - Hs)^T s = g - Hs$$

$$\beta\gamma^2(g - Hs)^T s = 1$$

$$\beta = 1, \quad \gamma = ((g - Hs)^T s)^{-\frac{1}{2}}, \quad a_1 = \frac{g - Hs}{((g - Hs)^T s)^{\frac{1}{2}}}$$

Symmetric Rank-One Update / Inverse

Lemma (Symmetric Rank-One Update / Inverse)

The inverse H^{-1} of the approximate Hessian in the symmetric rank-one update is

$$(H^{-1})^{next} = H^{-1} + \frac{(s - H^{-1}g)(s - H^{-1}g)^T}{(s - H^{-1}g)^T g}$$

This means, to compute H^{-1}, we

- **do not** have to i) update H and then ii) compute its inverse ($O(N^3)$), but
- simply can update H^{-1} using the formula above ($O(N^2)$).

Symmetric Rank-One Update / Inverse / Proof

Apply Morrison-Sherman to the rank-one update of the Hessian approximation:

$$
\begin{aligned}
(H^{-1})^{\text{next}} &= H^{-1} - \frac{H^{-1}(g-Hs)(g-Hs)^T H^{-1}}{(g-Hs)^T s\,(1 + \frac{(g-Hs)^T H^{-1}(g-Hs)}{(g-Hs)^T s})} \\
&= H^{-1} - \frac{(H^{-1}g-s)(H^{-1}g-s)^T}{\underbrace{(g-Hs)^T s + (g-Hs)^T H^{-1}(g-Hs)}_{\substack{=(g-Hs)^T(s+H^{-1}g-s)\\ =(g-Hs)^T H^{-1} g\\ =(H^{-1}g-s)^T g}}} \\
&= H^{-1} + \frac{(s-H^{-1}g)(s-H^{-1}g)^T}{(s-H^{-1}g)^T g}
\end{aligned}
$$

Note: Remember: $(A+ab^T)^{-1} = A^{-1} - \frac{A^{-1}ab^T A^{-1}}{1+b^T A^{-1} a}$.

Newton's Method (Review)

min-newton$(f, \nabla f, \nabla^2 f, x^{(0)}, \mu, \epsilon, K)$:
    for $k := 1, \ldots, K$:
        $\Delta x^{(k-1)} := -\nabla^2 f(x^{(k-1)})^{-1} \nabla f(x^{(k-1)})$
        if $-\nabla f(x^{(k-1)})^T \Delta x^{(k-1)} < \epsilon$:
            return $x^{(k-1)}$
        $\mu^{(k-1)} := \mu(f, x^{(k-1)}, \Delta x^{(k-1)})$
        $x^{(k)} := x^{(k-1)} + \mu^{(k-1)} \Delta x^{(k-1)}$
    return "not converged"

Q: How do we have to change the Newton algorithm to use a rank 1 update of the Hessian?

where

- f objective function
- ∇f, $\nabla^2 f$ gradient and Hessian of objective function f
- $x^{(0)}$ starting value
- μ step length controller
- ϵ convergence threshold for Newton's decrement
- K maximal number of iterations

Quasi-Newton Method / SR1

min-qnewton-sr1$(f, \nabla f, x^{(0)}, \mu, \epsilon, K)$:

  $A^{(0)} := I$

  for $k := 1, \ldots, K$:

    $\Delta x^{(k-1)} := -A^{(k-1)} \nabla f(x^{(k-1)})$

    if $-\nabla f(x^{(k-1)})^T \Delta x^{(k-1)} < \epsilon$:

      return $x^{(k-1)}$

    $\mu^{(k-1)} := \mu(f, x^{(k-1)}, \Delta x^{(k-1)})$

    $x^{(k)} := x^{(k-1)} + \mu^{(k-1)} \Delta x^{(k-1)}$

    $s^{(k)} := x^{(k)} - x^{(k-1)}$

    $g^{(k)} := \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$

    $A^{(k)} := A^{(k-1)} + \frac{(s^{(k)} - A^{(k-1)} g^{(k)})(s^{(k)} - A^{(k-1)} g^{(k)})^T}{(s^{(k)} - A^{(k-1)} g^{(k)})^T g^{(k)}}$

