Chapter 5

Finite-difference modeling of tensor fields

In this chapter I describe numerical algorithms based primarily on finite-differences that model wave propagation in arbitrary complex media. Spatial complexity is not restricted, and for seismic problems arbitrary stress-strain relationship can be employed. The numerical implementation is modular and written in Ratfor90 (Ratfor + Fortran90 or High Performance Fortran) such that complex methods can be easily composed by using simple building blocks. The scheme used in model experiments described in this thesis allows us , in particular, to model nonlinear source effects and wave propagation in anisotropic media. I use this algorithm for modeling seismic wave propagation, although any problem that is representable as a second order tensorial partial differential equation can be directly modeled. In particular, it allows us to look at effects introduced by the medium without having to switch methods depending on the medium type. In this manner one is sure that those observed effects are not artifacts of different types of algorithms, but originate from the medium itself. I use the popular Marmousi subsurface model to compute seismic wave propagation acoustically and elastically using an algorithm that employs identical building blocks.

Finite-difference approximations to full wave propagation problems have been investigated in the past and are used extensively when information about the complete wave field is required. In principle finite difference methods approximate the fundamental equations of the physics of wave propagation. All effects caused by the governing equations are automatically included, if used properly. One important parameter in finite difference algorithms is the way in which the medium and the wave field are discretized. If discretization is carried out correctly, the finite-difference equation represents a good approximation to the original problem. Dablain (1986) shows applications of high-order differencing to the scalar wave equation. Crucial parameters in finding a finite-difference solution to a continuous problem are the coefficients of the derivative operators. Holberg (1988) computes optimized coefficients for finite differences in such a way as to approximate the derivative operator by minimizing the least-squares error between the analytical and the approximate derivative. Other optimization criteria are possible, such as average-source spectrum weighted least-squares or an L^p norm minimization. In all those cases one tries to avoid growing error modes, while still retaining the best possible (most accurate and shortest) derivative operator. Virieux (1986) models p and sv waves in an isotropic earth model using a velocity stress formulation, that is similar to Mora (1987), who in his original inversion scheme used a conjugate gradient optimizer which applied an isotropic finite difference wave equation operator to invert seismic data in a least-squares sense. Etgen (1989) and Mikhailenko (1991) show some theoretical and numerical details of wave equation computations on grids that are more general than Cartesian grids, including arbitrary boundary conditions. I (1991) used finite differences for anisotropic imaging and material property calculation, and Cunha (1991) for isotropic reflectivity retrieval using reverse time migration.

As a programming language High Performance Fortran (F90,HPF) is practical and uses a reasonably compact notation. In particular it means that starting from the most general mathematical expression does not necessarily mean starting with the most difficult of all cases. As it turns out, the most general case can be written very simply. Reducing the space dimensionality, or reducing component dimensionality in the seismic case, is merely reflected in different implicit loop limits, while the code itself remains the same. Muir (1991a) implemented a Lattice-Boltzmann simulation in F90 with great simplicity. Ideally, I would have liked to design this modeling framework in C++, but at the time of developing this numerical tool box, either performance of C++ on parallel computers was too low for practical problems or C++ was not even available on parallel computer architectures.

I was inspired to develop such a general tool box by two things. First, Jon Claerbout's (1992)[1] programming method for his books *Processing versus Inversion* and *Applications*

[1] available on Internet: http://sepwww.stanford.edu/

Of Three Dimensional Filtering; second, the Stanford Exploration Project's *Programming Geophysics in C++* (1993)[1]. They both contain collections of processing subroutines that are used throughout the book as building blocks for larger subroutines and programs, at the same time they were small enough to be included in the text for practical illustration of numerical methods. When using those building blocks I was amazed by the versatility and easy maintainability. Secondly, I had written a several modeling and migration programs for various types of acoustic and elastic media. Those programs were very similar in structure and methodology and my aim was to consolidate those programs into one or more, that all use the same framework and building blocks. Those building blocks serve as a basis for a multitude of conceptually different modeling and migration programs. The blocks themselves can be modified and easily extended. I created a minimal set of the most functional modeling and migration building blocks that can be arranged with ease to create various numerical algorithms, from plain finite differences with exchangeable operators to spectral methods or any combination of those.

5.1 Wave equations

In the following sections I show the main features of the modeling scheme and building blocks by restricting myself to a general elastic wave equation. The modeling implementation is based on a wave equation of the abstract mathematical form

$$\underset{\sim}{\mathbf{a}}(\mathbf{x})\ \nabla\ \underset{\sim}{\mathbf{b}}(\mathbf{x})\ \nabla^T\ \mathbf{u}(\mathbf{x},t)\ -\ \frac{\partial}{\partial t^2}\mathbf{u}(\mathbf{x},t) = \mathbf{f}(\mathbf{x}_0,t) \tag{5.1}$$

$\mathbf{u}$ is an arbitrary wave field (scalar or vector); $\mathbf{f}$ is the force applied at source locations $\mathbf{x}_0$. ∇ is a general gradient operator and ∇^T the divergence operator applied to the wave field components in three dimensions. $\underset{\sim}{\mathbf{a}}$ and $\underset{\sim}{\mathbf{b}}$ are medium property fields such as density, velocity or stiffnesses. The second order equation (5.1) can be rewritten as a sequence of first order partial differential equations. Numerical calculations typically discretize those on a staggered grid for improved accuracy and simplified incorporation of explicit boundary conditions.

5.1.1 Elastic medium

For the anisotropic elastic case the differential operator ∇, defined by

$$\nabla^T = \frac{1}{2}\left(\frac{\partial}{\partial x_l} + \frac{\partial}{\partial x_k}\right) \qquad \text{with} \qquad k,l = 1,2,3 \tag{5.2}$$

operates on the displacement field $\mathbf{u}$ and results in the symmetric strain tensor $\underset{\sim}{\epsilon}$. The indices k and l, and in the following the indices i and j, range over the number of space axes and thus the number of vector components. This gives the following set – in the spatial domain – of first order equations:

$$\epsilon_{kl} = \frac{1}{2}(\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}) \tag{5.3}$$

$$\sigma_{ij} = b_{ijkl}\epsilon_{kl} \tag{5.4}$$

$$f_i = a_{ij}\frac{1}{2}(\frac{\partial}{\partial x_i} + \frac{\partial}{\partial x_j})\sigma_{ij} + \frac{\partial}{\partial t^2}u_j \tag{5.5}$$

where $\underset{\sim}{\mathbf{a}}$ is the density (usually a scalar $a \times$ identity I) and $\underset{\sim}{\mathbf{b}}$ represents the stiffness matrix of elastic moduli. In general the stiffness matrix is a sparse matrix, and can be simplified for different degrees of symmetry. Equation (5.4) represents the stress-strain relationship in the medium and the two second-order tensors $\underset{\sim}{\epsilon}$ and $\underset{\sim}{\sigma}$ are related via the moduli's fourth-order tensor $\underset{\sim}{\mathbf{b}}$. In equation (5.5) the stress field is converted into an acceleration field and a forcing function $\mathbf{f}$ is added. I use this set of first order equations in my implementation to have flexibility in changing parameters and boundary conditions.

5.1.2 Acoustic medium

In the special case of an acoustic medium the stiffness matrix and the stress tensor reduce to scalar values (particle velocity and pressure). One can derive the familiar set of first order equations, relating changes in pressure to changes in velocity:

$$\rho\frac{\partial}{\partial t}u_i = -\frac{\partial}{\partial x_i}p \tag{5.6}$$

$$\frac{\partial}{\partial t}p = K\frac{\partial}{\partial x_k}u_k \tag{5.7}$$

Combining those two equations into a second order partial differential equation and incorporating the source term q gives

$$K(\mathbf{x})\ \nabla\ \rho(\mathbf{x})\ \nabla^T\ p(\mathbf{x},t)\ -\ \frac{\partial}{\partial t^2}p(\mathbf{x},t) = q(\mathbf{x}_0,t) \tag{5.8}$$

This corresponds to the general form (5.1), where K is the bulk modulus and ρ is the density. All these quantities are scalar and can be regarded as tensors of rank one.

