Numerical solution of the time-parallelized weak-constraint 4DVar

Selime Gürol, CERFACS, Toulouse, France

joint work with Mike Fisher (ECMWF) and Serge Gratton (CERFACS/IRIT)

CIMI Workshop
Advances in optimization with application to data assimilation

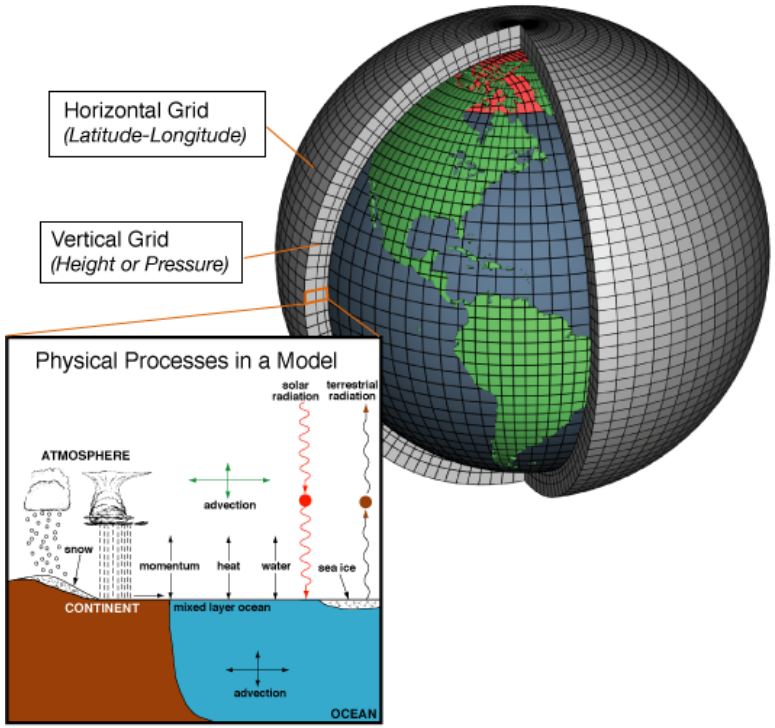
13 January 2016

Outline

- 4D-Var data assimilation problem
- Saddle point approach of 4D-Var
- Preconditioning of saddle point formulation
- Numerical results
- Conclusions

4D-Variational data assimilation problem

- Four-Dimensional Variational Data Assimilation (4D-Var) is the method used by most national and international Numerical Weather Forecasting Centres to provide initial conditions for their forecast models.

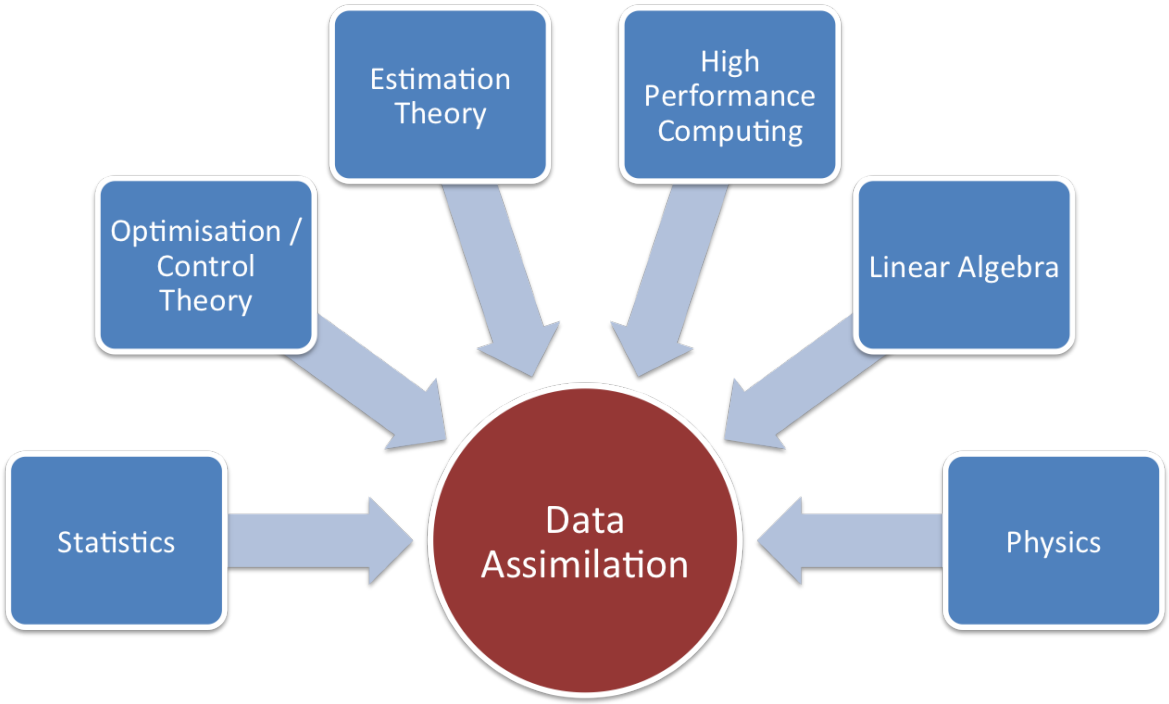


- has a grid with $\mathcal{O}(10^6)$ vertical columns
- has a timestep of $\mathcal{O}(10^3)$ seconds
- produces forecast $\approx$ 10 days ahead.

- Other applications: oceanography, aeronautics, hydrology, etc.

4D-Variational data assimilation problem

- Data assimilation involves different disciplines, different point of views



- In this talk, we will look at the problem from the optimization point of view

4D-Var Problem Formulation

$$\min_{\mathbf{x}\in\mathbb{R}^n} \underbrace{\frac{1}{2}\|x_0 - x_b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\sum_{j=0}^{N}\left\|\mathcal{H}_j(x_j) - y_j\right\|^2_{\mathbf{R}_j^{-1}}}_{J_o} + \underbrace{\frac{1}{2}\sum_{j=1}^{N}\|x_j - \mathcal{M}_j(x_{j-1})\|^2_{\mathbf{Q}_j^{-1}}}_{J_q}$$

- $\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{pmatrix} \in \mathbb{R}^n$ is the control variable where $x_j = x(t_j)$ defined at the start of each of a set of sub-windows that span the analysis window.
- x_b is the background given at the initial time (t_0).
- $y_j \in \mathbb{R}^{m_j}$ is the observation vector over a given time interval
- $\mathcal{H}_j$ maps the state vector x_j from model space to observation space
- $\mathcal{M}_j$ represents an integration of the numerical model from time t_{j-1} to t_j
- $\mathbf{B}$, $\mathbf{R}_j$ and $\mathbf{Q}_j$ are the covariance matrices of background, observation and model error.

