Some Numerical methods for Inverse Problems in Geosciences

Serge Gratton (N7)

Cours N7 2020

- Introduction
    - Basic concepts
- Data Assimilation from the parameter estimation point of view
    - Preliminary definitions and notations
    - Estimation under the Gaussian assumption
    - Estimation under the linearity assumption.
    - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
    - The problem
    - Steps in duality
    - Solution algorithms
    - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
    - Dual space solvers
    - Preconditioning in dual space
    - Ensemble Kalman filter
    - Domain decomposition solvers
- Weak constrained formulation (with Selime Gurol)

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Definition of an ”Analysis”

Atmosphère
Océan
Orthographie

It is the process of estimating

- The true state of a system at a given time
- Possibly other model parameters

It is based on

- Observational data (e.g. physical measurements)
- A model of the physical system
- Some ”a priori” information on the system
- Possibly additional constraints (desired properties of the solution that are not present in the model)

Loi de N

mat def. pos.

$\| I x - 0 \|_c^2$

$\varphi(x)$ "

$\frac{1}{2}\|Ax-b\|_2^2 + \alpha \|x\|_2^2$

$$\varphi(x) = \frac{1}{2} x^T A^T A x - x$$

$$= \frac{1}{2} x^T \underbrace{(A^T A + \alpha I)}_{} x - x^T A^T b + cte$$

vp $\geq \alpha > 0$

CNS: $x = (A^T A + \alpha I)^{-1} A^T b$

A classical analysis method (Bertgthorson Doos, 1955)

Based on

- Synchronous observations on-board ships (surface pressure)
- Model parameters : temperature, wind and pressure
- An evolution model defined on a spatial grid

This analysis was the following

1. Estimation of a first guess field obtained by extrapolation (time integration) of the model
2. Interpolation to get a predicted value at the observation location
3. Computation of the observed minus predicted quantities (a misfit)
4. Interpolation back on the model grid point of the misfit
5. Correction of the model parameters to reduce the misfit

This led to the *optimal interpolation process* still in use.

Wheather forecasts

In the domain of the numerical weather forecasting,

- The analysis prepares initial conditions for weather forecasts
- Observations of various types are available (data from satellites, ground stations, commercial flights, sounding balloons)
- The discrepency between observed and predicted values is minimized : optimization problem

We will focus on the formulation and solution of the optimization problem.

Note however that to have a reasonable solution, the data must be of a good quality in terms of coverage, and the models for the data have to be accurate.

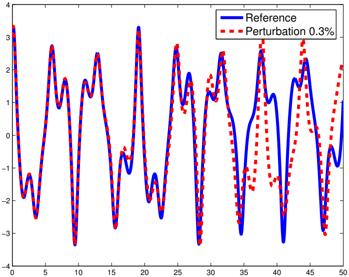
An important problem : nonlinearity (I)

- Solving a linear problem may be chalenging if it is large and/or very ill-conditioned
- Nonlinearity introduces additional difficulties.
- For instance on the Duffing's equation $\ddot{x} + \frac{5}{100}\dot{x} + x^3 = \frac{15}{2}\cos(t)$, sensitivity with respect to initial conditions. Take for as truth $x(0) = 3$, $\dot{x}(0) = 4$ and consider the perturbed conditions $x(0) = 3.01$, $\dot{x}(0) = 4.01$ and $x(0) = 3.02$, $\dot{x}(0) = 4.02$.
- Comparison of the solutions obtained by time integration and interpret error in initial condition as analysis error.
- The analysis error reduces the time period for which the forecast is close to the truth.

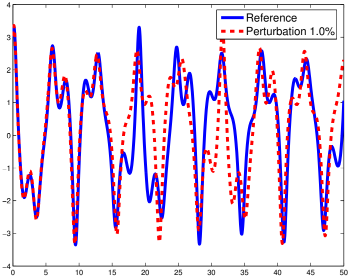
Perturbation of initial conditions : nonlinearity (II)

$$\ddot{x} + \frac{5}{100}\dot{x} + x^3 = \frac{3}{2}\cos(t)$$

Exact v.s. 0.3% pert.



Exact v.s. 1.0% pert.

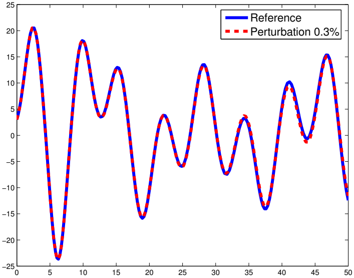


- Until $t = 35$, the forecast seems reasonable with a 0.3% perturbation
- With a 1.0% perturbation the two trajectories diverges already for $t = 15$

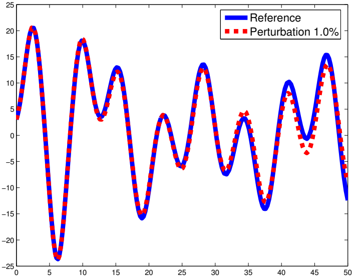
Perturbation of initial conditions : nonlinearity (III)

$$\ddot{x} + \frac{5}{100}\dot{x} + 10^{-2}x^3 = \frac{3}{2}\cos(t)$$

Exact v.s. 0.3% pert.



Exact v.s. 1.0% pert.



- The reduction of the nonlinear term enlarges the time validity for the forecast

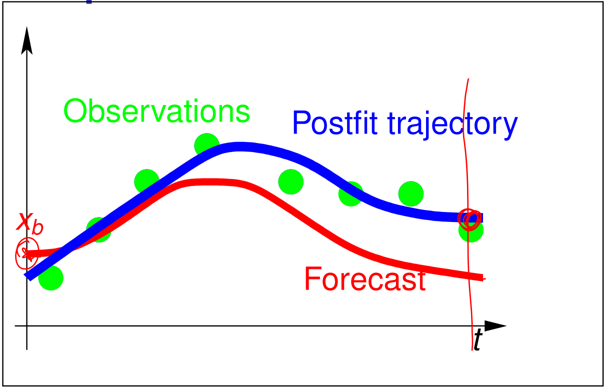
Angles to see Data Assimilation

The domain can be discussed from many angles :

- Variational analysis
- Optimization/Control theory
- Estimation theory
- Probability theory
- Numerical optimization/linear algebra
- High performance computing on parallel machines

Goal of this course show the connections between all this fields in the framework of Data Assimilation

Optimization point of view



- Goal : find the initial state of a dynamical system to perform forecasts
- Use observations and a model of the system
- Determine the initial state by solving an optimization problem (here, a control problem). Minimize the discrepancy between observations and model.

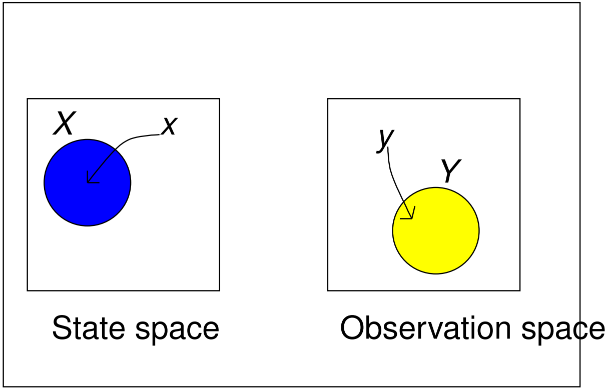
Parameter estimation view

- A priori knowledge on values of $x(0)$
- Observations : y_i
- Observation model $y_i = h_i(x)$ + noise
- Dynamical model $\dot{x}(t) = f(t, x(t))$ + noise
- Find $x(0)$

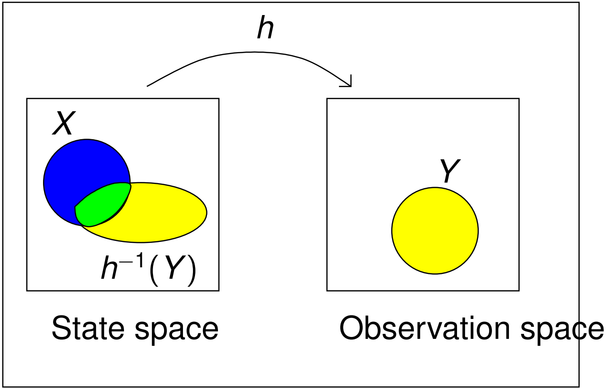
x_{t+1}

[illegible]

Estimation from set theory

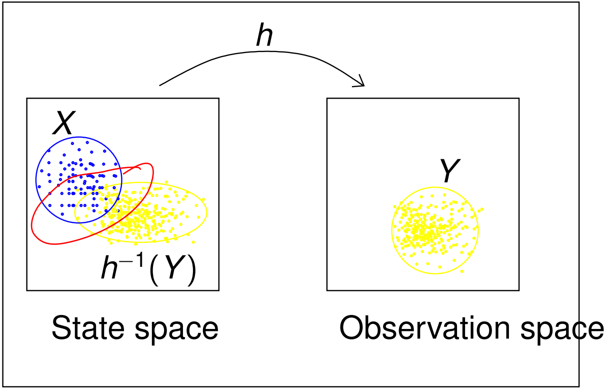


Estimation from set theory



Model $y = h(x)$, solution in $X \cap h^{-1}(Y)$.

Inclusion of statistical knowledge



High performance computing point of view

$2nm$

- The simplest instance of a Data Assimilation problem is a linear least squares problem
- Typical sizes would be for this problem 10^7 unknowns and $2 \cdot 10^7$ observations (including a priori infomation)
- The problem is not really sparse (it is structured)

$\frac{10^{21}}{10^{10}}$

- If no particular structure taken into account, the solution of the problem on a modern ($3 \cdot 10^9$ operations/s) computer would take 200 centuries of computation by the normal equations
- In terms of memory, available computers (2017) not able to store in core memory the matrix
- Therefore parallel iterative methods are sought for parallel computers

$\min_x \| A x - \|_2^2 \Leftrightarrow A^TAx = A^Tb$

$m \times n$

$n \le m$

$\boxed{2mn^2}$, nm, nm^2

mn^2

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Outline

Introduction

- Basic concepts

Data Assimilation from the parameter estimation point of view

- Preliminary definitions and notations
- Estimation under the Gaussian assumption
- Estimation under the linearity assumption.
- Filtering the system in full generality

Particle filter in a nutshell

Data Assimilation form the optimization point of view

- The problem
- Steps in duality
- Solution algorithms
- Sequential estimation and Kalman filter

Data assimilation from the linear algebra point of view

Specialized solvers

- Dual space solvers
- Preconditioning in dual space
- Ensemble Kalman filter
- Domain decomposition solvers

Definition

A probability space is the triplet $(\mathcal{S}, \mathcal{B}, \mathcal{P})$ where :

- $\mathcal{S}$ is the sample space, $\mathcal{B}$ is a collection of subsets (*sigma* algebra) of $\mathcal{S}$
- $\mathcal{P}$ is a probability measure (i.e. $\mathcal{P} \to \mathbb{R}^+$, $\mathcal{P}(\{\}) = 0$, $\mathcal{P}(\mathcal{S}) = 1$ and for countable disjoints sets, $\mathcal{P}(\sum_i S_i) = \sum_i \mathcal{P}(S_i)$).

Definition

A random variable is a measurable function $X \ : \ \mathcal{S} \to \mathbb{R}$.

Definition

The cumulative distribution of X is the function $F_X(x) = \mathcal{P}(X \leq x)$.

Definition

The mean or expectation of a random variable X is defined by $E(X) = \int_{-\infty}^{+\infty} x dF_X(x)$; for a continuous variable (our case), $dF_X(x) = p_X(x)dx$. The expectation operator is linear.

Definition

Two random variables are jointly distributed if they are both defined on the same probability space.

Definition

A random vector $X = (X_1, \dots X_n)$ is a mapping from $\mathcal{S}$ to $\mathbb{R}^n$ for which all the components X_i are jointly distributed. The joint probability distribution is given for $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ by

$$F_X(x) = \mathcal{P}(X_1 \leq x_1, \dots, X_n \leq x_n).$$

Definition

The components X_i are independent if the joint probability distribution is the product of the cumulative distributions, i.e. $F_X(x) = \prod F_{X_i}(x_i)$.

Definition

The cumulative distribution of X is the function $F_X(x) = \mathcal{P}(X \leq x)$.

Definition

A random vector X has the joint probability density function p_X if

$$F_X(x) = \int_{-\infty}^{x_1} \cdots \int_{-\infty}^{x_n} p_X(u)\, du_1 \dots du_n.$$

Definition

The mean or expected value of a random vector X is the vector $E(X) = (E(X_1), \dots, E(X_n))^T$. The covariance matrix is the $n \times n$ matrix $\mathrm{cov}(X) = E((X - \mu)(X - \mu)^T)$, where $\mu = E(X) \in \mathbb{R}^n$, i.e. $[\mathrm{cov}(X)]_{ij} = E((X_i - \mu_i)(X_j - \mu_j))$.

All covariance matrices in this lecture are assumed symmetric and positive definite !

Example

A random vector has a Gaussian (or Normal) distribution if its joint probability density function is

$$p_X(x) = \frac{1}{\sqrt{(2\pi)^n \det(C)}} \exp\left(-\frac{1}{2}(x-\mu)^T C^{-1}(x-\mu)\right).$$

One has

- $E(X) = \mu$
- $\operatorname{cov}(X) = C$

Notation : $X \sim \mathcal{N}(\mu, C)$.

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Definition

Suppose a random vector Y has a joint probability function $p_Y(y; x)$, where x is an unknown parameter vector. Suppose $y \in \mathbb{R}^n$ is a realization of Y.

Definition

A maximum likelihood estimator for x given y is a parameter that maximizes the log likelihood function $L(x) = \log p_Y(y; x)$.

Example

Suppose y is a realization of a Gaussian vector $\boxed{Y \sim \mathcal{N}(Ax, C)}$, where the parameter $x \in \mathbb{R}^n$ is unknown. The log likelihood is $-\frac{1}{2}(y - Ax)^T C^{-1}(y - Ax) + \log c$, and x is the solution of the linear least-squares problem

$A^+ y = (A^TA)^{-1}A^Ty$

$$\min_x \|Ax - y\|_{C^{-1}}.$$

$$\|x\|_{C^{-1}}^2 \equiv x^T C^{-1} x$$

Some rationale for the maximum likelihood

The information we have is that y is a realization of Y. We have (make a picture involving disjoint sets)

$$\begin{aligned} &\mathcal{P}(y_1 < Y_1 < y_1 + dy_1, \dots, y_n < Y_n < y_n + dy_n\} \\ = c \int_{y_1}^{y_1+dy_1} \cdots \int_{y_n}^{y_n+dy_n} &\exp\left(-\frac{1}{2}(y - Ax)^T C^{-1} (y - Ax)\right) dy_1 \dots dy_n. \\ &\sim c \exp\left(-\frac{1}{2}(y - Ax)^T C^{-1} (y - Ax)\right) dy_1 \dots dy_n \end{aligned}$$

Therefore if $-\|Ax - y\|^2_{C^{-1}}$ is enlarged, the probability that $y_i < Y_i < y_i + dy_i$ (i.e. y is close to a realization of Y) is increased. Note that $Y \sim \mathcal{N}(Ax, C)$ is equivalent to $X = Ax + N$, where $N \sim \mathcal{N}(0, C)$.

Inclusion of a priori information

- We may want to incorporate additional information by viewing the parameter vector as a realization of a random vector X.
- This analysis is referred to as Bayesian estimation
- It relies on the notion of conditional probability given $Y = y$, defined by the function :

$$p_{X|Y}(x|y) = \frac{p_{(X,Y)}}{p_Y(y)}.$$

$\boxed{Y = \mathcal{H}(x) + \varepsilon}$

X a priori

- If X and Y are independant random vectors, $p_{X|Y}(x|y) = p_X(x)$.

Definition

The maximum a posteriori estimator maximizes $p_{X|Y}(x|y)$ over x. It is a function of y.

a priori $\quad p_{x|y}(x|y) \quad \frac{(x-5)^2}{100}$

Example

We consider the random vector Y defined by $Y = AX + N$, where $X \sim \mathcal{N}(x_b, B)$ and $N \sim \mathcal{N}(0, R)$ are two independent random vectors. Let y be a realization of Y. The MAP of x is $x_b + \left(A^T R^{-1} A + B^{-1}\right)^{-1} A^T R^{-1}(y - Ax_b)$.

1

Proof.

The Bayes law reads $p_{X|Y}(x|y) = \frac{p_{Y|X}(y|x)p_X(x)}{p_Y(y)}$. Since $p_{Y|X}(y|x) = p_{AX+N|X}(y|x) = p_{Ax+N|X}(y|x)$, from the independence of N and X we get $p_{Y|X}(y|x) = p_{Ax+N}(y)$. From $Ax + N \sim \mathcal{N}(Ax, R)$, we get $p_{Y|X}(y|x) \propto \exp\left(-\frac{1}{2}(y - Ax)^T R^{-1}(y - Ax)\right)$. In addition, $p_Y(y)$ does not depend on x and $p_X(x) \propto \exp\left(-\frac{1}{2}(x - x_b)^T B^{-1}(x - x_b)\right)$. Therefore, the MAP minimizes $\frac{1}{2}(x - x_b)^T B^{-1}(x - x_b) + \frac{1}{2}(y - Ax)^T R^{-1}(y - Ax)$. □

$$\min_x \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|y - Ax\|^2_{R^{-1}}$$

- The random vectors are assumed Gaussian. However there might be systematic errors making the estimation unreliable.
- Supose $B = \sigma I$ and $A^T R^{-1} A$ is nonsingular. Then $\lim_{\sigma \to +\infty} x_{\text{MAP}} = \left(A^T R^{-1} A\right)^{-1} A^T R^{-1} y$, and we recover the maximum likelihood estimator.
- The useful matrix equality (Sherman-Morrison) $(A + UV^T)^{-1} = A^{-1} - A^{-1} U (I + V^T A^{-1} U)^{-1} V^T A^{-1}$ can be used to show that

$$\begin{aligned} x_{\text{MAP}} &= x_b + \left(B^{-1} + A^T R^{-1} A\right)^{-1} A^T R^{-1} (y - A x_b) \\ &= x_b + B A^T (R + A B A^T)^{-1} (y - A x_b) \quad (*) \end{aligned}$$

These two formula are computationally very different when $A \in \mathbb{R}^{m \times n}$ is such that $m >> n$ or $n >> m$.

