Dual Methods for large-scale inverse problems: theoretical and practical aspects

Serge Gratton
University of Toulouse
CERFACS-IRIT, Toulouse, France

Joint work with S. Gurol, D. Titley-Peloquin, Ph.L. Toint, and J. Tshimanga

Seminaire du CAS 2014
Introduction to data assimilation

Dual iterative solvers

Multigrid solvers

Summary and ongoing related work
Introduction to data assimilation

Dual iterative solvers

Multigrid solvers

Summary and ongoing related work
A dynamical system is characterized by state variables, e.g.

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

→ forecasting problems
A dynamical integration model predicts the state of the system given the state at an earlier time.

→ 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.
→ integrating may lead to very large prediction errors (inexact physics, discretization errors, approximated parameters)

Observational data are used to improve accuracy of the forecasts.
→ but the data are inaccurate (measurement noise, under-sampling)
A **dynamical integration model** predicts the state of the system given the state at an earlier time.

→ integrating may lead to very large **prediction errors**
  (inexact physics, discretization errors, approximated parameters)

**Observational data** are used to improve accuracy of the forecasts.

→ but the data are **inaccurate** (measurement noise, under-sampling)

DA uses a weighted combination of **data** and **dynamics**.
Example

known situation 2 days ago and background prediction
Example

- known situation 2 days ago and background prediction
- record data for the past 2 days
Example

- known situation 2 days ago and background prediction
- record data for the past 2 days
- minimize the difference between the model and observations
Example

- known situation 2 days ago and background prediction
- record data for the past 2 days
- minimize the difference between the model and observations
- make predictions for the following day
Example

- known situation 2 days ago and background prediction
- record data for the past 2 days
- minimize the difference between the model and observations
- make predictions for the following day

\[ \rightarrow \text{solve a large-scale non-linear weighted least-squares problem} \]
Solve a large-scale non-linear weighted least-squares problem:

\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} \| x - x_b \|_B^2 + \frac{1}{2} \sum_{j=0}^{N} \| \mathcal{H}_j(\mathcal{M}_j(x)) - y_j \|_{R_j}^2
\]

where

- \( x \equiv x(t_0) \) is the control variable
- \( \mathcal{M}_j \) 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 observations \( y_j \) and the background \( x_b \) are noisy
- \( B \) and \( R_j \) are covariance matrices
Solve a large-scale non-linear weighted least-squares problem:

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

Typically solved using a truncated Gauss-Newton algorithm (known as 4D-Var in the DA community).
Solve a large-scale non-linear weighted least-squares problem:

\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} \| x - x_b \|_B^{-1} + \frac{1}{2} \sum_{j=0}^{N} \| \mathcal{H}_j(\mathcal{M}_j(x)) - y_j \|_{R_j}^{-1}
\]

Typically solved using a truncated Gauss-Newton algorithm (known as 4D-Var in the DA community).

\[\rightarrow \text{linearize} \quad \mathcal{H}_j(\mathcal{M}_j(x^{(k)} + \delta x^{(k)})) \approx \mathcal{H}_j(\mathcal{M}_j(x^{(k)})) + H_j^{(k)} \delta x^{(k)}\]

\[\rightarrow \text{solve the linearized subproblem} \]

\[
\min_{\delta x^{(k)} \in \mathbb{R}^n} \frac{1}{2} \| \delta x^{(k)} - (x_b - x^{(k)}) \|_B^{-1} + \frac{1}{2} \| H^{(k)} \delta x^{(k)} - d^{(k)} \|_{R^{-1}}^{-1}
\]

\[\rightarrow \text{update} \quad x^{(k+1)} = x^{(k)} + \delta x^{(k)}\]
Optimality conditions for the linear subproblem:

\[(B^{-1} + H^T R^{-1} H)x = B^{-1}(x_b - x) + H^T R^{-1} d\]

Algorithmic considerations:

- **very large** problem size: \( m \sim 10^7 \) observations, \( n \sim 10^8 \) unknowns
Optimality conditions for the linear subproblem:

\[
\begin{align*}
(B^{-1} + H^T R^{-1} H) x &= B^{-1} (x_b - x) + H^T R^{-1} d \\
A x &= b
\end{align*}
\]

Algorithmic considerations:

- very large problem size: \( m \sim 10^7 \) observations, \( n \sim 10^8 \) unknowns
- iterative methods:
  - reorthogonalization?
Optimality conditions for the linear subproblem:

\[
\begin{align*}
(B^{-1} + H^T R^{-1} H) x &= B^{-1}(x_b - x) + H^T R^{-1} d \\
A &\quad b
\end{align*}
\]

Algorithmic considerations:

- **very large** problem size: \( m \sim 10^7 \) observations, \( n \sim 10^8 \) unknowns
- iterative methods:
  - reorthogonalization?
  - primal/dual formulations?  
  - preconditioning?  
- **very few** iterations performed
- **noisy data**: need **reliable** error estimates
Optimality conditions for the linear subproblem:

\[
(B^{-1} + H^T R^{-1} H) x = B^{-1}(x_b - x) + H^T R^{-1} d
\]

In this talk:

- Derivation of an optimal algorithm for the case where there are less observations than estimated variables
- A multigrid algorithm
  \(\rightarrow\) how to manage multi-fidelity?
Introduction to data assimilation

Dual iterative solvers

Multigrid solvers

Summary and ongoing related work
Defining the constraint as $\varepsilon = H\delta x - d$ we can write Lagrangian function for the problem as

$$
\mathcal{L}(\delta x, \varepsilon, \lambda) = \frac{1}{2}\|x + \delta x - x_b\|_B^{-1} + \frac{1}{2}\|\varepsilon\|_R^{-1} + \lambda^T(\varepsilon - H(\delta x) + d)
$$

