Proc. IEEE Nonlinear Signal and Image Processing Workshop, E. Coyle and G. Adams, eds.
(IEEE, Piscataway, 1998; CD-ROM, ISBN: 0 876346 09 4)

INVERSION BASED ON COMPLEX COMPUTATIONAL SIMULATIONS

Kenneth M. Hanson, Gregory S. Cunningham, and Suhail S. Saquib

Los Alamos National Laboratory, MS P940
Los Alamos, New Mexico 87545, USA
{kmh,cunning}@lanl.gov, saquib@polaroid.com

ABSTRACT

Experimental data often can only be interpreted by means of a computational simulation that approximately models the physical situation. We will discuss techniques that facilitate application to complex, large-scale simulations of the standard approach to inversion in which gradient-based optimization is used to find the parameters that best match the data. The fundamental enabling techniques are adjoint differentiation to efficiently compute the gradient of an objective function with respect to all the variables of a simulation and relatively new gradient-based optimization algorithms. These techniques will be illustrated through the simulation of the time-dependent diffusion of infrared light through tissue, which has been used to perform optical tomography [1]. The techniques discussed have a wide range of applicability to modeling including the optimization of models to achieve a desired design goal.

Keywords: simulation, inversion, adjoint differentiation, optimization

1. THE GENERAL PROBLEM

Frequently a physical situation can only be described fully by a computational model. We wish to address the general problem of finding the values of the parameters in such a model that come closest to matching a given set of data. In data matching the objective function to be minimized is often the negative logarithm of the likelihood of the data given their predicted values, which yields the maximum likelihood (ML) solution. Alternative approaches include regularized versions of maximum likelihood and Bayesian methods in which the objective function is the minus-log-posterior, yielding the maximum a posteriori (MAP) estimate.

This work was supported by the United States Department of Energy under contract number W-7405-ENG-36.
Saquib's current address: Polaroid Corporation, 750 Main St., Cambridge, MA 02139.

We confine ourselves to objective functions that depend on the parameters in a continuous and differentiable fashion. We do not avoid problems for which the objective function possesses multiple minima. However, because the techniques that we present make use of gradients in the optimization process, they will work effectively only when one can easily find the basin of attraction for the global minimum, for example, by multiscale or multiresolution optimization.

The proposed method of solving the inverse problem is generally applicable to a wide variety of problems in which the measurements for the process in question are adequately described by a predictive forward computational model. We believe it may be used in estimating geophysical structure from seismic data. Other potential application areas include modeling of the ocean, atmosphere, fluid flow, and shock-wave phenomena, as well as optimization of engineering designs in complex situations such as streamlining of airplane foils and automobile bodies to reduce drag.

Limitations of space preclude elucidation of the details of the techniques presented. The reader should refer to the cited articles for more complete information. Other enabling techniques that can not be included in this account are multiscale analysis to constrain and accelerate the optimization process, deformable geometric models for describing objects with sharp boundaries, and the Markov Chain Monte Carlo method of sampling the uncertainty distribution of the estimated parameters [2].

2. ADJOINT DIFFERENTIATION

We wish to address problems that require minimizing a scalar function φ by varying the many (10^3 to 10^6 or more) variables that comprise the parameters of the object model. This optimization problem would be intractable without knowing the gradient of φ, or sensitivities, with respect to the parameters on which it depends. We have uncovered a technique to calculate these crucial sensitivities, called adjoint differentiation

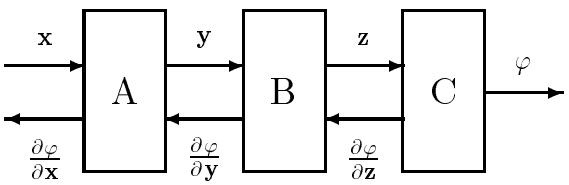


Figure 1: Data flow diagram showing a sequence of transformations, represented by the boxes A, B, and C connected by the arrows pointing to the right, starting with the data structure $\mathbf{x}$ and ending with the scalar φ. The data flow for the adjoint derivatives is indicated by the arrows pointed left.