4D-Var Problem Formulation

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|x_0 - x_b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{j=0}^{N} \left\|\mathcal{H}_j(x_j) - y_j\right\|^2_{\mathbf{R}_j^{-1}} + \frac{1}{2}\sum_{j=1}^{N} \|x_j - \mathcal{M}_j(x_{j-1})\|^2_{\mathbf{Q}_j^{-1}}$$

- Each x_i has $\approx 10^7$
- Each y_i has $\approx 10^5$
- The operators $\mathcal{H}_j$ and $\mathcal{M}_j$, and the matrices $\mathbf{B}$, $\mathbf{R}_j$ and $\mathbf{Q}_j$ are represented by codes that apply them to vectors
- We do not have access to their elements
- $\mathcal{H}_j$ and $\mathcal{M}_j$ involve integrations of the numerical model, and are computationally expensive.
- The covariance matrices $\mathbf{B}$, $\mathbf{R}_j$ and $\mathbf{Q}_j$ are less expensive to apply.

4D-Var Problem Formulation

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|x_0 - x_b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{j=0}^{N} \left\|\mathcal{H}_j(x_j) - y_j\right\|^2_{\mathbf{R}_j^{-1}} + \frac{1}{2}\sum_{j=1}^{N} \|x_j - \mathcal{M}_j(x_{j-1})\|^2_{\mathbf{Q}_j^{-1}}$$

- Each x_i has $\approx 10^7$
- Each y_i has $\approx 10^5$
- The operators $\mathcal{H}_j$ and $\mathcal{M}_j$, and the matrices $\mathbf{B}$, $\mathbf{R}_j$ and $\mathbf{Q}_j$ are represented by codes that apply them to vectors
- We do not have access to their elements
- $\mathcal{H}_j$ and $\mathcal{M}_j$ involve integrations of the numerical model, and are computationally expensive.
- The covariance matrices $\mathbf{B}$, $\mathbf{R}_j$ and $\mathbf{Q}_j$ are less expensive to apply.

We have a large-scale weighted nonlinear least-squares problem

Truncated Gauss-Newton method (Incremental 4D-Var)

1. Linearize the nonlinear cost function at the current iterate
2. Solve the linearized subproblem for the step $\boldsymbol{\delta x}^{(k)}$

$$\min_{\boldsymbol{\delta x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{L}\boldsymbol{\delta x}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{H}\boldsymbol{\delta x}-\mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

Truncated Gauss-Newton method (Incremental 4D-Var)

1. Linearize the nonlinear cost function at the current iterate
2. Solve the linearized subproblem for the step $\boldsymbol{\delta x}^{(k)}$

$$\min_{\boldsymbol{\delta x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{L}\boldsymbol{\delta x}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{H}\boldsymbol{\delta x}-\mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

where

- $\mathbf{L} = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}$
- $\mathbf{H} = diag(\mathbf{H_0}, \mathbf{H_1}, \ldots, \mathbf{H_N})$
- $\mathbf{D} = diag(\mathbf{B}, \mathbf{Q_1}, \ldots, \mathbf{Q_N})$ and $\mathbf{R} = diag(\mathbf{R_0}, \mathbf{R_1}, \ldots, \mathbf{R_N})$

Truncated Gauss-Newton method (Incremental 4D-Var)

1. Linearize the nonlinear cost function at the current iterate
2. Solve the linearized subproblem for the step $\boldsymbol{\delta x}^{(k)}$

$$\min_{\boldsymbol{\delta x}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{L}\boldsymbol{\delta x}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{H}\boldsymbol{\delta x}-\mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

where

- $\mathbf{L} = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}$
- $\mathbf{H} = diag(\mathbf{H_0}, \mathbf{H_1}, \ldots, \mathbf{H_N})$
- $\mathbf{D} = diag(\mathbf{B}, \mathbf{Q_1}, \ldots, \mathbf{Q_N})$ and $\mathbf{R} = diag(\mathbf{R_0}, \mathbf{R_1}, \ldots, \mathbf{R_N})$

3. Update the iterate $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta x}^{(k)}$

Solving the linearized subproblem

- From the optimality conditions:

$$\boxed{(\mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{L} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H})\boldsymbol{\delta x} = \mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{b} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

Solving the linearized subproblem

- From the optimality conditions:

$$\boxed{(\mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{L}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H})\delta\mathbf{x}=\mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{b}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

- $\mathbf{L}\delta\mathbf{x} = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix} \begin{pmatrix} \delta x_0 \\ \delta x_1 \\ \delta x_2 \\ \vdots \\ \delta x_N \end{pmatrix} = \begin{pmatrix} \delta x_0 \\ \delta x_1 - M_1 \delta x_0 \\ \delta x_2 - M_2 \delta x_1 \\ \vdots \\ \delta x_N - M_N \delta x_{N-1} \end{pmatrix}$

- Matrix-vector products with $\mathbf{L}$ can be parallelized in the time dimension

Solving the linearized subproblem

- From the optimality conditions:

$$\boxed{(\mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{L}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H})\delta\mathbf{x}=\mathbf{L}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{b}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

- $$\mathbf{L}\delta\mathbf{x}=\begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}\begin{pmatrix} \delta x_0 \\ \delta x_1 \\ \delta x_2 \\ \vdots \\ \delta x_N \end{pmatrix}=\begin{pmatrix} \delta x_0 \\ \delta x_1 - M_1\delta x_0 \\ \delta x_2 - M_2\delta x_1 \\ \vdots \\ \delta x_N - M_N\delta x_{N-1} \end{pmatrix}$$

- Matrix-vector products with $\mathbf{L}$ can be parallelized in the time dimension
- Preconditioning is difficult since

$$\mathbf{D}^{1/2}\widetilde{\mathbf{L}}^{-\mathbf{T}}(\mathbf{L}^{\mathbf{T}}\mathbf{D}^{-1}\mathbf{L})\widetilde{\mathbf{L}}^{-1}\mathbf{D}^{1/2}$$

can be ill-conditioned depending on the accuracy of $\widetilde{\mathbf{L}}^{-1}$.

