Action Constrained Quasi-Newton Methods:

Robert Gower
(Joint work with Jacek Gondzio and Peter Richtarik)

ISMP 2015, the 22nd International Symposium on Mathematical Programming.

Action Constrained Quasi-Newton Methods: for Preconditioning Sequences of Systems

Robert Gower
(Joint work with Jacek Gondzio and Peter Richtarik)

ISMP 2015, the 22nd International Symposium on Mathematical Programming.

Table of Contents

The Problem
- Motivation
- The Assumptions
- Iterative Methods Sample Action

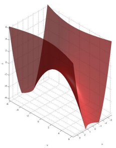
The Method
- Two Equivalent Formulations
- Adding a Symmetry Constraint

Combining with a linear solver
- Conjugate Gradients
- Parallel Limited memory LBFGS
- Numerics

Conclusions

Extensions
- Preconditioning Randomized methods

Newton's Method

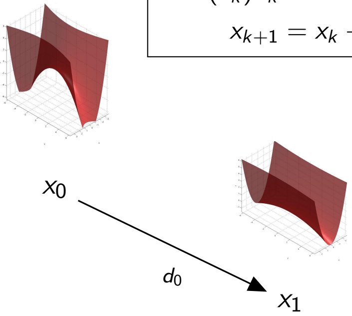


x_0

$$\boxed{\begin{aligned} \nabla^2 f(x_k) d_k &= -\nabla f(x_k). \\ x_{k+1} &= x_k + d_k \end{aligned}}$$

Newton's Method

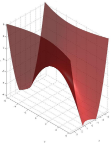
$$\boxed{\begin{aligned} \nabla^2 f(x_k) d_k &= -\nabla f(x_k). \\ x_{k+1} &= x_k + d_k \end{aligned}}$$

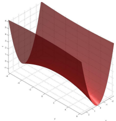


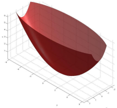
The Hessian "slowly" changes $\nabla^2 f(x_{k+1}) \approx \nabla^2 f(x_k)$.

Newton's Method

$$\boxed{\begin{aligned}\nabla^2 f(x_k) d_k &= -\nabla f(x_k).\\ x_{k+1} &= x_k + d_k\end{aligned}}$$





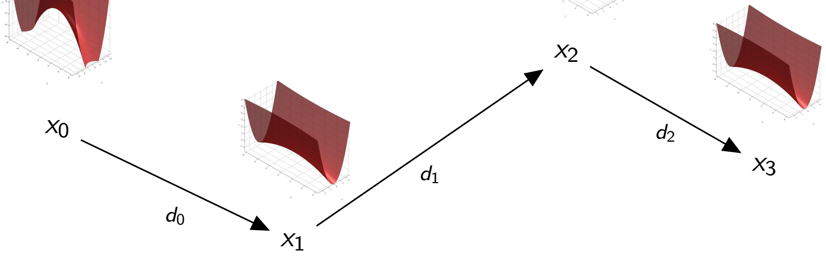


$$x_0 \xrightarrow{d_0} x_1 \xrightarrow{d_1} x_2$$

The Hessian "slowly" changes $\nabla^2 f(x_{k+1}) \approx \nabla^2 f(x_k)$.
Each Newton system is similar

Newton's Method

$$\boxed{\begin{aligned}\nabla^2 f(x_k) d_k &= -\nabla f(x_k).\\ x_{k+1} &= x_k + d_k\end{aligned}}$$



The Hessian "slowly" changes $\nabla^2 f(x_{k+1}) \approx \nabla^2 f(x_k)$.
Each Newton system is similar
Solving each system individually is a waste.

Solve Sequence of linear systems

Solve in x each

$$A_k x = b_k, \quad \text{for } k = 1, \dots,$$

where $A_k \in \mathbb{R}^{n \times n}$ nonsingular , $b \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$.

- Continuous $(A_k)_k$ changes "slowly" so that $\|A_{k+1}^{-1} - A_k^{-1}\|$ is "small"
- Cheap Action Only have access to sampled action $S_k \mapsto A_k S_k$ where $S_k \in \mathbb{R}^{n \times q_k}$

Solve Sequence of linear systems

Solve in x each

$$X_k A_k x = X_k b_k, \quad \text{for } k = 1, \ldots,$$

where $A_k \in \mathbb{R}^{n \times n}$ nonsingular , $b \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$.

- Continuous $(A_k)_k$ changes "slowly" so that $\|A_{k+1}^{-1} - A_k^{-1}\|$ is "small"
- Cheap Action Only have access to sampled action $S_k \mapsto A_k S_k$ where $S_k \in \mathbb{R}^{n \times q_k}$

Solution Strategy

Maintain a sequence of preconditioners $X_k \approx A_k^{-1} \in \mathbb{R}^{n \times n}$ where

$$X_{k+1} = X_k + \text{update}(S_{k+1}, A_{k+1} S_{k+1}),$$

where the update is low rank.

The Assumptions

The Continuous property can arise when $(A_k)_k$ is the result of evaluating a continuous matrix field, e.g, Newton type systems

$$f \in C^2(\mathbb{R}^n), \qquad \nabla^2 f(x_k) d_k = -\nabla f(x_k),$$

$$F \in C(\mathbb{R}^n, \mathbb{R}^n), \qquad DF(x_k) d_k = -F(x_k).$$

The Assumptions

The Continuous property can arise when $(A_k)_k$ is the result of evaluating a continuous matrix field, e.g, Newton type systems

$$f \in C^2(\mathbb{R}^n), \qquad \nabla^2 f(x_k) d_k = -\nabla f(x_k),$$

$$F \in C(\mathbb{R}^n, \mathbb{R}^n), \qquad DF(x_k) d_k = -F(x_k).$$

The Cheap action also in Newton systems. With Automatic differentiation

$$\left.\frac{d\nabla f(x_k + S_k y)}{dy}\right|_{y=0} = \nabla^2 f(x_k) S_k$$

$$\left.\frac{dF(x_k + S_k y)}{dy}\right|_{y=0} = DF(x_k) S_k.$$

Iterative Methods for $Ax = b$ Sample Action $S \mapsto AS$

Minimal Residual Methods

Starting from $x_0 = 0 \in \mathbb{R}^n$, the iterates are

$$x_{k+1} = \min_x \|Ax - b\|_2, \quad \text{subject to} \quad x = S\,y,$$

where $S \in \mathbb{R}^{n \times p}$ with $p \ll n$.

