

SACLANTCEN MEMORANDUM
serial no.: SM-333

**SACLANT UNDERSEA
RESEARCH CENTRE
MEMORANDUM**



**SAGA USER MANUAL 2.0:
AN INVERSION SOFTWARE PACKAGE**

P. Gerstoft

July 1997

The SACLANT Undersea Research Centre provides the Supreme Allied Commander Atlantic (SACLANT) with scientific and technical assistance under the terms of its NATO charter, which entered into force on 1 February 1963. Without prejudice to this main task – and under the policy direction of SACLANT – the Centre also renders scientific and technical assistance to the individual NATO nations.

This document is approved for public release.
Distribution is unlimited

SACLANT Undersea Research Centre
Viale San Bartolomeo 400
19138 San Bartolomeo (SP), Italy

tel: +39-187-540.111
fax: +39-187-524.600

e-mail: library@saclantc.nato.int

NORTH ATLANTIC TREATY ORGANIZATION

SAGA User Manual 2.0: An inversion software package

Peter Gerstoft

The content of this document pertains to work performed under Project 042-4 of the SACLANTCEN Programme of Work. The document has been approved for release by The Director, SACLANTCEN.



J.L. Spoelstra
Director

intentionally blank page

**SAGA User Manual 2.0:
An inversion software package**

Peter Gerstoft

Executive Summary: The use of full field inversion methods or matched field processing for target localization has been shown to be effective for determination of target bearing, range, and depth in deep and shallow water. The performance of these methods is strongly dependent on accurate information about the environmental parameters. Previously, the lack of this knowledge inhibited the application of these methods in shallow water.

The objective of this report is to document a general software package SAGA (Seismo Acoustic inversion using Genetic Algorithms) which has been developed for assisting in estimating environmental parameters. The approach taken here is data-independent and therefore the SAGA inversion code has been applied to many types of data: single and multi frequency data on a vertical array, coherent and incoherent transmission loss, reverberation data, reflection coefficients from the sea bottom, and tropospheric electromagnetic data.

Global optimization using a directed Monte Carlo (random) search based on genetic algorithms is used to identify representative set of parameters. Genetic algorithms are based on an analogy with biological evolution, one of the most efficient optimization systems. Both the maximum likelihood estimate as well as uncertainty estimates are provided. The results are presented using a Bayesian framework. This approach is also applicable to data fusion, i.e. when combining information from several sensors.

intentionally blank page

**SAGA User Manual 2.0:
An inversion software package**

Peter Gerstoft

Abstract: The purpose of many experiments is to identify environmental parameters which, when used as input to a forward model, accurately model the observed data. It is also useful to have some indication of the uncertainty of the estimated parameters. SAGA is a software package that helps the user determine the best set of parameters to match a given data set. At present, the package consists of seven modules, one for each forward model. The forward codes are: OAST (wavenumber integration transmission loss model), OAST (wavenumber integration reflection coefficients model), SNAP (normal modes), SNAPRD (adiabatic normal modes), POPP (normal mode reverberation model), PROSIM (broadband adiabatic normal modes), and TPEM (tropospheric parabolic equation). Several types of observed data can be used in these inversions: single or multi-frequency pressure on a vertical array, coherent or incoherent transmission loss, reflection coefficients, reverberation data, or tropospheric electromagnetic data. For any parameter estimation problem the issue of error assessment must be addressed. In SAGA it is addressed by estimating *a posteriori* distributions.

The word SAGA stems from Icelandic and simply means a long story, usually from the middle ages. It tends to go on forever. Scientifically it stands for Seismo-Acoustic inversion using Genetic Algorithms.

Keywords: genetic algorithms – inversion – optimization – SAGA

Contents

1	Introduction	1
2	Background	4
2.1	Optimization using Genetic Algorithms	4
2.2	<i>A posteriori</i> statistics	7
3	Cook book for inversion	10
4	Installing SAGA	13
4.1	SAGA directory	13
4.2	Loading SAGA files	13
4.3	SAGA platforms	14
4.4	Building SAGA	14
4.5	SAGA example files	15
5	SAGA: General features	17
5.1	Graphics	18
5.2	Useful scripts	19
6	SAGA: The input file	20
6.1	SAGA options	20
7	Format of the observed data	26
7.1	Covariance matrix file	26
7.2	General data input (option d)	27
7.3	Range data input (option D)	27
7.4	Vertical array data (option e)	27
7.5	Weighting file (option M)	28
8	The objective function	29
8.1	Transformation of data	29
8.2	Direct observations	30
8.3	Covariance matrix	31
9	Shape functions	32
10	Output files	35
10.1	The *.mat, *b.mat and *.ext files	35
11	Forward models and examples	36
11.1	Forward model: OAST	36
11.2	Forward model: OASR	44
11.3	Forward model: OASTG	45
11.4	Forward model: SNAP	46

11.5 Forward model: SNAPRD	53
11.6 Forward model: PROSIM	59
11.7 Forward model: POPP	63
11.8 Forward model: TPTEM	68
References	75
Annex A - Likelihood functions for a vertical array	80
A.1 Multi-frequency matched field processing	80
A.2 Multi-frequency matched mode processing	82
A.3 Estimation of the covariance matrix	83
Annex B - Regularization	85
B.1 Tikhonov regularization	85
B.2 Including the <i>a priori</i> probability distribution	86
Annex C - Updating SAGA	88
C.1 Porting a forward model into SAGA	88
C.2 Introducing a new option	90

1

Introduction

SAGA, Fig. 1, is a computer code for inversion of observed data. Its purpose can best be understood by dividing the inversion process into five parts:

- (I) Discretization of the environment and discretization or transformation of the data.
- (II) Efficient and accurate forward modeling.
- (III) A suitable objective function.
- (IV) Efficient optimization procedures.
- (V) Uncertainty analysis.

Item (I) is concerned with how to collect and discretize a wave field in order to have the necessary information for the inversion, and to determine the parameters for which inversion is feasible. Item (I) leads to a set of known environmental parameters and *a priori* bounds for the unknown parameters. Based on the above parameters, a replica field can be computed by a forward acoustic model, item (II). The observed data and the replica are then compared through an objective function, item (III). Through an iterative scheme, item (IV), the match between the observed and computed data is maximized by varying the environmental parameters. From the best models obtained, it is possible to provide estimates of the value of the parameters and their uncertainty and importance, item (V). The best solution is not very interesting without a proper statistical analysis of the result.

Complete inversion requires equal attention to all five items, as illustrated in Fig. 2. It is also clear that each item depends on its predecessor. Therefore it is natural that earlier research has focused on the first two items. The present code is concerned with the solution of items (III), (IV) and (V).

The forward models, item (II), presently used are: **SNAP** (normal modes) and **SNAPRD** (adiabatic normal modes) [1], **OAST** (wavenumber integration transmission loss model) and **OASR** (wavenumber integration reflection coefficients model) [2, 3], **POPP** (normal mode reverberation model) [4], **PROSIM** (broadband adiabatic normal modes) [5, 6, 7], and **TPEM** (tropospheric parabolic equation) [8, 9].

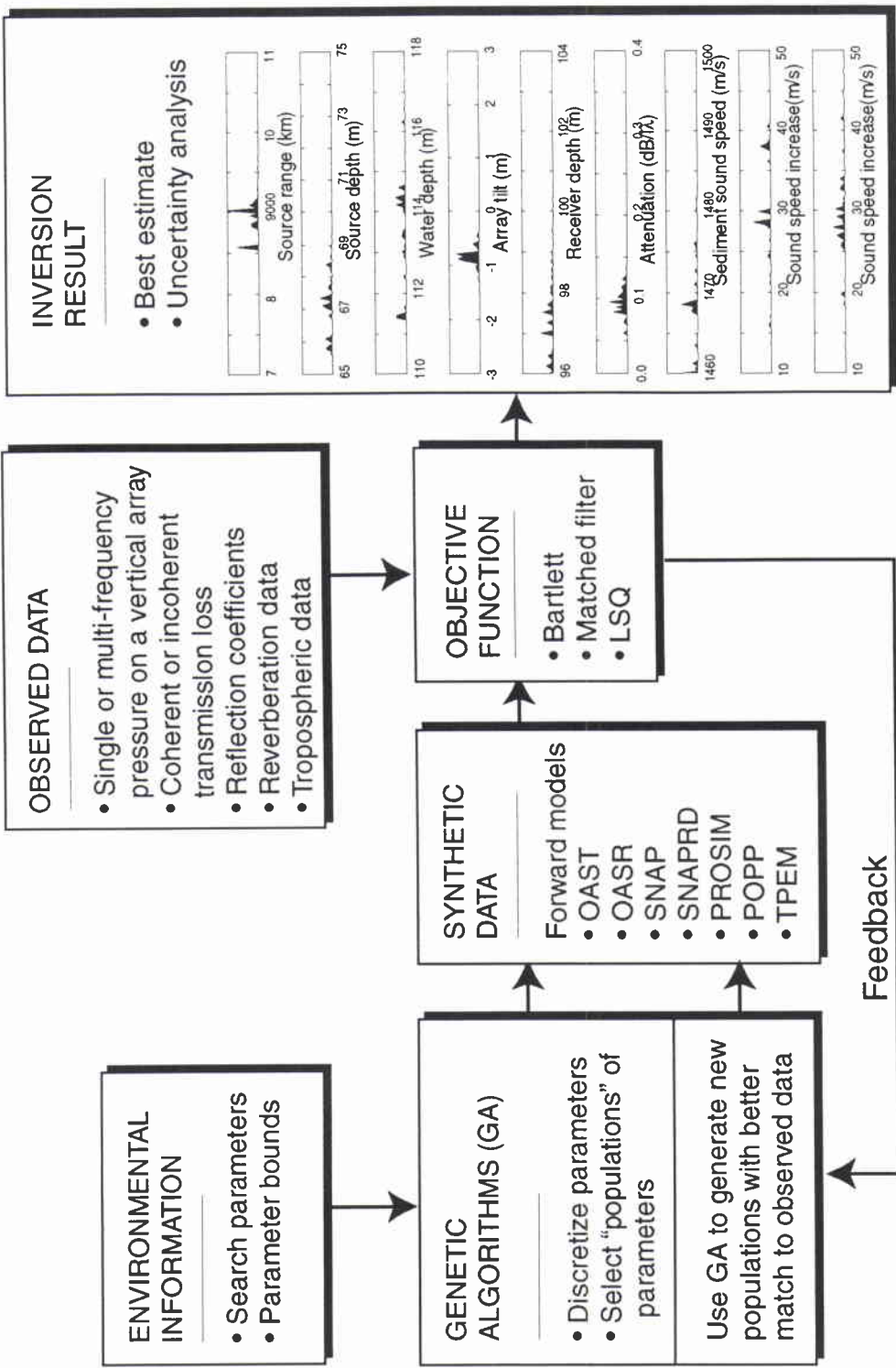


Figure 1 Flow diagram for the SAGA code.

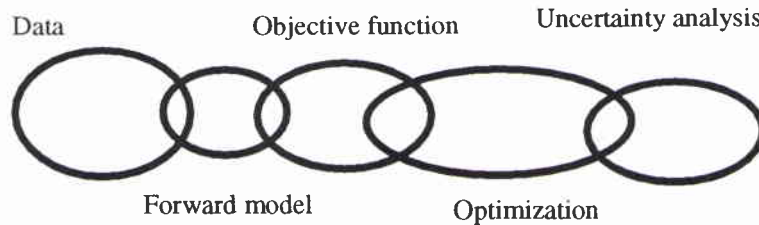


Figure 2 *The weakest link determines the outcome of an inversion procedure.*

The optimization, item (III), is based mainly on genetic algorithms (GA) [10], but also on Gauss-Newton (GN), a hybrid combination of GA and GN as described in [10, 11] and simulated annealing (SA). To ensure convergence to the global optimum, we use several populations in parallel for the genetic algorithms. The search efficiency can be increased by combining the GA and GN methods. This task is handled by the optimization module SAGA.

Two versions of SA are available: Fast SA (FSA) as described in [12] and Very Fast SA (VFSA) as described in [13, 14]. The simulated annealing routines are provided so that the user can assess the estimates obtained using different optimization methods. The SA methods do not provide an estimate of *a posteriori* distributions.

Analysis of the solution, item (IV), is achieved by examining *a posteriori* distributions with the post-processor module POST. The solution can also be assessed by local methods such as singular-value decomposition [10, 15, 16], or by plotting the ambiguity surface.

From the above description it is clear that inversion is a complex procedure and in order to be successful many aspects have to be considered: Forward modeling [17], signal processing [18, 19], optimization and estimation theory [20], global optimization [24], deterministic discrete inversion [21], deterministic continuous inversion [22], and stochastic inversion [23].

2

Background

The non-linear inverse problem is stated as an optimization problem: Find the model vector $\mathbf{m}$, or parameter set, that minimizes the objective function $\phi = f(\mathbf{p}, \mathbf{q}(\mathbf{m}))$, where $\mathbf{p}$ is the observed data and $\mathbf{q}$ is the synthetically generated data using a given forward model with a set of physical parameters $\mathbf{m}$. Normally, in genetic algorithms the objective function is maximized, but here it is minimized, in accordance with the precept of simulated annealing.

Global optimization methods accept that the objective function is irregular and attempt to find the global minimum, without an exhaustive search. An advantage of global optimization is that it requires only the value of the objective function at arbitrary points in space. The problem can then be solved without further knowledge of the objective function. Thus, once the global inversion method has been tuned, any forward model can be used. Early solutions to the global problem were attempted using a simple Monte Carlo method, whereas modern methods use directional searches such as *genetic algorithms* (GA) or *simulated annealing* (SA). The SA method has been applied to ocean acoustics in [12, 25, 26, 27].

2.1 Optimization using Genetic Algorithms

Genetic algorithms are analogous to biological evolution. They have been applied to seismics [28, 29, 30, 31], and ocean acoustics [10, 32, 33, 34, 35]. The basic principle of GA is simple: From all possible model vectors, an initial population of q members is selected. The fit of each member is computed based on the fit between the observed data and the computed data. Then through a set of evolutionary steps, the initial population evolves in order to become more fit. An evolutionary step consists of selecting a parental distribution from the population based on the individual's fit. The parents are then combined in pairs and operators are applied to them to form a set of children. Traditionally the crossover rate and mutation rate operators have been used. Finally, the children replace part of the population to increase the match between observed and synthetic data.

The environment is discretized into M parameters in a model vector $\mathbf{m}$. Each parameter m_j , $j = 1, \dots, M$, can assume 2^{n_j} discrete values according to an *a priori*

	*	*	*	*	*	*	*	*	
$i_j =$	0	0	0	0	0	0	0	0	$m_j^{\min}$
$i_j =$	0	0	0	0	0	0	0	1	$m_j^{\min} + 1 \Delta m$
$i_j =$	0	0	0	0	0	0	1	0	$m_j^{\min} + 2 \Delta m$
$i_j =$	0	0	0	0	0	0	1	1	$m_j^{\min} + 3 \Delta m$
$i_j =$	1	1	1	1	1	1	1	1	$m_j^{\max}$

Figure 3 Binary coding of model parameters.

probability distribution (Gaussian, rectangular or based on *a priori* information). Here a rectangular distribution between a lower and upper bound $[m_j^{\min}, m_j^{\max}]$ is used. We have, see Fig. 3,

$$m_j = m_j^{\min} + i_j \Delta m_j, \quad i_j = 0, \dots, 2^{n_j} - 1, \quad (1)$$

where

$$\Delta m_j = \frac{m_j^{\max} - m_j^{\min}}{2^{n_j} - 1}. \quad (2)$$

A major difference between simulated annealing and GA is that GA uses q model vectors at the same time, where q is the population size, while simulated annealing only uses one. GA consists essentially of three operators: selection, crossover and mutation.

Selection: In order to establish a new population, also with q members, $f q$ parents must be selected, $0 < f < 1$. The choice is made with a probability proportional to the fit of its members. The simplest probability is given by

$$p_k = \frac{1 - \phi(\mathbf{m}_k)}{\sum_{l=1}^q [1 - \phi(\mathbf{m}_l)]}, \quad k = 1, \dots, q. \quad (3)$$

The introduction of a temperature, as in simulated annealing, gives us the opportunity to stretch the probability and improve the algorithm performance [30]. Indeed, at the first stage of the procedure, by stretching the fit we avoid to choose as parents only the members with the better fit, which would otherwise tend to dominate the population; later in the optimization, this stretching leads to a better discrimination between models with very close fit. In order to take into account this new parameter, the probability is rewritten as

$$p_k = \frac{\exp[-\phi(\mathbf{m}^k)/T]}{\sum_{l=1}^q \exp[-\phi(\mathbf{m}^l)/T]}, \quad k = 1, \dots, q. \quad (4)$$

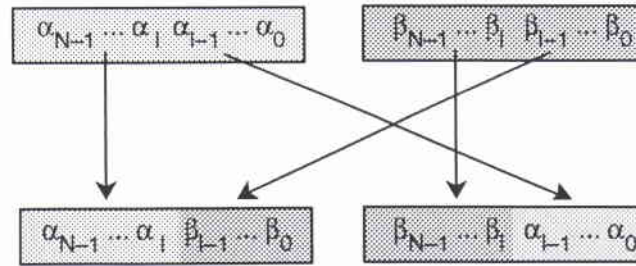


Figure 4 Crossover is a binary exchange of l bits between the binary codes for two model parameters. l is chosen randomly.

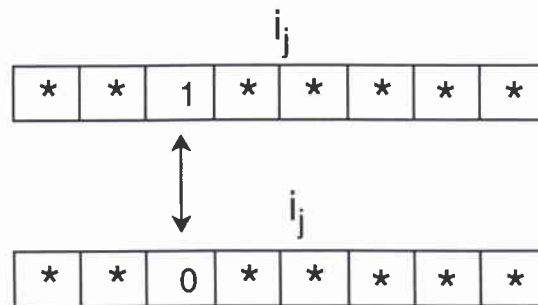


Figure 5 Mutation is a random change of one bit.

But, as with simulated annealing, the choice of the temperature T is difficult. It must be neither too high nor too low. A good compromise is a temperature of the same magnitude as the objective function, here $T = \min[\phi(\mathbf{m}^k)]$. During optimization, the fit increases and temperature decreases.

Crossover For each set of parents, each consisting of a model vector, two children are constructed, and for each parameter in the model vector, each child may either be a direct copy of one parent, with probability $1 - p_x$, or it can be a bit crossover of the two parents with crossover probability p_x , see Fig. 4. The crossover point is chosen randomly in the interval $[1, N - 1]$, where N is the number of bits used in the coding. Different techniques are available to perform this crossover of the population, e.g., single point crossover where the entire “chromosome” is used once, and the multiple point crossover where the chromosome is divided into genes related to each parameter on which the crossover is applied. Multiple point crossover is used here.

Mutation: This is a random change of one bit in the model vector, with probability p_m in order to better explore the search space (see Fig. 5).

It is possible that a run of a GA will approach a local minimum. In order to increase

the probability of finding the global minimum, several independent populations M_{par} are started. This is also advantageous for collecting statistical information in order to estimate the probabilities, as shown in the examples in the next section.

In the present implementation there are relatively few GA parameters which in principle have to be tuned for each application, and the precise value of each of these seems to be not that important. We usually only vary M_{par} in order to adjust the execution time to the available CPU time. Based on our experience, the following values are recommended:

- The population size q should be large enough to allow the model vectors to represent several minima, but also small enough to allow several iterations to be performed; $q = 64$ seems to be a good compromise.
- The reproduction size f should be large enough to allow the fittest individuals to stay in the population during the iterations; f should be less than 0.9, $f = 0.5$ is recommended.
- The crossover rate depends on how independent the parameters in the model vector are. A crossover rate p_x close to 1.0 seems to be a good choice for independent parameters; for dependent parameters a lower value, e.g. $p_x = 0.8$, is recommended.
- It has been found that a high mutation rate gives the best result, $p_m = 0.05$ is recommended.
- The number of forward computations N_{forw} for each population should be relatively low ($N_{\text{forw}} = 1000\text{--}5000$).
- The number of parallel populations N_{pop} depends on the application. To obtain a reasonable estimate of the inversion parameters, $N_{\text{pop}} = 1$ is sufficient. For computing the probability distribution it must be larger, e.g. $N_{\text{pop}} = 50$.

2.2 *A posteriori statistics*

Before measurement, the information about the models is reflected in the *a priori* distribution $\rho(\mathbf{m})$, and after the experiment, the information about the models is reflected in the *a posteriori* distribution $\sigma(\mathbf{m})$. These distributions are related through the likelihood function $\mathcal{L}(\mathbf{m})$, which is a measure of the goodness of fit between the observed data and the data generated using a physical model and the environment $\mathbf{m}$ (Bayes Theorem):

$$\sigma(\mathbf{m}) = \mathcal{L}(\mathbf{m})\rho(\mathbf{m}). \quad (5)$$

When maximizing $\sigma(\mathbf{m})$ the *Maximum A Posteriori* (MAP) estimate of the parameters is obtained and when maximizing $\mathcal{L}(\mathbf{m})$ a *Maximum Likelihood* (ML) estimate of the parameters is obtained.

Due to multi-dimensionality, the *a posteriori* distribution, often with $M > 10$, is not suitable for graphic display, and therefore mainly integral properties of this distribution are of interest. From the *a posteriori* probability distribution, information will be extracted to describe the solution. The following quantities are of interest: The MAP solution $\hat{\mathbf{m}}^{\text{MAP}}$

$$\hat{\mathbf{m}}^{\text{MAP}} \equiv \arg \max_{\mathbf{m} \in \mathcal{M}} \sigma(\mathbf{m}) , \quad (6)$$

the expectation $E_\sigma[\mathbf{m}]$

$$E_\sigma[\mathbf{m}] \equiv \int_{\mathcal{M}} \mathbf{m} \sigma(\mathbf{m}) d\mathbf{m} , \quad (7)$$

where $d\mathbf{m} = dm^1 \dots dm^M$. The covariance matrix $\text{Cov}_\sigma[\mathbf{m}]$

$$\text{Cov}_\sigma[\mathbf{m}] \equiv E_\sigma \left\{ [\mathbf{m} - E_\sigma(\mathbf{m})][\mathbf{m} - E_\sigma(\mathbf{m})]^T \right\} , \quad (8)$$

the 1-D marginal *a posteriori* probability densities $\sigma^i(m^i)$ for parameter m^i

$$\sigma^i(m^i) \equiv \int \sigma(\mathbf{m}) dm^1 \dots dm^{i-1} dm^{i+1} \dots dm^M , \quad (9)$$

and higher-dimensional marginals are defined similarly to Eq. (9). The marginal distributions are the most important in interpreting the inversion result.