  return "not converged"

where

- $A = H^{-1}$ the inverse of the approximative Hessian

Outline

Positive Definite Hessian Approximations

- There is no rank-one update with guaranteed positive definite Hessian approximation H.
- There are many rank-two update schemes with guaranteed positive definite Hessian approximation H.
- Most widely used: BFGS
  - developed independently by Broyden, Fletcher, Goldfarb and Shanno in 1970

$$H^{\text{next}} := H - \frac{Hs(Hs)^T}{s^T Hs} + \frac{gg^T}{g^T s}$$

BFGS

Lemma (BFGS)

The BFGS update

$$H^{next} := H - \frac{Hs(Hs)^T}{s^T Hs} + \frac{gg^T}{g^T s}$$

i) *fulfils the secant condition,*

ii) *yields symmetric H and*

iii) *yields positive definite H, if $g^T s > 0$.*

The inverse H^{-1} of the approximate Hessian is

$$\begin{aligned}(H^{-1})^{next} &= H^{-1} + \frac{(s - H^{-1}g)s^T + s(s - H^{-1}g)^T}{s^T g} - \frac{(s - H^{-1}g)^T g}{(s^T g)^2} ss^T \\ &= (I - \frac{sg^T}{s^T g})H^{-1}(I - \frac{gs^T}{s^T g}) + \frac{ss^T}{s^T g}\end{aligned}$$

BFGS / Proof (1/3)

i) BFGS fulfils the secant condition:

$$\begin{aligned} H^{\text{next}} s &= Hs - \frac{Hs(Hs)^T s}{s^T Hs} + \frac{gg^T s}{g^T s} \\ &= Hs - Hs + g = g \end{aligned}$$

ii) BFGS yields symmetric H: obvious.

iii) BFGS yields positive definite H:
If H is positive definite, it can be represented $H = LL^T$ with a non-singular L (Cholesky decomposition).

$$\begin{aligned} H^{\text{next}} &= LWL^T \\ W &:= I - \frac{\tilde{s}\tilde{s}^T}{\tilde{s}^T\tilde{s}} + \frac{\tilde{g}\tilde{g}^T}{\tilde{g}^T\tilde{s}}, \quad \tilde{s} := L^T s, \quad \tilde{g} := L^{-1} g \end{aligned}$$

H^{next} will be pos.def., if W is.

BFGS / Proof (2/3)

for any $v \in \mathbb{R}^N$:

$$\begin{aligned} 0 \overset{?}{<} v^T W v &= v^T v - \frac{(v^T \tilde{s})^2}{\tilde{s}^T \tilde{s}} + \frac{(v^T \tilde{g})^2}{\tilde{g}^T \tilde{s}} \\ &= ||v||^2 - \frac{||v||^2 ||\tilde{s}||^2 \cos^2 \theta_1}{||\tilde{s}||^2} + \frac{(v^T \tilde{g})^2}{\tilde{g}^T \tilde{s}} \\ &= ||v||^2 (1 - \cos^2 \theta_1) + \frac{(v^T \tilde{g})^2}{\tilde{g}^T \tilde{s}} \\ &= ||v||^2 \sin^2 \theta_1 + \frac{(v^T \tilde{g})^2}{\tilde{g}^T \tilde{s}} \end{aligned}$$

$$\tilde{g}^T \tilde{s} = g^T s \underset{\text{assumption iii}}{>} 0$$

- if $v = \lambda \tilde{s}, \lambda \in \mathbb{R}, \lambda \neq 0$:
  - $\sin^2 \theta_1 = 0$, but
  - $(v^T \tilde{g})^2 = \lambda^2 (\tilde{s}^T \tilde{g})^2 > 0$
- if $v \neq \lambda \tilde{s}, \lambda \in \mathbb{R}, \lambda \neq 0$:
  - $\sin^2 \theta_1 > 0$

BFGS / Proof (3/3)

To derive the inverse of the approximate Hessian,
apply Morrison-Sherman twice.

