Implementation of the Conjugate Gradient Algorithm in DSO

Susan E. Minkoff*

July 11, 1994

Abstract

Computation of the inner state parameters in DSO inversion requires solving a large normal matrix system. A combined conjugate gradient and Lanczos iterative technique can be used to both solve the system and approximate some of the spectrum of the normal operator. At each iteration of the conjugate gradient algorithm, a small tridiagonal matrix (of dimension equal to the number of iterations) is created which has extreme eigenvalues approximating those of the original matrix. A second matrix whose columns are the normalized residual vectors from the conjugate gradient algorithm allows the corresponding eigenvectors to be computed as well if desired. Implemented so that it can be applied to different DSO inversion problems, the conjugate gradient code provides the user with a tool to analyze the condition of the problem as well as the quality of the inversion results. Storage of the residual (Lanczos) vectors may be costly if large problems are being solved. Numerical experiments indicate that the largest eigenvalue is the best approximation in the spectrum and the smallest is generally the next best.

1 Introduction

It is well-known that the inverse problem of wave propagation in reflection seismology is difficult because multiple local minima generally render Newton-type methods unsuitable. The *D*ifferential *S*emblance *O*ptimization (DSO) inversion package was designed to allow quasi-Newton methods to be used to search through model space for parameters which explain seismic data. Separate the model parameters into short and long-wavelength components (which we call the inner and outer states respectively). Then one of the important features of the DSO approach is that gradient-based optimization can be used to solve the inversion problem when we solve first for the inner state and then the outer state parts of the model.

In this paper we discuss an implementation of the conjugate gradient algorithm used to solve the normal equations for the inner state problem. Although other iterative techniques exist as

*The Rice Inversion Project, Department of Computational and Applied Mathematics, Rice University, Houston TX 77251-1892

part of DSO for solving the normal equations, there was at least one important reason to implement the conjugate gradient code. The conjugate gradient algorithm automatically generates the parameters needed to construct the tridiagonal matrix which arises in the Lanczos iteration. The extreme eigenvalues of this tridiagonal matrix approximate those of the original matrix (in our case, the normal matrix). The eigenvalues and eigenvectors provide information about the condition of the normal matrix. They indicate which components of the model are the least and best determined. Finally, they provide a way to measure how well or poorly one set of inversion model parameters compares with another. (For a reference to the conjugate gradient and Lanczos algorithms see [GOLUB and LOAN, 1989]. For a discussion of the algorithmic construction of DSO see [SYMES and KERN, 1994])

DSO contains code modules (for example, optimization software) which are used by all modeling and inversion experiments performed in our group and modules written for specific applications (i.e., forward modeling code). The conjugate gradient algorithm is part of the generic half of DSO and thus can be used to solve many different types of problems (as well as estimate part of the normal operator spectrum). One could, for example, use the code without modification to invert real or synthetic data modeled using an acoustic, elastic, or viscoelastic simulator.

The next section of this paper describes in general terms the structure of the differential semblance optimization technique and where this code is situated in that framework. Section 3 covers the conjugate gradient and Lanczos algorithms, and section 4 describes their implementation in the context of DSO. A numerical example of the conjugate gradient (and Lanczos) algorithms is given in section 5.

2 How the Conjugate Gradient Algorithm Fits into DSO

Differential semblance optimization was motivated by the properties of wave propagation, specifically in the context of exploration seismology. One aspect of the DSO approach is that the model parameters sought in the inversion are grouped according to the influence they have on the data. The elements which have a *nonlinear* effect on the data become part of the outer (model) state. The inner state elements have a *linear* effect on the data. Finally, the elements in the parameter state are the *fixed* elements of the total model space.

Like the least squares objective function, the DSO objective function suffers from being highly nonconvex. However, the separation of the short and long-wavelength components of the model into the inner and outer states provides the following path to getting around this problem. Namely, if the differential semblance objective function is minimized first over the inner state variables, the remaining function of the outer state variables is smooth and convex in a large domain. Thus, it can be minimized effectively by gradient-based optimization methods. (See [SYMES and KERN, 1994].)

At each step in velocity model space (the most common outer state element), inversion must be carried out to estimate the parameters which define the inner state (such as the short-scale relative fluctuations in the elastic parameters or reflectivities). This inversion involves computing and solving the normal equations that arise in the process of the inner state minimization. (An explicit calculation of the normal equations for the DSO objective function may be found in [SYMES and KERN, 1994].)

Due to the large size of the normal matrices which are typical for seismic inverse problems, we prefer to use iterative methods to solve these linear systems. Since the normal operator is symmetric and positive definite, an obvious technique to use to solve this system is the conjugate gradient algorithm. The next section describes the conjugate gradient and Lanczos algorithms in general terms. (The following general description of the two algorithms may be found in expanded form in [GOLUB and LOAN, 1989].)

3 Discussion of the Conjugate Gradient and Lanczos Algorithms

3.1 The Conjugate Gradient Idea

The Hestenes-Stiefel conjugate gradient algorithm may be understood in the context of minimizing the function $\phi(x)$ defined by

$$\phi(x) = \frac{1}{2}x^t Ax - x^t b \tag{3.1}$$

where $b \epsilon \Re^n$, and $A \epsilon \Re^{n \times n}$ is assumed to be positive definite and symmetric. The minimizer of ϕ is $x = A^{-1}b$. So, minimizing the function ϕ and solving the linear system $Ax = b$ are seen to be equivalent problems.

