Computer Science Technical Report
TR-07-27
July 11, 2007

Adrian Sandu and Lin Zhang

"Discrete Second Order Adjoints in Atmospheric Chemical Transport Modeling"

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://www.eprints.cs.vt.edu

DISCRETE SECOND ORDER ADJOINTS IN ATMOSPHERIC CHEMICAL TRANSPORT MODELING

ADRIAN SANDU* AND LIN ZHANG*

Abstract. Atmospheric chemical transport models (CTMs) are essential tools for the study of air pollution, for environmental policy decisions, for the interpretation of observational data, and for producing air quality forecasts. Many air quality studies require sensitivity analyses, i.e., the computation of derivatives of the model output with respect to model parameters. The derivatives of a cost functional (defined on the model output) with respect to a large number of model parameters can be calculated efficiently through adjoint sensitivity analysis. While the traditional (first order) adjoint models give the gradient of the cost functional with respect to parameters, second order adjoint models give second derivative information in the form of products between the Hessian of the cost functional and a user defined vector.

In this paper we discuss the mathematical foundations of the discrete second order adjoint sensitivity method and present a complete set of computational tools for performing second order sensitivity studies in three-dimensional atmospheric CTMs. The tools include discrete second order adjoints of Runge Kutta and of Rosenbrock time stepping methods for stiff equations together with efficient implementation strategies. Numerical examples illustrate the use of these computational tools in important applications like sensitivity analysis, optimization, uncertainty quantification, and the calculation of directions of maximal error growth in three-dimensional atmospheric CTMs.

Key words. Second Order Adjoints, Chemical Transport Models, Stiff Equations, Sensitivity Analysis, Optimization, Hessian Singular Vectors

AMS subject classifications. 15A15, 15A09, 15A23

1. Introduction.

The chemical composition of the atmosphere is being significantly perturbed by anthropogenic emissions of trace gases and aerosols. This has important implications for urban, regional and global air quality, for human health, and for climate change. Atmospheric chemical transport models (CTMs) are essential tools for the study of air pollution, for environmental policy decisions, for the interpretation of observational data, and for producing air quality forecasts.

Many air quality studies require the computation of derivatives of the model output with respect to model parameters. Two important applications that require derivatives are sensitivity analysis and data assimilation. Sensitivity analyses are useful to identify those parameters that have the largest influence on the results of the simulation. Air quality predictions have large uncertainties associated with incomplete information on emissions, initial and boundary conditions, and with incomplete science elements; improvements in the predictive capabilities of CTMs require them to be better constrained by observational data through data assimilation. In a varia-

*Department of Computer Science, Virginia Polytechnic Institute and State University, 244 Knowledgeworks 2, 2202 Kraft Drive, Blacksburg, VA 24061. E-mail: { sandu@cs.vt.edu, lin83@vt.edu }. Phone: 540-231-2193. Fax: 540-231-9218.

tional approach the data assimilation problem is posed as a large-scale minimization problem, whose numerical solution requires the gradient of the cost function with respect to model parameters.

The derivatives of a cost functional (defined on the model output) with respect to a large number of model parameters can be calculated efficiently through adjoint sensitivity analysis. While the traditional (first order) adjoint models give the gradient of the cost functional (first order derivatives with respect to parameters), second order adjoint models give second derivative information in the form of products between the Hessian of the cost functional and a user defined vector.

Second order adjoints have been discussed in the literature in several contexts. Wang et al. [72] have developed the theory of second order adjoints and its applications in numerical weather prediction. Second order adjoints have been used in data assimilation within the numerical optimization algorithms by Wang et al., [73], Le Dimet et al., [34, 38], and Ozyurt et al., [8]. Daescu and Navon [21] have used second order adjoints of reduced order models to perform the optimization for data assimilation in a low-dimensional control space. Raffard and Tomlin [61] have applied second order adjoints for constrained PDE optimization programs in the context of air traffic flow. Charpentier et al. [19] have used second order adjoints for the optimal control of integral equations. Hessian vector products have been used in the calculation of Hessian singular vectors in the context of data assimilation ([6, 73, 62]). The use of Hessian information for uncertainty quantification have been explored by Le Dimet et al. [38] and by Alekseev et al. [1]. Alekseev and Navon [2] have used the Hessian spectrum to precondition the solution of ill-posed inverse problems.

Ozyurt and Barton [8, 9] have discussed the evaluation of second order adjoints for embedded functionals of stiff systems. Their approach is to derive the second order adjoint ODE and then solve it efficiently together with the forward, the tangent linear, and the first order adjoint models using backward differentiation formulas (DASSL). Different Jacobian matrices appearing in the definition of the second order adjoint ODE are derived from the original ODE using automatic differentiation. The LU factorizations of the forward model solution are reused in the tangent linear and the first and second order adjoint solutions, which leads to a computationally efficient process. The current paper discusses the calculation of second order adjoints for stiff ODEs arising in chemical kinetic models. This work is different from [8, 9] as we focus on discrete second order adjoints and we use one step methods of Runge-Kutta and Rosenbrock types. Moreover, we propose a transformation that eliminates the need of costly quadrature evaluations in [8, 9].

Most applications to date have focused on continuous second order adjoints (obtained by linearizing the underlying ordinary or partial differential equation models) [8, 34, 72]. Discrete second order adjoints (obtained by linearizing the numerical approximations of the model) have been obtained by automatic differentiation [19, 40, 7].

Other applications of automatic differentiation to obtain high order derivatives are discussed in [46, 10, 33, 36].

In this paper we discuss the mathematical foundations of the discrete second order adjoint sensitivity method and present a complete set of computational tools for performing second order sensitivity studies in three-dimensional atmospheric CTMs. The tools include discrete second order adjoints of Runge Kutta and of Rosenbrock time stepping methods for stiff equations together with efficient implementation strategies. Numerical examples show how second order adjoints can extend the range of validity of sensitivity analyses for nonlinear chemical kinetic models. The use of second order adjoint information in the optimization process for chemical data assimilation is exemplified on a simulation of real atmospheric conditions with real observations. We also illustrate second order adjoint based methodologies for the quantification of uncertainty in the chemical fields after data assimilation, and for the computation of the most important directions of error growth in a three-dimensional atmospheric CTM.

The main contributions of this work are the derivation of efficient discrete second order adjoint schemes for Runge-Kutta and Rosenbrock stiff solvers, the construction of second order adjoints for a three-dimensional chemical transport model, and the illustration of how these computational tools are essential in important applications like sensitivity analysis, optimization, uncertainty quantification, and the calculation of directions of maximal error growth in three-dimensional atmospheric CTMs.

The paper is organized as follows. Section 2 introduces the atmospheric chemistry and transport models, their solution by the operator splitting approach, and discusses the tangent linear and discrete adjoint models. Section 3 develops the mathematical framework for second order adjoint sensitivity analysis. Discrete second order adjoints for stiff ODEs are developed based on Runge Kutta methods (Section 3.3) and on Rosenbrock methods (Section 3.4). Section 4 discusses the construction of the discrete second order adjoint for the transport system. Section 5 illustrates important applications of second order adjoints, and Section 6 draws the conclusions of this work.

2. Atmospheric Chemistry and Transport Modeling. Consider a domain Ω which covers a region of the atmosphere. Let $\vec{n}$ be the outward normal vector on each point of the boundary $\partial\Omega$. At each time moment the boundary of the domain is partitioned into $\partial\Omega = \Gamma^{\mathrm{IN}} \cup \Gamma^{\mathrm{OUT}} \cup \Gamma^{\mathrm{GR}}$ where Γ^{GR} is the ground level portion of the boundary; Γ^{IN} is the set of (lateral or top) boundary points where $u \cdot \vec{n} \leq 0$ and Γ^{OUT} the set where $u \cdot \vec{n} > 0$. In the following u is the wind field vector, K the turbulent diffusivity tensor, ρ the air density in $moles/cm^3$, and c_i the mole-fraction concentration of chemical species i $(1 \leq i \leq n_s)$. The density of this species is $\rho\ c_i$ $(moles/cm^3)$. Let V_i^{dep} be the deposition velocity of species i, Q_i the rate

of surface emissions, and E_i the rate of elevated emissions for this species. The rate of chemical transformations f_i depends on absolute concentration values; the rate at which mole-fraction concentrations change is then $f_i(\rho c)/\rho$.

The evolution of concentrations in time is described by the material balance equations

$$\frac{\partial c_i}{\partial t} = -u \cdot \nabla c_i + \frac{1}{\rho}\nabla \cdot (\rho K \nabla c_i) + \frac{1}{\rho} f_i(\rho c) + E_i \ , \quad t^0 \le t \le t^{\mathrm{F}} \tag{2.1a}$$

$$c_i(t^0, x) = c_i^0(x) \ , \tag{2.1b}$$

$$c_i(t, x) = c_i^{\mathrm{IN}}(t, x) \quad \text{for} \quad x \in \Gamma^{\mathrm{IN}} \ , \tag{2.1c}$$

$$K\frac{\partial c_i}{\partial n} = 0 \quad \text{for} \quad x \in \Gamma^{\mathrm{OUT}} \ , \tag{2.1d}$$

$$K\frac{\partial c_i}{\partial n} = V_i^{\mathrm{dep}} c_i - Q_i \quad \text{for} \quad x \in \Gamma^{\mathrm{GR}} \ , \quad \text{for all} \ \ 1 \le i \le n_s \ . \tag{2.1e}$$

We refer to the system (2.1a)–(2.1e) as the *forward model*. To simplify the presentation, in this paper we consider as parameters the initial state c^0 of the model; it is known that this does not restrict the generality of the formulation. The solution of the forward model $c = c(t, c^0)$ is uniquely determined once the model parameters c^0 are specified.

The forward model (2.1a)–(2.1e) is solved by a sequence of N timesteps of length Δt taken between t^0 and $t^N = t^{\mathrm{F}}$. At each time step one calculates the numerical approximation $c^k(x) \approx c(t^k, x)$ at $t^k = t^0 + k\Delta t$ such that

$$c^{k+1} = \mathcal{N}_{[t^k, t^{k+1}]} \circ c^k \ , \quad c^N = \prod_{k=0}^{N-1} \mathcal{N}_{[t^k, t^{k+1}]} \circ c^0 \ . \tag{2.2}$$

The numerical solution operator $\mathcal{N}$ is based on an operator splitting approach, where the transport steps along each direction and the chemistry steps are taken successively. Operator splitting is standard practice in computational air pollution modeling [48]. It allows the development of the forward, tangent linear, and adjoint models with relative ease. Formally, if we denote by $\mathcal{T}$ the numerical solution operator for directional transport, and by $\mathcal{C}$ the solution operator for chemistry we have

$$\mathcal{N}_{[t,t+\Delta t]} = \mathcal{T}_X^{\Delta t/2} \circ \mathcal{T}_Y^{\Delta t/2} \circ \mathcal{T}_Z^{\Delta t/2} \circ \mathcal{C}^{\Delta t} \circ \mathcal{T}_Z^{\Delta t/2} \circ \mathcal{T}_Y^{\Delta t/2} \circ \mathcal{T}_X^{\Delta t/2} \ . \tag{2.3}$$

The numerical errors introduced by splitting are an important component of model errors (see e.g., [69]). In this paper, for the purpose of 4D-Var data assimilation, we assume the model errors to be small. Indeed, in computational air pollution modeling the splitting errors oscillate with the diurnal cycle and do not grow unboundedly for evolving time [48].

A perturbation δc^0 in the parameters c^0 propagates in time according to the tangent linear discrete model

$$\begin{aligned} \delta c^{k+1} &= \mathcal{N}'_{[t^k, t^{k+1}]} \circ \delta c^k \\ &= \mathcal{T}'^{\Delta t/2}_X \circ \mathcal{T}'^{\Delta t/2}_Y \circ \mathcal{T}'^{\Delta t/2}_Z \circ \mathcal{C}'^{\Delta t} \circ \mathcal{T}'^{\Delta t/2}_Z \circ \mathcal{T}'^{\Delta t/2}_Y \circ \mathcal{T}'^{\Delta t/2}_X \circ \delta c^k \end{aligned} \tag{2.4}$$

where $\mathcal{N}'$ is the tangent linear operator associated with the solution operator $\mathcal{N}$. For an operator splitting approach (2.3) $\mathcal{N}'$ is built from the tangent linear transport and chemistry operators $\mathcal{T}'$ and $\mathcal{C}'$.

To each tangent linear operator corresponds an adjoint operator (denoted here with a star superscript). The *(discrete) adjoint model* is

$$\begin{aligned} \lambda^k &= \mathcal{N}'^*_{[t^{k+1},t^k]} \circ \lambda^{k+1} + \phi^k \\ &= \mathcal{T}'^*_X{}^{\Delta t/2} \circ \mathcal{T}'^*_Y{}^{\Delta t/2} \circ \mathcal{T}'^*_Z{}^{\Delta t/2} \circ \mathcal{C}'^{*\Delta t} \circ \mathcal{T}'^*_Z{}^{\Delta t/2} \circ \mathcal{T}'^*_Y{}^{\Delta t/2} \circ \mathcal{T}'^*_X{}^{\Delta t/2} \circ \lambda^{k+1} + \phi^k \end{aligned} \tag{2.5}$$

The forcing function ϕ and the initial values λ^N are chosen such that the adjoint variables are sensitivities of a given cost functional with respect to the state variables.

The numerical experiments in this paper use the state-of-the-art regional atmospheric photochemistry and transport model STEM (Sulfur Transport Eulerian Model) [14] to solve the mass-balance equations for concentrations of trace species (2.2) in order to determine the fate of pollutants in the atmosphere. STEM has first order adjoint capabilities [64] and has been used extensively in real life chemical data assimilation studies [17, 45, 12, 18]. In this paper we endow STEM with the capability to compute second order adjoints and we illustrate several applications of this capability.

3. Second Order Adjoints for Stiff ODEs. In this section we review the derivation of continuous and discrete second order adjoint equations (SOA) in the context of ordinary differential equations (ODEs).

3.1. Continuous Second Order Adjoints. Consider a general (stiff) ODE

$$c' = R(t, c, p) \ , \quad c(t^0) = c^0(p) \ , \quad t^0 \le t \le t^{\mathrm{F}} \ .$$

For our application the vector $y(t) \in \mathbb{R}^{n_s}$ represents the time evolving concentrations of the chemical species starting from the initial configuration c^0. $p \in \mathbb{R}^{n_p}$ is a vector of model parameters. The rate of change in the concentrations c is determined by the nonlinear production/loss function $R = [R_1, \ldots, R_{n_s}]^T$.

Consider a cost functional

$$\Psi = \int_{t^0}^{t^{\mathrm{F}}} h\big(c(t), p\big)\, dt$$

defined on the time evolving concentrations. We want to efficiently obtain the first and second order derivatives of the cost function with respect to model parameters,

$$\frac{\partial \Psi}{\partial p_i} \ , \quad \frac{\partial^2 \Psi}{\partial p_i \partial p_j} \ , \quad 1 \le i, j \le n_p \ .$$

Note that the parameters can be transformed into variables by appending additional formal evolution equations for parameters $p' = 0$. This allows to always reduce

the sensitivity of the cost functional with respect to parameters to the sensitivity of the cost functional with respect to initial conditions. Moreover, the general cost functional defined as an integral of a function of the state along the trajectory can be reformulated as a cost functional defined on the state at the final time by appending an additional variable θ and an equation that performs the time integration. The equivalent system becomes:

$$\begin{bmatrix} c \\ p \\ \theta \end{bmatrix}' = \begin{bmatrix} R(t,c,p) \\ 0 \\ h(c,p) \end{bmatrix}, \quad \begin{bmatrix} c(t^0) \\ p(t^0) \\ \theta(t^0) \end{bmatrix} = \begin{bmatrix} c^0(p) \\ p \\ 0 \end{bmatrix}, \quad \Psi = \theta(t^{\mathrm{F}}) .$$

Without loss of generality the mathematical formulation of the stiff nonlinear differential equations which constitute the *forward model* is

$$\frac{dy}{dt} = f(t,y) , \qquad y\left(t^0\right) = y^0 , \qquad t^0 \leq t \leq t^{\mathrm{F}} . \tag{3.1}$$

The solution is $y(t) = [c^T, p^T, \theta]^T \in \mathbb{R}^n$, $n = n_s + n_p + 1$, and the model parameters are the initial conditions y^0. Again without loss of generality the cost functional is defined as a function of the state at the final time

$$\Psi\Big(y^0\Big) = g\Big(y\left(t^{\mathrm{F}}\right)\Big) . \tag{3.2}$$

In this paper we assume that the functions f and g are at least twice continuously differentiable.

We are interested to efficiently evaluate the first and second order sensitivities of the cost functional (3.2) with respect to changes in initial conditions

$$\frac{\partial \Psi}{\partial y_i^0} , \quad \text{and} \quad \frac{\partial^2 \Psi}{\partial y_i^0 \partial y_j^0} , \quad 1 \leq i,j \leq n .$$

Throughout this paper vectors will be represented in column format and an upper script $(\cdot)^T$ will denote the transposition operator. The gradient of a scalar function (e.g., $\partial\Psi/\partial y^0$) is a row vector. We denote the Jacobian of the time derivative function in (3.1) by

$$J_{i,j}(t,y) = \frac{\partial f_i\left(t,y\right)}{\partial y_j} , \qquad 1 \leq i,j \leq n . \tag{3.3}$$

The Hessian of the time derivative function in (3.1) is a 3-tensor of second order derivatives

$$H_{i,j,k}\left(t,y\right) = \frac{\partial J_{i,j}\left(t,y\right)}{\partial y_k} = \frac{\partial^2 f_i\left(t,y\right)}{\partial y_j\, \partial y_k} = \frac{\partial^2 f_i\left(t,y\right)}{\partial y_k\, \partial y_j} = H_{i,k,j}\left(t,y\right) , \quad 1 \leq i,j,k \leq n . \tag{3.4}$$

The Hessian allows to express the derivatives of the Jacobian times a user vector. As shown in Appendix A for any vectors u and v we have that

$$\frac{\partial}{\partial y}\left[J(t,y)\cdot u\right]\cdot v = \Big(H(t,y)\cdot u\Big)\cdot v = \Big(H(t,y)\cdot v\Big)\cdot u \ ,$$
$$\frac{\partial}{\partial y}\left[J^T(t,y)\cdot u\right]\cdot v = \Big(u^T\cdot H(t,y)\Big)\cdot v = \Big(H(t,y)\cdot v\Big)^T\cdot u \ ,$$

where the dot operator $(\cdot)$ denotes the regular tensor-vector product.