From Karush-Kuhn-Tucker conditions, the following stationary conditions are satisfied at any optimum:

$$
\nabla_{\delta x}\mathcal{L}(\delta x, \varepsilon, \lambda) = B^{-1}(x + \delta x - x_b) - H^T\lambda = 0
$$

$$
\nabla_{\varepsilon}\mathcal{L}(\delta x, \varepsilon, \lambda) = R^{-1}\varepsilon + \lambda = 0
$$

$$
\nabla_{\lambda}\mathcal{L}(\delta x, \varepsilon, \lambda) = \varepsilon - H\delta x + d = 0
$$
In a matrix form:

\[
\begin{bmatrix}
B^{-1} & 0 & -H^T \\
0 & R^{-1} & I \\
-H & I & 0
\end{bmatrix}
\begin{bmatrix}
\delta x \\
\varepsilon \\
\lambda
\end{bmatrix} =
\begin{bmatrix}
B^{-1}(x_b - x) \\
0 \\
-d
\end{bmatrix},
\]

after elimination:

\[
\begin{bmatrix}
I & 0 & -BH^T \\
0 & I & R \\
0 & I & -HBH^T
\end{bmatrix}
\begin{bmatrix}
\delta x \\
\varepsilon \\
\lambda
\end{bmatrix} =
\begin{bmatrix}
x_b - x \\
0 \\
-d + H(x_b - x)
\end{bmatrix}
\]
Therefore, the solution can be written as

$$\lambda = (HBH^T + R)^{-1}(d - H(x_b - x))$$
$$\delta x = x_b - x + BH^T \lambda$$
$$\varepsilon = -R\lambda$$

We therefore obtain

$$\delta x = x_b - x + BH^T (HBH^T + R)^{-1}(d - H(x_b - x)).$$
Exploiting the structure: Dual Approach

- Alternatively, the **exact solution** can be rewritten from duality theory or using Sherman-Morrison-Woodbury formula

\[
x_b - x_k + B H_k^T \left( (H_k B H_k^T + R)^{-1} (d_k - H_k (x_b - x_k)) \right)
\]

Lagrange mult.: requires solving a linear system **iteratively** in \( \mathbb{R}^m \)
Alternatively, the exact solution can be rewritten from duality theory or using Sherman-Morrison-Woodbury formula

\[ x_b - x_k + BH_k^T \left( H_k BH_k^T + R \right)^{-1} \left( d_k - H_k (x_b - x_k) \right) \]

Lagrange mult. : requires solving a linear system iteratively in \( \mathbb{R}^m \)

If \( m << n \), then performing the minimization in \( \mathbb{R}^m \) can reduce memory and computational cost.
Alternatively, the **exact solution** can be rewritten from duality theory or using Sherman-Morrison-Woodbury formula:

\[ x_b - x_k + BH_k^T \left( H_k B H_k^T + R \right)^{-1} (d_k - H_k (x_b - x_k)) \]

Lagrange mult. : requires solving a linear system **iteratively** in \( \mathbb{R}^m \).

If \( m << n \), then performing the minimization in \( \mathbb{R}^m \) can **reduce memory and computational cost**.
**Preconditioned CG algorithm**

### Initialization
- \( r_0 = A\delta x_0 - b, \ z_0 = Fr_0, \ p_0 = z_0 \)

### For \( i = 0, 1, ... \)

1. \( q_i = (B^{-1} + H^T R^{-1} H)p_i \)
2. \( \alpha_i = \langle r_i, z_i \rangle / \langle q_i, p_i \rangle \)
   - Compute the step-length
3. \( \delta x_{i+1} = \delta x_i + \alpha_i p_i \)
   - Update the iterate
4. \( r_{i+1} = r_i - \alpha_i q_i \)
   - Update the residual
5. \( r_{i+1} = r_{i+1} - RZ^T r_{i+1} \)
   - Re-orthogonalization
6. \( z_{i+1} = F r_{i+1} \)
   - Update the preconditioned residual
7. \( \beta_i = \langle r_{i+1}, z_{i+1} \rangle / \langle r_i, z_i \rangle \)
   - Ensure A-conjugate directions
8. \( R = [R, r/\beta_i] \)
   - Re-orthogonalization
9. \( Z = [Z, z/\beta_i] \)
   - Re-orthogonalization
10. \( p_{i+1} = z_{i+1} + \beta_i p_i \)
    - Update the descent direction
Theorem

Suppose that

1. \( BH^T G = FH^T \).
2. \( v_0 = x^b - x_0 \).

→ vectors \( \hat{r}_i, \hat{p}_i, \hat{v}_i, \hat{z}_i \) and \( \hat{q}_i \) such that

\[
\begin{align*}
    r_i &= H^T \hat{r}_i, \\
    p_i &= BH^T \hat{p}_i, \\
    v_i &= v_0 + BH^T \hat{v}_i, \\
    z_i &= BH^T \hat{z}_i, \\
    q_i &= H^T \hat{q}_i
\end{align*}
\]
Initialization steps

Given \( v_0; \ r_0 = (H^T R^{-1} H + B^{-1}) v_0 - b, \ldots \)

Loop: WHILE

<table>
<thead>
<tr>
<th>Step</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( H^T \hat{q}<em>{i-1} = H^T (R^{-1} HB^{-1} H^T + I_m) \hat{p}</em>{i-1} )</td>
</tr>
<tr>
<td>2</td>
<td>( \alpha_{i-1} = r_{i-1}^T z_{i-1} / \hat{q}<em>{i-1}^T \hat{p}</em>{i-1} )</td>
</tr>
<tr>
<td>3</td>
<td>( BH^T \hat{v}<em>i = BH^T (v</em>{i-1} + \alpha_{i-1} \hat{p}_{i-1}) )</td>
</tr>
<tr>
<td>4</td>
<td>( H^T \hat{r}<em>i = H^T (r</em>{i-1} + \alpha_{i-1} \hat{q}_{i-1}) )</td>
</tr>
<tr>
<td>5</td>
<td>( BH^T \hat{z}_i = FH^T \hat{r}_i = BH^T G \hat{r}_i \quad FHT = BHTG )</td>
</tr>
<tr>
<td>6</td>
<td>( \beta_i = (r_{i}^T z_{i} / r_{i-1}^T z_{i-1}) )</td>
</tr>
<tr>
<td>7</td>
<td>( BH^T \hat{p}_i = BH^T (-\hat{z}<em>i + \beta_i \hat{p}</em>{i-1}) )</td>
</tr>
</tbody>
</table>
Initialization

\[ \lambda_0 = 0, \hat{r}_0 = R^{-1}(d - H(x_b - x)), \]
\[ \hat{z}_0 = G\hat{r}_0, \hat{p}_1 = \hat{z}_0, k = 1 \]