One obvious choice for decreasing the function ϕ is to travel in the negative gradient direction $-\nabla\phi(x_c) = b - Ax_c$ from the current point x_c. One notices that the negative gradient direction is the residual direction r_c of the system at the current point. Unfortunately, as is well known, this method (steepest descent) may converge extremely slowly if the condition of the system (or ratio of largest to smallest eigenvalues) is large. The conjugate gradient algorithm, therefore, chooses to minimize ϕ in a set of directions $\{p_1, p_2, ...\}$ which do not necessarily correspond to the residual directions. One approach with obvious benefits is to choose linearly independent directions p_i so that each x_k solves

$$\min_{x \epsilon \mathrm{span}\{p_1,\ldots,p_k\}} \phi(x) \tag{3.2}$$

This choice of search directions ensures finite termination of the algorithm in at most n steps. We would like a vector p_k such that when we solve the one-dimensional minimization problem

$$\min_{\alpha} \phi(x_{k-1} + \alpha p_k) \tag{3.3}$$

we also solve the k-dimensional problem 3.2. Luckily, such a solution is possible if we require the directions p_k to be *A-conjugate* to the previous directions $p_1, ..., p_{k-1}$. The vectors $p_1, ..., p_k$ are A-conjugate if $P_{k-1}^t A p_k = 0$. These requirements can be satisfied and an algorithmic implementation is described in subsection 2.3 below.

3.2 The Lanczos Idea and Connection to the Conjugate Gradient Algorithm

Estimates of the eigenvalues and eigenvectors of the normal operator could be very useful for analyzing the inversion results we obtain from DSO. The Lanczos algorithm when applied to a symmetric matrix $A \epsilon \Re^{n \times n}$, generates a sequence of tridiagonal matrices $T_j \epsilon \Re^{j \times j}$ with extreme eigenvalues which are progressively better estimates of the extreme eigenvalues of A.

One way to motivate the Lanczos idea is to recall the Rayleigh quotient which can be used to approximate the eigenvalues of a matrix A. Let λ_1 be the largest eigenvalue of A and λ_n the smallest. For $Q_j = [q_1, ..., q_j]$ a matrix in $\Re^{n \times j}$ with orthonormal columns, we define the scalars M_j and m_j by

$$M_j = \max_{y \neq 0} \frac{y^t (Q_j^t A Q_j) y}{y^t y} \leq \lambda_1(A) \tag{3.4}$$

$$m_j = \min_{y \neq 0} \frac{y^t (Q_j^t A Q_j) y}{y^t y} \geq \lambda_n(A) \tag{3.5}$$

The Lanczos algorithm provides a way to compute the q_j so that the scalars M_j and m_j are better and better estimates of $\lambda_1(A)$ and $\lambda_n(A)$. Let $x = Q_j y$. Then the Rayleigh quotient changes most rapidly in the direction of its gradient which is a vector contained in $span\{x, Ax\}$. For this reason, the Lanczos vectors $\{q_i\}_1^j$ are chosen to be an orthonormal basis for the Krylov subspace

$$\kappa(A, q_1, j) \equiv \text{span}\{q_1, Aq_1, ..., A^{j-1} q_1\} = \text{span}\{q_1, ..., q_j\} \tag{3.6}$$

At the jth iteration of the Lanczos algorithm we have a matrix Q (the Lanczos matrix) whose columns are the normalized residuals resulting from the conjugate gradient algorithm (which can be shown to be orthonormal) and a symmetric, tridiagonal matrix $T \epsilon \Re^{j \times j}$. In fact, the Lanczos matrix "tridiagonalizes" the matrix A up to an error matrix.

$$A Q_j = Q_j T_j + r_j e_j^t. \tag{3.7}$$

The entries in T are combinations of the parameters generated in the conjugate gradient iteration (for details see the algorithm next section).

3.3 Algorithm

We present here a pseudocode version of the two algorithms (conjugate gradient and Lanczos) which have been implemented in DSO.

Variables Used

A :	normal operator
b:	data
r:	residual
x_0:	starting solution
x:	approximate solution
p:	conjugate gradient direction
β:	parameter used in computation of new direction p
α_c	step length in current direction p
α_p	step length in previous direction
rtr_c:	inner product of current residual with itself
rtr_p:	inner product of previous residual with itself
tol:	relative residual tolerance used for determining algorithm convergence
Q:	Lanczos matrix
T	tridiagonal matrix resulting from Lanczos process with eigenvalues approximating those of A
Z	matrix of eigenvectors of the tridiagonal matrix T
X	matrix of approximate eigenvectors of the original matrix A

Algorithm: (Conjugate Gradient/Lanczos) If $A \epsilon \Re^{n \times n}$ is symmetric and positive definite and $b \epsilon \Re^n$ then the following algorithm computes $x \epsilon \Re^n$ so that $Ax = b$. The algorithm also optionally approximates some of the eigenvalues and eigenvectors of the matrix A.

initialize:
  $r = b - Ax_0$
  $x = x_0$
for $k = 1$:iteration limit
  if eigenvector flag = true
    $Q(:, k) = r/\|r\|$
  end if
  if $k = 1$
    $\beta = 0$
    $p = r$
    $rtr_c =< r, r >$
  else
    $\beta = rtr_c/rtr_p$
    $p = r + \beta p$
  end if
  $ap = Ap$
  $ptap =< p, ap >$
  if $ptap < tol$
    break
  end if
  $\alpha_c = rtr_c/ptap$