- The maximum likelihood estimation works for nonlinear A and leads to

$$\min_x \|A(x) - y\|_{C^{-1}}.$$

- Proof of (*)

Proof.

From the SM formula, we get

$$\begin{aligned}\left(B^{-1} + A^T R^{-1} A\right)^{-1} &= B - BA^T R^{-1}(I + ABA^T R^{-1})^{-1} AB \\ &= B - BA^T (R + ABA^T)^{-1} AB,\end{aligned}$$

which yields

$\left(B^{-1} + A^T R^{-1} A\right)^{-1} A^T R^{-1} = BA^T (R + ABA^T)^{-1}$. □

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Best Linear Unbiased Estimation (BLUE) I

- Consider the linear model $Y = Ax + N$, where x is deterministic and N is a random vector satisfying $E(N) = 0$ and $R = \mathrm{cov}(N)$

Definition

The BLUE estimator for x from Y is the random vector X_{BLUE} which minimizes $J(X) = E(\|X - x\|_2^2)$ subject to $X = KY$ and for all value of x, $E(X) = x$.

Theorem

If A has full rank, the BLUE is $X_{\mathrm{BLUE}} = \left(A^T R^{-1} A\right)^{-1} A^T R^{-1} Y$.

Proof.

Starting from $E(X) = x$, we get $E(KY) = KAx = x$. This equality should hold for any x, i.e. $KA = I$. We have

$$\begin{aligned} J(X) &= E(\mathrm{trace}((X-x)(X-x)^T)) \\ &= E(\mathrm{trace}(KY-x)(KY-x)^T)) \quad (X = KY) \\ &= E(\mathrm{trace}(KN(KN)^T)) \quad (KA = I \text{ and } Y = Ax + N) \\ &= \mathrm{trace}(KRK^T) \quad (\mathrm{cov}(N) = R) \end{aligned}$$

Set $\hat{K} = \left(A^T R^{-1} A\right)^{-1} A^T R^{-1}$, and write $K = \hat{K} + \Delta$. From $(\hat{K} + \Delta)A = I = \hat{K}A$, we get $\Delta A = 0$. Furthermore, $J(KY) = \mathrm{trace}(KRK^T)) =$ $E(\mathrm{trace}(\hat{K}R\hat{K}^T) + \mathrm{trace}(\Delta R \Delta^T) + 2\mathrm{trace}\left(A^T R^{-1} A\right)^{-1} A^T \Delta^T)$, and $\Delta A = 0$ yields $J(KY) = J(\hat{K}Y) + \mathrm{trace}(\Delta R \Delta^T)$. Since R is positive definite, $\mathrm{trace}(\Delta R \Delta^T)$ is a scalar product for $n \times m$ matrices, and $E\mathrm{trace}(\Delta R \Delta^T) = 0$ if and only if $\Delta = 0$. □

Optimal Least mean squares estimation I

We consider the random vector Y defined by $Y = AX + N$, where $E(X) = x_b$, $\mathrm{cov}(X - x_b) = B$ and N is such that $E(N) = 0$ and $\mathrm{cov}(N) = R$. Let us also assume that $E((X - x_b)N^T) = 0$ (uncorrelated pair). For a random vector $\hat{X} = x_b + K(Y - Ax_b)$, we define the error covariance matrix $P(K) = E((\hat{X} - X - E(\hat{X} - X))(\hat{X} - X - E(\hat{X} - X))^T) = \mathrm{cov}(\hat{X} - X)$.

Definition

The optimal least mean squares estimator $X_{LMS} = \hat{K}Y$ is such that

$$a^T P(K) a \geq a^T P(\hat{K}) a,$$

for every matrix K and every vector a. This variational property is written in short $P(K) \geq P(\hat{K})$.

Optimal Least mean squares estimation II I

Theorem

The Optimal Least mean squares is obtained for $\hat{K} = BA^T(ABA^T + R)^{-1}$*. The associated covariance matrix is* $(B^{-1} + A^T R^{-1} A)^{-1}$*.*

Proof.

Optimal Least mean squares estimation II II

We set $X^o = X - x_b$ and $Y^o = Y - Ax_b$. Then

$$\begin{aligned} a^T P(K) a &= aE((X^o - KY^o)(X^o - KY^o)^T) \\ &= a^T (R_x - R_{xy}K^T - KR_{yx} + KR_yK^T)a, \end{aligned}$$

where $R_{xy} = R_{yx}^T = E(X^o Y^{oT})$, $R_x = E(X^o X^{oT})$ and $R_y = E(Y^o Y^{oT})$. Differentiating this expression with respect to aK and setting the derivative to 0 gives to $R_{xy} = KR_y$, i.e. $\hat{K} = R_{xy}R_y^{-1}$. A direct computation shows that $a^T P(K)a = a^T P(\hat{K})a + a^T(K - \hat{K})R_y(K - \hat{K})^T a$, which shows that $K = \hat{K}$ is the unique solution. In addition, $R_x = B$, $R_y = ABA^T + R$, and $R_{xy} = BA^T$, which shows that $\hat{K} = BA^T(ABA^T + R)^{-1}$. Then $P(\hat{K}) = B - BA^T(ABA^T + R)^{-1}A^T B$, and the result follows from the Sherman-Morrison formula. □

Conclusion

- Assume that the random vector Y defined by $Y = AX + N$, where $E(X) = x_b$, $\mathrm{cov}(X - x_b) = B$ and N is such that $E(N) = 0$ and $\mathrm{cov}(N) = R$.
- Under various statistical approaches, if the realization y of Y is available, it is reasonable to estimate x as the minimizer of the quadratic functional

$$\frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|Ax - y\|^2_{R^{-1}}.$$

- The solution of the problem is unique and can be expressed as

$$x_b + \left(B^{-1} + A^T R^{-1} A\right)^{-1} A^T R^{-1}(y - Ax_b),$$

$$\text{or } x_b + BA^T(R + ABA^T)^{-1}(y - Ax_b).$$

Outline

- Introduction
    - Basic concepts
- Data Assimilation from the parameter estimation point of view
    - Preliminary definitions and notations
    - Estimation under the Gaussian assumption
    - Estimation under the linearity assumption.
    - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
    - The problem
    - Steps in duality
    - Solution algorithms
    - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
    - Dual space solvers
    - Preconditioning in dual space
    - Ensemble Kalman filter
    - Domain decomposition solvers

The tracking system

We consider the following system:

- $\forall k \in \mathbb{N}_n$,

$$\boxed{\begin{aligned} Y_k &= \mathcal{H}_k(X_k) + \varepsilon_k^o \\ X_{k+1} &= \mathcal{M}_k(X_k) + \varepsilon_k^m, \end{aligned}} \qquad \begin{aligned} &(1a) \\ &(1b) \end{aligned}$$

- We suppose the following are independent

$$\left\{ (\varepsilon_k^m)_{k\in\mathbb{N}_n},\ (\varepsilon_k^o)_{k\in\mathbb{N}_n},\ X_0 \right\}, \tag{2}$$

- We are interested in the two following problems (filtering and smoothing)

$$\hat{x}_n(y_1..y_n) = \mathrm{argmax}\ \ x_n \mapsto f_{X_n|Y_0..Y_n}(x_n, y_1..y_n) \text{ and}$$

$$\hat{x}_{0:n}(y_1..y_n) = \mathrm{argmax}\ \ x_{0:n} \mapsto f_{X_{0:n}|Y_0..Y_n}(x_{0:n}, y_1..y_n)$$

Bayes formula and Markov property I

Consider $Y = \mathcal{H}(X) + \varepsilon^o$, where X and ε^o are independant. From Bayes rule,

$$f_{X|Y}(x|y) = \frac{f_{Y|X}(y|x)f_X(x)}{\int_{\mathbb{R}} f_{Y|X}(y|x)f_X(x)dx}, \tag{3}$$

Using our assumptions,

$$\boxed{f_{X|Y}(x|y) = \frac{f_{\varepsilon^o}(y - \mathcal{H}(x))f_X(x)}{\int_{\mathbb{R}} f_{Y|X}(y|x)f_X(x)dx}.} \tag{4}$$

Proof: In the continuous case, let

$$\varphi : \begin{pmatrix} X \\ Y \end{pmatrix} \mapsto \begin{pmatrix} X \\ Y - \mathcal{H}(X) \end{pmatrix},$$

Bayes formula and Markov property II

One has $p(B) = P[(X, \epsilon^o) \in \phi(B)] = P[(X, Y) \in B]$. We start from $p(B) = \int_{\varphi(B)} f_{X,\varepsilon^o}(x, \varepsilon^o) dx d\varepsilon^o$. We then have

$$\varphi : \begin{pmatrix} X \\ Y \end{pmatrix} \mapsto \begin{pmatrix} X \\ Y - \mathcal{H}(X) \end{pmatrix}, \ \varphi' : \begin{pmatrix} X \\ Y \end{pmatrix} \mapsto \begin{pmatrix} I & 0 \\ -\mathcal{H}'(X) & I \end{pmatrix},$$

$$det \left((\varphi)' \begin{pmatrix} X \\ Y \end{pmatrix} \right) = 1,$$

From the change of variable formula,

$$\begin{aligned} f_{X,\varepsilon^o}(x, y - \mathcal{H}(x)) \quad &\times \quad |(\varphi)'(x, y - \mathcal{H}(x))| \qquad (5) \\ &= \quad f_{X,Y}(x, y), \qquad (6) \end{aligned}$$

and using the independance of X, ε^o, we get

$$f_{X,\varepsilon^o}(x, y - \mathcal{H}(x)) = f_X(x) f_{\varepsilon^o}(y - \mathcal{H}(x)), \qquad (7)$$

and dividing by $f_X(x)$ yields the result.

The smoothing problem I

- Want to find a recurrence formula to update the conditional density to be able to follow a system on long time ranges.
- Split the algorithm into two parts: Analysis or assimilation of new observations and propagation using the dynamical equations

For brevity $\forall i \in \mathbb{N}_n,\ Z_i := X_0..X_i$, and $W_i := Y_0..Y_i$. For the analysis, one has

$$f[Z_k|W_k](z_k|w_k) = \frac{f[\varepsilon_k^o](y_k - \mathcal{H}_k(x_k)) \times f[Z_k|W_{k-1}](z_k|w_{k-1})}{f[Y_k|W_{k-1}](y_k|w_{k-1})}.$$

Proof:

$$\begin{aligned} f[Z_k|W_k] &= \frac{f[Z_k, Y_k, W_{k-1}]}{f[Y_k, W_{k-1}]} \\ &= \frac{f[Y_k, Z_k, W_{k-1}]}{f[Y_k, W_{k-1}]} \\ &= \frac{f[Y_k|Z_k, W_{k-1}] \times f[Z_k|W_{k-1}] \times f[W_{k-1}]}{f[Y_k|W_{k-1}] \times f[W_{k-1}]} \end{aligned} \tag{8}$$

The smoothing problem II

$$f[Z_k|W_k] = \frac{f[Y_k|Z_k, W_{k-1}] \times f[Z_k|W_{k-1}]}{f[Y_k|W_{k-1}]},$$

by change of variables and independance

$$f[Z_k|W_k](x_k, z_{k-1}|y_k, w_{k-1}) = \frac{f[\varepsilon_k^o](y_k - \mathcal{H}_k(x_k)) \times f[Z_k|W_{k-1}]()}{f[Y_k|W_{k-1}]()}.$$

The smoothing problem III

For the propagation, one has

$$f[Z_k|W_{k-1}](z_k, w_{k-1}) = f[\varepsilon^m_{k-1}](x_k - \mathcal{M}_{k-1}(x_{k-1})) \times f[Z_{k-1}|W_{k-1}],$$

which yields for the complete tracking system

$$\boxed{\begin{aligned} f[Z_n|W_n](z_n|w_n) \propto \prod_{k\leq n} f[\varepsilon^o_k](y_k - \mathcal{H}_k(x_k))\times \\ f[\varepsilon^m_k](x_k - \mathcal{M}_k(x_{k-1})) \end{aligned}} \tag{9}$$

In the case where errors follow Gaussian distributions, we have

$$\boxed{\begin{aligned} f[Z_n|W_n](z_n|w_n) \propto \exp\left\{ \frac{-1}{2} \left(\sum_{k\leq n} ||y_k - \mathcal{H}_k(x_k)||^2_{R_k^{-1}} + \right.\right. \\ \left.\left. ||x_k - \mathcal{M}_k(x_{k-1})||^2_{Q_k^{-1}} \right) \right\} \end{aligned}} \tag{10}$$

The smoothing problem IV

Proof:

$$\begin{aligned} f[Z_k|W_{k-1}] &= f[X_k, Z_{k-1}|W_{k-1}] \\ &= f[X_k|Z_{k-1}, W_{k-1}] \times f[Z_{k-1}|W_{k-1}] \end{aligned}$$

From the system definition, X_k conditionned to (Z_{k-1}, W_{k-1}) only depends on X_{k-1} therefore

$$f[Z_k|W_{k-1}] = f[X_k|X_{k-1}] \times f[Z_{k-1}|W_{k-1}]$$

which yields by change of variables

$$f[Z_k|W_{k-1}](z_k, w_k) = f[\varepsilon^m_{k-1}](x_k - \mathcal{M}_{k-1}(x_{k-1})) \times f[Z_{k-1}|W_{k-1}]$$

Conclusion : the 4D Var functional

- We assume that at t_i, $Y_i = \mathcal{A}(X_i) + N_i$, where $E(X_0) = x_b$, $\mathrm{cov}(X - x_b) = B$, $X_i = \mathcal{M}(t_i, X_0)$, and N_i is such that $E(N) = 0$ and $\mathrm{cov}(N) = R$.
- We are looking for an estimation of x_0 that minimizes

$$\frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{i=0}^{N} \|\mathcal{A}\mathcal{M}(t_i, x_0) - y_i\|^2_{R_i^{-1}}.$$

- The above functional is called the 4D Var functional.

A Data Assimilation experiment

We consider the problem of estimating inital conditions x_0 and $\dot{x}_0$ of the system described by $\ddot{x} + \frac{5}{100}\dot{x} + \alpha x^3 = \frac{15}{2}\cos(t)$, from (possibly noisy) observations of $x(t)$.

- The parameter α controls the nonlinearity of the problem. For $\alpha = 0$, if $\bar{x}$ is a particular solution of the problem for zero initial conditions, all the solutions are expressed by

$$x(t) = \bar{x}(t) + (x_0 + 20\dot{x}_0) - 20\dot{x}_0(0)\exp(-\frac{5}{100}t)$$

- Assume that noisy observations m_i of $x(t)$ are available at t_i.
- We want to minimize the linear least-squares functional

$$\min_{(x_0,\dot{x}_0)} \| \begin{pmatrix} x(t_1) \\ \vdots \\ x(t_m) \end{pmatrix} - \begin{pmatrix} m(t_1) \\ \vdots \\ m(t_m) \end{pmatrix} \|_R.$$

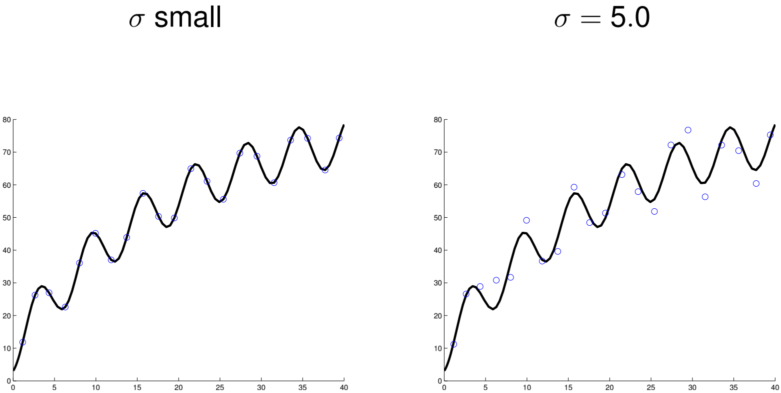
A Data Assimilation experiment

Solving the linear least squares problem ($\alpha = 0$)

- For each observed quantity $m(t_i)$, computation of the linear theoretical counterpart $x(t_i; x_0, \dot{x}_0) = \bar{x}(t_i) + \frac{\partial x(t_i; x_0, \dot{x}_0)}{\partial x_0} x_0 + \frac{\partial x(t_i; x_0, \dot{x}_0)}{\partial \dot{x}_0} \dot{x}_0$
- Solution of the linear least-squares problem, using either a direct method (for problem sizes that are small compared to the computer characteristics) or use e.g. a Conjugate Gradient based iterative solver.
- We take $R = \sigma^2 I$.

Linear case : exact observations (zero noise) v.s. noisy obs.

True trajectory and observations



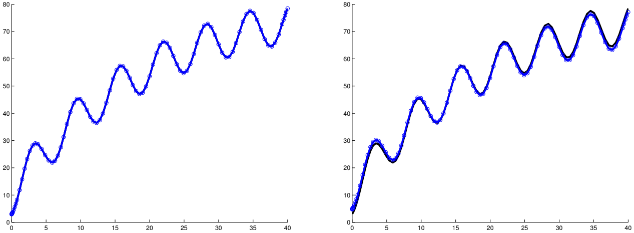
σ small

$\sigma = 5.0$

Linear case : exact observations (zero noise) v.s. noisy obs.