A problem with this approach is the derivation of a likelihood function [36]. The likelihood function depends on the stochastic model for the data, an example is given in Annex A. The standard approach in SAGA is to estimate empirically the likelihood function. Knowing that the likelihood function is usually related to the objective function $\phi(\mathbf{m})$ through an exponential $\mathcal{L} = \exp(-\phi(\mathbf{m})/\hat{\nu})$ [23], where $\hat{\nu}$ is the estimated noise power, the following scaling was used:

$$\mathcal{L}_{\text{emp}}(\mathbf{m}) = \exp(-[\phi(\mathbf{m}) - \phi(\mathbf{m}_0)]/T) , \quad (10)$$

where ϕ is any objective function and $\mathbf{m}_0$ is the estimated parameter vector corresponding to the optimal value of the objective function. T is the “temperature”. Experimentally, it was found that a good value for T is the average of the 50 best objective functions obtained during the optimization, minus the best value of the objective function. It should be noted that this value of T is not intended to estimate the noise, but rather to produce a reasonable value with which to estimate the uncertainties of the parameters. The advantage of this scheme is that it works irrespective of the stochastic model for the data or likelihood function.

During the optimization, all obtained samples of the search space are stored, and used to estimate the *a posteriori* probabilities using importance sampling [36].

For the N_{obs} observations, the *a posteriori* probability for the k th model vector is estimated by

$$\hat{\sigma}(\mathbf{m}_k) = \frac{\mathcal{L}(\mathbf{m}_k)\rho(\mathbf{m}_k)}{\sum_{j=1}^{N_{\text{obs}}} \mathcal{L}(\mathbf{m}_j)\rho(\mathbf{m}_j)} . \quad (11)$$

For the i th parameter m^i in the model vector the marginal probability distribution for obtaining the particular value κ can be found by summing Eq. (11):

$$\hat{\sigma}^i(\kappa) = \sum_{k=1}^{N_{\text{obs}}} \hat{\sigma}(\mathbf{m}_k) \delta(m_k^i - \kappa) , \quad (12)$$

where δ is the delta-function. In this particular implementation, several independent GA searches are started in parallel. It was found that when executing several parallel runs, it is sufficient to save the last model vectors in a population within each run [10].

3

Cook book for inversion

The following is a check list for doing inversions with **SAGA**:

1. Generate a baseline environmental model using all available information.
2. Decide on one or two appropriate forward models.
3. Perform a few forward runs based on the baseline model to identify the main propagation characteristics. For the inexperienced **SAGA** user it is probably advantageous to use a stand-alone version of the forward model.
4. Decide on type of observed data and **SAGA** data format.
5. Decide on objective function and scaling of observed data and replica.
6. Generate the data synthetically using the following procedure:
 - (a) Specify option W in the input file.
 - (b) Delete the *.in file.
 - (c) Run **SAGA**. It will generate a *.obs file containing synthetic data based on the environment in the *.dat file. It will stop when the data has been generated because there is no observed data (*.in file).
 - (d) Copy this file (*.obs) into the *.in file. **SAGA** always reads the observed data from the *.in file.
7. Make a short **SAGA** run using only a few hundred forward model calls, and invert only for a few environmental parameters.
8. Is the forward model sufficiently fast, i.e. a runtime about 1 second, or is it necessary to simplify the environment/forward model. Does it seem to produce reasonable results?
9. Genetic algorithms parallel evolution in that it is an indeterminate process. This is, however, impractical and the program stops when a certain number of forward solutions have been generated. The number of forward model runs is the product of the number of populations, N_{pop} , and number of forward model runs in a population, N_{forw} . Usually from 1,000–40,000 forward model

runs are carried out. The number of forward runs depends on available CPU-time, the information content in the data and the number of parameters to be estimated.

10. Generate a larger inversion example with more forward modeling runs and environmental parameters.
11. Use the post-processor program, **POST**, to analyze the results.

POST displays the best, maximum, and mean parameter vectors. These are: the parameter vector that gave the best match between the observed and calculated data; the parameter vector that corresponds to the maximum of the *a posteriori* probability distribution; and the parameter vector that corresponds to the mean of the *a posteriori* distribution.

In addition, the normalized standard deviation for each parameter i is computed, $\sigma_i / (F_{\max}^i - F_{\min}^i)$. This gives an indication of the spread of a distribution for each parameter. As a limiting case for a flat distribution, we obtain a normalized standard deviation of $1/(2\sqrt{3}) = 0.29$. For a distribution that is flat in half of the search interval, we obtain $1/(4\sqrt{6}) = 0.10$.

12. Plot the results from **POST**. This is best done using **MATLAB**. This will show the marginal *a posteriori* distributions. If option p is specified, a plot of the observed and synthetic data for the best environment is also shown.
13. Check if some parameters are coupled, i.e. depend on each other. This can be investigated using one of the following approaches:
 - (a) Stochastic approach. Plot the 2-D marginal *a posteriori* distribution between two parameters. The approach is similar to when the 1-D *a posteriori* distributions are plotted. First edit the **SAGA** option line in the *.dat file to include option m45 to compute the 2-D marginal *a posteriori* distribution between, e.g., parameters 4 and 5, and then run **POST**.
 - (b) Deterministic approach. Plot the ambiguity function between two parameters. The remaining parameters are kept at their nominal values. Edit the **SAGA** option line to include option C, and then run **SAGA**.

The result of both approaches is a contour plot. This can be plotted using **CPLLOT**, see Sect. 5.1.

14. In the event some of the parameters are coupled the inversion can be made more efficient by reparameterization using shape functions. This is option E, see also the examples in Sect. 9.
15. Are some parameters unimportant? It is not necessary to optimize unimportant parameters, or maybe they can be combined with other parameters to a single significant parameter. In the latter case this can be accomplished by the use of shape functions. This is option E, see also the examples in Sect. 9.

16. When the synthetic inversion appears to make sense, it is time to start on the actual data collected at sea.
17. It is assumed here that good data are available. Better data result in better and more robust inversions. Some assistance in transforming the data into SAGA format is available from the MATLAB files.
18. Start observed data inversion with a short run to ensure that every thing works well.
19. There is no way to automatically parameterize an environment, and repeated trial and error is the suggested approach. In order to check the robustness of the discretization of the environment, several environments should be tried out.
20. That's almost it. If the results look good, congratulations! Otherwise go back to step 1 and try again.

4

Installing SAGA

4.1 SAGA directory

The **SAGA** home directory is defined on UNIX systems by the following command in your `.login`:

```
setenv SAGADIR your-SAGA-root-directory
```

This could e.g. be `$HOME/saga`. For execution of **SAGA** you must include the scripts and executables in your search path, i.e. `$SAGADIR/bin`. The following line should be inserted in your `.login`:

```
set path = ($SAGADIR/bin $path)
```

4.2 Loading SAGA files

For users running UNIX, the whole directory tree is provided in a compressed **tar** file with the name

```
saga.tar.Z
```

Place this file in your desired root directory `$HOME` and issue the commands:

```
uncompress saga.tar.Z
```

```
tar xvf saga.tar
```

which will install the directory tree:

<code>\$SAGADIR</code>	SAGA root directory
<code>\$SAGADIR/src</code>	Source files for SAGA
<code>\$SAGADIR/src/snap</code>	Source files for SNAP range independent
<code>\$SAGADIR/src/snaprd</code>	Source files for SNAPRD adiabatic modes
<code>\$SAGADIR/src/oases</code>	Source files for OASES
<code>\$SAGADIR/src/popp</code>	Source files for POPP
<code>\$SAGADIR/src/tpem</code>	Source files for TPEM
<code>\$SAGADIR/src/prosim</code>	Source files for PROSIM
<code>\$SAGADIR/src/obj</code>	Object and library files
<code>\$SAGADIR/saga/bin</code>	SAGA scripts and destination of executables
<code>\$SAGADIR/examples</code>	Data for inversion with SAGA
<code>\$SAGADIR/saga/doc</code>	This document in $\LaTeX$ and Postscript format
<code>\$SAGADIR/mat</code>	Some MATLAB files
<code>\$SAGADIR/util</code>	Some utility programs

4.3 *SAGA platforms*

SAGA has been compiled on VMS, Digital UNIX, Linux with ABsoft 3.4 Fortran compiler, Linux with Gnu g77-0.58 Fortran compiler (the compiler does not yet support Fortran structures and thus **TPEM** was not compiled), SunOS, and SGI-unix.

4.4 *Building SAGA*

Set your default directory to `$SAGADIR/src` and edit the **makefile**. Set the definition of the Fortran compiler and your desired directories for libraries and executables (typically `$SAGADIR/bin`).

You should decide which forward models you would like to compile. This can easily be selected by commenting them in the “all line” in the **makefile**. Also the Fortran compiler options should be modified in the **makefile** for each forward model. For production runs, it is advisable to compile with a high level of optimization and without array checking.

After performing the changes, compile and link by issuing the command:

```
make all
```

which will generate all **SAGA** modules.

If the default parameter settings are too large or too small to run a given problem, the parameters may be altered in the parameter include file **comopt.h**. Sometimes

the parameters controlling the forward modeling routines should also be changed. See the manuals of the respective forward codes for details.

4.5 *SAGA example files*

The examples are located in the directory \$SAGADIR/examples. Most of the examples are described later in the text. A short description of each example follows, the CPU time is for a DEC AlphaStation 600-5/266.

4.5.1 *Forward model OAST*

layer11: Described in Sect. 11.1.4. CPU time: 11 s.

tellaro_oast: For comparison with **tellarosnap**. CPU time: 19 min.

simpleoast and **simplenap:** Simple input files to compare data generated from either OAST or SNAP and inverted using the other model.

4.5.2 *Forward model OASR*

rfl: Described in Sect. 11.2.2. CPU time: 15 s.

4.5.3 *Forward model SNAP*

sspmisa: Described in Sect. 11.4.4. The example is from the 1993 NRL workshop [37]. CPU time: 8 min.

tellarosnap: Described in Sect. 11.4.5. CPU time: 25 min.

genlmisa_ga: The example is from the 1993 NRL workshop [37]. CPU time: 8 min.

elba: Described in [38], it is similar to the SNAPRD example. CPU time: 4 min.

simplenap and **simpleoast:** Simple input files to compare data generated from either SNAP or OAST and inverted using the other model.

ys3: Described in Sect. 11.4.6. This example is using matched field inversion on a vertical array based on seven frequencies. The data is from the Yellow Shark experiment. CPU time: 4 h.

4.5.4 Forward model *SNAPRD*

elbard: Described in Sect. 11.5.3. CPU time: 27 min (three frequencies).

shot05: Described in Sect. 11.5.4. This example is using matched field inversion on a vertical array based on 25 frequencies.

tl_malta_rd: Described in Sect. 11.5.5. The data is range-dependent transmission loss at several frequencies and depths from the Malta experiment. CPU time: 28 h.

4.5.5 Forward model *PROSIM*

waa: Described in Sect. 11.6.3. The data is from the 1997 Matched Field Inversion Workshop. CPU time: 3 h.

mf: Described in Sect. 11.6.4. This is an example of using matched filter technique.

4.5.6 Forward model *POPP*

revpopp: Described in Sect. 11.7.2. CPU time: 10 min (one frequency).

rev_grad3: Described in Fig. 20. It uses reverberation data from six frequencies. CPU time: 12 h.

4.5.7 Forward model *TPEM*

tpem_ex: Described in Sect. 11.8.2. CPU time: 10 h.

tpem_rd: An example of a run with range dependent terrain profile.

runs_IEEE96: This directory contains the runs used to generate the figures in Ref. [39]. Note, that because the forward model has changed since the example was generated they cannot be reproduced precisely.

5

SAGA: General features

The inversion package consists of several programs. The command to run SAGA is:

```
saga filename forward-model
```

at present forward-model can be either oast, oasr, oastg, snap, snaprd, popp, prosim, tpem. The filename should be without extension.

The main program stores information which can be processed by the post-processor. The post-processor computes *a posteriori* probability densities, prepares plots of observed data and of data obtained from the inversion. The run command is:

```
post filename forward-model
```

File names are passed to SAGA and POST via environmental parameters. In UNIX systems a typical command file **saga** (in \$SAGADIR/bin) is (and a similar one for **post**):

```
#               the hash mark invokes the C-shell
setenv FOR001 $1.dat # input file
setenv FOR002 $1.in  # observed data input file
setenv FOR003 $1.wei # weighting of the data
setenv FOR007 $1.out # output file
setenv FOR009 $1.eof # shape function file
setenv FOR010 $1.mat # The best obtained parameters and
                    # objective function for each population.
setenv FOR011 $1b.mat # The $1.mat sorted according to fit (POST)
setenv FOR011 $1.m    # plotting parameters used by plotsaga (POST)
setenv FOR019 $1.plp  # plot parameter file
setenv FOR020 $1.plt  # plot data file
setenv FOR028 $1.cdr  # contour plot parameter file
setenv FOR029 $1.bdr  # contour plot data file
setenv FOR030 $1.obs  # synthetic generated from the initial environment
setenv FOR060 $1.ext  # same format as $1.mat, for each improved fit

if      ($argv[2] == "snap" ) then
    $SAGADIR/bin/sagasnap          # snap    executable
elseif ($argv[2] == "snaprd") then
```

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

    $SAGADIR/bin/sagasnaprd          # snaprd executable
elseif ($argv[2] == "oast" ) then
    $SAGADIR/bin/sagaoast           # oast   executable
elseif ($argv[2] == "oasr" ) then
    $SAGADIR/bin/sagaoasr          # oasr   executable
elseif ($argv[2] == "oastg" ) then
    $SAGADIR/bin/sagaoastg         # oast gradient executable
elseif ($argv[2] == "popp" ) then
    $SAGADIR/bin/sagapopp          # popp   executable
elseif ($argv[2] == "tpem" ) then
    $SAGADIR/bin/sagatpem          # tpem   executable
elseif ($argv[2] == "prosim" ) then
    $SAGADIR/bin/sagaprosim        # prosim executable
endif

```

Each time POST is executed it updates the files **results** and **results.m** in the current directory.

5.1 Graphics

Standard output files from POST can be plotted using MATLAB. Simply put **\$SAGA/MATLAB** in your MATLAB path, for example in the .login file:

```
setenv MATLABPATH $home/saga/MATLAB
```

start up MATLAB and execute

```
>> plotsaga
```

When plotting from MATLAB, each *a posteriori* distribution is plotted individually and when option **p** is specified a plot of the observed data and data corresponding to the best fit is also provided.

This processing offers more flexibility than the plotting based on the SACLANTCEN FIPLOT package. The MATLAB version is basing its plot on the *.plt file (data-values) and a *.m file (parameter values). If MATLAB is available it is strongly recommended to use this.

The contour plots are still based on the SACLANTCEN contour package by using the command:

```
cplot file-name
```

The plot files are based on the SACLANTCEN FIPLOT and CPLOT programs.

These programs are further described in Refs. [2, 40]. The **SAGA** output files used by the **FIPLOT** program are *.plt (data values) and *.plp (parameter values), and for **CPLOT** they are *.bdr (data values) and *.cdr (parameter values). These programs are available as part of the **OASES** package (anonymous ftp to [keel.mit.edu](ftp://keel.mit.edu)) in a UNIX version. The files *.m and *.mat from the program can be plotted using **MATLAB**.

5.2 Useful scripts

Some useful UNIX-scripts are provided in the \$SAGADIR/bin:

zap name

Kills all processes that has **name** in their command-field, e.g. **zap saga**.

zapnice name

For all processes with **name** in their command-field the CPU-priority is reduced by one.

zapmax name

For all processes with **name** in their command-field the CPU-priority is reduced to the minimum.

6

SAGA: The input file

The structure of the input file is given in Table 1, the file must have the extension *.dat.

6.1 SAGA options

The following options are supported by SAGA. Each option should be separated by a blank space. The pointer (*iopt*) is used internally in the program; it is of use only to the programmer.

FORWARD MODEL

(the forward model is specified at execution time, the following flags are set automatically):

- (*iopt*(30)=1) SNAP is used.
- (*iopt*(30)=2) SNAPRD is used.
- (*iopt*(30)=3) OAST is used.
- (*iopt*(30)=4) OASR is used.
- (*iopt*(30)=5) POPP is used.
- (*iopt*(30)=6) TPEM is used.
- (*iopt*(30)=7) PROSIM is used.

(*iopt*(12)=1) 3 indexes are used for pointing to the optimization parameters (used in SNAPRD, TPEM, PROSIM).

(*iopt*(12)=0) 2 indexes are used for pointing to the optimization parameters (used in SNAP, OASR, OAST, POPP).

INVERSION DOMAIN

(default *r*)

- w* (*iopt*(1)=1) (for OAST) inversion in wavenumber–depth domain.
- r* (*iopt*(1)=2) (for OAST) inversion in range–depth domain.
- (*iopt*(1)=3) (for OASR, automatic) inversion in angle domain.
- (*iopt*(1)=4) (for POPP, automatic) inversion in time domain.

Input parameter	Description
Block I: TITLE (1 line)	
TITLE	Title of run
Block II: OPTIONS (1 line)	
A B C ...	Output and computational options, see Sect. 6.1
Block III: GA- PARAMETERS (2 lines)	
niter q npop ... px pu pm	niter: Forward modeling runs for one population q: Population size npop: Number of parallel populations ...: Some options require more parameters (t0,...) px: Crossover rate (typically 0.8) pu: Update rate (typically 0.5) pm: Mutation rate (typically 0.05)
Block IV: FORWARD MODEL PARAMETER (many lines)	
* * * *	This block correspond to the forward model in the same format as given in the users manual for each forward model. See Sect. 11
Block V: OPTIMIZATION PARAMETER (nparm+1 lines)	
nparm parm index Fmin Fmax Ndiscr * * * * * *	nparm: Number of parameters to optimize parm: Pointer to physical parameter, see Sect. 11 index: Addresses a specific variable within a vector (For SNAPRD, PROSIM and TPEM: index represents two variables index , index2 , see Sect. 11) Fmin: Lower bound for the search parameter Fmax: Upper bound for the search parameter Ndiscr: Number of discrete values of the parameter

Table 1 SAGA *input file structure*.

OBSERVED DATA FORMAT

(See also Sect. 7)

(Should always be specified):

- d** (iopt(2)=1) reading data (subroutine `readdat2`). For the format see Sect. 7.2.
- D** (iopt(2)=2) reading data (subroutine `readdata`). This is only for data given as a function of range. The range block in the forward model should then contain **Rmin** and **Rmax** in one line. For the format see Sect. 7.3.
- e** (iopt(2)=3) reading complex pressure vector on a vertical array (subroutine `read_HP`). This option also requires option f or option F for objective function. For the format see Sect. 7.4.
- c** (iopt(13)=1) reading covariance matrix (subroutine `read_cov`). For the format see Sect. 7.1.
- W** (iopt(21)=1) writing calculated data using the baseline model to the *.obs file. The format is determined by the input option d, D, c, e. The *.obs file can then be copied to the *.in file and used as 'observed' data. Even when the input file does not exist and the program aborts, the *.obs file is written, and the *.obs file can then be copied to the *.in file.

SCALING OF COVARIANCE MATRIX

(default is no scaling)

- b** (iopt(20)=1) the covariance matrix is normalized by dividing it with the sum of the diagonal. The maximum obtainable Bartlett power is slightly less than one (depending on the noise in the data).
- B** (iopt(20)=2) the covariance matrix is normalized by the largest eigenvalue of the covariance matrix. Thereby, the maximum obtainable Bartlett power is one.

TRANSFORMATION OF CALCULATED DATA

(default is no transformation)

- s** (iopt(3)=1) weighting replica with $\sqrt{r}$. The observed data should be weighted before running **SAGA**. This option should only be used for transmission loss data. In this way the transmission loss is corrected for geometrical spreading.
- l** (iopt(3)=2) inversion with dB scaled observed and calculated data. The observed data should be expressed as $-20 \log p$.
- R** (iopt(3)=3) weighting replica with $1/\sqrt{r}$. The observed data should be weighted before running the program. Should only be used for transmission loss data. This places more emphasis on matching the closer range of the transmission losses.
- G** (iopt(25)=1) using only the magnitude of observed and calculated data.
- h** (iopt(27)=1) observed and calculated data are normalized to unity. This works only for a local method; for the global method it is defined through the objective function.

M (iopt(3)=4) both data and replica are weighted with the values read from the *.wei data file, see Sect. 7.5.

OBJECTIVE FUNCTION

(See also Sect. 8)

(Should **always** be specified)

Objective function that works with data formats option D, d, e

- N** (iopt(5)=0), Eq. (19), objective function is computed based on the sum of the squared error between the magnitude of the observed and calculated data using complex-valued data (if available). This matches only the shape of one curve using all frequency, range and depth information.
- n** (iopt(5)=1), Eqs. (22, 23), objective function is computed based on the sum of the squared error between the magnitude of the observed and calculated data as a function of range. This matches only the shape of a curve. There are as many curves as frequencies and depths.
- X** (iopt(5)=3), Eqs. (22, 24), objective function is computed based on the sum of the squared error between the magnitude of the observed and calculated data. This matches both the shape and the offset of a curve. There are as many curves as frequencies and depths.

Objective function that works with data format option e, (complex-valued data)

- f** (iopt(5)=4), Eq. (20), incoherent addition of Bartlett power across frequencies using pressure vector on an array(s). At least two phones required.
- F** (iopt(5)=5), Eq. (21), broadband matched filter. Incoherent addition of Bartlett power across phones using pressure vector for several frequencies. At least two frequencies required. This option works best in combination with PROSIM.

Objective function that works with data format option c, (covariance matrix)

- c** (iopt(13)=1), Eq. (25), the objective function is the Bartlett power. By default the Bartlett power for each frequency is summed.
- O** (iopt(18)=1), Eq. (26), the Bartlett power for each frequency is multiplied together.

Objective function is modified, this is independent of data format.

- L** (iopt(9)=1) regularization, see Annex B. This option has influence on the computation of the objective function in SAGA. The objective function then consists of two parts, the standard one measuring the fit of the data and one measuring the deviation from the baseline model. For this option we must first specify for which parameters the *a priori* information should be included. In Block IV, Table 1, after **Ndiscr** a positive number indicates *a priori* information and a negative indicates none.

q (iopt(23)=1) *a priori* information, see Annex B. This option does not affect the computation of the objective function in **SAGA**, but only the computation in **POST**. For this option we must first specify for which parameters the *a priori* information should be included. In Block IV, Table 1, after **Ndiscr** a positive number indicates *a priori* information and a negative indicates none. The *a priori* information used for each parameter is a triangular distribution with maximum at the baseline value and zero at the bounds.

OPTIMIZATION

(default is to use genetic algorithms)

- E** (iopt(17)=1) shape functions is used to describe the environmental input. A shape function file must be supplied, see Sect. 9.
- x** (iopt(24)=1) the baseline model in the input is included as an initial member of the first **t0** populations in the GA optimization.
- a** (iopt(4)=1) using simulated annealing for the optimization. The version is adapted from a code by Mike Collins. **niter** is the number of Metropolis steps, the number of forward models is then **niter*nparm**. Two additional parameters should be specified in the first line of the GA parameter block: **t0** is starting temperature (often 1) and **t1** indicating that quenching is used after iteration **t1**. **Ndiscr** determines the maximum allowed jump (often **Ndiscr** = 1). There is no post-processing when using simulated annealing, only the best model is displayed.
- v** (iopt(4)=3) using very fast simulated annealing (VFSA) for the optimization. The version is adapted from a code by M. Sen and based on the paper by Ingber [14]. It seems to be faster than the simulated annealing version described above. VFSA offers great flexibility in selecting optimization control parameters, but here only the initial temperature can be varied. **niter** is the number of Metropolis steps, the number of forward models is then **niter*nparm**. One additional parameters should be specified in the first line of the GA-parameter block: **t0** is starting temperature (often 1). There is no post-processing when using simulated annealing, only the best model is displayed.
- g** (iopt(4)=2) using Gauss-Newton for the optimization. At present only available using the analytic derivatives **OASTG**. The starting point for the local search is the baseline model given in the input. The stop criteria is that the improvement for each parameter is less than $0.2(F_{\max} - F_{\min})/N_{\text{discr}}$.
- H** (iopt(16)=1) the hybrid optimization scheme is used. At present only available using the analytic derivatives **OASTG**. A forward modeling call consists then of 5 Gauss-Newton steps, see [11].

PLOTS FROM POST

(default is to plot the *a posteriori* probability distribution on a scale corresponding to the search interval using the default weighting. A good data check is obtained by plotting the data as well as the best model using option p. A table of best fit, mean and most likely value for each parameter is given in the output.)

- p** (iopt(10)=1) a line plot of the data (full line) and the calculated data (dashed line) using the best found model. For a covariance matrix as input data, option **c**, we estimate the pressure vector as the first eigenvector of the covariance matrix. For a vertical array the power across the array is plotted.
- A** (iopt(19)=1) the starting model is superimposed on the *a posteriori* probability plot.
- i** (iopt(14)=1) plotting the *a posteriori* probability distributions in physical units, instead of plotting them all on a dimensionless scale from 0 to 1.
- I** (iopt(14)=0) plotting the *a posteriori* probability distributions on a dimensionless scale from 0 to 1. This creates one plot of the distributions.
- m13** (iopt(16)=1) computes the 2-D marginal *a posteriori* density distribution between two parameters, here parameters 1 and 3. Both parameters have to be one digit, i.e. less than 10. The plot is normalized so that the maximum is one. Usually a nice plot is obtained by using the scale 0-0.1. It can be plotted using **C**PLOT.
- u** (iopt(28)=2) the *a posteriori* probability distribution is produced using a uniform weighting of all the observed model vectors. This causes a flatter distribution.
- Pj** (iopt(28)=1) maximum likelihood based post-processing of the *a posteriori* probability distribution. It is based on Monte Carlo integration of the likelihood function. *j* indicates the number of modes used in the likelihood function; if it is not specified then *j* = 5. This requires the use of option c or f as choice of objective function.

PLOTS FROM SAGA:

- C** (iopt(8)=1) contour plot of the objective function versus first and second parameter given in the inversion. The first search parameter specifies the *x*-axis and the second the *y*-axis. The discretization along each axis is specified by **Ndiscr**. The maximum discretization in each direction is 200. It can be plotted using **C**PLOT.

MISCELLANEOUS:

- Q** (iopt(6)=1) debugging. This option writes lots of extra information.

7

Format of the observed data

The data file (*.in) contains information about the observed data. Even for synthetic data, this is the standard mode to pass observations to the **SAGA** program. Data can be written out from the program by using option W. The program supports four formats (**c**, **d**, **D** and **e**) which are described below. For other inversion purposes they might have to be modified.

For each of these files a “!” as the first character of a line means a comment. No blank lines!

7.1 Covariance matrix file

The covariance matrix file stores the covariance matrix calculated from the pressure on a vertical array for a single frequency. The program gives a warning if the first line does not contain the string “Covariance matrix”. The format of the file is:

```

loop over range          ! ranges as appearing in input file
  loop over freq          ! in increasing order
    read(2,*)dummy-title
    read(2,*)dummy-freq
    read(2,*)dummy-ndep
    do idep=1,ndep
      READ(2,*)receiver_depth          !rd
    enddo
    do idep1=1,ndep
      do idep=1,ndep
        READ(3,*)jdum,idum,cov(idep1,idep,ibart)
                                          ! idum, jdum is not used
      enddo
    enddo
  enddo
enddo

```

In this file, the covariance matrix for several frequencies can be stored, not all covariance matrices need to be read in. The program only stores the ones corresponding

to the frequencies to be used in the inversion. The covariance matrices should be given in the same order as the frequencies in the input file.

If the first or last receiver depth does not correspond to the depth in the input file a warning is issued.

7.2 General data input (option d)

This is a general data file for passing data to the **SAGA** program (using subroutine **readdat2**). The format is (data can be complex-valued):

```

loop over freq                ! in increasing order
  loop over depth
    loop over ranges
      read(2,*)idum,jdum,data      ! idum, jdum is not used
    enddo
  enddo
enddo

```

where **ranges** can mean either number of ranges or wavenumbers.

7.3 Range data input (option D)

This is for reading in pressure-data as a function of range (using subroutine **readdata**). Only points in the range **Rmin** to **Rmax** are used. **Rmin** and **Rmax** are given in the input file, units is meters. The format is (data must be real-valued):

```

reads to End_Of_File:
do i=1,10000
  read(2,*) rng, (press(i,j),j=1,ncurv)
enddo

```

where **[1...ncurv]** consists of an outer loop over frequency and an inner loop over depth.

7.4 Vertical array data (option e)

The vertical array file stores the pressure on a vertical array for several frequencies (using subroutine **readHP**). The format of the file is similar to the covariance matrix

format except for the inner do-loop. The program gives a warning if the first line does not contain the string "Hydrophone vectors". The format is:

```

loop over range          ! as appearing in input file
  loop over freq          ! in increasing order
    read(2,*)dummy-title
    read(2,*)dummy-freq
    read(2,*)dummy-ndep
    do idep=1,ndep
      READ(2,*)receiver_depth          !rd
    enddo
    do idep=1,ndep
      READ(2,*)idum,pres(idep,ibart) ! complex-valued pressure
    enddo
  enddo
enddo

```

In this file, the pressure vector for several frequencies can be stored, not all vectors need to be read in. The program only stores the ones corresponding to the frequencies to be used in the inversion.

There is no check when reading the ranges.

If the first or last receiver depth does not correspond to the depth in the input file a warning is issued.

If more receiver depths are specified in the *.in file, then those specified in the input file (*.dat) will be used. Clearly it is required that all the depths requested in the *.dat file are available in the *.in file.

7.5 Weighting file (option M)

The filename is *.wei. The purpose of this is to weight the data according to *a priori* knowledge by a user-specified function, instead of those supplied by the program ($1/\sqrt{r}$, $\sqrt{r}$, $-20 \log p$). The use of a simple function is recommended. The format is:

```

loop over freq
  loop over depth
    loop over range
      read(2,*)idum,weight(i)      ! idum is not used by program.
    enddo
  enddo
enddo

```

8

The objective function

It is important to use the correct Fourier transform pair when transforming observed data from time to frequency domain. If the wrong sign convention is used the inversion will be meaningless. We are using the following transform pair:

$$p(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} P(t) e^{-i\omega t} dt, \quad (13)$$

$$P(t) = \int_{-\infty}^{\infty} p(\omega) e^{i\omega t} dt. \quad (14)$$

One of the most important tasks in carrying out an inversion by means of optimization is to define a proper objective function. Here we have selected some of the most common.

Let p and q be the observed and calculated data, respectively. They are complex matrices and contain a number of frequencies (N_{freq}), depths (N_{dep}) and ranges or wavenumbers (N_x).

8.1 Transformation of data

Before the objective function is calculated it is possible to transform the calculated data to another form that might be more suitable for a particular inversion. The objective function will then be applied directly on the transformed data.

For option G the magnitude of both the computed and observed data is taken

$$p_{ijk} = |p_{ijk}|, \quad q_{ijk} = |q_{ijk}| \quad (15)$$

$$\text{for } i = 1, \dots, N_{\text{freq}}, \quad j = 1, \dots, N_{\text{dep}}, \quad j = 1, \dots, N_x \quad (16)$$

For option I only the computed data are transformed to a dB scale. The observed data are assumed to be already on a dB scale.

$$p_{ijk} = -20 \log |p_{ijk}|, \quad q_{ijk} = -20 \log |q_{ijk}| \quad (17)$$

$$\text{for } i = 1, \dots, N_{\text{freq}}, \quad j = 1, \dots, N_{\text{dep}}, \quad j = 1, \dots, N_x \quad (18)$$

8.2 Direct observations

The most general objective function is option N. $\hat{p}$ and $\hat{q}$ are the data normalized to unity¹:

$$\phi = |1 - \sum_{i=1}^{N_{\text{freq}}} \sum_{j=1}^{N_{\text{dep}}} \sum_{k=1}^{N_x} \hat{p}_{ijk}^* \hat{q}_{ijk}|. \quad (19)$$

This corresponds to the mean squared error.

Option f is the incoherent sum of the Bartlett power for each frequency. Both observed and computed data are based on vectors and not the usual correlation matrix:

$$\phi = 1 - \frac{1}{N_x N_{\text{freq}}} \sum_{k=1}^{N_x} \sum_{i=1}^{N_{\text{freq}}} \frac{|\sum_{j=1}^{N_{\text{dep}}} p_{ijk}^* q_{ijk}|^2}{\sum_{j=1}^{N_{\text{dep}}} |p_{ijk}|^2 \sum_{j=1}^{N_{\text{dep}}} |q_{ijk}|^2}. \quad (20)$$

Option F is a matched filter. We sum incoherently over depth and range, but coherently over frequency.

$$\phi_f = 1 - \frac{1}{N_x N_{\text{freq}}} \sum_{k=1}^{N_x} \sum_{j=1}^{N_{\text{dep}}} \frac{|\sum_{i=1}^{N_{\text{freq}}} p_{ijk}^T q_{ijk}|^2}{\sum_{i=1}^{N_{\text{freq}}} |p_{ijk}|^2 \sum_{i=1}^{N_{\text{freq}}} |q_{ijk}|^2}. \quad (21)$$

Classical matched filter is done in the time domain by correlating the observed time series and the synthetic time series. The above expression in the frequency domain is similar to classical matched filtering.

For option n and X we use only the magnitude of the observations:

$$\phi_{nX} = \sum_{i=1}^{N_{\text{freq}}} \sum_{j=1}^{N_{\text{dep}}} \sum_{k=1}^{N_x} (|p_{ijk}| - A|q_{ijk}|)^2, \quad (22)$$

where for option n we correct for the offset of the curve

$$A_n^2 = \frac{\sum_{i=1}^{N_{\text{freq}}} \sum_{j=1}^{N_{\text{dep}}} \sum_{k=1}^{N_x} |p_{ijk}|^2}{\sum_{i=1}^{N_{\text{freq}}} \sum_{j=1}^{N_{\text{dep}}} \sum_{k=1}^{N_x} |q_{ijk}|^2} \quad (23)$$

and for option X there is no offset correction,

$$A_X = 1. \quad (24)$$

¹* is the conjugate operator

8.3 Covariance matrix

For option c we use the Bartlett power. For a vector of pressure observations at frequency i , at range k , on a vertical array p_{ij} ($j = 1, N_{\text{dep}}$), the covariance matrix for a given frequency and range is constructed $R_{jl,ik} = p_{ijk} p_{ilk}^T$. For observed data, the covariance matrix is based on the ensemble average of the estimated covariance matrix formed for each time snapshot. Some help in the construction of this covariance matrix is available in the MATLAB directory. The MATLAB routines also writes the covariance matrix out in **SAGA** format. This matrix is read from the *.in file. q_{ij} ($j = 1, N_{\text{dep}}$) is a vector of calculated pressure on the vertical array,

$$\phi_c = \sum_{i=1}^{N_{\text{freq}}} \sum_{k=1}^{N_x} \sum_{j=1}^{N_{\text{dep}}} \left[R_{jj,ik} - q_{ijk}^* \sum_{l=1}^{N_{\text{dep}}} R_{jl,ik} q_{ilk} \right]. \quad (25)$$

It can be shown that the above formula is a maximum likelihood estimate if the noise is assumed Gaussian with known or constant mean, see e.g. [36]. If the noise is unknown and frequency dependent then option O provides a maximum likelihood estimate; it is obtained by replacing the summation in the above formula with a product,

$$\phi_c = \prod_{i=1}^{N_{\text{freq}}} \prod_{k=1}^{N_x} \sum_{j=1}^{N_{\text{dep}}} \left[R_{jj,ik} - q_{ijk}^* \sum_{l=1}^{N_{\text{dep}}} R_{jl,ik} q_{ilk} \right]. \quad (26)$$

This corresponds to summing the logarithmic powers (dB). From a practical point of view no significant difference has yet been found between the two objective functions, Eqs. (25) and (26).

9

Shape functions

Here regularization is introduced via shape functions. These can be seen as a coordinate transformation h_{ij} between the input parameters m_i required for the forward modeling program and a more efficient set of parameters μ_j :

$$m_i = \sum_j^{M_s} h_{ij} \mu_j, \quad (27)$$

where h_{ij} is the j th shape function or basis function, μ_j is the coefficient associated with this shape function and M_s is the number of shape functions used. The advantages of shape functions are:

- They constrain the solution to a certain class of expected profiles.
- They describe the variation of the parameters with fewer coefficients and in so doing reduce the number of unknowns. This can also constrain the solution to be more physically correct. For example, by linking the sound speed in the sediment together, we can describe them by a smoother function and, furthermore, reduce the number of unknowns.
- They link correlated parameters, resulting in better search performance.

Regularization using shape functions may decrease the correlation between parameters and this would improve the inversion results. This was done in [41] where we inverted for the slope and the offset of the water sound speed, instead of inverting for the absolute sound speed at discrete points.

For each of these files a “!” as the first character of a line means a comment— There should be no blank lines. The format is given in Table 2. The starting values of the shape function coefficients are given because we do not use the actual values in the forward-model-input block for the parameters described by shape functions.

9.0.1 Example of a shape function file

This example is based on the input file in Sect. 11.4.4, with **SNAP** as the forward model. An option E is then included in the option line of the input file. The example

Input parameter	Description
Block I: DIMENSIONS (1 line)	
Neof Neofvar Nblock	Neof: Number of shape functions Neofvar: Number of points to specify shape function Nblock: Number of blocks to specify shape functions
Block II: PARAMETERS (Neofvar lines)	
parm index	Repeated Neofvar times. parm: Pointer to physical parameter. For key see Sect. 11. index: Addresses a specific variable in a vector (for SNAPRD, PROSIM and TPDM, index refers to 2 variables, index and index2)
Block III-Nblock+IV: SHAPE FUNCTIONS	
NeofBlock NeofvarBlock a1 a2 ... aNeofBlock ...	The coefficients – repeated NeofvarBlock times
Block Nblock+V: starting amplitudes (1 line)	
$\mu_1 \mu_2 \mu_3 \dots$	μ_i is the starting amplitude for the shape function i

Table 2 Shape function file structure.

uses the same pointers as in SNAP, see Sect. 11.4.3.

Instead of the two sound speed points at the top and bottom of the water column (pointers 2-1 and 2-2), we will describe the water sound speed profile using two shape functions. The first shape function (11-1) is constant over depth, and the second shape function (11-2) describes the decrease in sound speed at the bottom. Hereby we have obtained a more efficient parameter set as the gradient is more important than the absolute offset. We will also reduce the number of free parameters in the bottom by binding the lower sediment speed (3-2) to the basement speed (12-1) using one shape function (11-3). In terms of coordinate mapping, Eq. (27), we express this as

$$\begin{bmatrix} 2-1 \\ 2-2 \\ 3-2 \\ 12-1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 11-1 \\ 11-2 \\ 11-3 \end{bmatrix}. \quad (28)$$

Because it consists of two sub-matrices, we describe the mapping matrix using two blocks, Nblock=2. The use of sub-matrices is especially useful for large problems.

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

!Shape function example for sspmis
3 4 2          ! Neof Neofvar Nblock
2 1            ! upper speed in water
2 2            ! Lower speed in water
3 2            ! lower sediment speed
12 1           ! basement speed
! The first block describe the water profile
2 2            ! the size of first sub-matrix
1 0
1 -1
! The second block describe the bottom profile
1 2            ! the size of second sub-matrix
1
1
1500 20 1750   ! starting amplitudes

```

Also the optimization parameter block of the input file must be changed:

```

5              ! nparm
11 1          1497.5 1502.5 51 ! upper sound-speed point
11 2           15    25   101 ! gradient of sound speed
11 3          1600   1800   51 ! basement speed
9 1           5000  10000  51 ! source range
8 1           0.01   100   51 ! source depth

```

10

Output files

The *.out is the default output file. For each execution of POST a summary of the estimated parameters is also added to the files **results** and **results.m** in the current directory. With minor modifications the **results.m** can be used with MATLAB to analyze the convergence of different inversion approaches.

The plot files (*.plt, *.plp, *.bdr and *.cdr) use the standard SACLANTCEN plotting programs, but as described in Sect. 5.1, they are better processed with MATLAB.

10.1 The *.mat, *.b.mat and *.ext files

All three files use the same format:

```
do i=1,q
    write(10,*)i,ipop,fit(i),(model(jn,i),jn=1,nparm)
enddo
```

where the variable **model** contains an integer representation of the model vector for each individual **i**. In total there are **q** model vectors. It is quite easy to read these files into MATLAB and plot the results.

***.mat**: The file is written from SAGA. At the end of each population, the last generation, containing **q** model vectors are written out to the *.mat file.

***.ext**: The file is written from SAGA. Each time an improved fit has been found it is written out. The **q** above is redefined as 1, and the variable **ipop** is redefined as number of forward modeling runs.

***b.mat**: The file is written from POST. It contains a sorted list of all the individuals used in the post-processing. Here **ipop**=1 and **q** is the number of populations times the number of model vectors in a generation.

11

Forward models and examples

A short description of the forward models that are currently incorporated into the **SAGA** program is given. This description requires familiarity with the particular forward code, which is strongly coupled to the program to ensure computational efficiency.

As each forward model is a subroutine it has some limitations relative to the stand-alone version. Firstly, it reflects the forward model at the time of porting. Secondly, the input subroutine is usually rewritten and therefore some options available in the original model might not be available in the **SAGA** version.

11.1 Forward model: *OAST*

The forward model for **OAST** follows *nearly* the format of the **SAFARI** manual [2]. We use the format described in the upgrade notes [3]. But it is based mainly on **OASES** Version 1.2. The source-receiver range can be read in using two formats. See the input examples.

In the transmission loss module of **OASES** the complex pressure is found by using the fast field approximation and then doing a Fourier transform from wavenumber domain to range domain. This has the disadvantage that the pressure is only computed at discrete intervals with a spacing of about one wavelength which causes an inaccurate determination of the phase. Therefore in **SAGA** full integration similar to that in the **OASES** pulse module is used.

It is difficult to determine the correct wavenumber sampling when using several frequencies and when the environment is changing during the optimization. Therefore automatic sampling has been introduced for the transmission loss module based on automatic sampling for the pulse program in **OASES** 2.0. Similarly to **OASES** it is activated by specifying a negative number of wavenumber samples. The minimum and maximum horizontal phase speeds (**Cmin** and **Cmax**) that should be used in the sampling are still important. The method used is based on the supplied **Cmin** and **Cmax**; it computes the minimum and maximum wavenumbers for the maximum frequency. For all frequencies the integrand is sampled in the range of these two wavenumbers. The number of required samples at each frequency can then be determined from

geometric considerations.

11.1.1 Change in input format

- Frequency line: The OASES standard is to specify:
`Fmin Fmax Nfreq.`
 Hereby `Nfreq` frequencies are used from `Fmin` to `Fmax`. If `Nfreq` is negative then $|\text{Nfreq}|$ frequencies are read individually in the next line:
`Fdum Fdum -Nfreq`
`freq(1) freq(2) ... freq(Nfreq)`
- Receiver line: The OASES standard is to specify:
`RDmin RDmax Nrd.`
 Hereby `Nrd` receiver depths are used from `RDmin` to `RDmax`. If `Nrd` is negative then $|\text{Nrd}|$ receivers are read individually in the next line:
`Rdum Rdum -Nrd`
`rd(1) rd(2) ... rd(Nrd)`
- For options D, f and c the *range format* lines are:
`no_of_ranges`
`range(1), ... range(no_of_ranges)`
 hence each used range must be specified. For option d the format is:
`R_min R_max`
 thus only ranges between `R_min` and `R_max` are used in the input file.
- If the shear speed is -999.999 then the layer is an acoustic gradient layer. The top sound speed corresponds to the P sound speed in the layer and the bottom sound speed to the P sound speed in the layer below.

11.1.2 Additional OAST options

- i** Incoherent averaging of transmission loss. At range r it is found by averaging the transmission loss in range from $r - 0.116r$ to $r + 0.116r$ as described in Ref. [42].
- t** Tilt of receiver array. It is measured as the horizontal deviation at the last receiver measured in meters. When this option is specified, it is the fourth parameter in the receiver-depth line of the input file. It works only for a single range.

A horizontal array can be simulated using this option. First specify the minimum and maximum receiver depth of the array. The tilt is relative to the full length of the array. The source-receiver distance is the distance between the source and the first receiver. By specifying a horizontal array in this way, the same options as for a vertical array can be used.

11.1.3 Pointers in OAST

The pointers **parm** and **index** are used to map between the optimization variable and the environmental parameter to be optimized. The first pointer **parm** points to the type of parameter and the second **index** identifies the actual parameter within that parameter type. For some parameters **index** is not important (source depth). The pointer **parm** can take the following values:

- 1 Depth of layer **index**. It should be between the depth of layer **index-1** and **index+1**.
- 2 P-speed of layer **index**.
- 3 S-speed of layer **index**.
- 4 P-attenuation of layer **index**.
- 5 S-attenuation of layer **index**.
- 6 Density of layer **index**.
- 7 Thickness of layer **index**. The thickness of one layer is changed by opening up the corresponding layer and keeping all other thicknesses constant; i.e. the depth below the corresponding layer will change.
- 8 Source depth.
- 9 Source-receiver range.
- 11 Shape function coefficient; **index** points to the shape function number.