Rewriting the linearized subproblem

- Making change of variables

$$\boxed{\delta \mathbf{p} = \mathbf{L}\delta \mathbf{x}}$$

the subproblem can also be rewritten as:

$$\boxed{(\mathbf{D}^{-1} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1})\delta \mathbf{p} = \mathbf{D}^{-1}\mathbf{b} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

- We name this approach as forcing formulation.

Rewriting the linearized subproblem

- Making change of variables

$$\boxed{\delta \mathbf{p} = \mathbf{L}\delta \mathbf{x}}$$

the subproblem can also be rewritten as:

$$\boxed{(\mathbf{D}^{-1} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1})\delta \mathbf{p} = \mathbf{D}^{-1}\mathbf{b} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

- We name this approach as forcing formulation.
- Preconditioning is straightforward

Rewriting the linearized subproblem

- Making change of variables

$$\boxed{\boldsymbol{\delta p} = \mathbf{L}\boldsymbol{\delta x}}$$

the subproblem can also be rewritten as:

$$\boxed{(\mathbf{D}^{-1} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1})\boldsymbol{\delta p} = \mathbf{D}^{-1}\mathbf{b} + \mathbf{L}^{-\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}}$$

- We name this approach as forcing formulation.
- Preconditioning is straightforward
- $\boldsymbol{\delta x} = \mathbf{L}^{-1}\boldsymbol{\delta p}$ is sequential $\rightarrow \delta x_j = M_j \delta x_{j-1} + \delta q_j$

A sequential algorithm in the time domain

- Matrix-vector products with the Jacobian and its transpose run one after the other.
- Model time-steps follow each other.

Alternative Solution: towards to a time-parallel algorithm

- Let us rewrite the problem as an equality constrained optimization problem:

$$\min_{(\mathbf{p},\mathbf{w})\in\mathbb{R}^{n+m}} f(\mathbf{p},\mathbf{w}) = \frac{1}{2}\|\mathbf{p}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{w}-\mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

$$\text{subject to} \quad \mathbf{p}_i = \mathbf{x}_i - \mathcal{M}_i(\mathbf{x}_{i-1})$$

$$\mathbf{w}_i = \mathcal{H}_i(\mathbf{x}_i)$$

for $i = 0, \ldots, N$, where

  - $\mathbf{y} = [\mathbf{y}_0; \mathbf{y}_1; \ldots; \mathbf{y}_N]$, $\mathbf{w} = [\mathbf{w}_0; \mathbf{w}_1; \ldots; \mathbf{w}_N]$ are m-dimensional vectors
  - $\mathbf{p} = [\mathbf{p}_0; \mathbf{p}_1; \ldots; \mathbf{p}_N]$ is an n-dimensional vector

Alternative Solution: towards to a time-parallel algorithm

- Let us rewrite the problem as an equality constrained optimization problem:

$$\min_{(\mathbf{p},\mathbf{w})\in\mathbb{R}^{n+m}} f(\mathbf{p},\mathbf{w}) = \frac{1}{2}\|\mathbf{p}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{w}-\mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

$$\text{subject to} \quad \mathbf{p}_i = \mathbf{x}_i - \mathcal{M}_i(\mathbf{x}_{i-1})$$

$$\mathbf{w}_i = \mathcal{H}_i(\mathbf{x}_i)$$

for $i = 0, \ldots, N$, where

  - $\mathbf{y} = [\mathbf{y}_0; \mathbf{y}_1; \ldots; \mathbf{y}_N]$, $\mathbf{w} = [\mathbf{w}_0; \mathbf{w}_1; \ldots; \mathbf{w}_N]$ are m-dimensional vectors
  - $\mathbf{p} = [\mathbf{p}_0; \mathbf{p}_1; \ldots; \mathbf{p}_N]$ is an n-dimensional vector

- This constrained problem can be solved by using Sequential Quadratic Programming (SQP) in which the constraints are linearized about the current point, and the objective function is replaced with a quadratic approximation.

Alternative Solution: towards to a time-parallel algorithm

- SQP solves:

$$\min_{(\boldsymbol{\delta p},\boldsymbol{\delta w})\in\mathbb{R}^{n+m}} = \frac{1}{2}\|\boldsymbol{\delta p}+\mathbf{p}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta w}+\mathbf{w}-\mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

$$\text{subject to} \quad \boldsymbol{\delta p} = \mathbf{L}\boldsymbol{\delta x} \quad \text{and} \quad \boldsymbol{\delta w} = \mathbf{H}\boldsymbol{\delta x}$$

Alternative Solution: towards to a time-parallel algorithm

- SQP solves:

$$\min_{(\boldsymbol{\delta p},\boldsymbol{\delta w})\in\mathbb{R}^{n+m}} = \frac{1}{2}\|\boldsymbol{\delta p}+\mathbf{p}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta w}+\mathbf{w}-\mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

$$\text{subject to} \quad \boldsymbol{\delta p} = \mathbf{L}\boldsymbol{\delta x} \quad \text{and} \quad \boldsymbol{\delta w} = \mathbf{H}\boldsymbol{\delta x}$$

- We can write the *Lagrangian function* for this problem as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\delta w},\boldsymbol{\delta p},\boldsymbol{\lambda},\boldsymbol{\mu}) =& \frac{1}{2}\|\boldsymbol{\delta p}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta w}-\mathbf{d}\|^2_{\mathbf{R}^{-1}} \\ &+ \boldsymbol{\lambda}^T(\boldsymbol{\delta p}-\mathbf{L}\boldsymbol{\delta x}) + \boldsymbol{\mu}^T(\boldsymbol{\delta w}-\mathbf{H}\boldsymbol{\delta x})\end{aligned}$$

Alternative Solution: towards to a time-parallel algorithm

- SQP solves:

$$\min_{(\boldsymbol{\delta p},\boldsymbol{\delta w})\in\mathbb{R}^{n+m}} = \frac{1}{2}\|\boldsymbol{\delta p}+\mathbf{p}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta w}+\mathbf{w}-\mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

$$\text{subject to} \quad \boldsymbol{\delta p} = \mathbf{L}\boldsymbol{\delta x} \quad \text{and} \quad \boldsymbol{\delta w} = \mathbf{H}\boldsymbol{\delta x}$$

