CENTRE EUROPÉEN DE RECHERCHE ET DE FORMATION AVANCÉE EN CALCUL SCIENTIFIQUE

On the use of the limited memory preconditioners for geosciences and aerodynamic shape optimization

Selime Gürol

Acknowledgements to

Francois Gallard, Mike Fisher, Serge Gratton, Benoit Pauwels, Philippe Toint, Jean Tshimanga

- In many engineering problems, we are dealing with the large-scale nonlinear optimization problem :

$$\min_{x\in\mathbb{R}^n} \quad f(x)$$
$$\text{subject to} \quad c_{\mathcal{E}}(x) = 0, \quad c_{\mathcal{I}}(x) \geq 0$$

where x are the variables, $f(x)$ is the smooth objective function, $c_{\mathcal{E}}(x), c_{\mathcal{I}}(x)$ are the smooth functions for the equality and inequality constraints.

- In many engineering problems, we are dealing with the large-scale nonlinear optimization problem :

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
$$\text{subject to} \quad c_{\mathcal{E}}(x) = 0, \quad c_{\mathcal{I}}(x) \geq 0$$

where x are the variables, $f(x)$ is the smooth objective function, $c_{\mathcal{E}}(x), c_{\mathcal{I}}(x)$ are the smooth functions for the equality and inequality constraints.

- Good algorithms to solve this problem should possess :
  - **Accuracy** : They should be able to identify a solution with precision, without being overly sensitive to errors
  - **Robustness** : They should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.

- In many engineering problems, we are dealing with the large-scale nonlinear optimization problem :

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
$$\text{subject to} \quad c_{\mathcal{E}}(x) = 0, \quad c_{\mathcal{I}}(x) \geq 0$$

  where x are the variables, $f(x)$ is the smooth objective function, $c_{\mathcal{E}}(x), c_{\mathcal{I}}(x)$ are the smooth functions for the equality and inequality constraints.

- Good algorithms to solve this problem should possess :
  - **Accuracy** : They should be able to identify a solution with precision, without being overly sensitive to errors
  - **Robustness** : They should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.
  - **Efficiency** : They should not require excessive computer time or storage

- In many engineering problems, we are dealing with the large-scale nonlinear optimization problem :

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
$$\text{subject to} \quad c_{\mathcal{E}}(x) = 0, \quad c_{\mathcal{I}}(x) \geq 0$$

where x are the variables, $f(x)$ is the smooth objective function, $c_{\mathcal{E}}(x), c_{\mathcal{I}}(x)$ are the smooth functions for the equality and inequality constraints.

- Good algorithms to solve this problem should possess :
  - **Accuracy** : They should be able to identify a solution with precision, without being overly sensitive to errors
  - **Robustness** : They should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.
  - **Efficiency** : They should not require excessive computer time or storage
- Beginning at x_0, optimization algorithms generates a **sequence of iterates** $\{x_k\}_{k=0}^{N}$ when it seems that a solution point has been approximated with sufficient accuracy.

- One of the most effective methods for large-scale nonlinear constrained optimization is the sequential quadratic programming (SQP) approach.

- In deciding how to move from one iterate x_k to the next x_{k+1}, the search direction p_k, **SQP uses the local information** and **solves the quadratic subproblem at the iterate** (x_k, λ_k) :

$$\begin{aligned} \min_{p \in \mathbb{R}^n} \quad & \frac{1}{2} p^T \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k) p + \nabla f(x_k)^T p \\ \text{subject to} \quad & \nabla c_{\mathcal{E}}(x_k)^T p + c_{\mathcal{E}}(x_k) = 0 \\ & \nabla c_{\mathcal{I}}(x)^T p + c_{\mathcal{I}}(x) \geq 0 \end{aligned}$$

where $\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i$ is the Lagrangian function.

- The objective in this subproblem is an **approximation to the change in the Lagrangian function in moving from** x_k **to** $x_k + p$ while the constraints are the linearizations of the constraints.

- We are dealing with a sequence of quadratic minimization problems :

$$\min_{p \in \mathbb{R}^n} \frac{1}{2} p^T A_k p + {b_k}^T p$$

where A_k consist of the curvature information and b_k consists of the gradient information at the current iteration.

- Once we solve this **quadratic subproblem**, we update the iterate : $x_{k+1} = x_k + p_k$

- We are dealing with a sequence of quadratic minimization problems :

$$\min_{p \in \mathbb{R}^n} \frac{1}{2} p^T A_k p + {b_k}^T p$$

where A_k consist of the curvature information and b_k consists of the gradient information at the current iteration.

- Once we solve this **quadratic subproblem**, we update the iterate : $x_{k+1} = x_k + p_k$

- Minimization of quadratic problems is equivalent to the solution of the linear systems :

$$A_k \, p = b_k$$

- Solve in sequence

$$\boxed{A_1 p = b_1,\ A_2 p = b_2,\ \ldots, A_k p = b_k}$$

with an **iterative method**.