$x = x + \alpha_c p$
$r = r - \alpha_c ap$
if eigenvalue flag = true
  if $k = 1$
    $T(k,k) = 1/\alpha_c$
    $T(k,k-1) = 0$
    $T(k-1,k) = T(k,k-1)$
  else
    $T(k,k) = rtr_c/(rtr_p\alpha_p) + 1/\alpha_c$
    $T(k,k-1) = -\sqrt{rtr_c/rtr_p}/\alpha_p$
    $T(k-1,k) = T(k,k-1)$
  end if
  Call LAPACK routine SSTEQR to get eigenvalues/vectors (Z) of T.
  Compute error in approximate eigenvalue for normal operator.
end if
$rtr_p = rtr_c$
$rtr_c =< r, r >$
$\alpha_p = \alpha_c$
if $\sqrt{rtr_c} < tol$
  break
end if
if eigenvector flag = true
  $QZ = X$
end if
end

4 Implementation Details of the Conjugate Gradient Algorithm in DSO

When the DSO package was first designed, it was written in Fortran 77 to ensure portability. Unfortunately, Fortran 77 does not allow the user to define data structures. Since we need to be able to store data with different geometric information depending on the type of experiment or physical model, the code was designed to mimic more modern object-oriented languages such as C. At the highest level, DSO programs manipulate *strings of filenames* called macrofiles. The data is stored out-of-core in files containing all dimensional and other information necessary to interpret the data arrays themselves. The filenames serve as pointers to the data. The conjugate gradient algorithm works entirely within this macrofile environment. Each mathematical instruction must be performed using a hierarchy of routines. For instance, suppose the inner state parameters to be inverted for are stored in two separate files which we will call file1 and file2. The residual vector in the solution of the normal system will be split between two files as well. One residual will correspond to the portion of solution stored in file1 and the remainder of the residual will come from solution file2. To compute the dot product of the residual with itself, one first sends the macrofile with the two filenames to a parsing routine. The next level works with only one file at a time. The third level is structured to handle one record of this single file. Finally, the fourth (and lowest level) will perform the dot product of the data in that record with itself.

To accommodate the conjugate gradient routines, the DSO job deck now requires the user to include two flags which indicate whether the conjugate gradient code should estimate the extreme eigenvalues and eigenvectors of the normal operator. The user has three options. He may choose not to estimate the eigenvalues and eigenvectors. He may choose to estimate only the eigenvalues, or he may choose to estimate both the eigenvalues and eigenvectors of the normal operator. The range of choices provided is necessary because the user should take care in requesting the eigenvectors of the normal operator. The eigenvectors require storage equal to the length of the estimated parameters times the number of conjugate gradient iterations performed. Currently no more than 100 iterations of the conjugate gradient algorithm may be performed, but this number may be increased in the future if necessary.

The job deck also requires the user to include a section containing output filenames where the eigenvalues and eigenvectors are to be stored upon completion of the inversion. Only one file is required to store the eigenvalues. This file will contain only one record and one trace, and the number of data samples will equal the number of iterations of the conjugate gradient algorithm performed. The number of output files necessary to store the eigenvectors, however, is the same as the number of inner state parameters sought in the inversion. For instance, if the user requests three reflectivity files be estimated, then three output files for the corresponding estimated eigenvectors will also be required.

The extreme eigenvalues of the the tridiagonal matrix T approximate those of the original matrix A. However, the corresponding eigenvectors of the matrix T must be transformed via multiplication by the Lanczos matrix Q to correspond to the eigenvectors of the original matrix A. Thus, at each iteration of the conjugate gradient algorithm, if the eigenvectors have been requested, the normalized residual vector must be stored. Separate routines store the Lanczos vectors and change bases from T to A. Each of these routines is also written at the macrofile level. Unfortunately, at the point in the algorithm when the new Lanczos vector is being stored in the Lanczos matrix, the algorithm is unaware of the total number of conjugate gradient iterations which will be performed. Each vector is stored as a column in the matrix (record in the data file). Thus, at iteration j, records $1, ..., j-1$ are retrieved. The header information which indicates the total number of records in the file is incremented, and the updated records $1, ..., j-1$ are rewritten to the file before record j is added.

The eigenvalues and eigenvectors of the tridiagonal matrix are estimated using the LAPACK routine SSTEQR [ANDERSON et al., 1992]. This routine computes *all* of the eigenvalues and, optionally, the eigenvectors of a symmetric tridiagonal matrix using the implicit QL or QR method. The eigenvalues are ordered according to size from smallest to largest on exit from this package. This routine was chosen for the accuracy that the QR method provides. The LAPACK routine called may later be changed to accommodate computation of only selected eigenvalues and eigenvectors of T. We note, however, that the matrix T is quite small. At the jth iteration of the conjugate gradient algorithm, $T \epsilon \Re^{j \times j}$. (A typical size for the matrix T might be a matrix with fifteen rows and fifteen columns.)

The LAPACK routine returns a data array (in unformatted Fortran) containing the eigenvalues and an unformatted Fortran matrix Z containing the eigenvectors (if requested) of the tridiagonal matrix. The Lanczos matrix Q is stored in formatted files. The output eigenvectors X of A also must be stored as formatted files. In order to write the records of the output matrix X only once and thus save on total i/o, we used the Gaxpy version of the matrix-matrix product. The Gaxpy algorithm can be found in [GOLUB and LOAN, 1989] and is repeated here.