Small perturbations of the solution (due to infinitesimally small changes δy^0 in the initial conditions)

$$\delta y(t) = \frac{\partial y(t)}{\partial y^0}\cdot \delta y^0 \tag{3.5}$$

propagate forward in time according to the *tangent linear model*

$$\frac{d\delta y}{dt} = J(t,y)\cdot \delta y \ , \qquad \delta y\left(t^0\right) = \delta y^0 \ , \qquad t^0 \le t \le t^{\mathrm{F}} \ . \tag{3.6}$$

The change in the cost functional (3.2) due to the small change δy^0 in the initial conditions is

$$\delta\Psi = \frac{\partial g}{\partial y}\left(t^{\mathrm{F}}\right)\cdot \delta y\left(t^{\mathrm{F}}\right) = \frac{\partial \Psi}{\partial y^0}\cdot \delta y^0 \ .$$

In the forward sensitivity analysis each integration of the tangent linear model (3.6) allows to compute the dot product of the gradient $\partial\Psi/\partial y^0$ with the vector of initial perturbations δy^0. The gradient is recovered after n tangent linear model (3.6) integrations initialized with linearly independent perturbation vectors.

A more efficient way of calculating the gradient $\partial\Psi/\partial y^0$ is provided by the *first order adjoint model* [22, 63, 64]

$$\frac{d\lambda}{dt} = -J^T(t,y)\cdot\lambda \ , \qquad \lambda\left(t^{\mathrm{F}}\right) = \frac{\partial g}{\partial y}\left(y(t^{\mathrm{F}})\right) \ , \qquad t^{\mathrm{F}} \ge t \ge t^0 \ . \tag{3.7}$$

The adjoint variables $\lambda(t)\in\mathbb{R}^n$ represent the sensitivities of the cost functional with respect to (changes in) the model solution

$$\lambda(t) = \left(\frac{\partial\Psi}{\partial y(t)}\right)^T \ ,$$

and in particular we have that the adjoint variable at the initial time is the transposed gradient of the cost functional with respect to initial conditions

$$\lambda(t^0) = \left(\frac{\partial\Psi}{\partial y^0}\right)^T \ .$$

We are now interested in obtaining the second order derivatives of the cost functional with respect to initial conditions. The Hessian of the cost functional is

$$\mathcal{H}_{i,j} = \frac{\partial^2\Psi}{\partial y_i^0\partial y_j^0} = \frac{\partial}{\partial y_j^0}\left(\frac{\partial\Psi}{\partial y_i^0}\right) = \frac{\partial\lambda_i(t^0)}{\partial y_j^0} \ , \qquad 1\le i,j\le n \ .$$

The Hessian has n^2 elements. In many problems (including our target application, atmospheric chemical transport problems) n is very large and computing the entire Hessian is not practical. We will therefore look to compute Hessian times vector products $\sigma = \mathcal{H} \cdot u$ for any user-defined vector u

$$\left(\mathcal{H} \cdot u\right)_i = \sum_{j=1}^{n} \mathcal{H}_{i,j}\ u_j = \sum_{j=1}^{n} \frac{\partial \lambda_i(t^0)}{\partial y_j^0}\ u_j = \frac{\partial \lambda_i(t^0)}{\partial y^0} \cdot u\ . \tag{3.8}$$

To compute such products we consider the variation of the cost functional (3.2) with respect to changes in initial conditions as a new cost functional that depends on both the initial state and on the initial perturbation

$$\delta\Psi\left(y^0, \delta y^0\right) = \frac{\partial \Psi}{\partial y^0} \cdot \delta y^0 = \lambda^T\left(t^0\right) \cdot \delta y^0 = \sum_{i=1}^{n} \frac{\partial \Psi}{\partial y_i^0}\ \delta y_i^0\ . \tag{3.9}$$

The gradient of $\delta\Psi$ with respect to changes in y^0 can be computed by the adjoint method. This gradient represents the product of the Hessian of Ψ times the initial perturbation vector,

$$\begin{aligned}
\left(\frac{\partial\, \delta\Psi}{\partial y^0}\right)_j &= \frac{\partial\, \delta\Psi}{\partial y_j^0} = \frac{\partial}{\partial y_j^0}\left(\sum_{i=1}^{n} \frac{\partial \Psi}{\partial y_i^0}\ \delta y_i^0\right) \\
&= \sum_{i=1}^{n} \frac{\partial^2 \Psi}{\partial y_i^0\ \partial y_j^0}\ \delta y_i^0 = \sum_{i=1}^{n} \frac{\partial^2 \Psi}{\partial y_j^0\ \partial y_i^0}\ \delta y_i^0 \\
&= \sum_{i=1}^{n} \mathcal{H}_{j,i}\ \delta y_i^0 \\
&= \left(\mathcal{H} \cdot \delta y^0\right)_j\ .
\end{aligned} \tag{3.10}$$

From (3.10) we see that the Hessian-vector products can be computed as the first order adjoint gradients of the cost functional $\delta\Psi$, if the tangent linear model is initialized with the user-defined vector $\delta y^0 = u$.

The cost functional (3.9) depends on both y and δy. Consequently, to evaluate $\delta\Psi$ one needs to consider both the forward (3.1) and the tangent linear model (3.6) evolving together forward in time:

$$\frac{d}{dt}\begin{bmatrix} y \\ \delta y \end{bmatrix} = \begin{bmatrix} f(t,y) \\ J(t,y) \cdot \delta y \end{bmatrix}\ , \qquad \begin{bmatrix} y \\ \delta y \end{bmatrix}\left(t^0\right) = \begin{bmatrix} y^0 \\ u \end{bmatrix}\ , \qquad t^0 \le t \le t^{\mathrm{F}}\ . \tag{3.11}$$

The Jacobian of the extended system (3.11) is

$$\begin{bmatrix} J(t,y) & 0 \\ \frac{\partial}{\partial y}\Big(J(t,y) \cdot \delta y\Big) & J(t,y) \end{bmatrix} = \begin{bmatrix} J(t,y) & 0 \\ H(t,y) \cdot \delta y & J(t,y) \end{bmatrix}$$

The adjoint of the (tangent linear model of the) extended system (3.11) for the cost

function (3.9) reads

$$\begin{aligned}
\frac{d}{dt}\begin{bmatrix}\sigma\\ \lambda\end{bmatrix} &= -\begin{bmatrix} J(t,y) & 0\\ H(t,y)\cdot\delta y & J(t,y)\end{bmatrix}^T\cdot\begin{bmatrix}\sigma\\ \lambda\end{bmatrix}\\
&= \begin{bmatrix} -J^T(t,y)\cdot\sigma - \Big(H(t,y)\cdot\delta y\Big)^T\cdot\lambda\\ -J^T(t,y)\cdot\lambda\end{bmatrix}\\
\begin{bmatrix}\sigma\\ \lambda\end{bmatrix}\left(t^{\mathrm{F}}\right) &= \begin{bmatrix}\frac{d^2g}{dy^2}\left(y(t^{\mathrm{F}})\right)\cdot\delta y\left(t^{\mathrm{F}}\right)\\ \frac{dg}{dy}\left(y(t^{\mathrm{F}})\right)\end{bmatrix}, \qquad t^{\mathrm{F}}\geq t\geq t^0\ .
\end{aligned} \tag{3.12}$$

The equation for λ in (3.12) is the first order adjoint equation (3.7).

The first equation in (3.12) is the *second order adjoint ordinary differential equation* and defines the time evolution of the *second order adjoint variable* σ,

$$\frac{d\sigma}{dt} = -J^T(t,y)\cdot\sigma - \Big(H(t,y)\cdot\delta y\Big)^T\cdot\lambda\ , \quad \sigma(t^{\mathrm{F}}) = \frac{d^2g}{dy^2}\left(y(t^{\mathrm{F}})\right)\cdot\delta y\left(t^{\mathrm{F}}\right)\ . \tag{3.13}$$

By the definition of first and second order adjoints we have that

$$\sigma(t) = \delta\lambda(t) = \frac{\partial\lambda(t)}{\partial y^0}\cdot\delta y^0 = \frac{\partial\lambda(t)}{\partial y(t)}\cdot\frac{\partial y(t)}{\partial y^0}\cdot\delta y^0 = \frac{\partial\lambda(t)}{\partial y(t)}\cdot\delta y(t)\ .$$

Consequently the second order adjoint equation (3.13) can also be obtained by formally taking the variation of the first order adjoint equation (3.7) with respect to changes δy^0 in the initial conditions y^0

$$\begin{aligned}
\frac{\partial}{\partial y^0}\left(\frac{d\lambda}{dt}\right)\cdot\delta y^0 &= \frac{\partial}{\partial y^0}\left(-J^T(t,y)\cdot\lambda\right)\cdot\delta y^0\\
\frac{d}{dt}\left(\frac{\partial\lambda}{\partial y^0}\cdot\delta y^0\right) &= \frac{\partial}{\partial y}\Big(-J^T(t,y)\cdot\lambda\Big)\cdot\frac{\partial y(t)}{\partial y^0}\cdot\delta y^0\\
\frac{d\sigma}{dt} &= \frac{\partial}{\partial y}\Big(-J^T(t,y)\cdot\lambda\Big)\cdot\delta y(t)\\
&= -J^T(t,y)\cdot\sigma - \Big(H(t,y)\cdot\delta y\Big)^T\cdot\lambda\\
\frac{\partial}{\partial y^0}\left(\frac{dg}{dy}\left(y(t^{\mathrm{F}})\right)\right)\cdot\delta y^0 &= \frac{dg}{dy^2}\left(y(t^{\mathrm{F}})\right)\cdot\frac{\partial y\left(t^{\mathrm{F}}\right)}{\partial y^0}\cdot\delta y^0\\
&= \frac{d^2g}{dy^2}\left(y(t^{\mathrm{F}})\right)\cdot\delta y\left(t^{\mathrm{F}}\right)\ .
\end{aligned}$$

3.2. Discrete Second Order Adjoints. Similar considerations hold for the discrete system

$$y^k = \mathcal{N}_k\left(y^{k-1}\right)\ , \quad k=1,\cdots,N\ , \quad y^0 \ \text{ given}\ , \tag{3.14}$$

and the discrete cost function of the form

$$\Psi\left(y^0\right) = g\left(y^N\right)\ . \tag{3.15}$$

The discrete system (3.14) represents a numerical discretization of (3.1) with a one-step numerical method. The cost function (3.15) is defined on the numerical solution, and approximates the continuous cost function (3.2) defined on the exact solution.

We denote the Jacobian matrix of the discrete time-marching operator by $\mathcal{N}_k'(y) = \partial \mathcal{N}_k / \partial y$, and the Hessian three-tensor by $\mathcal{N}_k''(y) = \partial^2 \mathcal{N}_k / \partial y^2$. The tangent linear model of (3.14) is

$$\delta y^k = \mathcal{N}_k' \left(y^{k-1} \right) \cdot \delta y^{k-1} \ , \quad k = 1, \cdots, N \ , \quad \delta y^0 = u \ . \tag{3.16}$$

The extended adjoint of the combined (3.14)–(3.16) for the cost function $\delta \Psi$ reads

$$\begin{aligned} \left[\begin{array}{c} \sigma^{k-1} \\ \lambda^{k-1} \end{array} \right] &= \left[\begin{array}{c} -\left(\mathcal{N}_k'(y^{k-1}) \right)^T \cdot \sigma^k - \left(\mathcal{N}_k''(y^{k-1}) \cdot \delta y^{k-1} \right)^T \cdot \lambda^k \\ -\left(\mathcal{N}_k'(y^{k-1}) \right)^T \cdot \lambda^k \end{array} \right] \ , \\ \left[\begin{array}{c} \sigma^N \\ \lambda^N \end{array} \right] &= \left[\begin{array}{c} \frac{\partial^2 g}{\partial y^2} \left(y^N \right) \cdot \delta y^N \\ \frac{\partial g}{\partial y} (y^N) \end{array} \right] \ , \qquad N \geq k \geq 1 \end{aligned} \tag{3.17}$$

At the end of the backward in time integration (3.17) provides the gradient and the Hessian vector product

$$\lambda^0 = \left(\frac{\partial \Psi}{\partial y^0} \right)^T \ , \qquad \sigma^0 = \frac{\partial^2 \Psi}{\left(\partial y^0 \right)^2} \cdot u \ .$$

We will now construct specific discrete second order adjoints for the cases where the numerical discretizations (3.14) are of Runge-Kutta and of Rosenbrock types.

3.3. Implicit Runge-Kutta methods. A general s-stage Runge-Kutta method [43, Section II.1] advances the numerical solution $y^k \in \mathbb{R}^n$ at time t^k to the solution $y^{k+1} \in \mathbb{R}^n$ at time $t^{k+1} = t^k + h$ using the formula:

$$\begin{aligned} y^{k+1} &= y^k + h \sum_{i=1}^{s} b_i k_i \ , \quad T_i = t^k + c_i h \ , \quad Y_i = y^k + h \sum_{j=1}^{s} a_{ij} k_j \ , \\ k_i &= f \left(T_i, Y_i \right) \ , \quad i = 1, \cdots, s \ . \end{aligned} \tag{3.18}$$

The coefficients a_{ij}, b_i and c_i are prescribed for the desired accuracy and stability properties. The stage derivative vectors k_i are defined implicitly, and require solving a (set of) nonlinear system(s) by simplified Newton-type methods.

For a general implicit method most of the coefficients $a_{ij} \neq 0$ and one large nonlinear system (3.18) of dimension ns (dimension of the system times the number of stages) needs to be solved to simultaneously find all the stage derivative vectors $k_1, \cdots, k_s$. We will call such methods fully implicit Runge Kutta methods (FIRK).

Singly diagonally implicit Runge Kutta methods (SDIRK) are characterized by a special structure of the coefficients ($a_{ij} = 0$ whenever $i > j$ and $a_{ii} = \gamma$ for all $i = 1, \cdots, s$). In this case the stage derivative vectors $k_1, \cdots, k_s$ are solved for successively, each requiring the solution of a nonlinear system of dimension n. Moreover the same LU factorization can be reused for all systems during the simplified Newton iterations.

Following [44, Section IV.8], for implementation purposes (3.18) is written in the equivalent form

$$\begin{aligned} y^{k+1} &= y^k + \sum_{i=1}^{s} d_i \, Z_i \ , \quad T_i = t^k + c_i h \ , \\ Z_i &= Y_i - y^k = h \sum_{j=1}^{s} a_{ij} \, f\left(T_j, \, y^k + Z_j\right) \ . \end{aligned} \tag{3.19}$$

Replacing the nonlinear system in k_i by a nonlinear system in Z_i has numerical advantages for stiff systems where f has a large Lipschitz constant. The coefficients d_i are defined based on A and b [44, Section IV.8] ($d = A^{-T} b$ if A is invertible; the formula can also be adapted to the case of non-invertible A).

With the compact notation

$$Z = \begin{bmatrix} Z_1 \\ \vdots \\ Z_s \end{bmatrix} \ , \quad F(Z) = \begin{bmatrix} f\left(T_1, \, y^k + Z_1\right) \\ \vdots \\ f\left(T_s, \, y^k + Z_s\right) \end{bmatrix} \ ,$$

and with the Kronecker product denoted $\otimes$, the $n \times n$ identity matrix denoted I_n, the method (3.19) can be written as

$$Z = (hA \otimes I_n) \cdot F(Z) \ . \tag{3.20}$$

The nonlinear system (3.20) in Z can be solved by Newton iterations of the form

$$\begin{aligned} &\left[I_{ns} - h\mathcal{J}\left(Z^{[m]}\right)\right] \Delta Z^{[m]} = Z^{[m]} - (hA \otimes I_n) \cdot F^{[m]} \\ &Z^{[m+1]} = Z^{[m]} - \Delta Z^{[m]} \ , \quad m = 0, 1, \cdots \end{aligned} \tag{3.21}$$

where $F^{[m]} = F\left(Z^{[m]}\right)$, I_{ns} is the $ns \times ns$ identity matrix, and the Jacobian matrix is

$$\mathcal{J} = \begin{bmatrix} a_{11} J\left(T_1, \, y^k + Z_1\right) & a_{12} J\left(T_2, \, y^k + Z_2\right) & \cdots & a_{1s} J\left(T_s, \, y^k + Z_s\right) \\ a_{21} J\left(T_1, \, y^k + Z_1\right) & a_{22} J\left(T_2, \, y^k + Z_2\right) & \cdots & a_{2s} J\left(T_s, \, y^k + Z_s\right) \\ \vdots & & \ddots & \vdots \\ a_{s1} J\left(T_1, \, y^k + Z_1\right) & a_{s2} J\left(T_2, \, y^k + Z_2\right) & \cdots & a_{ss} J\left(T_s, \, y^k + Z_s\right) \end{bmatrix} . \tag{3.22}$$

The cost of the Newton iterations (3.21) is dominated by the LU decompositions of the $ns \times ns$ matrices $I - h\mathcal{J}$.

Implementation Aspects of FIRK Methods. A more efficient approach is provided by simplified Newton iterations where all the ODE Jacobians in (3.22) are evaluated at the beginning of the current step ($J\left(T_i, \, y^k + Z_i\right) \approx J\left(t^k, \, y^k\right)$ for all i). This results in the following approximation of (3.22)

$$\mathcal{J} \approx A \otimes J\left(t^k, \, y^k\right) \tag{3.23}$$

Replacing the matrix (3.22) by (3.23) leads to simplified Newton iterations of the form

$$\begin{aligned} &\Big[I_{ns} - hA \otimes J\left(t^k,\, y^k\right)\Big] \cdot \Delta Z^{[m]} = Z^{[m]} - (hA \otimes I_n)\; F^{[m]} \\ &Z^{[m+1]} = Z^{[m]} - \Delta Z^{[m]}\;, \quad m = 0, 1, \cdots \end{aligned} \tag{3.24}$$

The computational workload spent in the LU factorization of the $ns \times ns$ matrix $I - hA \otimes J\left(t^k,\, y^k\right)$ is $\mathcal{O}(n^3 s^3)$ if full linear algebra is used.