- When iterative methods are used, preconditioning is necessary to attain convergence in a reasonable amount of time !

- In this talk, we will focus on the designing efficient preconditioners based on the information herited from the previous linear systems to accelerate the convergence rate of the current system.

Preconditioning

Second level preconditioning

Application to variational data assimilation

Application to aerodynamic shape optimization

Conclusions

- The aim of preconditioning techniques is to **transform a system** $Ap = b$ into a **new equivalent system**

$$FAp = Fb$$

with a **more favorable eigenvalues distribution of the matrix** FA**.**

- To perform such a transformation, one uses a so-called preconditioning matrix F.

- For symmetric and positive definite systems, the rate of convergence of the method, for instance Conjugate Gradients, depends on the distribution of the eigenvalues of A. The more clustered spectrum converges faster.

- For nonsymmetric problems, it is difficult to analyse the convergence only with the eigenvalue spectrum. However, a clustered spectrum (away from 0) often results in rapid convergence, especially when the preconditioned matrix is close to normal.

Designing good preconditioners

Ideally, the preconditioner F must :

- approximate the inverse of A
- make FA have more eigenvalue clusters
- decrease the condition number of FA compare to that of A
- be cheap to construct and apply
- The preconditioned system should be easy to solve

$\Rightarrow$ The preconditioned iteration should converge rapidly, while ensuring that each iteration is not too expensive

- Depending on the properties of the system, (symmetry, positive definiteness, sparsity, etc ..) there are several strategies to build good preconditioners. For instance, incomplete Cholesky factorization, domain decomposition techniques, diagonal scaling, sparse approximate inverses, etc.

Review Articles :

- M. Benzi (2002), Preconditioning techniques for large linear systems : A survey
- K. Chen (2005), Matrix preconditioning techniques and applications
- A. Wathen (2015), Preconditioning

$\Rightarrow$ We name these preconditioners as the first-level preconditioner.

Outline of the presentation

Preconditioning

Second level preconditioning

Application to variational data assimilation

Application to aerodynamic shape optimization

Conclusions

Second level preconditioning

- Let us assume that A is fixed along the sequence.
- Solve $Ap = b_1$:

$$\boxed{F_0 A p = F_0 b_1}$$

where F_0 is the first-level preconditioner and extract information info_1.

- Let us assume that A is fixed along the sequence.
- Solve $Ap = b_1$:

$$\boxed{F_0 A p = F_0 b_1}$$

where F_0 is the first-level preconditioner and extract information info_1.

- Solve $Ap = b_2$ using info_1 to precondition :

$$\boxed{F_1(F_0, \mathsf{info}_1) A p = F_1(F_0, \mathsf{info}_1) b_2}$$

and extract information info_2.

Second level preconditioning

- Let us assume that A is fixed along the sequence.
- Solve $Ap = b_1$:

$$\boxed{F_0 A p = F_0 b_1}$$

where F_0 is the first-level preconditioner and extract information info_1.

- Solve $Ap = b_2$ using info_1 to precondition :

$$\boxed{F_1(F_0, \text{info}_1) A p = F_1(F_0, \text{info}_1) b_2}$$

and extract information info_2.

- Solve $Ap = b_3$ using info_2 (and possibly info_1) to precondition

$$\boxed{F_2(F_0, \text{info}_1, \text{info}_2) A p = F_2(F_0, \text{info}_1, \text{info}_2) b_3}$$

and extract information info_3.

- Let us assume that A is fixed along the sequence.
- Solve $Ap = b_1$:

$$\boxed{F_0 A p = F_0 b_1}$$

  where F_0 is the first-level preconditioner and extract information info_1.
- Solve $Ap = b_2$ using info_1 to precondition :

$$\boxed{F_1(F_0, \mathsf{info}_1) A p = F_1(F_0, \mathsf{info}_1) b_2}$$

  and extract information info_2.
- Solve $Ap = b_3$ using info_2 (and possibly info_1) to precondition

$$\boxed{F_2(F_0, \mathsf{info}_1, \mathsf{info}_2) A p = F_2(F_0, \mathsf{info}_1, \mathsf{info}_2) b_3}$$

  and extract information info_3.
- ...

- Let us assume that A is fixed along the sequence.
- Solve $Ap = b_1$:

$$\boxed{F_0 A p = F_0 b_1}$$

where F_0 is the first-level preconditioner and extract information info_1.

- Solve $Ap = b_2$ using info_1 to precondition :

$$\boxed{F_1(F_0, \text{info}_1) A p = F_1(F_0, \text{info}_1) b_2}$$

and extract information info_2.

- Solve $Ap = b_3$ using info_2 (and possibly info_1) to precondition

$$\boxed{F_2(F_0, \text{info}_1, \text{info}_2) A p = F_2(F_0, \text{info}_1, \text{info}_2) b_3}$$

and extract information info_3.

