Computational Science Laboratory Technical Report CSL-TR-2-2013 July 22, 2013

Alexandru Cioaca,
Adrian Sandu, and Eric de Sturler

Efficient methods for computing observation impact in 4D-Var data assimilation

Computational Science Laboratory
Computer Science Department
Virginia Polytechnic Institute and State University
Blacksburg, VA 24060
Phone: (540)-231-2193
Fax: (540)-231-6075
Email: sandu@cs.vt.edu
Web: http://csl.cs.vt.edu

Innovative Computational Solutions

Efficient methods for computing observation impact in 4D-Var data assimilation

Alexandru Cioaca, Adrian Sandu, Eric de Sturler

Abstract

This paper presents a practical computational approach to quantify the effect of individual observations in estimating the state of a system. Such an analysis can be used for pruning redundant measurements, and for designing future sensor networks.

The mathematical approach is based on computing the sensitivity of the reanalysis (unconstrained optimization solution) with respect to the data. The computational cost is dominated by the solution of a linear system, whose matrix is the Hessian of the cost function, and is only available in operator form. The right hand side is the gradient of a scalar cost function that quantifies the forecast error of the numerical model. The use of adjoint models to obtain the necessary first and second order derivatives is discussed. We study various strategies to accelerate the computation, including matrix-free iterative solvers, preconditioners, and an in-house multigrid solver. Experiments are conducted on both a small-size shallow-water equations model, and on a large-scale numerical weather prediction model, in order to illustrate the capabilities of the new methodology.

1. Introduction

Data assimilation is the process that combines prior information, numerical model predictions, observational data, and the corresponding error statistics, to produce a better estimate of the state of a physical system. In this paper we consider the four dimensional variational (4D-Var) approach, which formulates data assimilation as a nonlinear optimization problem constrained by the numerical model. The initial conditions (as well as boundary conditions, forcings, or model parameters) are adjusted such as to minimize the discrepancy between the model trajectory and a set of time-distributed observations. In real-time operations, the analysis is performed in cycles: observations within an assimilation time window are used to obtain an optimal trajectory, which provides the initial condition for the next time window, and the process is repeated.

The quality and availability of observational data have a considerable impact on the accuracy of the resulting reanalysis (optimal initial conditions). We are interested to quantify rigorously the impact that different observations have

on the result of data assimilation. The assessment of contributions of observations has important applications such as detecting erroneous data (e.g., due to faulty sensors), pruning redundant or unimportant data, and finding the most important locations where future sensors should be deployed.

Early studies of observation impact were concerned with quantifying the predictability of the numerical model, using breeding vectors, potential vorticity and singular vectors [1, 2]. It was assumed that observations in areas of high uncertainty would significantly improve the reanalysis, which led to the concept of targeted and adaptive observations. Later research developed specialized methods such as ensemble transformation techniques [3, 4] and adjoint-based model sensitivity [5, 6]. Some of this research was validated through Observing System Simulation Experiments (OSSEs) [7, 8, 9]. Recent research shifted focus from the numerical model to studying the entire data assimilation system for ensemble-based methods [10], 3D-Var [11], nonlinear 4D-Var [12, 13] and incremental 4D-Var [14]. Important alternative approaches to asses the importance of observations are based on statistical design [15] and information theory [16, 17].

The focus of this work is on the sensitivity of the 4D-Var reanalysis to observations. The sensitivity equations are derived rigorously in the theoretical framework of optimal control and optimization [18, 19, 20]. Sensitivity analysis reveals subsets of data, and areas in the computational domain, which have a large contribution in reducing (or increasing) the forecast error. The solution of the 4D-Var sensitivity equations involves the solution of a linear system, whose system matrix is the Hessian of the 4D-Var cost function. This matrix is typically very large and available only in the form of matrix-vector products.

This work addresses two challenges associated with computing sensitivities to observations. The first challenge is the computation of the required first and second order derivatives. The solution discussed herein is based on first and second order adjoint models. The second challenge is obtaining an accurate solution of the large linear system that defines the sensitivities. Computational time is an important consideration, especially in applications where the solution is needed real-time. Several solutions are proposed in this work. A set of preconditioners is selected and tested to speed up the convergence of Krylov solvers. A multigrid strategy is also considered. Tests are conducted using two numerical models. The first one is the 2D shallow water equations, for which all the derivatives can be computed very accurately. The second test is the Weather Research and Forecast (WRF) model, widely used in numerical weather prediction. The experimental results illustrate the potential of the proposed computational approaches to speed up observation impact calculations in real life applications.

The paper is organized as follows. Section 2 reviews the 4D-Var data assimilation approach. Section 3 covers the theoretical framework of sensitivity analysis in the context of 4D-Var, and derives the equations for the sensitivities to observations. Section 4 discusses practical computational algorithms and their application to the shallow water equations. Section 5 presents the results obtained with the large-scale Weather Research and Forecast (WRF) model. A

qualitative discussion of the results is provided in Section 6. Conclusions are drawn in Section 7, and several directions of future research are highlighted.

2. Data Assimilation

Data assimilation (DA) is the process by which measurements are used to constrain model predictions [21, 22]. For this, three sources of information are combined: an a priori estimate of the state of the system (the "background"), knowledge of the physical laws governing the evolution of the system (captured by the numerical model), and sparse observations of the system. In four dimensional variational (4D-Var) assimilation an optimal initial state $\mathbf{x}_0^{\mathrm{a}}$ (the "reanalysis") is obtained by minimizing the cost function

$$\begin{aligned} \mathcal{J}(\mathbf{x}_0) &= \frac{1}{2}\left(\mathbf{x}_0 - \mathbf{x}_0^{\mathrm{b}}\right)^T \cdot \mathbf{B}_0^{-1} \cdot \left(\mathbf{x}_0 - \mathbf{x}_0^{\mathrm{b}}\right) \qquad (1a) \\ &\quad + \frac{1}{2}\sum_{k=0}^{N} \left(\mathcal{H}_k(\mathbf{x}_k) - \mathbf{y}_k\right)^T \cdot \mathbf{R}_k^{-1} \cdot \left(\mathcal{H}_k(\mathbf{x}_k) - \mathbf{y}_k\right) , \\ \mathbf{x}_0^{\mathrm{a}} &= \arg\min_{\mathbf{x}_0} \mathcal{J}(\mathbf{x}_0) \, . \qquad (1b) \end{aligned}$$

The first term of the sum (1a) quantifies the departure of the solution from the background state $\mathbf{x}_0^{\mathrm{b}}$ at the initial time t_0. The term is scaled by the inverse of the background error covariance matrix $\mathbf{B}_0$. The second term measures the mismatch between the forecast trajectory and the observations $\mathbf{y}_k$, which are taken at times $t_0, \ldots, t_N$ inside the assimilation window. When assimilating observations only at the initial time t_0, the method is known as three dimensional variational (3D-Var), as the additional "time" dimension is not present. $\mathcal{M}$ is the numerical model used to evolve the state vector $\mathbf{x}$ in time. $\mathcal{H}_k$ is the observation operator at assimilation time t_k, and maps the discrete model state $\mathbf{x}_k \approx \mathbf{x}(t_k) = \mathcal{M}_{t_0 \to t_k}(\mathbf{x}_0)$ to the observation space. $\mathbf{R}_k$ is the observations error covariance matrix. The weighting matrices $\mathbf{B}_0$ and $\mathbf{R}_k$ need to be predefined in order to have a fully-defined problem, and their quality influences the accuracy of the resulting reanalysis.

Since an analytical solution for the equation (1b) is not possible, the minimizer is computed iteratively using numerical optimization methods. Such methods typically require the gradient of the cost function, while Newton-type methods also require second-order derivative information. Higher-order information can be computed using techniques from the theory of adjoint sensitivity analysis [23]. In this case, first-order adjoint models provide the gradient of the cost function, while second-order adjoint models provide the Hessian-vector product. The methodology of building and using various adjoint models for optimization, sensitivity analysis, and uncertainty quantification can be found in [24, 25].

When 4D-Var is employed in an operational setting (in real time), the reanalysis (1b) has to be determined within a given time limit, and the iterative solver is stopped after a certain number of iterations, typically before complete

convergence. Although the most significant decrease in the cost function usually happens during the first iterations, it is likely the analysis is approximate and does not satisfy exactly the optimality conditions. Slow convergence is a known issue for the solution of highly nonlinear problems of PDE-constrained optimization. The resulting reanalysis can be interpreted as only partially assimilating the observations. Along with the problem of correctly defining the error statistics, it represents one of the practical challenges of data assimilation.