True (solid) and estimated (dotted) trajectories

σ small $\qquad\qquad$ $\sigma = 5.0$



- Values of $(x_0, \dot{x}_0)$: exact $(3.0, 4.0)$, no noise $(3.0, 4.0)$, and noise $(2.3, 4.1)$
- Good forecast even for noisy observations

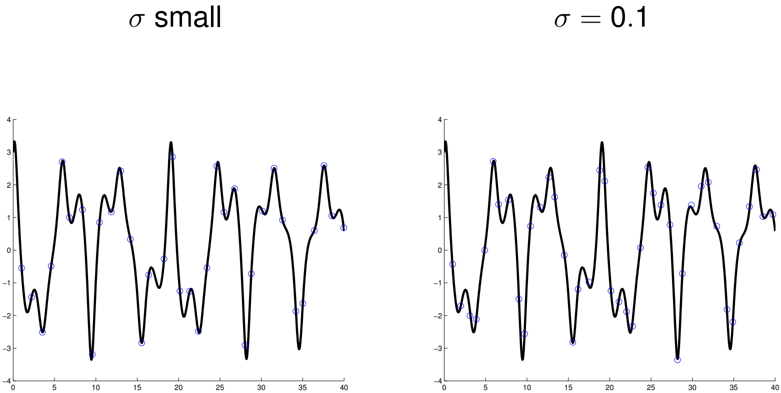
A Data Assimilation experiment

Solving the nonlinear least squares problem ($\alpha = 1$)

- Solution based on linearizations of the dynamics around $(x_0^k, \dot{x}_0^k)$. Starting point $(x_0, \dot{x}_0) = (2.5, 4.5)$
- For each observed quantity $m(t_i)$, computation of the linear theoretical counterpart is $x^k(t_i; x_0^k, \dot{x}_0^k) = x(t_i; x_0^k, \dot{x}_0^k) + \frac{\partial x(t_i; x_0^k, \dot{x}_0^k)}{\partial x_0} \delta x_0 + \frac{\partial x(t_i; x_0^k, \dot{x}_0^k)}{\partial \dot{x}_0} \delta \dot{x}_0$
- Update $(x_0^{k+1}, \dot{x}_0^{k+1}) = (x_0^k, \dot{x}_0^k) + (\delta x_0, \delta \dot{x}_0)$.
- Solution of the linear least-squares problem, using either a direct method (for problem sizes that are small compared to the computer characteristics) or use e.g. a Conjugate Gradient based iterative solver.
- We take $R = \sigma^2 I$.

Non Linear case : exact observations (zero noise) v.s. noisy obs.

True trajectory and observations



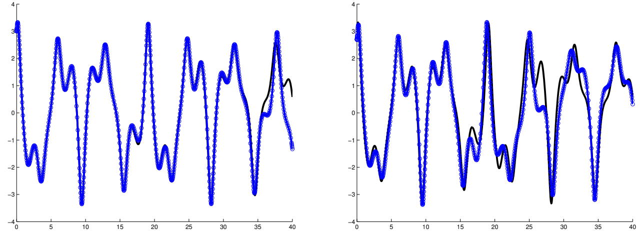
σ small

$\sigma = 0.1$

Non Linear case : exact observations (zero noise) v.s. noisy obs.

True (solid) and estimated (dotted) trajectories

σ small $\sigma = 0.1$



- Values of $(x_0, \dot{x}_0)$: exact $(3.0, 4.0)$, no noise $(3.0, 4.0)$, and noise $(2.3, 4.1)$
- Relative bad forecast for noisy observations and $t > 25$.

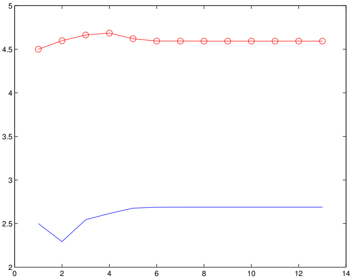
Coping with nonlinearity : guess for the analysis

- The Gauss-Newton algorithm for $\min_x \|f(x)\|$ reads :
- Choose x^k, solve $\min_{\delta x} \|f(x^k) + f'(x_k)\delta x\|$, update $x^{k+1} = x^k + \delta x_k$.
- A critical point of $\phi(x) = \|f(x)\|$ is a point where $\phi'(x) = 0$.
- In the case of nonlinear least-squares problems, the Gauss-Newton algorithm does not converge from any starting point to a critical point.
- Our Data Assimilation solution process will be a ”globalized” variant of the Gauss-Newton algorithm.

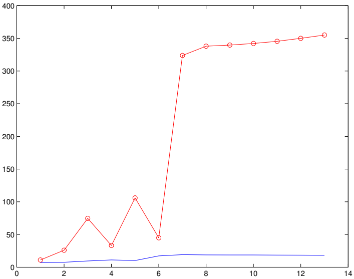
Coping with nonlinearity : convergence histories

Plot the iterates, the solution of the problem without noise is $(3, 4)$. Here the noisy problem is considered.

Starting from $(2.5, 4.5)$



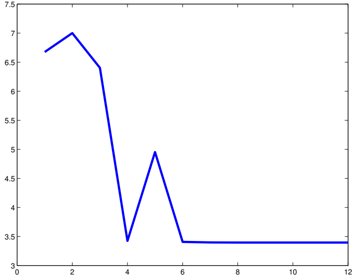
Starting from $(10, 15)$



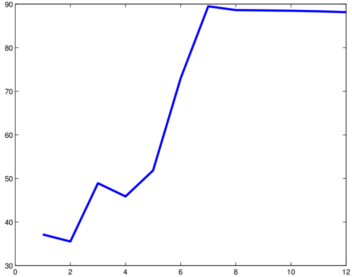
Coping with nonlinearity : residual histories

Plot of the nonlinear least-squares residual

Starting from $(2.5, 4.5)$



Starting from $(10, 15)$



No convergence when the starting point is far from the solution.

The filtering problem

Recall the system equations:

- $\forall k \in \mathbb{N}_n$,

$$\boxed{\begin{aligned} Y_k &= \mathcal{H}_k(X_k) + \varepsilon_k^o \\ X_{k+1} &= \mathcal{M}_k(X_k) + \varepsilon_k^m, \end{aligned}} \qquad \begin{aligned} &(11a) \\ &(11b) \end{aligned}$$

- We suppose the following are independent

$$\left\{(\varepsilon_k^m)_{k\in\mathbb{N}_n},\ (\varepsilon_k^o)_{k\in\mathbb{N}_n},\ X_0\right\}, \tag{12}$$

- We are interested in the following filtering problem

$$\hat{x}_n(y_1..y_n) = \text{argmax}\ \ x \mapsto f_{X_n|Y_0..Y_n}(x, y_1..y_n)$$

The filtering problem I

- Split the algorithm again into two parts: assimilation and

Recall also $\forall i \in \mathbb{N}_n,\ Z_i := X_0..X_i$, and $W_i := Y_0..Y_i$. For the analysis, one has

$$f[X_k|W_k](x_k, w_k) = \frac{f[\varepsilon_k^o](y_k - H_k(x_k))f[X_k|W_{k-1}](x_k, w_{k-1})}{f[Y_k|W_{k-1}](y_k, w_{k-1})}$$

Proof: From Bayes formula,

$$\begin{aligned} f[X_k|W_k](x_k|w_k) &= \frac{f[X_k, W_k](x_k, w_k)}{f[W_k](w_k)} \\ &= \frac{f[X_k, Y_k, W_{k-1}](x_k, y_k, w_k)}{f[Y_k, W_{k-1}](y_k, w_k)} \end{aligned}$$

The filtering problem II

We have by the earlier change of variables

$$\begin{aligned} f[X_k, Y_k, W_{k-1}]() &= f[X_k, \varepsilon_k^o, W_{k-1}](x_k, y_k - \mathcal{H}(x_k), w_{k-1}) \\ &= f[\varepsilon_k^o | X_k, W_{k-1}](y_k - \mathcal{H}_k(x_k) | x_k, w_{k-1}) \times \\ &\quad f[X_k, W_{k-1}](x_k, w_{k-1}) \\ &= f[\varepsilon_k^o | X_k, W_{k-1}](y_k - \mathcal{H}_k(x_k) | x_k, w_{k-1}) \\ &\quad \times f[X_k | W_{k-1}](x_k | w_{k-1}) \times f[W_{k-1}](w_{k-1}) \end{aligned}$$

We then write

$$f[X_k | W_k]() = \frac{f[\varepsilon_k^o | X_k, W_{k-1}](y_k - \mathcal{H}_k(x_k) | x_k, w_{k-1}) \times f[X_k | W_{k-1}]()}{f[Y_k | W_{k-1}](y_k | w_{k-1})}$$

The conclusion follows from the independence of ε_k^o and $(X_k,\ W_{k-1})$ which is a consequence of the system definition, and which can be established by induction.

The filtering problem III

For the propagation, one has

$$f[X_k|W_{k-1}]() = \int f[\varepsilon^m_{k-1}](x_k - M_k(x_{k-1}))f[X_{k-1}|W_{k-1}](x_{k-1}|w_{k-1})dx_k$$

which yields for the complete tracking system

$$\boxed{\begin{aligned} f[X_k|W_k]() &= \frac{f[\varepsilon^o_k](y_k - \mathcal{H}_k(x_k))}{f[Y_k|W_{k-1}]()} \times f[X_k|W_{k-1}](), && \text{(13a)} \\ f[X_k|W_{k-1}]() &= \int f[\varepsilon^m_{k-1}](x_k - \mathcal{M}_k(\xi))\times && \text{(13b)} \\ &\qquad f[X_{k-1}|W_{k-1}](\xi|w_{k-1})d\xi && \text{(13c)} \end{aligned}}$$

The filtering problem IV

Proof: We have that

$$\begin{aligned} f[X_k|W_{k-1}]() &= \frac{f[X_k, W_{k-1}](x_k, w_{k-1})}{f[W_{k-1}](w_{k-1})} \\ &= \frac{1}{f[W_{k-1}]()} \int f[X_k, X_{k-1}, W_{k-1}](., x_{k-1}, .) dx_{k-1}, \end{aligned}$$

but, $\forall x_{k-1}$,

$$\begin{aligned} f[X_k, X_{k-1}, W_{k-1}]() &= f[\varepsilon_k^m, X_{k-1}, W_{k-1}](x_k - \mathcal{M}_k(x_{k-1}), x_{k-1}, w_{k-1} \\ &= f[\varepsilon_k^m | X_{k-1}, W_{k-1}](x_k - \mathcal{M}_k(x_{k-1}) | x_{k-1}, w_{k-1}) \\ &\quad \times f[X_{k-1}|W_{k-1}](x_{k-1}|w_{k-1}) \\ &\quad \times f[W_{k-1}](w_{k-1}) \end{aligned}$$

where we used our change of variables.
Independance of ε_k^m and (X_{k-1}, W_{k-1}) concludes the proof.

Outline

- Introduction
    - Basic concepts
- Data Assimilation from the parameter estimation point of view
    - Preliminary definitions and notations
    - Estimation under the Gaussian assumption
    - Estimation under the linearity assumption.
    - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
    - The problem
    - Steps in duality
    - Solution algorithms
    - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
    - Dual space solvers
    - Preconditioning in dual space
    - Ensemble Kalman filter
    - Domain decomposition solvers

Particule filters for tracking

We are interested in

$$p(x_n|y_{0:n}) \text{ or } p(x_{0:n}|y_{0:n})$$

.

- We represent the estimated densities as

$$\hat{p}(x_n|y_{0:n}) = \sum_{i=1}^{n_p} w_n^i \delta(x_{0:n} - x_{0:n}^i),$$

  where i is the ensemble index, and w_n^i are weights.
- $\sum_{i=1}^{n_p} w_n^i = 1$
- Parameterization in terms of $(w_n^i, x_{0:n}^i)$.
- Use Monte Carlo estimation together with importance sampling to the Bayesian equations
- Challenge when $x_{0:n}^i$ lives in high dimensions.

Importance sampling

- An importance sampling distribution is chosen that satisties

$$q(x_{0:n}|y_{0:n}) = q(x_n|x_{0:n-1}, y_{0:n})q(x_{0:n-1}|y_{0:n})$$

- Bayes rule would rather lead to

$$q(x_{0:n}|y_{0:n}) = q(x_n|x_{n-1}, y_{0:n})q(x_{0:n-1}|y_{0:n-1})$$

- This assumption is made to allow a sequential estimation.

Weight choice

- From importance sampling, and for the joint PDF estimation, $w_n^i \propto \frac{p(x_{0:n}^i|y_{0:n})}{q(x_{0:n}^i|y_{0:n})}$
- Samples $(x_{0:n}^i)$ should be drawn according to the importance sampling density $q(x_{0:n}|y_{0:n})$
- Asuming that the density is given using samples, how to generate de weights? We are looking for a sequential estimation
- We know that $p(x_{0:n}|y_{0:n}) \propto p(y_n|x_n)p(x_n|x_{n-1})p(x_{0:n-1}|y_{0:n-1})$
- Using $q(x_{0:n}|y_{0:n}) = q(x_n|x_{n-1}, y_{0:n})q(x_{0:n-1}|y_{0:n-1})$ we get

$$w_n^i \propto \frac{p(y_n|x_n)p(x_n|x_{n-1})}{q(x_n|x_{n-1}, y_{0:n})} w_{n-1}^i$$

Final expression for the joint distribution

- We have

$$w_n^i \propto \frac{p(y_n|x_n)p(x_n^i|x_{n-1}^i)}{q(x_n^i|x_{n-1}^i, y_{0:n}^i)} w_{n-1}^i$$

- Inital weights have to be chosen
- The density $p(y_n|x_n^i)$ is available from the observation equation $y = h(x) + e$.
- The density $p(x_n^i|x_{n-1}^i)$ is available from the model equation $x_{n+1} = h(x_n) + \varepsilon$.
- We obtain the following representation

$$\hat{p}(x_n \bar{y}_{0:n}) = \sum_{i=1}^{n_p} w_n^i \delta(x_{0:n} - x_{0:n}^i),$$

Final expression for the marginal distribution

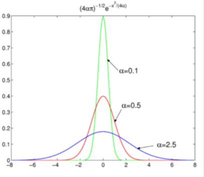
- We have by definition, and using the particular form of $\hat{p}(x_n|y_{0:n})$,

$$\hat{p}(x_n|y_{0:n}) = \int \hat{p}(x_{0:n}|y_{0:n})\, dx_{0:n-1} = \sum_{i=1}^{n_p} w_n^i \delta(x_n - x_n^i)$$

```
Naive Sequential Importance Sampling Algorithm
        For $i = 1 : n_p$ Do
1.          Draw $x_n^i \sim q(x_n|x_{n-1}^i, y_n)$
2.          Compute $\bar{w}_n^i = \frac{p(y_n|x_n)p(x_n|x_{n-1})}{q(x_n|x_{n-1},y_{0:n})} w_{n-1}^i$
        endDo
3.      Normalize $w_n^i = \frac{\bar{w}_n^i}{\sum_{i=1}^{n_p} \bar{w}_n^i}$
```

Representing continous densities

- We can ”smooth” the estimated density using for instance Gaussian distributions $\hat{p}(x_n \bar{y}_{0:n}) = \sum_{i=1}^{n_p} w_n^i \delta(x_{0:n} - x_{0:n}^i)$ represented by a sum of Gaussian



Needed when graphical representation of the density is needed.

Suffers from curse of dimensionality in high dimension. Good in 1 or 2 dimensions.

Particle filter in practice

- The performance of the algorithm is not good in general for high dimensions
- Most weights go to zero and few trajectory contribute to the density estimation
- Computations of particles with low weights is useless, and is a waste of computational power.
- Reasons for this behaviour
  - The particules are in a high dimensional space
  - The weightes are done by products, and whenever less probable states are encountered weights are getting really small
  - This maybe seen as an effect of curse of dimensionality: it is hard to sample efficiently
- This is the degeneracy phenomenon that must by tackled for practical applications

Tackle with degenerecence

- The phenomenon can be tracked using the effective sample size $\frac{1}{\sum_1^{n_p}(w_n^i)^2}$
- The key idea is resampling
- Eliminate samples with low weight and replace them (copy) with samples with high weights. Sampling with replacement.
- The new particules are drawn with probability given by the weigth using for instance the roulette wheel technique
- Often uniform weights are assigned to the resampled particles : $\frac{1}{n_p}$
-

Outline

Introduction

- Basic concepts

Data Assimilation from the parameter estimation point of view

- Preliminary definitions and notations
- Estimation under the Gaussian assumption
- Estimation under the linearity assumption.
- Filtering the system in full generality

Particle filter in a nutshell

Data Assimilation form the optimization point of view

- The problem
- Steps in duality
- Solution algorithms
- Sequential estimation and Kalman filter

Data assimilation from the linear algebra point of view

Specialized solvers

- Dual space solvers
- Preconditioning in dual space
- Ensemble Kalman filter
- Domain decomposition solvers

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

A nonlinear least squares problem

We define $x_0 \mapsto F(x_0)$ by

$$F(x_0) = \frac{1}{2} \begin{bmatrix} B^{-1/2}(x_0 - x_b) \\ R_0^{-1/2}\left(\mathcal{H}_0(x_0) - y_0\right) \\ \vdots \\ R_n^{-1/2}\left(\mathcal{H}_n(x_n) - y_n\right) \end{bmatrix},$$

then the minimization of the 4D Var functional reads $\min_{x_0} \|F(x_0)\|_2^2$. this is an unconstrained nonlinear least-squares problem.
Let $J(x)$ be the Jacobian matrix of $F(x)$ and, whenever $J(x)$ is differentiable, let $H(x)$ be the Hessian matrix of $F(x)$.

Existence and unicity of solutions

- Differentibility of F assumed.
- If $x_0 \mapsto \mathcal{H}_i(x_i) = \mathcal{H}_i(\mathcal{S}(t_i, t_0, x_0))$ (or if F) is an affine function of x_0 the optimization problem is a full rank overdetermined linear least squares problem. The solution corresponds to a linear system of equations (the normal equations).
- If the problem is non linear then existence and unicity are not guaranteed in general. An iterative algorithm has to be used in general.