- ...

$\Rightarrow$ F_k is called the second-level preconditioner.

- Approximate A^{-1} or its effect on a vector by using set of directions.
- Available information : $q = Ap$
- We can use the pairs (p, q) to approximate A^{-1}
- Which vectors have a product with A ?

$\longrightarrow$ for instance : descent directions from Conjugate Gradients
$(q_i = Ap_i)$

- We will focus on the idea of warm-start preconditioning techniques for second-level preconditioning [Morales and Nocedal, 1999], which is generalized by [Gratton et al., 2011] under the name of Limited Memory preconditioners (LMPs).

The idea :

- Let A and F_0 be **symmetric positive definite** matrices of order n
- Let S be any n by ℓ matrix of **rank** ℓ, with $\ell \leq n$ and $Y = AS$
- Find an update to F_0 : $F = \Delta F + F_0$ such that

$$\min_{\Delta F} \|\Delta F\|_{\mathcal{F}}$$
$$\text{subject to } F = F^T \text{ and } FY = S$$

- We combine the most recently observed information about the Hessian with the existing knowledge in our current Hessian approximation.

- We will focus on the idea of warm-start preconditioning techniques for second-level preconditioning [Morales and Nocedal, 1999], which is generalized by [Gratton et al., 2011] under the name of Limited Memory preconditioners (LMPs).

The idea :

- Let A and F_0 be **symmetric positive definite** matrices of order n
- Let S be any n by ℓ matrix of **rank** ℓ, with $\ell \leq n$ and $Y = AS$
- Find an update to F_0 : $F = \Delta F + F_0$ such that

$$\min_{\Delta F} \|\Delta F\|_{\mathcal{F}}$$

$$\text{subject to } F = F^T \text{ and } FY = S$$

- We combine the most recently observed information about the Hessian with the existing knowledge in our current Hessian approximation.

Definition :

$$\boxed{F = \left[I_n - S(S^TY)^{-1}Y^T\right] F_0 \left[I_n - Y(S^TY)^{-1}S^T\right] + S(S^TY)^{-1}S^T}$$

is called the LMP being an approximation to A^{-1}.

- $F_0 \equiv$ first-level preconditioner, $F \equiv$ second-level preconditioner

Some properties of the LMP

- F is symmetric and positive definite.
- $F = A^{-1}$ if S is of order n.
- At least ℓ eigenvalues are clustered at 1.
- The remaining part of the spectrum does not expand.
- It requires additional memory : we need to save the column vectors of the $S \in \mathbb{R}^{n \times \ell}$ and $AS \in \mathbb{R}^{n \times \ell}$.
- It is cheap to apply : one matrix-vector product by M and $8kn$ additional flops.

Preconditioning

Second level preconditioning

Application to variational data assimilation

Application to aerodynamic shape optimization

Conclusions

A dynamical system is characterized by state variables, e.g.

- velocity components
- pressure
- density
- temperature
- gravitational potential

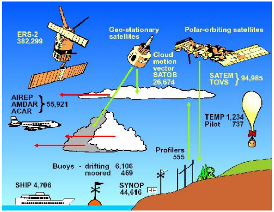
Goal : predict the state of the system at a future time from

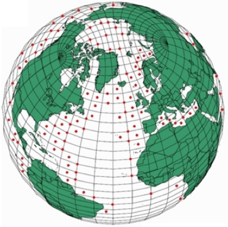
- dynamical integration model
- observational data

Applications : climate, meteorology, oceanography, neutronics, finance, ...
$\longrightarrow$ forecasting problems

- A **dynamical integration model** predicts the state of the system given the state at an earlier time.
  $\longrightarrow$ integrating may lead to very large **prediction errors**
  (inexact physics, discretization errors, approximated parameters)

- A **dynamical integration model** predicts the state of the system given the state at an earlier time.
  $\longrightarrow$ integrating may lead to very large **prediction errors** (inexact physics, discretization errors, approximated parameters)
- **Observational data** are used to improve accuracy of the forecasts.
  $\longrightarrow$ but the data are inaccurate (measurement noise, under-sampling
  $\longrightarrow$ 10^7 observations (10^9 variables) processed every day : inverse big data





Solve a large-scale non-linear weighted least-squares problem :

$$\min_{x \in \mathbb{R}^n} \quad \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{j=0}^{N} \left\| \mathcal{H}_j\left(\mathcal{M}_j(x)\right) - y_j \right\|^2_{R_j^{-1}}$$

where

- $x \equiv x(t_0)$ is the control variable in $\mathbb{R}^n$, $n \sim 10^6$.
- $\mathcal{M}_j$ are model operators : $x(t_j) = \mathcal{M}_j(x(t_0))$
- $\mathcal{H}_j$ are observation operators : $y_j \approx \mathcal{H}_j(x(t_j))$
- the obervations y_j and the background x_b are noisy
- B and R_j are error covariance matrices