It is obvious from the previous two examples that the equation structure is identical for different types of media. One can either treat the acoustic case as a completely self-contained problem or as an elastic problem where the material properties happen to be acoustic. Differences in spatial dimensionality or component dimensionality are reflected in the rank of the tensors and thus the bounds of the subscripts. From that I conclude that if the problem structure is identical the same numerical tools can be reused for different modeling purposes. In practical terms it is merely a matter of keeping track of the indices and making sure those indices are efficiently mapped in each case. Thus all the building blocks can reuse the same pieces of code if they are written in a general form.

5.2 Mathematical objects and spaces

The variables in a mathematical formula are usually represented in numerical applications as samples over a discrete field. Such variables are in reality tensorial quantities, which means they have a metric attached to them. On this metric different functions, such as basic addition, subtraction, multiplication, and division are defined. Besides those basic functions other operations such as taking a derivative or a taking an integral are given. Precisely the latter two operations should ideally be defined with the object itself, since then the object is self-contained and can be reused easily in various realizations. If, for example, the wave field is given on a curved general grid, suitable differentiating and integrating functions should be defined with it. This is very easily possible in a C++ implementation, but much harder to do in a Fortran implementation of the numerical algorithm. Thus I restrict myself to a regularly connected sampled grid, that is not completely general. All my differentiating or integrating function implementations assume such a regularly connected grid. The grid layout itself is outlined in 5.1 for the two-dimensional representation. The complete grid consists of several sub grids. In the interior part I calculate the wave field quantities as accurately as possible. Surrounding this portion of the grid are zones with material properties and wave equations that differ from the interior. Those properties can be adjusted in order to minimize artificial reflections from the boundaries of the interior grid.

Following is a list of basic object types that are used throughout the algorithm. The name in capital letters designates the object type, the argument is the variable that will receive those properties.

- `WFIELD(uu)` designates the variable uu as wave field. If it denotes a displacement, it has a certain number of vector components associated with it; if it denotes pressure it is a scalar field.

- `TENSOR(uu)` designates the variable uu as a second rank tensor. Thus it specifies stress or strain in the elastic medium.

- `STIFF(uu)` designates the variable uu as a fourth rank tensor and it denotes the elastic stiffness tensor.

- `BOUND(uu)` designates a masking object that defines the computational domain by specifying grid types, such as interior and exterior orientations.

Practically each of these objects is a macro that is expanded during compilation of the program. If a change to the object type definitons is centrally made, automatically all programs will receive the correct type. The major drawback in F90, unlike in C++, is the absence of functions and subroutines that are tied to those objects. I simulate them by defining all objects on the same computational field, such that operations like derivatives, convolutions or time updates always operate on the same sampled grid type.

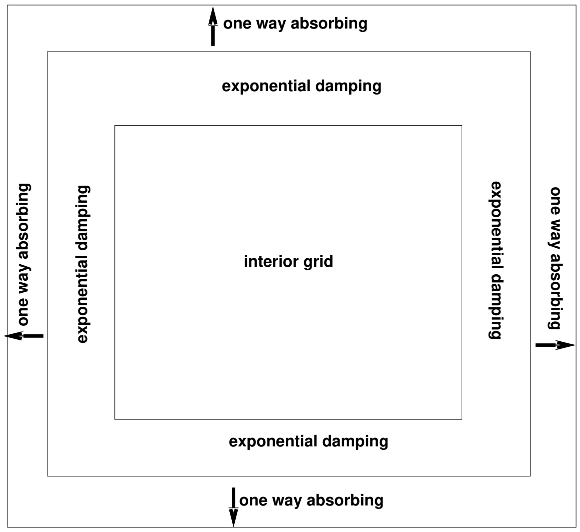


FIG. 5.1. Computational grid for the finite-difference modeling calculation. The main computational interior grid is surrounded by two boundary zones, whose function is to suppress unwanted reflection from the boundaries of the interior grid. genmod-computational [NR]

For efficient computation medium types are put into categories: acoustic, isotropic, cubic, transverse isotropic (TI), orthorombic, monoclinic and triclinic. These denote material property symmetries as described in detail in Auld (1973). Symmetries not only have a profound influence on seismic wave propagation, but also on the efficiency of the numerical algorithm. The internal representation in the program follows exactly the mathematical representation. Take for example the elastic stiffness matrix $\underset{\sim}{\mathbf{b}}$; its mathematical matrix notation for an isotropic solid is:

$$\begin{pmatrix} b_{11} & b_{12} & b_{12} & & & \\ b_{12} & b_{11} & b_{12} & & & \\ b_{12} & b_{12} & b_{11} & & & \\ & & & b_{33} & & \\ & & & & b_{33} & \\ & & & & & b_{33} \end{pmatrix}, \tag{5.9}$$

where b_{ij} are the elastic moduli. The mathematical form translates to a simple mask function, as can be seen in the code `getstress()` (page 63). For example, the isotropic elastic symmetry mask is as follows:

```
stiffmask(1,1)=1; stiffmask(1,2)=1; stiffmask(1,3)=1;
stiffmask(2,1)=1; stiffmask(2,2)=1; stiffmask(2,3)=1;
stiffmask(3,1)=1; stiffmask(3,2)=1; stiffmask(3,3)=1;
                                         stiffmask(4,4)=1;
                                                   stiffmask(5,5)=1;
                                                      stiffmask(6,6)=1;
```

Similarly, in the acoustic case, where the material parameter symmetry degenerates to

$$\begin{pmatrix} b_{11} & & \\ & b_{11} & \\ & & b_{11} \end{pmatrix} \tag{5.10}$$

it translates directly to

```
stiffmask(1,1)=1;
       stiffmask(2,2)=1;
                 stiffmask(3,3)=1;.
```

The mask is merely a scaled identity matrix. For many realistic media, the stiffness matrix is a very sparse matrix with only a few non-zero elements. Furthermore, symmetries force some of the components' values to coincide with each other. That is fortunate for numerical evaluations, since then not all components need to be actually stored in the computer's memory. In this implementation I store only nonzero elements that are different from each other; other nonzero elements are deduced by symmetry masks.

5.3 Modeling structure

The outline of a modeling algorithm for the general wave equation (5.1) can be written very concisely.

```
loop over experiments {

    apply displacement boundary conditions
    calculate strain
    apply strain boundary conditions
    calculate stress
    apply stress boundary conditions
    calculate divergence of stress
    apply acceleration boundary conditions
    time update the wave field
    record wave field

}
```

Items that are marked with boxes are functional modules that operate on a set of field variables. These boxes can be easily interchanged in a "plug and play" manner, such that a number of different numerical methods can be applied easily. The calculation of those highlighted functions are not tied to a specific implementation, they rather represent conceptual steps in computing field quantities. The code itself reflects closely the mathematical descriptions.