Loop on \( k \)

1. \( \hat{q}_i = \hat{A}\hat{p}_i \)
2. \( \alpha_i = \langle \hat{r}_{i-1}, \hat{z}_{i-1} \rangle_M / \langle \hat{q}_i, \hat{p}_i \rangle_M \)
3. \( \lambda_i = \lambda_{i-1} + \alpha_i \hat{p}_i \)
4. \( \hat{r}_i = \hat{r}_{i-1} - \alpha_i \hat{q}_i \)
5. \( \beta_i = \langle \hat{r}_{i-1}, \hat{z}_{i-1} \rangle_M / \langle \hat{r}_{i-2}, \hat{z}_{i-2} \rangle_M \)
6. \( \hat{z}_i = G\hat{r}_i \)
7. \( \hat{p}_i = \hat{z}_{i-1} + \beta_i \hat{p}_{i-1} \)

- \( \hat{A} = R^{-1}HBH^T + I_m \)
- \( G \) is the preconditioner.
- \( M \) is the inner-product.
- RPCG Algorithm: \( M = HBH^T \) preserves monotonic decrease on quadratic cost
- \( G \) should be symmetric w.r.t. to \( M \)
Initialization
\[ \lambda_0 = 0, \hat{r}_0 = R^{-1}(d - H(x_b - x)), \]
\[ \hat{z}_0 = G\hat{r}_0, \hat{p}_1 = \hat{z}_0, k = 1 \]

Loop on \( k \)
1. \( \hat{q}_i = \hat{A}\hat{p}_i \)
2. \( \alpha_i = <\hat{r}_{i-1}, \hat{z}_{i-1}>_M / <\hat{q}_i, \hat{p}_i>_M \)
3. \( \lambda_i = \lambda_{i-1} + \alpha_i \hat{p}_i \)
4. \( \hat{r}_i = \hat{r}_{i-1} - \alpha_i \hat{q}_i \)
5. \( \beta_i = <\hat{r}_{i-1}, \hat{z}_{i-1}>_M / <\hat{r}_{i-2}, \hat{z}_{i-2}>_M \)
6. \( \hat{z}_i = G\hat{r}_i \)
7. \( \hat{p}_i = \hat{z}_{i-1} + \beta_i \hat{p}_{i-1} \)

- \( \hat{A} = R^{-1}HBH^T + I_m \)
- \( G \) is the preconditioner.
- \( M \) is the inner-product.
- RPCG Algorithm: \( M = HBH^T \) preserves monotonic decrease on quadratic cost
- \( G \) should be symmetric w.r.t. to \( M \)
Minimization in dual space

1. **Iteratively** solve

\[(I_m + R^{-1}H_kBH_k^T)\lambda = d\]

2. Set \(\delta x_k = x_b - x_k + BH_k^T\lambda\)
Minimization in dual space

1. **Iteratively** solve

\[
(l_m + R^{-1}H_k B H_k^T) \lambda = d
\]

2. Set \( \delta x_k = x_b - x_k + B H_k^T \lambda \)

- **PSAS** (Courtier 1997): Preconditioned CG (PCG) with \( \tilde{R} \) inner product.

- **RPCG** (Gratton and Tshimanga 2009): PCG with \( H_k B H_k^T \)
  
  → It generates the same iterates as those generated by the **primal approach**.
Dual approach

Minimization in dual space

1. **Iteratively** solve

\[
(l_m + R^{-1}H_kB H_k^T)\lambda = d
\]

2. Set \(\delta x_k = x_b - x_k + B H_k^T \lambda\)

- **PSAS** (Courtier 1997): Preconditioned CG (PCG) with \(\tilde{R}\) inner product.
- **RPCG** (Gratton and Tshimanga 2009): PCG with \(H_kB H_k^T\)
  \[\rightarrow\] It generates the **same iterates** as those generated by the **primal approach**.

\[J(\delta x_k) = \frac{1}{2} \|\delta x_k - x_b + x_k\|_{B^{-1}}^2 + \frac{1}{2} \|H_k\delta x_k - d_k\|_{R^{-1}}^2\]
- **Observations**: SST (Sea Surface Temperature) and SSH (Sea Surface Height) observations from satellites. Sub-surface hydrographic observations from floats.

- **Number of observations (m)**: $10^5$

- **Number of state variables (n)**: $10^6$ for strong constraint and $10^7$ for weak constraint.

- **Computation**: 64 CPUs
It is possible to maintain the one-to-one correspondence between primal and dual iterates, under the assumption that

$$F_{k-1}H_k^T = BH_k^T G_{k-1}$$

where $F_{k-1}$ is a preconditioner for a primal solver and $G_{k-1}$ is a preconditioner for a dual solver (Gratton and Tshimanga 2009).

The preconditioner $G_{k-1}$ needs to be symmetric in $H_kBH_k^T$ inner product.