[4], that is apparently relatively unappreciated. Using the adjoint differentiation technique, the calculation of all these derivatives can be done in a computational time that is comparable to the forward calculation.

Suppose that a calculation proceeds as a sequence of transformations as shown in Fig. 1. The independent variables in the data structures designated by the vector $\mathbf{x}$ are transformed by block A to produce the dependent variables $\mathbf{y}$. These are transformed by blocks B and C to produce the dependent data structure $\mathbf{z}$ and the final scalar φ, respectively.

We call the sequence of transformations

$$\mathbf{x} \stackrel{\mathrm{A}}{\to} \mathbf{y} \stackrel{\mathrm{B}}{\to} \mathbf{z} \stackrel{\mathrm{C}}{\to} \varphi\,,$$

the forward calculation. We assume that the transformations are general, with the only restriction being that they are differentiable. Each transformation is self-contained; it requires only its input variables to calculate its output variables, e.g., module B uses only its input $\mathbf{y}$ to calculate its output $\mathbf{z}$. Therefore, each transformation should require nothing more than its input to implement the derivative of its output variables with respect to its input variables. The data structures are likewise general.

The chain rule allows us to calculate the derivatives of φ with respect to the ith component of $\mathbf{x}$,

$$\frac{\partial\varphi}{\partial x_i} = \sum_{jk} \frac{\partial\varphi}{\partial z_k}\frac{\partial z_k}{\partial y_j}\frac{\partial y_j}{\partial x_i}\,. \qquad (1)$$

Even if the transformations are nonlinear, this expression amounts to a product of matrices. The order of the summations can obviously be done in two different ways. If the sum over j is done before the sum over k, the calculation proceeds in the same direction as the forward model calculation. As the dimensions of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ are assumed to be large, this sequence results in very large intermediate matrices, which we would like to avoid.

On the other hand, if the sum on k is done before that on j, the sequence of calculations is

$$\mathbf{I} \stackrel{\mathrm{C}'^{\dagger}}{\longrightarrow} \frac{\partial\varphi}{\partial\mathbf{z}}^{\dagger} \stackrel{\mathrm{B}'^{\dagger}}{\longrightarrow} \frac{\partial\varphi}{\partial\mathbf{y}}^{\dagger} \stackrel{\mathrm{A}'^{\dagger}}{\longrightarrow} \frac{\partial\varphi}{\partial\mathbf{x}}^{\dagger}\,,$$

where, for example, $\mathrm{B}'^{\dagger}$ effectively multiplies $\frac{\partial\varphi}{\partial\mathbf{z}}$ by the adjoint of the matrix $\frac{\partial\mathbf{z}}{\partial\mathbf{y}}$. The symbol $\mathbf{I}$ at the beginning of the sequence represents the identity structure, indicating that the sensitivity calculation begins with the first adjoint derivative transformation $\mathrm{C}'^{\dagger}$. This sequence implies intermediate data structures (e.g. $\frac{\partial\varphi}{\partial\mathbf{y}}$) that mimic the normal data structures (e.g. $\mathbf{y}$) implying storage requirements identical to the forward calculation. If the forward transformations are nonlinear, the forward data may be required for the adjoint calculation. The backward flow of the adjoint derivatives is depicted in Fig. 1.

The adjoint differentiation calculation is straightforward to program [5]. Provided that the logic of the forward calculation is not too intricate, the adjoint derivative calculation should involve an amount of computation comparable to the forward computation.

We have coined the acronym Adjoint Differentiation In Code Technique (ADICT) [3] to describe a particular approach to adjoint differentiation. The unique feature of ADICT is that the computer code for the adjoint calculation is based on the simulation code with the explicit intent to "differentiate" the forward calculation. For optimization of a functional based on computation, it is desirable to have the gradient of the computation, not of the physics equations that the computation is supposed to approximate.