Calculating derivatives

When numerically implementing differential equations, some approximation to the continuous derivative needs to be found. There are several ways in which this discretization can be achieved. In my algorithm I use the well-known method of finite-difference approximation,

$$\frac{\partial}{\partial x}u(\mathbf{x}_k + \frac{1}{2}\Delta h) = \frac{1}{\Delta h}\sum_{i=k-n}^{k+n} b_k\, u_k \tag{5.11}$$

where the grid point k corresponds to the location x_k. The number of coefficients n and the actual values of b_k determine how well such a convolutional operator can approximate the true derivative at the point $x_0 + \frac{1}{2}\Delta h$. Normally, the derivative value is required on the location x_0, but when using a staggered grid, it is needed half-way between grid points. Additionally, first-order derivatives are computed more accurately between original grid locations, as the top plot in Figure 5.2 shows. The top plot shows the centered three point finite-difference stencil $-1, 0, 1$ on the left; on the right, the corresponding amplitude spectrum is depicted. In all the plots the stencils are normalized to unit amplitude in the spatial domain, while in the frequency domain amplitude and frequency axis are normalized to π. A comparison between the desired spectral behavior – a straight line going through the origin and π – and the numerical approximation is then easily possible. The centered stencil shows the worst spectral behavior, since it only approximates the extremely low frequency spectrum. It peaks at half Nyquist frequency and decays down to zero amplitude for all frequencies above. The spectral behavior of the staggered two point stencil is better, since it approximates a longer range of the low frequency spectrum. The next stencil is already 8 grid points in length and is derived from Central Limit Theorem arguments, as described in detail in the appendix to this chapter, such that repeated self-convolution, tends to a stable self-similar operator shape. Its spectrum approaches the theoretically desired curve for a wide range and deviates above 70 percent of Nyquist frequency. The bottom picture in the comparison depicts coefficients computed by using an optimization algorithm that finds unknown stencil coefficients by minimizing the residual differences between the approximated and desired spectrum. The operator is 16 grid points in total length. The optimization assumes that all frequencies in the spectrum are equally important and does not try to weigh the optimization's cost function according to frequency content of the source and the medium. However, such a weighting could readily be done and an even more optimal set of coefficients be derived for a particular source and medium configuration. If one would increase the operator length, one could get better matches between the desired and approximated spectrum. In the limit the stencil is of the size of the total number of grid points and approaches that of a derivative computed by Fourier Transforms (which is the desired straight line in all the comparison plots). In practical terms, there is trade-off between accuracy and operator length and an optimal balance has to be found depending on the particular wave propagation problem to be solved.

The subroutine `conv1dc()` (page 59) implements such a derivative calculation via a general convolution routine that acts on any one dimension. It can be used for convolving over any dimension in space and is the heart of the finite-difference algorithm. Figure 5.3 shows the effect of operator choice on seismic wave propagation. The figure contains a comparison of the three staggered operators in Figure 5.2. The snapshots show that the accuracy of propagating a wave front in a homogeneous medium improves with operator length and an optimum choice of coefficients. The inaccuracy of the derivative operator at high frequencies causes dispersion and introduces an angle-dependent and overall error in first breaks of the snapshots. The solid circle traces the analytically correct position of the wavefront for that time when the snapshot was taken. The spectrum of the wave field is such that for the short operator the inaccuracies dominate, while for the longer operators the spectrum is treated properly. All cases lie within the stability region of the finite-difference propagator and the maximum wavelength generated by the source signal is at the limit of about three grid points per spatial wave length.

```
# conv1dc: convolution 1 dimensional circular boundary condition
# u and w are conjugate to each other
# axis: axis over which to convolve
# stagger:  0 normal conv; 1 shift by one grid point
#%
subroutine conv1dc(conj,u,t,k,w,sidx,  axis, scale,bnd)
implicit none
#include "commons"
integer           conj,  t,k,  sidx
integer                                axis
real                                         scale
WFIELD(u)
WFIELDLAY(u)
TENSOR(w)
TENSORLAY(w)
BOUND(bnd)
BOUNDLAY(bnd)
integer ishift

if (conj==0) {                         #  the "gradient"
do ishift = nlo-stagger, nhi-stagger {
   w(sidx,:,:,:) = w(sidx,:,:,:) +
        coef(ishift+stagger)*cshift(u(t,k,:,:,:),dim=axis,shift=ishift) * scale
   }
}
else{
do ishift = nlo-stagger, nhi-stagger {  #  the "divergence"
   u(t,k,:,:,:) = u(t,k,:,:,:) +
       coef(ishift+stagger)*cshift(w(sidx,:,:,:),dim=axis,shift=ishift) * scale
   }
}
return
end
```

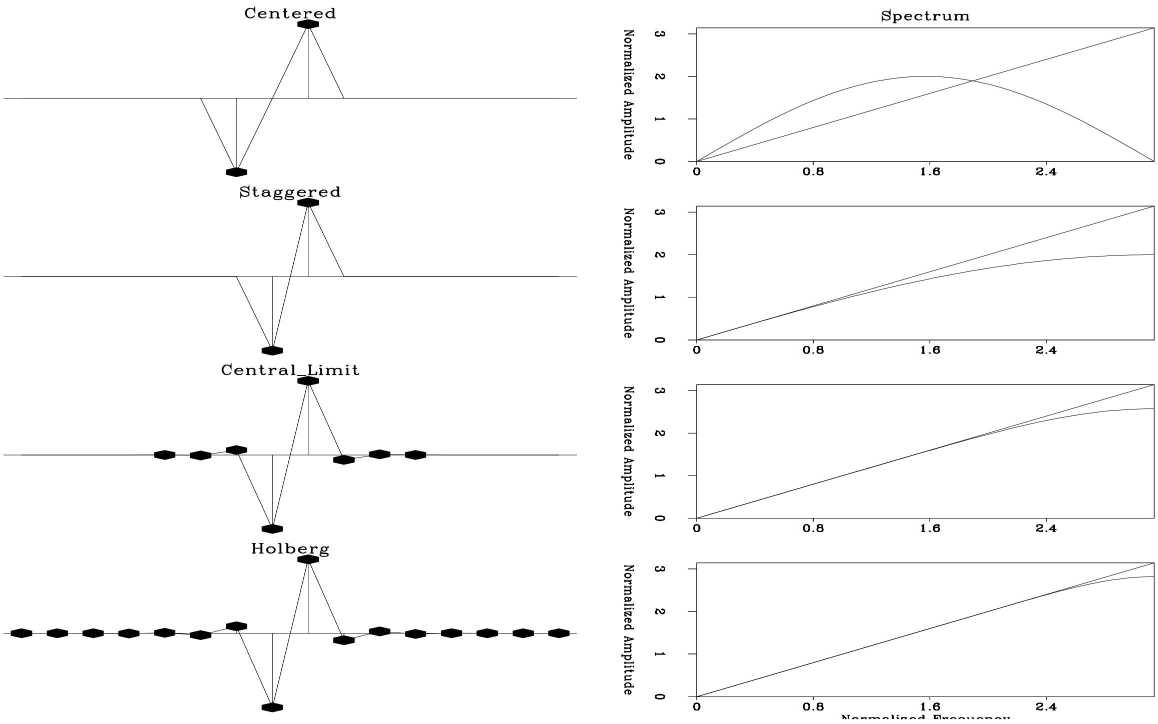


FIG. 5.2. Comparison between different convolutional operators. All stencils try to approximate the analytical spectral behavior of a first-order derivative. Spectral and spatial diagrams are normalized; the ideal spectral response is a straight line. genmod-derivatives [R]