Solution algorithm: Preconditioned Conjugate Gradient method (PCG)

→ Preconditioning with the quasi-Newton Limited Memory Preconditioner (Morales and Nocedal 2000) (Gratton, Sartenaer and Tshimanga 2011)

For linear case, Gratton, Gurol and Toint (2012) derive the quasi-Newton LMP in dual space which generates mathematically equivalent iterates to those of primal approach.
**F as a Quasi-Newton Limited Memory Preconditioner**

- **Quasi-Newton LMPs** are simply based on the idea that generates preconditioners by using LBFGS updating formula:

\[
F_{k+1} = (I_n - \tau_k p_k q_k^T)F_k(I_n - \tau_k q_k p_k^T) + \tau_k p_k p_k^T
\]

- \(\tau_k = 1/(q_k^T p_k)\)
- \(q_k = (B^{-1} + H^T R^{-1} H)p_k\)  \(\text{The pairs}\)

- \(\Delta F_k\) defined by \(\Delta F_k = F_{k+1} - F_k\), is the optimal solution to the problem:

\[
\min_{\Delta F_k} \left\| W^{1/2} \Delta F_k W^{1/2} \right\|_F
\]

subject to \(\Delta F_k = \Delta F_k^T, \quad F_{k+1} q_k = p_k\)

where \(W\) is any symmetric positive definite matrix satisfying \(Wp_k = q_k\)
G as a Quasi-Newton Limited Memory Preconditioner

- Giving $F$, we can find $G$ that satisfies $FH^T = BH^T G$
- $G$ can be derived by using the formula for $F$ using the relations

  \[ p_i = BH^T \hat{p}_i, \quad q_i = H^T \hat{q}_i \]

\[
G_{k+1} = (I_m - \hat{\tau}_k \hat{p}_k (M \hat{q}_k)^T) G_k (I_m - \hat{\tau}_k \hat{q}_k \hat{p}_k^T M) + \hat{\tau}_k \hat{p}_k \hat{p}_k^T M
\]

- $M = HBH^T$
- $\hat{p}_k$ is the search direction
- $\hat{q}_k = (I_m + R^{-1}HBH^T) \hat{p}_k$ \textit{The dual pairs}
- $\hat{\tau}_k = 1/(\hat{q}_k^T HBH^T \hat{p}_k)$
$\Delta G_k$ defined by $\Delta G_k = G_{k+1} - G_k$ is the optimum solution to the minimization problem defined as:

$$\min_{\Delta G_k} \left\| W^{1/2} M^{1/2} \Delta G_k M^{-1/2} W^{1/2} \right\|_F$$

subject to $M \Delta G_k = \Delta G_k^T M$, $G_{k+1} \hat{q}_k = \hat{p}_k$

The norm in this problem is considered as a weighted Frobenius norm, where $W$ is any symmetric positive definite matrix satisfying $W M^{1/2} \hat{p}_k = M^{1/2} \hat{q}_k$ and $M = HBH^T$. 
Preconditioning in dual space

- If $FA$ has eigenvalues $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$, PCG algorithm with zero initial starting vector satisfies the inequality:

$$\|x_{k+1} - x^*\|_A \leq 2\left(\frac{\sqrt{\mu_n} - \sqrt{\mu_1}}{\sqrt{\mu_n} + \sqrt{\mu_1}}\right)^k \|x^*\|_A$$

- When RPCG is used, the iterates are belongs to the affine subspace of $x_0 + \text{Im}(BH^T)$.

- If $GA$ has eigenvalues $\nu_1 \leq \nu_2 \leq \ldots \leq \nu_m$, Restricted PCG (version 3) with zero initial starting vector satisfies the inequality:

$$\|x_{k+1} - x^*\|_A \leq 2\left(\frac{\sqrt{\nu_m} - \sqrt{\nu_1}}{\sqrt{\nu_m} + \sqrt{\nu_1}}\right)^k \|x^*\|_A$$

where $A = B^{-1} + H^T R^{-1} H$ and $\hat{A} = I + R^{-1} HBH^T$
**Preconditioning in dual space**

- \(FABH^T = BH^T G \hat{A}\). Therefore, \(BH^T\) is an invariant subspace of \(FA\).
- Every eigenvalue of \(G \hat{A}\) is an eigenvalue of \(FA\). So, \(\mu_1 \leq \nu_1\) and \(\mu_n \geq \nu_n\).

If \(FA\) has eigenvalues \(\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n\) and \(G \hat{A}\) has eigenvalues \(\nu_1 \leq \nu_2 \leq \ldots \leq \nu_m\), RPCG with zero initial starting vector satisfies the inequality:

\[
\|x_{k+1} - x^*\|_A \leq 2\left(\frac{\sqrt{\nu_m} - \sqrt{\nu_1}}{\sqrt{\nu_m} + \sqrt{\nu_1}}\right)^k \|x^*\|_A \leq 2\left(\frac{\sqrt{\mu_n} - \sqrt{\mu_1}}{\sqrt{\mu_n} + \sqrt{\mu_1}}\right)^k \|x^*\|_A
\]

Happy breakdown of \(m < n\)
The modified (G-S) orthogonalization scheme writes

\[ r_i \leftarrow \prod_{j=1}^{i-1} \left( I_n - \frac{r_j r_j^T}{r_j^T F r_j} \right) r_i. \]

We suggest the following re-orthogonalization scheme

\[ \hat{r}_i \leftarrow \prod_{j=1}^{i-1} \left( I_m - \frac{\hat{r}_j w_j^T}{\hat{r}_j^T w_j} \right) \hat{r}_i. \]
Preconditioning in dual space

Iterations $i$

Cost function $J[v_i]$

$n = 200$                         $m = 40$

Algorithm 5
Algorithm 5 + ortho
Algorithm 3
Algorithm 3 + ortho

Figure: Orthogonalization issues.