Iterative Methods for $Ax = b$ Sample Action $S \mapsto AS$

Minimal Residual Methods

Starting from $x_0 = 0 \in \mathbb{R}^n$, the iterates are

$$x_{k+1} = \min_x \|Ax - b\|_2, \quad \text{subject to} \quad x = S\,y,$$

where $S \in \mathbb{R}^{n \times p}$ with $p \ll n$. In other words

$$x_{k+1} = S\,y^*, \quad y^* = \arg\min_y \|ASy - b\|_2,$$

Which requires calculating AS

Iterative Methods for $Ax = b$ Sample Action $S \mapsto AS$

Minimal Residual Methods

Starting from $x_0 = 0 \in \mathbb{R}^n$, the iterates are

$$x_{k+1} = \min_x \|Ax - b\|_2, \quad \text{subject to} \quad x = S\,y,$$

where $S \in \mathbb{R}^{n \times p}$ with $p \ll n$. In other words

$$x_{k+1} = S\,y^*, \quad y^* = \arg\min_y \|ASy - b\|_2,$$

Which requires calculating AS

Conjugate Gradients Method

CG solves, starting from $x_0 = 0 \in \mathbb{R}^n$,

$$x_{k+1} = \min_x \frac{1}{2} x^T A x - x^T b, \quad \text{subject to} \quad x = S\,y.$$

Iterative Methods for $Ax = b$ Sample Action $S \mapsto AS$

Minimal Residual Methods

Starting from $x_0 = 0 \in \mathbb{R}^n$, the iterates are

$$x_{k+1} = \min_x \|Ax - b\|_2, \quad \text{subject to} \quad x = S\,y,$$

where $S \in \mathbb{R}^{n \times p}$ with $p \ll n$. In other words

$$x_{k+1} = S\,y^*, \quad y^* = \arg\min_y \|ASy - b\|_2,$$

Which requires calculating AS

Conjugate Gradients Method

CG solves, starting from $x_0 = 0 \in \mathbb{R}^n$,

$$x_{k+1} = \min_x \frac{1}{2}x^T Ax - x^T b, \quad \text{subject to} \quad x = S\,y.$$

Which also requires calculating AS

$$x_{k+1} = S\,y^*, \quad y^* = \arg\min_y 1/2 y^T S^T ASy - S^T y^T b.$$

Simplified Problem

Temporarily forget the sequence of systems.

Given $X_k \approx A^{-1}$ and a sampled action $S \mapsto AS$, calculate an updated estimate $X_{k+1} \approx A^{-1}$.

[0] For a Bayesian interpretation see: Hennig, P. (2015). Probabilistic interpretation of linear solvers. SIAM Journal on Optimization, 25(1), 234260.

Update method: Two equivalent formulations

- A positive definite matrix $W \succ 0 \in \mathbb{R}^{n\times n}$ that defines geometry $\langle X, Y\rangle_{F(W^{-1})} \stackrel{\text{def}}{=} \mathbf{Tr}\left(W^{-1}X^TW^{-1}Y\right)$ in $\mathbb{R}^{n\times n}$.

Update method: Two equivalent formulations

- A positive definite matrix $W \succ 0 \in \mathbb{R}^{n\times n}$ that defines geometry $\langle X, Y\rangle_{F(W^{-1})} \overset{\text{def}}{=} \mathbf{Tr}\left(W^{-1}X^T W^{-1}Y\right)$ in $\mathbb{R}^{n\times n}$.

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad XAS = S.$$

Same action as inverse $XAS = A^{-1}AS$.

Update method: Two equivalent formulations

- A positive definite matrix $W \succ 0 \in \mathbb{R}^{n\times n}$ that defines geometry $\langle X, Y\rangle_{F(W^{-1})} \stackrel{\text{def}}{=} \mathbf{Tr}\left(W^{-1}X^T W^{-1}Y\right)$ in $\mathbb{R}^{n\times n}$.

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad XAS = S.$$

Same action as inverse $XAS = A^{-1}AS$.

The Approximate Inverse viewpoint

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}, Y\in\mathbb{R}^{n\times q}} \|X - A^{-1}\|_{F(W^{-1})} \quad \text{subject to} \quad X = X_k + Y(WAS)^T.$$

Best approximation in an affine space.

Update method: Two equivalent formulations

- A positive definite matrix $W \succ 0 \in \mathbb{R}^{n\times n}$ that defines geometry $\langle X, Y\rangle_{F(W^{-1})} \overset{\text{def}}{=} \textbf{Tr}\left(W^{-1}X^T W^{-1} Y\right)$ in $\mathbb{R}^{n\times n}$.

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad XAS = S.$$

$$\Updownarrow$$

Same action as inverse $XAS = A^{-1}AS$.

$$X \in A^{-1} + L\text{-}\textbf{Null}\,(AS)$$

The Approximate Inverse viewpoint

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}, Y\in\mathbb{R}^{n\times q}} \|X - A^{-1}\|_{F(W^{-1})} \quad \text{subject to} \quad X = X_k + Y(WAS)^T.$$

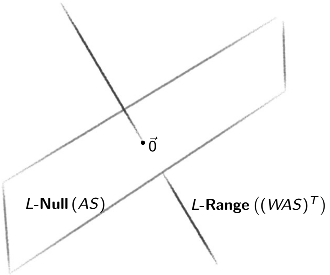
$$\Updownarrow$$

Best approximation in an affine space.