Algorithm: (Matrix Multiplication Gaxpy Version) If the matrices $A\epsilon\Re^{m\times r}$ and $B\epsilon\Re^{r\times n}$ are given then the following algorithm computes $C = AB$.

function:C=matmat.$jki(A, B)$
    m=rows(A); n = cols(B); r=cols(A)
    $C(1 : m, 1 : n) = 0$
    for $j = 1 : n$
        for $k = 1 : r$
            for $i = 1 : m$
                $C(i, j) = C(i, j) + A(i, k)B(k, j)$
            end
        end
    end
end matmat.jki

Finally, for each approximate eigenvalue we include an upper bound on the distance between this number and the closest eigenvalue of the normal matrix. The bound is easy to compute and depends on the last off-diagonal entry in the tridiagonal matrix T and the last entry in the corresponding eigenvector of T.

5 Numerical Examples

The experiment discussed in this section was designed to test the conjugate gradient algorithm's ability to solve the normal system as well as to estimate the extreme eigenvalues and eigenvectors of the normal operator. The synthetic data was generated from Gulf of Mexico data we received from Exxon Production Research Company. Starting with one 48-trace common-midpoint data gather from a survey done of this area, we performed an inversion to determine a realistic reflectivity for this data. The background velocity used was a smoothed version of the well-log (Figure 2). We had been given an estimate of the anisotropic air gun energy source in the form of a 31-term Legendre expansion in slowness. We chose the first trace of this source estimate as our target isotropic source. This source has a peak frequency of 15 Hz and is centered at 110ms.

Once we had determined the parameters which described our synthetic experiment (isotropic source, reflectivity, and background velocity), we generated the synthetic data which we would attempt to match in the inversion. The data (shown in Figure 1) consists of 13 plane-wave traces with slowness values ranging from $p_{min} = .1158$ ms/m to $p_{max} = .36468$ ms/m. The seismogram contained 3s worth of data.

The experiment we describe here (determination of the source and reflectivity) was done using a method known as alternation. Alternation dictates that the source parameters be fixed, and the reflectivity estimated by output least squares inversion. Then the reflectivity is updated and held fixed and the source estimated by output least squares inversion. The source is updated and this cycle is repeated until convergence.

The initial estimate of the reflectivity used for the inversions was $r = 0$. For the initial source estimate, we chose a Ricker wavelet also with peak frequency of 15 Hz but which had its peak centered in time at 0 ms (Figure 3). We were interested to see if the data contained sufficient information to move the source to its correct location in time during the course of the inversions. Each inversion round included an estimation of the source and an estimation of the reflectivity. In all, 26 rounds of inversions were performed to reduce the root mean square error to 5% of the data norm.

In this experiment the stopping criterion for the conjugate gradient algorithm was that the normal system be solved with a relative residual tolerance not more than 5% of the original residual. Figure 4 shows the target source (dashed line) plotted against the final inversion-estimated source (solid line). Figure 5 gives the same comparison for the final inversion-estimated reflectivity. The source was moved from its initial location of 0ms to the true location (110ms) during the course of the inversions, and the shape of both the source and reflectivity functions was recovered. The estimated source and reflectivity functions were scaled to correspond to the targets. When both the source and reflectivity are being estimated using the convolutional model, a scale ambiguity exists between these two parameters. For example, the source may be scaled up by a constant α and the reflectivity scaled down by $1/\alpha$ and the data fit just as well.

Figures 6-9 indicate how well our conjugate gradient algorithm estimated the eigenvalues and eigenvectors of the normal operator. In these experiments we reduced the relative residual tolerance to 1% and reran two of the inversions (namely alternation round 4 of the total 26 rounds). The graphs compare the vector Ax to the vector λx for the smallest and largest eigenvalue/eigenvector pairs for both the reflectivity and source experiments. The largest eigenvalue/eigenvector pair is the best approximation in the spectrum.

For the source inversion, fourteen conjugate gradient iterations were performed before convergence occurred. The eigenvalues ranged from λ_1=1.186129E+01 to λ_n=1.819164E-01. Twelve conjugate gradient steps were taken in the reflectivity inversion in order for the relative residual to be reduced to 1% of the data norm. The eigenvalues ranged from λ_1=3.902964E-05 to λ_n=1.183508E-06.

The theoretical bound described at the end of the last section, indicates that the largest eigenvalue/vector pair is the best approximate pair and the smallest the next best. In the case of the source inversion, the largest eigenvalue /eigenvector pair was seen to satisfy the eigenvalue equation (via this theoretical bound) with error = 6.53134E-04. The smallest eigenvalue/eigenvector pair for the source inversion satisfied the eigenvalue equation with error = 0.392509. In this case other eigenvalues towards the top of the spectrum actually had smaller error in relation to the normal operator spectrum than did the smallest. For the reflectivity inversion, the largest eigenvalue/eigenvector pair satisfied the eigenvalue equation with error = 2.36527E-07. The smallest had error = 6.18076E-07.