11.1.4 OAST example. File: layer11

This example has been used in a published paper [10]. The options (line two in the input below) specify that we are using wavenumber domain option w, reading data using format option d, writing out the data from the starting model to the *.obs file option W, we are only matching the magnitude of the observations option G, using objective function option N, plotting the measured data and the best inverted results option p, and the baseline environment are superimposed on the *a posteriori* distribution plot option A.

```
11 layers- example used in GA-paper.
w W d G N p A                ! Options
1000 32 1                    ! niter, q, npop
0.8 0.5 0.05                 !px, pu, pm
N J I T                      ! OASES options
100 100 1 0                  ! Fmin Fmax Nfreq
13                           ! Number of layers
0      0    0 0 0 0 0       ! First layer is vacuum
0  1500  0 0 0 0 1
50 1600  0 0.1 0.2 1.6
60 1600  0 0.1 0.2 1.6
```

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

70 1600 0 0.1 0.2 1.6
80 1600 0 0.1 0.2 1.6
90 1600 0 0.1 0.2 1.6
100 1800 0 0.1 0.2 2.0
110 1800 0 0.1 0.2 2.0
120 1800 0 0.1 0.2 2.0
130 1800 0 0.1 0.2 2.0
140 1800 0 0.1 0.2 2.0
150 2200 0 0.1 0.2 2.2
50                                     ! source depth
50 50 1                             ! first, last, number of receivers
1400 3000                           ! Cmin, cmax
64 1 64                             ! wavenumbers

1                                     ! The Range information is dummy
1300                                ! For wave number inversion

11                                  ! nparm q
2 3 1500 2300 128                  ! P-speed, lay 3
2 4 1500 2300 128                  ! P-speed, lay 4
2 5 1500 2300 128                  ! P-speed, lay 5
2 6 1500 2300 128                  ! P-speed, lay 6
2 7 1500 2300 128                  ! P-speed, lay 7
2 8 1500 2300 128                  ! P-speed, lay 8
2 9 1500 2300 128                  ! P-speed, lay 9
2 10 1500 2300 128                 ! P-speed, lay 10
2 11 1500 2300 128                 ! P-speed, lay 11
2 12 1500 2300 128                 ! P-speed, lay 12
2 13 1500 2300 128                 ! P-speed, lay 13

```

First the observed data should be generated. This is done by first deleting the *.in file. Then run SAGA using the command:

```
saga layer11 oast
```

It will write the observed data to the *.obs file. This file is then copied to the *.in file. Now, run SAGA again. The output of SAGA is the following:

```

best of all energy 3.2385509E-02
best-of all      deviation
1 1588.1          -11.8
2 1903.1          303.1
3 1638.5          38.5
4 1915.7          315.7
5 1991.3          391.3
6 2161.4          361.4
7 1966.1          166.1

```

SACLANTCEN SM-333

NATO UNCLASSIFIED

8	2287.4	487.4
9	2066.9	266.9
10	2060.6	260.6
11	2041.7	-158.2

It is seen that in the first layer, which is the most important one, the sound speed is estimated quite well. Here only one population was used and this is not a sufficiently large sample for statistics of the estimates. We therefore increase the number of populations to 50 and run SAGA again. When SAGA has finished, the post-processor is executed:

post layer11 oast

The result of the run is then

best fit (best, ppd, mean) 1.6866000E-03 2.7032106E-03 3.2024365E-02

			best-of all	most likely	mean	std-dev
1 P sound speed (m/s)	3		1594.4	1594.4	1600.0	0.011
2 P sound speed (m/s)	4		1607.0	1607.0	1607.0	0.019
3 P sound speed (m/s)	5		1588.1	1581.8	1616.0	0.044
4 P sound speed (m/s)	6		1619.6	1619.6	1652.1	0.057
5 P sound speed (m/s)	7		1607.0	1600.7	1766.6	0.188
6 P sound speed (m/s)	8		1814.9	1814.9	1782.2	0.144
7 P sound speed (m/s)	9		1751.9	1751.9	1788.6	0.157
8 P sound speed (m/s)	10		1733.0	1777.1	1764.1	0.154
9 P sound speed (m/s)	11		1537.7	1613.3	1770.1	0.281
10 P sound speed (m/s)	12		2048.0	1947.2	1937.9	0.195
11 P sound speed (m/s)	13		2073.2	2098.4	1971.1	0.325

The first line [best fit...] indicates the best value of the objective function obtained using the best, most likely and mean model vector. The value of the model vector then follows for each of the estimates. The output is explained in Sect. 3.

From these results it is observed that for the first four layers we have quite good results. The normalized standard deviation is small. For the lower layers, the sound speeds are not so well determined, the standard deviation is also quite high. It is natural that the sound speeds in the lower layers are less well determined, as they have less influence on wave propagation in water.

Next, the results are plotted using MATLAB:

```
>> plotsaga
SAGA filename ? layer11
```

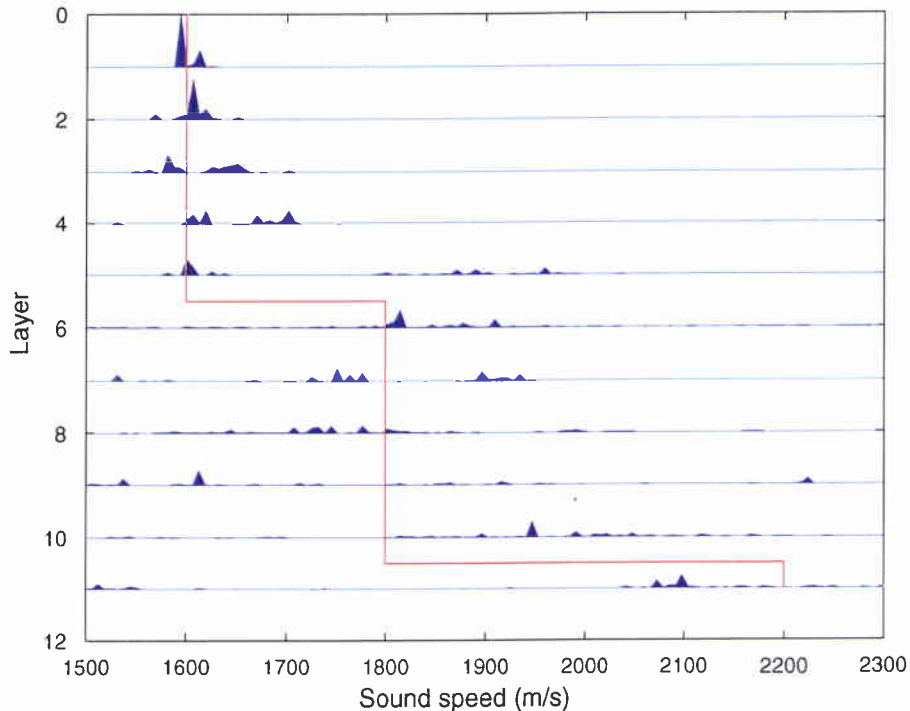


Figure 6 *The a posteriori distributions for layer11. The baseline environment that generated the data is shown in red.*

The *a posteriori* distributions for this case are shown in Fig. 6. When the distributions cluster around a certain value, it is an indication that a parameter is well determined. The above plot was produced by weighting each model vector according to its fit, as better fits are more likely to represent the true solution, see Sect. 2.2. Alternatively, we could use a uniform weighting of the sampled model vectors, Fig. 7. This can be done by entering option u in the **SAGA** option line. The distributions now become flatter and less centered around the true values.

The best obtained fit is shown in Fig. 8.

Using the information in the **layer11b.mat** file it is also possible to construct a plot of the most likely model vector. An example of that is given in Fig. 7 of Ref. [10]. It is also possible to plot the correlation between two parameters as presented in Fig. 8 of Ref. [10].

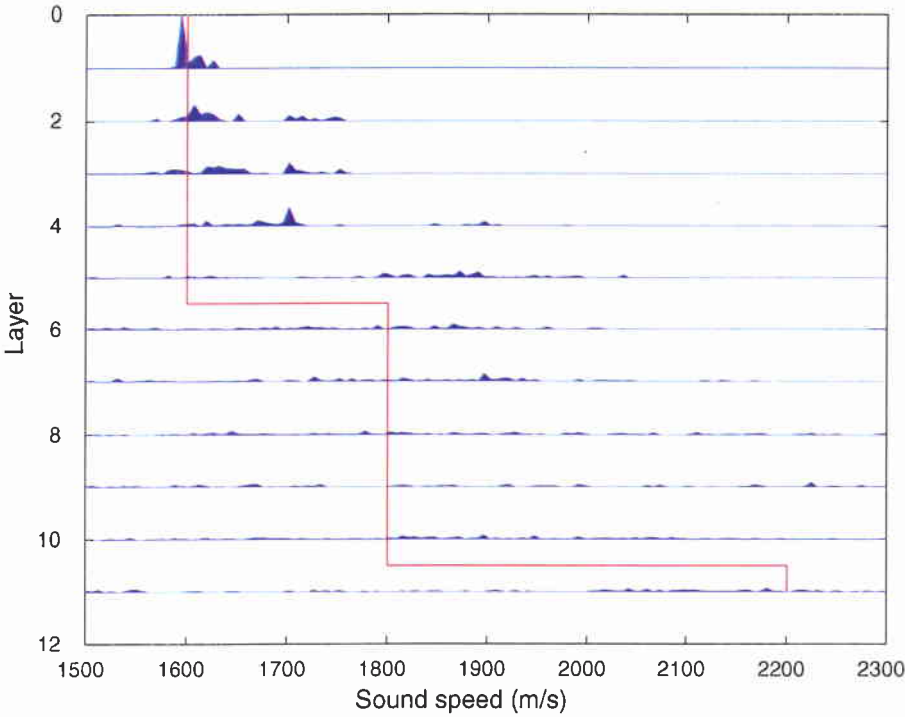


Figure 7 *The uniform a posteriori distributions for layer 11. This represent a histogram of the model vectors obtained from the inversion. The baseline environment that generated the data is shown in red.*

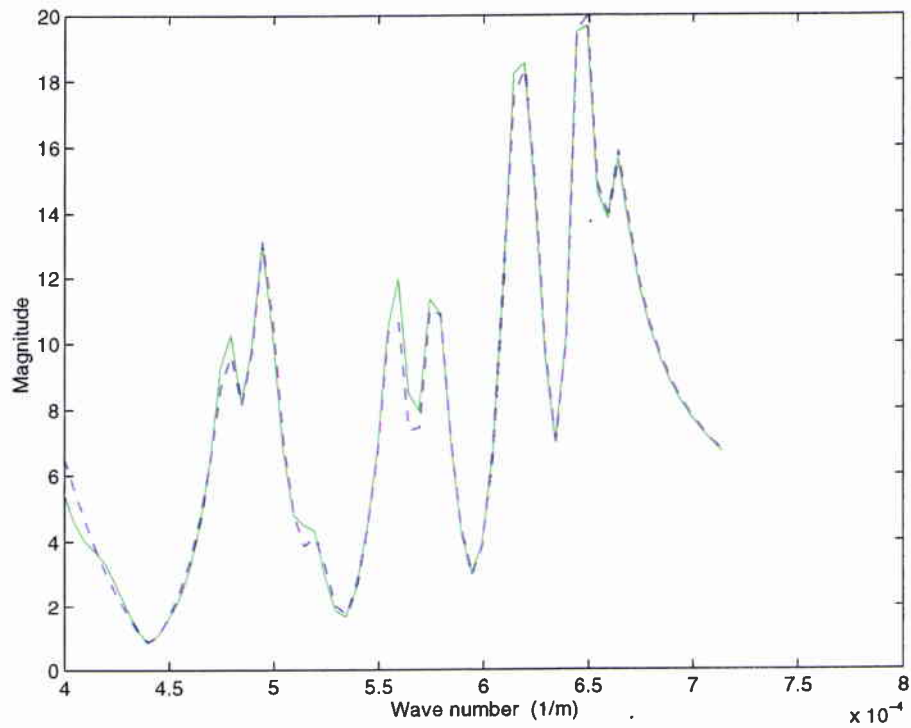


Figure 8 *The fit in wavenumber space for layer11. The green line is the observed data, and the blue line is the computed data with the best set of parameters.*

11.1.5 OAST example. File: tellaro_oast

This example shows an inversion for observed transmission loss data. The measured data was taken at the Tellaro site, in the Gulf of La Spezia. The input file for this run was used in Refs. [11, 43], for further details see these papers. This inversion using OASES is similar to the example with SNAP in Sect. 11.4.5. For this type of long range propagation SNAP is preferable, both because the CPU-time is less and because this code is easier to use. The additional information about shear and attenuation that can be obtained using OASES is not important in the present example.

11.2 Forward model: OASR

OASR is the reflection coefficient module of SAFARI. The forward model for OASR follows the format of the SAFARI manual [2, 3].

Some numerical codes model the bottom using the reflection loss as a function of angle of incidence, whereas others use the physical sound speed profile in the bottom. The SAGA-OASR module can be used to estimate a physical bottom from a given reflection loss curve.

11.2.1 Pointers in OASR

The pointers are the same as for OAST, see Sect. 11.1.3, except that the pointers to source and receiver geometry are not used.

11.2.2 OASR example. File: rfi

This synthetic example demonstrates the use of the OASR module. From the option line it is seen that the data is read using format option d, the calculated data from the starting model is written to the *.obs file option W, using objective function option N, and the observed and calculated data are then plotted with option p.

```

Refection coefficient.
d W N p                                ! options
1000 32 2                              ! niter, q, npop
0.8 0.5 0.05                          ! px, pu, pm

3

```

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

0 1500 0 0 0 1 0
0 2000 0 0.1 0.2 1.6 0
30 2500 0 0.1 0.2 1.6 0
50 100 2 ! frequencies: Fmin Fmax Nfreq
0 90 100 ! angle

2 ! nparm
2 2 1500 3093.75 999
2 3 1500 3093.75 999

```

The result of the inversion is quite good, as shown by the output from SAGA:

```

best of all energy 7.1087794E-07
best-of all deviation
1 1999.843 -0.157
2 2499.687 -0.313

```

Both sound speeds in the bottom are well determined. In this case there are so few variables that it does not seem necessary to display the *a posteriori* probability distributions.

11.3 Forward model: OASTG

This version should only be used if gradients of the parameter vector are used in the optimization [11]. At present it works with only one frequency. It is more complicated to use than the standard OAST module.

This forward model is based on OAST. The input file and the pointers are therefore identical to the ones described in Sect. 11.1. If gradients are not used it has the same functionality as OAST, but it uses more dynamic memory.

The program works with “exact” derivatives from OASES with acoustic layers, thus all optimization parameters should be in acoustic layers. The other layers can be of any type (e.g. elastic).

This program has great potential for problems that require computation of gradients, as for example Cramer-Rao bounds. When gradients are computed based on a finite difference approach there are generally large errors involved in the computation, as opposed to the present exact approach.

11.4 Forward model: SNAP

If the wave propagation is long-range it is practical to model the wavefield using normal modes. This is much faster than the more precise method of wavenumber integration.

The main part of the input follows the description in the **SNAP** manual [1]. The wave attenuation in water should be specified (in the water-depth line of the input file). If this attenuation is zero, the default attenuation is used, as in the original **SNAP** model.

11.4.1 Change in input format

- Receiver line: The **SNAP** standard is to specify:
`RDmin RDmax Nrd.`
 Hereby **Nrd** receiver depths are used from **RDmin** to **RDmax**. If **Nrd** is negative then $|\mathbf{Nrd}|$ receivers are read individually in the next line:
`Rdum Rdum -Nrd`
`rd(1) rd(2) ... rd(Nrd)`
- For options **D**, **f** and **c** the *range format* lines are:
`no_of_ranges`
`range(1), ... range(no_of_ranges)`
 hence each used range must be specified. For option d the format is:
`R_min R_max`
 thus only ranges between **R_min** and **R_max** are used in the input file.

11.4.2 Options in SNAP

In the input file an option line is introduced for **SNAP**. Currently, there are only two options:

- i** Incoherent addition of modes (only **SNAP**).
- t** Tilt of receiver array (both **SNAP** and **SNAPRD**). It is measured as the horizontal deviation at the last receiver measured in meters. When this option is specified, it is the fourth parameter in the receiver-depth line of the input file. It works only for a single range.

A horizontal array can be simulated using this option. First specify the minimum and maximum receiver depth of the array. The tilt is relative to the full length of the array. The source-receiver distance is the distance between the

source and the first receiver. By specifying a horizontal array in this way, the same options as for a vertical array can be used.

11.4.3 *Pointers in SNAP*

The pointer **parm** is used to map between the optimization variable and the environmental parameter to be optimized. The second parameter, **index**, points to the layer; for some parameters **index** is not important (source depth). The pointer **parm** can take the following values:

- 1 Water depth. The depth is changed by moving the last point in the water.
- 2 Sound speed in water; **index** refers to the actual point.
- 3 Sound speed in sediment; **index** refers to the actual point.
- 4 Attenuation. Depending on the value of **index** it refers to the following attenuation: **index** = 0(water), 1(sediment), 2(bottom), 3(shear bottom).
- 5 Surface roughness, **index** = 1(ocean surface), 2(bottom surface).
- 6 Sediment density.
- 8 Source depth.
- 9 Source-receiver range.
- 11 Shape-function coefficient; **index** points to the number.
- 12 Bottom P-speed.
- 13 Bottom S-speed. The shear in **SNAP** should have a low value (< 500 m/s), otherwise the perturbation correction is not valid.
- 14 Bottom density.
- 15 First receiver depth. The spacing is constant, thus it is a vertical translation.
- 16 Depth of speed point in water; **index** refers to the actual point.
- 17 Sediment thickness. The thickness is changed by moving the last point in the sediment.
- 18 Depth of speed point in sediment; **index** refers to the actual point.
- 19 Tilt.

11.4.4 *SNAP example. File: sspmisa*

This example is from the 1993 Matched Field Workshop [37], and has been used in a published paper [41]. The environment is shown in Fig. 9. The observed data are based on a synthetic data set from a normal mode code using a 250 Hz source in shallow water. The data are received on the 20 hydrophones spanning the whole water column. White Gaussian noise was added to the data vectors to obtain a SNR = 40 dB [37]. Only four parameters are unknown in this case; the source range and

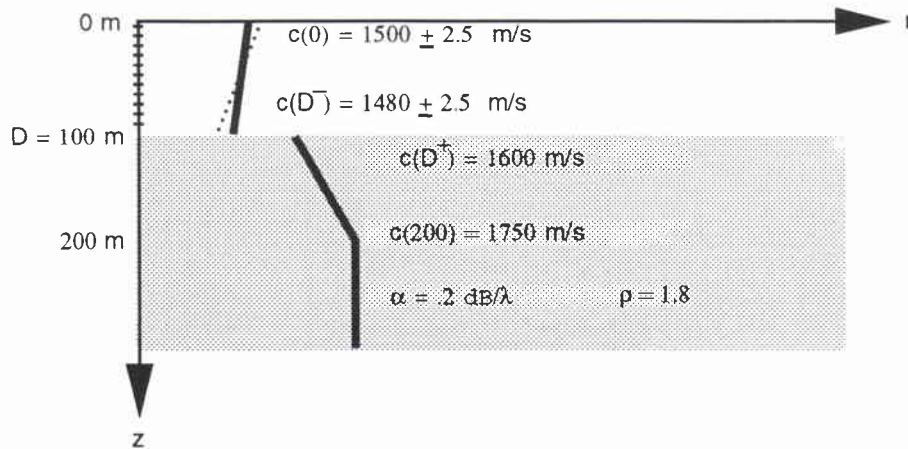


Figure 9 The environment for the *sspmisa* case. The source coordinates are (9.3 km, 78 m) for $SNR = 40$ dB.

depth and the ocean sound speed at the top and the bottom. Each of the parameters can take 51 discrete values.

The options in the input data file indicate that we are using a covariance matrix, option c, and that this matrix is scaled to a maximum Bartlett power of one, option b. We plot the variation of power across the array, option p. Option t in the SNAP-option line indicates that the array can be tilted; the fourth parameter in the receiver line defines the initial tilt. We use GA with 10 populations, each with 1000 individuals; 10,000 forward models are used in total.

```

sspmis_a    from the matched field workshop 93
c b p                                ! options
1000 32 10                                ! niter, q, npop
0.8 0.5 0.05                              ! px, pu, pm
t                                           ! snap options
1 100                                    ! No of freq, max modes
250.                                       ! freq
100 0 0 0                                ! wd scat(1) scat(2) beta
    0 1499.4
    100 1481.6
100 1.8 0.2                                ! Sediment depth density and attenuation
    0 1600
    100 1750
1.8 0.2 1750                              ! bottom density, attenuation and sound speed
0 0                                         ! bottom shear attenuation and sound speed

78                                         ! source depth
5 100 20 0                                ! first & last receiver, # receivers, tilt

```

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

1                                ! number of ranges
9300                             ! receiver range

4                                ! nparm
2 1      1497.5 1502.5 51        ! upper sound speed point
2 2      1477.5 1482.5 51        ! lower sound speed point
9 1      5000   10000 51         ! source range
8 1      0.01    100   51        ! source depth

```

After running SAGA and POST we obtain the results:

best fit (best, ppd, mean)	3.4361999E-04	2.1212101E-03	5.5390596E-04	
	best-of all	most likely	mean	std-dev
1 Top Water sound speed (m/s)	1 1499.7	1498.6	1499.12	0.117
2 Bottom Water sound speed (m/s)	2 1481.9	1480.8	1481.33	0.116
3 Source range (m)	1 9300.0	9300.0	9300.0	0.000
4 Source depth (m)	1 78.0	78.0	78.0	0.000

The first line [best fit...] indicates the value of the objective function that is obtained using the best, most likely and mean model vector. The value of the model vector then follows for each of the estimates. The output is explained in Sect. 3.

It is seen that the source coordinates are perfectly determined. The sound speeds are also quite well determined, but the normalized standard deviation is much larger. The same can be concluded from plotting the 1-D marginal *a posteriori* distributions (Fig. 10). By examining the 2-D marginal *a posteriori* distributions (Fig. 11) this can be explained.. The reason for the poor resolution of the parameters is due to a strong correlation between the ocean sound speed at the top and at the bottom. The 2-D marginal *a posteriori* distribution was produced by introducing option m12 on the option-line in the *.dat file and then running POST. The contour plot has been plotted using CPLOT, see Sect. 5.1.

The correlation of the two sound speeds can also be seen by fixing the source coordinates and then plotting a contour of the variation in the objective function for the two sound speeds, see Fig. 12. The ambiguity plot has been produced by putting option C in the option-line in the *.dat file; the number of discrete points for the two parameters has been changed to 40. It is then necessary to re-run SAGA, as this is seen as a 2-D exhaustive search. The contour plot has been plotted using CPLOT, see Sect. 5.1.

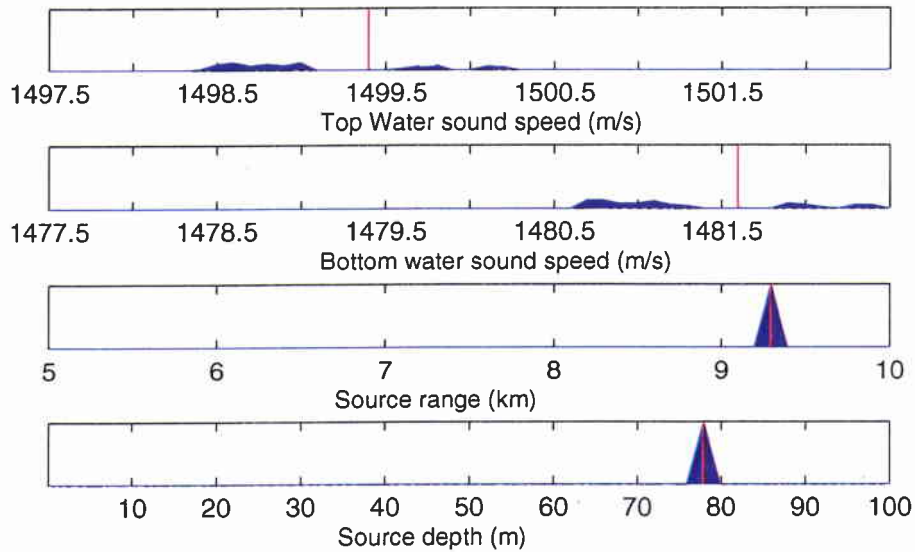


Figure 10 The 1-D a posteriori distributions for the sspmisa case. The red line indicates the true value.

11.4.5 SNAP example. File: tellaro_snap

This example shows an inversion for transmission loss data as described in Refs. [11, 43]. The input has been simplified relative to the examples in [11, 43]. The options in the input file show that we are reading data using the range format, option D. We are only using the magnitude of the pressure and the data is weighted with $1/\sqrt{r}$ to put more importance on the shorter ranges, option R. The objective function is based on least squares, option n. We are plotting observed and synthetic data which best matches the observed data, option p.