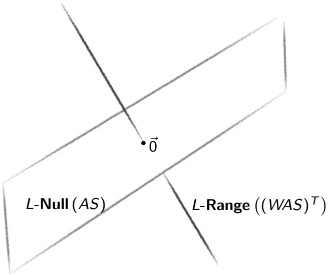
$$X \in X_k + L\text{-}\textbf{Range}\left((WAS)^T\right)$$

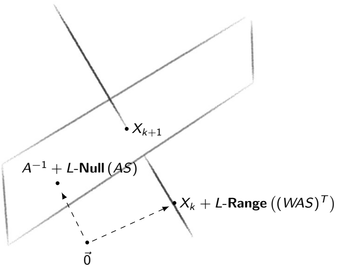
L-$\textbf{Null}\,(AS) \overset{\text{def}}{=} \{X \,|\, XAS = 0\}$ and L-$\textbf{Range}\left((WAS)^T\right) \overset{\text{def}}{=} \{X \,|\, X = Y(WAS)^T, Y \in \mathbb{R}^{n\times q}\}$

$$L\text{-}\mathbf{Null}\,(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$

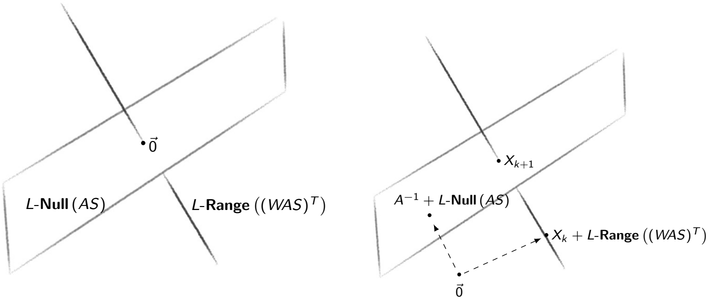


$$L\text{-}\mathbf{Null}\,(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$



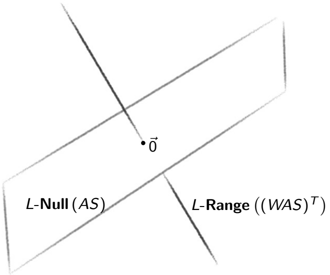


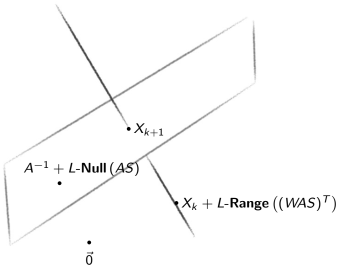
$$L\text{-}\mathbf{Null}(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$



$$X_{k+1} = \quad \left(A^{-1} + L\text{-}\mathbf{Null}(AS)\right) \quad \bigcap \quad \left(X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)\right)$$

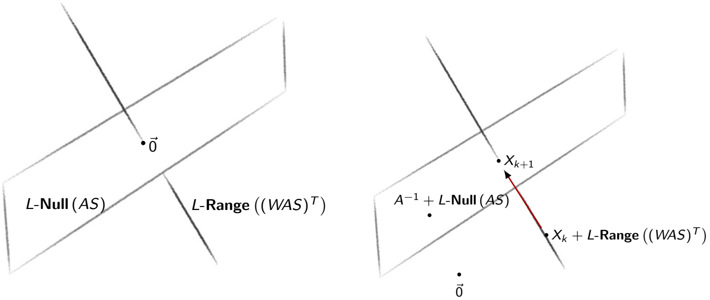
$$L\text{-}\mathbf{Null}\,(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$





$$X_{k+1} = \quad \left(A^{-1} + L\text{-}\mathbf{Null}\,(AS)\right) \quad \bigcap \quad \left(X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)\right)$$

$$L\text{-}\mathbf{Null}\,(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$

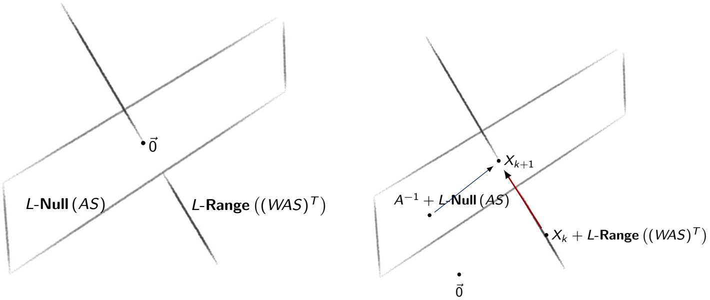


$$X_{k+1} = \quad \left(A^{-1} + L\text{-}\mathbf{Null}\,(AS)\right) \quad \bigcap \quad \left(X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)\right)$$

- Project X_k onto $A^{-1} + L\text{-}\mathbf{Null}\,(AS)$

$$X_{k+1} = \arg\min_X \|X - X_k\|^2_{F(W^{-1})} \quad \text{subject to} \quad XAS = S$$

$$L\text{-}\mathbf{Null}\,(AS) \oplus L\text{-}\mathbf{Range}\left((WAS)^T\right) = \mathbb{R}^{n\times n}$$



$$X_{k+1} = \left(A^{-1} + L\text{-}\mathbf{Null}\,(AS)\right) \bigcap \left(X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)\right)$$

I Project X_k onto $A^{-1} + L\text{-}\mathbf{Null}\,(AS)$

$$X_{k+1} = \arg\min_X \|X - X_k\|^2_{F(W^{-1})} \quad \text{subject to} \quad XAS = S$$