This linear algebra computational work can be reduced to $\mathcal{O}(n^3 s)$ by diagonalizing the inverse of the Runge-Kutta matrix [44, Section IV.8]

$$T^{-1}\, A^{-1}\, T = \Lambda = \mathrm{diag}\left\{\gamma, \alpha_k \pm \beta_k\, i\right\} \tag{3.25}$$

For typical FIRK methods with an odd number of stages the inverse Runge Kutta matrix A^{-1} has one real eigenvalue γ and $(s-1)/2$ pairs of complex conjugate eigenvalues $\alpha_k \pm \beta_k\, i$, while for an even number of stages there typically are $s/2$ complex conjugate pairs.

By changing the variables in the simplified Newton iterations (3.23)

$$Z^{[m]} = (T \otimes I_n)\; W^{[m]}\;, \qquad W^{[m]} = \left(T^{-1} \otimes I_n\right)\; Z^{[m]}\;,$$

and after premultiplying (3.24) by $h^{-1}T^{-1}A^{-1} \otimes I_n$ one obtains the simplified Newton iterations in the W variables

$$\begin{aligned} \left[h^{-1}\Lambda \otimes I_n - I_s \otimes J\left(t^k,\, y^k\right)\right] \cdot \Delta W^{[m]} &= \left(h^{-1}\Lambda \otimes I_n\right)\; W^{[m]} - \left(T^{-1} \otimes I_n\right)\; F^{[m]} \\ W^{[m+1]} &= W^{[m]} - \Delta W^{[m]}\;, \quad m = 0, 1, \cdots \end{aligned} \tag{3.26}$$

The required linear algebra reduces to one real LU decomposition of the $n \times n$ matrix $h^{-1}\gamma\, I_n - J\left(t^k,\, y^k\right)$ (cost $\sim n^3$) and one (or several) complex LU decompositions of the $n \times n$ matrices $h^{-1}(\alpha_k + i\,\beta_k)\, I_n - J\left(t^k,\, y^k\right)$ (cost for each system $\sim 4n^3$). Corresponding forward-backward substitutions are performed at each iteration.

Implementation Aspects of SDIRK Methods. For SDIRK methods the matrix (3.22) has a block lower triangular form. The system (3.20) can be written in the equivalent form

$$\begin{aligned} \Big(h(A - \gamma I_s) \otimes I_n\Big) \cdot F(Z) &= \left(h(A - \gamma I_s) \otimes I_n\right)\left(h^{-1}A^{-1} \otimes I_n\right) \cdot Z \\ &= \left((A - \gamma I_s)\; A^{-1} \otimes I_n\right) \cdot Z\;, \\ &= \left(\Theta \otimes I_n\right) \cdot Z\;, \end{aligned}$$

with

$$\Theta = (A - \gamma I_s)\; A^{-1}$$

which implies that sums of derivative values can be expressed as sums of Z variables

$$h \sum_{j=1}^{i-1} a_{ij}\, f\left(T_j,\, y^k + Z_j\right) = \sum_{j=1}^{i-1} \theta_{ij}\, Z_j\;.$$

Simplified Newton iterations are used to evaluate Z_i successively for each stage $i = 1, \cdots, s$:

$$\begin{aligned} G_i &= h^{-1}\gamma^{-1}\sum_{j=1}^{i-1}\theta_{ij}\, Z_j \\ \left[h^{-1}\gamma^{-1}\, I_n - J\left(t^k,\, y^k\right)\right]\, \Delta Z_i^{[m]} &= h^{-1}\gamma^{-1}\, Z_i^{[m]} - f\left(T_i,\, y^k + Z_i^{[m]}\right) - G_i \\ Z_i^{[m+1]} &= Z_i^{[m]} - \Delta Z_i^{[m]}\ , \quad m = 0, 1, \cdots \end{aligned} \tag{3.27}$$

3.3.1. The Tangent Linear Model of Runge Kutta Methods. The tangent linear model of a the Runge Method in the formulation (3.19) is obtained by taking the variation of (3.19) with respect to changes in initial conditions

$$\begin{aligned} y^{k+1} = y^k + \sum_{i=1}^{s} d_i\, Z_i\ , \qquad & \delta y^{k+1} = \delta y^k + \sum_{i=1}^{s} d_i\, \delta Z_i\ , \\ Z_i = h\sum_{j=1}^{s} a_{ij} f\left(T_j,\, y^k + Z_j\right)\ , \quad & \delta Z_i = h\sum_{j=1}^{s} a_{ij} J\left(T_j,\, y^k + Z_j\right)\cdot\left(\delta y^k + \delta Z_j\right)\ . \end{aligned} \tag{3.28}$$

The resulting formula (3.28) is the same as the one obtained by applying (3.19) to solve the forward model (3.1) together with the tangent linear model (3.6).

Using the compact notation (3.22) of the previous section and

$$\delta Z = \begin{bmatrix} \delta Z_1 \\ \vdots \\ \delta Z_s \end{bmatrix}\ , \quad \mathbb{1}_s = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^s\ , \quad \begin{bmatrix} \delta y^k \\ \vdots \\ \delta y^k \end{bmatrix} = \mathbb{1}_s \otimes \delta y^k\ ,$$

the equation (3.28) for the tangent linear variables δZ is:

$$\Big[I_{ns} - h\,\mathcal{J}\Big]\, \delta Z = h\,\mathcal{J}\, \left(\mathbb{1}_s \otimes \delta y^k\right) \tag{3.29}$$

This can also be obtained by taking the variation of equation (3.20).

Implementation Aspects of the Tangent Linear FIRK Methods. The $ns \times ns$ system of linear equations (3.29) can be solved directly for the tangent linear variables δZ. This is advantageous for very sparse systems. For non-sparse Jacobians one can avoid the $ns \times ns$ LU factorization of (3.29) by using the approximation (3.23) and solving the system (3.29) with the iterative scheme:

$$\begin{aligned} \Big[I_{ns} - hA \otimes J\left(t^k,\, y^k\right)\Big]\cdot \Delta\, \delta Z^{[m]} &= (I - h\mathcal{J})\ \delta Z^{[m]} - h\mathcal{J}\ \left(\mathbb{1}_s \otimes \delta y^k\right)\ , \\ \delta Z_i^{[m+1]} &= \delta Z_i^{[m]} - \Delta\, \delta Z_i^{[m]}\ , \quad m = 0, 1, \cdots \end{aligned}$$

Unlike the direct solution of (3.29) the iterative approach solves the tangent linear variables within a prescribed accuracy which is controlled by our implementation via the number of iterations. In case of iteration non-convergence the solution is

computed by the direct method. The LU decomposition of the system matrix $I_{ns} - hA \otimes J\left(t^k,\, y^k\right)$ (or, more exactly, several equivalent $n \times n$ real and complex LU factorizations) is (are) available from the direct solution (3.26). Thus the calculation of the tangent linear variables piggybacks the calculation of the forward variables. Each step of the forward solution is followed by the corresponding step of the tangent linear model, which reuses the same LU decomposition(s). The additional cost of computing the tangent linear variables in this direct-decoupled Runge Kutta approach is moderate. The accuracy of the tangent linear solution can be monitored via the embedded Runge Kutta scheme, and the step size control can be based on both the forward and the tangent linear error estimates.

Implementation Aspects of the Tangent Linear SDIRK Methods. For SDIRK methods the linear system (3.29) is block upper triangular, and the stage tangent linear variables δZ_i can be solved for successively for stages $i = 1, \cdots, s$

$$\Big[I - h\,\gamma\, J\left(T_i,\, y^k + Z_i\right)\Big] \cdot \delta Z_i = \sum_{j=1}^{i-1} \theta_{ij}\, \delta Z_j + h\,\gamma\, J\left(T_i,\, y^k + Z_i\right) \cdot \delta y^k \ . \tag{3.30}$$

For each stage a $n \times n$ system of linear equations needs to be solved. Each system has a different matrix (since the Jacobians in (3.30) are evaluated at different arguments) and therefore one needs to perform s different LU decompositions.

An iterative solution of (3.30) is possible based on the approximation (3.23)

$$\begin{gathered} G_i = h^{-1}\gamma^{-1} \sum_{j=1}^{i-1} \theta_{ij}\, \delta Z_j + \ J\left(T_i,\, y^k + Z_i\right) \cdot \delta y^k \\ \Big[h^{-1}\gamma^{-1}\, I_n - J\left(t^k,\, y^k\right)\Big] \cdot \Delta\, \delta Z_i^{[m]} = h^{-1}\gamma^{-1}\, \delta Z_i^{[m]} - \ J\left(T_i,\, y^k + Z_i\right) \cdot \delta Z_i^{[m]} - G_i \ , \\ \delta Z_i^{[m+1]} = \delta Z_i^{[m]} - \Delta\, \delta Z_i^{[m]} \ , \quad m = 0, 1, \cdots \end{gathered} \tag{3.31}$$

The iterative solution (3.31) uses the same $n \times n$ LU factorization of $h^{-1}\gamma^{-1}\, I_n - J\left(t^k,\, y^k\right)$ for all stages. Moreover, this LU decomposition is already available from the forward solution (3.27). Thus the tangent linear solution can be obtained at only a moderate additional cost. The accuracy with which the iterations (3.31) solve the system (3.30) is controlled in our implementation. In the case of non-convergence a direct solution is employed.

3.3.2. The Discrete First and Second Order Runge Kutta Adjoints. Following Hager [41] the first order discrete adjoint of the Runge Kutta method (3.18) is:

$$u_i = h\, J^T\left(T_i, Y_i\right) \cdot \left(b_i \lambda^{k+1} + \sum_{j=1}^{s} a_{j,i} u_j\right) \ , \quad i = s, \cdots, 1 \tag{3.32}$$

$$\lambda^k = \lambda^{k+1} + \sum_{j=1}^{s} u_j \ .$$

The discrete second order adjoint of the Runge Kutta method (3.18) is obtained by taking the variation of (3.32) with respect to changes in initial conditions:

$$\begin{aligned} w_i &= h\, J^T\left(T_i, Y_i\right) \cdot \left(b_i \sigma^{k+1} + \sum_{j=1}^{s} a_{j,i} w_j \right) \\ &\quad + h \left(H\left(T_i, Y_i\right) \cdot \delta Y_i \right)^T \cdot \left(b_i \lambda^{k+1} + \sum_{j=1}^{s} a_{j,i} u_j \right) \quad , \quad i = s, \cdots, 1 \qquad (3.33) \\ \sigma^k &= \sigma^{k+1} + \sum_{j=1}^{s} w_j \ . \end{aligned}$$

Recall that the forward stage variables are $Y_i = y_n + Z_i$.

Implementation Aspects of First Order Adjoints of FIRK Methods. Using the matrix (3.22) and the notation

$$U = \begin{bmatrix} u_1 \\ \vdots \\ u_s \end{bmatrix} \ , \qquad G = h \begin{bmatrix} b_1\, J^T(T_1, Y_1)\, \lambda^{k+1} \\ \vdots \\ b_s\, J^T(T_s, Y_s)\, \lambda^{k+1} \end{bmatrix} \ ,$$

the equation (3.32) for the adjoint stage vectors U can be written compactly as

$$\Big[I_{ns} - h\mathcal{J} \Big]^T \cdot U = G \ . \qquad (3.34)$$

Not surprisingly this $ns \times ns$ linear system is the transpose of the tangent linear model system (3.29). The system (3.34) can be solved directly for very sparse systems. For non-sparse systems the cost of the $ns \times ns$ LU factorization can be avoided using again the approximation (3.23) and solving the linear system (3.34) by iterations of the form

$$\begin{aligned} \Big[I_{ns} - hA \otimes J(t^k, y^k) \Big]^T \ \Delta U^{[m]} &= \Big[I_{ns} - h\mathcal{J} \Big]^T \cdot U^{[m]} - G \\ U^{[m+1]} &= U^{[m]} - \Delta U^{[m]} \ . \end{aligned}$$

The LU factorization of $I_{ns} - hA \otimes J(t^k, y^k)$ can be solved effectively in $\mathcal{O}(n^3 s)$ operations using the transformation (3.25) of the Runge Kutta matrix. Note that the real and complex LU factorizations are the same as the ones used in the forward and in the tangent linear model calculations.

Implementation Aspects of First Order Adjoints of SDIRK Methods. For SDIRK methods each stage requires to solve a different $n \times n$ linear system. Sequentially for each stage (in reverse order) we have

$$\Big[I_n - h\gamma\, J\left(T_i, Y_i\right) \Big] u_i = h\, J^T\left(T_i, Y_i\right) \cdot \left(b_i \lambda^{k+1} + \sum_{j=i+1}^{s} a_{j,i} u_j \right) \quad , \quad i = s, \cdots, 1 \ .$$

To avoid s different LU decompositions one can employ an iterative approach that re-uses the same LU decomposition for all stages

$$
\begin{aligned}
G_i &= \gamma^{-1}\, J^T\left(T_i, Y_i\right) \cdot \left(b_i \lambda^{k+1} + \sum_{j=i+1}^{s} a_{j,i} u_j \right) \\
\left[h^{-1}\gamma^{-1}\, I_n - J(t^k, y^k) \right] \Delta u_i^{[m]} &= h^{-1}\gamma^{-1}\, u_i^{[m]} - \; J^T\left(T_i, Y_i\right) \cdot u_i^{[m]} - G_i \;, \\
u_i^{[m+1]} &= u_i^{[m]} - \Delta u_i^{[m]} \;, \quad m = 0, 1, \cdots
\end{aligned}
$$

Our implementation controls the accuracy of the solution via the number of iterations; if non-convergence is detected a direct solution is employed.

Implementation Aspects of Second Order Adjoints of FIRK Methods. Using the matrix (3.22) and the notation

$$
G = h \begin{bmatrix} b_1\, J^T(T_1, Y_1)\, \sigma^{k+1} \\ \vdots \\ b_s\, J^T(T_s, Y_s)\, \sigma^{k+1} \end{bmatrix} + h \begin{bmatrix} \left(H\left(T_1, Y_1\right) \cdot \delta Y_1 \right)^T \cdot \left(b_1 \lambda^{k+1} + \sum_{j=1}^{s} a_{j,1} u_j \right) \\ \vdots \\ \left(H\left(T_s, Y_s\right) \cdot \delta Y_s \right)^T \cdot \left(b_s \lambda^{k+1} + \sum_{j=1}^{s} a_{j,s} u_j \right) \end{bmatrix}
$$

the second order adjoint equation (3.33) can be written compactly as

$$
\left[I_{ns} - h\mathcal{J} \right]^T \cdot W = G \;. \tag{3.35}
$$

This $ns \times ns$ linear system has the same matrix as the first order adjoint (3.34). For very sparse systems one can solve directly for the first order adjoint (3.34) and for the second order adjoint stage vectors (3.35) reusing the same $ns \times ns$ LU decomposition.

Like with the first order adjoint, one can avoid the $ns \times ns$ LU decomposition and solve the equation (3.33) by iterations of the form

$$
\begin{aligned}
\left[I - hA \otimes J(t^k, y^k) \right]^T \Delta W^{[m]} &= \begin{bmatrix} w_1^{[m]} - h\, J^T(T_1, Y_1) \cdot \left(\sum_{j=1}^{s} a_{j,1} w_j^{[m]} \right) \\ \vdots \\ w_s^{[m]} - h\, J^T(T_s, Y_s) \cdot \left(\sum_{j=1}^{s} a_{j,s} w_j^{[m]} \right) \end{bmatrix} - G \\
W^{[m+1]} &= W^{[m]} - \Delta W^{[m]} \;, \quad m = 0, 1, \cdots
\end{aligned}
$$

Our implementation controls the accuracy of the second order adjoint solution via the number of iterations; if non-convergence is detected a direct solution is employed.

Implementation Aspects of Second Order Adjoint of SDIRK Methods. Equation (3.33) is solved stage by stage ($i = s, s-1, \cdots, 1$)

$$
\begin{aligned}
G_i = \gamma^{-1}\, J^T\,(T_i, Y_i)\, \left(b_i \sigma^{k+1} + \sum_{j=i+1}^{s} a_{j,i} w_j \right) \\
+ \gamma^{-1} \left(H\,(T_i, Y_i) \cdot \delta Y_i \right)^T \cdot \left(b_i\, \lambda^{k+1} + \sum_{j=i+1}^{s} a_{j,i} u_j + \gamma u_i \right) \qquad (3.36) \\
\left[h^{-1}\gamma^{-1}\, I_n - J^T(T_i, Y_i) \right] w_i = G_i \ .
\end{aligned}
$$

The direct solution of (3.36) requires a different $n \times n$ LU factorization for each stage. To avoid this the systems (3.36) can be solved by an iterative approach which uses the same LU factorization for all stages:

$$
\begin{aligned}
\left[h^{-1}\gamma^{-1}\, I_n - J^T(t^k, y^k) \right] \Delta\, w_i^{[m]} = h^{-1}\gamma^{-1}\, w_i^{[m]} - \ J^T\,(T_i, Y_i)\ w_i^{[m]} - G_i \ , \qquad (3.37) \\
w_i^{[m+1]} = w_i^{[m]} - \Delta\, w_i^{[m]} \ , \quad m = 0, 1, \cdots
\end{aligned}
$$

3.4. Rosenbrock Methods. An s-stage Rosenbrock method [44, Section IV.7] computes the next-step solution by the formulas

$$
\begin{aligned}
y^{k+1} = y^k + \sum_{i=1}^{s} m_i k_i \ , \quad T_i = t^k + \alpha_i h \ , \qquad Y_i = y^k + \sum_{j=1}^{i-1} a_{ij} k_j \ , \qquad (3.38) \\
\left[h^{-1}\gamma^{-1} I_n - J(t^k, y^k) \right] k_i = f\,(\,T_i,\, Y_i\,) + \sum_{j=1}^{i-1} \frac{c_{ij}}{h} k_j + h\gamma_i f_t\left(t^k, y^k\right) \ .
\end{aligned}
$$

where s is the number of stages. The formula coefficients a_{ij}, c_{ij}, m_i, γ, and γ_i give the order of consistency and the stability properties. The ODE Jacobian J as well as the partial derivative of the ODE function with respect to time f_t are explicitly used in the formula. Only one LU factorization per step is used. Moreover, no iterations are needed in the solution process. Rosenbrock methods are well suited for mildly stiff problems and moderate accuracy [44]. They are designed to have excellent stability properties, preserve linear invariants (a.k.a. mass) and are computationally efficient. In [65] we have shown that Rosenbrock methods work well for solving atmospheric chemistry problems.