```
tellaro-data
D n R p          ! options
3000 64 15       ! niter, q, npop
0.8 0.5 0.05     ! px, pu, pm
                 ! option
1                ! No of freq
330.             ! the freq
15.0 0. 0. 1e-8  ! water-depth, roughness surf & bottom, small att.
  0  1523
 15  1523
24.0  1.7 0.25   ! sediment-depth, density, cp-attenuation
  0.  2000.       ! sediment sound speed
  3.  2000.       ! sediment sound speed
  9.  2000.       ! sediment sound speed
 24.  2000.       ! sediment sound speed
```

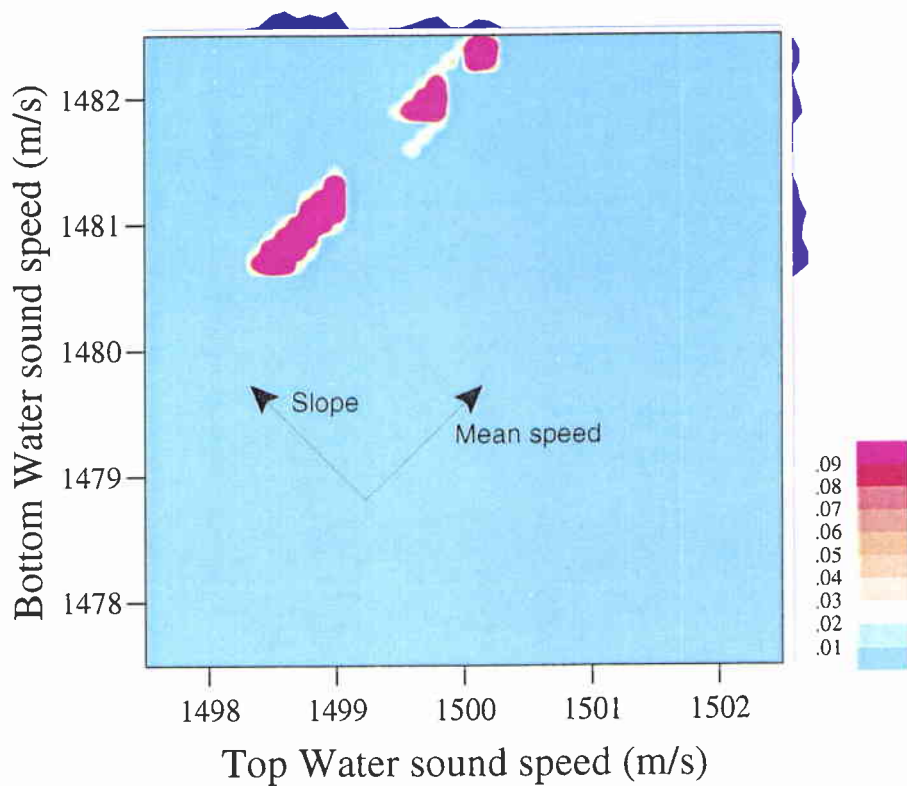


Figure 11 The 2-D marginal a posteriori distribution between the upper and lower ocean sound speed for the **sspmisa** case. It is based on the same data as Fig. 10. The marginal 1-D distribution for each parameter are displayed on the top and to the right. By re-parameterizing the sound speed as slope and mean, a better resolution is obtained.

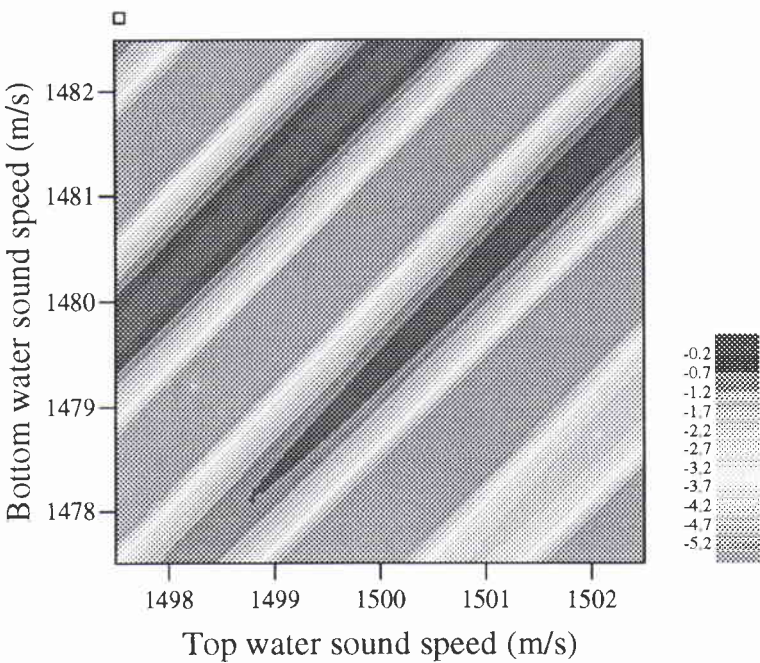


Figure 12 *Ambiguity surface for the sspmisa case.*

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

1.7 .25 2000
0. 0.                ! No shear

7                    ! source depth
5. 5. 1             ! NRD RD(1:NRD)
270 1400             ! for option D: Rmin, Rmax

6                    ! nparm
3 1 1480 1580 128    ! sediment speed 1
3 2 1480 1580 128    ! sediment speed 2
3 3 1480 1580 128    ! sediment speed 3
3 1 1480 1580 128    ! sediment speed 4
8 1 6.0 8.0 128      ! source depth
4 1 0. 0.5 128       ! attenuation in sediment

```

11.4.6 SNAP example. File: ys3

This example was used in the analysis of the Yellow Shark experimental data [44]. The source signal was transmitting 7 frequencies simultaneously in the band from 200–800 Hz. First the covariance matrix was estimated by averaging over several time snapshots using a multiple window technique [45, 46], as implemented in the MATLAB files. In the example the data at all frequencies are used simultaneously, but single frequencies could also be used. For results and discussion, see Ref. [44].

11.5 Forward model: SNAPRD

The range dependent version of SNAP allows for a more realistic description of the ocean environment. However, the inversion problem becomes more complicated. Both the CPU-time and the number of parameters required to describe the problem increase linearly with the number of range sectors.

The main part of the input follows the description in the SNAP manual [1]. The wave attenuation in water should be specified (in the water-depth line of the input file). If this attenuation is zero, the default attenuation is used, as in the original SNAP model.

11.5.1 Options in SNAPRD

For SNAPRD the range points should be sampled equidistantly. In the input file an option line is introduced for SNAPRD. Currently, there is only one option:

t Tilt measured as the horizontal deviation at the last receiver measured in meters. When this option is specified, it is the fourth parameter in the receiver-depth line of the input file. It works only for one range.

A horizontal array can be simulated using this option. First specify the minimum and maximum receiver depth of the array. The tilt is relative to the full length of the array. The source-receiver distance is the distance between the source and the first receiver. By specifying a horizontal array in this way, the same options as for a vertical array can be used.

11.5.2 Pointers in SNAPRD

Due to the increased dimensionality of the problem, we need 3 pointers to specify which parameter we are inverting. The two first pointers **parm** and **index** are the same as in **SNAP**, while the third pointer **index2** indicates the range sector.

11.5.3 SNAPRD example. File: **elbard**

This example is based on the data described in [38, 47]. Here we invert using a range dependent environment. The sound speed profile in the water is determined from an XBT at both the source and the receiver range. We use shape functions option E to produce a single range independent bottom and to describe the bottom sound speed more efficiently. The sound speed profile is modeled using the sound speed at the bottom interface and subsequent sound speed points are modeled using the increase in sound speed from the previous point.

We are using observations at 3 frequencies and adding them incoherently. The data is in covariance format, option c, and the covariance matrix was estimated by averaging over several time snapshots using a multiple window technique [45, 46] as implemented in the MATLAB files. We are plotting the variation of power across the array for each frequency, option p. The input file is:

```
North elba
E c p          ! options
2000 32 10     ! niter, q, npop
0.8 0.5 0.05   ! px, pu, pm
               ! snap options
3 100          ! No of freq, max_no_modes
164.795 169.92 175.048
2              ! Number of sectors
6              ! Sector length in km
129.9,0.,0.,0  ! water depth scatt(1),scatt(2),beta(0)
0.            1525.5
```

SACLANTCEN SM-333

NATO UNCLASSIFIED

```

56.8    1526.4
65.1    1523.1
68.9    1520.6
71.3    1517.2
75.6    1512.1
76.8    1510.9
79.9    1510.1
100.4    1508.4
129.9    1508.3
10 1.75 0.13      ! sediment thickness, r1, beta(1)
    0    1520.
    5    1580.
    10   1600
1.8, 0.15, 1600.  ! bottom r2 beta(2),c2
0.,0.             ! bottom shear beta(3) c2s

6               ! sector length SECTOR 2
127.1,0.,0.,0    ! water depth scatt(1),scatt(2),beta(0)
    0.0    1525.5
    56.4    1526.3
    60.1    1525.8
    66.6    1522.1
    70.6    1517.7
    77.9    1510.3
    86.1    1509.1
    101.1    1508.3
    127.1    1508.3
10 1.75 0.13      ! sediment thickness, r1, beta(1)
    0    1520.
    5    1580.
    10   1600
1.8, 0.15, 1600.  ! bottom r2 beta(2),c2
0.,0.             ! bottom shear beta(3) c2s

80             ! source depth
112.7 18.7 48 1  ! rec depth
1             ! number of ranges
5600          ! receiver range

7             ! nparm
9 1 1 5300 5900 128 ! source range
8 1 1 70 85 128 ! source depth
1 1 1 127 132 128 ! water depth
1 1 2 127 132 128 ! water depth
11 1 1 1510 1560 128 ! sound speed sed
11 2 1 0 100 128 ! sound speed sed
11 3 1 0 100 128 ! sound speed bot

```

The shape function file is:

```
! shape function file for the North Elba range dependent case
3 8 1          ! No. of shape functions, No. of points, No of blocks
3 1 1          ! sound speed sed
3 1 2          ! sound speed sed
3 2 1          ! sound speed sed
3 2 2          ! sound speed sed
3 3 1          ! sound speed sed
3 3 2          ! sound speed sed
12 1 1         ! sound speed basement
12 1 2         ! sound speed basement
! Block 1: coupling of sed 1
3 8
1 0 0
1 0 0
1 1 0
1 1 0
1 1 1
1 1 1
1 1 1
1 1 1
1 1 1
1520 20 30    ! starting values
```

For results and discussion, see Refs. [38, 47].

11.5.4 *SNAPRD* example. File: **shot05**

This example is based on the shot data described in [48, 49]. Here we invert using a range dependent environment. The sound speed profile in the water is determined from an XBT at both the source and the receiver range.

The stability of the parameter estimate depends on the time-frequency variability of the energy distribution in the received data. To demonstrate this variability, the array-averaged spectrogram is calculated for a 256 ms time window which slides over the time domain with a 10 ms increment (Fig. 13). It can be seen that the energy content remains at the same level for only a short period (from 250 to 450 ms) corresponding to the high energy segment of the signal. Parameter estimates based on this time segment are expected to be very stable, and it is used here to estimate the covariance matrix for the frequencies used.

The data is processed using two objective functions, the default multifrequency objective function, option c (unnormalized), and one where the covariance matrix for each frequency is normalized by dividing with the sum of the diagonal, option b

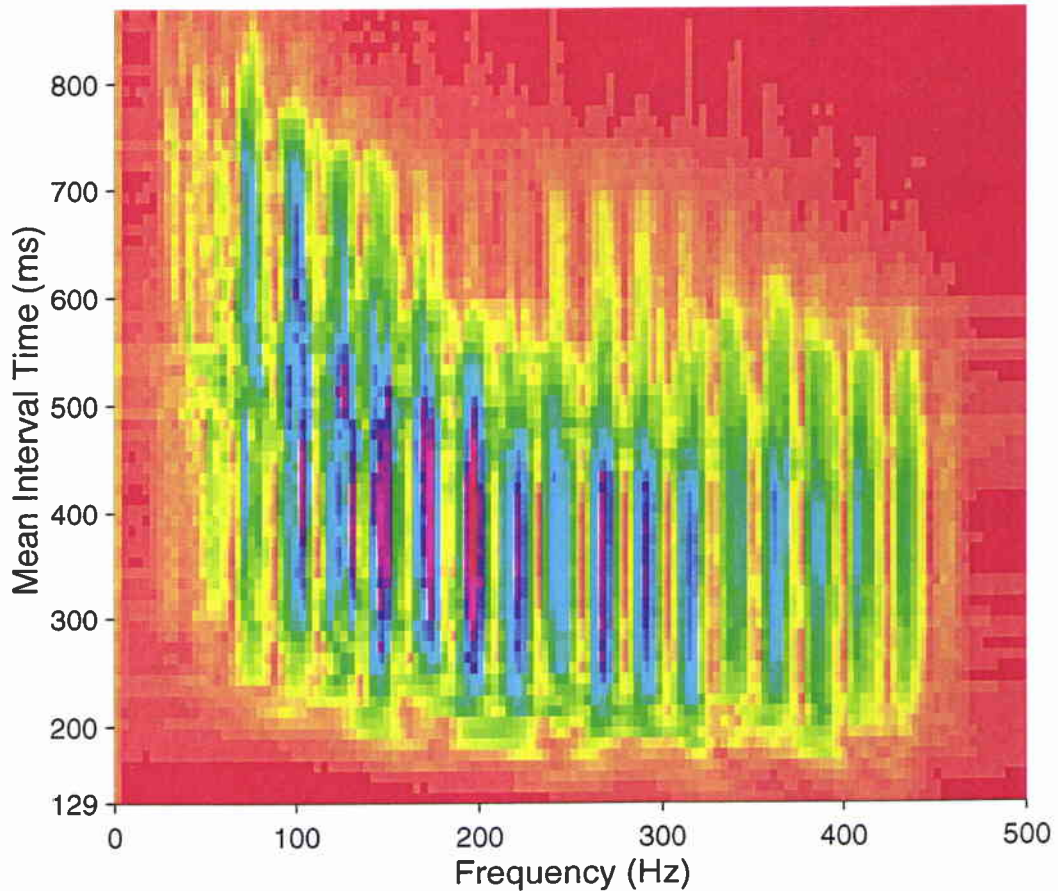


Figure 13 *Array-averaged spectrogram for the shot05 case.*

(normalized). A result of this inversion for the source coordinates using either of the objective functions is seen in Fig. 14 as a function of the number of frequencies in the inversion. The normalized-version results have unpredictable behaviour, especially in the case of source depth, where the estimate does not converge at all. The unnormalized objective function seems preferable, and is standard for SAGA. This plot can easily be reconstructed using the `results.m` file. For further details, see [48, 49].

11.5.5 SNAPRD example. File: `tl_malta_rd`

This example relates to data taken during the Winter Sun experiment. We are inverting coherent transmission loss at four depths and four frequencies, over ranges from 200 m to 17 km; the data are given using the data format option D. We are matching both the shape and the offset of the 16 curves, option X. As the

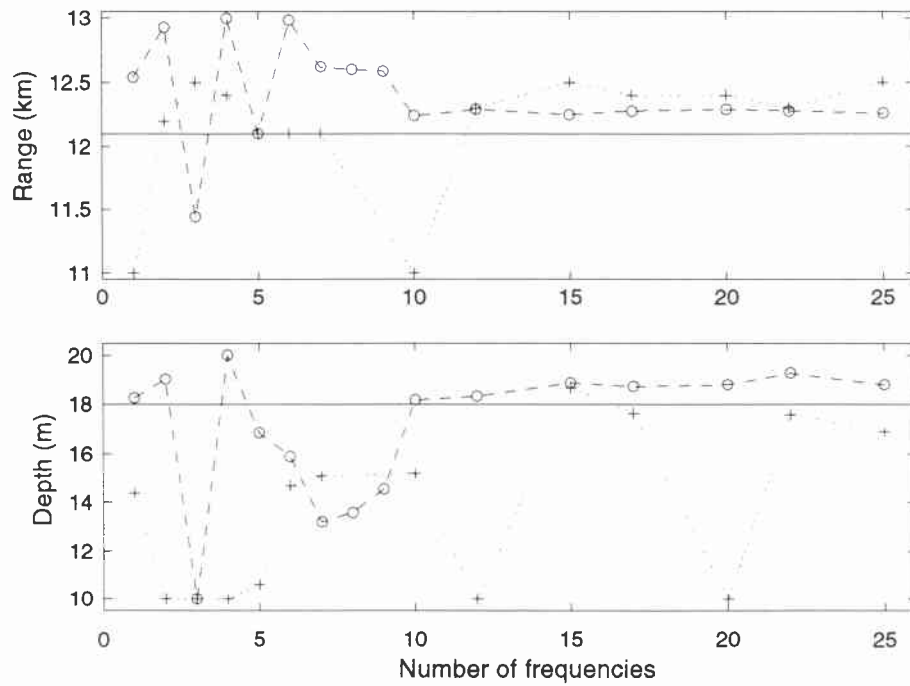


Figure 14 *Parameter estimates for source range (a) and source depth (b) for the shot05 case. Solid lines indicate the baseline model values. Lines designated with crosses represent the estimates of the normalized objective function. Lines designated with circles indicate the estimates of the unnormalized objective function.*

final result of the inversion is presented as transmission loss curves plotted in dB, we are inverting the transmission loss measured in dB, option 1, rather than using pressure magnitude. The bottom was slightly range dependent and therefore it was modeled with adiabatic modes with a new sector each 2 or 3 km. There is not enough information in the data to give an independent estimate of the bottom for each sector, therefore all bottoms were coupled together to find one representative bottom; this coupling of the parameters is done by using shape functions, option E.

Results of the inversion are shown in Fig. 15.

11.6 Forward model: PROSIM

PROSIM [5] is a broadband adiabatic normal mode program that is based on the ORCA program [6, 7]. In order to increase robustness and efficiency the search for eigenvalues is limited to the real wavenumber axis. Attenuation is accounted for using the same approximations as in SNAP. Shear and surface roughness is not yet included in the model. PROSIM is very fast for broadband normal mode calculations as it interpolates eigenvalues and mode functions as a function of frequency. However, if an error is detected in tracing the eigenvalues, a classical normal mode single frequency solution is used.

The main part of the input follows the description in the SNAP manual [1]. The frequencies that are used in PROSIM are specified by either:

```
Nfreq 0                ! the '0' is not read
freq(1) freq(2)... freq(Nfreq)
or
-1 Δt                  ! the Δt is only read for option F
Dfreq Fmin Fmax
```

For option F an unknown starting time has been introduced. This is due to the fact that we often do not know the absolute time, or that the frequency spacing is too large to obtain the full signal arrival time.

SACLANTCEN SM-333

NATO UNCLASSIFIED

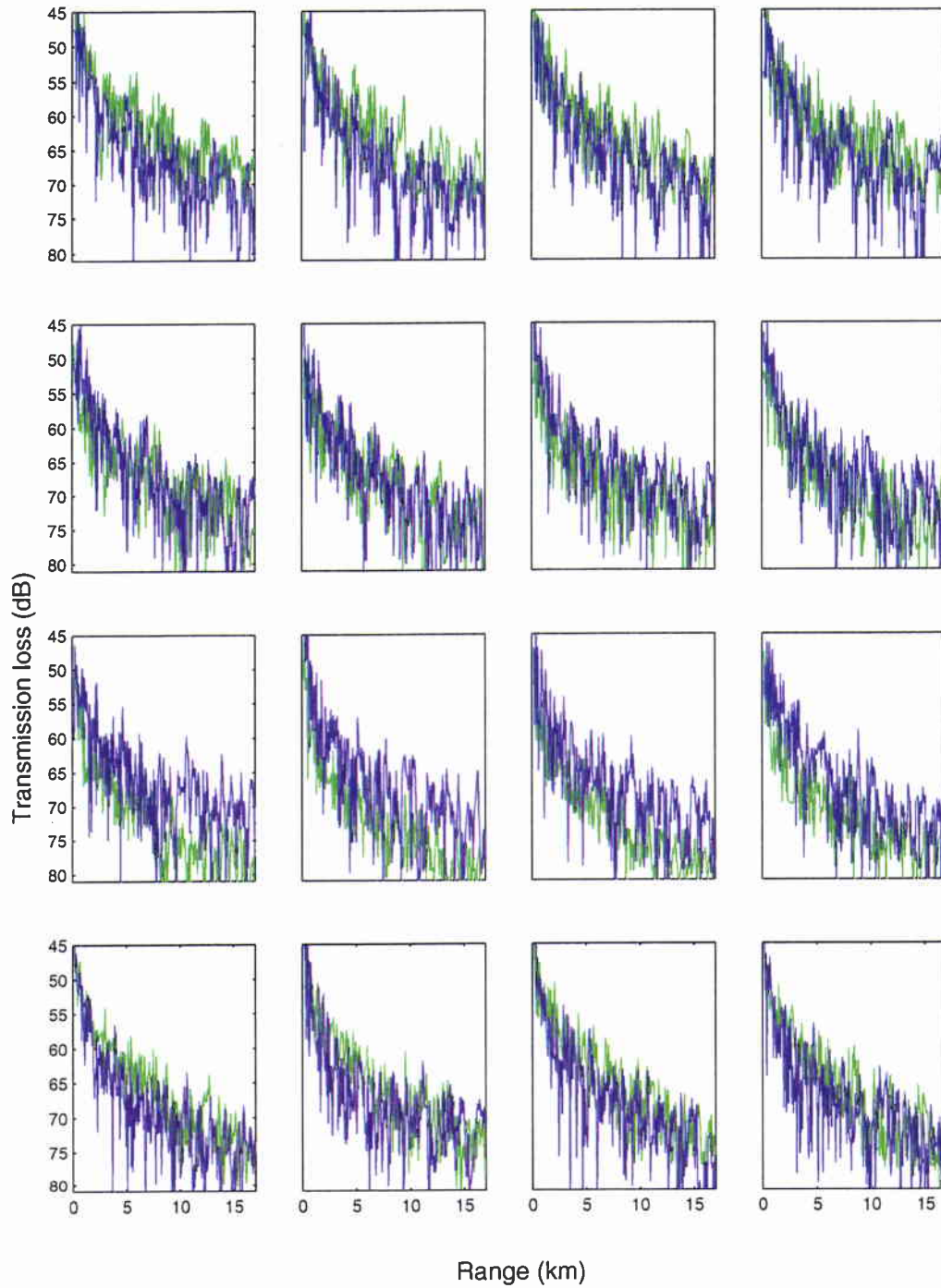


Figure 15 Transmission loss for the **tl_malta_rd** case. The observed data (green line) and the modeled data (blue line) with the best found environment is plotted. The plot in each column represents frequencies 299, 399, 599, 998 Hz and each row from the top, hydrophone depths 121, 101, 81 and 61 m.