There are several compilers of FORTRAN code that "automatically" produce adjoint differentiation code, including the well-known ADIFOR [6] and GRESS [7]. However, these approaches impose heavy memory requirements. More promising for application to large simulation codes is a code-based approach by Giering [8]. So far, our approach has been to manually code the adjoint code, sometimes a daunting task, but one that can be learned [5, 9]. Our experience with object-oriented design and programming indicates its tremendous advantage for linking calculations together to form a data-flow diagram made of autonomous transformations, as in our Bayes Inference Engine [3].

3. OPTIMIZATION

The ML or MAP solution is found by minimizing a scalar functional φ with respect to all the model parameters. Given the possibly large number of parameters, it is imperative to use the derivatives of φ with respect

to all parameters. Fortunately, there is a technique to efficiently calculate these gradients as described in the previous section. The standard approaches to gradient-based optimization of functions of many parameters are steepest descent and conjugate gradient. Beyond these there are a number of techniques that are generally referred to as quasi-Newton methods. Davidon pioneered the variable metric approach [10], which is based on building up an approximate expression for the inverse Hessian (the second-derivative matrix of φ). Our initial success with this algorithm leads us to believe that similar, but more adaptable algorithms are worth exploring, namely the limited-memory BFGS (Broyden-Fletcher-Goldfarb-Shanno) [11] and possibly the truncated Newton [12] algorithms, both of which seem to have strengths for different kinds of problems.

4. EXAMPLE: TIME-DEPENDENT DIFFUSION

As an example of the success of ADICT, let us summarize the results of Saquib et al. [1], who investigated the diffusion of infrared light through tissue. They solved the problem of inversion of time-resolved data to obtain the distribution of diffusion coefficients through which the light passed. Suppose that the intensity of diffused light at position (x, y) and time t is denoted by $U(x, y, t)$ and the source strength by $R(x, y, t)$. Then the time evolution of U through a region described by diffusion coefficients $D(x, y)$ and absorption coefficients $\mu_a(x, y)$ is given by the diffusion equation

$$\frac{\partial U}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial U}{\partial x}\right) + \frac{\partial}{\partial y}\left(D\frac{\partial U}{\partial y}\right) - c\mu_a U + R \ , \quad (2)$$

where the spatial and temporal dependence of the parameters has been suppressed. The quantity c is the speed of light.

We approach the computational problem in terms of discrete samples of U on a spatial grid with locations specified as a subscript s and in time with a superscript n. When the spatial position subscript is dropped, the resulting quantity is a column vector obtained by either row-ordering or column-ordering the corresponding two-dimensional field (e.g., U^n).

For simplicity we assume that the measurements are degraded by additive uncorrelated Gaussian noise. The minus-log-likelihood of the observations Y given D and μ_a is

$$\varphi = -\log P(Y|D, \mu_a) = \tfrac{1}{2}\chi^2 = \sum_{s,n} \frac{(Y_s^n - \tilde{U}_s^n)^2}{2\sigma_{s,n}^2} \ , \quad (3)$$

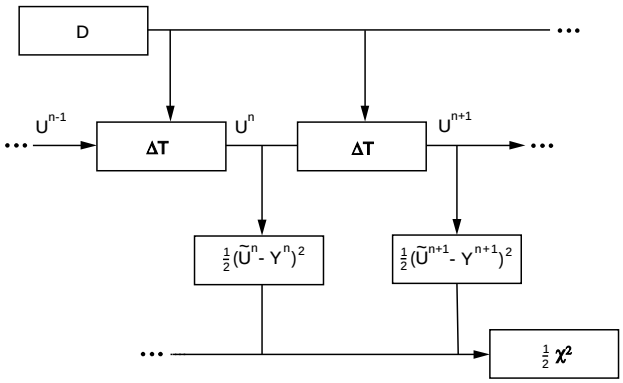


Figure 2: Data flow diagram for the forward calculation of the diffusion problem resulting in $\varphi = \frac{1}{2}\chi^2$. The adjoint differentiation calculation reverses this flow.

where $\sigma_{s,n}^2$ is the noise variance at spatial position s and time n and the tilde on the U indicates only those positions at which the light intensities are measured.