II Project A^{-1} onto $X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)$

$$X_{k+1} = \arg\min_X \|X - A^{-1}\|^2_{F(W^{-1})} \quad \text{subject to} \quad X \in X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)$$

The Explicit Update

The solution to

$$X_{k+1} = \arg\min_X \|X - A^{-1}\|^2_{F(W^{-1})} \quad \text{subject to} \quad X \in X_k + L\text{-}\mathbf{Range}\left((WAS)^T\right)$$

is given by the rank$-$(num. of columns in S) update

$$\begin{aligned} X_{k+1} &= X_k + (A^{-1} - X_k) Z W \\ &= X_k + (I - X_k A) S (S^T A^T W A S)^{-1} S^T A^T W, \end{aligned}$$

where

$$Z \overset{\text{def}}{=} AS(S^T A^T WAS)^{-1} S^T A^T.$$

Need to form and invert a small matrix.

Symmetric matrices

When A is symmetric, we add a symmetry constraint

$$X_{k+1} = \arg \min_{X \in \mathbb{R}^{n \times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad XAS = S, \quad X = X^T,$$

with solution[1]

$$X_{k+1} = A^{-1} + (I - WZ)(X_k - A^{-1})(I - ZW),$$

$$Z \stackrel{\text{def}}{=} AS(S^T A^T WAS)^{-1} S^T A^T.$$

- A rank-3(num. of columns in S) update
- Do not need A^{-1} to calculate

[1] Gower, R. M., & Gondzio, J. (2014). Action constrained quasi-Newton methods. arXiv:1412.8045v1, 134.

Choosing S and W

Action-Assimilate updates

$$\text{Not symmetric:} \quad X_{k+1} = A^{-1} + (X_k - A^{-1})(I - ZW)$$

$$\text{Symmetric:} \quad X_{k+1} = A^{-1} + (I - WZ)(X_k - A^{-1})(I - ZW),$$

$$\text{where} \quad Z = AS(S^T A^T WAS)^{-1} S^T A^T.$$

Choose S and W so that AS is W–*conjugate*

$$\boxed{S^T A^T WAS \quad = \quad \text{diagonal matrix}}$$

Because

- inverting diagonal matrix is more numerically stable
- convenient choice for W when used in together with Krylov method
- guarantees **Hereditary** property $\Rightarrow$ for local convergence
- Parallelizable limited memory updates!

Choosing S and W

Action-Assimilate updates

$$\text{Not symmetric:} \quad X_{k+1} = A^{-1} + (X_k - A^{-1})(I - ZW)$$

$$\text{Symmetric:} \quad X_{k+1} = A^{-1} + (I - WZ)(X_k - A^{-1})(I - ZW),$$

$$\text{where} \quad Z = AS(S^T A^T WAS)^{-1} S^T A^T.$$

Choose S and W so that AS is W*–conjugate*

$$\boxed{S^T A^T WAS \quad = \quad \text{diagonal matrix}}$$

Because

- inverting diagonal matrix is more numerically stable
- convenient choice for W when used in together with Krylov method
- guarantees **Hereditary** property $\Rightarrow$ for local convergence
- Parallelizable limited memory updates!

Theorem: Krylov Methods and Conjugacy[1]

Let $B \succ 0 \in \mathbb{R}^{n\times n}$ and $\langle x, y\rangle_B \overset{\text{def}}{=} x^T By$. Iterates of any Krylov method look like

$$x_{k+1} = \arg\min_x \|x_* - x\|_B^2 \quad \text{subject to} \quad x \in x_0 + \mathbf{Range}\,(S)\,,$$

where $Ax_* = b$. If iterates can be calculated by

$$x_{i+1} = x_i + s_i, \quad \text{for} \quad i = 0, \ldots, k,$$

where $[s_1, \ldots, s_k] = S$, then $S^T BS =$ diagonal matrix.

[1]Liesen, J., & Strako, Z. (2013). Krylov subspace methods : principles and analysis. Oxford: Oxford University Press

Theorem: Krylov Methods and Conjugacy[1]

Let $B \succ 0 \in \mathbb{R}^{n \times n}$ and $\langle x, y \rangle_B \overset{\text{def}}{=} x^T B y$. Iterates of any Krylov method look like

$$x_{k+1} = \arg\min_x \|x_* - x\|_B^2 \quad \text{subject to} \quad x \in x_0 + \mathbf{Range}\,(S)\,,$$

where $Ax_* = b$. If iterates can be calculated by

$$x_{i+1} = x_i + s_i, \quad \text{for} \quad i = 0, \ldots, k,$$

where $[s_1, \ldots, s_k] = S$, then $S^T B S =$ diagonal matrix.

Choose $AWA = B$ so that $S^T AWAS = S^T BS =$ diagonal matrix

[1] Liesen, J., & Strako, Z. (2013). Krylov subspace methods : principles and analysis. Oxford: Oxford University Press

Theorem: Krylov Methods and Conjugacy[1]

Let $B \succ 0 \in \mathbb{R}^{n\times n}$ and $\langle x, y\rangle_B \stackrel{\text{def}}{=} x^T By$. Iterates of any Krylov method look like

$$x_{k+1} = \arg\min_x \|x_* - x\|_B^2 \quad \text{subject to} \quad x \in x_0 + \mathbf{Range}\,(S)\,,$$

where $Ax_* = b$. If iterates can be calculated by

$$x_{i+1} = x_i + s_i, \quad \text{for} \quad i = 0, \ldots, k,$$

where $[s_1, \ldots, s_k] = S$, then $S^T BS =$ diagonal matrix.

Choose $AWA = B$ so that $S^T AWAS = S^T BS =$ diagonal matrix

name	B	W
Conjugate Gradients	A	A^{-1}
MINRES	$A^T A$	I
SYMMLQ	I	$(A^T A)^{-1}$.