Derivatives for the nonlinear least squares functional

- Let $\phi(x_0) = \|F(x_0)\|^2$
- $\nabla\phi(x_0) = J(x_0)^T F(x_0)$
- $\nabla^2\phi(x_0) = J(x_0)^T J(x_0) + S(x_0)$, $S(x_0) = \sum_{i=1}^{m} F_i(x_0) F_i''(x_0)$
- The necessary condition for optimality $J(x)^T F(x) = 0$. A point that satisfies this condition is a (first order) critical point.

Geometrical interpretation

- Minimum distance from the origin to the surface de $\mathbb{R}^m$, $\mathcal{S} \ : \ z = F(x)$
- For a critical point x such that $F(x) \neq 0$, $F(x)$ orthogonal to $\text{Im}(J(x))$ and, if $J(x)$ has full column rank,
  - the matrix $K(x) = \frac{1}{\|F(x)\|_2} J(x)^{+T} \sum_{i=1}^{m} F_i(x) F_i''(x) J(x)^{+}$ is the principal curvature matrix associated to $\mathcal{S}$ wrt the normal direction $F(x)$
  - Let $\kappa_1 \geq \cdots \geq \kappa_n$ be the eigenvalues of K. The quantites $\rho_i = \frac{1}{\kappa_i}$ are the principal curvature radii wrt to the normal direction $F(x)$
- We then have $\nabla^2 \phi(x) = J(x)^T (I - \|F(x)\|_2 K(x)) J(x)$

Critical points and extrema

- Let x be a critical point, and assume $J(x)$ has full colummn rank and $F(x) \neq 0$
- $\nabla^2 \phi(x)$ is symmetric positive definite iff $1 - \|F(x)\|_2 \kappa_1 > 0$. Then x is a local minimum of $\|F(x)\|_2$. This is the second order sufficient optimality condition.
- If $1 - \|F(x)\|_2 \kappa_n < 0$, then x is a local maximum of F
- If x is a local min, then $1 - \|F(x)\|_2 \kappa_n \geq 0$. This is the $2nd$ order necessary condition of optimality.
- A nonlinear least squares problem may have
  - no local minimum (ex: $x \mapsto \exp(2x)$)
  - many local minima (ex: $x \mapsto \sin(x)^2$)

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Saddle point and augmented Lagrangian

- Saddle point (SP) associated with a function $l(x, y) : X \times Y \to \mathbb{R}$ is a point $(\bar{x}, \bar{y})$ such that

$$l(\bar{x}, u) \leq l(\bar{x}, \bar{y}) \leq l(x, \bar{y}) \ (SP1)$$

- An equivalent definition of a SP is

$$\max_{y \in Y} l(\bar{x}, y) = l(\bar{x}, \bar{y}) = \min_{x \in X} l(x, \bar{y}) \ (SP2).$$

- For the minimization problem

$$\min_{\epsilon, x; \epsilon = Hx - y} \quad \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}},$$

- we consider the SP of the augmented Lagrangian associated with the optimization problem $L(x, \epsilon; \lambda) = \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}} + < \lambda, \epsilon + y - Hx >$

Convexity

- For a convex differentiable function f we consider the convex optimization problem

$$(\mathcal{P}) \ \min_{Ax=b} f(x)$$

- We have $L(x, \lambda) = f(x) + < \lambda, b - Ax >$
- Theorem: The saddle points of L are exactly the points $(\bar{x}, \bar{\lambda})$ such that
  1. $\bar{x}$ is a solution of $(\mathcal{P})$, and
  2. $\nabla f(\bar{x}) + A^T\bar{\lambda} = 0$ and $b = A\bar{x}$.
- We consider the saddle point problem for the Lagrangian of the data assimilation problem.
- From definition SP2, we introduce the direct problem $\sup_\lambda L(x, \epsilon; \lambda)$ and the adjoint problem $\inf_{x,\epsilon} L(x, \epsilon; \lambda)$.

The direct problem

From $L(x,\epsilon;\lambda) = \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}} + < \lambda, \epsilon + y - Hx >$, we get

$$\phi(x,\epsilon) = \sup_{\lambda} L(x,\epsilon;\lambda) = \begin{cases} +\infty \text{ if } \epsilon + y - Hx \neq 0 \\ \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}} \text{ if } \epsilon + y - Hx = 0 \end{cases}$$

This yields the infsup result

$$\inf_{x,\epsilon} \sup_{\lambda} L(x,\epsilon;\lambda) = \inf_{\epsilon, x; \epsilon = Hx - y} \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}},$$

where the inf is a min by convexity. Direct explicitation of the constraint leads to the problem $\inf_x \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|Hx - y\|^2_{R^{-1}}$ whose solution is given by the normal equations

$$\bar{x} = x_b + \left(B^{-1} + H^T R^{-1} H\right)^{-1} H^T R^{-1}(y - Hx_b).$$

The adjoint problem

Consider the infimum problem

$$\inf_{x,\epsilon} L(x,\epsilon;\lambda) = \inf_{x,\epsilon} \frac{1}{2}\|x - x_b\|^2_{B^{-1}} + \frac{1}{2}\|\epsilon\|^2_{R^{-1}} + < \lambda, \epsilon + y - Hx > .$$

The problem is convex differentiable. Zeroing the partial derivative wrt x and ϵ, we get

$$\begin{cases} B^{-1}(x - x_b) - H^T\lambda = 0 \\ R^{-1}\epsilon + \lambda = 0 \end{cases} \text{, which yields}$$

which yields

$$\begin{cases} x = x_b + BH^T\lambda \\ \epsilon = -R\lambda \end{cases} .$$

Keeping in mind SP2, we consider

$$\sup_{\lambda} L(x,\epsilon;\lambda) = \sup_{\lambda} \frac{1}{2}\|\lambda\|^2_R + \frac{1}{2}\|H^T\lambda\|^2_B + < \lambda, -R\lambda - HBH^T\lambda + y - Hx_b >$$

that leads to the adjoint maximization problem
$\sup_\lambda L(x,\epsilon;\lambda) = \sup_\lambda -\frac{1}{2}\|\lambda\|^2_R - \frac{1}{2}\|H^T\lambda\|^2_B + < \lambda, y - Hx_b >$.
The solution is $\bar{\lambda} = (R + HBH^T)^{-1}(y - Hx_b)$ that yields
$\bar{x} = x_b + BH^T(R + HBH^T)^{-1}(y - Hx_b)$

Outline

- Introduction
    - Basic concepts
- Data Assimilation from the parameter estimation point of view
    - Preliminary definitions and notations
    - Estimation under the Gaussian assumption
    - Estimation under the linearity assumption.
    - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
    - The problem
    - Steps in duality
    - Solution algorithms
    - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
    - Dual space solvers
    - Preconditioning in dual space
    - Ensemble Kalman filter
    - Domain decomposition solvers

Newton method

- Depends on the problem size. for reasonable size problems (machine?) it requires the solution of $\nabla^2\phi(x_k)\delta x_k = \nabla\phi(x_k)$, $x_{k+1} = x_k - \delta x_k$
- It converges locally quadratically to a critical point $\bar{x}$ under some assumptions ($\nabla^2\phi(\bar{x})$ is Lipschitz continuous around $\bar{x}$ and $\nabla^2\phi(x)$ is SPD)
- It is possible to make the convergence global under mild assumptions with trust-region techniques. Linesearch is also possible.
- If feasible (size, computational cost) this is a method of choice.
- Impraticable for the ocean and atmosphere data assimilation problems.

Quasi-Newton Method

- Several variantis available (SR1, PSB, DFP, BFGS,..)‘
- For BFGS, the matrice $\nabla^2\phi(x)$ is not computed, but its inverse H_k is approximated using the secant equation : $s_k = H_k y_k$, $s_k = x_{k+1} - x_k$, $y_k = \nabla\phi(x_{k+1}) - \nabla\phi(x_k)$.
- The matrix H_{k+1} is the closest matrix to H_k for some weighted norm
- Updating formula $H_{k+1} = (I - \rho_k s_k y_k^T)^T H_k (I - \rho_k s_k y_k^T) + \rho_k s_k s_k^T$, $\rho_k = \frac{1}{y_k^T s_k}$
- Limited memory implementation : H_{k+1} is kept in factored form and only the last pairs are taken into account $(y_i, s_i)_{i=1,\dots k}$. Example : L-BFGS, M1QN3.

Quasi-Newton method for data assimilation

- Gradient computation $\nabla \mathcal{J}(x_0) = B^{-1}(x_0 - x_b) + \sum_{i=0}^{n} M_{t_i,t_0}^T H_i^T R_i^{-1} \left(\mathcal{H}_i(x(t_i^k)) - y_i\right),$
- M_{t_i,t_0} and H_i are the jacobian matrices of $x_0 \mapsto \mathcal{M}(t, t_0)x_0$ and $x \mapsto H_i(x)$ at x_0 and at $x(t) = \mathcal{M}(t, t_0)x_0$
- Computing, storing at all step M_{t_i,t_0} is not feasible in large scale data assimilation

The Gauss-Newton algorithm

- Solution of a sequence of linear least squares problems $\min \|J(x_k)s_k + F(x_k)\|$ and update $x_{k+1} = x_k - s_k$
- It can be seen as a stationary iteration $x_{k+1} = G(x_k) = x_k - J(x_k)^\dagger F(x_k)$
- Does not require second order information. The linear least-squares problem can be solved by a *direct* or an *iterative* method (truncated Gauss-Newton method).

Local convergence properties of the Gauss-Newton algorithm

- The algorithm is not globally convergent. It is not locally convergent either : for some problems, a fixed point may be a repelling fixed point.
  - Consider $F(x) = (x + 1, -\lambda x^2 + x - 1)$.
  - the unique local minimum is $x = 0$.
  - It is possible to show that $x_{k+1} = -\lambda x_k + o(x_k)$
  - Therefore, if $|\lambda| > 1$, 0 is a repelling fixed point.

Example of nonlocal convergence of Gauss-Newton

The incremental 4D Var algorithm (Gauss-Newton)

- take $x_0^0 = x_b$
- For $i = 0, 1, 2, \dots$ solve iteratively the quadratic minimization problem
  $\min_{\delta x} S(\delta x)$, with

$$
\begin{aligned}
S(\delta x) &= \frac{1}{2}\|\delta x + x_0^k - x_b\|_{B^{-1}}^2 \\
&+ \frac{1}{2}\sum_{k=0}^{N} \left\| H_k M_{t_k,t_0}\delta x + \mathcal{H}_k(x_k^i) - y_k \right\|_{R_k^{-1}}^2
\end{aligned}
$$

- update $x_0^{k+1} = x_0^k + \delta x$
- End For

where we set $x_k^i = \mathcal{M}_k(x_k)$ and $M_{t_i,t_0}\delta x = \frac{\partial \mathcal{M}_k(x_k)}{\partial x_0}\delta x$.

Outline

- Introduction
    - Basic concepts
- Data Assimilation from the parameter estimation point of view
    - Preliminary definitions and notations
    - Estimation under the Gaussian assumption
    - Estimation under the linearity assumption.
    - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
    - The problem
    - Steps in duality
    - Solution algorithms
    - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
    - Dual space solvers
    - Preconditioning in dual space
    - Ensemble Kalman filter
    - Domain decomposition solvers

Optimality transfer for observations

- We consider $J_1(x_0) = \frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\|y_1 - H_1 x_0\|^2_{R_1^{-1}}$ and $J_{12}(x_0) = \frac{1}{2}\|y_2 - H_2 x_0\|^2_{R_2^{-1}}$.
- We set $J_2 = J_1 + J_{12}$
- Is it possible to update de minimum argument x_0^1 of J_1 to get the minimum argument x_0^2 of J_2?
- Application. Observations are obtained repeatly and we want to update the estimate accordingly, without processing all the information from the beginning.
- Note that from Theorem 19, the minimum covariance matrix for the estimation of x_0^1 is
  $P_1 = \left(B^{-1} + H_1^T R_1^{-1} H_1\right)^{-1}$ and
  $x_0^1 = x_b + P_1 H_1^T R_1^{-1}(y_1 - H_1 x_b)$

Theorem

For some constant $c \in \mathbb{R}$, we have $J_2(x_0) = c + \frac{1}{2}\|x_0 - x_0^1\|^2_{P_1^{-1}} + \frac{1}{2}\|y_2 - H_2 x_0\|^2_{R_2^{-1}}$. *Therefore,* $x_0^2 = x_0^1 + P_2 H_2^T R^{-1}(y_2 - H_2 x_0^1)$*, where* $P_2 = \left(P_1^{-1} + H_2^T R^{-1} H_2\right)^{-1}$.

Proof.

Since x_0^1 minimizes $J_1(x_0)$, we have the Taylor expansion $J_1(x_0) = J_1(x_0^1) + \frac{1}{2}(x_0 - x_0^1)^T \nabla^2 J_1(x_0^1)(x_0 - x_0^1)$. But $J_1(x_0)$ is quadratic, therefore, $\nabla^2 J_1(x_0^1) = B^{-1} + H_1^T R_1^{-1} H_1$, and we have $J_1(x_0) = J_1(x_0^1) + \frac{1}{2}\|x_0 - x_0^1\|_{P_1^{-1}}$, which yields the result. □

This Theorem shows how to update incrementaly the estimation for new coming observations. Note that the algorithm updates the solution and the covariance matrix.

Incremental least squares algorithm

We want to find x_0 that minimizes $J(x_0) = \frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N}\|y_k - H_k x_0\|^2_{R_k^{-1}}$. The following algorithm finishes with $z_N = x_0$.

```
Incremental least squares
1.  Set $z_0 = x_b$ and $P_0 = B$
2.  For k=0,2, ... N Do
3.    Compute $P_{k+1} = (P_k^{-1} + H_k^T R_k^{-1} H_k)^{-1}$
4.    Compute $z_{k+1} = z_k + P_{k+1} H_k^T R_k^{-1}(y_k - H_k z_k)$
8.  EndDo
```

Note that this algorithm may be unstable in the presence of rounding-errors (on a computer) and that alternative formulations based on orthogonal transformations are available.

Optimality transfer for model errors

- We consider $J_1(x_0, x_1) = \frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\|x_1 - Fx_0\|^2_{Q^{-1}}$ and $J_{12}(x_1) = \frac{1}{2}\|y - H_1 x_1\|^2_{R_1^{-1}}$.
- We set $J_2 = J_1 + J_{12}$.
- Is it possible to update the minimum argument $x_1^1 = Fx_b$ of J_1 to get the minimum argument x_1^2 of J_2?
- Application. Now we get observations from an object that is moving. The position of the object at t_i is x_i, and we would like to know the current position at t_1 of the object "knowing" an approximate dynamics $x_1 = Fx_0 + e_1$ and noisy observations $y = H_1 x_1 + e_1$.

Theorem

For some constant $c \in \mathbb{R}$, some vector $g \in \mathbb{R}^n$, and some matrix G, we have $J_2(x_0, x_1) = \|x_0 + Gx_1 + g\|^2_{B^{-1}+F^TQF} + \frac{1}{2}\|x_1 - Fx_b\|^2_{P_{0|1}^{-1}} + \frac{1}{2}\|y - Hx_1\|^2_{R^{-1}}$.

Therefore, $x_1^2 = Fx_b + P_2 H^T R^{-1}(y_1 - HFx_b)$, where $P_2 = \left(P_{0|1}^{-1} + H^T R^{-1} H\right)^{-1}$ and $P_{0|1} = Q + FBF^T$.

Proof From bloc Gaussian elimination, we have

$$\begin{pmatrix} B^{-1} + F^TQF & -F^TQ^{-1} \\ -Q^{-1}F & Q^{-1} \end{pmatrix}$$

$$= \begin{pmatrix} I & 0 \\ G^T & I \end{pmatrix} \begin{pmatrix} B^{-1} + F^TQF & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & G \\ 0 & I \end{pmatrix},$$ where $S = Q^{-1} - Q^{-1}F(B^{-1} + F^TQF)^{-1}F^TQ^{-1} = (Q + FBF^T)^{-1} = P_{0|1}^{-1}$, and $G = -(B^{-1} + F^TQ^{-1}F)^{-1}F^TQ^{-1}$.

We have

$J_1(x_0, x_1) =$

$$\frac{1}{2}\begin{pmatrix} x_0 - x_b \\ x_1 - Fx_b \end{pmatrix}^T \begin{pmatrix} B^{-1} + F^TQF & -F^TQ^{-1} \\ -Q^{-1}F & Q^{-1} \end{pmatrix} \begin{pmatrix} x_0 - x_b \\ x_1 - Fx_b \end{pmatrix}.$$

We denote $g = -(I + GF)x_b$, and get $J_1(x_0, x_1) = \frac{1}{2}\|x_0 + Gx_1 + g\|^2_{B^{-1}+F^TQF} + \frac{1}{2}\|x_1 - Fb_b\|_{P_{0|1}^{-1}}$. The conclusion follows from the fact that the minimum for J_2 is obtained for $x_0 = -Gx_1 - g$, and Theorem 19 provides an expression for x_1 minimizing $\frac{1}{2}\|x_1 - Fx_b\|^2_{P_{0|1}^{-1}} + \frac{1}{2}\|y - Hx_1\|^2_{R^{-1}}$.

Kalman filter

We want to find x_N such that $(x_1, \ldots x_N)$ minimizes $J(x_0, \ldots, x_N) = \frac{1}{2}\|x_0 - x_b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N} \|y_k - H_k x_k\|^2_{R_k^{-1}} + \frac{1}{2}\sum_{k=0}^{N} \|x_{k+1} - F_k x_k\|^2_{Q_k^{-1}}$. The following algorithm finishes with $z_N = x_N$.

```
Kalman filter
1.  Set $z_0 = x_b$ and $P_{0|-1} = B$
2.  For k=0,2, ... N Do
3.    Compute $P_k = (P_{k|k-1}^{-1} + H_k^T R_k^{-1} H_k)^{-1}$
4.    Compute $z_k = z_{k|k-1} + P_k^{-1} H_k^T R_k^{-1} (y_k - H_k z_{k|k-1})$
5.    Compute $P_{k+1|k} = Q_k + F_k P_k F_k^T$
6.    Compute $z_{k+1|k} = F_k z_k$
7.  EndDo
```

Again, algorithm may be unstable in the presence of rounding-errors (on a computer) an use orthogonal transformations recommended.