To compute (3) given D and μ_a, we need to solve the diffusion equation (2) forward in time to obtain the diffuse intensity U_s^n for all time n and spatial positions s. We will briefly summarize the approach to this calculation employed in Ref. [1], which should be consulted for the details.

4.1. Solving the Forward Problem

The general approach taken to solve this forward problem is to use the finite-difference method in which the spatial and temporal derivatives in Eq. (2) are replaced by their finite-difference approximations. This substitution results in a difference equation that needs to be solved forward in time. When solving the difference equation for U^{n+1}, the finite-difference approximations to the spatial derivatives can be evaluated either at time index $n+1$ or n. In the implicit method[1] for solving differential equations, the spatial derivatives are evaluated at the time instance $(n+1)$ when computing the diffuse intensity U^{n+1}. The implicit method is unconditionally stable for any value of Δt.

Substituting in Eq. (2) the equation to be solved to obtain U^{n+1} from U^n is, in vector notation,

$$AU^{n+1} = U^n + \bar{R}^{n+1/2} \ , \quad (4)$$

where A is a sparse matrix (because derivatives involve only local variables) whose elements depend on the D and μ_a values. $\bar{R}^{n+1/2}$ denotes the integrated source strength between time instances n and $n+1$.

[1] We note that in [1] a slightly different method, called the Alternating-Directions Implicit (ADI) method, was used for calculational efficiency.

The procedure for calculating the time-evolution of U is depicted in Fig. 2. The transformations denoted by ΔT essentially involve solving Eq. (4) to move forward by one time step. The minus-log-likelihood ($\frac{1}{2}\chi^2$) is the accumulation of the sum of the squares of the differences between the measurements Y_s^n and their predicted values $\tilde{U}_s^n$. Thus it gets a contribution from each measurement time. The assumed time-independent distribution of D is used in each time step calculation.

4.2. Gradient Computation

We designate the unknown parameters by the column vector $\theta = [D\ \mu_a]^T$. We need the derivative or sensitivity of $\varphi(\theta)$ with respect to θ to facilitate the solution of the inverse problem. ADICT requires us to work backwards in time using the same discretized equations that is used to compute the forward solution. The sensitivity of φ with respect to θ is obtained by computing the intermediate sensitivity of φ with respect to the light intensity U at all time steps. We present a brief outline of the approach.

The sensitivity of φ with respect to U^n is obtained recursively by using the sensitivity of φ with respect to U^{n+1}. Application of the chain rule yields

$$\frac{d\varphi}{dU^n} = \left[\frac{dU^{n+1}}{dU^n}\right]^T \frac{d\varphi}{dU^{n+1}} + \frac{\partial\varphi}{\partial U^n}\ , \qquad (5)$$

where $\frac{\partial\varphi}{\partial U^n}$ denotes the change in φ when only U^n is varied, keeping all other variables constant, while $\frac{d\varphi}{dU^n}$ denotes the total change in φ when U^n is varied along with all variables that depend on U^n. Differentiating Eq. (3) with respect to U_s^n, we obtain

$$\frac{\partial\varphi}{\partial U_s^n} = \frac{(Y_s^n - \tilde{U}_s^n)}{\sigma_s^2}\ .$$

This result flows backwards through the boxes that compute contributions to $\frac{1}{2}\chi^2$ at each time step. Differentiating Eq. (4) with respect to U^n, we obtain

$$\frac{dU^{n+1}}{dU^n} = A^{-1}\ . \qquad (6)$$

Using Eqs. (5) and (6), we obtain the sensitivity of φ with respect to U^n as

$$\frac{d\varphi}{dU^n} = (A^{-1})^T \frac{d\varphi}{dU^{n+1}} + \frac{\partial\varphi}{\partial U^n}\ . \qquad (7)$$

In the diagram this result comes from each ΔT transformation and flows backwards from U^{n+1} to U^n.