6 Conclusion

In DSO inversion the parameters in the model are estimated in two steps. In the first step, parameters which affect the data linearly (the inner state) are estimated via solution of a normal system. This large normal matrix system is best solved via iterative methods. The conjugate gradient algorithm implemented here is advantageous because it allows us to simultaneously generate a small, square tridiagonal matrix (with size the number of conjugate gradient iterations) whose extreme eigenvalues approximate those of the normal matrix. The conjugate gradient code (being part of the generic section of DSO modules) may be applied to different types of problems. Estimates of the extreme eigenvalues of the normal operator will allow us to understand how well various parameters in the problem are determined as well as the condition of the overall problem. The algorithm is implemented to be consistent with the DSO macrofile environment. The user may chose not to estimate the eigenvalues and eigenvectors of the normal operator. He may choose only the eigenvalues be estimated, or both the eigenvalues and eigenvectors can be approximated. One should keep in mind that storage requirements for estimating the eigenvectors may be large if the parameters estimated are large. We provide an error bound as well which indicates how close each approximate eigenvalue is to an element of the spectrum of the normal operator. The approximate eigenvalues improve as the residual tolerance for solving the normal system is reduced, and the largest eigenvalue most closely approximates an element of the spectrum.

Acknowledgement: This work was partially supported by the National Science Foundation, the Office of Naval Research, the Texas Geophysical Parallel Computation Project, the Schlumberger Foundation, and The Rice Inversion Project. TRIP Sponsors for 1994 are Advance Geophysical, Amoco Production Co., Conoco Inc., Cray Research Inc., Exxon Production Research Co., Interactive Network Technologies, Mobil Research and Development Corp., and Texaco Inc.

References

[ANDERSON et al., 1992] ANDERSON, E., BAI, Z., BISCHOF, C., DEMMEL, J., DONGARRA, J., CROZ, J. D., GREENBAUM, A., HAMMARLING, S., MCKENNEY, A., OSTROUCHOV, S., and SORENSEN, D. (1992). *LAPACK Users' Guide.* Society for Industrial and Applied Mathematics, Philadelphia.

[GOLUB and LOAN, 1989] GOLUB, G. and LOAN, C. V. (1989). *Matrix Computations.* The Johns Hopkins University Press, Baltimore.

[SYMES and KERN, 1994] SYMES, W. and KERN, M. (1994). Inversion of reflection seismograms by differential semblance analysis: Algorithm structure and synthetic examples. *Geophysical Prospecting*, in press.

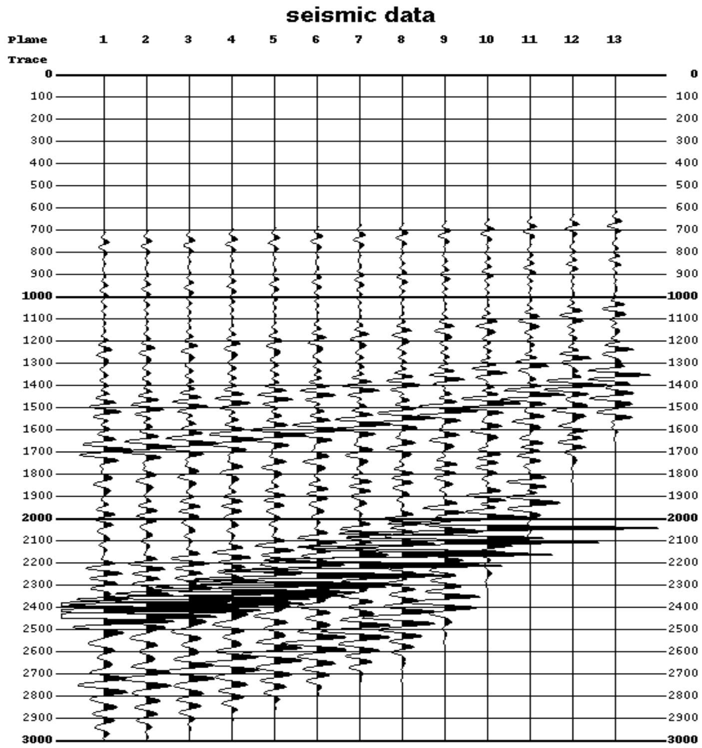

FIGURE 1: Synthetic common midpoint data used in the inversion experiments. The seismogram contains 13 traces of 3s worth of data. The vertical axis is time (in ms). The horizontal axis is slowness (in ms/m).

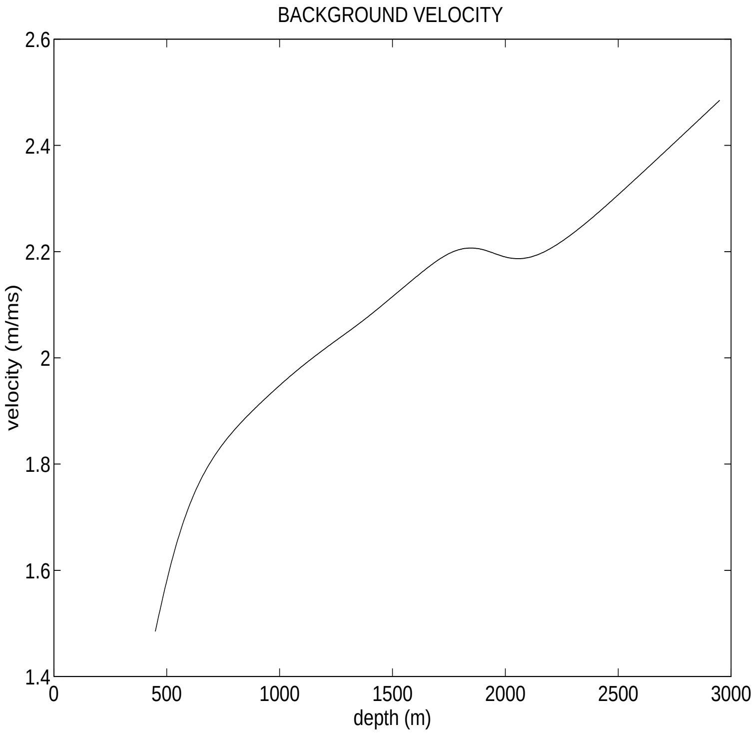

Figure 2: Background velocity used to generate the seismic data. The background velocity is held fixed in these inversion experiments.