- We can write the *Lagrangian function* for this problem as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\delta w},\boldsymbol{\delta p},\boldsymbol{\lambda},\boldsymbol{\mu}) =& \frac{1}{2}\|\boldsymbol{\delta p}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta w}-\mathbf{d}\|^2_{\mathbf{R}^{-1}} \\ &+ \boldsymbol{\lambda}^T(\boldsymbol{\delta p}-\mathbf{L}\boldsymbol{\delta x}) + \boldsymbol{\mu}^T(\boldsymbol{\delta w}-\mathbf{H}\boldsymbol{\delta x})\end{aligned}$$

- The stationary point of $\mathcal{L}$ satisfies the following equations:

$$\begin{aligned}\mathbf{D}^{-1}(\mathbf{L}\boldsymbol{\delta x}-\mathbf{b})+\boldsymbol{\lambda} &= 0\\ \mathbf{R}^{-1}(\mathbf{H}\boldsymbol{\delta x}-\mathbf{d})+\boldsymbol{\mu} &= 0\\ \mathbf{L}^T\boldsymbol{\lambda}+\mathbf{H}^T\boldsymbol{\mu} &= 0\end{aligned}$$

Alternative Solution: Saddle Point Approach

- In matrix form:

$$\begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{d} \\ \mathbf{0} \end{pmatrix}$$

  where $\mathcal{A}$ is a $(2n+m)$-by-$(2n+m)$ indefinite symmetric matrix.

- The solution of this problem is a saddle point.

Alternative Solution: Saddle Point Approach

- In matrix form:

$$\begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{d} \\ \mathbf{0} \end{pmatrix}$$

  where $\mathcal{A}$ is a $(2n+m)$-by-$(2n+m)$ indefinite symmetric matrix.

- The solution of this problem is a saddle point.

- SQP method thus results in solving a sequence of saddle-point systems

  $\rightarrow$ Solution algorithm: GMRES method with a general preconditioner.

Alternative Solution: Saddle Point Approach

- In matrix form:

$$\begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{d} \\ \mathbf{0} \end{pmatrix}$$

  where $\mathcal{A}$ is a $(2n+m)$-by-$(2n+m)$ indefinite symmetric matrix.

- The solution of this problem is a saddle point.

- SQP method thus results in solving a sequence of saddle-point systems

  $\rightarrow$ Solution algorithm: GMRES method with a general preconditioner.

Alternative Solution: Saddle Point Approach

- In matrix form:

$$\begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{d} \\ \mathbf{0} \end{pmatrix}$$

  where $\mathcal{A}$ is a $(2n+m)$-by-$(2n+m)$ indefinite symmetric matrix.

- The solution of this problem is a saddle point.

- SQP method thus results in solving a sequence of saddle-point systems

  $\rightarrow$ Solution algorithm: GMRES method with a general preconditioner.

Time-parallel algorithm

- Matrix-vector products with the Jacobian and its transpose can be run in parallel.
- Matrix-vector products with **L** can be performed in parallel.

Preconditioning Saddle Point Approach

$$\mathcal{A} = \begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}$$

- To solve the saddle point system with an iterative solver we need to find an efficient preconditioner!
- Preconditioning saddle point systems is the subject of much current research!
  - Nice review is given by Benzi, Golub and Liesen (2005).
- Most preconditioners in the literature assume that **D** and **R** are expensive, and **L** and **H** are cheap.
- The opposite is true in our case! **B** is the most computationally expensive block and calculations involving **A** are relatively cheap.

Preconditioning Saddle Point Formulation of 4D-Var

$$\mathcal{A} = \begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}$$

- The inexact constraint preconditioner proposed by (Bergamaschi et. al. 2005) is promising for our application. The preconditioner can be chosen as:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{D} & \mathbf{0} & \widetilde{\mathbf{L}} \\ \mathbf{0} & \mathbf{R} & \mathbf{0} \\ \widetilde{\mathbf{L}}^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \end{pmatrix} \Rightarrow \mathcal{P}^{-1} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \widetilde{\mathbf{L}}^{-\mathrm{T}} \\ \mathbf{0} & \mathbf{R}^{-1} & \mathbf{0} \\ \widetilde{\mathbf{L}}^{-1} & \mathbf{0} & -\widetilde{\mathbf{L}}^{-1}\mathbf{D}\widetilde{\mathbf{L}}^{-\mathrm{T}} \end{pmatrix}$$

where

  - $\widetilde{\mathbf{L}}$ is an approximation to the matrix $\mathbf{L}$
  - $\widetilde{\mathbf{B}} = [\widetilde{\mathbf{L}}^{\mathrm{T}} \quad \mathbf{0}]$ is a full row rank approximation of the matrix $\mathbf{B} \in \mathbb{R}^{n \times (m+n)}$

Preconditioning Saddle Point Formulation of 4D-Var

$$\underbrace{\begin{pmatrix} \mathbf{D} & \mathbf{0} & \mathbf{L} \\ \mathbf{0} & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^{\mathrm{T}} & \mathbf{H}^{\mathrm{T}} & \mathbf{0} \end{pmatrix}}_{\mathcal{A}_k} \underbrace{\begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\delta x} \end{pmatrix}}_{\mathbf{u}} = \underbrace{\begin{pmatrix} \mathbf{b} \\ \mathbf{d} \\ \mathbf{0} \end{pmatrix}}_{\mathbf{f}_k}$$

When solving a sequence of saddle point systems, can we further improve the preconditioning for the outer loops $k > 1$?

Can we find low-rank updates for the inexact constraint preconditioner that approximates $\mathcal{A}^{-1}$ or its effect on a vector?