3.4.1. The Tangent Linear Model of Rosenbrock Methods. To obtain the Rosenbrock tangent linear model one takes the variation of the method (3.38) with

respect to changes in the initial conditions

$$
\begin{aligned}
y^{k+1} = y^k + \sum_{i=1}^{s} m_i k_i, \quad \delta y^{k+1} = \delta y^k + \sum_{i=1}^{s} m_i \ell_i \qquad (3.39)\\
T_i = t^k + \alpha_i h \ , \quad Y_i = y^k + \sum_{j=1}^{i-1} a_{ij} k_j \ , \quad \delta Y_i = \delta y^k + \sum_{j=1}^{i-1} a_{ij} \ell_j \\
\Big[h^{-1}\gamma^{-1} I_n - J(t^k, y^k)\Big] \cdot k_i = f\left(T_i,\, Y_i\right) + \sum_{j=1}^{i-1} \frac{c_{ij}}{h} k_j + h\gamma_i f_t\left(t^k, y^k\right) \ , \\
\Big[h^{-1}\gamma^{-1} I_n - J(t^k, y^k)\Big] \cdot \ell_i = J\left(T_i,\, Y_i\right) \cdot \delta Y_i + \sum_{j=1}^{i-1} \frac{c_{ij}}{h} \ell_j \\
+ \Big(H(t^k, y^k) \cdot k_i\Big) \cdot \delta y^k + h\gamma_i J_t\left(t^k, y^k\right) \cdot \delta y^k
\end{aligned}
$$

Note that the tangent linear stages ℓ_i require explicitly the ODE Hessian H (due to the explicit presence of the ODE Jacobian in the forward Rosenbrock formula). The method requires a single $n \times n$ LU decomposition per step to obtain both the concentrations y^{k+1} and the sensitivities δy^{k+1}. The additional cost required for the tangent linear calculations is alleviated by the reuse of the same LU decomposition. However the calculation of the Hessian and the corresponding tensor-vector products in the tangent linear calculations can be relatively expensive.

3.4.2. The Discrete First and Second Order Adjoints of Rosenbrock Methods. The discrete first order adjoint of the (non-autonomous) Rosenbrock method (3.38) is [63]

$$
\begin{aligned}
\Big[h^{-1}\gamma^{-1} I_n - J^T(t^k, y^k)\Big] \cdot u_i = m_i \lambda^{k+1} + \sum_{j=i+1}^{s} \Big(a_{j,i} v_j + \frac{c_{j,i}}{h} u_j\Big) \ , \qquad (3.40)\\
v_i = J^T\left(T_i, Y_i\right) \cdot u_i \ , \qquad i = s, \cdots, 1 \ , \\
\lambda^k = \lambda^{k+1} + \sum_{i=1}^{s} \Big(H(t^k, y^k) \cdot k_i\Big)^T \cdot u_i + h J_t^T(t^k, y^k) \cdot \sum_{i=1}^{s} \gamma_i u_i + \sum_{i=1}^{s} v_i
\end{aligned}
$$

The formula (3.40) makes explicit use of the partial derivative of the Jacobian with respect to time J_t and of the ODE Hessian H (both computed once at the beginning of the time step). The same LU factorization is used by all stages; this is the LU factorization used in both the forward and the tangent linear methods.

The discrete second order adjoint of a Rosenbrock method is obtained by taking the variation of (3.40) with respect to changes in the initial condition

$$
\begin{aligned}
\Big[h^{-1}\gamma^{-1} I_n - J^T(t^k, y^k)\Big] \cdot w_i = m_i \sigma^{k+1} + \Big(H(t^k, y^k) \cdot \delta y^k\Big)^T \cdot u_i \qquad (3.41)\\
+ \sum_{j=i+1}^{s} \Big(a_{j,i} z_j + \frac{c_{j,i}}{h} w_j\Big) \ , \quad i = s, \cdots, 1 \ , \\
z_i = J^T\left(T_i, Y_i\right) \cdot w_i + \Big(H(T_i, Y_i) \cdot \delta Y_i\Big)^T \cdot u_i \ ,
\end{aligned}
$$

$$\sigma^k = \sigma^{k+1} + \sum_{i=1}^{s} \Big(H(t^k, y^k) \cdot k_i \Big)^T \cdot w_i + \sum_{i=1}^{s} \Big(H(t^k, y^k) \cdot \ell_i \Big)^T \cdot u_i$$
$$+ \frac{\partial}{\partial y^k} \left(\sum_{i=1}^{s} \Big(H(t^k, y^k) \cdot k_i \Big)^T \cdot u_i \right) \cdot \delta y^k$$
$$+ h J_t^T(t^k, y^k) \cdot \sum_{i=1}^{s} \gamma_i w_i + h \Big(H_t(t^k, y^k) \cdot \delta y^k \Big)^T \cdot \sum_{i=1}^{s} \gamma_i u_i + \sum_{i=1}^{s} z_i$$

the ODE Hessian with respect to time H_t. It also requires the derivative of the Hessian-vector products with respect to the solution ($\partial/\partial y\{(H(t,y)\cdot k)^T \cdot u\}$). This term involves third order derivatives of the ODE right hand side function

$$\frac{\partial^3 f_i(t,y)}{\partial y_j \partial y_k \partial y_\ell} \ .$$

The calculation of these high order derivatives is challenging, making the second order discrete adjoint Rosenbrock formula (3.41) less attractive in practice. Nevertheless the approach is useful for typical chemical mechanisms which involve only monomolecular and bimolecular reactions. For these mechanisms the underlying ODE function $f(t,y)$ is quadratic in y, and the third order derivatives of f vanish.

3.5. KPP. The implementation of numerical integrators for chemistry can be done automatically using the Kinetic PreProcessor KPP software tools [25]. KPP was extended [22, 63] to produce a rapid and efficient implementation of the code for sensitivity analysis of chemical kinetic systems. KPP builds Fortran77, Fortran90, C, or Matlab simulation code for chemical systems with chemical concentrations changing in time according to the law of mass action kinetics. KPP generates the following building blocks:

1. *Fun*: the ODE function $f(t,y)$;
2. *Jac, Jac_SP*: the ODE Jacobian $J(t,y)$ in full or in sparse format;
3. *KppDecomp*: sparse LU decomposition for the Jacobian;
4. *KppSolve, KppSolveTR*: solve sparse system with the Jacobian matrix and its transpose;
5. *Jac_SP_Vec* ($w \leftarrow J \cdot u$), *JacTR_SP_Vec* ($w \leftarrow J^T \cdot u$): sparse matrix-vector multiplication of the Jacobian (transposed or not) with user vector;
6. *Hess*: the ODE Hessian $H(t,y)$ represented in sparse 3-tensor format;
7. *Hess_Vec* ($w \leftarrow (H \cdot u) \cdot v$), *HessTR_Vec* ($w \leftarrow (H \cdot u)^T \cdot v$): sparse tensor product of the Hessian (or its transpose) with user vectors; same as the derivative of Jacobian (transposed) vector product times vector.

In [22, 63] we show how these KPP building blocks can be used to implement very efficiently code for direct and adjoint sensitivity analysis of chemical systems. A related approach was taken in the early application of the 4D-Var to chemical data assimilation by Fisher and Lary [35].

4. Discrete SOA for the Transport System. In the STEM model [14] the transport equation is solved using a directional x, y, and z split approach. The basic numerical techniques solve the one-dimensional transport equation

$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x} + \frac{1}{\rho}\frac{\partial}{\partial x}\left(\rho K \frac{\partial c}{\partial x}\right) \ , \quad c(t, x_{in}) = c_{in}(t) \ , \quad K\frac{\partial c}{\partial x}\bigg|_{x_{out}} = 0 \ . \tag{4.1}$$

In STEM the horizontal advection term is discretized by the third order upwind finite difference formula [47]

$$-\left(u\frac{\partial c}{\partial x}\right)\bigg|_{x=x_i} = \begin{cases} u_i\left(-c_{i-2} + 6c_{i-1} - 3c_i - 2c_{i+1}\right)/(6\Delta x) \ , & \text{if} \quad u_i \geq 0 \\ u_i\left(2c_{i-1} + 3c_i - 6c_{i+1} + c_{i+2}\right)/(6\Delta x) \ , & \text{if} \quad u_i < 0 \ . \end{cases} \tag{4.2}$$

The diffusion terms are discretized by the second order central differences

$$\frac{1}{\rho}\frac{\partial}{\partial x}\left(\rho K\frac{\partial c}{\partial x}\right)\bigg|_{x=x_i} = \frac{(\rho_{i+1}K_{i+1} + \rho_i K_i)(c_{i+1} - c_i) - (\rho_i K_i + \rho_{i-1}K_{i-1})(c_i - c_{i-1})}{2\rho_i \Delta x^2} \ . \tag{4.3}$$

For the inflow boundary the advection discretization drops to the first order upwind scheme, which makes the order of consistency of the whole scheme quadratic for the interior points of the domain. For the outflow boundary the advection discretization also drops to the first order upwind scheme.

The space semi-discretization leads to the linear ordinary differential equation

$$\frac{dc}{dt} = A(t)\, c(t) + B(t) \ , \tag{4.4}$$

where the matrix $A(t)$ depends on the wind field, the diffusion tensor, and the air density but it does not depend on the unknown concentrations (for the discretization schemes under consideration). The vector B(t) represents the Dirichlet boundary conditions.

The forward system is advanced in time from t^k to $t^{k+1} = t^k + \Delta t$ using Crank-Nicholson

$$c^{k+1} = \left(I - \frac{\Delta t}{2}A(t^{k+1})\right)^{-1}\left[\left(I + \frac{\Delta t}{2}A(t^k)\right)c^k + \Delta t\frac{B(t^k) + B(t^{k+1})}{2}\right] \tag{4.5}$$

The chosen discretization leads to pentadiagonal matrices and systems which can be solved very efficiently.

Equation (4.5) represents the forward discrete model for horizontal transport. The corresponding adjoint system is then advanced backwards in time using the discrete adjoint formulation

$$\lambda^k = \left(I + \frac{\Delta t}{2}A^T(t^k)\right)\ \left(I - \frac{\Delta t}{2}A^T(t^{k+1})\right)^{-1}\lambda^{k+1} \ . \tag{4.6}$$

Equation (4.6) is a consistent time discretization of the continuous adjoint equation. Because of the linear discretization the second order adjoint formula obtained by

taking the variation of (4.6) has the same form as (4.6)

$$\sigma^k = \left(I + \frac{\Delta t}{2} A^T(t^k)\right) \left(I - \frac{\Delta t}{2} A^T(t^{k+1})\right)^{-1} \sigma^{k+1} \ . \tag{4.7}$$

This means that the same adjoint transport routines are used for both the first order adjoint (4.6) and for the second order adjoint solutions (4.7). Moreover, it is possible to reuse the same LU decomposition of $I - (\Delta t/2)A(t^{k+1})$ for both the first and the second order adjoint calculations.

The vertical advection term is discretized by the first order upwind finite difference formula, while the vertical diffusion is discretized by the second order central differences on a non-uniform vertical grid. The top boundary condition is Dirichlet for inflow and Neumann for outflow, while the Neumann ground boundary condition accounts for emission and deposition fluxes. Similar considerations hold for the discrete adjoints of the vertical transport. The same discrete adjoint vertical transport routine are used for both the first and the second order adjoint solutions.

Other transport discretization schemes are possible, notably monotonic finite volume schemes for advection that use flux or slope limiting to avoid the creation of spurious oscillations. The analysis of the corresponding discrete adjoint schemes performed in [53] reveals possible pitfalls. A discussion of discrete second order adjoints for monotonic advection schemes is outside the scope of this paper.

5. Applications of Second Order Adjoints.

5.1. Sensitivity Analysis. We first consider the SAPRC-99 atmospheric chemistry mechanism [15, 16] which considers the gas-phase atmospheric reactions of volatile organic compounds (VOCs) and nitrogen oxydes (NO_X) in urban and regional settings. The chemical mechanism was developed at University of California, Riverside by Dr. W.P.L. Carter for use in airshed models for predicting the effects of VOC and NO_X emissions on tropospheric secondary pollutants formation such as ozone (O_3), peroxyacetyl nitrate (PAN), etc. In our analysis we consider the condensed fixed-parameter version of the SAPRC-99 mechanism which takes into consideration 235 reactions among 81 variable chemical species (in addition O_2, H_2, CH_4, and H_2O concentrations are considered fixed), and is currently incorporated into the three-dimensional regional-scale model STEM-II [13].

The 24 hours simulation interval starts at $t^0 = 12$pm and ends at $t^{\mathrm{F}} = 12$pm the next day. We consider two numerical methods. The first one is SDIRK-4 [44], a five-stage fourth order L-stable and stiffly accurate SDIRK method, with an embedded method of order three for error control. The second method is RODAS [44], a six stage fourth order stiffly accurate Rosenbrock method with error control. Both methods, their tangent linear models, and their first and second order discrete adjoints have been implemented in the KPP library, and use the specially prepared sparse Jacobians and Hessians. All simulations are carried out with rtol=1e-5 and atol=1e-3 $molec/cm^3$.

TABLE 5.1

Validation of the second order adjoints against finite differences of first order adjoints. The RMS norm of the relative difference decreases for smaller perturbations.

	$\Psi^1 = PAN\left(t^{\mathrm{F}}\right)$		$\Psi^2 = 0.5\, O_3^2\left(t^{\mathrm{F}}\right)$	
ε	SDIRK-4	RODAS	SDIRK-4	RODAS
0.1	1.15E-01	4.22E-07	1.22E-01	1.30E-01
0.01	2.99E-03	3.21E-09	7.90E-03	1.36E-02

We compute adjoint sensitivities of two different cost functions. The first one is the PAN concentration at the final time, the second is half the O_3 concentration squared at the final time

$$\Psi^1 = PAN\left(t^{\mathrm{F}}\right) \quad \text{and} \quad \Psi^2 = \frac{1}{2} O_3^2\left(t^{\mathrm{F}}\right) \ .$$

The initial NO_X concentrations are perturbed from their reference values as follows

$$NO\left(t^0\right) \leftarrow (1+\varepsilon) \cdot NO^{\mathrm{reference}}\left(t^0\right) \ , \quad NO_2\left(t^0\right) \leftarrow (1+\varepsilon) \cdot NO_2^{\mathrm{reference}}\left(t^0\right) \ . \tag{5.1}$$

For each cost function and each perturbation we compute the first order adjoints $\lambda^{1,2}(\varepsilon)$. For the reference solution we also compute the first and second order adjoints ($\lambda^{1,2}(0)$ and $\sigma^{1,2}$ respectively). Using the relation $\sigma^{1,2} \approx \lambda^{1,2}(\varepsilon) - \lambda^{1,2}(0)$ we validate our implementation by checking the second order adjoint against the finite difference of first order adjoints. Specifically we compute the RMS norm of the relative error for all n components

$$ERR^{1,2} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\frac{\lambda_i^{1,2}(\varepsilon) - \lambda_i^{1,2}(0) - \sigma_i^{1,2}}{\max\left(\left| \sigma_i^{1,2} \right|, tol \right)} \right)^2} \ . \tag{5.2}$$

These relative errors are reported in Table 5.1. We see that for both cost functions and for both methods the agreement between the second order adjoint and the finite difference of first order adjoints is improved with decreasing the perturbation magnitude. The agreement for the RODAS method results on the first cost function is excellent (we have no explanation for this).

The computational costs associated with the first and second order adjoints are reported in Table 5.2. For the SDIRK method the cost of the first order adjoint (including the forward model) is about twice the cost of the forward integration, while the cost of the second order adjoint is about three times the cost of a forward integration. For the Rosenbrock method the cost of the first order adjoint is more than twice the forward cost (due to the added overhead of computing Hessians). For the SAPRC-99 chemical mechanism the Hessian is constant, and the third order derivative terms that appear in the discrete second order Rosenbrock adjoint vanish. The cost

TABLE 5.2

CPU times for the 24 hours simulation of the SAPRC-99 box model. FWD denotes the forward model, TLM the tangent linear model, FOA the first order adjoint, and SOA the second order adjoint

	SDIRK-4		RODAS	
Simulation	CPU time	Scaled time	CPU time	Scaled time
FWD only	0.71 sec	1	0.30 sec	1
FWD + TLM	1.24 sec	1.75	0.57 sec	1.9
FWD followed by FOA	1.39 sec	1.96	0.67 sec	2.23
FWD + TLM followed by FOA + SOA	2.11 sec	2.97	1.23 sec	4.01

of Rosenbrock second order adjoint is about four times that of a forward integration. The sparse linear algebra implemented by KPP is extremely efficient. Due to this efficiency only modest cost savings result from reusing the (inexpensive) sparse LU decompositions in tangent linear and in adjoint calculations.

Next we show how second order adjoints can be used in sensitivity analysis, and can extend the range of validity of sensitivity analysis for highly nonlinear chemical systems. The results are shown in Figure 5.1. The changes of PAN concentrations at the end of the 24 hours interval are nonlinear with respect to the initial concentrations of NO and NO_2. We let the initial NO and NO_2 initial concentrations vary according to (5.1) within $\pm 40\%$ from their reference values ($\varepsilon \in [-0.4, +0.4]$). The change in the final PAN concentration is predicted by first and second order Taylor series about the reference initial concentrations. The first and the second terms in the Taylor series are obtained using the first and the second order adjoints respectively:

$$
\begin{aligned}
\Psi^1\left(c^0\right) &= PAN\left(t^{\mathrm{F}}\right)\big|_{c(t^0)=c^0} \ , \\
\Psi^1\left(c^0+\Delta c^0\right) &= PAN\left(t^{\mathrm{F}}\right)\big|_{c(t^0)=c^0+\Delta c^0} \ , \\
\Delta PAN &= \Psi^1\left(c^0+\Delta c^0\right) - \Psi^1\left(c^0\right) \\
&= \left(\nabla_{c^0}\Psi^1\left(c^0\right)\right)^T \cdot \Delta c^0 + \frac{1}{2}\left(\Delta c^0\right)^T \cdot \left(\nabla^2_{c^0,c^0}\Psi^1\left(c^0\right)\right) \cdot \Delta c^0 + \cdots \\
&= \lambda^T \cdot \Delta c^0 + \frac{1}{2}\sigma^T \cdot \Delta c^0 + \cdots
\end{aligned}
$$

We see in Figure 5.1 that the first order approximation is poor for large perturbations, while the second order approximation continues to work well for large deviations from reference.