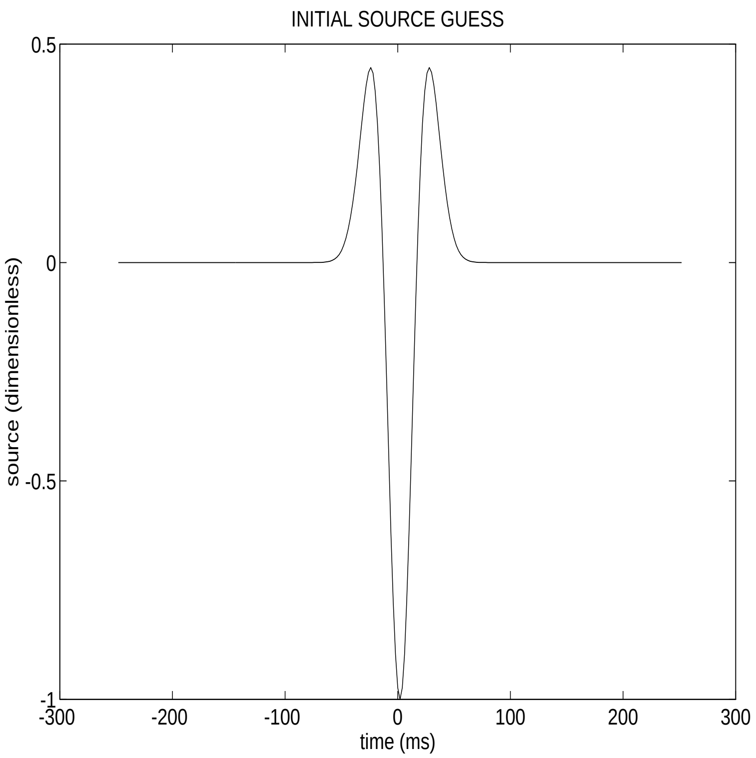

FIGURE 3: Initial guess for the seismic source (a 15Hz Ricker wavelet with peak at 0ms).

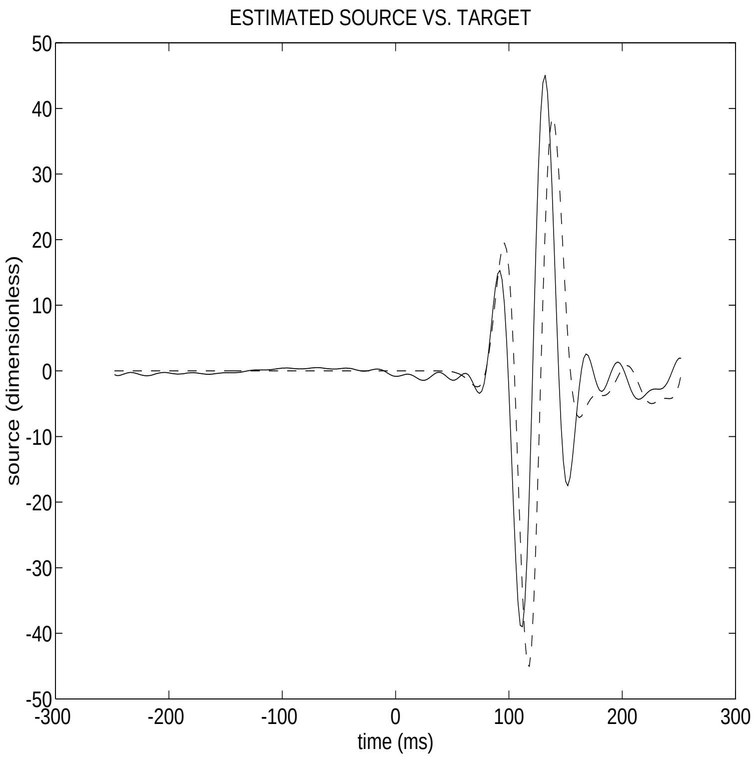

Figure 4: Final source inversion results. The Solid line is the estimated source (scaled) resulting from using the conjugate gradient algorithm for the inversions. The Dashed line is the target isotropic source.