Preconditioning Saddle Point Formulation of 4D-Var

- For $k = 1$, we have the inexact constraint preconditioner:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \quad \Rightarrow \quad \mathcal{P}^{-1}\mathcal{A}_1\,\mathbf{u} = \mathcal{P}^{-1}\mathbf{f}_1$$

Preconditioning Saddle Point Formulation of 4D-Var

- For $k = 1$, we have the inexact constraint preconditioner:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \quad \Rightarrow \quad \mathcal{P}^{-1}\mathcal{A}_1\,\mathbf{u} = \mathcal{P}^{-1}\mathbf{f}_1$$

- For $k > 1$, we want to find a low-rank update $\Delta\mathbf{B} = \mathbf{B} - \widetilde{\mathbf{B}}$ and use the updated preconditioner:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} \Rightarrow \mathcal{P}^{-1}\mathcal{A}_{k+1}\,\mathbf{u} = \mathcal{P}^{-1}\mathbf{f}_{k+1}$$

Preconditioning Saddle Point Formulation of 4D-Var

- For $k = 1$, we have the inexact constraint preconditioner:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \quad \Rightarrow \quad \mathcal{P}^{-1}\mathcal{A}_1\,\mathbf{u} = \mathcal{P}^{-1}\mathbf{f}_1$$

- For $k > 1$, we want to find a low-rank update $\Delta\mathbf{B} = \mathbf{B} - \widetilde{\mathbf{B}}$ and use the updated preconditioner:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} \Rightarrow \mathcal{P}^{-1}\mathcal{A}_{k+1}\,\mathbf{u} = \mathcal{P}^{-1}\mathbf{f}_{k+1}$$

How to obtain these updates?

$\rightarrow$ GMRES performs matrix-vector products with $\mathcal{A}$:

$$\underbrace{\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}}_{\mathcal{A}_k} \underbrace{\begin{pmatrix} \delta\mathbf{z}_j \\ \delta\mathbf{x}_j \end{pmatrix}}_{\mathbf{u}_j^{(k)}} = \underbrace{\begin{pmatrix} \mathbf{b}_j \\ \mathbf{c}_j \end{pmatrix}}_{\mathbf{f}_j^{(k)}}$$

$\rightarrow$ We can use the pairs $(\mathbf{u}_j^{(k)}, \mathbf{f}_j^{(k)})$ to find an update $\Delta\mathbf{B}_k$.

Preconditioning Saddle Point Formulation of 4D-Var

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \mathbf{A}\boldsymbol{\delta z} + \widetilde{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\delta x} \\ \widetilde{\mathbf{B}}\boldsymbol{\delta z} \end{pmatrix} + \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\boldsymbol{\delta x} \\ \Delta\mathbf{B}\boldsymbol{\delta z} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\boldsymbol{\delta x} \\ \Delta\mathbf{B}\boldsymbol{\delta z} \end{pmatrix} = \begin{pmatrix} \mathbf{b} - \mathbf{A}\boldsymbol{\delta z} - \widetilde{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\delta x} \\ \mathbf{c} - \widetilde{\mathbf{B}}\boldsymbol{\delta z} \end{pmatrix}$$

Preconditioning Saddle Point Formulation of 4D-Var

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta z} \\ \boldsymbol{\delta x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \mathbf{A}\boldsymbol{\delta z} + \widetilde{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\delta x} \\ \widetilde{\mathbf{B}}\boldsymbol{\delta z} \end{pmatrix} + \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\boldsymbol{\delta x} \\ \Delta\mathbf{B}\boldsymbol{\delta z} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\boldsymbol{\delta x} \\ \Delta\mathbf{B}\boldsymbol{\delta z} \end{pmatrix} = \begin{pmatrix} \mathbf{b} - \mathbf{A}\boldsymbol{\delta z} - \widetilde{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\delta x} \\ \mathbf{c} - \widetilde{\mathbf{B}}\boldsymbol{\delta z} \end{pmatrix}$$

- Let's define the vector $\mathbf{g} = \mathbf{b} - \mathbf{A}\boldsymbol{\delta z}$
- Then *the direct* and *adjoint conditions* are given respectively as:

$$\Delta\mathbf{B}\,\boldsymbol{\delta z} = \mathbf{c} - \widetilde{\mathbf{B}}\boldsymbol{\delta z} = \mathbf{q}$$

$$\Delta\mathbf{B}^{\mathrm{T}}\boldsymbol{\delta x} = \mathbf{g} - \widetilde{\mathbf{B}}^{\mathrm{T}}\boldsymbol{\delta x} = \mathbf{p}$$

Preconditioning Saddle Point Formulation of 4D-Var

- When there are j pairs, we have set of equations:

$$\Delta \mathbf{B}_k^T \mathbf{X} = \mathbf{P}, \tag{1}$$
$$\Delta \mathbf{B}_k \mathbf{Y} = \mathbf{Q}, \tag{2}$$

where

$$\mathbf{Y} = [\delta \mathbf{z}_0, \ldots, \delta \mathbf{z}_j],$$
$$\mathbf{X} = [\delta \mathbf{x}_0, \ldots, \delta \mathbf{x}_j],$$
$$\mathbf{P} = [\mathbf{p}_0, \ldots, \mathbf{p}_j],$$
$$\mathbf{Q} = [\mathbf{q}_0, \ldots, \mathbf{q}_j]$$

are full column rank matrices.

Preconditioning Saddle Point Formulation of 4D-Var

- When there are j pairs, we have set of equations:

$$\Delta \mathbf{B}_k^T \mathbf{X} = \mathbf{P}, \tag{1}$$
$$\Delta \mathbf{B}_k \mathbf{Y} = \mathbf{Q}, \tag{2}$$

where

$$\mathbf{Y} = [\delta \mathbf{z}_0, \ldots, \delta \mathbf{z}_j],$$
$$\mathbf{X} = [\delta \mathbf{x}_0, \ldots, \delta \mathbf{x}_j],$$
$$\mathbf{P} = [\mathbf{p}_0, \ldots, \mathbf{p}_j],$$
$$\mathbf{Q} = [\mathbf{q}_0, \ldots, \mathbf{q}_j]$$

are full column rank matrices.

Lemma: Consistency

Let us assume that each set of equations (1) and (2) has solutions. Let us also assume that $\mathbf{P}^T\mathbf{Y}$ is non singular. Then, Eqs. (1) and (2) are consistent if and only if the relation

$$\mathbf{P}^T\mathbf{Y} = \mathbf{X}^T\mathbf{Q}$$

holds.

Rank-k update

- An update $\Delta \mathbf{B}_k$ to $\mathbf{B}_0$ can be written as

$$\Delta \mathbf{B}_k = \mathbf{V}\mathbf{Z}\mathbf{U}^T, \tag{3}$$

where $\mathbf{U}$ is a n-by-k matrix, $\mathbf{Z}$ is a k-by-k matrix and $\mathbf{V}$ is a m-by-k matrix.