$Serge G. \ (CERFACS)$

Linearizing CG

CAS 2014 30 / 66
The quasi-Newton LMP in dual space (Linear case)

- The **quasi-Newton LMP**: The descent directions $p_i, i = 1, ..., l$ generated by a CG method are used.
The quasi-Newton LMP: The descent directions $p_i, i = 1, ..., l$ generated by a CG method are used.

$$F_i = \left( I_n - \frac{p_i p_i^T A}{p_i^T A p_i} \right) F_{i-1} \left( I_n - \frac{A p_i p_i^T}{p_i^T A p_i} \right) + \frac{p_i p_i^T}{p_i^T A p_i},$$

$\rightarrow F = F_l.$
The quasi-Newton LMP: The descent directions \( p_i, i = 1, \ldots, l \) generated by a CG method are used.

\[
F_i = \left( I_n - \frac{p_ip_i^T A}{p_i^T Ap_i} \right) F_{i-1} \left( I_n - \frac{Ap_ip_i^T}{p_i^T Ap_i} \right) + \frac{p_ip_i^T}{p_i^T Ap_i},
\]

\[\Rightarrow F = F_l.\]

The corresponding quasi-Newton preconditioner in dual space is given as

\[
G_i = \left( I_m - \frac{\hat{p}_i\hat{p}_i^T \hat{A}C}{\hat{p}_i^T \hat{A}C \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A}\hat{p}_i\hat{p}_i^T C}{\hat{p}_i^T \hat{A}C \hat{p}_i} \right) + \frac{\hat{p}_i\hat{p}_i^T C}{\hat{p}_i^T \hat{A}C \hat{p}_i},
\]

where \( i = 1, \ldots, l, \ C = HBH^T, \ \hat{A} = I_m + R^{-1}HBH^T \) and \( \hat{p}_i \) is the search direction.

\[\Rightarrow G = G_l.\]
The quasi-Newton LMP in dual space (Linear case)

- **The quasi-Newton LMP**: The descent directions $p_i$, $i = 1, ..., l$ generated by a CG method are used.

\[ F_i = \left( I_n - \frac{p_ip_i^T A}{p_i^T Ap_i} \right) F_{i-1} \left( I_n - \frac{Ap_ip_i^T}{p_i^T Ap_i} \right) + \frac{p_ip_i^T}{p_i^T Ap_i}, \]

\[ \rightarrow F = F_l. \]

- The corresponding quasi-Newton preconditioner in dual space is given as

\[ G_i = \left( I_m - \frac{\hat{p}_i\hat{p}_i^T \hat{A}C}{\hat{p}_i^T \hat{A}C \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A}p_ip_i^T C}{\hat{p}_i^T \hat{A}C \hat{p}_i} \right) + \frac{\hat{p}_i\hat{p}_i^T C}{\hat{p}_i^T \hat{A}C \hat{p}_i}, \]

where $i = 1, ..., l$, $C = HBH^T$, $\hat{A} = I_m + R^{-1}HBH^T$ and $\hat{p}_i$ is the search direction.

\[ \rightarrow G = G_l. \]

\[ \rightarrow \text{This preconditioner satisfies the relation: } FH^T = BH^T G \text{ and it is symmetric in the } C \text{ inner product (Gratton, Gurol and Toint 2012).} \]
Preconditioning in dual space

The quasi-Newton LMP in dual space (Nonlinear case)

- For **nonlinear case**, inheriting the previous preconditioner may not be possible!

![Cost function graph](image)

$n = 196 \quad m = 64$
$\text{itermax} = 10 \quad \text{Imem} = 10$

- PCG with quasi-Newton LMP
- RPCG with quasi-Newton LMP

Iterations

Cost function
For **nonlinear case**, inheriting the previous preconditioner may not be possible!

- Why?
For **nonlinear case**, inheriting the previous preconditioner may not be possible!

- Why?
  - The **augmented matrix** \( G_k \) **changes** along the outer iterations \( (k) \) because of the nonlinearity.
The quasi-Newton LMP in dual space (Nonlinear case)

- For **nonlinear case**, inheriting the previous preconditioner may not be possible!

  - Why?
    - The **augmented matrix** $G_k$ **changes** along the outer iterations ($k$) because of the nonlinearity.
    - This causes **loss of symmetry** for the preconditioner $G_{k-1}$ with respect to the current inner product $H_k B H_k^T$. 

![Graph showing cost function vs iterations for different methods.](Image)
The quasi-Newton LMP in dual space (Nonlinear case)

Solution:
- **Re-generate** the pairs and the preconditioner using the current inner product.
Solution:

- **Re-generate** the pairs and the preconditioner using the current inner product.

\[
G_i = \left( I_m - \frac{\hat{p}_i \hat{p}_i^T \hat{A} C}{\hat{p}_i^T \hat{A} C \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A} \hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i} \right) + \frac{\hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i}
\]

where \(i = 1, \ldots, l\), \(C = HBH^T\), \(\hat{A} = I_m + R^{-1}HBH^T\) and \(\hat{p}_i\) is the search direction.
The quasi-Newton LMP in dual space (Nonlinear case)

Solution: 

- **Re-generate** the pairs and the preconditioner using the current inner product.

\[
G_i = \left( I_m - \frac{\hat{p}_i \hat{p}_i^T \hat{A}}{\hat{p}_i^T \hat{A} \hat{C} \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A} \hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i} \right) + \frac{\hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i}
\]

where \( i = 1, \ldots, l, \ C = HBH^T, \ \hat{A} = I_m + R^{-1} HBH^T \) and \( \hat{p}_i \) is the search direction.

→ It is **costly** for large-scale problems.
The quasi-Newton LMP in dual space (Nonlinear case)

Solution:

- **Re-generate** the pairs and the preconditioner using the current inner product.

\[
G_i = \left( I_m - \frac{\hat{p}_i \hat{p}_i^T \hat{A} \hat{C}}{\hat{p}_i^T \hat{A} \hat{C} \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A} \hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} \hat{C} \hat{p}_i} \right) + \frac{\hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} \hat{C} \hat{p}_i}
\]

where \( i = 1, \ldots, l \), \( C = HBH^T \), \( \hat{A} = I_m + R^{-1} HBH^T \) and \( \hat{p}_i \) is the search direction.

→ **It is costly** for large-scale problems.

- **Define a criterion** on whether we precondition the system or not by using a measure on symmetry.