3. Sensitivity of the Analysis to Observations

The sensitivity of the analysis to observations is derived in the context of unconstrained optimization, and the presentation follows [19]. Consider the problem of finding a vector $\mathbf{x} = (x_1, x_2, ..., x_n)^T \in \mathbb{R}^n$ that minimizes the twice continuously differentiable cost function

$$\min_{\mathbf{x}} \mathcal{J}(\mathbf{x}, \mathbf{u}) \,.$$

The function also depends on the vector of parameters $\mathbf{u} \in \mathbb{R}^m$. The implicit function theorem applied to the first order optimality condition

$$\nabla_{\mathbf{x}} \mathcal{J}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = 0 \tag{2}$$

guarantees there exists a vicinity of $\bar{\mathbf{u}}$ where the optimal solution is a smooth function of the input data, $\mathbf{x} = \mathbf{x}(\mathbf{u})$ and $\nabla_{\mathbf{x}} \mathcal{J}(\mathbf{x}(\mathbf{u}), \mathbf{u}) = 0$. The sensitivity of the optimal solution with respect to the parameters

$$\nabla_{\mathbf{u}} \mathbf{x} = (\nabla_{\mathbf{u}} \mathbf{x}_1, \nabla_{\mathbf{u}} \mathbf{x}_2, ..., \nabla_{\mathbf{u}} \mathbf{x}_n) \in \mathbb{R}^{m \times n}$$

can be expressed as

$$\nabla_{\mathbf{u}} \mathbf{x}(\mathbf{u}) = -\nabla^2_{\mathbf{u},\mathbf{x}} \mathcal{J}(\mathbf{u}, \mathbf{x}) \cdot \left[\nabla^2_{\mathbf{x}_0,\mathbf{x}_0} \mathcal{J}(\mathbf{u}, \mathbf{x})\right]^{-1} . \tag{3}$$

Consider now a scalar functional $\mathcal{E}$ that represents some quantity of interest of the optimal solution, $\mathcal{E}(\mathbf{x}(\mathbf{u}))$. Using chain rule differentiation we obtain its sensitivity to parameters

$$\nabla_{\mathbf{u}} \mathcal{E} = \nabla_{\mathbf{u}} \mathbf{x} \cdot \nabla_{\mathbf{x}} \mathcal{E} = -\nabla^2_{\mathbf{u},\mathbf{x}} \mathcal{J} \cdot (\nabla^2_{\mathbf{x}_0,\mathbf{x}_0} \mathcal{J})^{-1} \cdot \nabla_{\mathbf{x}} \mathcal{E} \,. \tag{4}$$

For the 4D-Var cost function (1a) the first-order necessary condition reads

$$\nabla_{\mathbf{x}_0} \mathcal{J}(\mathbf{x}_0^{\mathrm{a}}) = \mathbf{B}_0^{-1} \left(\mathbf{x}^{\mathrm{a}} - \mathbf{x}^{\mathrm{b}}\right) + \sum_{k=1}^{N} \mathbf{M}_{0,k}^T \mathbf{H}_k^T \mathbf{R}_k^{-1} \left(\mathcal{H}_k(\mathbf{x}_k) - \mathbf{y}_k\right) = 0 \,, \tag{5}$$

where $\mathbf{M}_{0,k} = (\mathcal{M}_{t_0 \to t_k})'$ is the tangent linear propagator associated with the numerical model $\mathcal{M}$, and $\mathbf{H}_k = (\mathcal{H}_k)'$ is the tangent linear approximation of the observation operator. Differentiating (5) with respect to observations $\mathbf{y}_k$ yields

$$\nabla^2_{\mathbf{y}_k,\mathbf{x}_0} \mathcal{J}(\mathbf{x}_0^{\mathrm{a}}) = -\mathbf{R}_k \, \mathbf{H}_k \, \mathbf{M}_{0,k} \,, \tag{6}$$

which then provides the following analysis sensitivity to observations

$$\nabla_{\mathbf{y}_k} \mathbf{x}_0^{\mathrm{a}} = \mathbf{R}_k^{-1} \mathbf{H}_k \mathbf{M}_{0,k} \left(\nabla_{\mathbf{x}_0,\mathbf{x}_0} \mathcal{J}(\mathbf{x}_0^{\mathrm{a}})\right)^{-1} . \tag{7}$$

In the context of data assimilation we consider $\mathcal{E}(\mathbf{x}^{\mathrm{a}})$ to be a forecast score, i.e., a performance metric for the quality of the reanalysis. If the 4D-Var problem is defined and solved correctly, and if the data is accurate, then the reanalysis $\mathbf{x}^{\mathrm{a}}$ should provide a better forecast than the background $\mathbf{x}^{\mathrm{b}}$; this is quantified by $\mathcal{E}(\mathbf{x}^{\mathrm{a}}) \leq \mathcal{E}(\mathbf{x}^{\mathrm{b}})$. Validating the forecast against a reference solution is often used as a way to assess the quality of the initial condition. Since one does not have access to the state of the real system, the reanalysis is verified against another solution of higher accuracy (the "verification" forecast). Specifically, we define the forecast score as

$$\mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = (\mathbf{x}_{\mathrm{f}}^{\mathrm{a}} - \mathbf{x}_{\mathrm{f}}^{\mathrm{v}})^T \mathbf{C} \, (\mathbf{x}_{\mathrm{f}}^{\mathrm{a}} - \mathbf{x}_{\mathrm{f}}^{\mathrm{v}}) \tag{8}$$

where $\mathbf{x}_{\mathrm{f}}^{\mathrm{a}} = \mathcal{M}_{t_0 \to t_{\mathrm{f}}}(\mathbf{x}_0^{\mathrm{a}})$ is the model forecast at verification time t_{f}, $\mathbf{x}_{\mathrm{f}}^{\mathrm{v}}$ is the verification forecast at t_{f}, and C is a weighting matrix that defines the metric in the state space. For example, C could restrict $\mathcal{E}$ to a subset of grid points, in which case we will quantify the influence of assimilated observations in reducing the forecast error in the corresponding subdomain.

Using the chain rule differentiation for the forecast score we obtain

$$\nabla_{\mathbf{y}_k} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = \nabla_{\mathbf{y}_k} \mathbf{x}_0^{\mathrm{a}} \cdot \nabla_{\mathbf{x}_0^{\mathrm{a}}} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) \, .$$

This leads to the following expression for the forecast sensitivity to observations

$$\nabla_{\mathbf{y}_k} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = \mathbf{R}_k^{-1} \mathbf{H}_k \mathbf{M}_{0,k} \left(\nabla_{\mathbf{x}_0,\mathbf{x}_0} \mathcal{J}(\mathbf{x}_0^{\mathrm{a}})\right)^{-1} \nabla_{\mathbf{x}_0^{\mathrm{a}}} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) \, . \tag{9}$$

Obtaining the sensitivity (9) is the main goal of this paper. We summarize the big picture from a systems theory perspective. Data assimilation takes as inputs the following parameters: the background estimate of the state of the atmosphere, the observations, the error statistics, and the forecast model. It produces a better initial condition. We perform a forecast using this new estimate, and compute a metric of the forecast error as the mismatch against a verification forecast. We trace back the reduction of the forecast error to the input parameters (specifically, to the observations). This process involves the following three computational steps.

3.1. Forecast sensitivity to reanalyzed initial condition

We first compute the sensitivity of the forecast score (8) to the optimal initial condition:

$$\nabla_{\mathbf{x}_0^{\mathrm{a}}} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = \mathbf{M}_{0,\mathrm{f}}^T \cdot \nabla_{\mathbf{x}_{\mathrm{f}}^{\mathrm{a}}} \mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = 2 \, \mathbf{M}_{0,\mathrm{f}}^T \cdot \mathbf{C} \cdot (\mathbf{x}_{\mathrm{f}}^{\mathrm{a}} - \mathbf{x}_{\mathrm{f}}^{\mathrm{v}}) \, . \tag{10}$$

The gradient (10) is computed by running the first-order adjoint model, initialized with the forecast error $\mathbf{x}_{\mathrm{f}}^{\mathrm{a}} - \mathbf{x}_{\mathrm{f}}^{\mathrm{v}}$. The first-order adjoint model evolves the forecast error field backward in time to produce a field of sensitivities at the initial time. This calculation reveals regions in the initial condition to which the output (forecast error, in this case) is most sensitive. This step requires just one adjoint model run and does not add a significant computational load to the method as a whole.

3.2. Forecast sensitivity through the 4D-Var system

The second step consists in solving a large-scale linear system of the form:

$$\nabla^2_{\mathbf{x}_0,\mathbf{x}_0}\mathcal{J}(\mathbf{x}_0^{\mathrm{a}})\cdot\mu_0 = \nabla_{\mathbf{x}_0^{\mathrm{a}}}\mathcal{E}(\mathbf{x}_0^{\mathrm{a}})\,. \tag{11}$$

The system matrix is the Hessian of the 4D-Var cost function evaluated at the reanalysis. The right-hand side is the vector of sensitivities (10). The linear system (11) solves the matrix-vector product $(\nabla^2_{\mathbf{x}_0,\mathbf{x}_0}\mathcal{J})^{-1}\,\nabla_{\mathbf{x}_0}\mathcal{E}$ in (9). The inverse of the 4D-Var Hessian approximates the covariance matrix of the reanalysis error [26, 27]. The solution μ_0 will be referred to as "supersensitivity", and is a crucial ingredient for the computation of forecast sensitivities to all data assimilation parameters . The present work focuses on efficiently solving the linear system (11), as it presents the main computational burden of the entire methodology.