Rank-k update

- An update $\Delta \mathbf{B}_k$ to $\mathbf{B}_0$ can be written as

$$\Delta \mathbf{B}_k = \mathbf{V}\mathbf{Z}\mathbf{U}^T, \tag{3}$$

where $\mathbf{U}$ is a n-by-k matrix, $\mathbf{Z}$ is a k-by-k matrix and $\mathbf{V}$ is a m-by-k matrix.

Theorem

Suppose that the multiple secant equations (1) and (2) are consistent. Then the rank-k matrix in the form of (3) given by

$$\Delta \mathbf{B}_k = \mathbf{Q}(\mathbf{P}^T\mathbf{Y})^{-1}\mathbf{P}^T = \mathbf{Q}(\mathbf{X}^T\mathbf{Q})^{-1}\mathbf{P}^T$$

solves the multiple secant equations.

Rank-k update

- A special case for $k = 1$ reduces to a rank-1 update:

$$\Delta \mathbf{B}_k = \frac{\mathbf{q}\,\mathbf{p}^T}{\mathbf{q}^T \delta \mathbf{x}}$$

- This update called as two-sided-rank-one (TR1) by Griewank and Walther (2002), and used to update Jacobian matrix in a constrained optimisation problem. Haber (2005) also used this update to approximate the Jacobian in a quasi-Newton method for nonlinear inverse problems.

Rank-k update

- A special case for $k = 1$ reduces to a rank-1 update:

$$\Delta \mathbf{B}_k = \frac{\mathbf{q}\,\mathbf{p}^T}{\mathbf{q}^T \delta \mathbf{x}}$$

- This update called as two-sided-rank-one (TR1) by Griewank and Walther (2002), and used to update Jacobian matrix in a constrained optimisation problem. Haber (2005) also used this update to approximate the Jacobian in a quasi-Newton method for nonlinear inverse problems.

- TR1 update
  - generalizes the classical symmetric rank-one (SR1) update.
  - maintains the validity of all previous secant conditions.
  - is invariant with respect to linear transformations
  - It has apparently no least change characterization in terms of any particular matrix norm.

Rank-$2k$ update

Theorem

Let us define the matrix $\Delta\mathbf{B}_k$ *as* $\Delta\mathbf{B}_k = \mathbf{B}_{k+1} - \mathbf{B}_k$. *Let us assume also that the multiple secant equations are consistent. Then the solution of the problem*

$$\min_{\Delta\mathbf{B}_k} \|\Delta\mathbf{B}_k\|_F$$

$$\text{subject to } \Delta\mathbf{B}_k^T \mathbf{X} = \mathbf{P}$$

$$\Delta\mathbf{B}_k \mathbf{Y} = \mathbf{Q}$$

is given by

$$\Delta\mathbf{B}^* = \mathbf{X}^{T\dagger}\mathbf{P}^T + (\mathbf{I} - \mathbf{X}\mathbf{X}^\dagger)\mathbf{Q}\mathbf{Y}^\dagger,$$

or equivalently

$$\Delta\mathbf{B}^* = (\mathbf{Y}^{T\dagger}\mathbf{Q}^T + (\mathbf{I} - \mathbf{Y}\mathbf{Y}^\dagger)\mathbf{P}\mathbf{X}^\dagger)^T.$$

Rank-$2k$ update

- A special case for $k = 1$ reduces to a rank-2 update:

$$\Delta \mathbf{B}_k = \frac{\delta \mathbf{x}\, \mathbf{p}^T}{\delta \mathbf{x}^T \delta \mathbf{x}} + \frac{\mathbf{q}\, \delta \mathbf{z}^T}{\delta \mathbf{z}^T \delta \mathbf{z}} - \frac{\delta \mathbf{x}\, \delta \mathbf{x}^T \mathbf{q}\, \delta \mathbf{z}^T}{\delta \mathbf{x}^T \delta \mathbf{x}\, \delta \mathbf{z}^T \delta \mathbf{z}}$$

- We name this update as the least-Frobenius two-sided rank-2 update (FTR2). This update
  - generalizes the classical Powell-symmetric-Broyden (PSB) update
  - maintains the validity of all previous secant conditions when used with the block formula
  - is not invariant with respect to linear transformations

Rank-$2k$ update

Theorem

Let us define $\Delta \mathbf{B}_k = \mathbf{B}_{k+1} - \mathbf{B}_k$ *and assume that multiple secant equations are consistent. Let* $\mathbf{V} \in \Re^{m \times k}$ *and* $\mathbf{S} \in \Re^{n \times k}$ *be such that* $\mathbf{X}^T\mathbf{V}$, $\mathbf{Y}^T\mathbf{S}$ *are symmetric and positive definite matrices. Then,*

$$\Delta \mathbf{B}_k^* = \mathbf{V}(\mathbf{X}^T\mathbf{V})^{-1}\mathbf{P}^T + (\mathbf{Q} - \mathbf{V}(\mathbf{X}^T\mathbf{V})^{-1}\mathbf{X}^T\mathbf{Q})(\mathbf{Y}^T\mathbf{S})^{-1}\mathbf{S}^T$$

is the solution of the problem

$$\min_{\Delta \mathbf{B}_k} \|\mathbf{W}_1^{-1}\Delta \mathbf{B}_k \mathbf{W}_2^{-1}\|_F$$
$$\text{subject to } \Delta \mathbf{B}_k^T \mathbf{X} = \mathbf{P}$$
$$\Delta \mathbf{B}_k \mathbf{Y} = \mathbf{Q}$$

Here $\mathbf{W}_1 \in \Re^{m \times m}$ *and* $\mathbf{W}_2 \in \Re^{n \times n}$ *are nonsingular matrices such that* $\mathbf{W}_1\mathbf{W}_1^T\mathbf{X} = \mathbf{V}$ *and* $\mathbf{W}_2^T\mathbf{W}_2\mathbf{Y} = \mathbf{S}$.