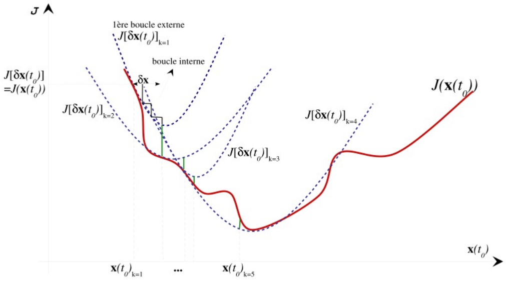
→ Solve a large-scale non-linear weighted least-squares problem

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2}||x - x_b||^2_{B^{-1}} + \frac{1}{2}\sum_{j=0}^{N} ||\mathcal{H}_j(\mathcal{M}(t_j, t_0)(x)) - \mathbf{y}_j||^2_{R_j^{-1}}$$

→ Typically solved using a Gauss-Newton algorithm

- solve the linearized subproblem

$$\min_{p^{(k)} \in \mathbb{R}^n} \quad \frac{1}{2}||p^{(k)} - (x_b - x^{(k)})||^2_{B^{-1}} + \frac{1}{2}\left\|H^{(k)}p^{(k)} - d^{(k)}\right\|^2_{R^{-1}}$$



- update $x^{(k+1)} = x^{(k)} + p^{(k)}$

From optimality conditions, at Gauss-Newton iteration k we have :

$$\underbrace{(B^{-1} + H_k^T R^{-1} H_k)}_{A_k} p = \underbrace{B^{-1}(x_b - x) + H_k^T R^{-1} d_k}_{b_k}$$

where A_k is a large, symmetric and positive definite matrix.

From optimality conditions, at Gauss-Newton iteration k we have :

$$\underbrace{(B^{-1} + H_k^T R^{-1} H_k)}_{A_k} p = \underbrace{B^{-1}(x_b - x) + H_k^T R^{-1} d_k}_{b_k}$$

where A_k is a large, symmetric and positive definite matrix.

Solution :

- Solve in sequence

$$\boxed{A_1\, p = b_1,\; A_2\, p = b_2,\; \ldots\; , A_k\, p = b_k,\; \ldots}$$

  by a preconditioned Krylov method (Conjugate Gradients or Lanczos method).

- Precondition the first linear system with B (first-level preconditioner).

- Use limited memory preconditioning for second-level preconditioning (Morales and Nocedal 2000, Gratton, et al. 2011)

$\longrightarrow$ There are three particular cases for $\mathbf{F}$ [Gratton et al., 2011] :

$$F = \left[I_n - S(S^T AS)^{-1} S^T A\right] F_0 \left[I_n - AS(S^T AS)^{-1} S^T\right] + S(S^T AS)^{-1} S^T$$

1. The quasi-Newton LMP :
   $\longrightarrow$ $\mathbf{S}$ is a column matrix consisting of the descent directions generated by a CG or Lanczos method.
   $\longrightarrow$ It amounts to the preconditioner proposed by [Morales and Nocedal, 1999].

2. The spectral LMP :
   $\longrightarrow$ $\mathbf{S}$ is a column matrix consisting of the eigenvectors of $\mathbf{A}$.
   $\longrightarrow$ It amounts to the preconditioner proposed by [Fisher, 1998]). In practise, eigenpairs are approximated with Ritz pairs.

3. The Ritz LMP :
   $\longrightarrow$ $\mathbf{S}$ is a column matrix consisting of the Ritz pairs.

Numerical performance of LMPs on ocean DA

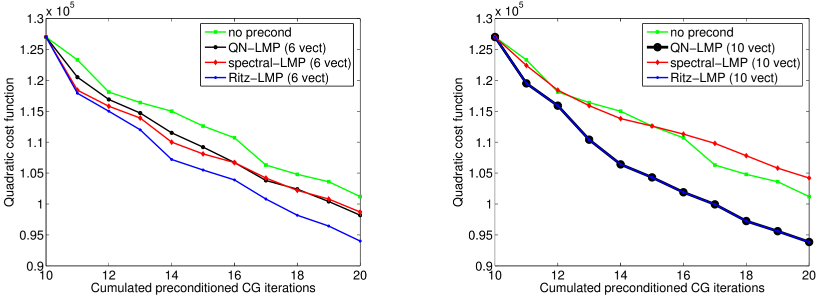
[Tshimanga et al., 2008]

- A realistic outer/inner loop configuration is considered :
  - 3 outer loops of Gauss-Newton (linearization)
  - 10 inner loops of conjugate gradient (on each of the 3 systems)
- The performance is measured by the value of the quadratic cost function
- The convergence of Ritz pairs is measured by the backward errors :

$$\frac{\|Az_i - \theta_i z_i\|}{\|A\|\|z_i\|}$$

- An unpreconditioned conjugate gradient is run on the first system to produce 10 vectors from which 2, 6 and 10 relevant ones are selected :
  - Ritz-vectors are selected according to their convergence
  - Descent directions are selected as the latest ones