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

The Incremental 4D-Var

- In the Incremental 4D Var approach, we have to solve a series of linear systems $(B^{-1} + H_i^T H_i)\delta x = H_i^T d_i$.
- the linear systems are large (10^6) and dense.
- For large problems (atmosphere DA, ocean DA), solving these linear systems is the dominant computational part of the algorithm

  $\Longrightarrow$ need for efficient methods (memory, cpu).

Linear system occuring in the method will be denoted $A_i x = b_i$, or $Ax = b$ if no-confusion possible.

Two main approaches:

- direct solvers:
  compute a factorization and use the factors to solve the linear system,
- iterative solvers:
  build a sequence $(x_k) \rightarrow x^*$.

Direct solvers

Express the matrix A as the product of matrices having simple structures (i.e. diagonal, triangular).
Example: LU for unsymmetric matrices, LL^T (Cholesky) for symmetric positive definite matrices, LDL^T for symmetric indefinite matrices.

Advantages

- accurate - backward stable,
- perform the factorization once but use it for many solves.

Drawbacks

- complex for sparse matrices,
- can become prohibitive (CPU, memory) for large problems.

Why libraries should be preferred to numerical recipies ?

Solution of an SPD linear system

Size	500	1000	1500
Num. Recip.	2.9	30.2	106
BLAS 1	0.4	12.0	45
BLAS 3	0.2	1.4	5

Elapsed time (sec) on a SGI 02K

Sparse matrices

The effect of the ordering

$$A = \begin{pmatrix} x & x & x & x \\ x & x & & \\ x & & x & \\ x & & & x \end{pmatrix} = \begin{pmatrix} x & & & \\ x & x & & \\ x & x & x & \\ x & x & x & x \end{pmatrix} \begin{pmatrix} x & x & x & x \\ & x & x & x \\ & & x & x \\ & & & x \end{pmatrix}$$

while

$$PAP^T = \begin{pmatrix} x & & & x \\ & x & & x \\ & & x & x \\ x & x & x & x \end{pmatrix} = \begin{pmatrix} x & & & \\ & x & & \\ & & x & \\ x & x & x & x \end{pmatrix} \begin{pmatrix} x & & & x \\ & x & & x \\ & & x & x \\ & & & x \end{pmatrix}$$

$\Rightarrow$ symbolic analysis.

When pivoting is required : e.g. the unsymmetric case

Trade-off between

- preserve sparsity,
- preserve stability (pivot selection).

Alternative to direct solvers when memory and/or CPU constraints.

Two main approaches

- Stationary/asymptotic method: $x_k = f(x_{k-1})$ with $x_k \to x^* \ \forall x_0$.
- Krylov method: $x_k = x_0 + span\{r_0, Ar_0, .., A^{k-1}r_0\}$ with $r_k = b - Ax_k$ subject to some constraints/optimality conditions.

Advantages

- cheap both in memory and in CPU,
- might be stop at suitable accuracy.

Drawbacks

- may converge slowly or diverge,
- often many parameters to tune.

Stopping criteria based on backward error analysis

The error introduced during the computation are interpreted in terms of perturbations of the initial data, and the computed solution is considered as exact for the perturbed problem. The backward error measures in fact the distance between the data of the initial problem and those of the perturbed problem. Let x_j be an approximation of the solution $x = A^{-1}b$. Then

$$\begin{aligned}\eta_j &= \min\{\varepsilon > 0;\ \|\Delta A\| \le \varepsilon\|A\|_2,\ \|\Delta b\| \le \varepsilon\|b\|_2 \\ &\qquad \text{and } (A+\Delta A)x_j = b + \Delta b\} \\ &= \frac{\|b - Ax_j\|_2}{\|A\|_2\|x_j\|_2 + \|b\|_2}\end{aligned}$$

is called the *normwise backward error* associated with x_j.
If perturbations are only considered on b or if $\|A\|_2$ is unknown we can also consider

$$\begin{aligned}\eta_j &= \min\left\{\varepsilon > 0;\ \|\Delta b\| \le \varepsilon\|b\|_2 \text{ and } Ax_j = b + \Delta b\right\} \\ &= \frac{\|b - Ax_j\|_2}{\|b\|_2}.\end{aligned}$$

Basic scheme

Let x_0 be given and $B \in \boldsymbol{R}^{n\times n}$ a nonsingular matrix, compute

$$x_k = x_{k-1} + B(b - Ax_{k-1}). \tag{14}$$

Note that $b - Ax_{k-1} = A(x_* - x_{k-1}) \Rightarrow$ the best B is A^{-1}.

Theorem

The stationary sheme defined by (14) converges to $x^* = A^{-1}b$ for any x_0 iff $\rho(I - BA) < 1$, where $\rho(I - BA)$ denotes the spectral radius of the iteration matrix $(I - BA)$.

Some well-known schemes

Depending on the choice of B we obtain some of the best known stationary methods. Let decompose $A = L + D + U$, where L is lower triangular part of A, U the upper triangular part and D is the diagonal of A.

- $B = I$: Richardson method,
- $B = D^{-1}$: Jacobi method,
- $B = (L + D)^{-1}$: Gauss-Seidel method.

Notice that B has always a special structure and inverse must never been explicitely computed. $z = M^{-1}y$ reads *solve the linear system* $Mz = y$.

Krylov methods: Motivations

For the sake of simplicity of exposure, we assume $x_0 = 0$. This does not mean a loss of generality, because the situation $x_0 \neq 0$ can be transformed with a simple shift to the system

$$Ay = b - Ax_0 = \bar{b},$$

for which obviously $y_0 = 0$. The minimal polynomial $q(t)$ of A is the unique monic polynomial of minimal degree such that $q(A) = 0$. It is constructed from the eigenvalues of A as follows. If the distinct eigenvalues of A are $\lambda_1, ..., \lambda_\ell$ and if λ_j has index m_j (the size of the largest Jordan block associated with λ_j), then the sum of all indices is

$$m = \sum_{j=1}^{\ell} m_j, \text{and } q(t) = \prod_{j=1}^{\ell} (t - \lambda_j)^{m_j}. \tag{15}$$

When A is diagonalizable m is the number of distinct eigenvalues of A; when A is a Jordan block of size n, then $m = n$.

If we write

m

The Krylov subspace approach

The Krylov methods for identifying $x_k \in \mathcal{K}(A, b, k)$ can be distinguished in four classes

- The Ritz-Galerkin approach:
  construct x_k such that $b - Ax_k \perp \mathcal{K}(A, b, k)$. Conjugate gradient method
- The minimum norm residual approach:
  construct $x_k \in \mathcal{K}(A, b, k)$ such that $||b - Ax_k||_2$. is minimal
- The Petrov-Galerkin approach:
  construct x_k such that $b - Ax_k$ is orthogonal to some other k-dimensional subspace.
- The minimum norm error approach:
  construct $x_k \in A^T\mathcal{K}(A, b, k)$ such that $||b - Ax_k||_2$ is minimal.

Constructing a basis of $\mathcal{K}(A, b, k)$

- The obvious choice $b, Ab, ...A^{k-1}b$ is not very attractive from a numerical point of view since the vectors A^jb becomes more and more colinear to the eigenvector associated with the dominant eigenvalue. In finite precision calculation, this leads to a lost of rank of this set of vectors.
- A better choice is to use the Arnoldi procedure.

The Arnoldi procedure

This procedure builds an orthonormal basis of $\mathcal{K}(A, b, k)$.

Arnoldi's algorithm

```
$v_1 = b/\|b\|$
For j=1,2, ... k-1 Do
  Compute $h_{ij} = v_i^T A v_j$ for $i = 1, j$
  Compute $w_j = Av_j - \sum_{i=1}^{j} h_{ij} v_i$
  $h_{j+1,j} = \|w_j\|$
  If $(h_{j+1,j} = 0)$ then Stop
  $v_{j+1} = w_j / h_{j+1,j}$
EndDo
```

The Arnoldi procedure properties

Proposition

If the Arnoldi procedure does not stop before the k^{th} step, the vectors $v_1, ..., v_k$ form an orthonormal basis of the Krylov subspace $\mathcal{K}(A, b, k-1)$

The CG algorithm (A is spd and large !)

- CG is an iterative method for solving

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}x^TAx - b^Tx \qquad or \qquad Ax = b$$

- Iterations: Given $x_0 \in \mathbb{R}^n$; $A \in \mathbb{R}^{n\times n}$; $b \in \mathbb{R}^n$
  - Set $r_0 \leftarrow Ax_0 - b_0$; $p_0 \leftarrow -r_0$; $i \leftarrow 0$
  - Loop on i

$$\begin{array}{lcl} \alpha_i & \leftarrow & (r_i^T r_i)/(p_i^T A p_i) \\ x_{i+1} & \leftarrow & x_i + \alpha_i p_i \\ r_{i+1} & \leftarrow & r_i + \alpha_i A p_i \\ \beta_{i+1} & \leftarrow & (r_{i+1}^T r_{i+1})/(r_i^T r_i) \\ p_{i+1} & \leftarrow & -r_{i+1} + \beta_{i+1} p_i \end{array}$$

- r_i are residuals; p_i are descent directions; $\alpha_i p_i$ are steps.

The CG properties (in exact arithmetic !)

- Orthogonality of the residuals: $r_i^T r_j = 0$ if $i \neq j$.
- *A*-conjugacy of the directions: $p_i^T A p_j = 0$ if $i \neq j$.
- The distance of the iterate x_i to the solution x^* is related to the condition number of A, denoted by $\kappa = \frac{\lambda_{max}}{\lambda_{min}}$ (≥ 1):

  $$||x_i - x^*||_A \leq \eta_i \; ||x_0 - x^*||_A \quad \text{with} \quad \eta_i = 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^i$$

  $\Rightarrow$ The smaller $cond(A) \equiv \kappa$ is, the faster the convergence.
- Exact solution found exactly in r iterations, where $r \leq n$ is the number of distinct eigenvalues of $A \in \mathbb{R}^{n \times n}$.
  $\Rightarrow$ The more clustered the eigenvalues are, the faster the convergence.

Why ”preconditionning”?

The number of iterations of the Krylov solver is related to the spectrum of the matrix A:

- The convergence is fast for matrices with few distinct eigenvalues in their spectrum.
- For CG we have the well-known upper bound on the convergence rate

$$\|x_k - x^*\|_A \leq 2 \cdot \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k \|x_0 - x^*\|_A.$$

- The closer A is from I, the faster the convergence.

$\Rightarrow$ solve an equivalent linear system that has a better eigenvalue distribution: $M^{1/2}AM^{1/2}x = M^{1/2}b$ with M symmetric and positive definite.

The preconditioner constraints

The preconditioner should

- be cheap to generate and to store,
- be cheap to apply,
- ensure a fast convergence.

With a good preconditioner the computing time for preconditioned solution should be less than for the unpreconditioned one.

The particular case of CG

For CG let M be given in a factored form (i.e. $M = CC^T$), then CG can be applied to

$$\tilde{A}\tilde{x} = \tilde{b},$$

with $\tilde{A} = C^T AC$, $C\tilde{x} = x$ and $\tilde{b} = C^T b$.
Let define:

$$\begin{array}{rcl} x_k & = & C\tilde{x}_k, \\ C\tilde{p}_k & = & p_k, \\ \tilde{r}_k & = & C^T r_k, \\ z_k & = & CC^T r_k, \end{array}$$

Note that $\tilde{x}_k - \tilde{x}_0 = \sum \gamma_i \tilde{A}^i \tilde{r}_0$ writes
$x_k - x_0 = \sum \gamma_i C(C^T AC)^i C^T r_0$.

Using $\tilde{A} = C^T AC, x_k = C\tilde{x}_k, p_k = C\tilde{p}_k, r_k = C^T\tilde{r}_k$ we can write the CG algorithm for both the preconditioned variables and the unpreconditioned ones.

```
Conjugate Gradient algorithm
1.  Compute $\tilde{r}_0 = \tilde{b} - \tilde{A}\tilde{x}_0$ and $\tilde{p}_0 = \tilde{r}_0$
2.  For k=0,2, ... Do
3.      $\alpha_k = \tilde{r}_k^T\tilde{r}_k/\tilde{p}_k^T\tilde{A}\tilde{p}_k$        $= r_k^T CC^T r_k/p_k^T Ap_k = r_k^T z_k/p_k^T Ap$
4.      $\tilde{x}_{k+1} = \tilde{x}_k + \alpha_k\tilde{p}_k$        $\overset{C}{\Rightarrow} x_{k+1} = x_k + \alpha_k p_k$
5.      $\tilde{r}_{k+1} = \tilde{r}_k - \alpha_k\tilde{A}\tilde{p}_k$        $\overset{C^T}{\Rightarrow} r_{k+1} = r_k - \alpha_k Ap_k$
6.      $\beta_k = \tilde{r}_{k+1}^T\tilde{r}_{k+1}/\tilde{r}_k^T\tilde{r}_k$        $= r_{k+1}^T CC^T r_{k+1}/r_k^T CC^T r_k = r_{k+1}^T z$
7.      $\tilde{p}_{k+1} = \tilde{r}_{k+1} + \beta_k\tilde{p}_k$        $\overset{C}{\Rightarrow} p_{k+1} = r_{k+1} + \beta_k p_k$
8.      if $\tilde{x}_k$ accurate enough then stop
9.  EndDo
```

Writing the algorithm only using the unpreconditioned variables leads to:

Preconditioned Conjugate Gradient algorithm

```
Compute $r_0 = b - Ax_0$, $z_0 = Mr_0$ and $p_0 = r_0$
For k=0,2, ... Do
  $\alpha_k = r_k^T z_k / p_k^T A p_k$
  $x_{k+1} = x_k + \alpha_k p_k$
  $r_{k+1} = r_k - \alpha_k A p_k$
  $z_{k+1} = M r_{k+1}$
  $\beta_k = r_{k+1}^T z_{k+1} / r_k^T z_k$
  $p_{k+1} = r_{k+1} + \beta_k p_k$
  if $x_k$ accurate enough then stop
EndDo
```

Preconditioner taxonomy

There are two main classes of preconditioners

- implicit preconditioners:
  approximate A with a matrix M such that solving the linear system $Mz = r$ is easy.
- explicit preconditioners:
  approximate A^{-1} with a matrix M and just perform $z = Mr$.
- Updating a preconditioner when multiple right-hand sides are given in sequence

The governing ideas in the design of the preconditioners are very similar to those followed to define iterative stationary schemes. Consequently, all the stationnary methods can be used to defined preconditioners.

A sequence of linear least-squares problems

- Originally developped for SPD linear systems with multiple right-hand sides (RHS).
- Solve systems $Ax = b_1$, $Ax = b_2$, ..., $Ax = b_r$ with RHS in sequence, by iterative methods: Conjugate Gradient (CG) or variants.
- Precondition the CG using information obtained when solving the previous system.
- Extension of the idea to nonlinear process such as Gauss-Newton method. The matrix of the normal equations varies along the iterations.

Why to precondition ?

- Transform $Ax = b$ in an equivalent system having a more favorable eigenvalues distribution.
- Use a preconditioning matrix H (which must be cheap to apply).
- Ideas to design preconditioner H:
  - H approximates A^{-1}.
  - $cond(HA) < cond(A)$.
  - HA has eigenvalues more clustered than those of A.
- Note: when a preconditioning is used, residuals are:
  - Orthogonal if H is factored in LL^T.
  - Conjugate w.r.t. H if H is not factored.

Preconditioning techniques considered (I)

- We consider techniques to precondition or improve an existing preconditioner (second level preconditioning) :
  - Solve $Ax = b_1$ and extract information $info_1$.
  - Use $info_1$ to solve $Ax = b_2$ and extract information $info_2$.
  - Use $info_2$ (and possibly $info_1$) to solve $Ax = b_3$ and ...
  - ...
- $Info_k$ will be:
  - residuals;
  - descent directions;
  - steps;
  - or other vectors such as eigenvectors of A ...

Preconditioning techniques considered (II)

We study and compare two approaches:

- Deflation [Frank, Vuik, 2001].
- Limited Memory Preconditioners (LMP): Preconditioners based on a set of A-conjugate directions.
    - Generalization of known preconditioners: spectral [Fisher, 1998], L-BFGS [Nocedal, Morales, 2000], warm start [Gilbert, Lemaréchal, 1989].

We cover:

- Theoretical properties.
- Numerical experiments (data assimilation).

Deflation Techniques

- Given $W \in \mathbb{R}^{n \times k}$ $(k \ll n)$ formed with appropriate information obtained when solving the previous system.
- Consider the oblique projector $P = I - AW(W^T AW)^{-1} W^T$.
- Split the solution vector as follows $x^* = \underbrace{(I - P^T)x^*}_{direct} + \underbrace{P^T x^*}_{iterative}$.
- Compute $(I - P^T)x^*$ with a direct method.
- Compute $P^T x^*$ with an iterative method.

Some Properties (Deflation)

- Computation of $(I - P^T)x^*$:
  - $(I - P^T)x^* = W(W^TAW)^{-1}W^TAx^* = W(W^TAW)^{-1}W^Tb$.
- Computation of P^Tx^*:
  - Any solution of the compatible singular system $PAy = Pb$ satisfies $P^Tx^* = P^Ty$.
  - Note: $PA = (PA)^T$ and $cond(PA) \ \leq \ cond(A)$.
  - Use CG with $y_0 = 0$ to solve $PAy = Pb$ and compute P^Ty.

Limited Memory Preconditioners (LMP)

- $$\begin{aligned} H_{k+1} &= [I - W(W^TAW)^{-1}W^TA][[I - W(W^TAW)^{-1}W^TA]^T \\ &+ \sigma W(W^TAW)^{-1}W^T \end{aligned}$$

  with $W = [w_1, \ldots, w_k]$.
- Particular forms
  - The w_i's are the descent directions obtained from CG:
    $w_i = p_i$
    $\Rightarrow$ L-BFGS preconditioner.
  - The w_i's are eigenvectors of A: $w_i = v_i$
    $\Rightarrow$ spectral preconditioner.
  - The w_i's are Ritz-vectors of A wrt an orthognal basis V :
    $w_i = Vz_i$ and for some θ_i, $V^T(Aw_i - \theta_i w_i) = 0$
    $\Rightarrow$ Ritz preconditioner.