3.3. Forecast sensitivity to the 4D-Var parameters

From (9) the forecast sensitivity to observations is obtained as follows:

$$\begin{aligned}\mu_k &= \mathbf{M}_{0,k}\,\mu_0\,,\\ \nabla_{\mathbf{y}_k}\mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) &= \mathbf{R}_k^{-1}\,\mathbf{H}_k\,\mu_k\,.\end{aligned}$$

The index k selects the observation time t_k. The supersensitivity μ_0 at t_0 is propagated forward to time t_k using the tangent linear model, to obtain the vector μ_k. This solution is applied the linearized observation operator $\mathbf{H}_k$, and is scaled by $\mathbf{R}_k^{-1}$, the inverse covariance matrix of the observational errors. The sensitivity equations for other parameters can be found in [19]. For example, the forecast sensitivity to the background estimate is

$$\nabla_{\mathbf{x}_0^{\mathrm{b}}}\mathcal{E}(\mathbf{x}_0^{\mathrm{a}}) = \mathbf{B}_0^{-1}\,\mu_0\,.$$

This provides insight about the meaning of supersensitivity: it represents a time-dependent field that quantifies the sensitivity of the forecast score to the information assimilated at a certain time. At t_0 this information is the background, and at other times is the observations.

4. Numerical Tests with the Shallow Water Equations

4.1. Numerical model

The first model used to study the performance of the computational methodology is based on the shallow-water equations (SWE). The two-dimensional PDE system (12) approximates a thin layer of fluid inside a shallow basin:

$$\begin{aligned}\frac{\partial}{\partial t}h + \frac{\partial}{\partial x}(uh) + \frac{\partial}{\partial y}(vh) &= 0\\ \frac{\partial}{\partial t}(uh) + \frac{\partial}{\partial x}\left(u^2h + \frac{1}{2}gh^2\right) + \frac{\partial}{\partial y}(uvh) &= 0\\ \frac{\partial}{\partial t}(vh) + \frac{\partial}{\partial x}(uvh) + \frac{\partial}{\partial y}\left(v^2h + \frac{1}{2}gh^2\right) &= 0\,.\end{aligned} \tag{12}$$

Here $h(t, x, y)$ is the fluid layer thickness, and $u(t, x, y)$ and $v(t, x, y)$ are the components of the velocity field. The gravitational acceleration is denoted by g. The spatial domain is $\Omega = [-3, 3]^2$ (spatial units), and the integration window is $t_0 = 0 \leq t \leq t_f = 0.1$ (time units).

The numerical model uses a finite volume-type scheme for space discretization and a fourth-order Runge-Kutta scheme for time discretization [28]. A square $q \times q$ discretization grid is used, and the numerical model has $n = 3q^2$ variables

$$\mathbf{x} = \begin{bmatrix} \hat{h} \\ \hat{uh} \\ \hat{vh} \end{bmatrix} \in \mathbb{R}^n \ .$$

We call the discretized system of equations *the forward model* (FWD), used to simulate the evolution of the nonlinear system (12) forward in time. We are interested in computing the derivatives of a cost function $\mathcal{J}(\mathbf{x}_0)$ with respect to model parameters, like the initial condition. These derivatives can be computed efficiently using adjoint modeling. The theory and applications of adjoint models to data assimilation can be found in [29, 30]. The distinction is made between continuous adjoints, obtained by linearizing the differential equations, and discrete adjoints, obtained by linearizing the numerical method. Construction of adjoint models is a work intensive and error prone process. An attractive approach is automatic differentiation (AD) [31]. This procedure parses the source code of the FWD model and generates the code for the discrete adjoint model using line by line differentiation.

We build the adjoint SWE model through automatic differentiation using the TAMC tool [32, 33]. The tangent-linear model (TLM) propagates perturbations forward in time. The first-order adjoint model (FOA) propagates perturbations backwards in time, and efficiently computes the gradient of a scalar cost function of interest ($\nabla_{\mathbf{x}_0} \mathcal{J}$). The second-order adjoint model (SOA) computes the product between the Hessian of the cost function and a user-defined vector ($\nabla^2_{\mathbf{x}_0,\mathbf{x}_0} \mathcal{J} \cdot u$) [25]. Second-order adjoint models are considered to be the best approach to compute Hessian-vector products, but have yet to become popular in practice because of their computational demands. When one does not have access to the second-order adjoint, Hessian-vector products can be computed through various approximations, such as finite differences of first order adjoints.

The overhead of running adjoint models has to be taken into account for the design of the computational strategy. Table 1 presents the CPU times of TLM, FOA and SOA shallow models, normalized with respect to the CPU time of a single FWD model run. One SOA integration is about 3.5 times more expensive than a single first-order adjoint run, while the FOA takes 3.7 times longer than the forward run. The adjoint model runs take a significant computational time. This effort depends on the numerical methods used in the FWD model, and on the automatic differentiation tool employed. For certain numerical methods it is possible to develop efficient strategies based on reusing computations, which lead to adjoint times smaller than forward model times. An example can be found in [25] where the adjoint SWE equations are derived by hand and then

FWD	1		
TLM	2.5	FWD + TLM	3.5
FOA	3.7	FWD + FOA	4.7
SOA	12.8	FWD + TLM + FOA + SOA	20

Table 1: Normalized CPU times of different sensitivity models. The forward model takes one time unit to run.

solved numerically.

4.2. Data Assimilation Scenario

The 4D-Var data assimilation system used in the numerical experiments is set up as follows:

- The computational grid uses $q = 40$ grid points in each directions, for a total of 4800 model variables. The timestep is 0.001 (time units).
- The reference solution is obtained as follows. The initial h field is a Gaussian bell centered on the grid. The initial u and v are constant fields. We run the forecast model from the initial solution for 100 time steps. The solutionprovides the reference trajectory for the experimental setup.
- The background solution $\mathbf{x}^b$ is generated by adding a correlated perturbation to the reference solution $\mathbf{x} = [h, u, v]$. The background error covariance $\mathbf{B}_0$ corresponds to a standard deviation of 5% of the reference field values. The spatial error correlation uses a Gaussian decay model, with a correlation distance of 5 grid points. This dictates how the 4D-Var method spreads the information from one grid point to its neighbors.
- Synthetic observations are generated from the reference model results. The observation frequency is set to once every 20 time steps. We add normal random perturbations to simulate observation errors. The observation error covariance matrix $\mathbf{R}$ is diagonal (i.e., the observation errors are uncorrelated). The standard deviation is 1% of the largest absolute value of the observations for each variable.
- The observation operator $\mathcal{H}$ is linear and selects observed variables at specified grid points.

We use the L-BFGS-B solver [34] to minimize the 4D-Var cost function. We allow the solver to run for 400 iterations (which reduces the norm of gradient of the 4D-Var cost function from a magnitude of $1e+7$ to $1e-4$). Note that one cannot afford to obtain such a high quality optimal solution with a large-scale model. The SWE test case allows to compute the sensitivity to observations in a setting where numerical optimization errors are negligible.

4.3. Particularities of the linear system

The solution of the linear system (11) is the central step of the entire computational process. As mentioned in Section 3.1, the right hand side is the gradient of the forecast aspect with respect to initial conditions, and is obtained at the cost of one FOA run. The adjoint model propagates backward in time the mismatch between the forecast and the verification.

The system matrix in (11) is the Hessian of the 4D-Var cost function, evaluated at the reanalysis. For large-scale models like the atmosphere, the Hessian cannot be computed and manipulated in an explicit form due to its dimension. In practice, one evaluates directly the Hessian-vector product by running the second-order adjoint model. When SOA is not available, one can approximate Hessian-vector products through finite differences of FOA gradients.

$$\nabla^2_{\mathbf{x}_0,\mathbf{x}_0}\mathcal{J}(\mathbf{x}_0^{\mathrm{a}})\cdot\mathbf{u}\approx\frac{\nabla_{\mathbf{x}_0}\mathcal{J}(\mathbf{x}_0^{\mathrm{a}}+\epsilon\cdot\mathbf{u})^T-\nabla_{\mathbf{x}_0}\mathcal{J}(\mathbf{x}_0^{\mathrm{a}})^T}{\epsilon}\,. \tag{13}$$

A third method to compute Hessian-vector products is the Gauss-Newton approximation of the Hessian, also known in literature as the "Hessian of the auxiliary cost function":

$$\nabla^2_{\mathbf{x}_0,\mathbf{x}_0}\mathcal{J}(\mathbf{x}_0^{\mathrm{a}})\cdot\mathbf{u}\approx\mathbf{B}_0^{-1}\cdot\mathbf{u}+\sum_{k=1}^{N}\mathbf{M}_{0,k}^T\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\,\mathbf{M}_{0,k}\cdot u\,. \tag{14}$$

The formulation above is obtained in a similar fashion to the formulation of incremental 4D-Var [35], by differentiating the 4D-Var cost function and ignoring higher-order terms. These higher-order terms are negligible when the solution is close to the optimum. Computationally, the Gauss-Newton Hessian-vector product is obtained by running the TLM model forward in time starting from the seed vector, and then using its output to initialize a FOA model run backward in time.