5.2. The Chemical Transport Model and the Test Case. The previous sensitivity analysis application is performed in a box model. We now consider full three-dimensional chemistry and transport simulations. The numerical experiments

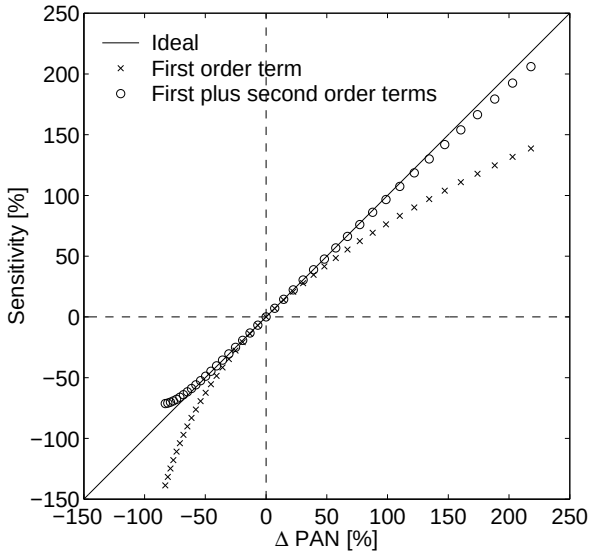
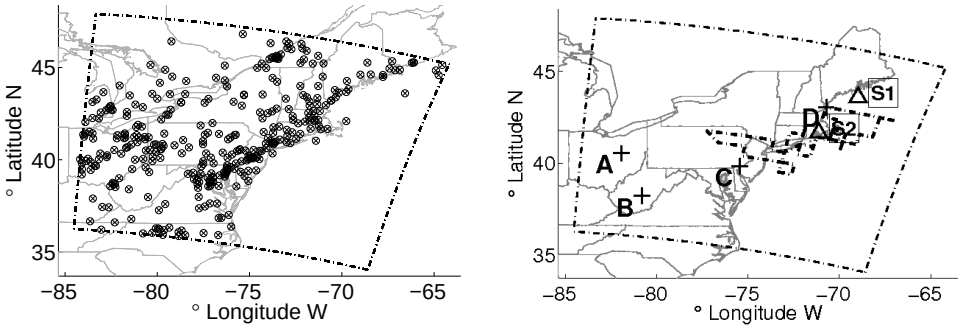


FIG. 5.1. *Sensitivity of final PAN concentration with respect to the initial concentrations of NO and NO_2. The changes in PAN concentration for different changes in the initial conditions Δc^0 are shown against the first order approximation ($\lambda^T \cdot \Delta c^0$, marked with "x") and against the second order approximation ($\lambda^T \cdot \Delta c^0 + 1/2 \cdot \sigma^T \cdot \Delta c^0$, marked with "o"). The first order sensitivity analysis is inaccurate for this highly nonlinear system, while the second order sensitivity analysis accurately predicts the changes in PAN.*

use the state-of-the-art regional atmospheric photochemistry and transport model STEM [14].

The test case is a real-life simulation of air pollution in North–Eastern United States in July 2004 as shown in Figure 5.2 (the dash-dotted line delimits the computational domain). The computational domain covers $1500 \times 1320 \times 20$ Km with a horizontal resolution of 60×60 Km and a variable vertical resolution (resulting in a 3-dimensional computational grid of $25 \times 22 \times 21$ points). Real data is used for the initial concentrations, meteorological fields, boundary values, and emission rates starting at 0 GMT of July 20^{th}, 2004. This data corresponds to the ICARTT (International Consortium for Atmospheric Research on Transport and Transformation) [37] campaign in July 2004. A detailed description of the ICARTT fields and data can be found in [71].

Data assimilation is the process of integrating observational data and model predictions to obtain an optimal representation of the state of the atmosphere. As more chemical observations in the troposphere are becoming available, chemical data assimilation is expected to play an essential role in air quality forecasting, similar to the role it has in numerical weather prediction. Variational techniques for data assimilation are well-established [30, 54, 70], and the 4D-Var framework is the current state-of-the-art in meteorological [20, 60] and chemical [31, 32, 51, 63, 64, 66, 67] data



(a) ICARTT ground stations (b) selected stations

FIG. 5.2. *(a) The location of the ground measuring stations in support of the ICARTT campaign (340 in total) (b) The location of the two ozonesondes (S1, S2) and the path of the P3-B flight that provide observations used in data assimilation. Also shown are the locations of four selected stations (A–D) that will be used to illustrate the assimilation results.*

assimilation.

The observations used in this paper for data assimilation are real ozone (O_3) measurements taken during the ICARTT [37] campaign in summer 2004 [71]. Ground level ozone measurements are provided hourly by the EPA's AirNow network of ground stations (340 in total) whose locations are shown in Figure 5.2(a). Elevated ozone measurements are taken by two ozonesondes and a P3-B flight, all shown in Figure 5.2(b). More ozone observations are available from two Mozaic flights. The setting for data assimilation is described in detail in [18]. We denote by z^k the observations available at discrete times t^k, $k = 1, \cdots, N$. The last observation is taken at the final time, $t^N = t^{\mathrm{F}}$. The observations are functions of the state at the corresponding time; a linear mapping operator D interpolates the model states (available on the grid) to observations

$$z^k = D \cdot y^k + \varepsilon^k \ , \quad \langle \varepsilon^k \rangle = 0 \ , \quad \langle \varepsilon^k \cdot \left(\varepsilon^k\right)^T \rangle = R_k \ ,$$

where the observational and representativeness errors ε^k are assumed to be normally distributed and uncorrelated random variables with zero mean and covariances R_k, $\varepsilon^k \in \mathcal{N}(0, R_k)$.

We consider the situation where the initial state y^0 is uncertain, and is represented as a normally distributed random variable with mean y^B ("background state") and covariance B. In four-dimensional variational (4D-Var) approach to data assimilation one uses the information from both the background state and the observations to determine the most likely initial state y^a as the minimizer of the following cost functional

$$\Psi\left(y^0\right) = \frac{1}{2}\left(y^0 - y^B\right)^T \ B^{-1} \ \left(y^0 - y^B\right) + \frac{1}{2}\sum_{k=1}^{N}\left(Dy^k - z^k\right)^T \ R_k^{-1} \ \left(Dy^k - z^k\right) \quad (5.3)$$

The first term represents a penalty for the departure from the background value y^B. The next terms measure the mismatch between model predictions and observations. In the case where the model is linear, the background uncertainty is Gaussian $y^0 \in \mathcal{N}(y^B, B)$, and the observation uncertainties are Gaussian, the aposteriori probability density of the initial state is also Gaussian. In this situation the cost function (5.3) represents the negative logarithm of the aposteriori Gaussian probability density function, and the minimizer y^a of (5.3) represents the maximum likelihood estimator of the state. In the following numerical experiments only the initial O_3 concentration is considered uncertain. All first and second order adjoint derivatives of (5.3) are computed with respect to the initial ozone concentration field, unless specified otherwise.

The efficient numerical minimization of (5.3) requires the gradient of the cost function ($\lambda^0 = (\partial\Psi/\partial y^0)^T$) as well as second order derivative information in the form of Hessian vector products ($\sigma^0 = \partial^2\Psi/(\partial y^0)^2 \cdot u$). These derivatives are obtained via the first and second order adjoint models as follows. Consider the CTM represented compactly as (2.2). First the forward and the tangent linear models are solved together forward in time:

$$
\begin{aligned}
& y^0 = y\left(t^0\right) \\
& \delta y^0 = u \\
& \text{Save } y^0,\ \delta y^0 \text{ on tape} \\
& \text{FOR } k = 1, 2, \cdots, N-1, N \text{ DO} \\
& \qquad y^k = \mathcal{N}_k\left(y^{k-1}\right) \\
& \qquad \delta y^k = \mathcal{N}_k'\left(y^{k-1}\right) \cdot \delta y^{k-1} \\
& \qquad \text{Save } y^k,\ \delta y^k \text{ on tape} \\
& \text{END FOR}
\end{aligned}
\tag{5.4}
$$

Next the first and second order adjoint models defined for the cost function (5.3) are solved together, backward in time:

$$
\begin{aligned}
& \sigma^N = 0 \\
& \lambda^N = 0 \\
& \text{Read } y^N,\ \delta y^N \text{ from tape} \\
& \text{FOR } k = N, N-1, \cdots, 2, 1 \text{ DO} \\
& \qquad \lambda^k = \lambda^k + D^T R_k^{-1} \cdot \left(D\, y^k - z^k\right) \\
& \qquad \sigma^k = \sigma^k + D^T R_k^{-1} D \cdot \delta y^k \\
& \qquad \text{Read } y^{k-1},\ \delta y^{k-1} \text{ from tape} \\
& \qquad \lambda^{k-1} = -\left(\mathcal{N}_k'(y^{k-1})\right)^T \cdot \lambda^k \\
& \qquad \sigma^{k-1} = -\left(\mathcal{N}_k'(y^{k-1})\right)^T \cdot \sigma^k - \left(\mathcal{N}_k''\left(y^{k-1}\right) \cdot \delta y^{k-1}\right)^T \cdot \lambda^k \\
& \text{END FOR} \\
& \lambda^0 = \lambda^0 + B^{-1} \cdot \left(y^0 - y^B\right) \\
& \sigma^0 = \sigma^0 + B^{-1} \cdot \delta y^0 \ .
\end{aligned}
\tag{5.5}
$$

5.3. Validation of the 3D Second Order Adjoints. We have implemented second order adjoint capabilities in the three-dimensional STEM model according to (5.4)–(5.5). The chemistry is solved using Rosenbrock methods implemented efficiently via KPP, as discussed in Section 3. The second order adjoints for transport are implemented as discussed in Section 4. The CPU times associated with the first and second order adjoint calculation are reported in Table 5.3. We see that the CPU time needed for a second order adjoint calculation is less than twice the time for a first order adjoint calculation.

Simulation	CPU time	Scaled time
FWD only	35.8 min	1
FWD followed by FOA	83.92 min	2.34
FWD + TLM followed by FOA + SOA	127.6 min	3.56

TABLE 5.3

CPU times for a 12 hours three-dimensional chemistry and transport simulation. FWD denotes the forward model, TLM the tangent linear model, FOA the first order adjoint, and SOA the second order adjoint. Shown are the wall clock times and the times relative to the forward model run.

Validation of Second Order Adjoints against Finite Differences. We now validate the correctness of the three-dimensional second order adjoints against finite differences of first order adjoints. The cost function for this experiment is (5.3). Additional experiments (not shown here) using cost functions defined as the sum of concentrations of ground level PAN over the coastal area at final time lead to similar conclusions.

We run two 12-hour simulations, one with the reference initial conditions and the other with perturbed initial NO_2 concentrations. The second order adjoint σ for the reference run is shown in Figure 5.3(a) (ground level values). The difference $\Delta\lambda$ of first order adjoints computed in the perturbed run and in the reference run is shown in Figure 5.3(b) (ground level values). The second order adjoint is very similar to the finite difference of first order adjoints, confirming the correctness of our implementation.

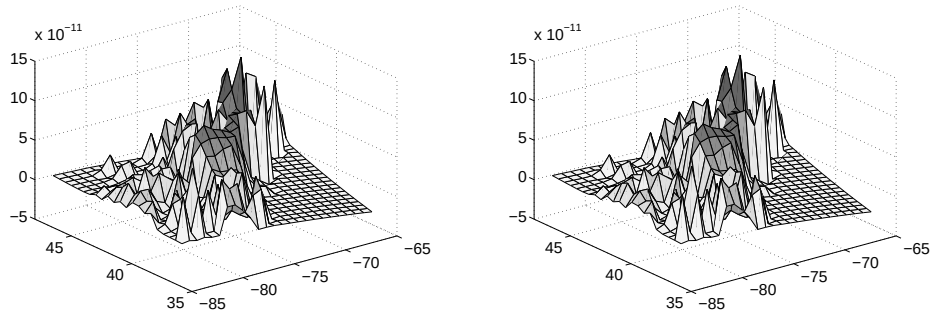
Validation of Hessian Symmetry. Some of the applications discussed next (optimization, Hessian eigenvalue decomposition) require the Hessian to be symmetric. Second order adjoint methodology provides Hessian-vector products. Even if the full Hessian of the cost function (5.3) is not available we check its symmetry as follows.

The Hessian product with two vectors δy^1 and δy^2, are computed starting from the same set of initial concentrations y^0,

$$\sigma^1 = \mathcal{H}(y^0) \cdot \delta y^1 \ , \quad \sigma^2 = \mathcal{H}(y^0) \cdot \delta y^2 \ ,$$

and dot products are taken between the second order adjoints and the perturbations. The two dot products are equal if the computed Hessian is symmetric

$$\left(\delta y^2\right)^T \cdot \sigma^1 = \left(\delta y^2\right)^T \cdot \mathcal{H}(y^0) \cdot \delta y^1 = \left(\delta y^1\right)^T \cdot \mathcal{H}(y^0) \cdot \delta y^2 = \left(\delta y^1\right)^T \cdot \sigma^2 \ .$$



(a) second order adjoint σ for the reference run

(b) finite difference $\Delta\lambda$ of first order adjoints (perturbed minus reference)

FIG. 5.3. *Validation of the second order adjoint for the three-dimensional chemical transport model against finite difference of first order adjoints.*

In our experiments we set the first vector to be the set of initial conditions, $\delta y^1 = y^0$. The second vector is chosen in two ways. First, δy^1 is advanced from t^0 to t^{F} using the tangent linear model and δy^2 is taken to be the solution of the tangent linear model at the final time. Second, δy^2 is taken to be a vector with random entries (scaled element-wise by y^0 to preserve the relative magnitude among concentrations of different species).

For each test we run 1 hour, 4 hours and 8 hours simulations. The results are shown in Table 5.4. The two products are close to each other in both tests, which indicates that the Hessian (computed by the second order adjoint method) is symmetric. Small differences are acceptable considering the large size of the vectors (10^5).

TABLE 5.4
Results for Hessian symmetry tests

Test	Product	1h	4h	8h
1	$\left(\delta y^1\right)^T \cdot \sigma^2$	2.5837e+4	1.8897e+5	3.2224e+5
	$\left(\delta y^2\right)^T \cdot \sigma^1$	2.4010e+4	1.8806e+5	3.1812e+5
2	$\left(\delta y^1\right)^T \cdot \sigma^2$	1.3012e+4	9.8862e+4	1.8372e+5
	$\left(\delta y^2\right)^T \cdot \sigma^1$	1.3012e+4	9.8705e+4	1.8316e+5

5.4. Data Assimilation Using Second Order Information. The minimization problem (5.3) arising in data assimilation is large-scale, with the typical number of control variables in chemical transport models ranging between $10^4 - 10^7$. In our experiments we minimize (5.3) for the initial ozone conditions, which amounts to 11,550 control variables. In this section we investigate the performance of different numerical optimization methods. The first and second order derivatives (where needed) are

computing according to (5.4)–(5.5).

Large-scale unconstrained optimization solvers are discussed in [55, 58, 59]. Two classes of numerical optimization methods are popular in variational data assimilation: quasi-Newton and nonlinear conjugate gradients. They require only first order gradient information, and can handle very large scale and highly nonlinear problems. In our numerical tests we consider one method from each class to minimize (5.3).

Comparisons of different optimization methods for data assimilation with fluid flow models are given by Alekseev and Navon [3] and Daescu and Navon [23]. Quasi-Newton methods approximate the inverse of the Hessian matrix by a symmetric positive definite matrix, which is updated at every step using the new search directions and the new gradients. The Broyden Fletcher Goldfarb Shanno (BFGS) Hessian update formula [57] has proved effective in many applications. The limited-memory version (L-BFGS) [52] can handle very large problems by storing only a finite number of search directions and gradients used in the approximation of the inverse Hessian. The L-BFGS-B implementation of Zhu, Byrd, and Nocedal [11, 74] is able to handle bound constraints and has become the gold standard in variational data assimilation. In our previous data assimilation work [18, 24, 64] we have successfully used this package for chemical data assimilation with full 3D models. This L-BFGS-B implementation [11, 74] is used for the numerical results reported in this paper.

Extensions of the linear conjugate gradients method to nonquadratic problems have been extensively studied [59]. The basic idea is to express the new search directions recursively as linear combinations of the negative gradients and previous search directions. Different coefficients for the linear combination lead to different methods including Fletcher-Reeves, Polak-Ribiere, Hestenes-Stiefel, etc. The numerical tests reported in this paper use the CG+ Conjugate Gradient package of Liu, Nocedal, and Walz [39] with the Fletcher-Reeves version of nonlinear conjugate gradients (FR-CG). Additional tests (not reported here) have shown that FR performs slightly better than the other nonlinear conjugate gradient methods for the test problem under consideration.

We next discuss two optimization methods that use second order information in the form of Hessian-vector products. The first method is a version of nonlinear conjugate gradients that uses Hessian-vector products to compute search directions. The second is the Hessian-free Newton method, which approximately solves the Newton's equation using an iterative method that requires only Hessian-vector products.

5.4.1. Daniel's Nonlinear Conjugate Gradients. Daniel's nonlinear conjugate gradients method [26, 27, 28, 29] uses explicit Hessian-vector products in the calculation of the new search direction. This approach has been traditionally considered impractical for large scale optimization problems due to the need for second order information [42]. Since second order adjoints can provide Hessian-vector prod-

ucts efficiently we revisit Daniel's method and use it to solve the data assimilation problem (5.3).

We next describe Daniel's method and show how it can be efficiently implemented using a single forward and backward model run (during which both first and second order adjoints are computed). In the first step one computes the gradient via one first order adjoint model run, and initializes the product of the Hessian and the search direction by either running the second order adjoint model or by approximating the Hessian with the identity matrix:

Initialization (we have $x_0 = y^B$)	
0.1	Compute in a forward-backward run $g_0 = \left(\partial\Psi/\partial y\left(x_0\right)\right)^T$
0.2	Set $d_0 = -g_0$
0.3	Compute in another forward-backward run $v_0 = \partial^2\Psi/\partial y^2\left(x_0\right)\cdot d_0$ (or let $v_0 = d_0$)

For each iteration one constructs the one-dimensional quadratic model along the search direction, updates the point in state space, updates the gradient, the search direction, and the product between the Hessian and the search direction. The computational cost at each step is dominated by one forward-backward run with the gradient evaluated by first order adjoint and two Hessian-vector products evaluated by the second order adjoint. Note that two Hessian-vector products can be computed simultaneously in a single backward run, and computational savings are possible by reusing the LU decompositions.