Spectral Properties for LMP (I)

- Theorem : the spectrum $\mu_1, \ldots, \mu_n$ of the preconditioned matrix $H_{k+1}A$ satisfies:

$$\begin{cases} \mu_j = 1, & \text{for } j = 1, \ldots, k \\ \lambda_{j-k}(A) \leq \mu_j \leq \lambda_j(A), & \text{for } j = k+1, \ldots, n, \end{cases}$$

  where $\lambda_j(A)$ is the $j-$th eigenvalue of A (increasing order assumed).

- Note: the matrix A to precondition is the same (only the RHS changes).

Proof I

Assume that the w_i have been normalized, without loss of generality:

$$W = [w_0, w_1, \dots, w_k] \tag{17}$$

with $w_i = \frac{w_i}{(w_i^T A w_i)^{\frac{1}{2}}}$. Let $\underline{W}$, be the A-conjugate complement of W in $\mathbb{R}^n$. The following conjugacy relations hold: $W^T A W = I_{k+1},\ \underline{W}^T A W = 0,\ \underline{W}^T A \underline{W} = I_{n-k-1}$. We have

$$\begin{aligned} H_{k+1} &= [I - \sum_{i=0}^{k} \frac{A w_i w_i^T}{w_i^T A w_i}]^T [I - \sum_{i=0}^{k} \frac{A w_i w_i^T}{w_i^T A w_i}] + \sum_{i=0}^{k} \frac{w_i w_i^T}{w_i^T A w_i} && (18) \\ &= [I - A \sum_{i=0}^{k} w_i w_i^T]^T [I - A \sum_{i=0}^{k} w_i w_i^T] + \sum_{i=0}^{k} w_i w_i^T . && (19) \end{aligned}$$

Proof II

This can be rewritten in terms of matrices

$$
\begin{aligned}
H_{k+1} &= [I - AWW^T]^T [I - AWW^T] + WW^T && (20)\\
&= [I - WW^T A][I - AWW^T] + WW^T. && (21)
\end{aligned}
$$

It can be shown as follows that $I - WW^T A = \underline{W}\,\underline{W}^T A$. Using the conjugacy relations, we have $[W, \underline{W}]^T A [W, \underline{W}] = I$ and since $\mathrm{span}[W, \underline{W}] = \mathbb{R}^n$, the matrix $[W, \underline{W}]$ is nonsingular. Multiplying $[W, \underline{W}]^T A [W, \underline{W}] = I$ left by $[W, \underline{W}]$ and right by $[W, \underline{W}]^{-1}$ yields $[W, \underline{W}][W, \underline{W}]^T A = I$ which is equivalent to $WW^T A + \underline{W}\,\underline{W}^T A = I$. Therefore, (21) writes

$$
\begin{aligned}
H_{k+1} &= [\underline{W}\,\underline{W}^T A][A \underline{W}\,\underline{W}^T] + WW^T && (22)\\
&= [W, \underline{W}] \begin{pmatrix} I_{k+1} & 0 \\ 0 & \underline{W}^T A A \underline{W} \end{pmatrix} [W, \underline{W}]^T && (23)\\
&= [W, \underline{W}] \begin{pmatrix} I_{k+1} & 0 \\ 0 & Q^T A Q \end{pmatrix} \begin{pmatrix} W^T \\ \underline{W}^T \end{pmatrix} && (24)
\end{aligned}
$$

Proof III

The matrix $Q = A^{\frac{1}{2}}\underline{W}$ is an orthogonal matrix of $\mathbb{R}^{n\times(n-k-1)}$. Since $V\Lambda V^T = Q^T A Q$ with $V^T V = I$, is a spectral decomposition of the matrix $Q^T A Q$, we have

$$H_{k+1} = [W, \underline{W}V] \begin{pmatrix} I_{k+1} & 0 \\ 0 & \Lambda \end{pmatrix} \begin{pmatrix} W^T \\ V^T \underline{W}^T \end{pmatrix}. \tag{25}$$

The matrix $A^{\frac{1}{2}} H_{k+1} A^{\frac{1}{2}}$ has the same spectrum as the matrix $H_{k+1}A$ and can be written

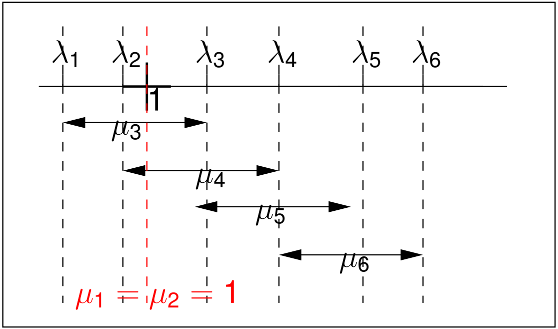
$$\begin{aligned} A^{\frac{1}{2}} H_{k+1} A^{\frac{1}{2}} &= A^{\frac{1}{2}}[W, \underline{W}V] \begin{pmatrix} I_{k+1} & 0 \\ 0 & \Lambda \end{pmatrix} \begin{pmatrix} W^T \\ V^T \underline{W}^T \end{pmatrix} A^{\frac{1}{2}} && (26) \\ &= [A^{\frac{1}{2}}W, A^{\frac{1}{2}}\underline{W}V] \begin{pmatrix} I_{k+1} & 0 \\ 0 & \Lambda \end{pmatrix} \begin{pmatrix} W^T A^{\frac{1}{2}} \\ V^T \underline{W}^T A^{\frac{1}{2}} \end{pmatrix} && (27) \end{aligned}$$

This is a diagonalization of the matrix $A^{\frac{1}{2}} H_{k+1} A^{\frac{1}{2}}$, because $[A^{\frac{1}{2}}W, A^{\frac{1}{2}}\underline{W}V]$ is orthogonal. The result follows from (see Horn

Proof IV

anf Johnson) $\lambda_{j-k}(A) \leq \lambda_j(\Lambda) = \lambda_j(Q^T A Q) \leq \lambda_j(A)$ with $j = k+2, \ldots, n$.

Spectral properties for LMP (II)



- Eigenvalues translated to 1.
- The rest of the spectrum is not expanded compared to the spectrum of A.

Existence of a factored form for the LMP (not the Cholesky factor !)

- L-BFGS:
  - A possible factored form is $H_{k+1} = L_{k+1} L_{k+1}^T$ where:

$$L_{k+1} = \prod_{i=0}^{k} \left(I - \frac{s_i {y_i}^T}{{y_i}^T s_i} + \frac{s_i}{\sqrt{{y_i}^T s_i}} \frac{{r_i}^T}{\|r_i\|} \right),$$

    with $s_i = x_{i+1} - x_i$ and $y_i = r_{i+1} - r_i$.
  - Same cost in memory and CPU as the unfactored form.
- Spectral:
  - A possible factored form is $H_{k+1} = L_{k+1}^2$ where:

$$L_{k+1} = I + \sum_{i=1}^{k+1} \left(\frac{1}{\sqrt{\lambda_i}} - 1 \right) \frac{v_i {v_i}^T}{{v_i}^T v_i}.$$

  - Same cost in memory as the unfactored form.

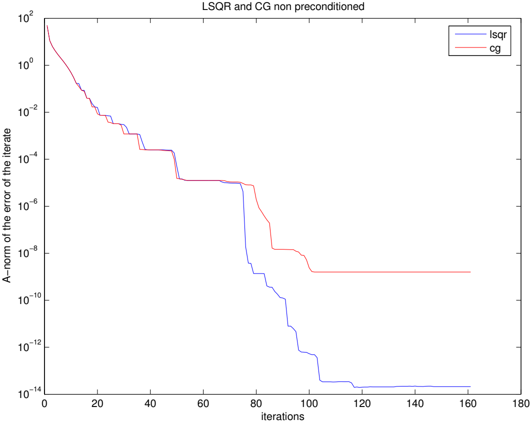
Why looking for a factored form $H = LL^T$?

- With a non factored form, we use CG preconditioned by H.
- With a factored form, we solve $L^T A L u = L^T b$; $x = Lu$.
  Advantages:
  - When accumulating preconditioners, symmetry and positiveness are still maintained:

    $$L_1^T A L_1 y_1 = L_1^T b_1, \quad L_2^T (L_1^T A L_1) L_2 y_2 = L_2^T L_1^T b_2, \quad \ldots$$

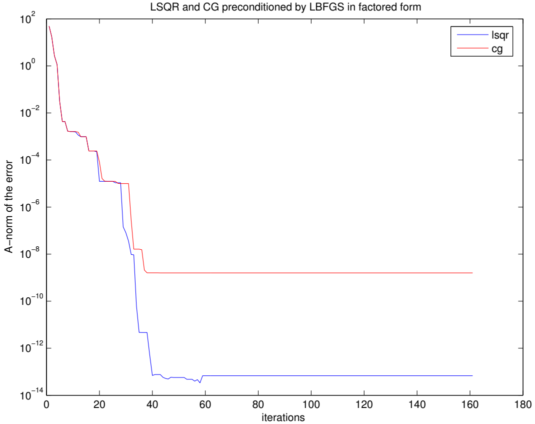
  - Least-squares $min_x \|Ax - b\|$ or $AA^T x = A^T b$:
    LSQR (or CGLS) is more accurate than CG in presence of rounding errors but works with (A, A^T, L, L^T, b) instead of $(A^T A, A^T b, H)$.
  - More appropriate if reorthogonalization of the residuals is used.

Experiments with unpreconditioned LSQR



LSQR is better than CG !

Experiments with preconditioned LSQR

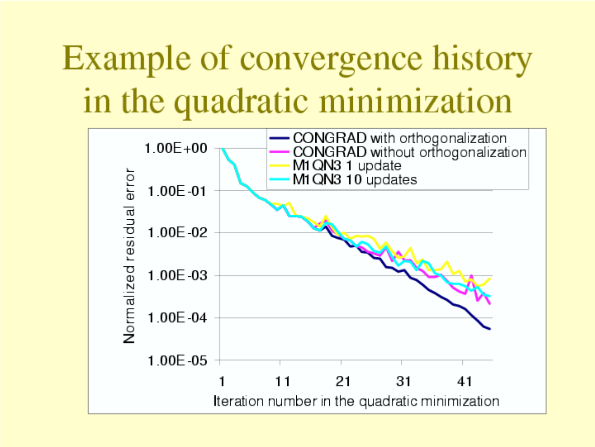


LSQR is again better than CG !

Why to reorthogonalize the residuals ?

- In finite precision, residuals often loose orthogonality (or conjugacy) and theoretical convergence is then slowed down.
- Reorthogonalization of residuals in CG is terribly successful when matrix-vector product is very expensive compared to other computations in CG (see example in the next slide).
- Note: to restore orthogonality or conjugacy, working with $L^T A L$ and the canonical inner-product is better (memory, CPU, error propagation) than working on A preconditioned by H.

Example of reorthogonalization effect : CERFACS data assimilation system (1 000 000 unknowns)



Example of convergence history in the quadratic minimization

Experiments with a data assimilation problem

Problem formulation: nonlinear least-squares problem

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

- Size of real (operational) problems : $x \in \mathbb{R}^{10^6}$, $y_j \in \mathbb{R}^{10^5}$.
- The observations y_j are noisy.
- Solution strategy : Incremental 4DVAR (i.e. inexact/truncated Gauss-Newton algorithm).

Main ingredients

- Sequence of linear symmetric positive definite systems to solve:

$$A_i^T A_i x = A_i^T b_i$$

- Whose matrix varies.

Experiments description

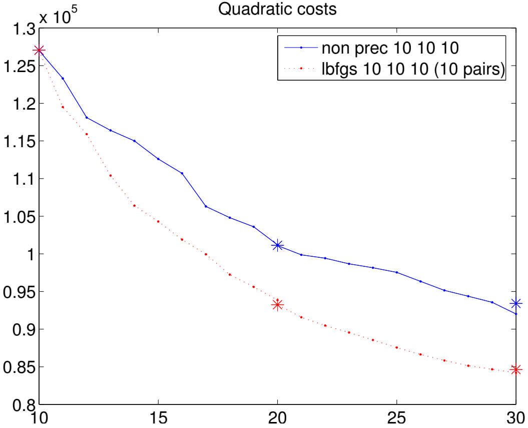
Algorithmic variants tested:

- Use CG to solve the normal equations.
- Compare with Deflation (spectral) 3 LMP preconditioning techniques:
  - Ritz technique (using Ritz information).
  - Spectral preconditioner (using spectral info.).
  - L-BFGS preconditioner (using descent directions).
- Where spectral information is needed, use Ritz (vectors) as approximations of the eigenvectors.
- Ritz vectors are obtained by mean of a variant of CG: the Lanczos algorithm which combines linear and eigen solvers.

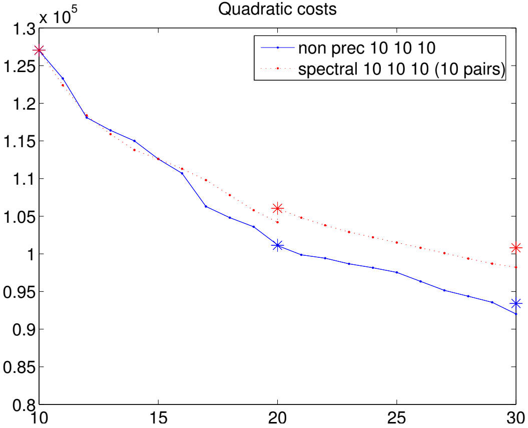
Experiment large system OPA_VAR

- OPA ocean general circulation model.
- Comparison of the effect on the second quadratic minimization (the first is used to obtain the information).
- Comparison of the quadratic/Nonlinear cost.
- Information : the 10 vectors obtained using the first outer-loop (10 cg steps).

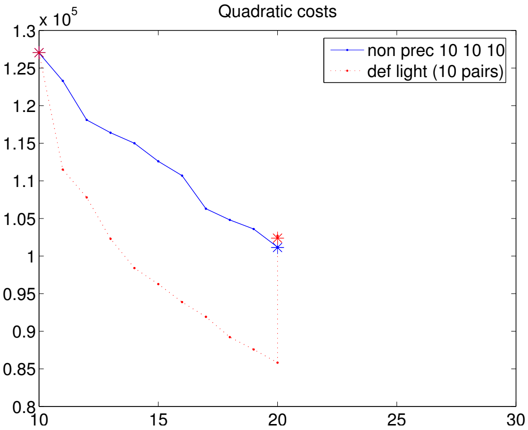
L-BFGS better than no preconditioning



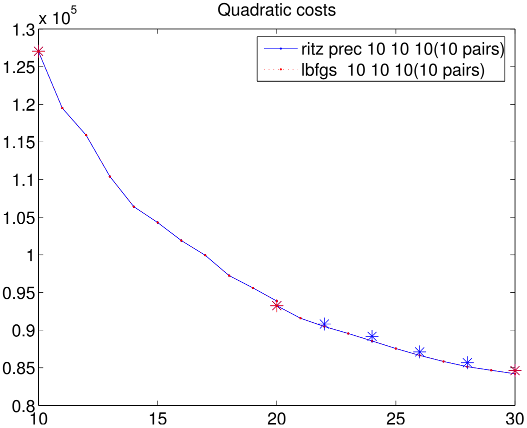
Spectral is worse than no preconditioning



Something strange about deflation



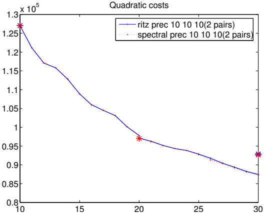
Ritz = L-BFGS for 10 pairs

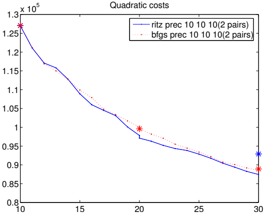


Preconditioner comparison

- Comparison with the same number of pair (unfair, memory-wise spectral=bfgs cheaper than deflation=L-BFGS)
- Number of pair considered : 6 and 2

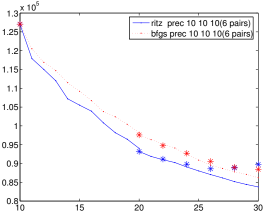
Preconditioner comparison

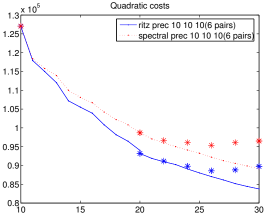




2 pairs: Ritz/Spectral a bit better the other preconditioners!

Preconditioner comparison





6 pairs: Ritz outperforms the other preconditioners!

Remarks on our system !

- Spectral preconditioner does not work in our case (k vectors).
- L-BFGS/Ritz preconditioner: ($2k/k$)
  - Based on by-products of CG.
  - More efficient than the spectral preconditioner or than no preconditioner.
  - Ritz is a stabilized version of spectral
- Deflation: the computationally ”light” version is not efficient ($2k$).
- There is a ”full version” for all the preconditoners : work to prepare the preconditioner not neglectable, but robust wrt large matrix changes. The full form are being compared. They all have the same cost. Usefull when starting a system with poor initial guess.— versions of the Ritz/Spectral/L-BFGS that do have the same property (and unfortunately cost). Usefull when starting a system with poor initial guess.