For our SWE model, both finite difference and Gauss-Newton approximations provide Hessian-vector products that verify within machine precision with the Hessian-vector products obtained from second-order adjoint models. However, finite difference is less stable than Gauss-Newton since it relies on perturbing the system.

Yet another strategy is to build limited-memory approximations of the Hessian from information collected during the data assimilation process. In [36] the authors use the Lanczos pairs generated by the iterative solver employed to minimize the 4D-Var cost function. This type of approximation is usually helpful for building preconditioners, but is not accurate enough to be used as the system matrix in (11).

Corresponding to the spatial discretization chosen for our experiment, the size of the model solution is 4800 variables. Accordingly, the size of the 4D-Var Hessian matrix is 4800×4800. The explicit form of this matrix can be obtained through matrix-vector products with the e_i unity vectors (SOA model). This is not feasible in practice, but our SWE model is small enough to allow us

to build the full Hessian and analyze its properties. Thus, we find out the Hessian is symmetric to machine precision, which confirms the superior quality of second-order information obtained with the SOA model. Also, because the 4D-Var optimization problem in Section 4.2 is solved accurately, the reanalysis is close to the optimum and the 4D-Var Hessian evaluated at this point is positive definite. Our tests show that when evaluated far from the optimum, the 4D-Var Hessian is indefinite. This has consequences for real-time operations where only a limited number of iterations are allowed.

The structure of the Hessian matrix exhibits some regularities, characteristic to information matrices and their covariance counterparts. In literature, this structure is known as "near block-Toeplitz" [37]. The first 1600 rows correspond to the model variables of h, the next 1600 rows to u and the last 1600 to v. The matrix elements scale differently in each one of these three blocks. Some obvious features occur on the diagonals, rows and columns, spaced every 40 or 1600 rows and columns. This hints at the fact that the 4D-Var Hessian approximates the inverse of the covariance matrix of the reanalysis errors [26, 27]. We interpret these patterns as arising from due to the discretization scheme stencil (each point of the grid is correlated to its East, West, North, and South neighbors). In addition, each variable is weakly connected to the other two variables, corresponding to a distance of 1600 rows/columns. This structure can be predicted without building the explicit form of the Hessian, from prior information such as the background error covariance matrix $\mathbf{B}_0$.

The spectrum of the matrix is of great interest for our analysis, since it will influence the convergence of the iterative solvers. The eigenvalues of the SWE Hessian are displayed in Figure 1, sorted in ascending order. The condition number of the Hessian (ratio between largest and smallest eigenvalues) is $\sim 10^4$, which makes the matrix moderately well-conditioned. However, since the eigenvalues are not clustered together, we expect slow convergence.

4.4. Matrix-free linear solvers

The choice of solvers for the linear system (11) is limited to "matrix-free" algorithms. Direct solvers and basic iterative methods are ruled out since they require the full system matrix, which is not available. Krylov-based iterative solvers require only matrix-vector products and exhibit superior performance over basic iterative methods. However, their convergence depends on the eigenvalues of the system matrix. As seen in Figure (1), the Hessian is positive definite, but its spectrum is scattered. Preconditioning can considerably improve the convergence of iterative solvers.

Additional challenges arise in large-scale 4D-Var data assimilation. The reanalysis can be far from the minimizer, when the minimizing algorithm is stopped before reaching the minimum; in this case, the resulting Hessian matrix can be indefinite. Although by definition a Hessian matrix is symmetric, the symmetry can be lost when approximations such as finite differences are employed. In an operational setting where the sensitivities are used to target adaptive observations, results have to be delivered in real time; the key is to provide the best possible solution in a given time.

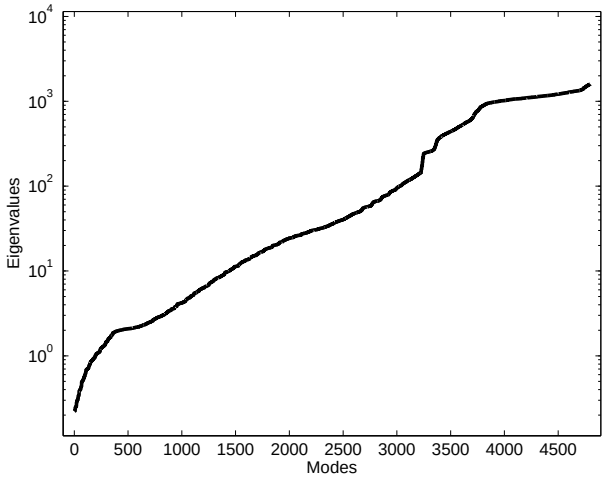


Figure 1: Eigenvalues of the SWE 4D-Var Hessian at the reanalysis (optimal solution), sorted in ascending order.

The matrix-free iterative solvers used to solve the SWE supersensitivity system (11) are listed in Table 2. The list includes the most popular algorithms currently used for large linear systems. Detailed information about each solver can be found in the scientific literature [38, 39].

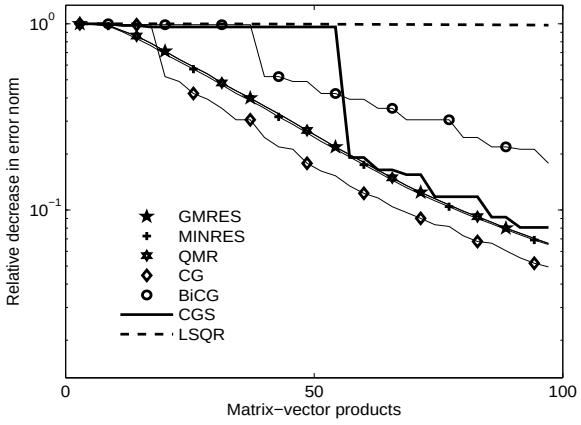
Generalized Minimum Residual	GMRES	nonsymmetric
Minimal Residual	MINRES	symmetric
Conjugate Gradients	CG	symmetric positive-def.
Quasi-Minimum Residual	QMR	nonsymmetric
Biconjugate Gradients Stabilized	BiCGSTAB	nonsymmetric
Conjugate Gradients Squared	CGS	nonsymmetric
Least Squares	LSQR	nonsymmetric

Table 2: List of iterative methods used to solve the (SWE) system (11).

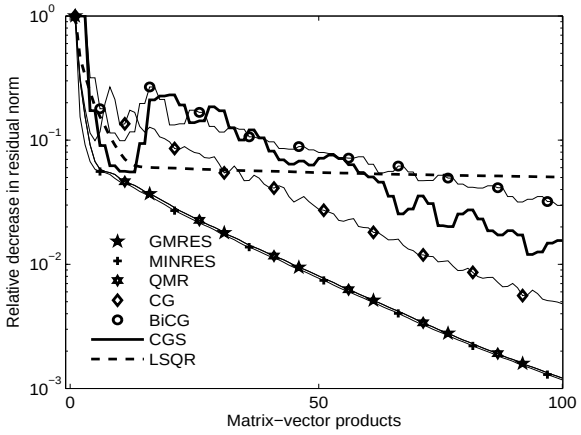
We used the iterative solvers implemented in the PETSc [40] software package. PETSc supports matrix and vector operations and contains an extensive set of solvers and preconditioners. We interfaced PETSc with our shallow water model and solved the linear system with each of the methods above. Also, we double-checked the results with our own Fortran and MATLAB implementation of the algorithms. The initial guess was set to a vector of zeroes and no preconditioner was used for the results presented in this section. We compare the convergence of the linear solvers by monitoring the decrease in the residual norm and the error norm at each iteration. The error norm was computed as a root mean square error with respect to a reference solution μ_0^{REF} obtained by solving the system directly using the full Hessian, and this error metric has the following expression:

$$RMSE = \frac{\|\mu_0 - \mu_0^{REF}\|}{\sqrt{n}}. \tag{15}$$

We allocate a budget of 100 matrix-vector products as SOA runs. BiCGSTAB, CGS use 2 matrix-vector products per iteration, which means 50 iterations. The other solvers use just 1, so they will run for 100 iterations within our budget. Figure 2(a) plots the relative decrease in the norm of the error and Figure 2(b) the relative decrease in the norm of the residual. Table 3 presents the solution error and residual norm decrease after 100 matrix-vector products of each solver.



(a) Error norm



(b) Residual norm

Figure 2: Convergence of non-preconditioned iterative solvers for the (SWE) supersensitivity system (11).