Numerical performance of LMPs on ocean DA



- (Inexact) spectral-LMP is sensitive to the error on the approximation of the exact eigenpairs by Ritz pairs
- The Ritz-LMP may perform better than the (inexact) spectral-LMP and the **quasi-Newton LMP**
- The Ritz-LMP and the **quasi-Newton LMP** are analytically equivalent when they are constructed with all available information from a CG-like method run on a same matrix

[Tshimanga et al., 2008]

Airbus aims to use the new generation engines in order to provide significant improvement in terms of Specific Fuel Consumption, while increasing the nominal range of the re-engined airplane.

- This type of new engine is characterized by larger pylon (connection between nacelle and wing) and nacelles, and leads to install the engine closer to the wing.
- The fairing shape and stiffness design of the pylon is multidisciplinary : has to tackle strong geometrical layout constraints as well as aero-elastic and aerodynamic interactions with wing and nacelle.
- A multidisciplinary compromise drives the engine positioning and the pylon shape design.

The MDO project at IRT Saint Exupéry assess the impact of the engine position variation on the global aircraft performances, such as the aircraft operational cost, the Cash Operating Cost (COC) :

$\min_{x^{\mathrm{a}}, x^{\mathrm{s}}, z} COC(x^{\mathrm{a}}, x^{\mathrm{s}}, z)$ (Upper level)

subject to — subject to

Aerodynamics
x^{a} minimizes $f^{\mathrm{a}}(x^{\mathrm{a}}, z)$
subject to $g^{\mathrm{a}}(x^{\mathrm{a}}, z) \leq 0.$

Structure
x^{s} minimizes $f^{\mathrm{s}}(x^{\mathrm{s}}, z)$
subject to $g^{\mathrm{s}}(x^{\mathrm{s}}, z) \leq 0.$

(Lower level)

▷ The MDO problem is formulated as a bilevel optimization problem.

Multidisciplinary design optimization (MDO)

of an aircraft engine pylon

The MDO project at IRT Saint Exupéry assess the impact of the engine position variation on the global aircraft performances, such as the aircraft operational cost, the Cash Operating Cost (COC) :

$\min_{x^{\mathrm{a}}, x^{\mathrm{s}}, z} COC(x^{\mathrm{a}}, x^{\mathrm{s}}, z)$ (Upper level)

subject to — subject to

Aerodynamics
x^{a} minimizes $f^{\mathrm{a}}(x^{\mathrm{a}}, z)$
subject to $g^{\mathrm{a}}(x^{\mathrm{a}}, z) \leq 0$.

Structure
x^{s} minimizes $f^{\mathrm{s}}(x^{\mathrm{s}}, z)$
subject to $g^{\mathrm{s}}(x^{\mathrm{s}}, z) \leq 0$.

(Lower level)

▷ Aerodynamics is optimized more slowly than Structure.
▷ As many as different alternative displacements $z = (dX; dZ)$ are envisaged, similar bound constrained Aerodynamics problems are solved.

- Aerodynamics is parameterized as a nonlinear problem with about 400 variables subject to bound-constraints [Guénot et al. 2018].

$$\min_{x^a \in \mathbb{R}^n} f(x^a, z_1), \quad \min_{x^a \in \mathbb{R}^n} f(x^a, z_2), \quad \dots \quad \min_{x^a \in \mathbb{R}^n} f(x^a, z_k)$$

$$\text{s.t. } \ell \preceq x^a \preceq u \quad \text{s.t. } \ell \preceq x^a \preceq u \quad \dots \quad \text{s.t. } \ell \preceq x^a \preceq u$$

- Sequence of bound constrained optimization problems
- One instance effectively solved by the L-BFGS-B algorithm
- Function evaluations for aero simulations are time consuming
- Assume that the curvature of f is only moderately sensitive to $\boldsymbol{z}$

▶ **Goal :** solve Aerodynamics with fewer objective evaluations by using second-level preconditioners.

▶ **Idea :** We seek to incorporate curvature information from an earlier instance Aerodynamics($\boldsymbol{z}'$) into L-BFGS-B and solve instance Aerodynamics($\boldsymbol{z}$)

- L-BFGS-B is an optimization algorithm for differentiable functions subject to bound-constraints (also called "box-constraints") :

$$\text{minimize } f(x) \text{ subject to } \ell \preceq x \preceq u.$$

  BFGS [Dennis and Moré, 1977] is an algorithm for unconstrained optimization named after Broyden, Fletcher, Goldfarb and Shanno.

  L-BFGS [Nocedal, 1980] is a variant of BFGS using limited memory.