Quasi-Newton Method / BFGS

min-qnewton-bfgs$(f, \nabla f, x^{(0)}, \mu, \epsilon, K)$:

  $A^{(0)} := I$

  for $k := 1, \ldots, K$:

    $\Delta x^{(k-1)} := -A^{(k-1)} \nabla f(x^{(k-1)})$

    if $-\nabla f(x^{(k-1)})^T \Delta x^{(k-1)} < \epsilon$:

      return $x^{(k-1)}$

    $\mu^{(k-1)} := \mu(f, x^{(k-1)}, \Delta x^{(k-1)})$

    $x^{(k)} := x^{(k-1)} + \mu^{(k-1)} \Delta x^{(k-1)}$

    $s^{(k)} := x^{(k)} - x^{(k-1)}$

    $g^{(k)} := \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$

    $A^{(k)} := A^{(k-1)} + \frac{(s^{(k)} - A^{(k-1)} g^{(k)})(s^{(k)})^T + s^{(k)}(s^{(k)} - A^{(k-1)} g^{(k)})^T}{(s^{(k)})^T g^{(k)}}$

    $\qquad - \frac{(s^{(k)} - A^{(k-1)} g^{(k)})^T g^{(k)}}{((s^{(k)})^T g^{(k)})^2} s^{(k)} (s^{(k)})^T$

  return "not converged"

where

- $A = H^{-1}$ the inverse of the approximative Hessian

Avoid Materialization of A

- In the previous form, BFGS still requires N^2 storage to materialize the inverse A of the approximate Hessian.

- For any vector $v \in \mathbb{R}^N$, images $A^{(K)}v$ can be computed from the recursive formula from vectors $g^{(k)}, s^{(k)}$ $(k = 1, \ldots, K)$

$$A^{(K+1)} = (I - \frac{s^{(K)}(g^{(K)})^T}{(s^{(K)})^T g^{(K)}})A^{(K)}(I - \frac{g^{(K)}(s^{(K)})^T}{(s^{(K)})^T g^{(K)}}) + \frac{s^{(K)}(s^{(K)})^T}{(s^{(K)})^T g^{(K)}}$$
$$= \left(\prod_{k=K}^{\downarrow 1}(I - \frac{s^{(k)}(g^{(k)})^T}{(s^{(k)})^T g^{(k)}})\right) A^{(0)} \left(\prod_{k=1}^{K}(I - \frac{g^{(k)}(s^{(k)})^T}{(s^{(k)})^T g^{(k)}})\right) + \ldots$$

Compute Image Av without Materialization of A

bfgs-image-iha$(v, (s^{(k)})_{k=1,\dots,K}, (g^{(k)})_{k=1,\dots,K}, (\rho^{(k)})_{k=1,\dots,K}, A^{(0)})$:

    $q := v$

    for $k := K, \dots, 1$:

        $\alpha_k := \rho^{(k)}(s^{(k)})^T q$

        $q := q - \alpha_k g^{(k)}$

    $r := A^{(0)} q$

    for $k := 1, \dots, K$:

        $\beta := \rho^{(k)}(g^{(k)})^T r$

        $r := r + s^{(k)}(\alpha_k - \beta)$

    return r

where

- $v \in \mathbb{R}^N$ vector whose image to compute, usually $\nabla f(x^{(k)})$
- $(s^{(k)})_{k=1,\dots,K}, (g^{(k)})_{k=1,\dots,K}$ as defined earlier
- $\rho^{(k)} := 1/(g^{(k)})^T s^{(k)}$
- $A^{(0)}$ initial inverse Hessian, e.g. I.