Solver	Relative decrease in residual norm	Relative decrease in error norm
GMRES	2.219e-1	6.62e-2
MINRES	2.164e-1	6.53e-2
PCG	9.461e-1	4.95e-2
QMR	2.219e-1	6.62e-2
BiCGSTAB	9.461e-1	5.54e-2
CGS	1.124e-1	1.48e-2
LSQR	9.792e0	9.83e-1

Table 3: Solution error and residual norms after 100 matrix-vector products of each solver for the (SWE) supersensitivity system (11). The scaling is done with respect to the initial guess error and residual norms, respectively.

The decrease in the solution error and residual norms are as expected from the theory of Krylov solvers. CG provides the best error reduction. GMRES, MINRES and QMR show the best performance for reducing the residual. CG is known for its superior performance over other solvers when dealing with symmetric and positive definite matrices. It acts on reducing the A-norm of the error, as opposed to GMRES, MINRES and QMR, which act upon the residual. For symmetric positive definite matrices, the latter three are equivalent, which explains their similar behavior. CGS and BiCGSTAB exhibit a slow initial convergence, but CGS eventually catches up with GMRES. LSQR has the worst performance, confirming that a least-squares approach is not suitable for solving this problem. In consequence, CG would be the ideal solver to use when we can guarantee the system matrix is symmetric and positive-definite. Otherwise, one should use GMRES (or MINRES), with the amendment that the numerical workload per iteration is slightly larger than for CG.

4.5. Preconditioned Krylov solvers

We next explore preconditioning strategies to improve the convergence of the iterative methods. The Krylov solvers perform better when the matrix eigenvalues are clustered. As seen in Figure 1, the eigenvalues of the SWE Hessian matrix are scattered across various orders of magnitude. This explains why no method converged to the actual solution.

Building effective preconditioners for the supersensitivity linear system (11) is challenging. Preconditioners require a good understanding of the underlying problem and the structure of the matrix; this is difficult without having access to the full system matrix. The matrix-free constraint excludes certain preconditioning techniques such as incomplete factorizations, wavelet-based, or variations of the Schur complement. Moreover, basic preconditioners such as diagonal cannot be constructed solely from matrix-vector products, without a significant computational effort. We consider here preconditioning strategies that rely on curvature information collected during the numerical minimization process. Predicting the structure of the Hessian matrix can also help with the solution of the problem. We next describe the proposed preconditioners.

4.5.1. Diagonal of Hessian

The diagonal of the matrix is one of the most popular practical preconditioners, and was proved to be the optimal diagonal preconditioner in [41]. When we only have access to the matrix under the form of an operator, its diagonal is not readily available. Therefore, we use the diagonal preconditioner in this test only as a reference for the performance of the other preconditioners. In a real setting, one has access to neither the actual diagonal, nor banded or arrow preconditioners.

4.5.2. Diagonal of the background covariance matrix

Preconditioners that do not require any supplementary computations can be obtained from $\mathbf{B}_0$, the covariance matrix of the background errors in 4D-Var. In practice, this matrix cannot be manipulated with ease due to its size. However, its diagonal is accessible, and we use it as a preconditioner in the following tests. This choice has been reported to provide better convergence in incremental 4D-Var under certain conditions [36].

4.5.3. Row sum

The system matrix (11) approximates the inverse of a covariance matrix. Covariance matrices have their larger elements on the diagonal, and under some conditions they have a diagonally dominant structure. Consequently we use the sum of row elements to build an approximation of the diagonal. This can be computed with just one second-order adjoint run, where the Hessian is multiplied by a vector of ones. The diagonal preconditioner used in our tests is built from the output of the second-order adjoint and taking the absolute value.

4.5.4. Probing and extrapolating

This approach takes advantage of the results in [42, 43] where the possibility of block diagonal approximations of the 4D-Var Hessian is explored. The values for a certain variable and for a certain vertical level (not applicable here since we have a 2D model) are assigned a constant value. We approximate these values by using Hessian-vector products to "probe" the matrix. For our three-variable model we run three Hessian-vector products with unity vectors to extract one column (row) of the Hessian at one time. The value of the corresponding diagonal element is used as an approximation for all diagonal elements in that block.

To be specific, we consider three unity vectors for our 4800×4800 Hessian that have the value 1 at positions 1, 1601 and 3201 respectively, and zeros everywhere else. The corresponding Hessian-vector products will extract the columns 1, 1601 and 3201, which correspond to the three different variables in our Hessian. The approximation uses the value found at coordinates $(1, 1)$ for the entire first diagonal block (up to coordinates 1600, 1600), the value found at coordinates $(1601, 1601)$ for the entire second block, and so forth. This approximation can be refined by probing for more elements from the same block. If there are many blocks that have to be probed and the computational burden

increases significantly, one can employ coloring techniques to probe for more than one element with the same matrix-vector product.

4.5.5. Quasi-Newton approximation

The Hessian matrix can also be approximated from data collected throughout the minimization process. Quasi-Newton solvers such as L-BFGS build Hessian approximations, and refine them with information generated at each iteration. These approximations are sufficiently accurate along the descent directions to improve the convergence of the minimization iterations. The approximations preserve matrix properties such as symmetry and positive definiteness, and allow limited memory implementations appropriate for large-scale models. We store the approximation of the Hessian as generated over the last 10 iterations of minimizing the 4D-Var cost function with L-BFGS. This will be used as a preconditioner for the linear system and does not require any supplementary model runs. Our tests showed that using more than 10 vector pairs does not improve further the quality of the resulting preconditioner.

4.5.6. Eigenpairs

This preconditioning method is borrowed from 4D-Var data assimilation literature [36]. During the minimization of the 4D-Var cost function the leading eigenvalues and eigenvectors are calculated via a Lanczos process. An approximation of the Hessian (evaluated at the current reanalysis) can be generated from the leading eigenvalues or eigenvectors, and used as a preconditioner for the supersensitivity system (11). In our tests we use the leading 50 eigenpairs to approximate the Hessian.

4.5.7. Randomized SVD

Randomized SVD [44] computes an approximate singular value decomposition of a matrix only available as an operator. The algorithm requires two ensembles of matrix-vector products, and one singular value decomposition and one QR decomposition with smaller matrices. All matrix-vector products can be executed in parallel as they are independent of each other. The number of input vectors used can vary and the accuracy of the approximation is proportional to the size of the ensemble. For our tests we used 50 different input vectors.

4.5.8. Performance of preconditioned algorithms

The experiments to compare the performance of the preconditioners were conducted with GMRES as the linear solver, because of its generality. The norm of the error against the reference solution and that of the residual are shown in Table 4 and Figures 3(a), 3. A comparison with the results in Table 2 and Figures 2(a), 2 reveals that all preconditioners improve convergence. L-BFGS LMP starts off with the best decrease, but then it stops accelerating, and after 100 iterations has the worst performance among all preconditioners. The preconditioners formed from probing, leading eigenpairs, and randomized SVD, perform almost as well as the exact diagonal. Finally, the row sum preconditioner also shows good results, comparable to the latter preconditioners.

Preconditioner	Relative decrease in residual norm	Relative decrease in error norm
None	1.3e-3	7.2e-3
Diagonal	8.0e-5	1.2e-3
Coloring	8.0e-5	1.2e-3
Row sum	1.2e-4	1.9e-3
L-BFGS	3.8e-4	1.6e-2
Eigenpairs	8.0e-5	1.7e-3
RandSVD	8.0e-5	1.2e-3

Table 4: Solution error and residual norms after 100 non-preconditioned iterations of GMRES for the (SWE) supersensitivity system (11). The scaling is done with respect to the initial guess error and residual norms, respectively.

The conclusion is that some preconditioners can decrease the error after 100 iterations by a factor of up to 100. After 25 iterations the preconditioned algorithm reaches the same accuracy that the unpreconditioned algorithm achieves after 100 iterations. This improvement of 75% in the computation time is very significant for large-scale models.

4.6. Multigrid solver

Multigrid (MG) describes a class of numerical methods that speed up numerical solutions by alternating computations on coarser or finer levels [45, 46]. These methods can be defined geometrically (using a grid) or purely algebraically. We refer to each fine-grid-to-coarse-grid sweep as a "multigrid cycle", "V-cycle" or "cycle" for short.

Our linear system (11) is appropriate for the multigrid approach because one can run the SWE model on different spatial discretizations. Consider the 40×40 grid used in the previous tests as the fine-level grid (4800 variables). We can simulate the same scenario coarser grid, for example 20×20 (1200 variables) and 10×10 (300 variables). For simplicity and clarity, we use only the first two levels in our test.

Traditional MG uses smoothers that require the full matrix, and one challenge is to build a matrix-free approach. Here we use GMRES as smoother. The MG theory does not guarantee convergence for Krylov-based methods, but there are reports of them being used successfully. A second challenge consists in designing the operators that transfer the problem between grids. One needs to restrict the residual of the linear system from the fine grid to the coarse grid and to prolongate the correction from the coarse grid back to the fine grid. We use a projection operator that computes the mean value of a square of size 2×2 to reduce our field by a factor of four; the interpolation operator is the transpose of the projection operator.

To assess the performance of the two-level multigrid method we limit the number of model runs to 100. We run multigrid GMRES with one, two and three cycles, and allocate the 100 model runs uniformly across cycles and levels.