  L-BFGS-B [Byrd et al., 1995] extends L-BFGS to bound-constraints.

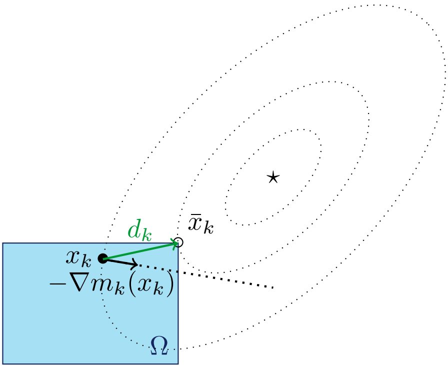
- L-BFGS-B is a quasi-Newton method : an alternative to Newton's method where the Hessian $\nabla^2 f(x_k)$, which contains the curvature information of f at the iterate x_k, is approximated by a matrix B_k.

  The objective is approximated by a quadratic m_k near the iterate x_k :

$$f(x) \approx m_k(x) = f(x_k) + \nabla f(x_k)^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top B_k (x - x_k).$$

Step 1 : Find a descent direction

Find $\bar{x}_k$ that approximately minimizes m_k in Ω and set $d_k = \bar{x}_k - x_k$.



Step 2 : Minimize f in this direction (*line search*)

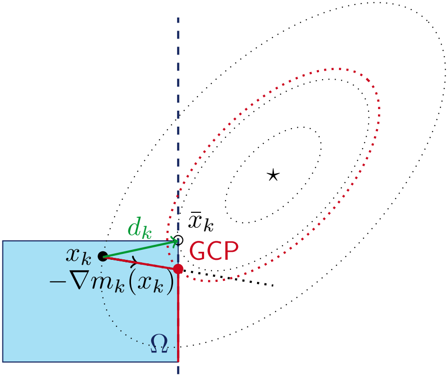
$x_{k+1} = \text{argmin}\{f(x) : \ell \preceq x \preceq u$ with $x = x_k + \lambda d_k$ for some λ in $\mathbb{R}_+\}$.

Step 1 : Find a descent direction

Find $\bar{x}_k$ that approximately minimizes m_k in Ω and set $d_k = \bar{x}_k - x_k$.

a. Guess the bounds active at $\bar{x}_k$:

b. Minimize m_k on the active space.



Step 2 : Minimize f in this direction (*line search*)

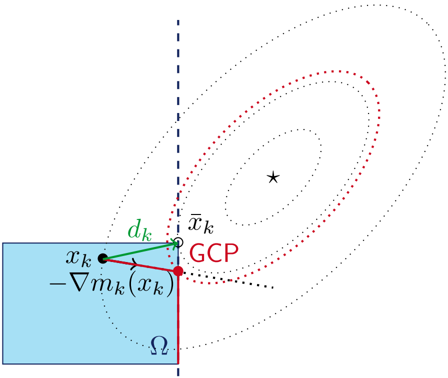
$x_{k+1} = \operatorname{argmin}\{f(x) : \ell \preceq x \preceq u$ with $x = x_k + \lambda d_k$ for some λ in $\mathbb{R}_+\}$.

Step 1 : Find a descent direction

Find $\bar{x}_k$ that approximately minimizes m_k in Ω and set $d_k = \bar{x}_k - x_k$.

a. Guess the bounds active at $\bar{x}_k$:
   Find the Generalized Cauchy Point (GCP) that minimizes m_k on the projected steepest descent path (Projection onto the feasible set).
   Select the GCP's active bounds.

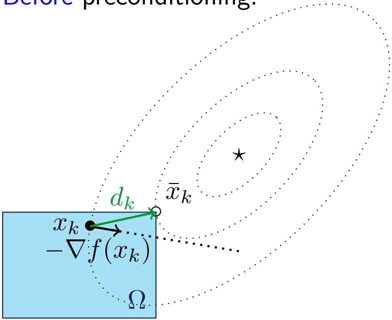
b. Minimize m_k on the active space.



Step 2 : Minimize f in this direction (*line search*)

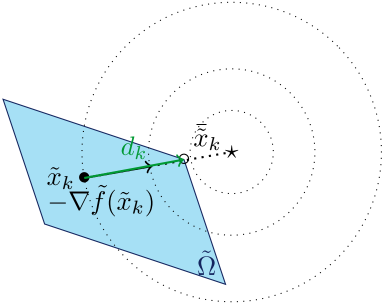
$x_{k+1} = \mathrm{argmin}\{f(x) : \ell \preceq x \preceq u$ with $x = x_k + \lambda d_k$ for some λ in $\mathbb{R}_+\}$.

Before preconditioning.



Steepest descent $-\nabla f(x_k)$ is orthogonal to the level lines of f. It may not be directed at the unconstrained minimizer of f.

After preconditioning.