For $k \geq 1$ (we have $x = x_k$, d_k, $g_k = \partial\Psi/\partial y(x_k)$, and $v_k = \partial^2\Psi/\partial y^2(x_k)\cdot d_k$)	
k.1	Find α_k via line-search such that $\Psi(x_k + \alpha d_k) \leq \Psi(x_k) + c\,\alpha\, g_k^T d_k)$
k.2	Update the solution: $x_{k+1} = x_k + \alpha_k d_k$
k.3	Compute in a single forward-backward run: $g_{k+1} = \left(\partial\Psi/\partial y\left(x_{k+1}\right)\right)^T$ $a_{k+1} = \partial^2\Psi/\partial y^2\left(x_{k+1}\right)\cdot g_{k+1}$ $b_{k+1} = \partial^2\Psi/\partial y^2\left(x_{k+1}\right)\cdot d_k$
k.4	Compute $\beta_k = \left(g_{k+1}^T v_k\right) / \left(d_k^T v_k\right)$ (which ensures that $d_{k+1}^T \partial^2\Psi/\partial y^2(x_k) d_k = 0$)
k.5	Update the search direction: $d_{k+1} = -g_{k+1} + \beta_k d_k$
k.6	Update the product: $v_{k+1} = \partial^2\Psi/\partial y^2\left(x_{k+1}\right)\cdot d_{k+1} = -a_{k+1} + \beta_k\, b_{k+1}$

Here we denote by x_k the vector of initial concentrations y^0 after k optimization iterations.

5.4.2. Hessian Free Newton. The minimization of (5.3) can be carried out in principle using Newton's method. With x_k denoting the value of the solution after k

Newton iteration the process is

$$\left(\frac{\partial^2 \Psi}{\partial y^2}(x_k)\right) \cdot \Delta x = \left(\frac{\partial \Psi}{\partial y}(x_k)\right)^T \tag{5.6}$$
$$x_{k+1} = x_k - \Delta x \ , \quad k = 0, 1, \cdots$$

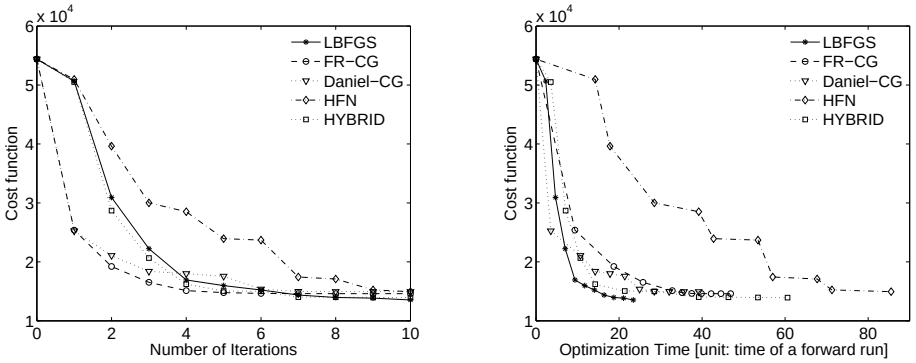
Each iteration requires to solve a linear system. The system matrix is the Hessian and the right hand side vector the gradient computed at the current iterate. Since the Hessian is a large symmetric matrix, a sensible approach is to solve the system using the linear conjugate gradients iterative method. The linear system solution (5.6) needs to be only as accurate as the solution of the nonlinear system. Therefore one can stop the conjugate gradient process after only a few iterations. Moreover, the conjugate gradients algorithm only requires matrix times vector products. The Hessian-vector products needed are computed by the second order adjoint. Because the full Hessian is not required the approach is called *Hessian-free Newton* (HFN).

Consequently each outer Newton iteration requires several inner iterations of the linear conjugate gradients to solve (5.6). Each of the inner iterations performs one forward integration of the forward and the tangent linear models (5.4), followed by one reverse integration of the first and second order adjoint models (5.5). The computational cost of each inner iteration is therefore relatively expensive.

For the numerical experiments we use the HYBRID code of Morales and Nocedal [56] to test the stand-alone HFN method. This code also implements an enriched optimization algorithm that allows to interlace L-BFGS and HFN iterations and use the information collected by one type of iteration to improve the performance of the other. We will refer to the enriched method as the "HYBRID" method. In the numerical experiments reported here we alternate five L-BFGS iterations with one HFN iteration.

5.4.3. Optimization Results. Data assimilation experiments use a 12 hours data assimilation window which starts at 12 GMT (8 EDT) on July 20^{th}, 2004. We asses the performance of five optimization methods used to minimize the cost function (5.3). L-BFGS-B and the Fletcher-Reeves Nonlinear Conjugate Gradients (FR-CG) methods require only first order derivative information. Daniel Nonlinear Conjugate Gradients (Daniel-CG), HFN and the HYBRID methods require second order derivative information. Since L-BFGS-B is considered the gold standard in variational data assimilation we will use its solution as a reference.

When solving real large-scale variational data assimilation problems with very expensive evaluations of the function, the gradient, and the Hessian-vector products the optimization process is typically not run to convergence. In practice the number of iterations is predefined (based on an estimate of the feasible computational time). Following this approach in our numerical experiments each method is allowed to take a fixed number of ten iterations. Each iteration of L-BFGS-B founds a new solution



(a) Decrease of the cost function vs. number of iterations

(b) Decrease of the cost function vs. scaled CPU time

FIG. 5.4. *Decrease of the cost function with (a) the number of iterations, and with (b) the scaled CPU time. The relative performance of five different optimization methods is shown.*

point ("NEW_X"), and can use multiple model runs during the line search. For HFN we consider ten outer (Newton) iterations; each iteration founds a new solution point, and can use multiple inner (linear conjugate gradients) iterations. For the Fletcher-Reeves and the Daniel nonlinear conjugate gradients each iteration produces a new solution point.

The decrease of the cost function with the number of iterations is reported in Figure 5.4(a). All methods are able to drive the cost function from a value of about 55,000 down to about 14,000 after ten iterations. Beyond ten iterations further decrease in the cost function is small, indicating that all solutions have approached the optimum.

The decrease of the cost function versus the computational time is reported in Figure 5.4(b). On the abscissa we use scaled time units, where one unit is the CPU time of one forward run (with only the nonlinear model, and without any derivative calculations). The cost of each optimization is estimated based on the number of model runs and the relative timings for the first and second order adjoint calculations given in Table 5.3. The results in Figure 5.4(b) indicate that that L-BFGS-B method is the most efficient method. Daniel's CG method performs better than FR-CG, especially during the first few iterations. HFN converges toward the solution in a small number of outer iterations, but at the cost of many inner iterations. This makes the total computational cost of HFN to be the highest among all methods. The HYBRID method starts with five L-BFGS iterations, and during them its performance is similar to that of L-BFGS-B. After the HFN step the HYBRID method becomes slightly slower than L-BFGS-B.

The quality of each optimization solution is measured by the norm of the gradient

of the cost function, and by the R^2 correlation factor and root mean square (RMS) difference between the observations and the model predictions (when the model is initialized with the solution of the optimization process $y^0 = y^a$). The R^2 correlation factor and the RMS difference between the set of all individual observations z_i, $i = 1, \cdots, m$ in the assimilation window and the set of all corresponding model predictions $(Dy)_i$ are defined as follows

$$R^2(Dy, z) = \frac{\left(m \sum_{i=1}^m (Dy)_i \, z_i - \sum_{i=1}^m (Dy)_i \sum_{i=1}^m z_i\right)^2}{\left(m \sum_{i=1}^m (Dy)_i^2 - \left(\sum_{i=1}^m (Dy)_i\right)^2\right)\left(m \sum_{i=1}^m z_i^2 - \left(\sum_{i=1}^m z_i\right)^2\right)},$$

$$RMS(Dy, z) = \sqrt{\frac{1}{m} \sum_{i=1}^m \Big((Dy)_i - z_i\Big)^2}.$$

Table 5.5 shows the norm of the gradient of the cost function, the R^2 correlation factor, and the RMS difference between observations and model predictions when initialized with the background and with each of the optimization solutions. A good solution has a small norm of gradient, a small RMS difference between observations and model predictions, as well as a large correlation coefficient between observations and model predictions.

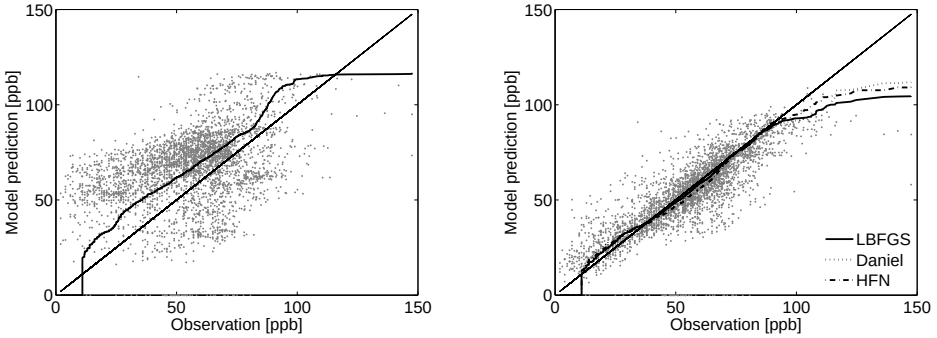
The results in Table 5.5 indicate that all optimized solutions show a considerable improvement from the background state. Model predictions are much closer to the observations (in both the R^2 and the RMS metrics) when the simulation is initialized with any of the optimal solutions. The norm of gradient indicates that the L-BFGS-B and Daniel solutions are the closest to the optimum, while the HFN solution is the farthest. Overall the L-BFGS-B solution is slightly better than the others, and considering the computational time we conclude that L-BFGS-B performs best on the data assimilation problem under consideration.

TABLE 5.5

The quality of different optimized solutions measured by the norm of gradient, the correlation coefficient, and root mean square distance between model predictions and observations.

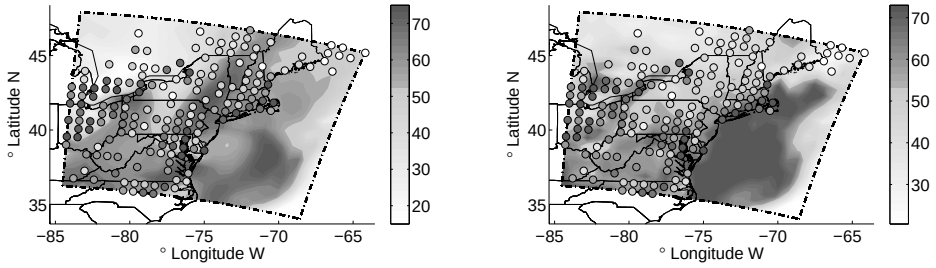
	BG	L-BFGS	FR-CG	Daniel-CG	HFN	HYBRID
$\|\partial\Psi/\partial y\|$	4147.38	493.09	757.70	490.61	795.83	559.46
RMS	24.76	11.94	12.67	12.68	12.93	12.24
R^2	0.15	0.68	0.65	0.64	0.64	0.67

The scatter and quantile-quantile plots of Figure 5.5 also illustrate how the correlation between model predictions (represented on the y-axes) and the observations (represented on the x-axes) is improved through data assimilation. The background case shown in Figure 5.5(a) has a correlation coefficient $R^2 = 0.15$. The spread of the scatter plot is large, and the quantile-quantile plot shows a visible bias. Figure



(a) Background ($R^2 = 0.15$) (b) Assimilated ($R^2 \geq 0.64$)

FIG. 5.5. *Scatter plots and quantile-quantile plots of model-observations agreement: (a) before data assimilation, and (b) after data assimilation. The solutions obtained with different optimization methods show a similar agreement with observations.*



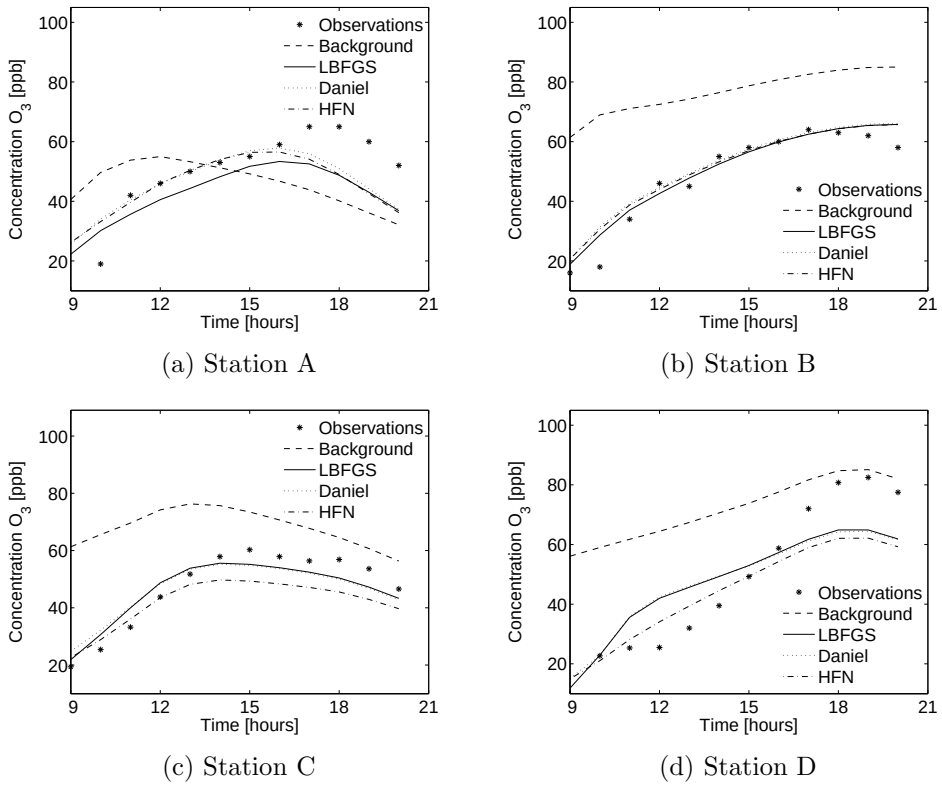
(a) Original model prediction (b) Model prediction after assimilation

FIG. 5.6. *Ground level ozone distribution in Northeastern U.S. at 3pm EDT on July 20, 2004. (a) Before data assimilation, and (b) after data assimilation.*

5.5(b) shows the scatter and the quantile-quantile plots after data assimilation. The spread of the scatter plot is much smaller, which is quantified by the larger correlation coefficients between model predictions and observations ($R^2 \geq 0.6$). The solutions are started from the optimized initial conditions computed with the L-BFGS-B, Daniel-CG, and HFN methods. We see that L-BFGS-B, Daniel-CG, and HFN quantile-quantile plots overlap with the ideal line for most of the range of values, showing a considerable decrease in model results bias.

The ground level ozone fields at 3pm EDT of July 20, 2004 using L-BFGS-B solutions as initial conditions are shown in Figure 5.6. Visually there is a better agreement between model predictions and observations after assimilation, especially near the West boundary.

To show the time evolution of ground level ozone concentrations we select four



(a) Station A (b) Station B

(c) Station C (d) Station D

FIG. 5.7. *Time series (in EDT time) of ozone concentrations at the four selected stations. The model runs are initialized with the background ozone concentration and with the assimilated ozone fields obtained with L-BFGS-B, HFN, and Daniel-CG optimization methods. The ozone time series after data assimilation are much closer to observations.*

AirNow stations A–D, shown in Figure 5.2(b). The ozone time series initialized using the background and different optimized solutions at these four stations are illustrated in Figure 5.7. The ozone time series after data assimilation are much closer to observations than the non-assimilated time series, which indicates the improvement in model predictions after data assimilation. The time series initialized with different optimized conditions are very similar, indicating that all optimization methods find the same optimum point.

5.5. Uncertainty quantification. When the model is linear, and the background and observation uncertainties are Gaussian, the aposteriori probability density of the initial state is also Gaussian (with mean y^a and covariance $P^a(t^0)$, i.e., $y^0 \in \mathcal{N}\left(y^a, P^a(t^0)\right)$). In this case the cost function (5.3) is quadratic and represents the negative logarithm of the aposteriori Gaussian probability density function

$$\Psi\left(y^0\right) = -\log\ p^a\left(y^0\right)\ , \quad p^a\left(y^0\right) = \text{const}\times\exp\Big(-\frac{1}{2}\left(y^0 - y^a\right)^T\ \left(P^a\left(t^0\right)\right)^{-1}\ \left(y^0 - y^a\right)\Big)\ .$$

It is easy to see that the Hessian of the cost function equals the inverse of the aposteriori covariance matrix, $\partial^2\Psi/\partial y^2 = \left(P^a\left(t^0\right)\right)^{-1}$.

For nonlinear models with non-Gaussian uncertainty probability densities one solves the nonlinear minimization problem (5.3) to obtain the analyzed initial condition

$$y^a = \arg\min_{y^0} \ \Psi\left(y^0\right) \ .$$

The Hessian of the cost function (5.3), evaluated at the optimal initial condition y^a, offers an approximation of the aposteriori covariance matrix of the uncertainty in the analyzed initial conditions:

$$P^a\left(t^0\right) \approx \left(\frac{\partial^2\Psi}{\partial y^2}(y^a)\right)^{-1} \ . \tag{5.7}$$

We expect this to be a good approximation if the errors are relatively small, if their propagation in time obeys the tangent linear model, and if the distribution of uncertainty is not far from Gaussian.

Our goal is now to characterize the aposteriori errors, i.e., to quantify the uncertainty in the initial state y^a after the assimilation of observations. For this let (λ_i^P, v_i), $i = 1, \cdots, n$, be the eigenvalue-eigenvector pairs of the aposteriori covariance matrix $P^a\left(t^0\right)$. The eigenvectors are orthogonal to each other (because of symmetry) and have norm one. Moreover, all the eigenvalues are non-negative $\lambda_i^P \geq 0$.