Similar use of the chain rule yields the sensitivity of φ with respect to θ, which result flows out of the top of the ΔT box in Fig. 2 and gets added to the total derivative of φ with respect to the D vector.

4.3. Inversion

The problem of reconstructing the unknown parameters D and μ_a from the measurements Y_s^n is an ill-posed inverse problem. Some form of regularization is necessary to make the solution well behaved. We accomplished this by incorporating an image model in the reconstruction process that models our *a priori* knowledge regarding the unknown fields D and μ_a. Markov random fields (MRF) have been extensively used in image processing applications. We model D as a generalized Gaussian MRF (GGMRF) [13] with an energy function of the form

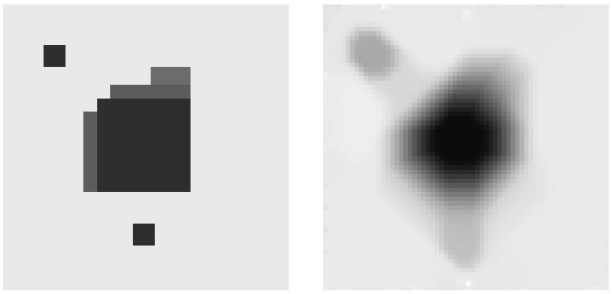
$$u\left(\frac{D}{\sigma_D}\right) = \sum_{\{s,r\}\in\mathcal{N}} \frac{b_{s-r}}{p}\left|\frac{D_s - D_r}{\sigma_D}\right|^p\ , \qquad (8)$$

where $\mathcal{N}$ is the set of all neighboring pixel pairs. The popular choice of $p = 2$ in the signal-processing literature yields a quadratic cost function, which tends to excessively penalize large deviations resulting in blurred edges. It is possible to provide good edge preservation in the reconstructed image for $p \approx 1$ [14]. Furthermore, the form of the model facilitates the estimation of the strength of this prior directly from the data [14].

Our example consists of a simulation of time-resolved data for a 6.4-cm-square section of tissue. Figure 3 shows as a 64×64 image the original diffusion coefficients, which range in value from 0.7 to 1.4 $\text{cm}^2\text{ns}^{-1}$. The absorption coefficients are set to a constant value of 0.1 cm^{-1}. The values of these coefficients, as well as the physical dimensions of the problem, have been chosen to reflect those of real tissues. Although the above method can be used to estimate D and μ_a simultaneously, we will restrict ourselves to the simpler case of just estimating D and assume that μ_a is known.

We assume that there are four pulsed sources placed at the midpoints of each side of the square region. There are 52 detectors evenly spaced along the sides, which measure the time-dependent signal in response to each pulsed source. Gaussian noise is added to the simulated signals with an rms value of 3% of the rms signal value over the 1.0 ns observation time, corresponding to a signal-to-noise ratio of 30 dB. The time step used is $\Delta t = 0.005$ ns and the detector resolution is 0.02 ns.

Figure 3 shows the MAP reconstruction for $p = 1.1$ obtained using 70 iterations of the conjugate gradient algorithm (taking about nine hours on an HP 9000/755). The reconstruction is remarkably good considering that effectively only four views are used. This result confirms the value of incorporating ADICT into a simulation code to solve this inversion problem in which roughly 4000 parameters are determined from



(a) (b)

Figure 3: The original distribution of diffusion coefficients (a) and their reconstructed values (b) derived from time-dependent measurements made around the periphery in response to short pulses introduced at the middle of each of the four sides.

approximately 10000 measurements (54 detectors $\times$ 50 time samples $\times$ 4 source positions).