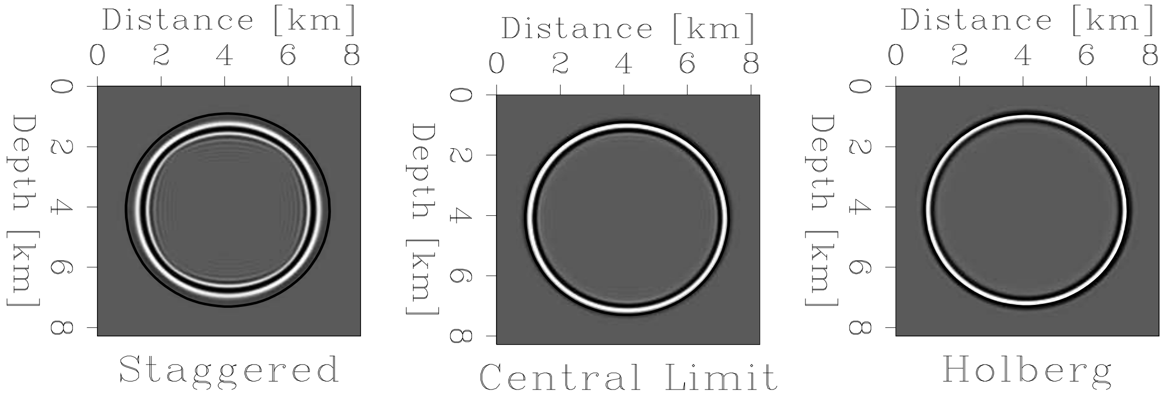


FIG. 5.3. Comparison between different snapshots, computed using the stencils given in Figure 5.2. The analytically computed wavefront in this homogeneous medium is overlaid on top of the wave field. The staggered two-point filter shows dispersion as well as reduced propagation velocity since the spectrum of the wave field is not adequately treated by the difference stencil. In contrast, the central limit and the optimized stencil give similar results: the peaks of the wave field lie on top of the theoretical curve. genmod-waves2D [CR]

Strain calculation

Subroutine `del()` (page 62) calculates the strain tensor given in equation (5.3) and represented as a matrix operation in equation (5.13). I deviate here from the mathematical notation (5.3) and use abbreviated indices for speed and efficiency. Symmetries in tensor components allow indices to be abbreviated by mapping a 3×3 double index ij to a single index I (Auld, 1973). The abbreviation rule is

$$11 \rightarrow 1\ ;\ 22 \rightarrow 2\ ;\ 33 \rightarrow 3\ ;\ 23 \rightarrow 4\ ;\ 13 \rightarrow 5\ ;\ 11 \rightarrow 6 \tag{5.12}$$

Thus the tensor product can be written as regular 3×6 matrix multiplication:

$$\underset{\sim}{\epsilon} = \mathbf{K}^T\, \mathbf{u} \qquad = \qquad \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \end{pmatrix} = \begin{pmatrix} \frac{\partial}{\partial x_1} & & \\ & \frac{\partial}{\partial x_2} & \\ & & \frac{\partial}{\partial x_3} \\ & \frac{\partial}{\partial x_3} & \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} & & \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} & \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \tag{5.13}$$

The strain components are calculated on staggered grid, such that their location coincides with the location of the stiffness component, with which it interacts in the next calculation step. The parameter `stagger` triggers the shifting of the evaluation point. If `conj==0`, then subroutine `del()` (page 62) will evaluate ∇^T in (5.1), performing an outer product; otherwise it calculates an inner product. Displacement uu and strain ss are in that sense conjugate to each other. The actual derivative calculation is carried out using the previously introduced subroutine `conv1dc()` (page 59), which is a finite-difference approximation to a partial derivative. Instead of using a convolution for derivative computation, I could take a Fourier derivative, then I would perform spatial pseudo-spectral modeling. One could even think of alternating between the two methods depending on which spatial derivative is calculated.

```
# apply symmetric Lagrangian derivative
#
subroutine del (conj,uu,t,ss,bnd)
implicit none
#include "commons"
integer         conj,   t
WFIELD(uu)
WFIELDLAY(uu)
TENSOR(ss)
TENSORLAY(ss)
BOUND(bnd)
BOUNDLAY(bnd)
```

```
integer ivec,iten,direction
real scale

if(conj==0) {ss = 0.;        }
else        {uu(t,:,:,:,:)=0.}

do ivec=1,ncomp {
do iten=1,ntcomp{    direction=trudim(deriv(tcomp(iten),icomp(ivec)))
 if (direction>0){   scale = scaling(tcomp(iten),icomp(ivec))
  if (conj==0)  { stagger = stagc0(tcomp(iten),icomp(ivec)) }
  else          { stagger = stagc1(tcomp(iten),icomp(ivec)) }
    call conv1dc(conj,uu,t,ivec,ss,iten,direction,ddx(direction)*scale,bnd)
  }
}}

return
end
```

Stress calculation

Following the general algorithm on page 57, the next step is a stress calculation shown in subroutine `getstress()` (page 63). It performs the double inner product of a stiffness matrix with a strain tensor that results in the stress tensor. Again I use the matrix form of equation (5.4), which can be written as

$$\begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{pmatrix} = \begin{pmatrix} c_{11} & & \cdots & & c_{16} \\ & & & & \\ \vdots & & & & \\ & & \ddots & & \\ & & & & \\ c_{61} & \cdots & & & c_{66} \end{pmatrix} \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \end{pmatrix} \tag{5.14}$$

Since the stiffness matrix is generally sparse I use and store only nonzero elements, as indicated by the index `k` in the subroutine, for the calculation of stress components. This general routine can be used for many types of calculations as long as the operation can be cast as a tensor inner product.

```
# getstress:    Hooke's Law
# get stresses from strain
#%
subroutine getstress(strain,stiff,stress,bnd)
implicit none
#include "commons"
TENSOR(strain)
TENSORLAY(strain)
TENSOR(stress)
TENSORLAY(stress)
STIFF(stiff)
STIFFLAY(stiff)
BOUND(bnd)
BOUNDLAY(bnd)
```

```
integer i,j,k,l

stress=0.
do i=1,ntcomp {
do j=1,ntcomp {              k = ccin(tcomp(i),tcomp(j))
   if (k!=0)  {
      stress(i,:,:,:) = stiff(k,:,:,:) * stiffmask(tcomp(i),tcomp(j))*
                        strain(j,:,:,:) + stress(i,:,:,:)
      }
}}
return
end
```

Acceleration calculation

After the stress calculation, the next step computes acceleration components by applying the tensor divergence operator to the stress field. Since the divergence operation ∇ is merely the conjugate of ∇^T, I am reusing subroutine `del()` (page 62) with `conj==1`. This calculates the acceleration components from the stress tensor by performing an inner product of the ∇ operator with the stress $\underset{\sim}{\sigma}$ and multiplying it with the inverse density $\underset{\sim}{\rho}^{-1}$ to obtain the second time derivative of $\mathbf{u}$.

The wave operator

For convenience, the right-hand side of equation (5.1) is often combined into a single operator L:

$$L = \underset{\sim}{\mathbf{a}}(\mathbf{x})\ \nabla\ \underset{\sim}{\mathbf{b}}(\mathbf{x})\ \nabla^T\ . \tag{5.15}$$

```
# apply the simple wave equation operator without src term
#
subroutine wave(u,strain,stiff,stress,rho3,bnd)
implicit none
include "commons"
WFIELD(u)
TENSOR(strain)
TENSOR(stress)
STIFF(stiff)
MODEL(rho3)
BOUND(bnd)

call del(0,u,3,strain,bnd)

call getstress(strain,stiff,stress,bnd)

call del(1,u,3,stress,bnd)

u(3,:,:,:,:) = u(3,:,:,:,:) * rho3

return
end
```

In the implementation one can compose the wave operator L in a short subroutine `wave()` (page 64) when the boundary conditions are neglected. This newly created operator simplifies calculations of complicated time updates of the equation (5.1) while still allowing a compact notation.