For MG with one cycle we allocate the model runs as 33 model runs to the initial fine grid smoother (F), then 33 model runs to the coarse grid solver (C), and 34 model runs on the final fine grid smoothing. For two cycle we distribute these 100 model runs as $20F + 20C + 20F + 20C + 20F$. The same applies for three cycles, where we have 14 model runs on each grid. We are interested in a conclusive reduction in the residual (or error), especially after projecting the correction from the coarse grid to the fine grid.

Table 5 shows theMG solver results. The rows represent the different MG scenarios described above, plus a standard approach without MG, on the first line. The columns represent MG cycles. Each cycle is composed of two levels: fine and coarse. The MG algorithm starts on the fine grid by smoothing out the errors, then projects the residual of the intermediate solution on the coarse grid, and performs another smoothing of the errors. The result is projected back to the fine grid and used to correct the solution. This is called "Correction Scheme" as opposite to "Full Approximation Scheme" and is repeated for as many cycles as necessary. In each table entry we display the residual and error norms. For fine grid columns, the norms are computed on the fine grid, and correspond to the solution obtained after smoothing. For coarse grid columns, the displayed norms were still computed on the fine grid, after prolongating the correction from the coarse grid to the fine grid, and applying it to the solution. We show all the intermediate solutions in order to analyze the MG behavior for each cycle. The solution error norm decreases after projecting and applying the correction from the coarse grid to the fine grid after each stage. This was not trivial to accomplish, as it required crafting the prolongation operator as described above. The improvement is not reflected by the solution residual norm which sometimes shows an increase after prolongation, for example when using MG with one cycle. By comparing the final solution error norm obtained for different MG scenarios, it is inferred that better results are obtained with using fewer cycles, and more smoother iterations per cycle. This can be explained in terms of the Krylov solvers having more iterations available to build the Krylov space; the Krylov space information is lost when switching from one grid to another.

Cycle	1	1	2	2	3	3	Final
Level	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine
Residual							4.0.e-4
Error							1.9e-2
Residual	1.1e-2	3.1e-2					7.0.e-4
Error	7.7e-2	4.2e-2					2.6e-2
Residual	1.1e-2	1.4e-1	3.0e-3	8.1e-2			1.0e-3
Error	9.1e-2	6.4e-2	5.5e-2	4.4e-2			4.0e-2
Residual	2.5e-2	3.9e-1	1.1e-2	2.7e-1	8.0e-3	1.8e-2	6.0e-3
Error	1.1e-2	8.3e-2	6.7e-2	6.5e-2	5.3e-2	5.2e-2	4.4e-2

Table 5: Residual and error norms of solutions obtained at each multigrid stage (SWE).

MG provides the ability to run the model at a coarser resolution which in turn reduces computing time. This is very useful when dealing with large-scale models and their adjoints. The results reported in Table 5 are very good, even if they were produced using a basic MG algorithm. The performance of MG could be improved considerably by tuning the selection of coarse grids, building more accurate transfer operators, and testing additional matrix-free smoothers.

5. Numerical Tests with the Weather Research and Forecast Model

In this section we consider a realistic test case based on the Weather Research and Forecasting (WRF) model.

5.1. Numerical model

The WRF model [47] is a state-of-the-art numerical weather prediction system that can be used for both operational forecasting and atmospheric research. WRF is the result of a multiagency and university effort to build a highly parallelizable code that can run across scales ranging from large-eddy to global simulations. WRF accounts for multiple physical processes and includes cloud parameterization, landsurface models, atmosphere-ocean coupling, and broad radiation models. The terrain resolution can be as fine as 30 seconds of a degree.

The auxiliary software package WRFPLUS [48] provides the corresponding tangent-linear and first-order adjoint models. WRFPLUS is becoming a standard tool for applications such as data assimilation [49] and sensitivity analysis [50]. However, the adjoint model is work in progress and misses certain atmospheric processes. Because of this incompleteness, the computed sensitivities are only approximations of the full WRF gradients and Hessians. This will not affect the main conclusion of this study, namely that the proposed systematic approach to solving sensitivities to observations is feasible in the context of a real atmospheric model. Nevertheless, we expect that the sensitivity approximations have a negative impact on the convergence of the iterative solvers.

There is no second-order adjoint model developed for WRF to this point. This poses a challenge to our methodology, as it requires second-order derivatives. We consider several ways to approximate second-order information using the available tangent-linear or first-order adjoint models. First, we compute Hessian-vector products through finite differences of gradients obtained via first-order adjoint model. Unfortunately, our tests show that this approximation is marred by large errors and fails to produce useful results. Further investigation revealed that the adjoint model dampens the perturbations introduced in the system. The second approach is the Gauss-Newton approximation discussed in Section 4.3. The seed vector provides the initial condition to the tangent linear model, which propagates it to the final time. The result is mapped back to the initial time through the adjoint model. This is feasible for WRF since the required numerical tools are available. The Gauss-Newton approach introduces additional approximation errors in the second order sensitivity, beyond the incompleteness of the first order adjoint model.

WRF has the ability to perform forecasts on mesoscale domains defined and configured by the user. The simulation scenario selected covers a region across the East Coast of North America, centered on Virginia, and takes place over a time period of 6 hours starting on June 6th 2006 12:00 UTC. For simplicity, we assimilate only surface observations at the final time $t_0 + 6h$ obtained from NCEP. We start our simulations from reanalyzed fields, that is, simulated atmospheric states reconciled with observations (i.e., using data assimilation). In particular, we use the North American Regional Reanalysis (NARR) data set that covers the North American continent (160W-20W; 10N-80N) with a spatial resolution of 10 minutes of a degree, 29 pressure levels (1000-100 hPa, excluding the surface), a temporal resolution of three hours, and runs from 1979 until present.

The spatial discretization is a regular grid with 30 points on the East-West and North-South directions, and a horizontal resolution of 25 km. Since the atmosphere has different physical properties along with altitude, the vertical discretization involves 32 levels. A fixed time step of 30 seconds is used. The wall clock time for one time step of the forward (WRF) model is ~ 1.5 seconds. The wall clock time for one time step of the adjoint (WRFPLUS) model is ~ 4.5 seconds, about three times larger. For finer grid resolutions or for nested grids the computational effort can increase significantly; one needs the power of parallel architectures for computing sensitivities in an operational setting.

The experiment starts with minimizing the 4D-Var cost function until the norm of the gradient is reduced from $\sim 10^3$ to $\sim 10^{-3}$. The data assimilation procedure in WRFDA is an incremental approach revolving around the solution of a linear system as obtained with CG. The forecast error is obtained by comparing this reanalysis against a verification forecast represented by the corresponding NARR reanalysis. This forecast error was propagated backward in time through the adjoint model to obtain the right-hand side of the supersensitivity system (11). All results below use Hessian-vector products computed using the Gauss-Newton approximation.

5.2. Solution of the linear system

To solve the linear system associated with WRF we use the GMRES algorithm from the PETSc software library, since this algorithm can handle nonsymmetric and indefinite matrices. We select a subset of the preconditioners used with the SWE model. The first preconditioner (and the easiest to obtain) is the diagonal of the covariance matrix $\mathbf{B}_0$. The second preconditioner is the sum of elements in each row. The third preconditioner is a limited memory quasi-Newton approximation that uses information gathered throughout the data assimilation process. As shown in [51], the descent directions generated by the minimizer can be used to build the limited memory preconditioner through the L-BFGS formula. The fourth and last preconditioner used is the randomized SVD with 100 random vectors, computed in parallel at the equivalent total cost of just two model runs. The decrease in the norm of residual is presented in Figure 4(a) and in Table 6.

Preconditioner	Relative decrease in residual norm
None	7.2e-2
Background	7.6e-2
Row sum	4.5e-1
LMP	1.1e-1
Randomized SVD	2.2e-1

Table 6: Solution residual norm after 100 preconditioned iterations of GMRES for the (WRF) supersensitivity system (11). The scaling is done with respect to the initial guess residual norms.

As we can see from these results, the convergence of GMRES did not improve considerably through preconditioning. Moreover, while the unpreconditioned solver reduces the error of the residual monotonically, the preconditioned ones do not. The row sum preconditioner performs better than all the others in the first 15 iterations, then starts departing from the solution. A similar behavior can be observed for the preconditioner obtained from randomized SVD, which performs best between the 15-th and 30-th iterations. The diagonal of $\mathbf{B}_0$ preconditioner is the best for the next 50 iterations, except for a small interval where the LMP is slightly better. After 100 iterations the unpreconditioned residual is the smallest. In conclusion, it is really difficult to pinpoint one particular preconditioner as performing best for our WRF model. The fact that each solver leads to a residual that first decreases, then starts to increase requires further investigation. We think that this behavior is due to the large approximation errors made in computing first and second-order information. We are working with a 4D-Var reanalysis that is not optimal, and with adjoint models that are incomplete. Moreover, we employ Gauss-Newton approximation of the 4D-Var Hessian, and the ignored higher order terms may be non-negligible at the suboptimal solution. Other errors are associated with the way WRF deals with boundary conditions. Our methodology is affected by all these factors and the problem cannot be solved to a high degree of accuracy without improving the quality of each of these elements.