Under the Gaussian assumption the aposteriori error in the initial condition is a Gaussian random process which can be described in terms of the eigenvalues and eigenvectors of the covariance matrix

$$Err = y^0 - y^a = \sum_{i=1}^{n} \xi_i \sqrt{\lambda_i^P}\, v_i \ , \quad \xi_i \in \mathcal{N}(0,1) \ , \tag{5.8}$$

where ξ_i are independent Gaussian random variables. The principal components $\sqrt{\lambda_i^P}\, v_i$ of the aposteriori error are along the directions of the largest eigenvalues of the covariance matrix. According to (5.7) the largest eigenvalues of the covariance matrix are (approximated by) the inverses of the smallest Hessian eigenvalues $\lambda_i^P = 1/\lambda_i^H$, while the corresponding eigenvectors are the same. To characterize the aposteriori error we estimate its principal components (5.8) from the Hessian eigenvalues and eigenvectors as follows.

The largest five and the smallest five eigenvalues of the Hessian of the cost function are computed using the ARPACK package [49]. The simulation is initialized with the optimal solution y^a of the data assimilation problem given by L-BFGS-B. The second adjoint model is used to provide the Hessian-vector products required by ARPACK. These eigenvalues are reported in Table 5.6. The inverses of the Hessian eigenvalues approximate the eigenvalues of the aposteriori covariance matrix $P^a(t^0)$

and are also reported in Table 5.6. These eigenvalues represent variances of the principal components (5.8) in the units (molecules of O_3 per cm^3 of air)^2. The square root of the covariance eigenvalues represent the standard deviations of each of the principal components; we report the standard deviations in the more convenient units of parts-per-billion (ppb). The conversion is done by dividing the concentration to the ground level air density ($\rho = 2.4 \times 10^{19}$ molecules/cm^3) and multiplying the results by 10^9. We see that the error is dominated by the first principal component (along which the standard deviation is 47 ppb).

TABLE 5.6

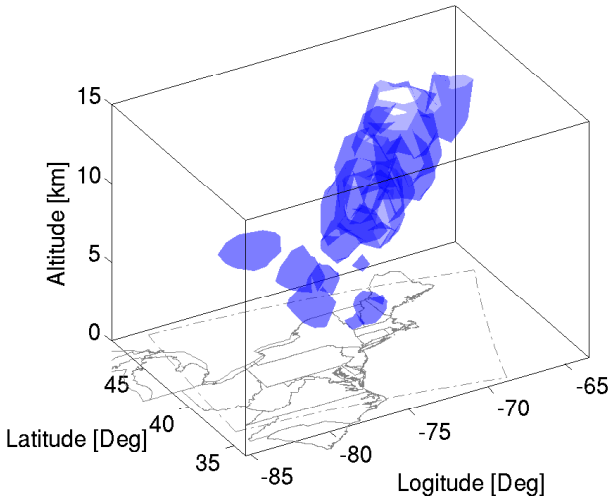
The smallest and largest five eigenvalues of the Hessian and the corresponding eigenvalues of the aposteriori covariance matrix.

	First	Second	Third	Fourth	Fifth
λ^H_{small} (mlc/cm^3)$^{-2}$	7.54×10^{-25}	1.15×10^{-23}	4.04×10^{-23}	8.47×10^{-23}	1.42×10^{-22}
λ^P_{large} (mlc/cm^3)^2	1.33×10^{24}	8.70×10^{22}	2.48×10^{22}	1.18×10^{22}	7.04×10^{21}
$\sqrt{\lambda^P_{\text{large}}}$ (ppb)	47	12	7	4	3
λ^H_{large} (mlc/cm^3)$^{-2}$	4.17×10^{-22}	3.79×10^{-22}	3.34×10^{-22}	2.76×10^{-22}	2.12×10^{-22}
λ^P_{small} (mlc/cm^3)^2	2.40×10^{21}	2.64×10^{21}	3.00×10^{21}	3.62×10^{21}	4.72×10^{21}
$\sqrt{\lambda^P_{\text{small}}}$ (ppb)	2.04	2.14	2.28	2.51	2.86

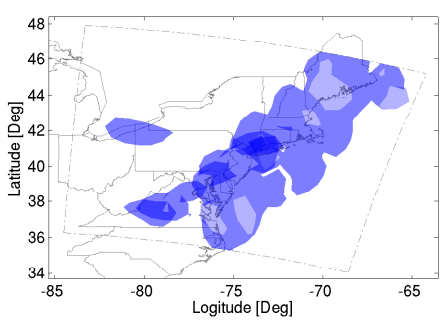
To visualize the spatial distribution of the error we plot the 2 ppb isosurface of the first principal error component $\sqrt{\lambda^P_1} v_1$ in Figure 5.8. The unit conversion from molecules/cm^3 to ppb is done using the appropriate air density in each vertical layer. The principal component the error is located at high altitudes. This can be explained by the dense observational network at the ground level used in this data assimilation study, see Figure 5.2; the assimilation of these observations reduces the uncertainty in ozone initial concentrations at low altitudes. In contrast the number of observations at high altitudes is low and considerable uncertainty remains after data assimilation. One possible conclusion is that more high altitude observations are needed to further reduce the global level of uncertainty.

5.6. Directions of important error growth. We now look into the problem of how uncertainties propagate forward in time through the model. Specifically we want to estimate which perturbations at the initial time grow to have the largest impact on the solution accuracy at the final time. These "directions of maximal error growth" [5] are important in several applications. First, in order to have an accurate forecast (an accurate solution at the final time) one needs to reduce the uncertainty in the initial state along these directions [51]. New observations added to increase the accuracy of the simulation (through data assimilation) are most useful if placed along these directions [50]. Next, in a Monte Carlo approach, a small ensemble of runs can represent well the uncertainty in a large-dimensional system if it is initialized with perturbations along the directions of maximal error growth [4].

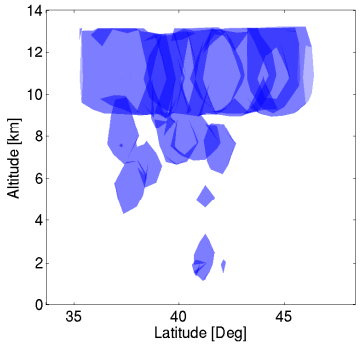
Following (2.4)–(2.5) we denote by $\mathcal{N}'$, $\mathcal{N}'^*$ the tangent linear and the adjoint



(a) 3D View, 2 ppb Error



(b) Top View, 2 ppb Error



(c) East View, 2 ppb Error

FIG. 5.8. *First principal component of the error in the initial ozone filed. The 2 ppb error isosurface is shown in (a) 3D View, (b) Top View, and (c) East View.*

model solution operators on the interval $[t^0, t^{\mathrm{F}}]$. The model is initialized at t^0 with the optimal state y^a (for which the error covariance is $P^a(t^0)$). Perturbations (small errors) in the initial conditions δy^0 propagate forward in time according to the tangent linear model (3.16), and grow at the final time to

$$\delta y(t^{\mathrm{F}}) = \mathcal{N}' \delta y^0 \ . \tag{5.9}$$

The error covariance matrix $P^a(t^0)$ evolves into the forecast error covariance matrix at t^{F}

$$P^f\left(t^{\mathrm{F}}\right) = \mathcal{N}' \cdot P^a\left(t^0\right) \cdot \mathcal{N}'^* \ .$$

The principal components of the forecast uncertainty (uncertainty at the final time) are along the dominant eigenvectors of the forecast error covariance matrix $P^f(t^{\mathrm{F}})$. We want to find the directions δy^0 at the initial time which grow through (5.9) into the dominant eigenvectors of P^f at the final time. We have that:

$$P^f(t^{\mathrm{F}})\,\delta y(t^{\mathrm{F}}) = \theta_{\max}\,\delta y(t^{\mathrm{F}}) \Leftrightarrow \left(\mathcal{N}' \cdot P^a(t^0) \cdot \mathcal{N}'^*\right) \mathcal{N}'\,\delta y^0 = \theta_{\max}\,\mathcal{N}'\,\delta y^0$$
$$\Leftrightarrow \mathcal{N}'^*\,\mathcal{N}'\,\delta y^0 = \theta_{\max}\,\left(P^a(t^0)\right)^{-1}\,\delta y^0$$

The inverse covariance matrix can be approximated by the Hessian of the cost function (5.7). We see that the dominant eigenvectors in this case are the solution of the generalized eigenvalue problem

$$\mathcal{N}'^*\,\mathcal{N}'\,\delta y^0 = \theta_{\max}\,\left(\frac{\partial^2 \Psi}{\partial\,(y^0)^2}(y^a)\right)\,\delta y^0 \tag{5.10}$$

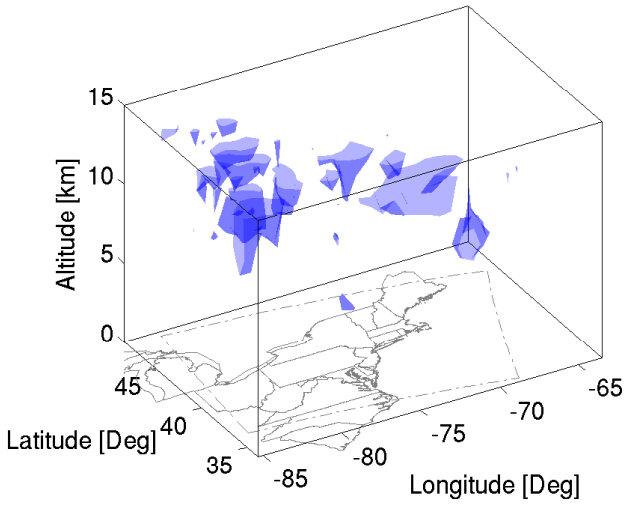
The generalized eigenvectors in (5.10) are called *Hessian singular vectors* in the data assimilation literature [4]. The matrix times vector products $\mathcal{N}'^*\,\mathcal{N}'\,\delta y^0$ needed to evaluate the left hand side are computed by one forward integration of the TLM ($\delta y(t^{\mathrm{F}}) = \mathcal{N}'\,\delta y^0$) followed by one backward integration of the adjoint ($\mathcal{N}'^*\,\delta y(t^{\mathrm{F}})$). The adjoint variable is initialized with the final value of the TLM integration. The Hessian times vector products needed to evaluate the right hand side are obtained by the second order adjoint.

For numerical experiments we run the STEM model for 8 hours. The simulation is initialized with the optimal solution of the data assimilation problem given by L-BFGS-B. The dominant generalized eigenvalues (5.10) are computed using the JDQZ package which implements a Jacobi-Davidson algorithm [68]. Table 5.7 shows the largest five generalized eigenvalues (5.10). We see that the fifth generalized eigenvalues is two orders of magnitude smaller than the first. few directions at the initial time have a large impact on the final time uncertainty. Figure 5.9 presents the Hessian singular vector associated with the largest generalized eigenvalue in Table 5.7. Most of the area where initial perturbations have a large impact is at high altitudes, which is not surprising given that most of the uncertainty in the ozone field is at high altitudes.

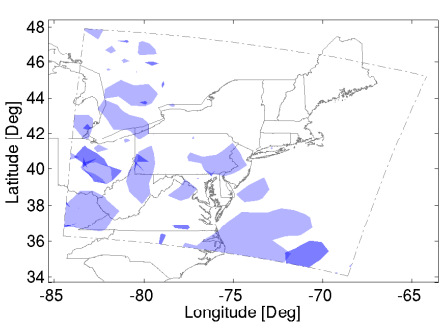
TABLE 5.7
The largest five Hessian singular eigenvalues.

	First	Second	Third	Fourth	Fifth
θ	0.16×10^{-15}	0.12×10^{-15}	0.66×10^{-16}	0.59×10^{-16}	0.17×10^{-17}

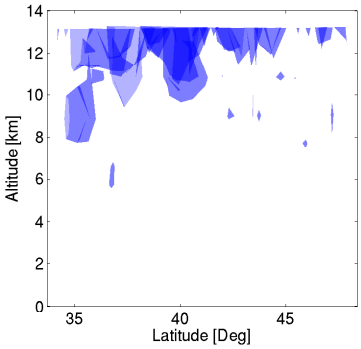
6. Conclusions. In this paper we discuss the computation of second order adjoints for stiff systems and their application in chemical transport modeling. First order adjoints allow to efficiently calculate the derivative of a cost functional (defined



(a) 3D View



(b) Top View



(c) East View

FIG. 5.9. *The 0.02 isosurface of the dominant Hessian singular vector: (a) 3D view, (b) top view, and (c) East view.*

on the model output) with respect to a large number of model parameters. Second order adjoints allow to efficiently calculate products between the Hessian of the cost functional and user defined vectors.

One important component of chemical transport models is the solution of stiff chemical kinetics. We derive the second order discrete adjoint formulations for three classes of stiff numerical solvers: fully implicit Runge Kutta, singly diagonally-implicit Runge Kutta, and Rosenbrock methods. For each we discuss in detail efficient implementation aspects which are based on reusing the expensive LU decompositions. Specifically the tangent linear model calculations "piggyback" the forward model calculations and use the same LU decompositions. Similarly the second order adjoint

calculations "piggyback" the first order adjoint calculations and reuse the same Jacobians and LU decompositions.

The other important component of chemical transport models are the convection and diffusion processes. The discrete second order adjoints for the transport solution use the same routines as the first order adjoints. This is due to the linear finite difference discretization of the convection and diffusion terms used in the model under consideration.

The first application of second order adjoints is to extend the validity range of sensitivity analysis to larger perturbations. We illustrate how in a nonlinear chemical box model the change in the final peroxy acetyl nitrate concentration due to changes in initial NO_X concentrations is predicted poorly by the first order sensitivity analysis. The prediction is considerably improved with second order sensitivity analysis. Second order adjoints can be useful in chemical transport modeling to better represent the sensitivity of a given receptor with respect to changes in initial conditions, emissions, meteorological conditions, etc. It can be useful to quantify the sensitivity of adjoint variables (which give areas that influence a given receptor) to other model parameters like the wind fields etc. Finally, the sensitivity of the optimal solution in data assimilation with respect to changes in observations, in the initial conditions, and in model parameters, can also be computed using second order adjoints.

The use of the second order adjoints in the optimization process for chemical data assimilation is numerically investigated. We consider two methods that require Hessian-vector products in addition to gradient information: the Hessian free Newton method and the Daniel nonlinear conjugate gradients. For reference we employ two methods that require only first order derivative information: the Fletcher-Reeves nonlinear conjugate gradients and the limited memory BFGS quasi-Newton method. L-BFGS is considered the gold standard in data assimilation. We also consider a hybrid method that interlaces L-BFGS and HFN iterations. Each iteration of the methods that require Hessian-vector products is more expensive than an iteration of methods that require only gradient information; therefore the question arises whether the computation of the second order information pays off during the optimization process. For the test problem under consideration L-BFGS is still the most effective method, followed closely by Daniel's nonlinear conjugate gradients. Hessian free Newton converges in a small number of iterations but the overall computational cost is relatively large. Until recently the use of second order information in large scale data assimilation problems was considered prohibitive. Methods that require only gradient information have been studied extensively, and the software that implements them is quite mature. In the authors opinion the possibility to compute efficiently Hessian-vector products via second order adjoint modeling opens the door for new research to develop numerical optimization algorithms that can use effectively this information. Future work will be devoted to developing new optimization methods that can make

efficient use of Hessian-vector products.

The Hessian evaluated at the optimal initial condition provides an approximation of the inverse aposteriori error covariance matrix. The eigenvectors of the Hessian associated with its smallest eigenvalues approximate the principal components of the aposteriori error field. This allows to estimate the remaining errors in the distribution of pollutants after data assimilation. In this paper we illustrate this uncertainty quantification method and compute the first five principal components of the error in the ozone field. The data assimilation test problem under consideration incorporates observations from a dense network at ground level; after data assimilation the largest levels of uncertainty are at high altitudes, where the observational network is sparse.

Some perturbations at the initial time grow to have the largest impact on the solution accuracy at the final time. These "most important directions of error growth" can be estimated using second order information. Specifically the Hessian singular vectors are those directions at the initial time which grow into the dominant eigenvectors of the covariance matrix at the final time. We illustrate their computation via the solution of a generalized eigenvalue problem, where the right hand side matrix is the Hessian of the cost function. For our test problem the most important initial time perturbations are also at high altitudes.

The efficient calculation of second order adjoints for three-dimensional atmospheric chemistry and transport models is demonstrated in this paper. While the cost of a first order adjoint computation is slightly over two times the cost of a forward simulation, the cost of a second order adjoint calculation is three and a half times the cost of a forward model run. The availability of Hessian-vector products opens the door for new analyses with chemical transport models. It allows to extend the validity of sensitivity analysis to large perturbations of the parameters. A quantification of uncertainty after data assimilation becomes possible. The Hessian-vector products allow to estimate the most important directions of error growth. Second order information is also useful in the large scale numerical optimization routines for data assimilation, but more work is needed to make these optimization methods competitive with quasi-Newton methods.

Acknowledgments. The authors thank professor Henk van der Vorst for sharing the JDQZ source code. The authors thank dr. Tianfeng Chai and professor Gregory R. Carmichael of the University of Iowa for providing the ICARTT computational setting for the data assimilation experiments. This work was supported by the National Science Foundation through the awards NSF CAREER ACI 0093139, NSF ITR AP&IM 0205198, and NSF CCF 0635194, by NASA through the award AIST-2005, and by the Houston Advanced Research Center through the award H-59/2005.

Appendix A. The ODE Model, the Jacobian, and the Hessian.

In this paper we consider all vectors to be column vectors. Gradients of scalar

functions are by default row vectors. An upper script $(\cdot)^T$ denotes the transposition operator.