11.6.1 Options in PROSIM

In the input file an option line is introduced for PROSIM. Currently, there is only one option:

- p** Positive gradient. Both sound speed and density in sediment and subbottom must increase as a function of depth. If, during optimization of an environment, negative gradients are found, the forward model will not be executed.

11.6.2 Pointers in PROSIM

Due to the increased dimensionality of the problem we need 3 pointers to specify which parameter we are inverting for. The two first pointers **parm** and **index** are nearly the same as in SNAP, while the third pointer **index2** points to the range sector.

The pointer **parm** is used to map between the optimization variable and the environmental parameter to be optimized. The second parameter, **index**, points to the layer; for some parameters **index** is not used (e.g. source depth). The pointer **parm** can take the following values:

- 1 Water depth. The depth is changed by moving the last point in the water.
- 2 Sound speed in water; **index** refers to the actual point.
- 3 Sound speed in sediment; **index** refers to the actual point.
- 4 Attenuation. Depending on the value of **index** it refers to the following attenuation: **index** = 1(sediment), 2(bottom).
- 6 Sediment density.
- 8 Source depth.
- 9 Source-receiver range.
- 11 Shape-function coefficient; **index** points to the number.
- 12 Bottom P-speed.
- 14 Bottom density.
- 15 First receiver depth. The spacing is constant, thus it is a vertical translation.
- 16 Depth of speed point in water; **index** refers to the actual point.
- 17 Sediment thickness. The thickness is changed by moving the last point in the sediment.
- 18 Depth of speed point in sediment; **index** refers to the actual point.
- 20 Time delay ΔT .

11.6.3 *PROSIM* example. File: *waa*

This example is from the Matched Field Workshop, Vancouver, 1997 [50]. The options indicate that we are using a pressure vector, option e. The Bartlett power is added incoherently across frequencies, option f. We use data from 25 to 500 Hz with a spacing of 5 Hz, a total of 96 frequencies are used. For most other normal mode codes the CPU requirement for so many frequencies is so large that it is not practical to carry out an inversion. For the observed data and the replica with best estimated environment, the variation of power across the array is plotted, option p. We use GA with 20 populations, each with 1000 individuals; 20,000 forward models are used in total.

```

waa.dat, 96 frequencies all phones.
f e p          ! options
1000 32 20      ! niter, q, npop
0.8 0.5 0.05    ! px, pu, pm
                ! options
p              ! prosim option
-1 0            ! number of frequencies
5 25 500        ! frequencies, Df, Fmin, Fmax
1              ! number of sectors
1 00 00         ! dum, phase speed (00 = all), dum

100 0 0 10. 0.  !depth, dum, dum, range, dum
0. 1480.
100 1460.
50 1.500 -0.23  ! sediment
0. 1550
50 1700
2.00 -0.23 1800.0 ! bottom
0 0          ! not used

20           ! source depth
1 99.9999 100 ! rec depth
1           ! number of ranges
5000.       ! receiver range

9           ! nparm
9 1 1 5000 5400 128 ! source range
8 1 1 10 30 128 ! source depth
1 1 1 100 120 128 ! water depth
17 1 1 10 50 128 ! sediment thickness
3 1 1 1500 1600 128 ! sound speed sed
3 2 1 1550 1750 128 ! sound speed sed
12 1 1 1600 1800 128 ! sound speed sed
6 1 1 1.40 1.85 128 ! density in sediment
14 1 1 1.600 2.00 128 ! density in half space

```

The *a posteriori* distributions for this case is shown in Fig. 16. When the distributions cluster around a certain value it is an indication that a parameter is well determined. The best obtained fit is shown in Fig. 17.

11.6.4 PROSIM example, file: **mf**

This example is based on the **waa** case from the Matched Field Workshop, Vancouver, 1997 [50, 51]. The input file is nearly the same as in the previous example. Only one phone at a depth of 50 m and a range of 5 km is used, but data from frequencies from 25–200 Hz with a spacing of 1 Hz is used coherently. In the option line, option F is introduced in stead of option f.

The best obtained fit is plotted both in the frequency domain (Fig. 18) and in the time domain (Fig. 19). The transmission loss plot is produced directly from **SAGA**, whereas the time series first requires an FFT of the data. It should be noted that in the matched field approach used here, only the relative phase across all frequencies is determined. In order to plot the comparison of the time series the average phase difference between the observed and computed data was determined. The time series have been convolved with a Ricker wavelet with a center frequency of 80 Hz. Finally, due to the coarse frequency sampling the absolute arrival time cannot be determined.

11.7 Forward model: **POPP**

The normal mode program **POPP** is used together with the reverberation model as described in Ref. [4]. The backscattering is estimated using Lambert's law with a power coefficient. The data should be expressed in dB and the matching is done in the dB domain. For an example of using this module, see Ref. [52] and Fig. 20.

11.7.1 Pointers in **POPP**

The pointer **parm** is used to map between the optimization variable and the environmental parameter to be optimized. The second parameter, **index**, points to the layer; for some parameters **index** is not important (e.g. source depth). The pointer **parm** can take the following values:

- 1 Water depth. The depth is changed by moving the last point in the water.
- 2 Sound speed in water; **index** refers to the actual point.
- 3 Sound speed in sediment; **index** refers to the actual layer.
- 4 Attenuation in sediment; **index** refers to the actual layer.

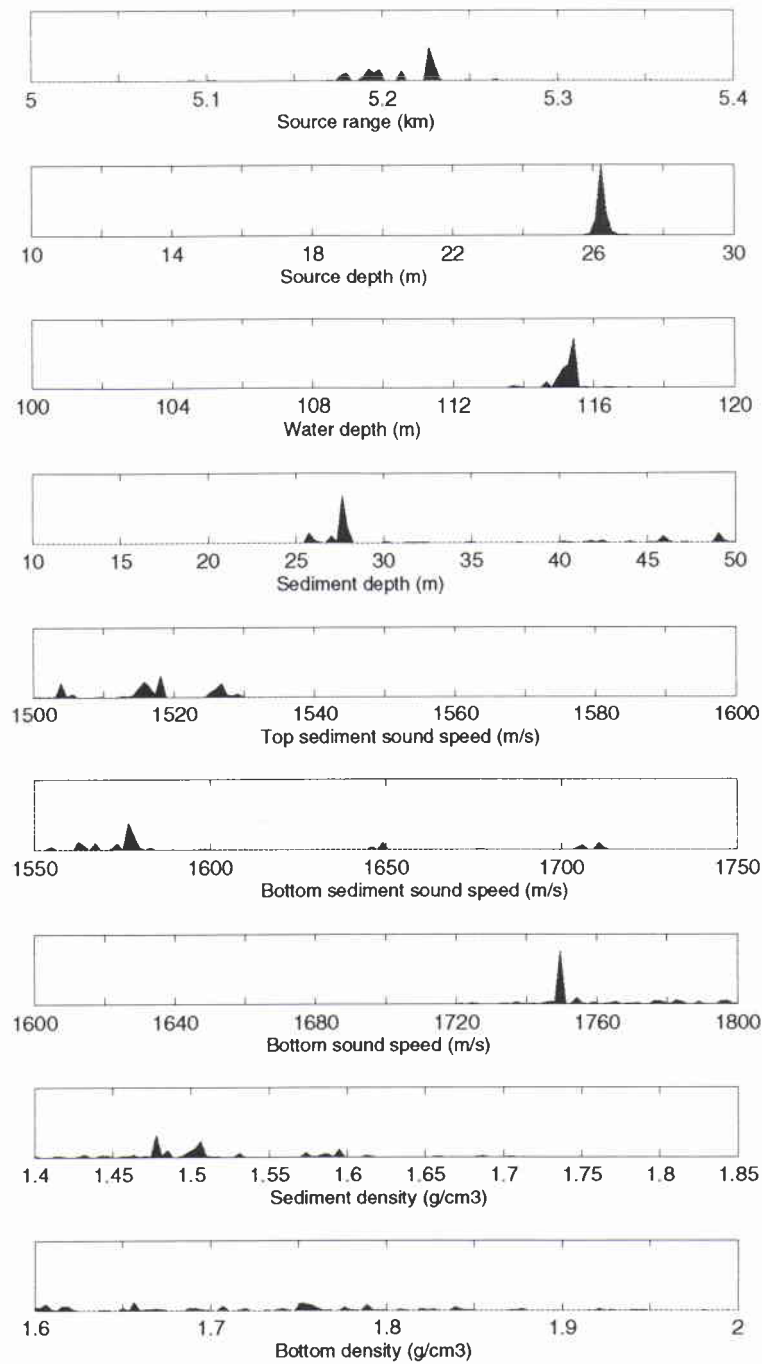


Figure 16 The a posteriori distributions for the waa case.

SACLANTCEN SM-333

NATO UNCLASSIFIED

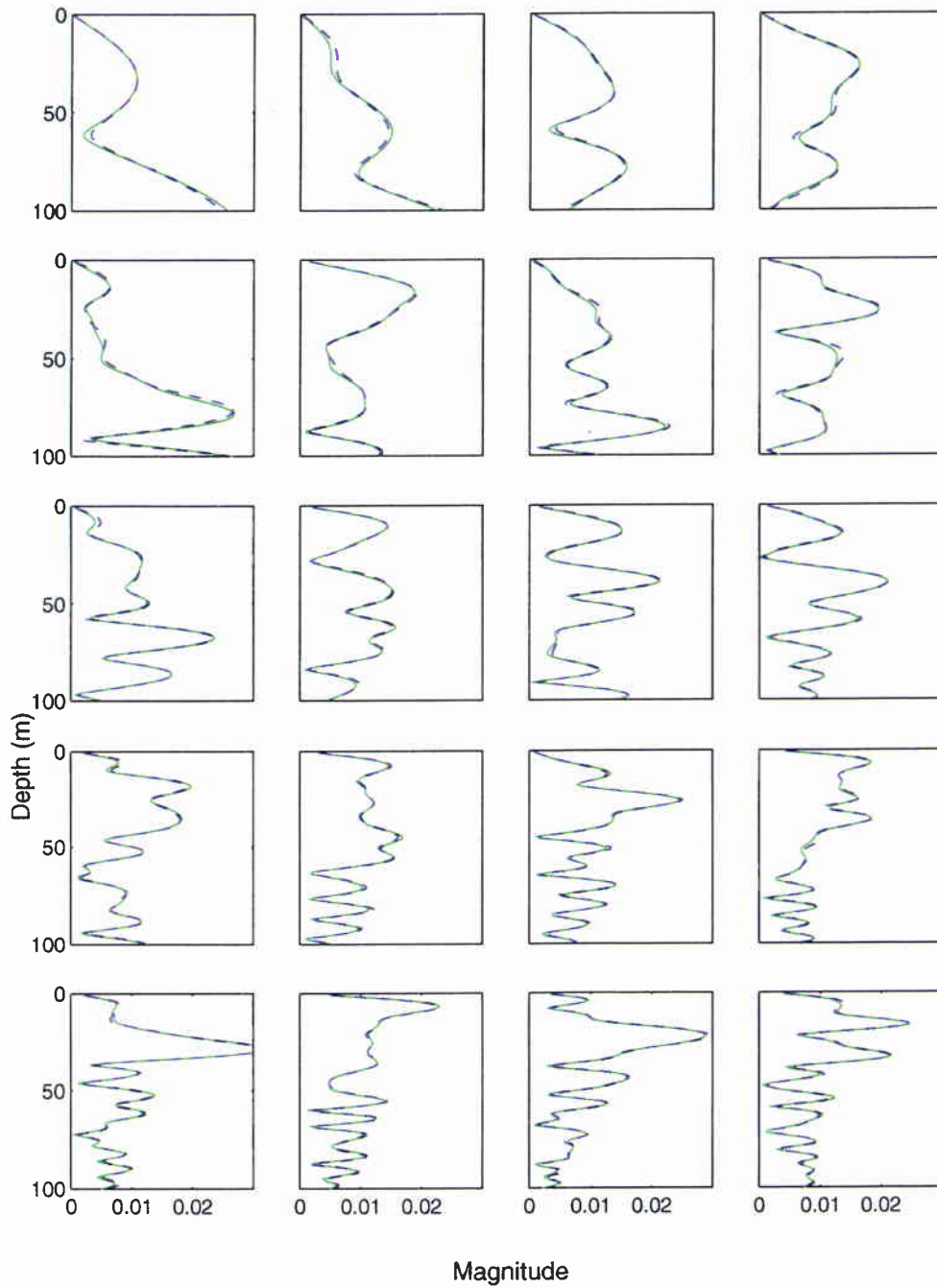


Figure 17 *Magnitude of the field for the waa case at every fifth of 96 frequencies. The observed data (green line), and the computed data (blue line) with the best set of parameters at every fifth frequency, i.e. from 25 Hz to 200 Hz in steps of 5 Hz.*

NATO UNCLASSIFIED

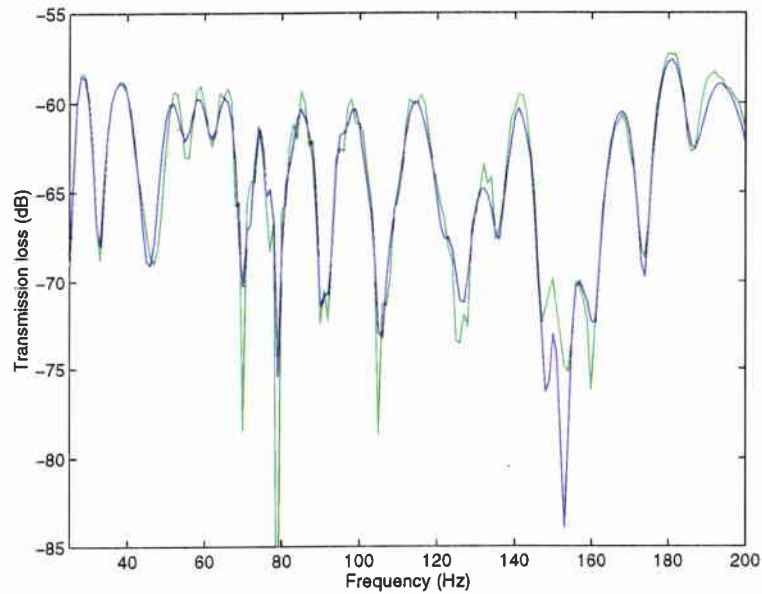


Figure 18 *The transmission loss for the mf case. The green line is the observed data and the blue line the modeled data with the best environment. The agreement is quite good.*

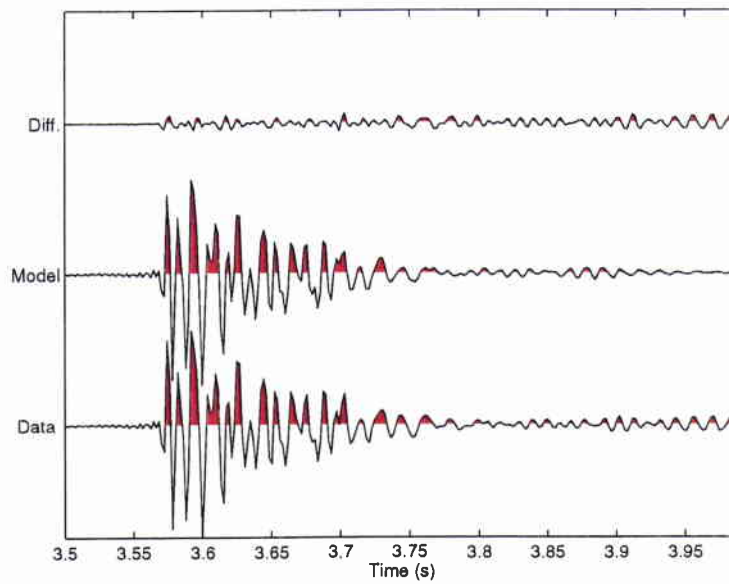


Figure 19 *The time series for the mf case. The observed data, the modeled data and the difference between the two time series are plotted.*

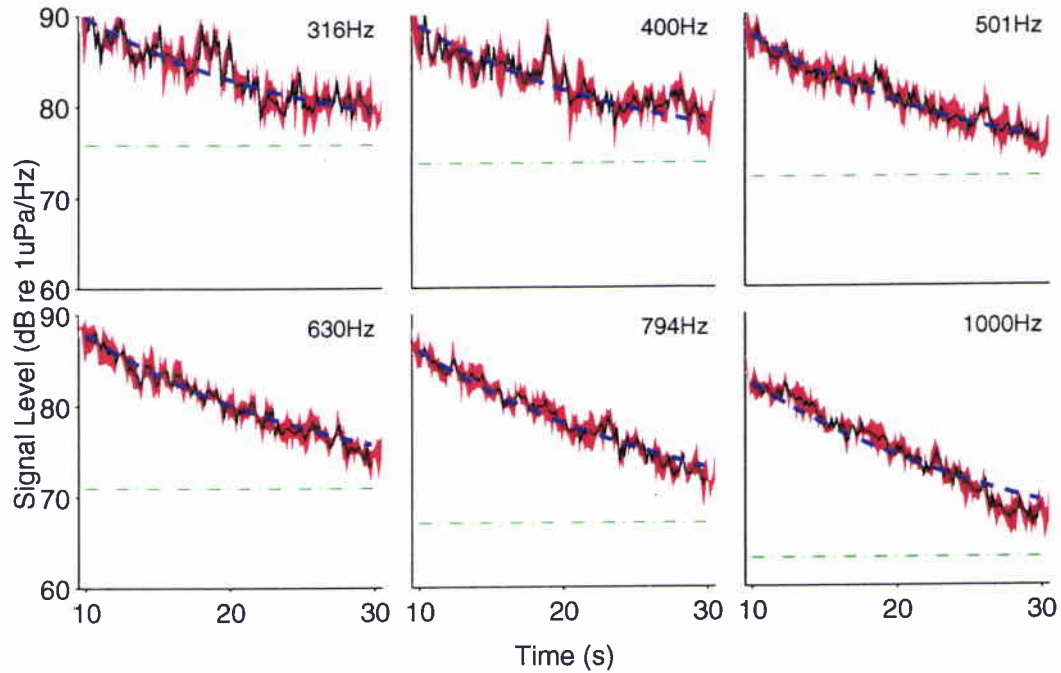


Figure 20 *Inversion of reverberation data from several shots (File: popp_grad3). The green dash-dotted line shows the estimated noise level in the data, the red area indicates the variation of the five used shots, the black thin full line is the average signal level of the shots. This is here computed as the first eigenvector of the covariance matrix, where the covariance matrix is constructed as an ensemble average of several shots. The thick blue dashed line shows the synthetic data generated from the best estimated model.*

- 5 Thickness of each layer in sediment; **index** refers to the actual layer.
- 6 Density in sediment; **index** refers to the actual layer.
- 8 Source depth.
- 11 Shape-function coefficient; **index** points to the number.
- 15 Receiver depth.
- 16 Depth of speed point in water; **index** refers to the actual point.
- 17 Lambert scattering strength.
- 18 Power in Lambert's law.

11.7.2 POPP example. File: revpopp

This example is based on reverberation data from the North Elba site. There is not very much information in the data, thus we are mainly inverting for the Lambert scattering strength and the sound speed of the bottom halfspace. The options

SACLANTCEN SM-333**NATO UNCLASSIFIED**

indicate that we are reading observed data using option d, matching the absolute level, option X, and we are plotting the observed data and the best fitting model, option p. We use GA with 1 population with 1000 individuals; 1000 forward models are used in total. The ambient noise level is determined from the data.

```

Elba reverberation data
d X p                                ! options

1 64 1                                ! niter, q, npop
0.8 0.5 0.05                          ! px, pu, pm

130. 1.0 0.                          ! water depth, density, attenuation
    0. 1521.
    36. 1519.
    38. 1514.
    50. 1511.
    130. 1508.

-1. -1.                              ! no more water points
0. 0.                                ! shear values not implemented
-1. 1.8 1600. 0.09                    ! sed thknss, dens, speed, att (-1= last lay)
0. 0.                                ! roughness surface, bottom
2 398 630.                          ! Nfreq, freq
1 60.                                ! Nsd sd
1 60.                                ! Nrd rd
-1 -1                                ! No. of comp. layers, modes (-1 = default)
2                                     ! isb (1=surf, 2=bott)
1                                     ! tau0
198.8 197                            ! source level 1...Nfreq
-30 -30                              ! lambert coefficient 1...Nfreq
76 76                                ! ambient noise level 1...Nfreq

10.03067 30 0.1706666                ! tmin, tmax, tinc

6                                     ! nparm
8 1 50 70 128                        ! source depth
3 1 1520 1620 128                    ! vel sed 1
1 1 120. 140. 128                    ! water depth
15 1 50. 70. 128                     ! recvr depth
17 1 -50 -20 128                     ! lambert DMU for first frequency
17 2 -50 -20 128                     ! lambert DMU for second frequency

```

11.8 Forward model: TPEM

The Terrain Parabolic Equation Model (TPEM) calculates the electromagnetic field in height and range for electromagnetic propagation in the troposphere. It allows for range-dependent refractivity environments and variable terrain. TPEM is based on

a source code originally developed by Fred Tappert from the University of Miami, for propagation over a smooth surface. It is a pure PE model based on the split-step Fourier method and is described in [8, 9] (Email: barrios@nosc.mil). A good general description of electromagnetic field propagation in the troposphere is given in [53]. The version used in **SAGA** is based on **TPEM** 1.0 originally developed for a PC.