5. DISCUSSION

We have presented some useful tools that permit one to efficiently estimate parameters of a complicated forward model from measurements. The use of a forward model is important because many steps in a model that might describe a physical system and measurement scenario may not easily be directly inverted. This general approach allows one to attempt to construct complete models to fully account for the observations.

The primary technique in the toolkit is the Adjoint Differentiation In Code Technique (ADICT) that yields derivatives of a functional (objective function) based on a forward computational code with respect to all the parameters in the computational model. The derivatives of the computed functional are desirable when minimizing that functional. In the partcular approach that we suggest, adjoint differentiation is accomplished through code rather than by storing derivative matices. The resulting calculational time for the derivatives is comparable to that of the forward model calculation. ADICT can, in principle, can be implemented for any code that is differentiable. Adjoint differentiation is being applied to some large hydrodynamic codes, e.g., to calculate the dynamics of the ocean and atmosphere.

ADICT permits the use of gradient-based optimization algorithms, which are quite efficient. We have had some success with a quasi-Newton method of optimization and we therefore suggest that the limited-memory BFGS algorithm [11] may be very useful for problems involving many parameters of mixed type.

When optimizing nonlinear models it is possible to encounter objective functions with multiple local minima, which can pose difficulties for gradient-based algorithms. However, for many kinds of problems, the desired minimum can be found either by knowledgeably choosing at good starting point or by using a multiresolution approach, i.e. by finding the minimum at coarse resolution and then working toward finer resolutions [15]. If this approach does not work for a particular problem, to find the global minimum it may be necessary to resort to stochastic optimization algorithms (simulated annealing or genetic algorithm), which are notoriously inefficient compared to gradient-based approaches. It might be possible to combine the best of both approaches through a hybrid algorithm in which the stating point of a gradient-based algorithm is chosen stochastically.

It is often desirable to use a high-level model to describe an object or situation of interest. As an example, we have found it very useful to employ a deformable geometric model to represent the boundary of an object that we wish to reconstruct from projection data [2, 5]. In three dimensions, such a model might consist of a surface represented by many triangles. An even higher-level model would use spline-based patches, which would result in fewer parameters, but not necessarily much less computation time. High-level models are often invoked to help regularize or control the inversion problem in the belief that it should be easier to solve a problem involving fewer variables. However, we have come to realize that it may be more desirable to use a very flexible description involving many parameters. The flexibility of such a model can be controlled through the use of a prior or graded constraint function, which effectively reduce the number of degrees of freedom of that model. Such constraints often take the form of an integral of the square of a derivative of some quantity, which bascially acts to smooth that quantity. The advantage of this general approach is that it allows one to choose the prior that is most appropriate for the problem and even locally turn off the control when that is indicated by the data or circumstance [16].

In constructing the Bayes Inference Engine [3], we have uncovered another basic tool for model building. In order to easily accommodate a variety of deformable geometric models in conjunction with a variety of potential measurement scenarios, we decided to employ an intermediate elemental representation for the object of interest. Every high-level model is converted to the elemental representation before the measurement process can be accomplished. For a physical object the elemental representation is a discretized density image (array of pixels) in 2D or a voxelated array in 3D. This

approach permits implementation of a new high-level object model without writing new code to calculate the result of each type of measurement.

Beyond its calculational advantage, we have come to recognize the underlying value of the elemental representation as a basic modeling tool. Its importance lies in the fact that when one uses and sees only a high-level model, deficiencies in matching the data can only be displayed in terms of the gradients of φ with respect to the model parameters. It can be difficult to recognize what aspects of the model do not accommodate the data. However, by displaying the gradients in the elemental representation, it can become evident how the high-level model needs to be changed, and possibly augmented, to better match the data [16].

An important message that we wish to get across is that one should not be afraid of using models that contain large numbers of variables. With reasonable constraints on the models, one can easily accommodate lots of parameters. Our experience indicates that there is often no big penalty associated with using many parameters, either in terms of computational speed or ill posedness.