Quasi-Newton Method / BFGS w/o Materialization of A

min-qnewton-bfgs-nomat$(f, \nabla f, x^{(0)}, \mu, \epsilon, K)$:
  for $k := 1, \ldots, K$:
    $\Delta x^{(k-1)} := -\text{bfgs-image-iha}(\nabla f(x^{(k-1)}, s^{(1:k-1)},$
        $g^{(1:k-1)}, \rho^{(1:k-1)}, I)$
    if $-\nabla f(x^{(k-1)})^T \Delta x^{(k-1)} < \epsilon$:
      return $x^{(k-1)}$
    $\mu^{(k-1)} := \mu(f, x^{(k-1)}, \Delta x^{(k-1)})$
    $x^{(k)} := x^{(k-1)} + \mu^{(k-1)} \Delta x^{(k-1)}$
    $s^{(k)} := x^{(k)} - x^{(k-1)}$
    $g^{(k)} := \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$
    $\rho^{(k)} := 1/(g^k)^T s^{(k)}$
  return "not converged"

Avoid Materialization of A

- Storing all vectors $g^{(1:K)}, s^{(1:K)}$ requires $2KN$ storage, i.e. is only memory efficient for $K \ll N$.

- Instead of computing the inverse A of the approximate Hessian by all these vectors, we could
  - forget the older ones, i.e.,
  - just store and compute the $M \ll N$ most recent ones.

- This approach is called **Limited Memory BFGS** (L-BFGS)

Quasi-Newton Method / L-BFGS

min-qnewton-lbfgs$(f, \nabla f, x^{(0)}, \mu, \epsilon, K, M)$:
  for $k := 1, \dots, K$:
    $k_0 := \max\{1, k-1-M+1\}$
    $\Delta x^{(k-1)} := -\text{bfgs-image-iha}(\nabla f(x^{(k-1)}, s^{(k_0:k-1)},$
      $g^{(k_0:k-1)}, \rho^{(k_0:k-1)}, I)$
    if $-\nabla f(x^{(k-1)})^T \Delta x^{(k-1)} < \epsilon$:
      return $x^{(k-1)}$
    $\mu^{(k-1)} := \mu(f, x^{(k-1)}, \Delta x^{(k-1)})$
    $x^{(k)} := x^{(k-1)} + \mu^{(k-1)} \Delta x^{(k-1)}$
    $s^{(k)} := x^{(k)} - x^{(k-1)}$
    $g^{(k)} := \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$
    $\rho^{(k)} := 1/(g^k)^T s^{(k)}$
  return "not converged"

Implementations need to ensure that the old vectors $s^{(1:k_0-1)}, g^{(1:k_0-1)}$ do not consume any memory (i.e., are overwritten by the more recent ones).

Summary

- Rank One Updates $A + ab^T$ of a matrix A can be inverted fast (in $O(N^2)$; if an inverse of A is available; Sherman-Morrison formula).
- Quasi-Newton methods are Newton methods with approximated Hessian.
  - approximations should share properties of the Hessian
    - secant condition, symmetry, positive definiteness
  - maintain the inverse of the Hessian (not the Hessian itself)
- symmetric rank one update:
  - only one such rank one update (not even pos.def.)
- BFGS rank two update:
  - one out of many such rank two updates
  - pos.def.

Summary (2/2)

- Images of a vector under the inverse Hessian can be computed even without materializing the inverse Hessian:
  - compute the image recursively from the images under the rank one update steps
  - Limited Memory BFGS (L-BFGS)

Further Readings

- Quasi-Newton methods are not covered by Boyd and Vandenberghe, 2004

- BFGS:
  - Nocedal and Wright, 2006, ch. 6
  - Griva, Nash, and Sofer, 2009, ch. 12.3
    the update formulas for the inverse are in ch. 13.5.
  - Sun and Yuan, 2006, ch. 5.1

- L-BFGS:
  - Nocedal and Wright, 2006, ch. 7
  - Griva, Nash, and Sofer, 2009, ch. 13.5

References

- Boyd, Stephen and Lieven Vandenberghe (2004). *Convex Optimization*. Cambridge University Press.
- Griva, Igor, Stephen G. Nash, and Ariela Sofer (2009). *Linear and Nonlinear Optimization*. Society for Industrial and Applied Mathematics.
- Nocedal, Jorge and Stephen J. Wright (2006). *Numerical Optimization*. Springer Science+ Business Media.
- Sun, Wenyu and Ya-Xiang Yuan (2006). *Optimization Theory and Methods, Nonlinear Programming*. Springer.