Time update

Once the acceleration at the current time is computed, the displacement at the next time step has to be calculated. This operation is independent of the spatial derivative calculations and one is left with many choices. For efficiency and accuracy I am exclusively employing explicit time update methods; pro and cons are discussed in (Emerman et al., 1982), where explicit schemes are recommended for elastic wave propagation. The most common approach is to use a Taylor expansion in the neighborhood of the current time step:

$$\frac{\partial^2}{\partial t^2}\mathbf{u} = \frac{1}{\Delta t}(\mathbf{u}(t+\Delta t) - 2\mathbf{u}(t) + \mathbf{u}(t-\Delta t)) \approx \frac{2}{\Delta t^2}\sum_{k=1}^{n}\frac{\Delta t^{2k}}{(2k)!}\left(\frac{\partial^{2k}}{\partial t^{2k}}\mathbf{u}\right) \quad n = 1, 2, 3, ... \tag{5.16}$$

For the time update one can choose upto what order in time to carry out the update. The straight-forward approach uses a Taylor expansion in the neighborhood of the current time step to calculate wave field values at a future time. The simplest update is a 2nd order update, which is used in the subroutine `update2()` (page 65).

```
# update2
# second order time update
#%
subroutine update2( dt2, u , bnd )
#include "commons"
real                dt2
WFIELD(u)
WFIELDLAY(u)
BOUND(bnd)
BOUNDLAY(bnd)

do ic = 1, ncomp {
  where(bnd==INTERIOR .or. bnd==FREE)
  u(3,ic,:,:,:)=2.*u(2,ic,:,:,:)-u(1,ic,:,:,:)+dt2*u(3,ic,:,:,:)
  u(1,ic,:,:,:)=u(2,ic,:,:,:)
  u(2,ic,:,:,:)=u(3,ic,:,:,:)
  endwhere
}

return
end
```

If one wants to calculate to higher orders of time one can use the general version of the time update of order n, which applies the wave operator L repeatedly to the wave field, such that successive partial derivatives get added into the time update. The update equation (5.16) can be expressed in terms of spatial derivative rather than temporal derivatives by realizing that $-L = \frac{\partial^2}{\partial t^2}\mathbf{u}$. Alternatively, one can employ a Chebychev time update that similarly calculates $L\mathbf{u}$, and that uses coefficients of a Chebychev Polynomials, instead of Taylor coefficients. Time updates thus can be achieved recursively to high accuracy, in such a manner that the choice of time step becomes less critical and one is free to choose parameters appropiate to the problem, not the particular numerical algorithm.

5.4 Examples

The outlined modeling program lends itself easily to comparisons between different types of modeling, for example acoustic versus elastic, or isotropic versus anisotropic modeling. Since the same pieces of code are used, one can be sure that differences in the modeling results are most likely produced by medium properties themselves and not by artifacts introduced by a difference in the modeling algorithms.

5.4.1 Marmousi velocity model

The Marmousi data has in recent years been used as a benchmark to compare different acoustic imaging and velocity estimation algorithms (Versteeg and Grau, 1990). Seismic data together with the velocity and density models are available from the Institute Francais du Petrole. As an example, I use the Marmousi subsurface model, shown in Figure 5.4 with its original pressure wave velocity and density. Since the original subsurface model lacks the specification of a shear wave velocity model for the elastic modeling, I created shear wave velocities assuming a constant Possion's ratio. Thus, P-and S-wave velocities in the following examples are strongly correlated. The elastic and acoustic seismogram modeling results are shown in Figures 5.6 and 5.7. In both cases pressure sources were used and horizontal and vertical displacement components were recorded on the surface. The left side of the figures show the acoustically modeled seismograms, while the right side of the figure shows the difference between elastic and acoustic modeling. Note that the figures are plotted to the same scale. The converted energy in the elastic wave field is nearly as large as the acoustic primary and reflected energy. The elastically converted waves are mainly visible in conversions at sharp material interfaces. The absolute first P-wave arrival time is not changed at all: the amplitude is simply reduced because of energy leaking into converted modes.

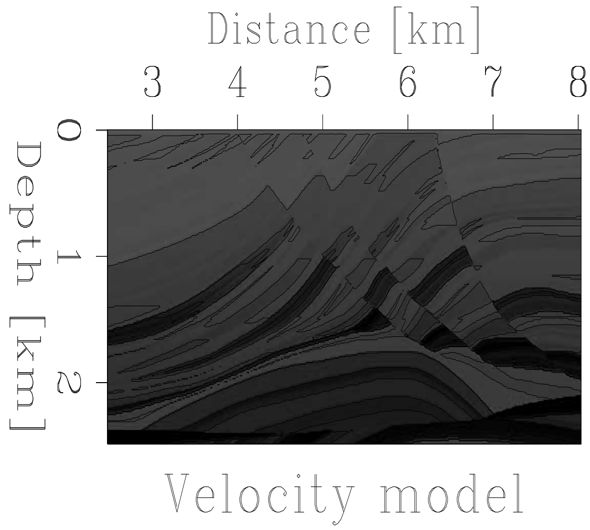


FIG. 5.4. A portion of the Marmousi subsurface velocity model that is used to model seismic wave propagation. genmod-marm [CR]

Figure 5.5 compares the divergence of both types of modeling, acoustic and elastic. It shows the fully computed wave field cube in time and space. The front cube is a snapshot of the wave field at 0.810 seconds. The source was close to the surface at horizontal distance of about 3 km. The top face shows the seismogram that would have been recorded if receivers had been placed at the surface of the model. Originating from the source A cone that originates from the source, extends in time away from the source. The delimiting energy corridor is the direct and surface wave energy, while at later times the reflection from the interfaces are recorded up to a maximum recoding time of 3 seconds. The side panel displays the wave field that would have been recorded with receivers in a bore hole traversing the formation at a horizontal distance of 3.6 km, extending from the surface down to the maximum depth of the model. Mode conversions can be seen on all three faces of the elastic wave field data cube.

The divergence is computed from the wave field components using an identical procedure that allows one to compare the P-wave types more easily. The wave field shows that, as expected, pure acoustic modeling cannot account for most of the mode-converted waves. However, a realistic modeling of the earth should include mode-converted waves (p, sv and sh). Since mode conversions are absent in the pure acoustic modeling, the amplitudes of even the primary waves are different. No energy leaks into other modes, but rather remains within the P-wave mode. The most noteworthy event is the strong amplitude of a mode-converted reflection from the top free surface. A non-converted reflection is hardly visible on the acoustic modeling plot, but the free surface boundary conditions clearly cause a strong conversion. Internal mode conversions are less prominent, but are still present.