[1] Liesen, J., & Strako, Z. (2013). Krylov subspace methods : principles and analysis. Oxford: Oxford University Press

Theorem: Krylov Methods and Conjugacy[1]

Let $B \succ 0 \in \mathbb{R}^{n \times n}$ and $\langle x, y \rangle_B \overset{\text{def}}{=} x^T B y$. Iterates of any Krylov method look like

$$x_{k+1} = \arg\min_x \|x_* - x\|_B^2 \quad \text{subject to} \quad x \in x_0 + \textbf{Range}\,(S),$$

where $Ax_* = b$. If iterates can be calculated by

$$x_{i+1} = x_i + s_i, \quad \text{for} \quad i = 0, \ldots, k,$$

where $[s_1, \ldots, s_k] = S$, then $S^T B S =$ diagonal matrix.

Choose $AWA = B$ so that $S^T AWAS = S^T BS =$ diagonal matrix

name	B	W
Conjugate Gradients	A	A^{-1}
MINRES	$A^T A$	I
SYMMLQ	I	$(A^T A)^{-1}$.

[1] Liesen, J., & Strako, Z. (2013). Krylov subspace methods : principles and analysis. Oxford: Oxford University Press

The Conjugate Gradients (CG) method

Calculates an approximate solution x_{k+1} to $Ax^* = b$ by

$$x_{k+1} = \arg\min_{x} \|x - x^*\|_A, \quad \text{subject to } x \in x_0 + \mathbf{Range}\,(S)$$

The columns of S are A conjugate, that is $S^T A S =$ diagonal matrix.

Symmetric Action-Assimilate update $W = A^{-1}$

The symmetric update

$$X_{k+1} = \arg\min_{X \in \mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad XAS = S, \quad X = X^T,$$

is then given by

$$X_{k+1} = \mathrm{quNac}(X_k, S, AS)$$

$$\stackrel{\text{def}}{=} S(S^T AS)^{-1}S^T + \left(I - S(S^T AS)^{-1}S^T A\right) X_k \left(I - AS(S^T AS)^{-1}S^T\right)$$

An update with many names: block BFGS, Balancing Precon. and LMP

[1]Gratton, S., Sartenaer, A., & Ilunga, J. T. (2011). On a Class of Limited Memory Preconditioners for Large-Scale Nonlinear Least-Squares Problems. SIAM Journal on Optimization, 21(3), 912935.

Quadratic Hereditary property

If $\mathbf{S} = [S_1, \ldots, S_k] \in \mathbb{R}^{n \times n}$ is non-singular and $S_i^T A S_j = 0$ for $1 \leq j < i \leq k$ and

$$X_{i+1} = \mathrm{quNac}(X_i, S_i, AS_i), \quad \text{for } i = 1, \ldots, k,$$

then X_{k+1} "inherits" previous actions

$$X_{k+1} A S_i = S_i, \quad \text{for } i = 1, \ldots, k.$$

Corollary: $X_{k+1} A \mathbf{S} = \mathbf{S} \Rightarrow X_{k+1} = A^{-1}$.

For sequence of system matrices $(A_k)_k$, choose S_k conjugate to A_k as an approximation!

Quadratic Hereditary property

If $\mathbf{S} = [S_1, \ldots, S_k] \in \mathbb{R}^{n \times n}$ is non-singular and $S_i^T AWAS_j = 0$ for $1 \leq j < i \leq k$ and

$$X_{i+1} = \mathrm{quNac}(X_i, S_i, AS_i, W), \quad \text{for } i = 1, \ldots, k,$$

then X_{k+1} "inherits" previous actions

$$X_{k+1} AS_i = S_i, \quad \text{for } i = 1, \ldots, k.$$

Corollary: $X_{k+1} A\mathbf{S} = \mathbf{S} \Rightarrow X_{k+1} = A^{-1}$.

For sequence of system matrices $(A_k)_k$, choose $A_k S_k$ conjugate to W_k as an approximation!

Preconditioned Sequence using Conjugate Gradients

quNac: quasi-Newton Action Assimilate

Set $X_0 = I$.
For $k = 1, \ldots,$
`Proxy solve` using CG
$$(x_k, S_k, A_k S_k) = \texttt{Precon\_CG}(X_k A_k x = X_k b_k)$$
`Update Preconditioner`
$$X_{k+1} = \text{quNac}(X_k, S_k, A_k S_k)$$
End For

[2]Morales, J. L., & Nocedal, J. (2000). Automatic Preconditioning by Limited Memory Quasi-Newton Updating. SIAM Journal on Optimization, 10(4), 10791096.

Preconditioned Sequence using Conjugate Gradients

quNac: quasi-Newton Action Assimilate

Set $X_0 = I$.
For $k = 1, \ldots,$
`Proxy solve` using CG
$(x_k, S_k, A_k S_k) = \texttt{Precon\_CG}(X_k A_k x = X_k b_k)$
`Update Preconditioner`
$X_{k+1} = \text{quNac}(X_k, S_k, A_k S_k)$
End For

What about Limited Memory variants?

[2] Morales, J. L., & Nocedal, J. (2000). Automatic Preconditioning by Limited Memory Quasi-Newton Updating. SIAM Journal on Optimization, 10(4), 10791096.

Preconditioned Sequence using Conjugate Gradients

LquNac: Limited Memory quasi-Newton Action Assimilate

Set $X_0 = I$.
For $k = 1, \ldots,$
`Proxy solve` using CG
$(x_k, S_k, A_k S_k) = \texttt{Precon\_CG}(OpX_k A_k x = OpX_k b_k)$
`Update Preconditioner`
$OpX_{k+1}(v) = \mathrm{quNac}(OpX^0_{k+1}(v), S_k, A_k S_k)$
End For

What about Limited Memory variants? This Limited Memory variant is a parallelizable LBFGS preconditioner[2]

[2] Morales, J. L., & Nocedal, J. (2000). Automatic Preconditioning by Limited Memory Quasi-Newton Updating. SIAM Journal on Optimization, 10(4), 10791096.