For a parabolic equation method it is required that the solution converge for smaller step sizes in range and depth. The size of these steps depend on the particular environment and the terrain profile. Based on a few environments we have empirically found the following step sizes to be adequate: About $\lambda/3$ in depth, and in range either 10λ for range-dependent or 200λ for range-independent environments. For range-dependent environments smaller steps are required because the field generally varies more with range. These limits are implemented in the broadband version of **TPEM** and are significantly larger than the ones used in [9].

In **SAGA** the range steps are increased by a factor 5, i.e. to 50λ for range-dependent and 1000λ for range-independent environments. The step size in meters are written out during execution of the code. A doubling in either step size causes roughly a halving in CPU time and, therefore, a significant CPU-time saving can be obtained by changing these default values.

The default step sizes can be changed by specifying a negative center frequency and in the next line specifying

```
dr dz ln
```

dr dz is specified in meters and **ln** is the FFT-power for the Fourier transform in height. The maximum height of the computational domain is $dz \cdot 2^{1n}$.

The maximum height is specified in the input file (either as antenna height or maximum refractivity profile height). This height combined with Δz will determine the number of points needed in the Fourier transform. Therefore, in order to reduce CPU-time these heights should be as small as possible. The program is also much faster if running in a range-independent mode, which is specified by using "0" terrain points. This is partly because larger range steps can then be taken.

The magnitude of the field is best computed by using the **SAGA** option G. If the magnitude of the field is required in dB, then use option I.

11.8.1 Pointers in **TPEM**

TPEM uses three pointers to specify a variable. The pointer **parm** is used to map between the optimization variable and the environmental parameter to be optimized.

The second parameter, **index**, points to the layer; for some parameters **index** is not important (e.g. source height). The pointer **parm** can take the following values:

- 1 (**rf.refmsl**) Refractivity points in M-units; **index** indicates at which height and **index2** at which range.
- 2 (**rf.hmsl**) Height corresponding to a refractivity point; **index** indicates at which height and **index2** at which range.
- 3 (**tr.terx**) Terrain x-coordinates.
- 4 (**tr.tery**) Terrain y-coordinates.
- 5 Source height.
- 6 (**sv.antht**) Antenna height.
- 9 Source-receiver range.
- 11 Shape-function coefficient; **index** points to the number.

11.8.2 TPTEM example. File: **tpem_ex**

This example has been used in paper [39]. First the data from the initial model is written to the *.obs file, option W. This file can then be copied to the *.in file and be used in subsequent inversions. The data format is the complex-valued field on a vertical array, option e. The objective function is incoherent Bartlett, option f. This type of environment is efficiently described using a tri-linear profile, see Fig. 21, where the refractivity profile is described in terms of M-deficit, base height and thickness. This type of profile is efficiently described using shape functions and read from the *.eof file, option E. When running POST, a plot of the observed data and the synthetic data for the best environment is produced, option E. We are optimizing over the three environmental parameters: M-deficit, base height and thickness.

TPTEM optimizing for the tri-linear profiles

W f e p E ! options

2000 64 20 ! niter, q, npop
0.8 0.5 0.05 ! px, pu, pm

1000. : Frequency in MHz (100 to 20000)
1 : Field type (1:complex field, 0:magnitude)
50. : Transmitter height in meters
0 : Ant pattern(0=Omni, Gaussian, Sin(x)/x, Csc-Sq, Ht-finder)
30 : Beamwidth in degrees (full 3 dB to 3 dB width)
0 : Elevation angle in degrees
100 : Maximum receiver height in meters
90000. : Maximum receiver range in meters
1 50 : Number of range, height points to output

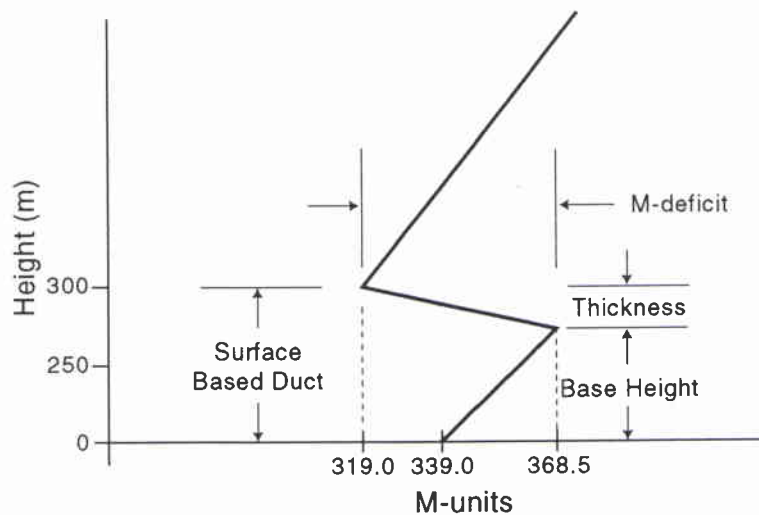


Figure 21 Modified vertical refractivity profile for a tri-linear profile.

```

1  4          ! refractivity profile points (Range, Height)

0              !First profile is at range zero
0.            339.          ! Refractivity
250.          368.5
300.          319.
400.          330.8

0              ! terrain points => range-independent env.

3  ! nparm
11 2 1 1 100 128 ! refraction deficit
11 3 1 1 400 128 ! base height
11 4 1 1 100 128 ! thickness

```

The shape function file contains the three basic parameters used to describe the atmospheric profile in Fig. 21, see Ref. [39].

```

! shape function file for the TPEM
5 7 1          ! No. of shape functions, No. of points, No. of blocks
1 1 1          ! refractivity 1 (bottom)
1 2 1          ! refractivity 2
1 3 1          ! refractivity 3
1 4 1          ! refractivity 4 (top)
2 2 1          ! height
2 3 1          ! height
2 4 1          ! height (top)
! Block 1: coupling of all parameters using the tri-linear profiles

```

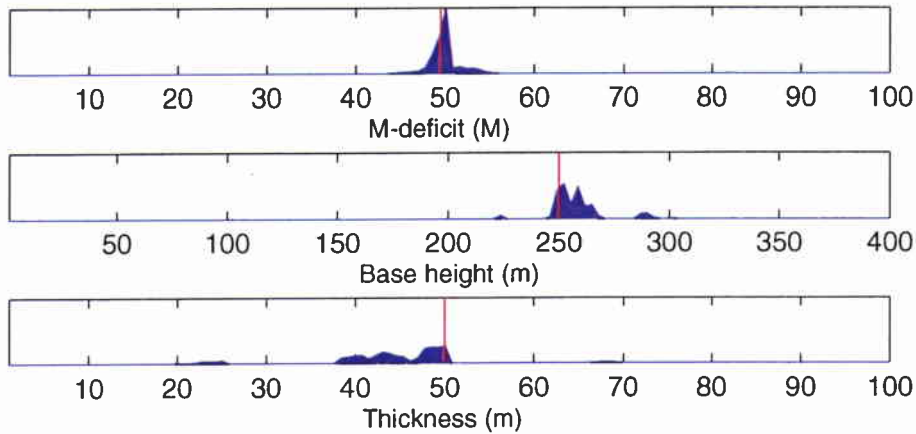


Figure 22 The *a posteriori* distributions for the `tpem_ex` case. The red line indicates the true value.

```
! c(1) Deficit base thick top
5 7
  1      0      0      0      0      ! first refractivity point
  1      0  0.118      0      0      ! second
  1     -1  0.118      0      0      ! third
  1     -1      0 -0.118 .118      ! fourth
  0      0      1      0      0      ! second height
  0      0      1      1      0      ! third
  0      0      0      0      1      ! fourth
! Now comes the starting values
339  49.5  250   50 1000      ! starting values
```

The *a posteriori* distributions for this inversion are shown in Fig. 22. When the distributions cluster around a certain value it is an indication that a parameter is well determined. The best obtained fit is shown in Fig. 23. Since there is no noise in the observed data the match is quite good.

It is also of interest to plot the ambiguity surface between two parameters. This will illustrate the relative importance of the two parameters. The contour plot is produced by introducing option C in the option line. When running SAGA it will then produce a contour plot of the objective function between the first two specified optimization parameters. In order to reduce the number of forward modeling runs, it is advisable to reduce the discretization for each of the two parameters to about 40. An example of an ambiguity surface is given in Fig. 24. The ambiguity surface is shown using either complex-valued received signal or just using the magnitude. The magnitude is produced by introducing option G in the option-line.

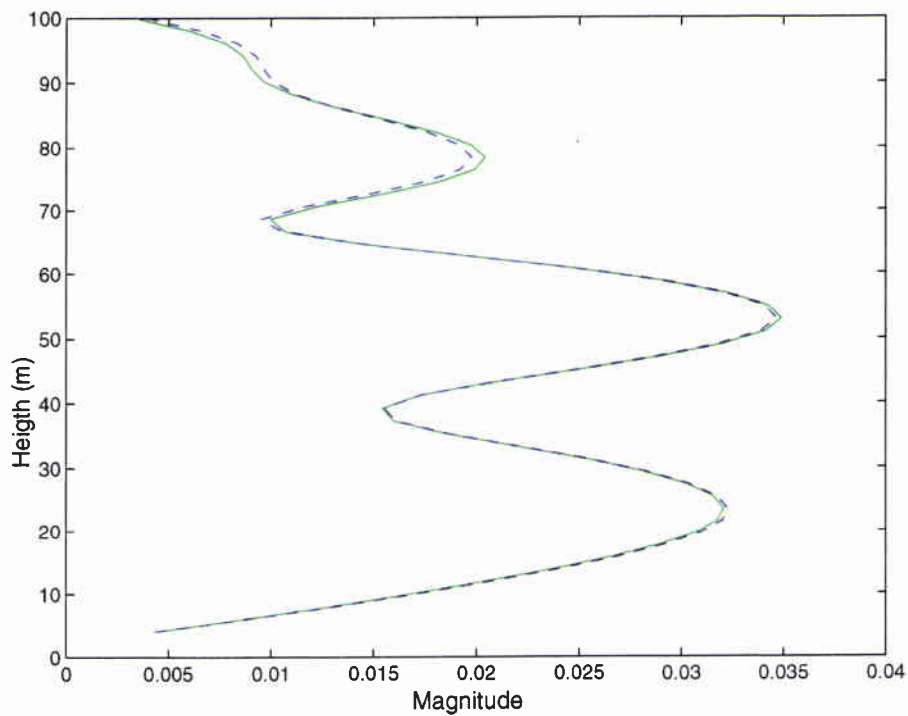


Figure 23 *The fit for the **tpem_ex** case. The magnitude of the observed data (green line), and the magnitude of the synthetic data (blue line) with the best set of parameters.*

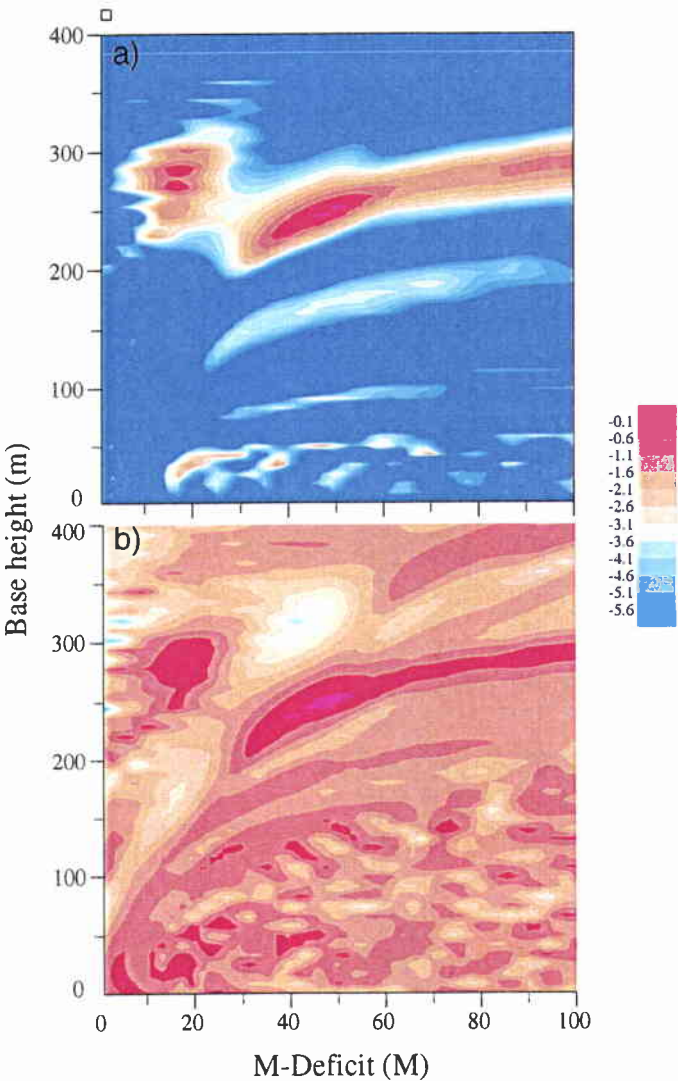


Figure 24 *Contour plot of the ambiguity function between base height and M-deficit for the tpem_ex case. Phase (a) or magnitude (b) of the received signal are used in the processing.*

References

- [1] F.B. Jensen and M.C. Ferla, "SNAP: The SACLANTCEN normal-mode acoustic propagation model," SM-121, SACLANT Undersea Research Centre, La Spezia, Italy (1979).
- [2] H. Schmidt, "SAFARI: Seismo-acoustic fast field algorithm for range independent environments. User's guide," SR-113, SACLANT Undersea Research Centre, La Spezia, Italy (1987).
- [3] H. Schmidt, "OASES Version 1.6: Application and upgrade notes," Massachusetts Institute of Technology, Cambridge, MA (1993).
- [4] D.D. Ellis, "A shallow water normal mode reverberation model," J. Acoust. Soc. Am. **97**, 2804–2814 (1995).
- [5] P.L. Nielsen, Personal communication (1997).
- [6] S.J. Levinson, E.K. Westwood, R.A. Koch, S.K. Mitchell and C.V. Sheppard, "An efficient and robust method for underwater acoustic normal mode computations," J. Acoust. Soc. Am. **97**, 1576–1585 (1995).
- [7] E.K. Westwood, C.T. Tindle and N.R. Chapman, "A normal mode model for acousto-elastic ocean environments," J. Acoust. Soc. Am. **100**, 3631–3645 (1996).
- [8] A.E. Barrios, "Parabolic equation modeling in horizontally inhomogeneous environments," IEEE Trans. on Ant. and Prop. **36**, 791–797 (1992).
- [9] A.E. Barrios, "A terrain parabolic equation model for propagation in the troposphere," IEEE Trans. on Ant. and Prop. **42**, 90–98 (1994).
- [10] P. Gerstoft, "Inversion of seismoacoustic data using genetic algorithms and *a posteriori* probability distributions," J. Acoust. Soc. Am. **95**, 770–782 (1994).
- [11] P. Gerstoft, "Inversion of acoustic data using a combination of genetic algorithms and the Gauss–Newton approach," J. Acoust. Soc. Am. **97**, 2181–2190 (1995).
- [12] M.D. Collins, W.A. Kuperman and H. Schmidt, "Nonlinear inversion for ocean-bottom properties," J. Acoust. Soc. Am. **92**, 2770–2783 (1992).
- [13] L. Ingber, "Very fast simulated reannealing," Mathematical Computer Modeling **12**, 967–973 (1989).

- [14] L. Ingber, "Simulated annealing: Practice versus theory," *Mathematical Computer Modeling* **18**, 29–57 (1993).
- [15] A. Caiti, T. Akal and R.D. Stoll, "Estimation of shear wave velocity in shallow marine sediments," *IEEE J. Oceanic Eng.* **19**, 58–72 (1994).
- [16] P. Gerstoft and A. Caiti, "Acoustic estimation of bottom parameters: error bounds by local and global methods," in *Second European Conference on Underwater Acoustics*, edited by L. Bjørnø (European Commission, Luxembourg, 1994), pp. 887–892.
- [17] F.B. Jensen, W.A. Kuperman, M.B. Porter and H. Schmidt, *Computational Ocean Acoustics* (American Institute of Physics, New York, 1994).
- [18] A. Tolstoy, *Matched Field Processing for Underwater Acoustics* (World Scientific, Singapore, 1993).
- [19] S.M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory* (Prentice Hall, Englewood Cliffs, NJ, 1993).
- [20] Y. Bard, *Nonlinear Parameter Estimation* (Academic Press, San Diego, CA, 1974).
- [21] W. Menke, *Geophysical Data Analysis: Discrete Inverse Theory* (Academic Press, San Diego, CA, 1989).
- [22] R.A. Parker, *Geophysical Inverse Theory* (Princeton University Press, Princeton, NJ, 1994).
- [23] A. Tarantola, *Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation* (Elsevier, Amsterdam, 1987).
- [24] M.K. Sen and P.L. Stoffa, *Global Optimization in Geophysical Inversion* (Elsevier, Amsterdam, 1995).
- [25] M.D. Collins and W.A. Kuperman, "Focalization: Environmental focusing and source localization," *J. Acoust. Soc. Am.* **90**, 1910–1922 (1991).
- [26] C.E. Lindsay and N.R. Chapman, "Matched field inversion for geophysical parameters using adaptive simulated annealing," *IEEE J. Oceanic Eng.* **18**, 224–231 (1993).
- [27] S.E. Dosso, M.L. Yersey, J.M. Ozard and N.R. Chapman, "Estimation of ocean bottom properties by matched-field inversion of acoustic field data," *IEEE J. Oceanic Eng.* **18**, 232–239 (1993).
- [28] J.A. Scales, M.L. Smith and T.L. Fisher, "Global optimization methods for highly nonlinear inverse problems," *J. Comp. Phys.* **103**, 258–268 (1992).

- [29] M. Sambrige and G. Drijkoningen, "Genetic algorithms in seismic waveform inversion," *Geophys. J. Int.* **109**, 323–343 (1992).
- [30] P.L. Stoffa and M.K. Sen, "Multiparameter optimization using genetic algorithms: Inversion of plane wave seismograms," *Geophysics* **56**, 1794–1810 (1991).
- [31] M.K. Sen and P.L. Stoffa, "Rapid sampling of model space using genetic algorithms: examples from seismic waveform inversion," *Geophys. J. Int.* **108**, 281–292 (1992).
- [32] S.M. Jesus and A. Caiti, "Estimating geoacoustic bottom properties from towed array data," *J. Comp. Acoust.* **4**, 273–290 (1996).
- [33] Z.-H. Michalopoulou, H. Martynov and M.B. Porter, "Simulated annealing and genetic algorithms for broadband source localization," in *Third European Conference on Underwater Acoustics*, edited by J.S. Papadakis (Crete University Press, Crete, 1996), pp. 409–414.
- [34] O. Lotsberg and S.M. Jesus, "Matched field inversion of geoacoustic properties from towed array data in shallow water," in *Third European Conference on Underwater Acoustics*, edited by J.S. Papadakis (Crete University Press, Crete, 1996), pp. 601–606.
- [35] M.J. Rendas and G. Bienvenu, "Tuning genetic algorithms for underwater acoustics using *a priori* statistical information," *Proc. IEEE ICASSP-97*, Munich (1997).
- [36] P. Gerstoft and C.F. Mecklenbräuker, "Ocean acoustic inversion with estimation of *a posteriori* probability distributions," submitted to *J. Acoust. Soc. Am.* (1997).
- [37] M.B. Porter and A. Tolstoy, "The matched field processing benchmark problems," *J. Comp. Acoust.* **2**, 161–185 (1994).
- [38] D.F. Gingras and P. Gerstoft, "Inversion for geometric and geoacoustic parameters in shallow water: Experimental results," *J. Acoust. Soc. Am.* **97**, 3589–3598 (1995).
- [39] D.F. Gingras, P. Gerstoft and N.L. Gerr, "Electromagnetic matched field processing: Basic concepts and tropospheric simulations," to appear *IEEE Trans. on Ant. and Prop.* (1997).
- [40] H.G. Schneider, "Acoustic models at SACLANTCEN," SM-285, SACLANT Undersea Research Centre, La Spezia, Italy (1995).
- [41] P. Gerstoft, "Global inversion by genetic algorithms for both source position and environmental parameters," *J. Comp. Acoust.* **2**, 251–266 (1994).

- [42] C.H. Harrison and J.A. Harrison, "A simple relationship between frequency and range averages for broadband sonar," *J. Acoust. Soc. Am.* **97**, 1314–1317 (1995).
- [43] M. Lambert, "Inversion of seismo-acoustic real data using genetic algorithms," SM-276, SACLANT Undersea Research Centre, La Spezia, Italy (1994).
- [44] J.-P. Hermand and P. Gerstoft, "Inversion of broadband multitone acoustic data from the Yellow Shark summer experiments," *IEEE J. Oceanic Eng.* **21**, 324–346 (1996).
- [45] D.J. Thompson, "Jackknifing using multiple-window spectra," *Proc. IEEE ICASSP-94*, Adelaide (1994).
- [46] C.F. Mecklenbräuker, D. Maiwald and J.F. Böhme, "F-test in matched field processing: identifying multimode propagation", *Proc. IEEE ICASSP-95*, Detroit (1995).
- [47] P. Gerstoft and D.F. Gingras, "Parameter estimation using multi-frequency range-dependent acoustic data in shallow water," *J. Acoust. Soc. Am.* **99**, 2839–2850 (1996).
- [48] G. Haralabus and P. Gerstoft, "Stability of parameter estimates using multi-frequency inversion techniques," SR-253, SACLANT Undersea Research Centre, La Spezia, Italy (1996).
- [49] G. Haralabus and P. Gerstoft, "Variability of multi-frequency parameter estimates in shallow water," in *Third European Conference on Underwater Acoustics*, edited by J.S. Papadakis (Crete University Press, Crete, 1996), pp. 355–360.
- [50] N.R. Chapman and A. Tolstoy, *Geoacoustic Benchmark Inversion Workshop*, Vancouver (1997).
- [51] M. Siderius, P. Gerstoft and P.L. Nielsen, "Broadband acoustic inversion using a sparse array," in *Matched Field Inversion Workshop*, Vancouver (1997).
- [52] D.D. Ellis and P. Gerstoft, "Using inversion techniques to extract bottom scattering strengths and sound speed from shallow water reverberation data," in *Third European Conference on Underwater Acoustics*, edited by J.S. Papadakis (Crete University Press, Crete, 1996), pp. 887–892.
- [53] H.V. Hitney, "Refractive effects from VHF to EHF: Part A, Propagation mechanisms," AGARD Lecture Series, LS-196 (1994).
- [54] A. J. W. Duijndam, "Bayesian estimation in seismic inversion. Part I: Principles," *Geophysical Prospecting* **36**, 878–898 (1988).