The first and second derivatives of a scalar function are

$$\Psi\colon \mathbb{R}^n \to \mathbb{R} \quad \Rightarrow \quad \frac{\partial \Psi}{\partial y} = \left[\frac{\partial \Psi}{\partial y_1}, \cdots, \frac{\partial \Psi}{\partial y_n}\right] \quad \text{and} \quad \frac{\partial^2 \Psi}{\partial y^2} = \begin{bmatrix} \frac{\partial^2 \Psi}{\partial y_1^2} & \cdots & \frac{\partial^2 \Psi}{\partial y_1 \partial y_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi}{\partial y_n \partial y_1} & \cdots & \frac{\partial^2 \Psi}{\partial y_n^2} \end{bmatrix}$$

The Jacobian of a multidimensional vector function is represented as

$$h\colon \mathbb{R}^n \to \mathbb{R}^m\ , \quad h(y) = \begin{bmatrix} h_1\left(y_1 \cdots y_n\right) \\ \vdots \\ h_m\left(y_1 \cdots y_n\right) \end{bmatrix} \quad \Rightarrow \quad \frac{\partial h}{\partial y} = \begin{bmatrix} \frac{\partial h_1}{\partial y_1} & \cdots & \frac{\partial h_1}{\partial y_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m}{\partial y_1} & \cdots & \frac{\partial h_m}{\partial y_n} \end{bmatrix}$$

Consider a coupled system of stiff nonlinear differential equations which constitute the *forward model*

$$\frac{dy}{dt} = f(t, y)\ , \qquad y\left(t^0\right) = y^0\ , \qquad t^0 \le t \le t^{\mathrm{F}}\ .$$

The Jacobian of the time derivative function is

$$J_{i,j}(t, y) = \frac{\partial f_i\left(t, y\right)}{\partial y_j}\ , \qquad 1 \le i, j \le n\ .$$

The Hessian contains second order derivatives of the time derivative functions. More exactly, the Hessian is a 3-tensor such that

$$H_{i,j,k}\left(t, y\right) = \frac{\partial J_{i,j}\left(t, y\right)}{\partial y_k} = \frac{\partial^2 f_i\left(t, y\right)}{\partial y_j\, \partial y_k} = \frac{\partial^2 f_i\left(t, y\right)}{\partial y_k\, \partial y_j} = H_{i,k,j}\left(t, y\right)\ , \qquad 1 \le i, j, k \le n\ .$$

For each component i of the ODE derivative function there is a Hessian matrix $H_{i,:,:}$.

The Hessian allows to conveniently express the derivatives of the Jacobian times a user vector:

$$\begin{aligned} \frac{\partial}{\partial y}\left[J(t, y) \cdot u\right] &= \left(\frac{\partial}{\partial y_j}\left[J(t, y) \cdot u\right]_i\right)_{i,j} = \left(\frac{\partial}{\partial y_j}\left[\sum_{m=1}^{n} J_{i,m}(t, y)\, u_m\right]\right)_{i,j} \\ &= \left(\sum_{m=1}^{n} \frac{\partial J_{i,m}(t, y)}{\partial y_j}\, u_m\right)_{i,j} = \left(\sum_{m=1}^{n} H_{i,m,j}(t, y)\, u_m\right)_{i,j} \\ &= \left(\sum_{m=1}^{n} H_{i,j,m}(t, y)\, u_m\right)_{i,j} \\ &= H(t, y) \cdot u \end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial y}\left[J^T(t,y)\cdot u\right] &= \left(\frac{\partial}{\partial y_j}\left[J^T(t,y)\cdot u\right]_i\right)_{i,j} = \left(\frac{\partial}{\partial y_j}\left[\sum_{m=1}^{n} J_{m,i}(t,y)\,u_m\right]\right)_{i,j} \\
&= \left(\sum_{m=1}^{n}\frac{\partial J_{m,i}(t,y)}{\partial y_j}\,u_m\right)_{i,j} = \left(\sum_{m=1}^{n} H_{m,i,j}(t,y)\,u_m\right)_{i,j} \\
&= \left(\sum_{m=1}^{n} u_m\,H_{m,i,j}(t,y)\right)_{i,j} \\
&= u^T\cdot H(t,y)
\end{aligned}$$

For any vectors $u, v \in \mathbb{R}^n$ we have that

$$\begin{aligned}
\frac{\partial}{\partial y}\left[J(t,y)\cdot u\right]\cdot v = (H(t,y)\cdot u)\cdot v &= \sum_{j,m=1}^{n} H_{i,j,m}(t,y)\,u_m\,v_j \\
&= \sum_{j,m=1}^{n} H_{i,m,j}(t,y)\,v_j\,u_m = (H(t,y)\cdot v)\cdot u \\
&= \frac{\partial}{\partial y}\left[J(t,y)\cdot v\right]\cdot u
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial}{\partial y}\left[J^T(t,y)\cdot u\right]\cdot v = \Big(u^T\cdot H(t,y)\Big)\cdot v &= \sum_{j,m=1}^{n} u_m\,H_{m,i,j}(t,y)\,v_j \\
&= \sum_{m=1}^{n}\Big(H(t,y)\cdot v\Big)_{m,i}\,u_m = \Big(H(t,y)\cdot v\Big)^T\cdot u
\end{aligned}$$

REFERENCES

[1] Alekseev A.K. and Navon I.M. On estimation of temperature uncertainty using the second order adjoint problem. *International Journal of Computational Fluid Dynamics*, 16(2):113–117, 2002.

[2] A. Alekseev and I.M. Navon. The analysis of an ill-posed problem using multiscale resolution and second order adjoint techniques. *Computer Methods in Applied Mechanics and Engineering*, 190(15–17):1937–1953, 2001.

[3] A.K. Alekseev and I.M. Navon. Comparison of advanced large–scale minimization algorithms for the solution of inverse problems, 2002. Technical Report.

[4] J. Barkmeijer, R. Buizza, and T.N. Palmer. 3D-Var Hessian singular vectors and their potential use in the ECMWF Ensemble Prediction System. *Quarterly Journal of the Royal Meteorological Society*, 125:2333–2351, 1999.

[5] J. Barkmeijer, R. Buizza, T.N. Palmer, K. Puri, and J.F. Mahfouf. Tropical singular vectors computed with linearized diabatic physics. *Quarterly Journal of the Royal Meteorological Society*, 127:685–708, 2001.

[6] J. Barkmeijer, M. van Gijzen, and F. Bouttier. Singular vectors and estimates of the analysis error covariance metric. *Q. J. R. Meteor. Soc.*, 124(549):1695–1713, 1998.

[7] Ozyurt B.D. and Barton P.I. Application of targeted automatic differentiation to large scale dynamic optimization. Lecture Notes in Computational Science and Engineering, pages 235–247. Springer, 2005.

[8] Ozyurt B.D. and Barton P.I. Cheap second order directional derivatives of stiff ode embedded functionals. *SIAM Journal on Scientific Computing*, 26(5):1725–1743, 2005.

[9] Ozyurt B.D. and Barton P.I. Large-scale dynamic optimization using the directional second-order adjoint method. *Industrial Engineering and Chemistry Research*, 44(6):1804–1811, 2005.

[10] J.D. Beley, S. Garreau, F. Thevenon, and M. Masmoudi. Application of higher order derivatives to parameterization. pages 335–341, 2002.

[11] R.H. Byrd, P. Lu, and J. Nocedal. A Limited-Memory Algorithm For Bound–Constrained Optimization . *SIAM Journal on Scientific Computing*, 16(6):1190–1208, 1995.

[12] G.R. Carmichael, T. Chai, A. Sandu, E.M. Constantinescu, and D. Daescu. Predicting air quality. *Journal of Computational Physics*, 2006. Submitted.

[13] G.R. Carmichael, L.K. Peters, and T. Kitada. A second generation model for regional-scale transport/ chemistry/ deposition. *Atmospheric environment*, 20:173–188, 1986.

[14] G.R. Carmichael, Y. Tang, G. Kurata, I. Uno, D. Streets, J.H. Woo, H. Huang, J. Yienger, B. Lefer, R. Shetter, D. Blake, E. Atlas, A. Fried, E. Apel, F. Eisele, C. Cantrell, M. Avery, J. Barrick, G. Sachse, W. Brune, S. Sandholm, Y. Kondo, H. Singh, R. Talbot, A. Bandy, D. Thorton, A. Clarke, and B. Heikes. Regional-scale Chemical Transport Modeling in Support of the Analysis of Observations obtained During the Trace-P Experiment. *Journal of Geophysical Research*, 108(D21 8823):10649–10671, 2003.

[15] W.P.L. Carter. Implementation of the SAPRC-99 Chemical Mechanism into the Models-3 Framework. Technical report, Report to the United States Environmental Protection Agency, January 2000.

[16] W.P.L. Carter. Documentation of the SAPRC-99 Chemical Mechanism for VOC Reactivity Assessment. Technical Report No. 92-329, and 95-308, Final Report to California Air Resources Board Contract, May 2000.

[17] T. Chai, G.R. Carmichael, A. Sandu, Y. Tang, and D.N. Daescu. Chemical data assimilation with trace-p flight measurements. *Journal of Geophysical Research*, 111(D02301), 2006.

[18] T. Chai, G.R. Carmichael, Y. Tang, A. Sandu, M. Hardesty, P. Pilewskie, S. Whitlow, E.V. Browell, M.A. Avery, V. Thouret, P. Nedelec, J.T. Merrill, , and A.M. Thomson. Four Dimensional Data Assimilation Experiments with ICARTT (International Consortium for Atmospheric Transport and Transformation) Ozone Measurements. *Journal of Geophysical Research*, 2006. Submitted.

[19] I. Charpentier, N. Jakse, and F. Veerse. Second order exact derivatives to perform optimization on self-consistent integral equations problems. pages 189–195, 2002.

[20] P. Courtier, J.-N. Thepaut, and A. Hollingsworth. A strategy for operational implementation of 4D-Var, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120:1367–1387, 1994.

[21] D. Daescu and I.M. Navon. Efficiency of a pod-based reduced second-order adjoint model in 4d-var data assimilation. *International Journal of Numerical Methods in Fluids*, 53:985–1004, 2007.

[22] D. Daescu, A. Sandu, and G.R. Carmichael. Direct and Adjoint Sensitivity Analysis of Chemical Kinetic Systems with KPP: II – Numerical Validation and Applications. *Atmospheric Environment*, 37:5097–5114, 2003.

[23] D.N. Daescu and I.M. Navon. An analysis of a hybrid optimization method for variational data assimilation. *International Journal of Computational Fluid Dynamic*, 17(4):299–306, 2003.

[24] D.N. Daescu and I.M. Navon. Adaptive observations in the context of 4d-var data assimilation.

Meteorology and Atmospheric Physics, 84(4):205–226, 2004.

[25] V. Damian, A. Sandu, M. Damian, F. Potra, and G.R. Carmichael. The kinetic preprocessor kpp – a software environment for solving chemical kinetics. *(Computers and Chemical Engineering)*, 26:1567–1579, 2002.

[26] J.W. Daniel. The conjugate gradient method for linear and nonlinear operator equations. *SIAM Journal on Numerical Analysis*, 4(1):10–26, mar 1967.

[27] J.W. Daniel. The conjugate gradient method for linear and nonlinear operator equations. *SIAM Journal on Numerical Analysis*, 4:10–26, 1967.

[28] J.W. Daniel. Convergence of the conjugate gradient method with computationally convenient modifications. *Numerische Mathematik*, 10:125–131, 1967.

[29] J.W. Daniel. A correction concerning the convergence rate for the conjugate gradient method. *SIAM Journal on Numerical Analysis*, 7:277–280, 1970.

[30] F.X. Le Dimet and O. Talagrand. Variational algorithms for analysis and assimilation of meteorological observations. *Tellus*, 38 A:97–110, 1986.

[31] H. Elbern and H. Schmidt. Ozone episode analysis by 4D-Var chemistry data assimilation. *Journal of Geophysical Research*, 106(D4):3569–3590, 2001.

[32] H. Elbern, H. Schmidt, O. Talagrand, and A. Ebel. 4D-variational data assimilation with an adjoint air quality model for emission analysis. *Environmental Modeling and Software*, 15:539–548, 2000.

[33] Y.G. Evtushenko, E.S. Zasuhina, and V.I. Zubov. Fad method to compute second order derivatives. pages 327–333, 2002.

[34] LeDimet F., Ngodock H., Loung B., and Verron J. Sensitivity analysis in variational data assimilation. *J. Meteor. Soc. Japan*, 75(B):245–255, 1997.

[35] M. Fisher and D.J. Lary. Lagrangian four-dimensional variational data assimilation of chemical species. *Quarterly Journal of the Royal Meteorological Society*, 121:1681–1704, 1995.

[36] H. Flanders. Application of ad to a family of periodic functions. pages 319–326, 2002.

[37] International Consortium for Atmospheric Research on Transport and Transformation (ICARTT). ICARTT web site: `http://www.al.noaa.gov/ICARTT`, 2006.

[38] LeDimet F.X., Navon I., and Daescu D. Second order information in data assimilation. *Monthly Weather Review*, 130(3):629–648, 2002.

[39] J. C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, 2:21–42, 1992.

[40] R. Griesse and A. Walther. Towards Matrix-Free AD-Based Preconditioning of KKT Systems in PDE-Constrained Optimization. In *GAMM Annual Meeting 2005 - Luxembourg*, pages 47–50. PAMM, 2005.

[41] W. Hager. Runge-Kutta methods in optimal control and the transformed adjoint system. *Numerische Mathematik*, 87(2):247–282, 2000.

[42] W. W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, 16:170–192, 2005.

[43] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems.* Springer-Verlag, Berlin, 1993.

[44] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems.* Springer-Verlag, Berlin, 1991.

[45] A. Hakami, J.H. Seinfeld, T. Chai, Y. Tang, G.R. Carmichael, and A. Sandu. Adjoint sensitivity analysis of ozone non-attainment over the continental united states. *Environmental Science and Technology*, 2006.

[46] J. Hoefkens, M. Berz, and K. Makino. Efficient high-order methods for odes and daes. pages 343–348, 2002.

[47] W.H. Hundsdorfer and J.G. Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, volume 33 of *Springer Series in Computational Mathemat-*

ics. Springer Verlag, 2003.

[48] D. Lanser and J.G. Verwer. Analysis of operator splitting for advection-diffusion-reaction problems from air pollution modeling. Technical Report MAS-R9805, Centrum voor Wiskunde en Informatica, 1998.

[49] R. Lehoucq, K. Maschhoff, D. Sorensen, and C. Yang. Arpack software (parallel and serial). *URL: http://www.caam.rice.edu/software/ARPACK*.

[50] Palmer T. Leutbecher M., Barkmeijer J. and Thorpe A. Potential improvement of forecasts of two severe storms using targeted observations. *Q.J.R. Meteorol. Soc.*, 128(583):1641–1670, 2002.

[51] W. Liao, A. Sandu, T. Chai, and G.R. Carmichael. Total energy singular vector analysis for atmospheric chemical transport models. *Monthly Weather Review*, 134(9):2443–2465, 2006.

[52] D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[53] Z. Liu and A. Sandu. Analysis of Discrete Adjoints of Numerical Methods for the Advection Equation. International Journal on Numerical Methods for Fluids, 2007. Accepted.

[54] A.C. Lorenc. Analysis methods for numerical weather prediction. *Q.J.R. Meteorol. Soc.*, 1128:1177–1194, 1986.

[55] J.L. Morales. A Numerical Study of Limited Memory BFGS Methods. *Applied Mathematics Letters*, 15(4):481–487, 2002.

[56] J.L. Morales and J. Nocedal. Enriched Methods for large–scale unconstrained optimization. *Computational Optimization and Applications*, 21:143–154, 2002.

[57] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 24:773–782, 1980.

[58] J. Nocedal. *Large Scale Unconstrained Optimization*. The State of the Art in Numerical Analysis. Oxford University Press, 1997.

[59] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, 1999.

[60] F. Rabier, H. Jarvinen, E. Klinker, J.F. Mahfouf, and A. Simmons. The ECMWF operational implementation of four-dimensional variational assimilation. I: Experimental results with simplified physics. *Quarterly Journal of the Royal Meteorological Society*, 126:1148–1170, 2000.

[61] R.L. Raffard and C.J. Tomlin. Second order adjoint-based optimization of ordinary and partial differential equations with application to air traffic flow. In *2005 American Control Conference*. Portland, OR, USA., June 8-10, 2005.

[62] Lakshmivarahan S., Honda Y., and Lewis J.M. Second-order approximation to the 3dvar cost function: application to analysis/forecast. *Tellus*, 55(5):371–384, 2003.

[63] A. Sandu, D. Daescu, and G.R. Carmichael. Direct and Adjoint Sensitivity Analysis of Chemical Kinetic Systems with KPP: I – Theory and Software Tools. *Atmospheric Environment*, 37:5083–5096, 2003.

[64] A. Sandu, D. Daescu, G.R. Carmichael, and T. Chai. Adjoint sensitivity analysis of regional air quality models. *Journal of Computational Physics*, 204:222–252, 2005.

[65] A. Sandu, J.G. Verwer, J.G. Blom, E.J. Spee, G.R. Carmichael, and F.A. Potra. Benchmarking stiff ode solvers for atmospheric chemistry problems ii: Rosenbrock methods. *Atmospheric Environment*, 31:3459–3472, 1997.

[66] C. Sandu, A. Sandu, B. Chan, and M. Ahmadian. A response spectral decomposition approach for the control of nonlinear mechanical systems. *Journal for Heavy Vehicle Systems*, Acceprted, 2006.

[67] A.J. Segers. Data assimilation in atmospheric chemistry models using kalman filterin, 2002.

[68] G.L.G. Sleijpen, J.G.L. Booten, D.R. Fokkema, and H.A. van der Vorst. Jacobi-davidson type methods for generalized eigenproblems and polynomial eigenproblems. *B.I.T.*, 36(3):595–

633, 1996.

[69] B. Sportisse. An analysis of operator splitting techniques in the stiff case. *Journal of Computational Physics*, 161(1), 2000.

[70] O. Talagrand and P. Courtier. Variational assimilation of meteorological observations with the adjoint vorticity equation. Part I: Theory. *Quarterly Journal of the Royal Meteorological Society*, 113:1311–1328, 1987.

[71] Y. Tang, G.R. Carmichael, N. Thongboonchoo, T. Chai, L.W. Horowitz, R.B. Pierce, J.A. Al-Saadi, G. Pfister, J.M. Vukovich, M.A. Avery, G.W. Sachse, T.B. Ryerson, J.S. Holloway, E.L. Atlas, F.M. Flocke, R.J. Weber, L.G. Huey, J.E. Dibb, D.G. Streets, and W.H. Brune. The influence of lateral and top boundary conditions on regional air quality prediction: a multi-Scale study coupling regional and global chemical transport models. *SUBMITTED to Journal of Geophysical Research*, 2006.

[72] Wang Z., Navon I.M., Le Dimet F.X., and Zou X. The second order adjoint analysis: theory and applications. *Meteorology and Atmospheric Physics*, 50(1-3):3–20, 1992.

[73] Wang Z., Droegemeier K., and White L. The adjoint newton algorithm for large-scale unconstrained optimization in meteorology applications. *Computational Optimization and Applications*, 10(3):283–320, 1998.

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