Unravelling Theorem

If the columns of $S := [s_1, \ldots, s_q]$ are $A-$conjugate and

$$X_{k+1} = \text{quNac}(X_k, S, AS)$$
$$= \arg \min_{X \in \mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad XAS = S, \quad X = X^T$$

Unravelling Theorem

If the columns of $S := [s_1, \dots, s_q]$ are $A-$conjugate and

$$
\begin{aligned}
X_{k+1} &= \mathrm{quNac}(X_k, S, AS) \\
&= \arg \min_{X \in \mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad XAS = S, \quad X = X^T
\end{aligned}
$$

then $X_{k+1} = X_k^q$ where $X_k^1 \stackrel{\text{def}}{=} X_k$ and

$$
\begin{aligned}
X_k^{i+1} &= \mathrm{quNac}(X_k^i, s_i, As_i) \\
&= \arg \min_{X \in \mathbb{R}^{n\times n}} \|X - X_k^i\|_{F(A)} \quad \text{subject to} \quad XAs_i = s_i, \quad X = X^T, \\
&= \mathrm{BFGS}(X_k^i, s_i, As_i), \quad \text{for } i = 0, \dots, q-1.
\end{aligned}
$$

Unravelling Theorem

If the columns of $S := [s_1, \ldots, s_q]$ are $A-$conjugate and

$$\begin{aligned} X_{k+1} &= \mathrm{quNac}(X_k, S, AS) \\ &= \arg\min_{X \in \mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad XAS = S, \quad X = X^T \end{aligned}$$

then $X_{k+1} = X_k^q$ where $X_k^1 \stackrel{\text{def}}{=} X_k$ and

$$\begin{aligned} X_k^{i+1} &= \mathrm{quNac}(X_k^i, s_i, As_i) \\ &= \arg\min_{X \in \mathbb{R}^{n\times n}} \|X - X_k^i\|_{F(A)} \quad \text{subject to} \quad XAs_i = s_i, \quad X = X^T, \\ &= \mathrm{BFGS}(X_k^i, s_i, As_i), \quad \text{for } i = 0, \ldots, q-1. \end{aligned}$$

That is

$$X_{k+1}(v) = \text{LBFGS-two-loops}(X_k, [s_1, \ldots, s_q], [As_1, \ldots, As_q], v)$$

but also

$$X_{k+1}(v) = \mathrm{quNac}(X_k, S, AS)(v) = \underline{S}\,\underline{S}^T v + \left(I - \underline{S}\,\underline{S}^T A\right) X_k \left(I - A\underline{S}\,\underline{S}^T\right) v,$$

where $\underline{S} = S(S^T AS)^{-1/2}$

Comparing the L-BFGS two-loop recursion with the LquNac

Input $OpX_k^0 : \mathbb{R}^n \to \mathbb{R}^n, \underline{S} = S(S^TAS)^{-1/2} = \left[\underline{s}_1, \dots, \underline{s}_q\right]$ and $v \in \mathbb{R}^n$.

Algorithm 3.1: LquNac

$\alpha \leftarrow \underline{S}^T v$

$\boxed{z} \leftarrow v - A\underline{S}\alpha$

$r \leftarrow OpX_k^0(z)$

$\beta \leftarrow (A\underline{S})^T r$

$\boxed{z} \leftarrow r + \underline{S}(\alpha - \beta)$

Output: z

Algorithm 3.2: two-loop recursion

. . . **for** $i = 1, \dots, q$ **do**

  $\alpha_i \leftarrow \underline{s}_i^T v$;

  $\boxed{v} \leftarrow v - \alpha_i A\underline{s}_i$;

$r \leftarrow OpX_k^0(v)$;

for $i = q, \dots, 1$ **do**

  $\beta_i \leftarrow (A\underline{s}_i)^T r$;

  $\boxed{r} \leftarrow r + \underline{s}_i(\alpha_i - \beta_i)$;

Output: r

Matrix-vector product instead of a loop.

Comparing the L-BFGS two-loop recursion with the LquNac

Input $OpX_k^0 : \mathbb{R}^n \to \mathbb{R}^n, \underline{S} = S(S^T AS)^{-1/2} = \left[\underline{s}_1, \dots, \underline{s}_q\right]$ and $v \in \mathbb{R}^n$.

Algorithm 3.3: LquNac

$\alpha \leftarrow \underline{S}^T v$
$\boxed{z} \leftarrow v - A\underline{S}\alpha$
$r \leftarrow OpX_k^0(z)$
$\beta \leftarrow (A\underline{S})^T r$
$\boxed{z} \leftarrow r + \underline{S}(\alpha - \beta)$
Output: z

Algorithm 3.4: two-loop recursion

$\dots$ **for** $i = 1, \dots, q$ **do**
  $\alpha_i \leftarrow \underline{s}_i^T v$;
  $\boxed{v} \leftarrow v - \alpha_i A\underline{s}_i$;
$r \leftarrow OpX_k^0(v)$;
for $i = q, \dots, 1$ **do**
  $\beta_i \leftarrow (A\underline{s}_i)^T r$;
  $\boxed{r} \leftarrow r + \underline{s}_i(\alpha_i - \beta_i)$;
Output: r

Matrix-vector product instead of a loop.

Comparing the L-BFGS two-loop recursion with the LquNac

Input OpX_k^0 … nd $v \in \mathbb{R}^n$.