6. Visual Analysis of Sensitivity Results

In this section we illustrate the sensitivity analysis results. Consider the SWE data assimilation test case described in Section 4.2, except two of the observations are faulty. The sensitivity analysis results should reflect this inconsistency in observations.

Our approach is to modify the value of observations corresponding to h, u, v at two locations, before starting the assimilation process. This is done only for the final time of the assimilation window. The modified observations are located on the North-South median line, at coordinates 10x20 and 30x20 on the 40x40 grid, as shown in Figure 5. The two locations were chosen to be isolated

from each other so that the associated sensitivities will have a smaller chance of totally overlapping. Due to the symmetry of the locations, it is expected the results will be easier to study intuitively.

The fields of supersensitivities corresponding to h, u and v are plotted in Figures 5(a), 5(b), 5(c). The sensitivities have nonzero values and a pulse-like structure centered at the grid points containing the faulty observations. This indicates that the forecast error is most sensitive to the data assimilation parameters defined on these areas, such as the faulty observations. Although we modified the value of observations at two individual sites, the sensitivities are shaped as a pulse because the correlation between model variables spreads the errors spatially. When passing the supersensitivity through the TLM model to obtain the sensitivity to parameters defined at future times, the shape and location of the sensitivity is preserved (not shown here). This confirms the theory of 4D-Var that the information (or errors) in the observations are also spread in time.

7. Conclusions

In data assimilation the sensitivity of a forecast aspect to observations provides a quantitative metric of the impact each data point has on reducing forecast uncertainty. This metric can be used in hindsight to prune redundant data, to identify faulty measurements, and to improve the parameters of the data assimilation system. The metric can also be used in foresight to adaptively configure, and deploy, sensor networks for future measurements.

This work provides a systematic study of computational strategies to obtain sensitivities to observations in the context of 4D-Var data assimilation. Solution efficiency is of paramount importance since the models of interest in practice are large scale, and the computational cost of sensitivities is considerable; moreover, in an operational setting, the sensitivities have to be solved in faster-than-real-time (e.g., for dynamically deploying new sensors).

The cost of computing sensitivities to observations is dominated by the solution of a large-scale linear system, whose matrix is the Hessian of the 4D-Var cost function. In practice, this matrix is available only in operator form (i.e., matrix-vector products obtained via second order adjoint models).

The main contributions of this paper are to formulate the computational challenges associated with sensitivities to observations, and to present solutions to address them. We consider a set of matrix-free linear solvers, build specific preconditioners, and compare their performance on two numerical models. For the SWE test, the results are very promising: certain preconditioners as well as the multigrid approach lead to significant efficiency improvements in the solution of the linear system. The results for the WRF test are less clear cut: preconditioning brings only a modest improvement, and we attribute this to the limited accuracy with which derivatives are computed by the (currently incomplete) WRF adjoint model. Future work with WRF should focus both on finding better preconditioners, and on developing a more accurate adjoint model.

Acknowledgments.

This work was supported by National Science Foundation through the awards NSF DMS-0915047, NSF CCF-0635194, NSF CCF-0916493 and NSF OCI-0904397; and by AFOSR through the award FA9550–12–1–0293–DEF.

References

[1] T. Palmer, R. Gelaro, J. Barkmeijer, R. Buizza, Singular vectors, metrics, and adaptive observations, J. Atmos. Sci. 55 (4) (1998) 633–653.

[2] W. Liao, A. Sandu, G. Carmichael, et al, Total energy singular vector analysis for atmospheric chemical transport models., Mon. Weather Rev. 134 (9) (2006) 2443–2465.

[3] C. H. Bishop, Z. Toth, Ensemble transformation and adaptive observations, J. Atmos. Sci. 56 (11) (1999) 1748–1765.

[4] C. H. Bishop, B. J. Etherton, S. J. Majumdar, Adaptive sampling with the ensemble transform Kalman filter, Mon. Weather Rev. 129 (3) (2001) 420–436.

[5] T. Bergot, A. Doerenbecher, A study on the optimization of the deployment of targeted observations using adjoint-based methods, Q. J. R. Meteorol. Soc. 128 (583) (2002) 1689–1712.

[6] N. Fourrie, A. Doerenbecher, T. Bergot, A. Joly, Adjoint sensitivity of the forecast to TOVS observations, Q. J. R. Meteorol. Soc. 128 (586) (2002) 2759–2777.

[7] A. Joly, D. Jorgensen, The fronts and Atlantic storm-track experiment (FASTEX): Scientific objectives and experimental design, Bull. Am. Meteorol. Soc. 78 (9) (1997) 1917–1941.

[8] R. H. Langland, Z. Toth, R. Gelaro, et al, The North Pacific Experiment (NORPEX-98): Targeted observations for improved North American Weather Forecasts, Bull. Am. Meteorol. Soc. 80 (7) (1998) 1363–1384.

[9] N. Fourrie, D. Marchal, F. Rabier, et al, Impact study of the 2003 North Atlantic THORPEX regional campaign, Q. J. R. Meteorol. Soc. 132 (615) (2006) 275–295.

[10] J. Liu, E. Kalnay, Estimating observation impact without adjoint model in an ensemble Kalman filter, Q. J. R. Meteorol. Soc. 134 (634) (2008) 1327–1335.

[11] N. L. Baker, R. Daley, Observation and background adjoint sensitivity in the adaptive observation-targeting problem, Q. J. R. Meteorol. Soc. 126 (565) (2000) 1431–1454.

[12] R. H. Langland, N. L. Baker, Estimation of observation impact using the NRL atmospheric variational data assimilation adjoint system, Tellus A 56 (3) (2004) 189–201.

[13] D. N. Daescu, I. M. Navon, Adaptive observations in the context of 4D-Var data assimilation, Met. and Atm. Phys. 85 (4) (2004) 205–226.

[14] Y. Tremolet, Computation of observation sensitivity and observation impact in incremental variational data assimilation, Tellus A 60 (5) (2008) 964–978.

[15] L. M. Berliner, Z. Q. Lu, C. Snyder, Statistical design for adaptive weather observations, J. Atmos. Sci. 56 (15) (1999) 2536–2552.

[16] D. Zupanski, A. Y. Hou, S. Q. Zhang, M. Zupanski, C. D. Kummerow, S. H. Cheung, Applications of information theory in ensemble data assimilation, Q. J. R. Meteorol. Soc. 133 (627) (2007) 1533–1545.

[17] K. Singh, A. Sandu, M. Jardak, et al, Information theoretic metrics to characterize observations in variational data assimilation, Proc. Int. Conf. on Comp. Sci. 9 (1) (2012) 1047–1055.

[18] F. X. LeDimet, H. E. Ngodock, et al, Sensitivity analysis in variational data assimilation, J. Meteor. Soc. Japan 75 (1) (1997) 245–255.

[19] D. N. Daescu, On the sensitivity equations of four-dimensional variational (4D-Var) data assimilation, Mon. Wea. Rev. 136 (8) (2008) 3050–3065.

[20] D. N. Daescu, R. Todling, Adjoint sensitivity of the model forecast to data assimilation system error covariance parameters, Q. J. R. Meteorol. Soc. 136 (653) (2010) 2000–2012.

[21] R. Daley, Atmospheric data analysis, Cambridge University Press, Cambridge, 1991.

[22] E. Kalnay, Atmospheric modeling, data assimilation and predictability, Cambridge University Press, Cambridge, 2002.

[23] D. G. Cacuci, Sensitivity theory for nonlinear systems. i. Nonlinear functional analysis approach, J. of Math. Phys. 22 (12) (1981) 2794–2802.

[24] A. Sandu, D. N. Daescu, G. R. Carmichael, T. Chai, Adjoint sensitivity analysis of regional air quality models, J. Comput. Phys. 204 (1) (2005) 222–252.

[25] A. Cioaca, A. Sandu, M. Alexe, Second-order adjoints for solving PDE-constrained optimization problems, Optim. Meth. Softw. 27 (4-5) (2011) 625–653.

[26] I. Gejadze, On analysis error covariances in variational data assimilation, SIAM J. Sci. Comput. 4 (30) (2008) 1847–1874.

[27] I. Gejadze, F. X. LeDimet, V. Shutyaev, On error covariances in variational data assimilation, Russ. J. Numer. Anal. Math. Model. 22 (2) (2008) 163–175.

[28] R. Liska, B. Wendroff, Composite schemes for conservation laws, SIAM J. Numer. Anal. 35 (6) (1998) 2250–2271.

[29] Z. Wang, I. M. Navon, F. X. LeDimet, X. Zou, The second order adjoint analysis: theory and applications, Met. and Atm. Phys. 50 (1-3) (1992) 3–20.