→ Sensitive to the threshold value.
The quasi-Newton LMP in dual space (Nonlinear case)

Solution:

- **Re-generate** the pairs and the preconditioner using the current inner product.

\[
G_i = \left( I_m - \frac{\hat{p}_i \hat{p}_i^T \hat{A} C}{\hat{p}_i^T \hat{A} C \hat{p}_i} \right) G_{i-1} \left( I_m - \frac{\hat{A} \hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i} \right) + \frac{\hat{p}_i \hat{p}_i^T C}{\hat{p}_i^T \hat{A} C \hat{p}_i}
\]

where \( i = 1, \ldots, l \), \( C = HBH^T \), \( \hat{A} = I_m + R^{-1} HBH^T \) and \( \hat{p}_i \) is the search direction.

→ It is **costly** for large-scale problems.

- **Define a criterion** on whether we precondition the system or not by using a measure on symmetry.

  → Sensitive to the threshold value.

**Objective:**

- A **robust algorithm** that handles this sensitivity
- A **globally convergent algorithm**
- Most cases are not highly nonlinear : → **Perturb as little as possible the preconditioner of the linear case, and check a posteriori**
Global convergence can be ensured by inserting the Gauss-Newton strategy in a trust region framework.

Trust-region method simply solves the following problem at iteration $k$:

$$\min_{\delta x_k \in \mathbb{R}^n} J(\delta x_k) = \frac{1}{2} \|\delta x_k - x_b + x_k\|_{B^{-1}}^2 + \frac{1}{2} \|H_k \delta x_k - d_k\|_{R^{-1}}^2$$

subject to $\|\delta x_k\|_{F_{k}^{-1}} \leq \Delta_k$ (primal approach)

where $\Delta_k$ is the trust region radius.
The preconditioner $\tilde{G}_{k-1}$ that is inherited from previous iteration may **not be symmetric in the current inner product** and may **not be positive-definite in the full dual space** (merely in one direction).
The preconditioner $\tilde{G}_{k-1}$ that is inherited from previous iteration may not be symmetric in the current inner product and may not be positive-definite in the full dual space (merely in one direction).

We need to adapt the strategy in the trust region algorithm.
Trust-region in dual space

- The preconditioner $\tilde{G}_{k-1}$ that is inherited from previous iteration may not be symmetric in the current inner product and may not be positive-definite in the full dual space (merely in one direction).

- We need to adapt the strategy in the trust region algorithm.

TR Step calculation with a flexible (Steihaug) RPCG algorithm

1. Check the positive-definiteness along the steepest descent direction.
2. Compute the Cauchy step
3. Compute the step beyond the Cauchy step with the RPCG algorithm (ignoring symmetry problem)
4. Backtrack along the CG path if needed
The preconditioner $\tilde{G}_{k-1}$ that is inherited from previous iteration may not be symmetric in the current inner product and may not be positive-definite in the full dual space (merely in one direction).

We need to adapt the strategy in the trust region algorithm.

---

**TR Step calculation with a flexible (Steihaug) RPCG algorithm**

1. Check the positive-definiteness along the steepest descent direction.
2. Compute the Cauchy step.
3. Compute the step beyond the Cauchy step with the RPCG algorithm (ignoring symmetry problem).
4. Backtrack along the CG path if needed.
Trust-region in dual space

Flexible trust region algorithm

1. Initialization
2. Compute the step by the flexible (Steihaug) RPCG algorithm
3. Accept the step beyond the Cauchy step if
   \[ f(y_k) < f(x_k^C) \]
4. Accept the trial point according to the ratio of achieved to predicted reduction
5. Update the trust region

→ The global convergence can be proved!
Globally convergent algorithm in dual space

Trust-region in dual space

Flexible trust region algorithm

1. Initialization
2. Compute the step by the flexible (Steihaug) RPCG algorithm
3. Accept the step beyond the Cauchy step if
   \[ f(y_k) < f(x_k^C) \]
4. Accept the trial point according to the ratio of achieved to predicted reduction
5. Update the trust region

→ The global convergence can be proved!

→ This approach is similar to the approach that computes the magical step proposed by (Conn, Gould, Toint 2000).
The dynamical model is considered to be the nonlinear heat equation defined by

\[
\delta x \frac{\delta x}{\delta t} - \delta^2 x \frac{\delta^2 x}{\delta u^2} - \delta^2 x \frac{\delta^2 x}{\delta v^2} + f[x] = 0 \text{ in } \Omega \times (0, \infty)
\]

\[
x[u, v, t] = 0 \text{ on } \delta \Omega \times (0, \infty)
\]

where the temperature variable \(x[u, v, t]\) depend on both time \(t\) and position given by spatial coordinates \(u\) and \(v\). The function \(f[x]\) is defined by

\[
f[x] = \exp[\eta x]
\]
Numerical experiment on heat equation (2/3)

\[ f[x] = \exp[\eta x] \quad \eta = 2 \]

- \( n = 196 \)
- \( m = 64 \)
- \( \text{itermax} = 10 \)
- \( \text{Imem} = 10 \)

Graph showing the cost function against iterations for different methods:
- PCG with quasi-Newton LMP
- RPCG with quasi-Newton LMP
- PCG without preconditioner
Numerical Results

Numerical experiment on heat equation (3/3)

\[ f(x) = \exp[\eta x] \quad \eta = 4.2 \]

Linearizing CG

- $n = 196$
- $m = 64$
- $\text{itermax} = 10$
- $\text{Imem} = 10$

![Graph showing cost function vs iterations for different methods. The graph compares PCG with quasi-Newton LMP, PCG without preconditioning, and RPCG with quasi-Newton LMP. The cost function decreases as the iterations increase, with each method showing a different rate of convergence.](image-url)
• Introduction to data assimilation
• Dual iterative solvers
• Multigrid solvers
• Summary and ongoing related work
First order necessary optimality condition

\[ \nabla_x f(x^*) = 0. \]

Then \( x^* \) is referred to as first-order critical point of \( f \)

System of nonlinear equations can be solved using:
- Newton’s iterations with a multigrid linear solver,
- A multigrid nonlinear solver, the full approximation scheme
- Problem: obtaining a global convergence for strongly nonlinear problems

In general, the optimization point of view is interesting: globalization techniques ensure a global convergence to first order critical points