Algorithm … op recursion

$\alpha \leftarrow \underline{S}^T v$

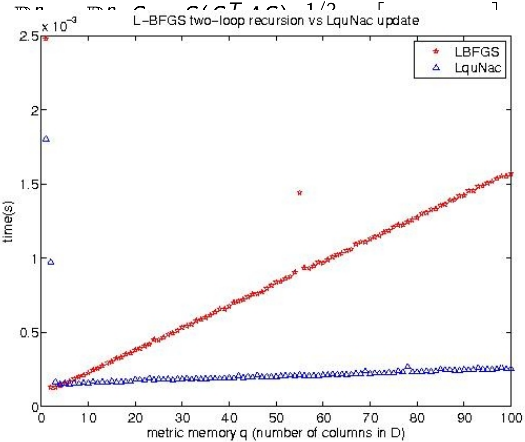
$\boxed{z} \leftarrow v - A$…

$r \leftarrow OpX_k^0(z$…

$\beta \leftarrow (A\underline{S})^T r$

$\boxed{z} \leftarrow r + \underline{S}($…);

Output: z



upto 6× Speedup on 6 cores!

Logistic L2 Regression tests:

$\min_w L_w(y, X) + \|w\|_2^2$

$$L_w(y, X) = \sum_{i=1}^{m} \ln\left(1 + \exp(-y_i \left\langle x^i, w \right\rangle)\right).$$

quNac	vs	BFGS
41		3
quNac	vs	Newton_CG
31		12
LquNac	vs	LBFGS
27		17
LquNac	vs	Newton_CG
14		29

Table : # fastest runs on 44 binary classifications problems from LibSVM

Logistic L2 Regression tests:

$\min_w L_w(y, X) + \|w\|_2^2$

$$L_w(y, X) = \sum_{i=1}^{m} \ln\left(1 + \exp(-y_i \left\langle x^i, w \right\rangle)\right).$$

quNac	vs	BFGS
41		3
quNac	vs	Newton_CG
31		12
LquNac	vs	LBFGS
27		17
LquNac	vs	Newton_CG
14		29

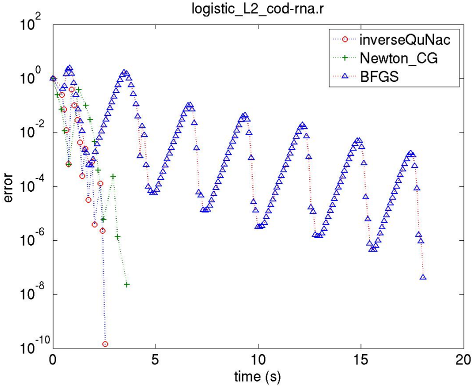
Table : # fastest runs on 44 binary classifications problems from LibSVM

Logistic L2 Regression tests:

$\min_w L_w(y, X) + \|w\|_2^2$

$$L_w(y, X) = \sum_{i=1}^{m} \ln\left(1 + \exp(-y_i \left\langle x^i, w \right\rangle)\right).$$

Dimensions $(n; m) = (8; 157413)$



Logistic pseudo-Huber Regression tests:

$$\min_{w} L_w(y, X) + R_\mu(w) \stackrel{\text{def}}{=} \mu \sum_{i=1}^{n} \left(\sqrt{1 + x_i^2/\mu^2} - 1 \right).$$

quNac	vs	BFGS
32		10
quNac	vs	Newton_CG
37		4
LquNac	vs	LBFGS
18		25
LquNac	vs	Newton_CG
24		16

Table : # fastest runs on 44 binary classifications problems from LibSVM

[2] Fountoulakis, K., & Gondzio, J. (2013). A Second-Order Method for Strongly Convex l1-regularization Problems.

Logistic pseudo-Huber Regression tests:

$$\min_{w} L_w(y, X) + R_\mu(w) \stackrel{\text{def}}{=} \mu \sum_{i=1}^{n} \left(\sqrt{1 + x_i^2/\mu^2} - 1 \right).$$

quNac	vs	BFGS
32		10
quNac	vs	Newton_CG
37		4
LquNac	vs	LBFGS
18		25
LquNac	vs	Newton_CG
24		16

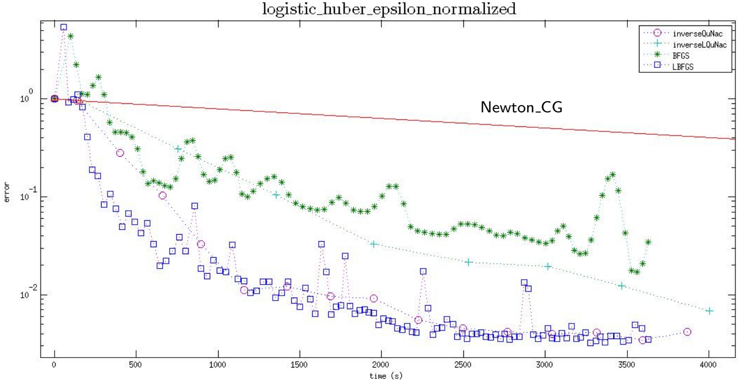
Table : # fastest runs on 44 binary classifications problems from LibSVM

[2] Fountoulakis, K., & Gondzio, J. (2013). A Second-Order Method for Strongly Convex l1-regularization Problems.