[30] A. Sandu, L. Zhang, Discrete second order adjoints in atmospheric chemical transport modeling, J. Comput. Phys. 227 (12) (2008) 5949–5983.

[31] A. Griewank, On automatic differentiation, Mathematical Programming: recent developments and applications 6 (1989) 83–107.

[32] R. Giering, Tangent linear and adjoint model compiler, User Manual, TAMC Version 4.

[33] R. Giering, T. Kaminski, Recipes for adjoint code construction, ACM Trans. Math. Software 24 (4) (1998) 437–474.

[34] C. Zhu, R. H. Byrd, P. Lu, J. Nocedal, Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization, ACM Transactions on Mathematical Software (TOMS) 23 (4) (1997) 550–560.

[35] P. Courtier, J. N. Thepaut, A. Hollingsworth, A strategy for operational implementation of 4D-Var using an incremental approach, Q.J.R. Meteorol. Soc. 120 (519) (1994) 1367–1387.

[36] Y. Tremolet, Incremental 4D-Var convergence study, Tellus A 59 (5) (2007) 706–718.

[37] M. Ng, Iterative methods for Toeplitz systems, Oxford University Press, 2004.

[38] H. A. Vorst, Iterative Krylov methods for large linear systems, Cambridge University Press, Cambridge, 2003.

[39] Y. Saad, Iterative methods for sparse linear systems, PWS Pub. Co., Boston, 1996.

[40] S. Balay, J. Brown, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, H. Zhang, Petsc web page, 2012, URL: http://www.mcs.anl.gov/petsc.

[41] G. E. Forsythe, E. G. Strauss, On best conditioned matrices, Proc. Amer. Math. Soc. 6 (3) (1955) 340–345.

[42] M. Zupanski, A preconditioning algorithm for large-scale minimization problems, Tellus A 45 (5) (1993) 478–492.

[43] W. Yang, I. M. Navon, F. Courtier, A new Hessian preconditioning method applied to variational data assimilation experiments using NASA general circulation models, Mon. Wea. Rev. 124 (5) (1996) 1000–1017.

[44] P. G. Martinsson, E. Liberty, F. Woolfe, V. Rokhlin, M. Tygert, Randomized algorithms for the low-rank approximation of matrices, Proc. Natl. Acad. Sci. USA 104 (51) (2007) 20167–20172.

[45] R. P. Fedorenko, A relaxation scheme for the solution of elliptic differential equations, UdSSR Comput. Math. Phys. 1 (5) (1961) 1092–1096.

[46] W. Hackbusch, Multigrid methods and applications, Springer, Berlin, 2003.

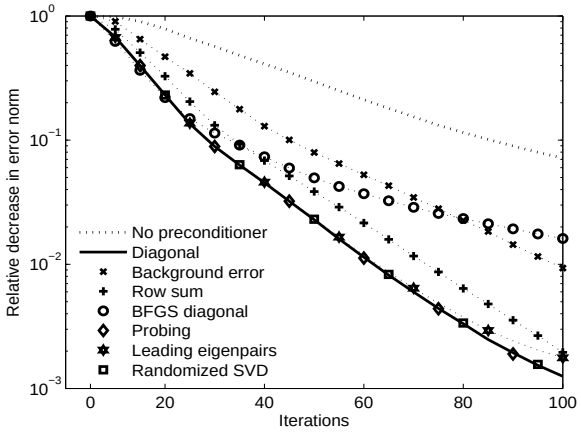
[47] W. C. Skamarock, J. B. Klemp, J. Dudhia, D. D. Gill, et al, Coauthors, 2008: A description of the advanced research wrf version 3, Tech. rep. (2008).

[48] Q. Xiao, Z. Ma, W. Huang, X. Y. Huang, D. Barker, Y. H. Kuo, J. J. Michalakes, Development of the WRF tangent linear and adjoint models: Nonlinear and linear evolution of initial perturbations and adjoint sensitivity analysis at high southern latitudes, in: WRF/MM5 users workshop, Boulder, Colorado, 2005, pp. 27–29.

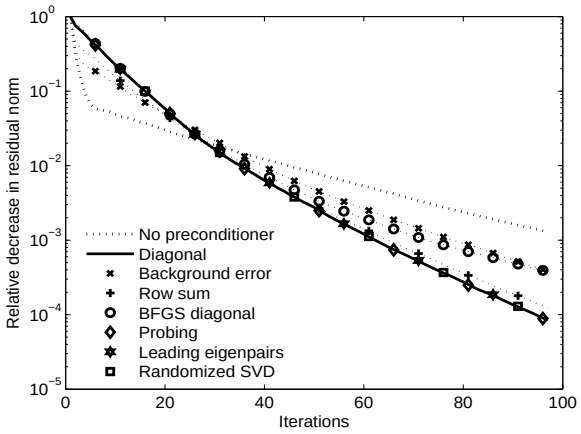
[49] C. S. Schwartz, Z. Liu, Y. Chen, X. Y. Huang, Impact of assimilating microwave radiances with a limited-area ensemble data assimilation system on forecasts of Typhoon Morakot, Weather and Forecasting 27 (2) (2012) 424–437.

[50] A. Cioaca, V. Zavala, E. Constantinescu, Adjoint sensitivity analysis for numerical weather prediction: applications to power grid optimization, in: Proceedings of the first international workshop on high-performance computing, networking and analytics for the power grid, HiPCNA-PG '11, ACM, New York, NY, USA, 2011, pp. 35–42. doi:10.1145/2096123.2096133.
URL http://doi.acm.org/10.1145/2096123.2096133

[51] J. Tshimanga, S. Gratton, A. T. Weaver, A. Sartenaer, Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation, Q. J. R. Meteorol. Soc. 134 (632) (2008) 751–769.



(a) Error norm



(b) Residual norm

Figure 3: Convergence of non-preconditioned iterative solvers for the (SWE) supersensitivity system (11).

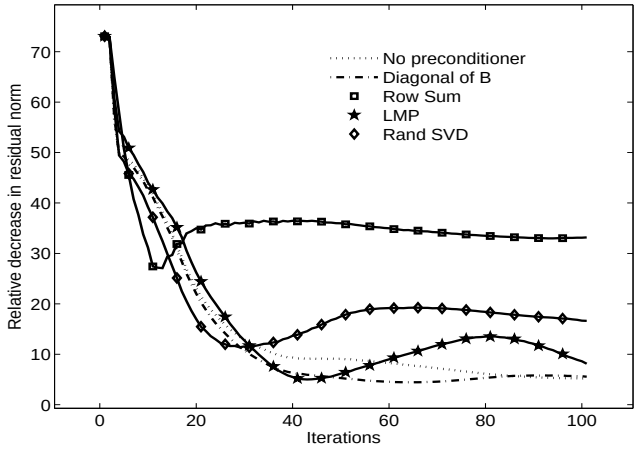
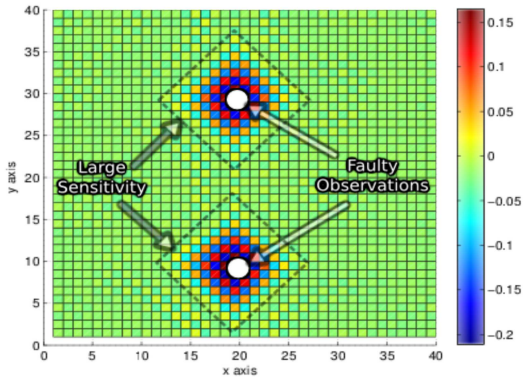
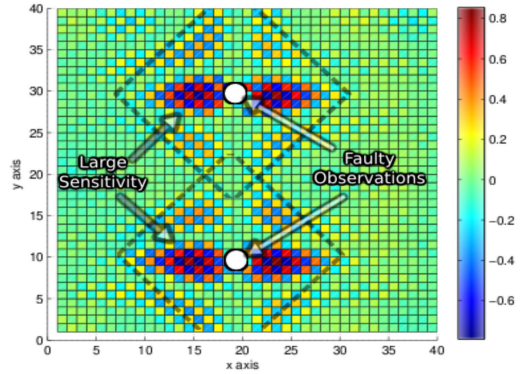


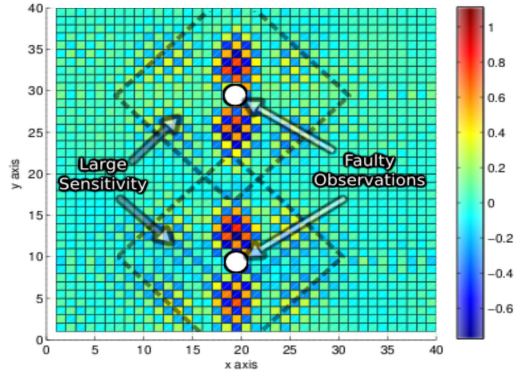
Figure 4: Convergence of preconditioned iterative solvers for the (WRF) supersensitivity system (11).



(a) Observations for h



(b) Observations for u



(c) Observations for v

Figure 5: Fields of forecast sensitivities to observations, represented on the computational grid.