Preconditioning is a change of variable $\tilde{x} = L^{-1}x$ that yields a new objective function $\tilde{f}(\tilde{x}) \stackrel{\text{def}}{=} f(L\tilde{x})$ with more spherical level lines.

Preconditioning of bound-constrained optimization

- By preconditioning the minimization of $f(x)$ in $\Omega = \{x : \ell \preceq x \preceq u\}$ we mean applying a change of variable

$$\tilde{x} = L^{-1}x$$

and minimizing

$$\tilde{f}(\tilde{x}) \stackrel{\text{def}}{=} f(L\tilde{x})$$

in the polyhedron $\tilde{\Omega} \stackrel{\text{def}}{=} L^{-1}(\Omega) = \{\tilde{x} : \ell \preceq L\tilde{x} \preceq u\}$.

- For quasi-Newton type methods, preconditioning can be considered as providing a better initial Hessian.
- We aim to get better oriented descent directions towards the minimum, accordingly fewer evaluations of f.
- The price to pay is linear contraints : Projection onto bounds become projection onto a polytope.

- Consider an earlier instance Aerodynamics(z'), already solved. During the minimization, secant pairs (s_i, y_i) :

$$B_k s_i = y_i$$

for $i = 1, ..., \ell$ are saved (accumulating curvature information) to construct the inverse approximation to B_k.

- We propose to precondition upcoming instances Aerodynamics(z) with the limited memory preconditioner B_k^{-1}.

The change of variable L is obtained by splitting : $B_k^{-1} = LL^\top$.

The L-BFGS-B Fortran code [Zhu et al., 1997] is wrapped in the Python library SciPy and ready to be used in GEMS (IRT's software for MDO).

Implementation requirements

- A more flexible implementation of the L-BFGS-B algorithm was necessary to enable preconditioning.
- This new code needed to be consistent with the Fortran reference.

L-BFGS-B as a GEMS optimization library

- L-BFGS-B is implemented in the Python language.
- This implementation is validated on CUTEr [Gould et al., 2003] test problems and used for the Aerodynamics problem.
- For Aerodynamics, the cost of projection is negligible relative to the very expensive evaluation cost. In analysing the results we focus on the number of function evaluations.

PROBLEM	n	(P-)L-BFGS-B nact	(P-)L-BFGS-B nfg	(P-)L-BFGS-B time	P-L-BFGS-B nact	P-L-BFGS-B nfg	P-L-BFGS-B time	nfg gain	f - f_prec
		maxcor = 10							
ALLINIT	4	1	17	0.06	1	**15**	0.08	**+12%**	**-6e-14**
BIGGS5	6	1	63	0.28	1	**36**	0.17	**+43%**	**+6e-03**
BQPGASIM	50	10	17	0.12	10	**13**	0.20	**+24%**	**-4e-10**
HATFLDA	4	0	41	0.27	0	**40**	0.13	**+2%**	**-2e-10**
HATFLDB	4	1	33	0.09	1	**21**	0.08	**+36%**	**-1e-12**
HATFLDC	25	0	23	0.13	0	**16**	0.15	**+30%**	**+8e-12**
HS110	10	0	8	0.08	0	8	0.04	+0%	**-3e-13**
MAXLIKA	8	3	56	0.22	1	**68**	0.35	**-21%**	**+1e-02**
PALMER1	4	0	36	0.16	0	**31**	0.15	**+14%**	**-5e-13**
PALMER2	4	0	32	0.11	0	**26**	0.16	**+19%**	**-1e-13**
PALMER3	4	1	12	0.06	1	**9**	0.03	**+25%**	**+2e-08**
PALMER4	4	1	13	0.08	1	**8**	0.03	**+38%**	**-2e-08**
PROBPENL	500	0	4	0.02	0	**3**	0.26	**+25%**	**-1e-18**
PSPDOC	4	1	12	0.04	1	12	0.12	+0%	**+7e-11**
S368	8	2	13	0.11	3	**10**	0.09	**+23%**	**+2e-01**
MAX								**+43%**	**+2e-01**
MEAN								**+18%**	**+1e-02**
MIN								**-21%**	**-2e-08**

L-BFGS-B set up with a limited memory preconditioner yields significant gain in terms of evaluation cost on average.