Rank-2k update

- A special case for $k = 1$ reduces to a rank-2 update:

$$\Delta \mathbf{B}_k = \frac{\mathbf{v}\,\mathbf{p}^T}{\delta \mathbf{x}^T \mathbf{v}} + \frac{\mathbf{q}\,\mathbf{s}^T}{\delta \mathbf{z}^T \mathbf{s}} - \frac{\mathbf{v}\,\delta \mathbf{x}^T \mathbf{q}\,\mathbf{s}^{\mathbf{T}}}{\delta \mathbf{x}^T \mathbf{v}\,\delta \mathbf{z}^T \mathbf{s}}$$

- We name this update as the weighted least-Frobenius two-sided rank-2 update (WFTR2). This update
  - generalizes the classical Davidon-Fletcher-Powell (DFP) update
  - maintains the validity of all previous secant conditions when used with the block formula
  - is invariant with respect to linear transformations

How to use these updates in GMRES?

- We can write the block update formulas in the form of

$$\mathbf{B}_{k+1} = \mathbf{B}_0 + \mathbf{W}_k \mathbf{D}_k \mathbf{U}_k^T.$$

- Using this structure, the first-level preconditioner can then be updated from

$$\begin{aligned}\mathcal{P}_k &= \begin{pmatrix} \mathbf{A} & \mathbf{B}_0^T \\ \mathbf{B}_0 & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \mathbf{U}_k \mathbf{D}_k^T \mathbf{W}_k^T \\ \mathbf{W}_k \mathbf{D}_k \mathbf{U}_k^T & \mathbf{0} \end{pmatrix} \\ &= \mathcal{P}_0 + \underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{U}_k \\ \mathbf{W}_k & \mathbf{0} \end{pmatrix}}_{\mathbf{F}} \underbrace{\begin{pmatrix} \mathbf{D}_k \mathbf{U}_k^T & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_k^T \mathbf{W}_k^T \end{pmatrix}}_{\mathbf{G}}\end{aligned}$$

- $\mathcal{P}_k^{-1}$ can be obtained by using Sherman-Morrison-Woodbury formula.

How to use these updates in GMRES?

- We can write the block update formulas in the form of

$$\mathbf{B}_{k+1} = \mathbf{B}_0 + \mathbf{W}_k \mathbf{D}_k \mathbf{U}_k^T.$$

- Using this structure, the first-level preconditioner can then be updated from

$$\begin{aligned}\mathcal{P}_k &= \begin{pmatrix} \mathbf{A} & \mathbf{B}_0^T \\ \mathbf{B}_0 & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \mathbf{U}_k \mathbf{D}_k^T \mathbf{W}_k^T \\ \mathbf{W}_k \mathbf{D}_k \mathbf{U}_k^T & \mathbf{0} \end{pmatrix} \\ &= \mathcal{P}_0 + \underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{U}_k \\ \mathbf{W}_k & \mathbf{0} \end{pmatrix}}_{\mathbf{F}} \underbrace{\begin{pmatrix} \mathbf{D}_k \mathbf{U}_k^T & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_k^T \mathbf{W}_k^T \end{pmatrix}}_{\mathbf{G}}\end{aligned}$$

- $\mathcal{P}_k^{-1}$ can be obtained by using Sherman-Morrison-Woodbury formula.
- Then, the preconditioned system

$$\mathcal{P}_k^{-1} \mathcal{A}_{k+1}\, \mathbf{u} = \mathcal{P}_k^{-1} \mathbf{f}_{k+1}$$

is solved in GMRES.

Numerical Results

Implementation platform

- We used the Object Oriented Prediction System (OOPS) developed by ECMWF
- OOPS consists of simplified models of a real-system

The model

- It is a two-layer quasi-geostraphic model with 1600 grid-points

Implementation details

- There are 100 observations of stream function every 3 hours, 100 wind observations plus 100 wind-speed observations every 6 hours
- The error covariance matrices are assumed to be horizontally isotropic and homogeneous, with Gaussian spatial structure
- The analysis window is 24 hours, and is divided into 8 subwindows
- 3 outer loops with 10 inner loops each are performed

Methods

1. Standard weak-constrained 4D-Var formulation
   $\rightarrow$ Solution method is preconditioned conjugate-gradients
2. Saddle point formulation with an updated inexact constraint preconditioner
   $\rightarrow$ Solution method is GMRES
   $\rightarrow$ The initial preconditioner is chosen as

$$\mathcal{P}_0 = \begin{pmatrix} \mathbf{D} & \mathbf{0} & \widetilde{\mathbf{L}} \\ \mathbf{0} & \mathbf{R} & \mathbf{0} \\ \widetilde{\mathbf{L}}^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \end{pmatrix} \quad \text{with} \quad \widetilde{\mathbf{L}} = \begin{pmatrix} \mathbf{I} & & & \\ -\mathbf{I} & \mathbf{I} & & \\ & \ddots & \ddots & \\ & & -\mathbf{I} & \mathbf{I} \end{pmatrix}.$$

$$\mathcal{P}_0^{-1} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \widetilde{\mathbf{L}}^{-\mathrm{T}} \\ \mathbf{0} & \mathbf{R}^{-1} & \mathbf{0} \\ \widetilde{\mathbf{L}}^{-1} & \mathbf{0} & -\widetilde{\mathbf{L}}^{-1}\mathbf{D}\widetilde{\mathbf{L}}^{-\mathrm{T}} \end{pmatrix} \quad \text{and} \quad \widetilde{\mathbf{L}}^{-1} = \begin{pmatrix} \mathbf{I} & & & \\ \mathbf{I} & \mathbf{I} & & \\ \vdots & \ddots & \ddots & \\ \mathbf{I} & \cdots & \mathbf{I} & \mathbf{I} \end{pmatrix}.$$

Second-level preconditioners:

1. $\mathcal{T}_k$: The preconditioner obtained by using the TR1 update
2. $\mathcal{F}_k$: The preconditioner obtained by using the weighted least-Frobenius update