Use the trust-region mechanism to handle multi-level problems and keep the global convergence property
Basic trust-region algorithm (Conn, Gould, Toint (2000))

Until convergence

- Choose a local model $m_k$ of $f$ around $x_k$
- Compute a trial point $x_k + s_k$ that decreases this model within the trust region
- Compute the predicted reduction $m_k(x_k) - m_k(x_k + s_k)$
- Evaluate change in the objective function $f(x_k) - f(x_k + s_k)$
- If achieved change $\sim$ predicted reduction then
  - Accept trial point as new iterate $x_{k+1} = x_k + s_k$
  - Possibly enlarge the trust region
  - Reject the trial point $x_{k+1} = x_k$
  - Shrink the trust region

Serge G. (CERFACS)
Ingredients of a basic convergence theory

On the function. On $\mathbb{R}^n$,

- Assume $f$ is twice-continuously differentiable, $f$ is bounded below, $\|\nabla_{xx} f\|$ is bounded above

On the model. On the trust region $B_k = \{x_k + s, \|s\| \leq \Delta_k\}$,

- Assume that $m_k$ is twice-continuously differentiable, $m_k(x_k) = f(x_k)$, $\nabla f(x_k) = \nabla m_k(x_k)$, $\|\nabla_{xx} m_k\|$ is bounded above uniformly in $k$

Assume that the step $s$ provides a sufficient decrease of $m_k$ (e.g. it has a better decrease than the Cauchy step obtain by minimizing $m_k(x)$ for $x = x_k - t\nabla f(x_k) \in B_k$)

The algorithm converges to a first order critical point of $f$
TR mechanism

Trust-region method

x

y
TR mechanism
TR mechanism
TR mechanism
TR mechanism
TR mechanism
**Multigrid setting**

- Level parameter \( i, x \in \mathbb{R}^{n_i} \). Particular levels \( r \) finest, 0 coarsest.

<table>
<thead>
<tr>
<th>Level ( r )</th>
<th>( \mathbb{R}^n )</th>
<th>( x_r ) ( f_r = f ) ( h_r = f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
</tr>
<tr>
<td>Level ( i + 1 )</td>
<td>( \mathbb{R}^{n_{i+1}} )</td>
<td>( x_{i+1} ) ( f_{i+1} ) ( h_{i+1} )</td>
</tr>
<tr>
<td>( R_{i+1} \downarrow \uparrow P_{i+1} )</td>
<td>( \mathbb{R}^{n_i} )</td>
<td>( x_i ) ( f_i ) ( h_i )</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
</tr>
<tr>
<td>Level 0</td>
<td>( \mathbb{R}^{n_0} )</td>
<td>( x_0 ) ( f_0 ) ( h_0 )</td>
</tr>
</tbody>
</table>

The functions \( f_i \) represent \( f \) on the coarse spaces, it is e.g. the discretization of \( f \) on the coarse space.
The functions \( h_i \) are modifications of the \( f_i \)'s to ensure inter-level coherence. The prolongations and restrictions are full rank, linear and satisfy \( P_{i+1} = R_{i+1}^T \) (up to a scalar).
Performing the recursion: 2-level scheme

- Scaled norm for the trust region: \( \| P_{i+1} s \|_{i+1} \overset{\text{def}}{=} \| s \|_i \)

- Trust-region constraint preservation based on

\[
\| x_{i+1,k+1} - x_{i+1,k} \|_{i+1} = \| P_{i+1} (x_{i,0} - x_{i,*}) \|_{i+1} = \| x_{i,0} - x_{i,*} \|_i \\
\leq \sum_{k=1}^{*} \| x_{i,k-1} - x_{i,k} \|_i
\]
First order coherence between levels

- Two successive levels parameters $i$ and $i + 1$. For the iterate $x_{i+1,k}$ set $x_{i,0} = R_{i+1}x_{i+1}$. The immediate coarse model is defined by $h_i(x_{i,0} + s) = f_i(x_{i,0} + s) + < v_i, s >$, where $v_i = R_{i+1} \nabla_x h_{i+1}(x_{i+1,k}) - \nabla_x f_i(x_{i,0})$. Therefore $\nabla_x h_i(x_{i,0}) = R_{i+1} \nabla_x h_{i+1}(x_{i+1,k})$

- Linear coherence

\[
\begin{align*}
  h_{i+1}(x_{i+1,k} + P_{i+1}s) &= h_{i+1}(x_{i+1,k}) + \\
  &\quad< R_{i+1} \nabla_x h_{i+1}(x_{i+1,k}), s > + o(s) \\
  h_i(x_{i,0} + s) &= h_i(x_{i,0}) + \\
  &\quad< \nabla_x h_i(x_{i,0}), s > + o(s)
\end{align*}
\]

- Recursion useful only if $\| R_{i+1} \nabla_x h_{i+1}(x_{i+1,k}) \| \geq \kappa \| \nabla h_{i+1}(x_{i+1,k}) \|$

- Linear correction: similar to the full approximation scheme!
Recursive multi-scale algorithm

Until convergence

- Choose either a Taylor or a (useful) recursive model
  - Taylor model: compute a Taylor step satisfying a sufficient decrease property
  - Recursive: apply the algorithm recursively
- Evaluate change in the objective function
- If achieved change $\sim$ predicted reduction then
  - Accept trial point as new iterate
  - Possibly enlarge the trust region
else
  - Reject the trial point
  - Shrink the trust region

The algorithm is proved globally convergent to first order critical points
Theoretical results

- The algorithm is proved globally convergent to first order critical points:
  \[ \lim_{i \to +\infty} \nabla f(x_i) = 0. \]

- The worse case complexity of the algorithm to get a gradient less than \( \epsilon \) is \( O(1/\epsilon^2) \) steps.
Computing the step providing sufficient decrease

Approximatively solve (suff. decrease) the local TR subproblem
\[
\min_{\|s\|_i \leq \Delta} Q(s) = \frac{1}{2} < Hs, s > + < g, s >
\]