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

Preconditioned CG algorithm

```
Initialization

  ▶ $r_0 = A\delta x_0 - b,\ z_0 = Fr_0,\ p_0 = z_0$

For $i = 0, 1, \ldots$

   1. $q_i = (B^{-1} + H^T R^{-1} H) p_i$
   2. $\alpha_i =< r_i, z_i > / < q_i, p_i >$          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. $\beta_i =< r_{i+1}, z_{i+1} > / < r_i, z_i >$   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. $BH^{\mathrm{T}}G = FH^{\mathrm{T}}$.
2. $v_0 = x^b - x_0$.

$\rightarrow$ *vectors* $\widehat{r}_i$, $\widehat{p}_i$, $\widehat{v}_i$, $\widehat{z}_i$ *and* $\widehat{q}_i$ *such that*

$$\begin{aligned} r_i &= H^{\mathrm{T}}\widehat{r}_i, \\ p_i &= BH^{\mathrm{T}}\widehat{p}_i, \\ v_i &= v_0 + BH^{\mathrm{T}}\widehat{v}_i, \\ z_i &= BH^{\mathrm{T}}\widehat{z}_i, \\ q_i &= H^{\mathrm{T}}\widehat{q}_i \end{aligned}$$

Initialization steps

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

Loop: WHILE

1. $H^{\mathrm{T}} \widehat{q}_{i-1} \; = \; H^{\mathrm{T}} (R^{-1} H B^{-1} H^{\mathrm{T}} + I_m) \widehat{p}_{i-1}$
2. $\alpha_{i-1} \; = \; r_{i-1}^{\mathrm{T}} z_{i-1} \, / \, \widehat{q}_{i-1}^{\mathrm{T}} \widehat{p}_{i-1}$
3. $B H^{\mathrm{T}} \widehat{v}_i \; = \; B H^{\mathrm{T}} (v_{i-1} + \alpha_{i-1} \widehat{p}_{i-1})$
4. $H^{\mathrm{T}} \widehat{r}_i \; = \; H^{\mathrm{T}} (r_{i-1} + \alpha_{i-1} \widehat{q}_{i-1})$
5. $B H^{\mathrm{T}} \widehat{z}_i \; = \; F H^{\mathrm{T}} \widehat{r}_i \; = \; B H^{\mathrm{T}} G \widehat{r}_i$ $\quad F H^T = B H^T G$
6. $\beta_i \; = \; (r_i^{\mathrm{T}} z_i \, / \, r_{i-1}^{\mathrm{T}} z_{i-1})$
7. $B H^{\mathrm{T}} \widehat{p}_i \; = \; B H^{\mathrm{T}} (-\widehat{z}_i + \beta_i \widehat{p}_{i-1})$

Initialization

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

Loop on k

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

- $\widehat{A} = R^{-1}HBH^T + I_m$
- 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

Restricted PCG (version 2)

Initialization steps

Loop: WHILE