The performance of the second level preconditioners

→ Last 8 pairs were used to construct the preconditioner

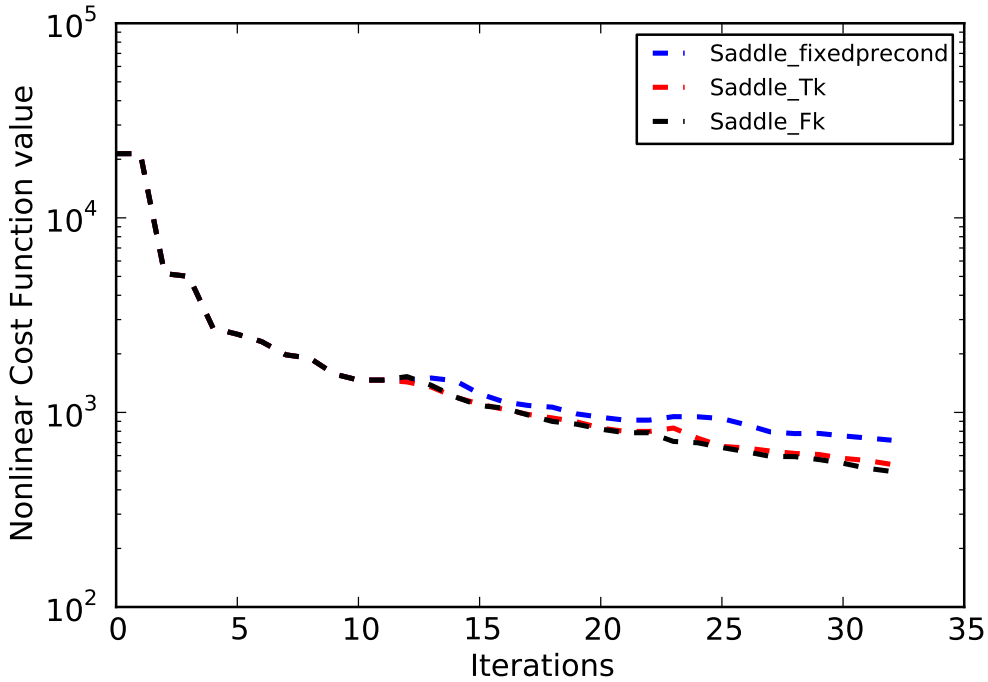


Figure: Nonlinear cost function values along iterations

- Second-level preconditioners obtained by using updates accelerate the convergence
- The performance of the least-Frobenius and TR1 update are very similar.

Overall performance compared with the standard 4DVar formulation

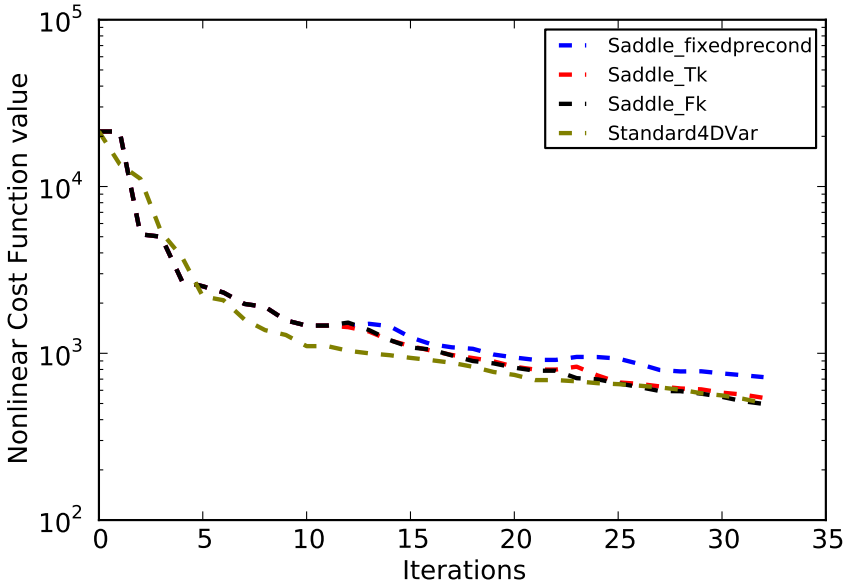


Figure: Nonlinear cost function values along iterations

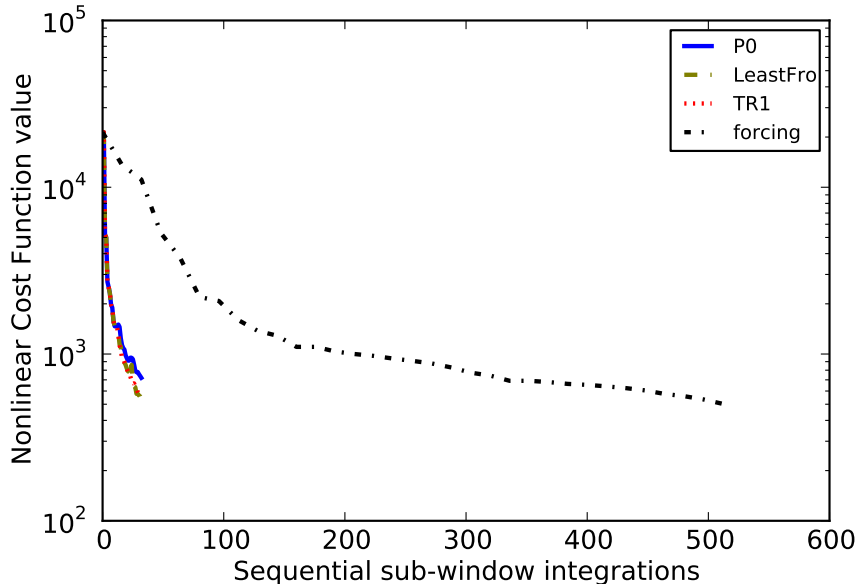


Figure: Nonlinear cost function values along sequential subwindow integrations

- At each iteration the standard 4DVar formulation requires one application of $\mathbf{L}^{-1}$, followed by one application of $\mathbf{L}^{-\mathrm{T}}$ (16 sequential subwindow integrations)
- At each iteration of saddle point formulation require one subwindow integration (provided that $\mathbf{L}$ and $\mathbf{L}^{T}$ are applied simultaneously)

Conclusions

- The saddle point formulation of weak-constraint 4D-Var allows parallelisation in the time dimension.
- Finding an effective preconditioner is a key issue in solving the saddle point systems.
- The inexact constraint preconditioner can be used to precondition the saddle point formulation of 4D-Var.
- When solving a sequence of saddle point systems, a low-rank low-cost update formulas can be found to further improve preconditioning.
- The derived general class of low-rank updates can be used to update Jacobian information used in constrained optimisation, in the solution of nonlinear equations.

Thank you for your attention !