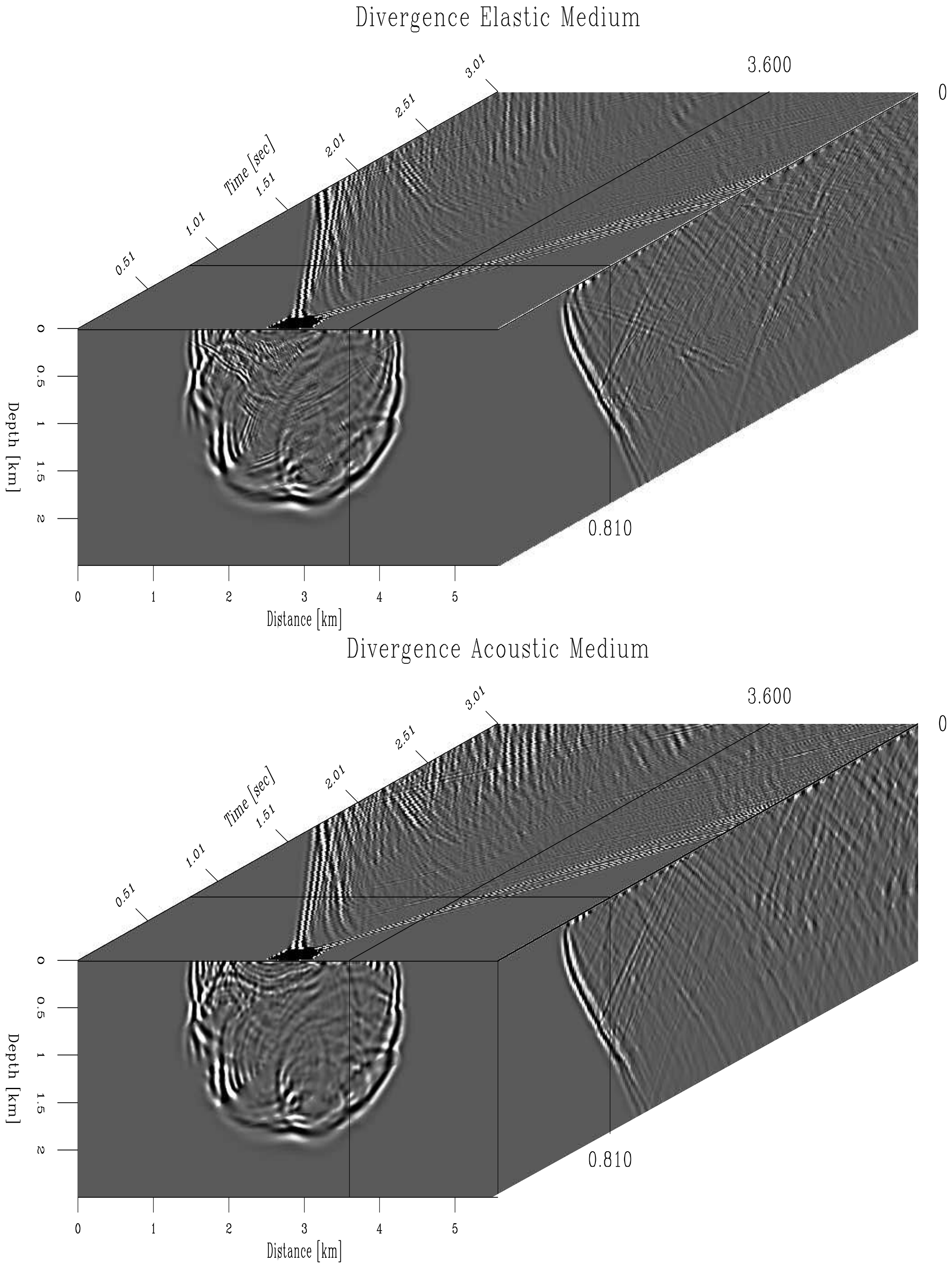


FIG. 5.5. Divergence computed from acoustic and elastic modeling schemes. The acoustic modeling does not show mode-converted waves, while the elastic one does show conversions from P to S waves. Energy converts at the free surface resulting in a secondary wavefront propagating downward. But mode conversions are occuring also at sharp interfaces with large contrast. $\boxed{\text{genmod-cubes}}$ [CR]

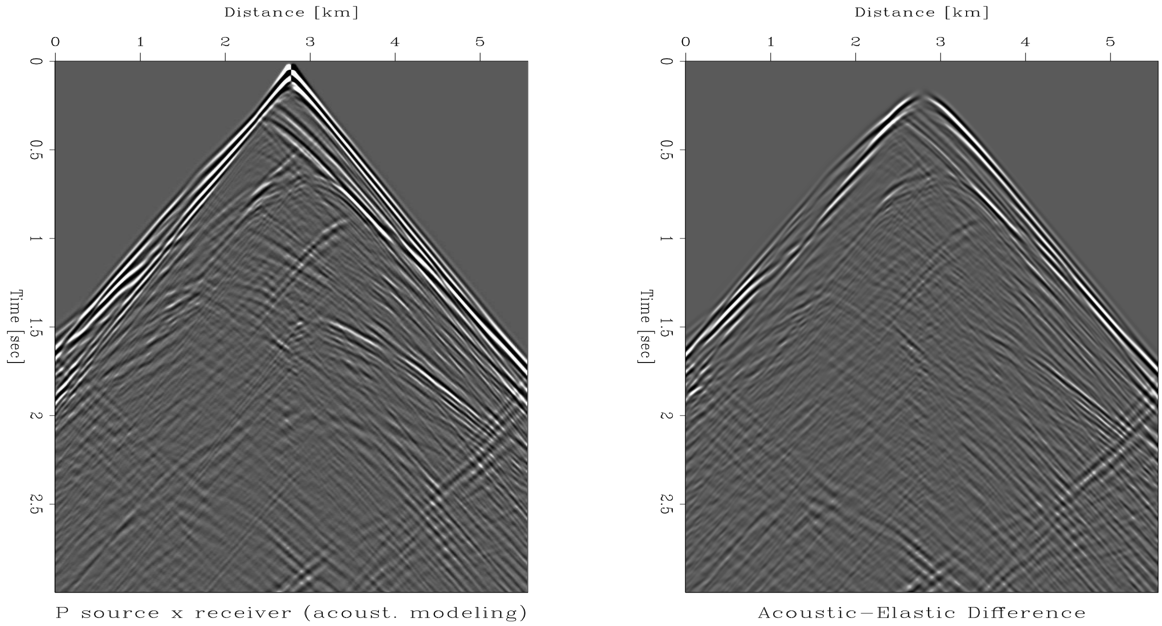


FIG. 5.6. The left plot shows an acoustically modeled seismogram (p source into x receiver): the right plot shows the difference between acoustic and elastic modeling clipped to the same scale. genmod-compx [CR]

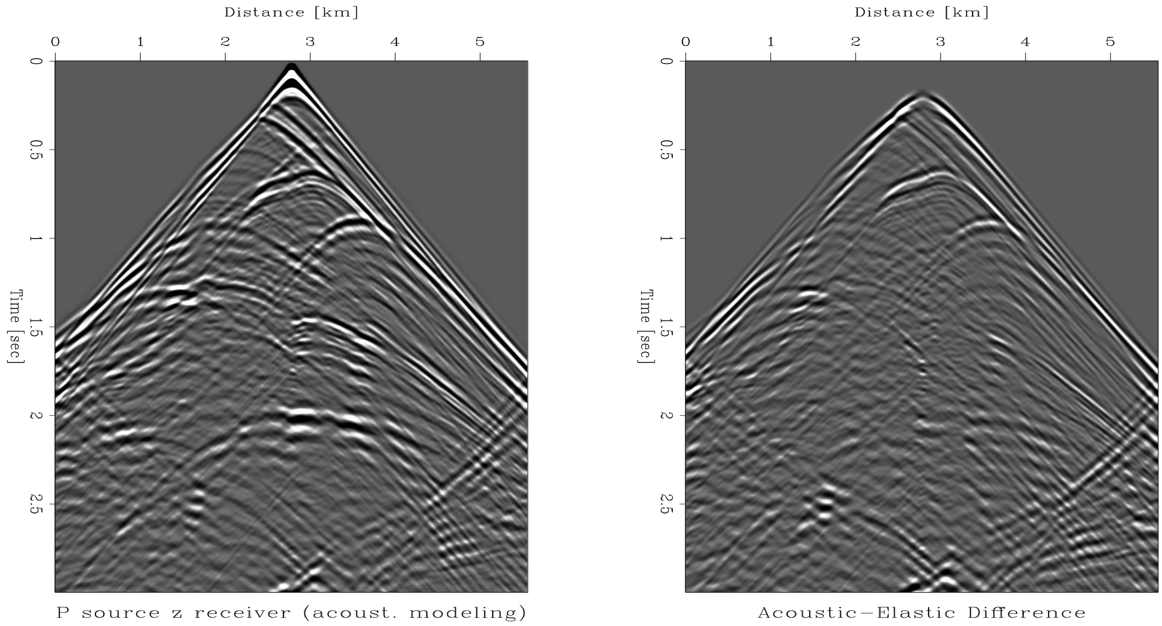


FIG. 5.7. The left plot shows an acoustically modeled seismogram (p source into z receiver), the right plot shows the difference between acoustic and elastic modeling clipped to the same scale. genmod-compz [CR]

5.4.2 Summary

By expressing a second order wave equation in terms of mathematical tensor operators, I have accumulated a tool box for acoustic and elastic 2D and 3D modeling and migration algorithms. These building blocks can be put together in many different variations and the existing set can easily be expanded. Only one version of the basic source code has to be maintained. The computational speed is slower than for special purpose programs, but those modules are easy to understand, small enough to be included into scientific papers and, most important, their code is not redundant. I see this as a big improvement over monolithic algorithms and source codes and it helps to demonstrate concepts and their numerical implementation in manageable, comprehensible pieces.