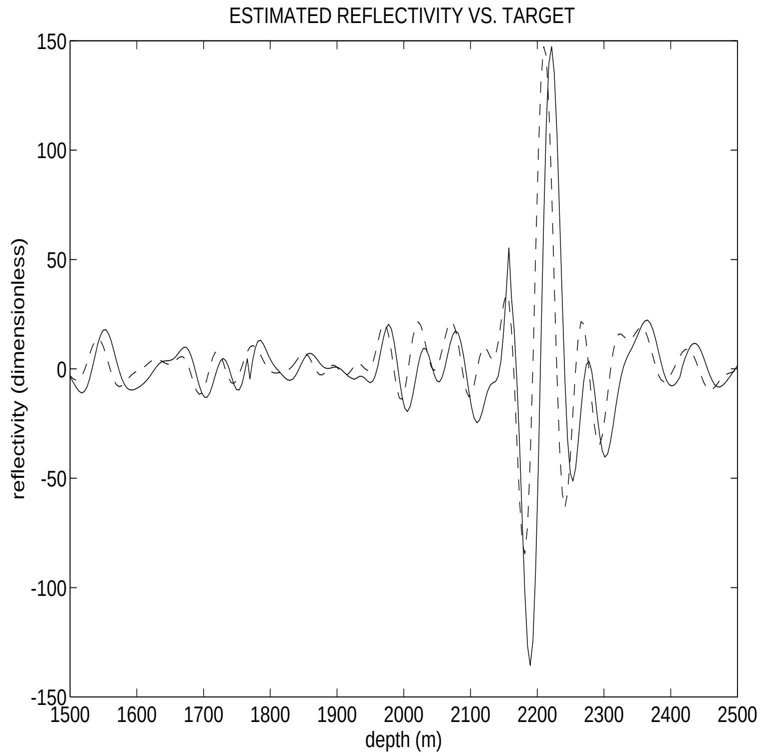

FIGURE 5: Final reflectivity inversion results. The Solid line is the reflectivity estimate (scaled) resulting from using the conjugate gradient algorithm for the inversions. The Dashed line is the target reflectivity for this synthetic experiment.

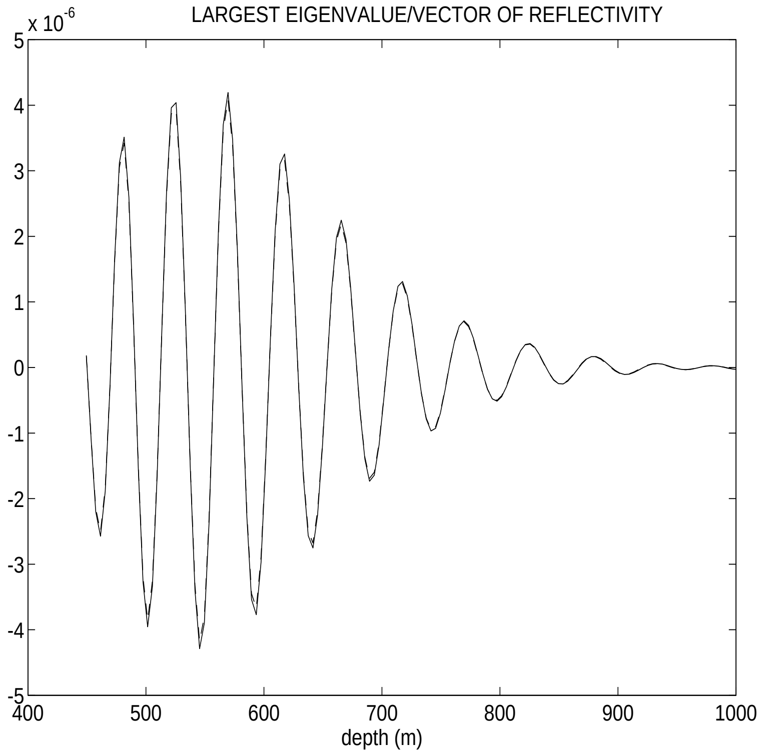

Figure 6: Plot of the degree to which the eigenvalue equation $Ax = \lambda x$ is satisfied by an approximate eigenpair of the normal matrix from the conjugate gradient/Lanczos algorithm. The normal matrix A corresponds to the fourth round of alternation for the *reflectivity*. Solid line: λx where λ is an approximation to the *largest* eigenvalue (λ_1=3.902964E-05) of the normal operator (and x the corresponding eigenvector). Dashed line: Ax.

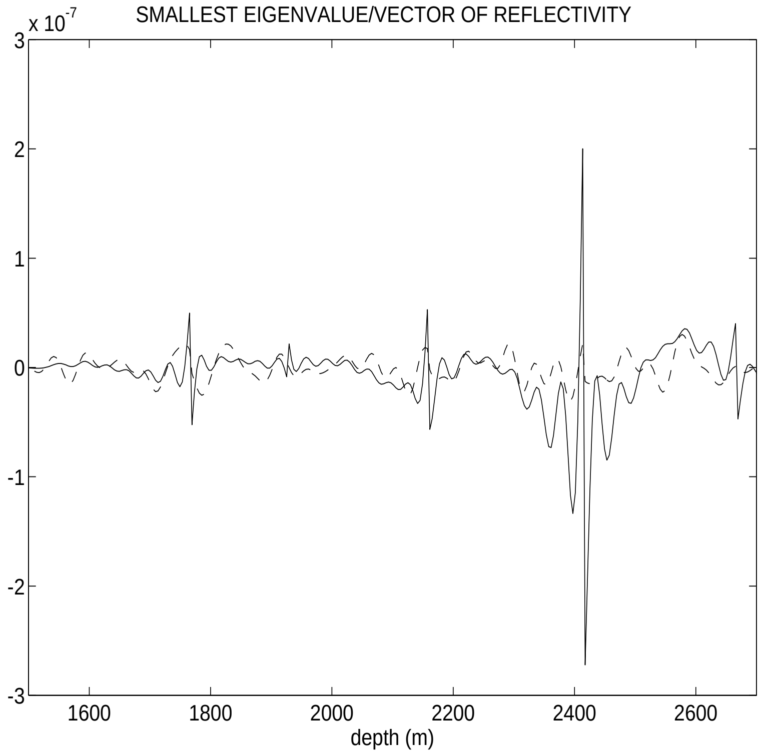

FIGURE 7: Plot of the degree to which the eigenvalue equation $Ax = \lambda x$ is satisfied by an approximate eigenpair of the normal matrix from the conjugate gradient/Lanczos algorithm. The normal matrix A corresponds to the fourth round of alternation for the *reflectivity*. Solid line: λx where λ is an approximation to the *smallest* eigenvalue (λ_n=1.183508E-06) of the normal operator (and x the corresponding eigenvector). Dashed line: Ax.