PROBLEM	n	maxcor = 5							
		(P-)L-BFGS-B			P-L-BFGS-B				
		nact	nfg	time	nact	nfg	time	nfg gain	f - f_prec
BIGGS5_00	6	1	63	0.61	1	**36**	0.13	**+43%**	**+6e-03**
BIGGS5_01 (9.13e-03)	6	1	62	0.24	1	**28**	0.10	**+55%**	**+6e-03**
BIGGS5_02 (1.60e-02)	6	1	59	0.29	1	**37**	0.14	**+37%**	**+6e-03**
BIGGS5_03 (2.45e-02)	6	1	62	0.20	1	**37**	0.20	**+40%**	**+6e-03**
BIGGS5_04 (5.05e-02)	6	1	65	0.25	1	**42**	0.17	**+35%**	**+6e-03**
BIGGS5_05 (5.29e-02)	6	1	57	0.20	1	**42**	0.15	**+26%**	**-7e-06**
BIGGS5_06 (5.24e-02)	6	1	65	0.27	1	**43**	0.18	**+34%**	**-7e-06**
BIGGS5_07 (9.16e-02)	6	1	99	0.36	1	**43**	0.18	**+57%**	**-2e-05**
BIGGS5_08 (1.50e-02)	6	1	72	0.28	1	**36**	0.16	**+50%**	**+6e-03**
BIGGS5_09 (3.38e-02)	6	1	59	0.22	1	**42**	0.17	**+29%**	**+6e-03**
BIGGS5_10 (9.60e-02)	6	1	111	0.42	1	**44**	0.23	**+60%**	**-2e-05**
BIGGS5_11 (6.55e-02)	6	1	64	0.28	1	**46**	0.23	**+28%**	**+6e-03**
BIGGS5_12 (6.45e-02)	6	1	65	0.30	1	**42**	0.22	**+35%**	**+6e-03**
MAX								**+60%**	**+6e-03**
MEAN								**+41%**	**+4e-03**
MIN								**+26%**	**-2e-05**

- We use GEMS platform to perform a bi-level formulation based on aero-structure optimization.

- Similar aerodynamic optimization problems are solved at each iteration of the system optimization.

- P-L-BFGS-B algorithm allowed to a significant gain in computational time. Similar cost function value reduction is obtained after 8 iterates of P-LBFGS-B instead of 16 iterates L-BFGS-B. [Gallard, Gratton, Gürol, Pauwels, Toint (2020)]

Conclusions

- Preconditioning is a key issue and widely used method in the computational efficiency of the iterative solvers.
- When solving a sequence of (slowly varying) linear systems or quadratic subproblems, inherited information can be used to further accelerate the convergence.
- LMPs are already operational in numerical weather forecast, and their potential use for other areas such as ocean data assimilation is well-known.
- We have shown as well the performance of the LMPs for indefinite systems arising in time-parallel formulation of the variational data assimilation [Fisher et al., 2018].
- Recently, we show that there is a potential in accelerating the convergence of the aerodynamic shape optimization by using the LMPs. In this case a special attention needs be paid for the constraints.

Thank you for your attention !

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995).
A limited memory algorithm for bound constrained optimization.
SIAM Journal on Scientific Computing, 16(5) :1190–1208.

Dennis, Jr, J. E. and Moré, J. J. (1977).
Quasi-newton methods, motivation and theory.
SIAM review, 19(1) :46–89.

Fisher, M. (1998).
Minimization algorithms for variational data assimilation.
In *Seminar on Recent Developments in Numerical Methods for Atmospheric Modelling, 7-11 September 1998*, pages 364–385, Shinfield Park, Reading. ECMWF, ECMWF.

Fisher, M., Gratton, S., Gürol, S., Trémolet, Y., and Vasseur, X. (2018).
Low rank updates in preconditioning the saddle point systems arising from data assimilation problems.
Optimization Methods and Software, 33(1) :45–69.

Gould, N. I., Orban, D., and Toint, P. L. (2003).
Cuter and sifdec : A constrained and unconstrained testing environment, revisited.

ACM Transactions on Mathematical Software (TOMS), 29(4) :373–394.

Gratton, S., Sartenaer, A., and Tshimanga, J. (2011).
On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides.
SIAM Journal on Optimization, 21(3) :912–935.

Morales, J. L. and Nocedal, J. (1999).
Automatic preconditioning by limited memory quasi-newton updating.
SIAM Journal on Optimization, 10 :1079–1096.

Nocedal, J. (1980).
Updating quasi-newton matrices with limited storage.
Mathematics of computation, 35(151) :773–782.

Tshimanga, J., Gratton, S., Weaver, A. T., and Sartenaer, A. (2008).
Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation.
Quarterly Journal of the Royal Meteorological Society, 134(632) :751–769.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997).
Algorithm 778 : L-BFGS-B : Fortran subroutines for large-scale bound-constrained optimization.

ACM Transactions on Mathematical Software (TOMS), 23(4) :550–560.

- Assume that a preconditioner $F(= CC^T)$ is nonsingular inverse approximation of the matrix A, the system $Ax = b$ can then be transformed by using :

1. left preconditioner :

$$F\,A\,x = F\,b$$

2. split preconditioner :

$$C^T\,A\,C\,y = C^T\,b, \qquad x = C\,y$$

3. right preconditioner

$$A\,F\,y = b, \qquad x = F\,y$$

- These systems have the same solution but may be easier to solve.
- The choice depends on the availability of the matrices, the choice of the iterative method, problem characteristics, etc.
- When using Krylov subspace methods, F can be applied as an operator.