5.5 Accuracy of reflection coefficients

The question if finite-difference modeling is adequate for amplitude versus offset analysis (AVO) has long been a subject of discussion. Obviously, if temporal and spatial discretization is small, it can approximate continuous derivatives well, and finite-differences give the correct result. However, the question remains: does it give correct responses for discretizations that are desirable and commonly used? i.e., as large time steps that are desired for usual seismogram discretizations and as large spatial steps that can be tolerated by the model description. A definite answer cannot be given directly, because it depends on several factors: how a staggered grid is employed, how boundary conditions are treated, and what kind of time update is used.

I chose to test whether FD modeling of reflection amplitudes is adequate to allow AVO analyses by comparing it to an analytical solution, the Zoeppritz plane wave energy equations (1919, 1985) for a single reflector. Based on these findings, I used this FD modeling algorithm to generate synthetic datasets for a number of anisotropic media that had been analyzed for SEG's 1992 AVO workshop (Karrenbach et al., 1992).

Cunha (1992a) described a modeling scheme which takes discontinuities in medium properties into account by applying different FD operators to the wave field and the medium on a non-staggered grid. Different implementations of a staggered grid and derivatives can influence the accuracy of reflection responses. In this thesis, I am not employing this method, but rather use cascaded derivative operators on a staggered grid. The test medium consists of two completely homogeneous blocks. The source is located at some distance from the interface and receivers are parallel to the interface. A P wave source generates an incident wave field that is reflected at the interface and recorded at the receivers. Figure 5.8 shows the theoretically expected amplitude responses for the reflection experiment in Figure 5.9, plotted versus the horizontal slowness. For extracting the modeled amplitudes out of the synthetic dataset I use the following scheme. Since Zoeppritz equations give plane wave amplitude ratios versus a certain emergence angle, I convert from the point source experiment to a plane wave representation. I apply a slant-stack inversion procedure to find the best slant-stack representation of the modeled CMP gather, which reduces artifacts in the $\tau - p$ domain caused by the limited aperture. Secondly, I find the incident amplitude without any reflecting segment. To achieve this I carry out a dual experiment by placing receivers at the mirror image location in depth in the medium. The dual experiment, Figure 5.9, estimates all propagation effects but omits reflection at the boundary. For that reason the two signals have different phases. Since the medium properties are known, I can identify the reflection response and extract the appropriate signal for each horizontal slowness. Performing a Fourier Transform on that signal results in a certain amplitude for the dominant frequency component. The ratio of incident versus reflected amplitude is then compared to the analytical solution of the Zoeppritz equations. As can be seen from Figure 5.11, the reflection amplitudes produced by the modeling scheme are in nearly perfect agreement with analytical solutions to the reflection amplitudes up to the critical angle. Figures 5.8 and 5.10 show the analytically calculated responses in comparison to the modeled extracted responses. Such a test

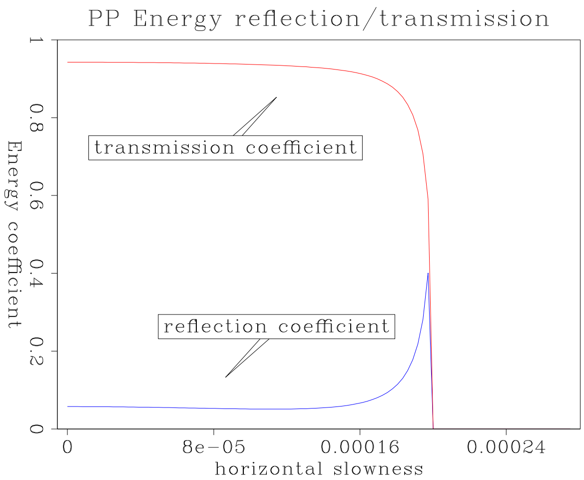


FIG. 5.8. Analytically calculated energy reflection and transmission coefficients from Zoeppritz equations. genmod-iso1.ref [CR]

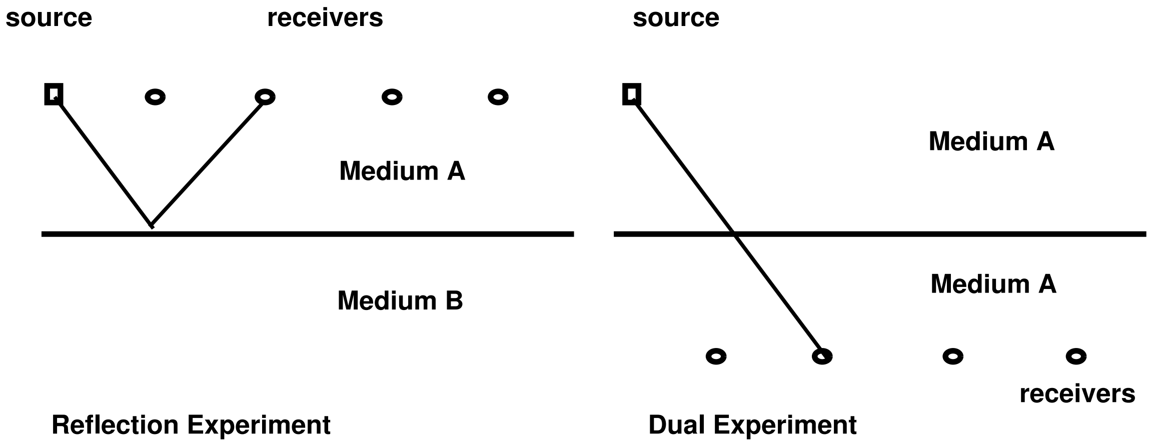


FIG. 5.9. The reflection experiment records reflected energy from the interface. The dual experiment determines the energy that would have been recorded if the interface had not been present. This double experiment can be used to measure reflected as well as incident energy uniquely. genmod-dual [NR]

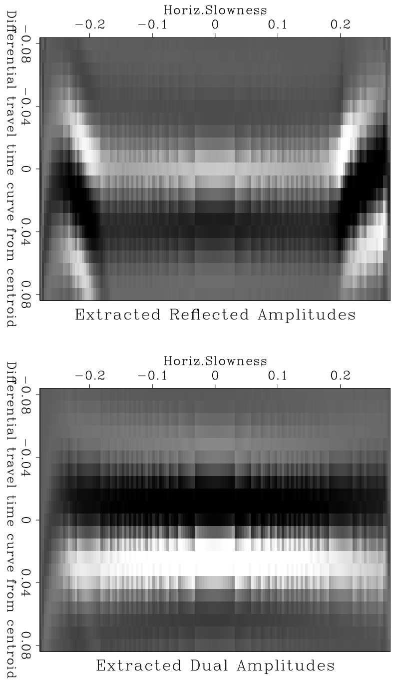


FIG. 5.10. Extracted amplitudes from the FD modeling experiment and the dual experiment. The dual experiment estimates all propagation effects but omits reflection at the boundary. For that reason the two signals have different phases. genmod-extraiso [CR]

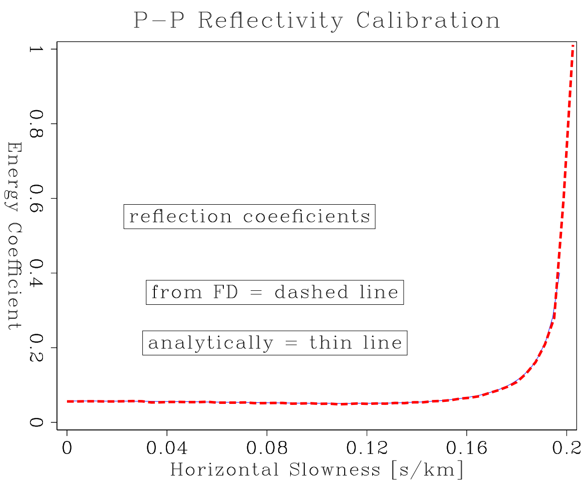


FIG. 5.11. PP calibration curve for the FD modeling response. Reflection amplitudes were extracted using the dual experiment. Zoeppritz plane-wave energy reflection coefficients are calculated analytically. genmod-ppcalibrate [CR]

is clearly necessary if one wants to make sure the modeling algorithm produces correct reflection coefficients at medium interfaces. The above test gives one confidence in the modeling algorithm for this type of blocky model.