6. ACKNOWLEDGEMENTS

We have had helpful conversations with John Skilling, Steve Gull, Rudy Henninger, Maria Rightley, Paul Maudlin, and Bryan Travis.

7. REFERENCES

[1] S. S. Saquib, K. M. Hanson, and G. S. Cunningham. Model-based image reconstruction from time-resolved diffusion data. In *Medical Imaging: Image Processing*, K. M. Hanson, ed., *Proc. SPIE*, 3034:369–380, 1997.

[2] K. M. Hanson, G. S. Cunningham, and R. J. McKee. Uncertainties in tomographic reconstructions based on deformable models. in *Medical Imaging: Image Processing*, K. M. Hanson, ed., *Proc. SPIE*, 3034:276–286, 1997.

[3] K. M. Hanson and G. S. Cunningham. A computational approach to Bayesian inference. In M. M. Meyer and J. L. Rosenberger, eds., *Computing Science and Statistics* **27**, pages 202–211. Interface Foundation, Fairfax Station, VA 22039-7460, 1996.

[4] W. C. Thacker. Automatic differentiation from an oceanographer's perspective. In A. Griewank and G. F. Corliss, eds., *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pages 191–201. SIAM, Philadelphia, 1991.

[5] X. L. Battle, G. S. Cunningham, and K. M. Hanson. 3D tomographic reconstruction using geometrical models. In *Medical Imaging: Image Processing*, K. M. Hanson, ed., *Proc. SPIE*, 3034:346–357, 1997.

[6] C. Bischof, A. Carle, P. Khademi, and A. Mauer. The ADIFOR 2.0 system for the automatic differentiation of FORTRAN 77 programs. Technical Report ANL-MCS-P481-1194, Argonne National Laboratory, 1995.

[7] J. E. Horwedel, E. M. Oblow, B. A. Worley, and F. G. Pin. GRESS 3.0 Gradient Enhanced Software System. Technical Report PSR-231, RSIC Peripheral Shielding Routine Collection, Oak Ridge National Laboratory, 1994.

[8] R. Giering. Tangent linear and Adjoint Model Compiler. Technical Report TAMC 4.7, Max-Planck-Institut für Meteorologie, 1997.

[9] R. L. Henninger, P. J. Maudlin, M. L. Rightley, and K. M. Hanson. Application of forward and adjoint techniques to hydrocode sensitivity analysis. In F. Graviana et al., eds., *Proc. 9th Nuclear Explosives Code Developers' Conference*. Lawrence Livermore National Laboratory, 1996 (proceedings are classified; article available from author as report LA-CP-96-235.

[10] W. C. Davidon. Variable metric method for minimization. Technical Report ANL-5990, Argonne National Laboratory, 1966.

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

[12] S. G. Nash. Preconditioning of truncated-Newton methods. *SIAM J. Sci. Stat. Comput.*, 6:599–616, 1985.

[13] C. A. Bouman and K. Sauer. A generalized Gaussian image model for edge-preserving MAP estimation. *IEEE Trans. Image Processing*, 2:296–310, 1993.

[14] S. S. Saquib, C. A. Bouman, and K. Sauer. ML parameter estimation for Markov random fields, with applications to Bayesian tomography. Technical Report TR-ECE 95-24, School of Electrical and Computer Engineering, Purdue University, 1995.

[15] G. S. Cunningham, I. Koyfman, and K. M. Hanson. Improved convergence of gradient-based reconstructions using multi-scale models. in *Image Processing*, M. H. Loew and K. M. Hanson, eds., *Proc. SPIE*, 2710:145–155, 1996.

[16] K. M. Hanson, R. L. Bilisoly, and G. S. Cunningham. Kinky tomographic reconstruction. in *Image Processing*, M. H. Loew and K. M. Hanson, eds., *Proc. SPIE*, 2710:156–166, 1996.