- Steihaug truncated CG
- Exact Moré-Sorensen Method on small dimension spaces
- Take advantage of the good smoothing properties of linear Gauss-Seidel
  - Compute \( s_0 \) by minimizing along the largest gradient component
  - Perform some Gauss-Seidel cycles (minimization along coordinate axes) to obtain \( s_1 \)
  - Take \( s_1 \) if \( \|s_1\|_i \leq \Delta \)
  - Else, if \( s_1 \) is gradient related \( < g, s_1 > \leq \kappa \|s_1\| \|g\| \), backtrack
  - Else minimize \( Q(s) \) for \( \|s\|_i \leq \Delta \) on the path \([0, s_0, s_1]\)
Multigrid

Computing the step providing sufficient decrease

Approximatively solve (suff. decrease) the local TR subproblem
\[ \min_{\|s\|_i \leq \Delta} Q(s) = \frac{1}{2} < Hs, s > + < g, s > \]

- Steihaug truncated CG
- Exact Moré-Sorensen Method on small dimension spaces
- Take advantage of the good smoothing properties of linear Gauss-Seidel
  - Compute \(s_0\) by minimizing along the largest gradient component
  - Perform some Gauss-Seidel cycles (minimization along coordinate axes) to obtain \(s_1\)
  - Take \(s_1\) if \(\|s_1\|_i \leq \Delta\)
  - Else, if \(s_1\) is gradient related \((< g, s_1 > \leq \kappa \|s_1\| \|g\|)\), backtrack
  - Else minimize \(Q(s)\) for \(\|s\|_i \leq \Delta\) on the path \([0, s_0, s_1]\)
Some important algorithmic options

Matlab implementation: code *rmtr*

- Perform a linesearch on $s_{i,k}$
- Type of cycling between levels (V, W,...)
- Number of Gauss-Seidel smoothing cycles

Comparison of a the pure multilevel solution with the mesh refinement (MR) defined by:
- The problem at level $i$ is solved using Taylor iterations at level $i$ only
- The solution prolonged from level $i$ to $i+1$ is the starting point for the solution at level $i+1$
Elliptic problems

- Variational approach (Lax-Milgram) leads to solving
  \[ \min_v a(v, v) - b(v) \]

- Experiments performed on the model problem

- The algorithm exhibits similar performance as the classical linear multigrid: similar convergence factor, number of smoothing cycles constant when adding finer levels
Minimum surface problem

Solve

\[
\min_x \int \int \sqrt{1 + (\partial_s x)^2 + (\partial_t x)^2} \, ds \, dt
\]

on the square \[\{(s, t) \mid 0 \leq s \leq 1, 0 \leq t \leq 1\}\]. The boundary conditions are

\[
x(t, s) = \begin{cases} 
  f(s), & t = 0, 0 \leq s \leq 1 \\
  0, & s = 0, 0 \leq t \leq 1 \\
  f(s), & t = 1, 0 \leq s \leq 1 \\
  0, & s = 1, 0 \leq t \leq 1 
\end{cases}
\]

where \( f(t) = \sin(4\pi t) + \frac{1}{10} \sin(120\pi t) \)

Highly nonlinear problem!
Minimum surface problem (discretization)

Discretization by finite differences of the operators $\partial_s x$ and $\partial_s t$

The number of variables in level $i$ is $n_i = (2^{k+2} - 1)^2$ ($= 3^2, 7^2, \ldots$) The prolongation operator from $\mathbb{R}^{n_i}$ to $\mathbb{R}^{n_{i+1}}$ writes $J \otimes J$, the restriction operator is $1/\|J\|^2 \cdot J^T \otimes J^T$, where $J$ is defined by

$$J^T = \begin{pmatrix} 1/2 & 1 & 1/2 & 0 & \cdots & 0 \\ 0 & 0 & 1/2 & 1 & 1/2 & 0 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \end{pmatrix} \in \mathbb{R}^{\sqrt{n_i} \times \sqrt{n_{i+1}}}$
The solution of the problem

Minimum surface solution
Numerical illustrations

All on finest

1000 Matrix-Vector products
\[ \|g\| = 1.7 \times 10^{-2} \]

1700 Matrix-Vector products
\[ \|g\| = 1.6 \times 10^{-2} \]
Comparison **mesh refinement** vs **multigrid** ($\|g\| \leq 10^{-7}$)

<table>
<thead>
<tr>
<th>Problem size</th>
<th>$127^2$</th>
<th>$255^2$</th>
<th>$511^2$</th>
<th>$1023^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hessian (finest)</td>
<td>11</td>
<td>8</td>
<td>24</td>
<td>16</td>
</tr>
<tr>
<td>Gradient (finest)</td>
<td>11</td>
<td>32</td>
<td>27</td>
<td>40</td>
</tr>
<tr>
<td>Function (finest)</td>
<td>15</td>
<td>40</td>
<td>30</td>
<td>70</td>
</tr>
<tr>
<td>Taylor it / Cycles (finest)</td>
<td>1664</td>
<td>56</td>
<td>2693</td>
<td>66</td>
</tr>
<tr>
<td>Lin. alg. work (Gflops)</td>
<td>0.78</td>
<td>0.06</td>
<td>3.19</td>
<td>0.31</td>
</tr>
<tr>
<td>Matlab elapsed time (s)</td>
<td>22</td>
<td>19</td>
<td>142</td>
<td>111</td>
</tr>
</tbody>
</table>
Introduction to data assimilation

Dual iterative solvers

Multigrid solvers

Summary and ongoing related work
Conclusion

Conclusions

- Very large scale highly-nonlinear problems can be solved using projection methods, multigrid ideas, domain decomposition adapted to nonlinearity.

- Gradient information has to be used, preconditioning is needed for inverse problems. Not the same as for linear systems.

- In some severe cases, having "more" than a local minimum might be needed. Model reduction and exploration techniques are useful.

- Further work: consider stochastic dynamical systems: weak constrained problem. Errors between time-steps correlated. Having a parallel code is a challenge.
References