SACLANTCEN SM-333

NATO UNCLASSIFIED

- [55] J. Lefebvre, H. Roussel, E. Walter, D. Lecointe and W. Tabbara, "Prediction from wrong models: The Kriging approach," IEEE Ant. Prop. Magazine, **38**(4), 35-45 (1996).
- [56] D. Maiwald and J.F. Böhme, "Multiple testing for seismic data using bootstrap," Proc. IEEE ICASSP-94, **6**, Adelaide (1994), pp. 89-92.
- [57] M.L. Hinich, "Maximum likelihood estimation of a radiating source in a waveguide", J. Acoust. Soc. Am. **66**, 480-483 (1977).
- [58] E.C. Shang, "Source depth estimation in wave guides," J. Acoust. Soc. Am. **86**, 1960-1964 (1985).
- [59] S. Rajan, "Waveform inversion of geoacoustic parameters in the ocean bottom," J. Acoust. Soc. Am. **95**, 3228-3241 (1992).

NATO UNCLASSIFIED

Annex A

Likelihood functions for a vertical array

Depending on the model used for the error or noise distribution of the data, a specific likelihood function must be used. Thus prior knowledge about the error function is required. Usually this is not directly available and some simple and reasonable models must be used. Two special cases are considered for multi-frequency vertical array data. The noise distribution on each phone is assumed complex Gaussian. First, in Sect. A.1 it is assumed that the noise is independent on each phone and, second, in Sect. A.2 it is assumed independent for each *significant* mode.

Often, a distinction is made between errors due to noise in the data and errors due to an incomplete forward model, because neither the theory nor the environmental model is adequate. If both error types belong to the same distribution, there is no reason to consider them separately [54]. Here only one error term is considered.

Recently, there has been progress in describing both errors using Kriging [55], in which errors due to noise and incomplete forward modeling are considered separately. Both noise and modeling errors are assumed zero-mean Gaussian and independent. The modeling errors are assumed to possess a given correlation structure depending on the “distance” between two environmental models. This correlation structure is chosen empirically. Clearly, the same value of the model parameters should correspond to the same value of the modeling error.

A.1 Multi-frequency matched field processing

The relation between the observed complex-valued data vector $\mathbf{q}(\omega)$ on an N -element hydrophone antenna and the predicted data $\mathbf{p}(\mathbf{m}, \omega)$ at a angular frequency ω is described by the model

$$\mathbf{q}(\omega) = \mathbf{p}(\mathbf{m}, \omega) + \mathbf{e}(\omega) , \quad (29)$$

where $\mathbf{e}(\omega)$ is the error term. The predicted data is given by $\mathbf{p}(\omega) = \mathbf{w}(\mathbf{m}, \omega)S(\omega)$, where the complex deterministic source term $S(\omega)$ is unknown. The transfer function $\mathbf{w}(\mathbf{m}, \omega)$ is obtained using an acoustic propagation model and an environmental model $\mathbf{m}$.

The errors are assumed to be additive; they stem from many sources: errors in describing the environment, errors in the forward model, instrument and measure-

ments errors, and noise in the data. For the predicted acoustic field “reasonably close” to the true field, this error term is assumed complex Gaussian distributed, stationary with zero mean and diagonal covariance matrix $\nu(\omega)\mathbf{I}$, where the error power spectrum ν is unknown. Or written compactly $\mathbf{e}(\omega) \sim \text{CN}(0, \nu(\omega)\mathbf{I})$, where CN symbolizes the Complex Gaussian distribution. Thus, the data $\mathbf{q}(\omega)$ on the receiving array are also complex Gaussian distributed, $\mathbf{q}(\omega) \sim \text{CN}(\mathbf{p}(\omega, \mathbf{m}), \nu(\omega)\mathbf{I})$. For the derivation of a maximum likelihood estimate, it is further assumed that the data for each frequency bin are uncorrelated across frequency and time snapshot and that for each time snapshot the source term $S(\omega)$ can vary whereas the error power $\nu(\omega)$ is constant.

In the following, we will often abbreviate $\mathbf{q}_l = \mathbf{q}(\omega_l)$, where $\{\omega_l | l = 1, \dots, L\}$ is a suitable set of frequency bins. We have $\mathbb{E}\mathbf{q}_l\mathbf{q}_l^\dagger = \mathbf{R}_l = \mathbf{p}_l\mathbf{p}_l^\dagger + \nu_l\mathbf{I}$. Under these assumptions the likelihood function evaluates to

$$\mathcal{L} \propto \prod_{l=1}^L \frac{1}{\nu_l^N} \exp\left(-\frac{\phi_l}{\nu_l}\right), \quad (30)$$

where²

$$\phi_l = \text{tr}\hat{\mathbf{R}}_l - \frac{\mathbf{w}_l^\dagger \hat{\mathbf{R}}_l \mathbf{w}_l}{\mathbf{w}_l^\dagger \mathbf{w}_l}. \quad (31)$$

Optimization for ν_l yields the closed form ML solution

$$\hat{\nu}_l = \frac{1}{N} \phi_l. \quad (32)$$

The $N \times N$ Hermitian matrix $\hat{\mathbf{R}}_l$ denotes the estimated cross-spectral density matrix of the observed data “in phone-space”, see Sect. A.3. With these definitions, the ML objective function can be written as

$$\phi = \prod_{l=1}^L \phi_l = \prod_{l=1}^L \left(\text{tr}\hat{\mathbf{R}}_l - \frac{\mathbf{w}_l^\dagger \hat{\mathbf{R}}_l \mathbf{w}_l}{\mathbf{w}_l^\dagger \mathbf{w}_l} \right). \quad (33)$$

Using a global optimization procedure, the minimum $\hat{\phi}^{\text{ML}}$ for the ML solution $\hat{\mathbf{m}}^{\text{ML}}$ is estimated. The estimate, Eq. (32), is biased. The bias stems from the degrees of freedom in the estimated parameters: source signal S and nonlinear parameters $\mathbf{m}$, see [56]. For simplicity this bias is neglected here. As the noise power spectral density now is estimated, Eq. (32), the likelihood function is given by

$$\begin{aligned} \mathcal{L}(\mathbf{m}) &= p(\mathbf{m}|\mathbf{q}) \\ &\propto \prod_{l=1}^L (\hat{\nu}_l^{\text{ML}})^{-N} \exp\left(-\frac{\phi_l(\mathbf{m})}{\hat{\nu}_l^{\text{ML}}}\right) \\ &\propto \prod_{l=1}^L \exp\left(-N \frac{\phi_l(\mathbf{m}) - \hat{\phi}_l^{\text{ML}}}{\hat{\phi}_l^{\text{ML}}}\right). \end{aligned} \quad (34)$$

²† refers to the Hermitian transpose

The problem, as addressed above, is then to integrate this multi-dimensional probability distribution. Often this integral can be evaluated with sufficient accuracy using the information from the global search. In some cases it might be necessary to increase the sampling of the model space in order to obtain convergence.

According to Eq. (34), the likelihood function has a stronger maximum when more hydrophones are used. When inverting observed data, there is a limit to how much useful information can be obtained by adding additional hydrophones, as they then become strongly correlated. At high SNR it is expected that the main error contribution is due to inadequate forward modeling. Further, the number of uncorrelated hydrophones is approximately the same as the number of propagating modes, because this limits the degrees of freedom in the random part of the acoustic wave field. The number of uncorrelated hydrophones is estimated as the rank of the covariance matrix.

A.2 Multi-frequency matched mode processing

Normal modes provide a complete description of the field at long ranges, and thus one can equivalently process the data in phone-space or in mode-space. The matched mode approach is described by Tolstoy [18], Hinich [57], and Shang [58]. Modal processing is discussed here as an alternative noise estimate when there are more hydrophones than propagating modes.

We assume that the observed field of N sensors can be approximately expressed via a set of J *significant* modes, expressed in a $N \times J$ matrix $\mathbf{V}(\omega) = (\mathbf{v}_1, \dots, \mathbf{v}_J)$, and the corresponding complex valued modal amplitudes³ $\check{\mathbf{q}}(\omega) = (\check{q}_1, \dots, \check{q}_J)'$ are:

$$\mathbf{q}(\omega) \approx \sum_{j=1}^J \mathbf{v}_j \check{q}_j = \mathbf{V}(\omega) \check{\mathbf{q}}(\omega) . \quad (35)$$

It is assumed that we have more hydrophones than modes. We can invert this relationship in a least-squares sense and estimate the vector of modal amplitudes $\check{\mathbf{q}}_l = \check{\mathbf{q}}(\omega_l)$ in mode-space from the observation $\mathbf{q}_l = \mathbf{q}(\omega_l)$ in phone-space,

$$\check{\mathbf{q}}_l = (\mathbf{V}_l^\dagger \mathbf{V}_l)^{-1} \mathbf{V}_l^\dagger \mathbf{q}_l . \quad (36)$$

Note that the modes and modal amplitudes depend on the environment. When optimizing the environment $\mathbf{m}$, the modal amplitudes will change with the environment.

We assume a simple relationship between the observed modal amplitudes $\check{\mathbf{q}}_l$ and synthetic generated modal amplitudes,

$$\check{\mathbf{q}}_l = \check{\mathbf{p}}_l(\mathbf{m}) + \check{\mathbf{e}}_l(\mathbf{m}) , \quad (37)$$

³The symbol $\check{}$ refers to the mode-space.

where $\check{\mathbf{p}}_l(\mathbf{m}) = S_l \check{\mathbf{w}}_l(\mathbf{m})$ is the complex-valued modal amplitudes of the synthetic data and $\check{\mathbf{e}}_l$ represents the error term for each mode. It is assumed that the noise covariance matrix is diagonal for the J significant modes and the noise power $\check{\nu}$ is identical for all J modes.

Using a similar approach to that in Sect. A.1, the objective function is

$$\check{\phi}_l = \text{tr} \hat{\mathbf{R}}_l - \frac{\check{\mathbf{w}}_l^\dagger \hat{\mathbf{R}}_l \check{\mathbf{w}}_l}{\check{\mathbf{w}}_l^\dagger \check{\mathbf{w}}_l}, \quad (38)$$

where $\hat{\mathbf{R}}_l$ is the estimated covariance matrix between the modes. But using the expression for the modes, Eq. (35), the objective function in mode-space, Eq. (38), is seen to be equivalent to the objective function in phone-space, i.e. Eq.(31) expressed in phone-space,

$$\check{\phi}_l \approx \phi_l. \quad (39)$$

The noise estimate is obtained using the approximation in Eq. (35),

$$\hat{\nu}_l = \frac{1}{J} \hat{\nu}_l^{\text{ML}} \approx \frac{1}{J} \hat{\phi}_l^{\text{ML}}. \quad (40)$$

The likelihood function becomes

$$\begin{aligned} \mathcal{L}(\mathbf{m}) = p(\mathbf{m}|\mathbf{q}) &\propto \prod_{l=1}^L (\hat{\nu}_l^{\text{ML}})^{-J} \exp \left(-\frac{\check{\phi}_l(\mathbf{m})}{\hat{\nu}_l^{\text{ML}}} \right) \\ &\propto \prod_{l=1}^L \exp \left(-J \frac{\phi_l(\mathbf{m}) - \hat{\phi}_l^{\text{ML}}}{\hat{\phi}_l^{\text{ML}}} \right). \end{aligned} \quad (41)$$

The advantage of this formulation is that it does not depend directly on the number of hydrophones, but only on the number of propagating modes. For many hydrophones ($N \gg J$) this likelihood function seems more realistic. The precise value of J is not yet clear. For simplicity, J is assumed to be independent of frequency. Only the objective function is affected by the choice of J . All the propagating modes are incorporated in the forward model.

A.3 Estimation of the covariance matrix

In order to estimate the covariance matrix $\mathbf{R}_l$, the received time signal is divided into K time frames. Each frame was short-time Fourier transformed using the multiple-windows technique described in Refs. [45, 46],

$$\mathbf{q}_{k,p}(\omega) = \sum_{t=0}^{T-1} \nu_t^p \mathbf{q}(t + kT) e^{-j\omega t}, \quad \text{for } \begin{cases} k = 0, \dots, K-1 \\ p = 0, \dots, P-1 \end{cases} \quad (42)$$

where ν^p is a special set of P orthonormal data tapers [45, 46]. The correlation matrix $\mathbf{R}$ was estimated at each selected frequency ω_l as the ensemble average, i.e.

$$\hat{\mathbf{R}}(\omega_l) = \frac{1}{KP} \sum_{k=0}^{K-1} \sum_{p=0}^{P-1} \mathbf{q}_{k,p}(\omega_l) \mathbf{q}_{k,p}^\dagger(\omega_l). \quad (43)$$

In order to obtain a good estimation of the noise, it is required that $KP \gg N$, where N is the number of hydrophones. In order to “just” estimate the signal and the unknown parameters $\mathbf{m}$, the number of averages, KP , can be much smaller for a received signal with sufficient SNR.

Annex B

Regularization

Regularization was originally associated with the Thikhonov technique for filtering out the high frequencies of a solution. Today it is used in a wider sense; here we will understand it as an approach that restricts parts of the inverse solution.

It is well-known that the numerical solutions to inverse problems are often ill-conditioned. One remedy to ill-conditioning is regularization. The idea of regularization is to introduce *a priori* knowledge on the physical solution. This can be done in several ways:

- 1) Reparametrization – shape functions, see Sect. 9.
- 2) Thikhonov regularization. See Sect. B.1 below.
- 3) Weighting the likelihood function with the *a priori* distribution. See Sect. B.2.

B.1 Thikhonov regularization

Given the *a priori* estimate $\mathbf{m}_0$ of the solution it seems natural to minimize the difference between the solution $\mathbf{m}$ and the estimate $\mathbf{m}_0$

$$\phi_{\text{reg}} = \|L(\mathbf{m}_0 - \mathbf{m})\| . \quad (44)$$

Normally, L would be the identity matrix or a discrete approximation to a derivative operator. Due to different physical dimensions of the parameters, an *ad hoc* regularization could be

$$\phi_{\text{reg}} = 1/N_{\text{parm}} \sum \frac{|m_i - m_i^{\text{apri}}|^\gamma}{|m_i^{\text{max}} - m_i^{\text{min}}|^\gamma} , \quad (45)$$

where m_i^{apri} is the *a priori* expected value, $m_i^{\text{max}} - m_i^{\text{min}}$ is the search interval, and the exponent γ controls the shape of the penalization. We use $\gamma = 1$.

The objective function combining the *a priori* knowledge and the data is then

$$\phi = \phi_B + \lambda \phi_{\text{reg}} , \quad (46)$$

where λ is a Lagrange multiplier. We use $\lambda = 1$ and thus place equal emphasis on the data and the *a priori* information.

For option L we have implemented Eq. (46) with $\lambda = 1$ and $\gamma = 1$.

B.2 Including the *a priori* probability distribution

Usually, when solving inverse problems, the question is “What is the environmental model for this given data set?” This is normally an ill-posed question. A better question is “What can be inferred from the data about the environmental model given some environmental information?” Thus some *a priori* information should be included in the inverse problem. *A priori* information is always used in global inversion schemes. The model structure is selected based on *a priori* knowledge and uniform *a priori* distributions are used between the minimum and maximum bounds for the parameters.

One possibility is to include the *a priori* model in the objective function, see e.g. [23, 59]. This has the disadvantage that the distribution must be known explicitly. For Gaussian *a priori* distribution the objective function consists of two terms, one measuring the match between observed and synthetic data and the second penalizing the deviation from the *a priori* model. This approach is used in linearized inversions in order to regularize the solution.

Here a simple approach is used: the obtained likelihood function is multiplied with the *a priori* distribution, according to Eq. (5). This distribution can be arbitrary. for example a smoothed distribution obtained from inversion of other data. A simple triangular distribution is often used:

$$\rho(m^i) \propto \begin{cases} (m^i - m_u^i)/(m_m^i - m_u^i) & \text{for } m_m^i < m^i < m_u^i \\ (m^i - m_l^i)/(m_m^i - m_l^i) & \text{for } m_l^i < m^i < m_m^i \\ 0 & \text{otherwise} \end{cases} \quad (47)$$

where $m_l^i < m_u^i < m_m^i$ are the abscissa of lower bound, maximum, and upper bound of the *a priori* distribution, respectively.

A priori information is also used in the parameterization of the forward model. The choices made when doing this have a significant influence on the inverse solution, probably more than including *a priori* information for each parameter. In discretizing the environment, the physics should be carefully considered and described efficiently. Shape functions [11] is a useful method to obtain an efficient description which provide a mapping between the environmental model and the numerical forward model. This could, for example, be used to limit the search to only positive gradients in the sediment, or to obtain a more efficient description of the environment.

It is assumed that there is vanishing correlation between the *a priori* distributions of the individual parameters. Thus, $\rho(\mathbf{m}) = \rho_1(m^1)\rho_2(m^2)\dots$. Otherwise the distributions become too complicated. If parameters are correlated, the search can be limited by using a correlated *a priori* distribution. However, this seems too com-

plicated and shape functions are used instead to map a correlated model vector to another representation with a lower degree of correlation.

Annex C

Updating SAGA

It is expected that the user may wish to update the **SAGA** program to invert other types of data. This may require inclusion of other forward modeling codes into the **SAGA** package, or a change in the objective function. This appendix provides guidance on the implementation of some of these changes. The author is also willing to help in this process.

C.1 Porting a forward model into SAGA

The following steps are necessary for porting a forward code (here called **fm**) to **SAGA**:

1. In order to do inversion with a new program it should be reliable. The dimensions of the matrices should be automatically checked. It should not crash. It should have an execution time of around one second.
2. Change the forward code to a subroutine. It should have the following structure, using the same subroutine names:

```
program sagaporting
call input
call forwardinit
do i=1,10
    call forw2
enddo
end
```

3. All the input should be read in one subroutine. The variables should be transferred to a computational subroutine via common blocks that are located in one include file. (For **SAGA** it is useful if this routine only contains a few variables, so that the declared variables in **fm** does not interfere with **SAGA** variables).
4. The computational subroutine should not have any internal read/write statements. Informational write statements are useful, but these should be lumped together and surrounded by an `if (flagpu .gt. number) then` statement [**flagpu** is an integer variable in a common block that is updated at each call].

5. It should be possible to call the computational subroutine several times. Some initialization of the forward program might only be necessary once, provided the variables are stored in common blocks. These initializations should be moved to the start of the subroutine **forwardinit**. An entry call **forw2** can then be inserted below these lines for subsequent calls.
6. The output of the program should be stored to a variable **resp**. It has the following structure:

```

do i_freq=1,M_freq
  do i_depth=1,M_depth
    i_pointer=(i_depth-1+ (i_freq-1)*M_depth)*M_range
    do i_range=1,M_range
      resp(i_range+i_pointer)=press(i_range,i_depth,i_freq)
    enddo
  enddo
enddo

```

7. Check that **resp** contains precisely the same results when the forward model is called twice.

Only when the stand-alone version is working well is it time to interface it with the SAGA modules. It is much easier to work with the forward model alone, so it is important that all of the above works well before interfacing with SAGA.

The interface between SAGA and the forward model is done through two subroutines, **fm_init** that contains the **input** subroutine and the **forwardinit** subroutine, and **fm_inter** that transfer the SAGA-pointers to physical variables in the forward model.

1. Make a list of the variables that could be optimized and assign a number to each variable. During the optimization the variables are addressed through a pointer called **par2phy**. If the variable is a vector then the **index** is addressed by the one pointer **par2lay**. If the variable is a matrix it is indexed by two pointers, **par2lay** and **par3**, for the first and second index, respectively.
2. Copy one of the ***inter.f** files from one of the other forward models in **/src** to **/src/fm_inter.f** and modify this file according to the list of variables.
3. Now the forward model is ready to be ported into SAGA: copy the program in a directory below the **saga/src** directory, i.e. **/fm**.
4. Copy one of the ***init.f** files from one of the other forward models in **/src** to **/src/fm_init.f** and modify this file. The input subroutine should be called **input**. The main forward subroutine is called **forwardinit**. The entry statement is called **forw2**. Carefully change the copied file into the one used for the present forward model.
5. Update the **makefile** and compile.

C.2 Introducing a new option

It is quite easy to introduce a new option into **SAGA**. First find the subroutine **gaoptions** in file **gasub.f**. Find an unused letter and a corresponding value of the pointer **iopt**. Most new options will probably only require modification in a particular subroutine. All computations related to the objective function are done in subroutine **cost.f**.

Document Data Sheet

NATO UNCLASSIFIED

Security Classification NATO UNCLASSIFIED		Project No. 042-4
Document Serial No. SM-333	Date of Issue July 1997	Total Pages 85 pp.
Author(s) Gerstoft, P.		
Title SAGA user manual 2.0: an inversion software package		
Abstract The purpose of many experiments is to identify environmental parameters which, when used as input to a forward model, accurately model the observed data. It is also useful to have some indication of the uncertainty of the estimated parameters. SAGA is a software package that helps the user determine the best set of parameters to match a given data set. At present, the package consists of seven modules, one for each forward model. The forward codes are: OAST (wavenumber integration transmission loss model), OAST (wavenumber integration reflection coefficients model), SNAP (normal modes), SNAPRD (adiabatic normal modes), POPP (normal mode reverberation model), PROSIM (broadband adiabatic normal modes), and TPEM (tropospheric parabolic equation). Several types of observed data can be used in these inversions: single or multi-frequency pressure on a vertical array, coherent or incoherent transmission loss, reflection coefficients, reverberation data, or tropospheric electromagnetic data. For any parameter estimation problem the issue of error assessment must be addressed. In SAGA it is addressed by estimating <i>a posteriori</i> distributions. The word SAGA stems from Icelandic and simply means a long story, usually from the middle ages. It tends to go on forever. Scientifically it stands for Seismo-Acoustic inversion using Genetic Algorithms.		
Keywords genetic algorithms – inversion – optimization – SAGA		
Issuing Organization North Atlantic Treaty Organization SACLANT Undersea Research Centre Viale San Bartolomeo 400, 19138 La Spezia, Italy [From N. America: SACLANTCEN (New York) APO AE 09613]		Tel: +39 (0)187 540 111 Fax: +39 (0)187 524 600 E-mail: library@saclantc.nato.int

NATO UNCLASSIFIED

Initial Distribution for SM 333

Scientific Committee of National Representatives

SCNR Belgium	1	<i>National Liaison Officers</i>	
SCNR Canada	1		
SCNR Denmark	1	NLO Canada	1
SCNR Germany	1	NLO Denmark	1
SCNR Greece	1	NLO Germany	1
SCNR Italy	1	NLO Italy	1
SCNR Netherlands	1	NLO Netherlands	1
SCNR Norway	1	NLO Spain	1
SCNR Portugal	1	NLO UK	3
SCNR Spain	1	NLO USA	4
SCNR Turkey	1		
SCNR UK	1	Sub-total	30
SCNR USA	2	SACLANTCEN library	21
French Delegate	1		
SECGEN Rep. SCNR	1		
NAMILCOM Rep. SCNR	1	Total	51