Logistic pseudo-Huber Regression tests:

$$\min_w L_w(y, X) + R_\mu(w) \stackrel{\text{def}}{=} \mu \sum_{i=1}^{n} \left(\sqrt{1 + x_i^2/\mu^2} - 1 \right).$$

Dimension $(n; m) = (2000, 400'000)$



Logistic pseudo-Huber Regression tests:

$$\min_w L_w(y,X) + R_\mu(w) \overset{\text{def}}{=} \mu \sum_{i=1}^{n} \left(\sqrt{1 + x_i^2/\mu^2} - 1 \right).$$

Dimension $(n; m) = (7129; 44)$

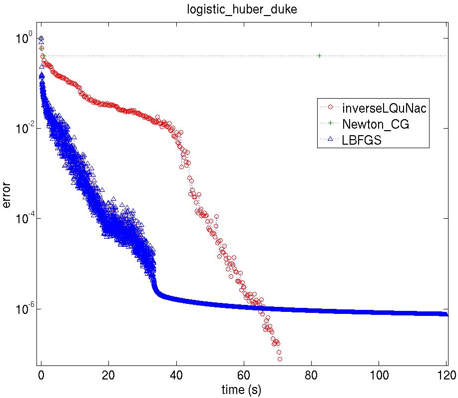


Table : ems from LibSVM

Conclusions

- Tool for updating preconditioners for slowly changing linear systems.
- A connection between quasi-Newton and Approximate Preconditioning formulation.
- Guideline for implementing with any Krylov method
- Successful implementation with CG
- Parallelizable L-BFGS type preconditioner
- Extensions to other iterative methods for solving systems

G.,R & Gondzio, J, Action constrained quasi-Newton methods January 2014, Technical Report ERGO-14-020

G.,R & P Richtárik, Randomized Iterative Methods for Inverting Matrices (in progress, August 2015?)

All block and randomized variants of Kaczmarz method (ARM), Gauss-Seidel, Coordinate descent, Randomized Newton ...

Randomized Methods

Let $S \in \mathbb{R}^{n\times p}$ be a random matrix and $W \succ 0 \in \mathbb{R}^{n\times n}$ then iterate $k = 1, \ldots,$

$$x_{k+1} = \arg\min_x \|x - x_k\|_{W^{-1}}, \quad \text{subject to } S^T A x = S^T b,$$

where $\langle x, y\rangle_{W^{-1}} = x^T W^{-1} y$.

A sampled action $S \to S^T A$ comes free!

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad S^T A X = S^T.$$

Same left action as inverse $S^T A X = S^T A A^{-1}$.

Randomized Methods

Let $S \in \mathbb{R}^{n\times p}$ be a random matrix and $W \succ 0 \in \mathbb{R}^{n\times n}$ then iterate

$$x_{k+1} = \arg\min_{x} \|x - x_k\|_{W^{-1}}, \quad \text{subject to } S^T A x = S^T b,$$

and

$$\mathbf{E}\left[\|x_k - x_*\|_{W^{-1}}^2\right] \leq \left(1 - \lambda_{\min}\mathbf{E}\left[W^{1/2} Z W^{1/2}\right]\right)^k \|x_0 - x_*\|_{W^{-1}}^2$$

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X \in \mathbb{R}^{n\times n}} \|X - X_k\|_{F(W^{-1})} \quad \text{subject to} \quad S^T A X = S^T.$$

and

$$\mathbf{E}\left[\|X_k - A^{-1}\|_{F(W^{-1})}^2\right] \leq \left(1 - \frac{1}{\mathbf{Tr}\left(\left(W^{1/2}\mathbf{E}\left[Z\right]W^{1/2}\right)^{-1}\right)}\right)^k \|X_0 - A^{-1}\|_{F(W^{-1})}^2$$

Where $Z = A^T S (S^T A W A^T S)^{-1} S^T A$.

Randomized Methods

Let $S \in \mathbb{R}^{n\times p}$ be a random matrix and $A \succ 0 \in \mathbb{R}^{n\times n}$ then iterate

$$x_{k+1} = \arg\min_{x} \|x - x_k\|_A, \quad \text{subject to } S^T A x = S^T b,$$

and

$$\mathbf{E}\left[\|x_k - x_*\|_A^2\right] \leq \left(1 - \frac{\lambda_{\min}(A)}{\mathbf{Tr}\,(A)}\right)^k \|x_0 - x_*\|_A^2$$

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad S^T A X = S^T, \quad X = X^T,$$

and

$$\mathbf{E}\left[\|X_k - A^{-1}\|_{F(A)}^2\right] \leq \left(1 - \frac{1}{\mathbf{Tr}\,(A)\,\mathbf{Tr}\,(A^{-1})}\right)^k \|X_0 - A^{-1}\|_{F(A)}^2$$

Where $Z = A^T S (S^T A W A^T S)^{-1} S^T A$.

When $W^{-1} = A$ and $S = e_i$ with probability $p_i = A_{ii}/\mathbf{Tr}\,(A)$

Randomized Methods

Let $S \in \mathbb{R}^{n\times p}$ be a random matrix and $\succ 0 \in \mathbb{R}^{n\times n}$ then iterate

$$x_{k+1} = \arg\min_{x} \|x - x_k\|_A, \quad \text{subject to } S^T A x = S^T b,$$

and

$$\mathbf{E}\left[\|x_k - x_*\|_A^2\right] \leq \left(1 - \frac{\lambda_{\min}(A)}{\mathbf{Tr}\,(A)}\right)^k \|x_0 - x_*\|_A^2$$

Quasi-Newton viewpoint: Action-Assimilate

$$X_{k+1} = \arg\min_{X\in\mathbb{R}^{n\times n}} \|X - X_k\|_{F(A)} \quad \text{subject to} \quad S^T A X = S^T, \quad X = X^T,$$

and

$$\mathbf{E}\left[\|X_k - A^{-1}\|_{F(A)}^2\right] \leq \left(1 - \frac{1}{\mathbf{Tr}\,(A)\,\mathbf{Tr}\,(A^{-1})}\right)^k \|X_0 - A^{-1}\|_{F(A)}^2$$

Where $Z = A^T S (S^T A W A^T S)^{-1} S^T A$.

G.,R & P Richtárik, Randomized Iterative Methods for Linear Systems arXiv:1506.03296