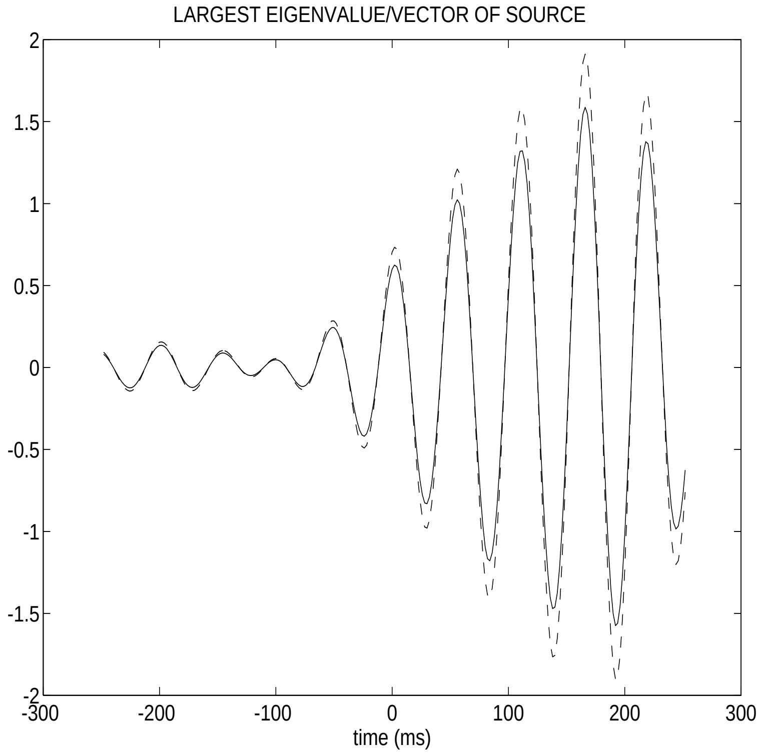

Figure 8: Plot of the degree to which the eigenvalue equation $Ax = \lambda x$ is satisfied by an approximate eigenpair of the normal matrix from the conjugate gradient/Lanczos algorithm. The normal matrix A corresponds to the fourth round of alternation for the *source*. Solid line: λx where λ is an approximation to the *largest* eigenvalue (λ_1=11.86129) of the normal operator (and x the corresponding eigenvector). Dashed line: Ax.

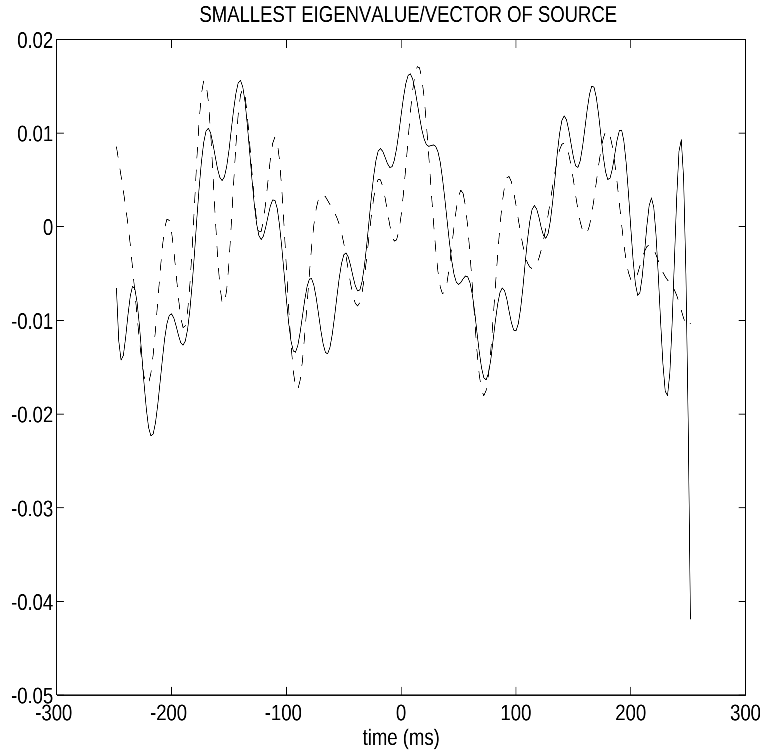

FIGURE 9: Plot of the degree to which the eigenvalue equation $Ax = \lambda x$ is satisfied by an approximate eigenpair of the normal matrix from the conjugate gradient/Lanczos algorithm. The normal matrix A corresponds to the fourth round of alternation for the *source*. Solid line: λx where λ is an approximation to the *smallest* eigenvalue (λ_n=.1819164) of the normal operator (and x the corresponding eigenvector). Dashed line: Ax.