```
1. $\widehat{q}_{i-1} \;=\; R^{-1} t_{i-1} \,+\, \widehat{p}_{i-1}$
2. $\alpha_{i-1} \;=\; w_{i-1}^{\mathrm{T}} \widehat{r}_{i-1} \,/\, \widehat{q}_{i-1}^{\mathrm{T}} t_{i-1}$
3. $\widehat{v}_i \;=\; \widehat{v}_{i-1} \,+\, \alpha_{i-1} \widehat{p}_{i-1}$
4. $\widehat{r}_i \;=\; \widehat{r}_{i-1} \,+\, \alpha_{i-1} \widehat{q}_{i-1}$
5. $\widehat{z}_i \;=\; G \widehat{r}_i$
6. $w_i \;=\; HBH^{\mathrm{T}} \widehat{z}_i$
7. $\beta_i \;=\; w_i^{\mathrm{T}} \widehat{r}_i \,/\, w_{i-1}^{\mathrm{T}} \widehat{r}_{i-1}$
8. $\widehat{p}_i \;=\; -\widehat{z}_i \,+\, \beta_i \widehat{p}_{i-1}$
9. $t_i \;=\; -w_i \,+\, \beta_i t_{i-1}$
```

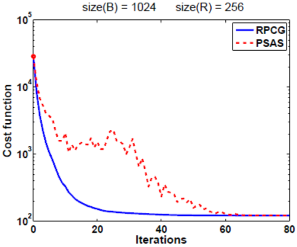
Dual approach

Minimization in dual space

1. **Iteratively** solve

$$(I_m + R^{-1} H_k B H_k{}^T)\lambda = d$$

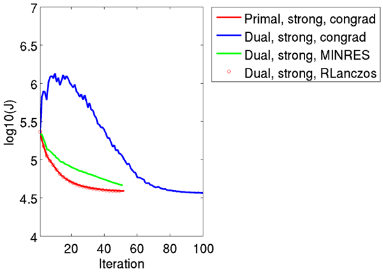
2. **Set** $\delta x_k = x_b - x_k + B H_k{}^T \lambda$

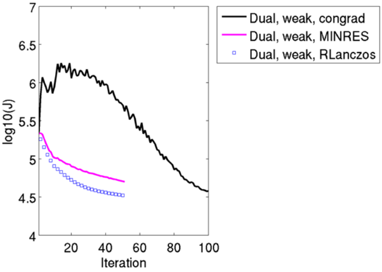


- **PSAS** (Courtier 1997): Preconditioned CG (PCG) with $\tilde{R}$ inner product.
- **RPCG** (Gratton and Tshimanga 2009): PCG with $H_k B H_k^T$
  → It generates the **same iterates** as those generated by the **primal approach.**

→ Cost function: $J(\delta x_k) = \frac{1}{2}\|\delta x_k - x_b + x_k\|^2_{B^{-1}} + \frac{1}{2}\|H_k \delta x_k - d_k\|^2_{R^{-1}}$

- 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





Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers

- It is **possible to maintain** the one-to-one correspondance between primal and dual iterates, under the assumption that

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

  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 $H_k B H_k^T$ inner product.

- Solution algorithm: **Preconditioned Conjugate Gradient method (PCG)**

  $\rightarrow$ 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

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

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

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

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

$$\text{subject to } \; \Delta F_k = \Delta F_k^T, \quad F_{k+1} q_k = p_k$$

where W is any symmetric positive definite matrix satisfying $W p_k = q_k$

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

$$\begin{aligned} p_i &= BH^{\mathrm{T}}\widehat{p}_i, \\ q_i &= H^{\mathrm{T}}\widehat{q}_i \end{aligned}$$

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

- $M = HBH^T$
- $\widehat{p}_k$ is the search direction
- $\widehat{q}_k = (I_m + R^{-1}HBH^T)\widehat{p}_k$
- $\widehat{\tau}_k = 1/(\widehat{q}_k^T HBH^T \widehat{p}_k)$

The dual pairs

ΔG_k defined by $\Delta G_k = G_{k+1} - G_k$ is the optimum solution to the minimization problem defined as:

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

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

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

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

$$\|\mathbf{x_{k+1}} - \mathbf{x}^*\|_\mathbf{A} \leq \mathbf{2}(\frac{\sqrt{\mu_\mathbf{n}} - \sqrt{\mu_\mathbf{1}}}{\sqrt{\mu_\mathbf{n}} + \sqrt{\mu_\mathbf{1}}})^\mathbf{k} \|\mathbf{x}^*\|_\mathbf{A}$$

- When RPCG is used, the iterates are belongs to the affine subspace of $x_0 + Im(BH^T)$.
- If GA has eigenvalues $\nu_1 \leq \nu_2 \leq ... \leq \nu_m$, Restricted PCG (version 3) with zero initial starting vector satisfies the inequality:

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

where $A = B^{-1} + H^T R^{-1} H$ and $\widehat{A} = I + R^{-1} HBH^T$

- $FABH^T = BH^T G\widehat{A}$. Therefore; BH^T is an invariant subspace of FA
- Every eigenvalue of $G\widehat{A}$ is an eigenvalue of FA. So, $\mu_1 \leq \nu_1$ and $\mu_n \geqslant \nu_n$.

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

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

Happy breakdown of $m < n$

1. The modified (G-S) orthogonalization scheme writes

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

2. We suggest the following re-orthogonalization scheme

$$\hat{r}_i \leftarrow \prod_{j=1}^{i-1} \left(I_m - \frac{\hat{r}_j w_j^{\mathrm{T}}}{\hat{r}_j^{\mathrm{T}} w_j} \right) \hat{r}_i.$$

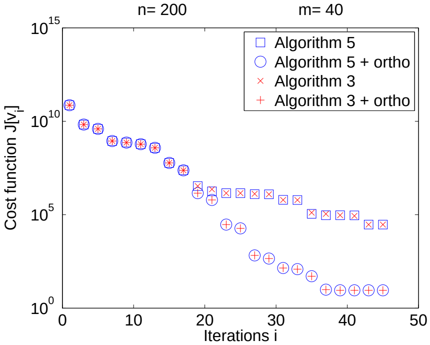


Figure: Orthogonalization issues.

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.

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

$\rightarrow F = F_l$.

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

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

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

$\rightarrow$ This preconditioner satisfies the relation: $FH^T = BH^T G$ and it is symmetric in the C inner product (Gratton, Gurol and Toint 2012).

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers
- Weak constrained formulation (with Selime Gurol)

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers
- Weak constrained formulation (with Selime Gurol)

Outline

- Introduction
  - Basic concepts
- Data Assimilation from the parameter estimation point of view
  - Preliminary definitions and notations
  - Estimation under the Gaussian assumption
  - Estimation under the linearity assumption.
  - Filtering the system in full generality
- Particle filter in a nutshell
- Data Assimilation form the optimization point of view
  - The problem
  - Steps in duality
  - Solution algorithms
  - Sequential estimation and Kalman filter
- Data assimilation from the linear algebra point of view
- Specialized solvers
  - Dual space solvers
  - Preconditioning in dual space
  - Ensemble Kalman filter
  - Domain decomposition solvers
- Weak constrained formulation (with Selime Gurol)

Why saddle-point formulation?

- 4D-Var is a sequential algorithm.
- Parallelization of 4D-Var in the spatial domain has been performed by a spatial decomposition, and distribution over processors of the model grid.

  → The number of grid points (associated with each processor) are independent of the resolution of the model.
- BUT, increasing the resolution of the model, increases the work per processor since higher resolutions require shorter timesteps.
- In order to keep the work per processor constant, parallelization in the time dimension is required.

Saddle-point formulation allows parallelization in the time dimension.

Outline

- Problem formulation and the standard solution
- Saddle point approach
- Preconditioning saddle point formulation
- Numerical results
- Conclusions

Outline

- Problem formulation and the standard solution
- Saddle point approach
- Preconditioning saddle point formulation
- Numerical results
- Conclusions

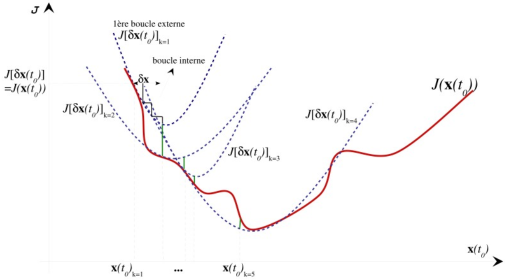
Weak-constraint 4D-Var

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

- $\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_N \end{pmatrix} \in \mathbb{R}^n$ is the control variable (with $x_j = x(t_j)$)
- 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.

How to solve the nonlinear problem?

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



ref: A. Vidard, NEMOVAR presentation, 2008.

Outer loop:

1. Linearize the nonlinear cost function at the current iterate
2. Solve the linearized subproblem (quadratic problem) for the increment (step) $\delta\mathbf{x}^{(k)}$
3. Update the iterate $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}$

The linearized subproblem (Inner loop)

- The linearized problem at the k-th outer loop is given by

$$\boxed{\min_{\boldsymbol{\delta x}} \ \frac{1}{2}\|\delta x_0 - b^{(k)}\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{j=0}^{N}\left\|H_j^{(k)}\delta x_j - d_j^{(k)}\right\|^2_{\mathbf{R}_j^{-1}} + \frac{1}{2}\sum_{j=1}^{N}\|\underbrace{\delta x_j - M_j^{(k)}\delta x_{j-1}}_{\delta q_j}}$$

- $\boldsymbol{\delta x} = \begin{pmatrix} \delta x_0 \\ \delta x_1 \\ \vdots \\ \delta x_N \end{pmatrix} \in \mathbb{R}^n$ is the increment.

- The vectors $b^{(k)}$, $c_j^{(k)}$ and $d_j^{(k)}$ are defined by

$$\begin{aligned} b^{(k)} &= x_b - {x_0}^{(k)} \\ c_j^{(k)} &= \bar{q} - q_j^{(k)} \\ d_j^{(k)} &= y_j - \mathcal{H}_j({x_j}^{(k)}) \end{aligned}$$

  and they are calculated at the outer loop.

Rewriting the linearized subproblem

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

where

- $\mathbf{L} = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix}$
- $\mathbf{d} = \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_N \end{pmatrix}$ and $\mathbf{b} = \begin{pmatrix} b \\ c_1 \\ \vdots \\ c_N \end{pmatrix}$
- $\mathbf{H} = diag(\mathbf{H_0}, \mathbf{H_1}, \ldots, \mathbf{H_N})$
- $\mathbf{D} = diag(\mathbf{B}, \mathbf{Q_1}, \ldots, \mathbf{Q_N})$ and $\mathbf{R} = diag(\mathbf{R_0}, \mathbf{R_1}, \ldots, \mathbf{R_N})$

Rewriting the linearized subproblem

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

- $\mathbf{L}\boldsymbol{\delta}\mathbf{x} = \begin{pmatrix} I & & & & \\ -M_1 & I & & & \\ & -M_2 & I & & \\ & & \ddots & \ddots & \\ & & & -M_N & I \end{pmatrix} \begin{pmatrix} \delta x_0 \\ \delta x_1 \\ \delta x_2 \\ \vdots \\ \delta x_N \end{pmatrix} = \begin{pmatrix} \delta x_0 \\ \delta x_1 - M_1 \delta x_0 \\ \delta x_2 - M_2 \delta x_1 \\ \vdots \\ \delta x_N - M_N \delta x_{N-1} \end{pmatrix}$
- Matrix-vector products with **L** can be parallelized in the time dimension

Rewriting the linearized subproblem

- Making change of variables

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

the subproblem can also be rewritten as

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

- $\delta\mathbf{x} = \mathbf{L}^{-\mathbf{1}}\delta\mathbf{p}$ is sequential $\rightarrow \delta x_j = M_j \delta x_{j-1} + \delta q_j$

The linearized subproblems

State Formulation

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

- Matrix-vector products with $\mathbf{L}$ can be parallelized in the time dimension.
- Solution algorithm: Preconditioned Lanczos or PCG type methods.
- 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 $\widetilde{\mathbf{L}}^{-\mathbf{1}}$.

Forcing Formulation

$$\min_{\delta\mathbf{p}} \ \frac{1}{2}\|\delta\mathbf{p} - \mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\mathbf{H}\mathbf{L}^{-\mathbf{1}}\delta\mathbf{p} - \mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

- Matrix-vector products with $\mathbf{L}^{-\mathbf{1}}$ is sequential.
- Solution algorithm: Preconditioned Lanczos or PCG type methods.
- Preconditioning is straightforward. The structure is similar to the strong-constraint case.

Outline

- Problem formulation and the standard solution
- Saddle point approach
- Preconditioning saddle point formulation
- Numerical results
- Conclusions

Saddle Point Approach

- Let us consider weak-constraint 4D-Var as a constrained problem:

$$\min_{(\boldsymbol{\delta}\mathbf{p},\boldsymbol{\delta}\mathbf{w})} \ \frac{1}{2}\|\boldsymbol{\delta}\mathbf{p}-\mathbf{b}\|^2_{\mathbf{D}^{-1}} + \frac{1}{2}\|\boldsymbol{\delta}\mathbf{w}-\mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

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

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

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

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

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

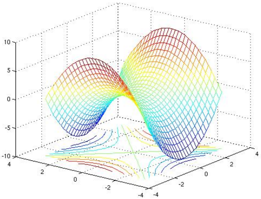
Saddle Point Approach

- In matrix form:

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

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

- The solution of this problem is a saddle point



→ This approach is time-parallel.

→ Solution algorithm: GMRES method with a preconditioner.

Outline

- Problem formulation and the standard solution
- Saddle point approach
- Preconditioning saddle point formulation
- Numerical results
- Conclusions

Preconditioning

- A preconditioner attempts to improve the spectral properties of the system matrix $\mathcal{A}$.
- When using GMRES, a clustered spectrum often results in rapid convergence, especially the departure from normality of the preconditioned matrix is not too high (Benzi et. al 2005).

$\rightarrow$ When solving an indefinite saddle point system with GMRES, it is crucial to find an efficient preconditioner.

Efficient preconditioner $\mathcal{P}$

- is an approximation to $\mathcal{A}$
- the cost of constructing and applying the preconditioner should be less than the gain in computational cost
- exploits the block structure of the problem for saddle point systems

Implementation

- Solving a system $\mathcal{A}\,\mathbf{u} = \mathbf{f}$ with a preconditioner $\mathcal{P}$ requires solving

$$(\mathcal{P}^{-1}\mathcal{A})\mathbf{u} = \mathcal{P}^{-\mathbf{1}}\mathbf{f}$$

Preconditioning Saddle Point Systems

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

- Block preconditioners (Kuznetsov (1995), Murphy, Golub and Wathen (2000), Bramble and Pasciak (1988))

  $$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & -\mathbf{S} \end{pmatrix}, \quad \mathcal{P} = \begin{pmatrix} \mathbf{A} & \mathbf{B}^{T} \\ \mathbf{0} & \mathbf{S} \end{pmatrix}$$

  where $\mathbf{S} = \mathbf{B}\mathbf{A}^{-\mathbf{1}}\mathbf{B}^{\mathbf{T}}$ is the Schur complement (the unpreconditioned 4D-Var Hessian).
- Constraint preconditioners (Bergamaschi et. al (2004), Gould and Wathen (2000), Benzi et al. 2005)

  $$\mathcal{P} = \begin{pmatrix} \widetilde{\mathbf{A}} & \mathbf{B}^{T} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}$$

  It is assumed that solving the system involving $\mathcal{P}$ is significantly easier than solving the original system.
- Hermitian and skew Hermitian splitting of $\mathcal{A}$, stationary iterative methods, multilevel methods, ... (Benzi et al (2005))

Preconditioning Saddle Point Formulation of 4D-Var

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

- **B** is the most computationally expensive block and calculations involving **A** are relatively cheap.
- The inexact constraint preconditioner proposed by (Bergamaschi et. al. 2005) is promising for our application. The preconditioner can be chosen as:

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

  where

  - $\widetilde{\mathbf{L}}$ is an approximation to the matrix **L**
  - $\widetilde{\mathbf{B}} = [\widetilde{\mathbf{L}}^{\mathrm{T}} \quad \mathbf{0}]$ is a full row rank approximation of the matrix $\mathbf{B} \in \mathbb{R}^{n\times(m+n)}$
- The numerical experiments with OOPS show that this preconditioner is effective and it is cheap to obtain and easy to apply.

Preconditioning Saddle Point Formulation of 4D-Var

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

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:

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix}$$

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

$$\mathcal{P} = \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix}$$

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

$$\underbrace{\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix}}_{\mathcal{A}_k} \underbrace{\begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{pmatrix}}_{\mathbf{u}_j^{(k)}} = \underbrace{\begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}}_{\mathbf{f}_j^{(k)}}$$

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

Preconditioning Saddle Point Formulation of 4D-Var

$$\begin{pmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \mathbf{A} & \widetilde{\mathbf{B}}^{\mathrm{T}} \\ \widetilde{\mathbf{B}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \mathbf{A}\mathbf{u}_1 + \widetilde{\mathbf{B}}^{\mathrm{T}}\mathbf{u}_2 \\ \widetilde{\mathbf{B}}\mathbf{u}_1 \end{pmatrix} + \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\mathbf{u}_2 \\ \Delta\mathbf{B}\mathbf{u}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} \Delta\mathbf{B}^{\mathrm{T}}\mathbf{u}_2 \\ \Delta\mathbf{B}\mathbf{u}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{b} - \mathbf{A}\mathbf{u}_{\mathbf{1}} - \widetilde{\mathbf{B}}^{\mathrm{T}}\mathbf{u}_{\mathbf{2}} \\ \mathbf{c} - \widetilde{\mathbf{B}}\mathbf{u}_{\mathbf{1}} \end{pmatrix}$$

- Let's define the vectors $\mathbf{r}_b$ and $\mathbf{r}_c$ as

$$\mathbf{r}_b = \mathbf{b} - \mathbf{A}\mathbf{u}_1 - \widetilde{\mathbf{B}}^{\mathrm{T}}\mathbf{u}_2$$

$$\mathbf{r}_c = \mathbf{c} - \widetilde{\mathbf{B}}\mathbf{u}_1$$

- Then we have

$$\Delta\mathbf{B}^{\mathrm{T}}\mathbf{u}_2 = \mathbf{r}_b \tag{28}$$

$$\Delta\mathbf{B}\mathbf{u}_1 = \mathbf{r}_c \tag{29}$$

→ We want to find an update $\Delta\mathbf{B}$ satisfying these equations.

Preconditioning Saddle Point Formulation of 4D-Var

- A rank-1 update to $\Delta\mathbf{B}$ can be given by

$$\boxed{\Delta\mathbf{B} = \alpha\mathbf{v}\mathbf{w}^{\mathrm{T}}}$$

  where $\mathbf{v} \in \mathbb{R}^{n}$ and $\mathbf{w} \in \mathbb{R}^{n+m}$.

- Substituting this relation into equations (1) and (2), we get

$$\begin{aligned} \Delta\mathbf{B}^{\mathrm{T}}\mathbf{u}_2 &= \mathbf{r}_b \quad \Leftrightarrow \quad & \alpha\mathbf{w}\mathbf{v}^{\mathrm{T}}\mathbf{u}_2 &= \mathbf{r}_b \\ \Delta\mathbf{B}\mathbf{u}_1 &= \mathbf{r}_c \quad \Leftrightarrow \quad & \alpha\mathbf{v}\mathbf{w}^{\mathrm{T}}\mathbf{u}_1 &= \mathbf{r}_c \end{aligned}$$

  from which we obtain that

$$\begin{aligned} \mathbf{w} &= \mathbf{r}_b/\alpha\mathbf{v}^{\mathrm{T}}\mathbf{u}_2 \\ \mathbf{v} &= \mathbf{r}_c/\alpha\mathbf{w}^{\mathrm{T}}\mathbf{u}_1 \end{aligned}$$

- With the choice of

$$\boxed{\mathbf{w} = \mathbf{r}_b} \quad \text{and} \quad \boxed{\mathbf{v} = \mathbf{r}_c}$$

  we can show that α is compatible and given as

$$\boxed{\alpha = 1/\mathbf{v}^{\mathrm{T}}\mathbf{u}_2 = 1/\mathbf{w}^{\mathrm{T}}\mathbf{u}_1}$$

Preconditioning Saddle Point Formulation of 4D-Var

- As a result, an inexact constraint preconditioner $\mathcal{P}$ can be updated from

$$\mathcal{P}_{j+1} = \mathcal{P}_j + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^T \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} = \mathcal{P}_j + \begin{pmatrix} \mathbf{0} & \alpha\mathbf{w}\mathbf{v}^{\mathrm{T}} \\ \alpha\mathbf{v}\mathbf{w}^{\mathrm{T}} & \mathbf{0} \end{pmatrix},$$

  where $\mathbf{w} = \mathbf{r}_b$, $\mathbf{v} = \mathbf{r}_c$ and $\alpha = 1/\mathbf{v}^{\mathrm{T}}\mathbf{u}_2$.

- We can rewrite this formula as

$$\mathcal{P}_{j+1} = \mathcal{P}_j + \underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{w} \\ \mathbf{v} & \mathbf{0} \end{pmatrix}}_{\mathbf{F}} \underbrace{\begin{pmatrix} \alpha\mathbf{w}^T & \mathbf{0} \\ \mathbf{0} & \alpha\mathbf{v}^T \end{pmatrix}}_{\mathbf{G}}$$

  where $\mathbf{F}$ is an $(2n+m)$-by-2 matrix and $\mathbf{G}$ is an 2-by-$(2n+m)$ matrix.

- Using the Sherman-Morrison-Woodbury formula on this equation gives the inverse update as

$$\boxed{\mathcal{P}_{j+1}^{-1} = \mathcal{P}_j^{-1} - \mathcal{P}_j^{-1}\mathbf{F}(\mathbf{I}_2 + \mathbf{G}\mathcal{P}_j^{-1}\mathbf{F})^{-1}\mathbf{G}\mathcal{P}_j^{-1}}$$

Preconditioning Saddle Point Formulation of 4D-Var

- Assume that we have set of equations satisfying

$$\Delta\mathbf{B}^{\mathrm{T}}\mathbf{U}_1 = \mathbf{R}_b,$$
$$\Delta\mathbf{B}\,\mathbf{U}_2 = \mathbf{R}_c,$$

  where $\mathbf{U}_1, \mathbf{U}_2, \mathbf{R}_b$ and $\mathbf{R}_c$ are full-rank column matrices.

- An inexact constraint preconditioner $\mathcal{P}$ can be updated by using the block formula

$$\mathcal{P}_{j+1} = \mathcal{P}_0 + \begin{pmatrix} \mathbf{0} & \Delta\mathbf{B}^{\mathrm{T}} \\ \Delta\mathbf{B} & \mathbf{0} \end{pmatrix} = \mathcal{P}_0 + \underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{W} \\ \mathbf{V} & \mathbf{0} \end{pmatrix}}_{\mathbf{F}\in\mathbb{R}^{(2n+m)\times 2j}} \underbrace{\begin{pmatrix} \mathbf{Z}\mathbf{W}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}^T\mathbf{V}^T \end{pmatrix}}_{\mathbf{G}\in\mathbb{R}^{2j\times(2n+m)}}$$

  where $\boxed{\mathbf{W} = \mathbf{R}_b}$, $\boxed{\mathbf{V} = \mathbf{R}_c}$ and $\boxed{\mathbf{Z} = (\mathbf{U}_2^{\mathrm{T}}\mathbf{V})^{-1} = (\mathbf{U}_1^{\mathrm{T}}\mathbf{W})^{-1}}$

- The inverse update similarly is written as

$$\boxed{\mathcal{P}_{j+1}^{-1} = \mathcal{P}_0^{-1} - \mathcal{P}_0^{-1}\mathbf{F}(\mathbf{I}_{2j} + \mathbf{G}\mathcal{P}_0^{-1}\mathbf{F})^{-1}\mathbf{G}\mathcal{P}_0^{-1},}$$

  by using the Sherman-Morrison-Woodbury formula.

Preconditioning Saddle Point Formulation of 4D-Var

- We have shown that it is possible to find a low-cost low-rank update for the inexact constraint preconditioner.

  → This update is not unique!

- It has been found that the most sucessful updates are the ones which solves

  $$\min_{\Delta \mathbf{B} \in \mathcal{S}} \|\Delta \mathbf{B}\|_F$$

  where $\|.\|_F$ is the Frobenius norm. It helps preserve information from the previous iterations.

- Among all updates, the update that we have introduced is the least Frobenius norm update?

  → NO!

→ Next slides are dedicated to find the least-Frobenius norm update.

Preconditioning Saddle Point Formulation of 4D-Var

- Remember that we want to find an update such that

$$\Delta\mathbf{B}^{\mathrm{T}}\mathbf{u}_1 = \mathbf{r}_b \tag{1}$$

$$\Delta\mathbf{B}\,\mathbf{u}_2 = \mathbf{r}_c \tag{2}$$

- Any solution $\Delta\mathbf{B}$ satisfying Equation (1) can be written as [Lemma 2.1](Sun 1999)

$$\Delta\mathbf{B}^{\mathrm{T}} = \mathbf{r}_b{\mathbf{u}_2}^{\dagger} + \mathbf{S}(\mathbf{I} - \mathbf{u}_2\mathbf{u}_2^{\dagger}),$$

  where † denotes the pseudo-inverse and $\mathbf{S}$ is an $(n+m)\times n$ matrix. Inserting this relation into (2) yields

$$\mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{r}_b{}^{\mathrm{T}}\mathbf{u}_1 + (\mathbf{I} - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{u}_2{}^{\mathrm{T}})\mathbf{S}^{\mathrm{T}}\mathbf{u}_1 = \mathbf{r}_c.$$

- If this equation admits one solution, its least Frobenius norm solution,

$$\min_{\mathbf{S}^{\mathrm{T}}\in\mathbb{R}^{m\times n}} \|(\mathbf{r}_c - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{r}_b^{\mathrm{T}}\mathbf{u}_1) - (\mathbf{I} - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{u}_2{}^{\mathrm{T}})\mathbf{S}^{\mathrm{T}}\mathbf{u}_1\|_F,$$

  can be written as [Lemma 2.3](Sun 1999)

$$(\mathbf{S}^{\mathrm{T}})^* = (\mathbf{I} - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{u}_2{}^{\mathrm{T}})^{\dagger}(\mathbf{r}_c - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{r}_b^{\mathrm{T}}\mathbf{u}_1)\mathbf{u}_1^{\dagger}$$

Preconditioning Saddle Point Formulation of 4D-Var

- Substituting the solution for **S** into $\Delta\mathbf{B}$ yields that

$$\boxed{\Delta\mathbf{B}^{*} = \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{r}_b^{\mathrm{T}} + (\mathbf{I} - \mathbf{u}_2{}^{\mathrm{T}\dagger}\mathbf{u}_2{}^{\mathrm{T}})\mathbf{r}_c\mathbf{u}_1{}^{\dagger}}$$

- This formula can be rewritten as

$$\Delta\mathbf{B}^{*} = \begin{bmatrix}\mathbf{u}_2^{\mathrm{T}\dagger} & (\mathbf{I} - \mathbf{u}_2^{T\dagger}\mathbf{u}_2^{\mathrm{T}})\end{bmatrix}\begin{bmatrix}\mathbf{r_b}{}^{\mathrm{T}} \\ \mathbf{r}_c\mathbf{u}_1^{\dagger}\end{bmatrix} = \mathbf{V}\mathbf{W}^{\mathrm{T}},$$

  where **V** is an n-by-$(n+1)$ matrix and **W** is an $(n+m)$-by-$(n+1)$ matrix.

- The preconditioner can be updated by using the following formula

$$\mathcal{P}_1 = \mathcal{P}_0 + \begin{pmatrix}\mathbf{0} & \mathbf{W}\mathbf{V}^{T} \\ \mathbf{V}\mathbf{W}^{T} & \mathbf{0}\end{pmatrix} = \mathcal{P}_0 + \underbrace{\begin{pmatrix}\mathbf{0} & \mathbf{W} \\ \mathbf{V} & \mathbf{0}\end{pmatrix}}_{\mathbf{F}}\underbrace{\begin{pmatrix}\mathbf{W}^{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}^{T}\end{pmatrix}}_{\mathbf{G}}$$

Preconditioning Saddle Point Formulation of 4D-Var

- The inverse formula is then given by

$$\boxed{\mathcal{P}_1^{-1} = \mathcal{P}_0^{-1} - \mathcal{P}_0^{-1}\mathbf{F}(\mathbf{I}_{2(n+1)} + \mathbf{G}\mathcal{P}_0^{-1}\mathbf{F})^{-1}\mathbf{G}\mathcal{P}_0^{-1}}$$

  where **F** is an $(2n+m)$-by-$2(n+1)$ matrix and **G** is an $2(n+1)$-by-$(2n+m)$ matrix.

- When we have system of equations, the block formula can be used to update the preconditioner

$$\boxed{\mathcal{P}_j^{-1} = \mathcal{P}_0^{-1} - \mathcal{P}_0^{-1}\mathbf{F}(\mathbf{I}_{2(n+j)} + \mathbf{G}\mathcal{P}_0^{-1}\mathbf{F})^{-1}\mathbf{G}\mathcal{P}_0^{-1}}$$

  where **F** is an $(2n+m)$-by-$2(n+j)$ matrix and **G** is an $2(n+j)$-by-$(2n+m)$ matrix.

$\rightarrow$ We have found the least-Frobenius update however for n is large this preconditioner is computationally expensive.

Outline

- Problem formulation and the standard solution
- Saddle point approach
- Preconditioning saddle point formulation
- Numerical results
- Conclusions

Numerical Results

Implementation platform

- We used the Object Oriented Prediction System (OOPS)
- 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

Numerical Results

Methods

1. Forcing formulation with **D** preconditioning
   → Solution method is preconditioned conjugate-gradients
2. Saddle point formulation with an updated inexact constraint preconditioner
   → Solution method is GMRES
   → The initial preconditioner is chosen as

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

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

   → The low-rank low-cost preconditioner is denoted as $\mathcal{C}_k$
   → The low-rank least-Frobenius preconditioner is denoted as $\mathcal{F}_k$

Numerical Results with OOPS qg-model

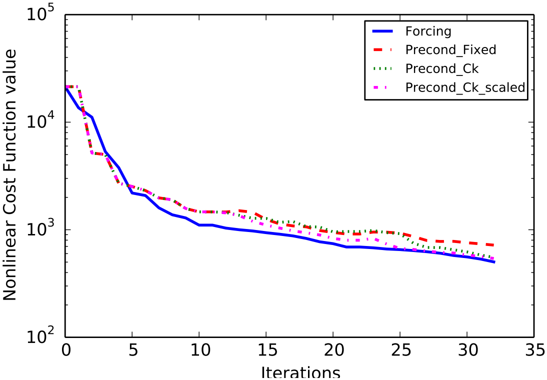


Figure: Nonlinear cost function values along iterations

→ Last 8 pairs were used to construct the preconditioner

→ $\Delta \mathbf{B} = \alpha \mathbf{v}\mathbf{w}^{\mathrm{T}}$ (where $\mathbf{v} = \mathbf{r}_c \in \mathbb{R}^n$, $\mathbf{w} = \mathbf{r}_b \in \mathbb{R}^{n+m}$ and $\alpha = 1/\mathbf{v}^{\mathrm{T}}\mathbf{u_2}$)

Numerical Results with OOPS qg-model

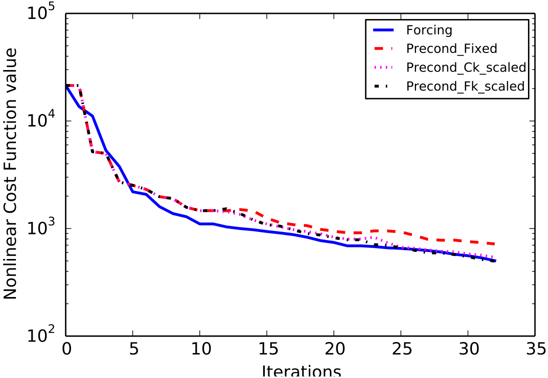


Figure: Nonlinear cost function values along iterations

→ Last 8 pairs were used to construct the preconditioner

Numerical Results with OOPS qg-model

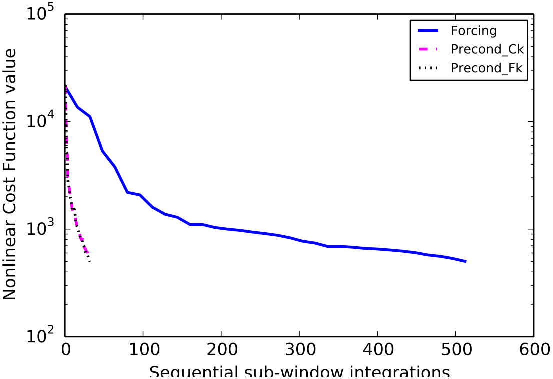


Figure: Nonlinear cost function values along sequential subwindow integrations

→ At each iteration the forcing 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

Outline

- ► Problem formulation and the standard solution
- ► Saddle point approach
- ► Preconditioning saddle point formulation
- ► Numerical results
- ► Conclusions

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 formula can be found to further improve preconditioning.
- The preconditioned GMRES algorithm for saddle point formulation is competitive with the existing algorithms and has the potential to allow 4D-Var to remain computationally viable on next-generation computer architectures.

Thank you for your attention !