5.6 Splitting the wave operator

When one applies the elastic wave equation operator for spatially varying media, several first order derivatives have to be calculated in the process. The wave equation can formally be split into two components so that derivatives are taken with respect to medium parameters and with respect to the propagating wave field. Splitting the wave equation operator allows one to adapt derivative operators to the physical quantities to be differentiated. In particular the adaption can be guided by special properties of the quantity. Practical problems can arise since the derivated quantities in general complex media have to be interpolated back to collocation points.

In seismic modeling adaptive techniques are well known. Usually the adaption of a quantity is in the direction of a physical dimension such as time or space. In general, adaption is guided by criteria to improve numerical accuracy and stability or to increase the speed of the calculation. Cunha (1992b) discussed the notion of treating the propagating isotropic elastic wave field differently from the underlying medium. For anisotropic media, Igel et. al (1992) showed an interpolation error analysis of FD operators and their spatial distribution in the case of a staggered grid. In this section I combine both notions for anisotropic media.

5.6.1 Formal splitting of the wave operator

The general wave equation (5.1) involves calculating tensor products and derivatives, where the wave operator **L**, introduced in a previous section, is defined as

$$\mathbf{Lu} = \underset{\sim}{\mathbf{a}}\nabla(\underset{\sim}{\mathbf{b}}\underset{\sim}{\epsilon}) \; . \tag{5.17}$$

∇ is a partial derivative operator matrix (5.2) acting on its argument, $\underset{\sim}{\mathbf{b}}$ is a medium property tensor and **u** is the displacement field. Using the chain rule for covariant derivatives and applying it to the tensor product, one can rewrite equation (5.17) to

$$l_i = a_j\{\sum_{jkl}[\,\frac{1}{2}(\,\frac{\partial}{\partial x_i}+\frac{\partial}{\partial x_j})\;\mathbf{b_{ijkl}}\,]\;\epsilon_{\mathbf{kl}}\;+\;\sum_{jkl}\;\mathbf{b_{ijkl}}\;[\,\frac{1}{2}(\,\frac{\partial}{\partial x_i}+\;\frac{\partial}{\partial x_j}\;)\;\epsilon_{\mathbf{kl}}\;]\}\;, \tag{5.18}$$

where [...] indicates the scope of the partial derivatives and where i,j,k,l = 1,2,3 is the range of tensor components. Equation (5.18) is written in terms of its tensor elements and is valid for arbitrary anisotropic media. In the first term of equation (5.18) derivatives of the elastic constants are computed; in the second term only derivatives of the strain tensor are computed. The stiffness components in the second term act as constant coefficients for the derivated matrix elements. The derivative operation in equation (5.18) appears in both terms. It acts, however, on two different entities. In the first term the derivative of medium properties is taken and in the second term the derivative of the wave field is computed. In the end, summation of both terms produces the complete derivative without taking a derivative of the medium and wave field product, as equation (5.17) would suggest. For a medium with constant properties the first term vanishes, leaving the second term in equation (5.18), whose Fourier transform represents the Christoffel equation (Auld, 1973, Dellinger, 1991). From an economical point of view of equation (5.18), some expensive computations can be saved if the medium derivatives are calculated once and are then reused through the course of computation, but the real opportunity lies in computing the solution to equation (5.17) more accurately and more realistically for certain types of media. In particular, the derivative properties can be adjusted to match the properties the quantities.

5.6.2 Adaption of the derivative operator

Two basic observables in the wave equation are the wave field (e.g. stress, displacement) and the medium properties (stiffnesses, velocities). If the frequency content of the wave field and the material properties are similar, the same type of derivative operators can be used for both. Then (5.17) should be used rather than (5.18). The necessary condition is that model and wave field roughness are of the same scale. For example, if the medium properties vary smoothly in space then a fairly long derivative operator can be adequate to approximate the analytical derivative numerically. However, if the medium is varying rapidly, the same derivative operator may not be adequate. As an extreme example, realistic modeling of wave propagation in a random medium is affected by the length of the finite difference operator. A long operator implicitly averages medium properties. To describe local effects, a much shorter operator would be necessary, but for the same prescribed spacing, that operator might give a less accurate derivative. A layered model is an extreme case. The spatial variability in the horizontal direction is zero, while the variability in depth occurs in discontinuous steps. Applying equation (5.17) along with conventional finite difference operators to such a model would implicitly assume a continuous and band-limited medium. Equation (5.18) allows us to "separate" the medium from the wave field. One can adapt the derivative operators to the properties of the observables they are operating on.

5.6.3 Practical issues

In practical implementations, problems arise when one calculates derivatives of quantities on a staggered grid. A staggered grid (cf. Figure 5.12) can be chosen such that the outcome of a derivative calculation falls onto the grid point of the quantity, and is multiplied together in the next step of the algorithm. In such a scheme no quantity needs to be interpolated, and thus there is no loss in accuracy. Examining the partitioned constitutive equation, one can see how a commonly used staggered grid is chosen:

$$\begin{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{pmatrix} \\ \begin{pmatrix} \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{pmatrix} & \begin{pmatrix} c_{14} & c_{15} & c_{16} \\ c_{24} & c_{25} & c_{26} \\ c_{34} & c_{35} & c_{36} \end{pmatrix} \\ \begin{pmatrix} c_{41} & c_{42} & c_{43} \\ c_{51} & c_{52} & c_{53} \\ c_{61} & c_{62} & c_{63} \end{pmatrix} & \begin{pmatrix} c_{44} & c_{45} & c_{46} \\ c_{54} & c_{55} & c_{56} \\ c_{64} & c_{65} & c_{66} \end{pmatrix} \end{pmatrix} \begin{pmatrix} \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{pmatrix} \\ \begin{pmatrix} \epsilon_4 \\ \epsilon_5 \\ \epsilon_6 \end{pmatrix} \end{pmatrix} \tag{5.19}$$

The 4 quadrants of the stiffness matrix c are abbreviated by $\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ and their locations on the grid are noted in Figure 5.12. Stiffness values in the first diagonal A_{11} quadrant and the first portion of the $\sigma_{1..3}$ and $\epsilon_{1..3}$ vectors are chosen to lie on the primary grid. The second diagonal quadrant and the second part of $\sigma_{4..6}$ and $\epsilon_{4..6}$ are then necessarily chosen to lie on the secondary grid. The primary cube center coincides with the secondary cube edge. That scheme causes the derivative computation to evaluate quantities at the correct collocation points. In anisotropic elastic media up to orthorhombic symmetry, the previous notion works well, since the off-diagonal elements A_{12}, A_{21} of the stiffness matrix are zero and thus do not have to be evaluated. In anisotropic systems beyond orthorhombic those elements are non zero, such that even in non-split schemes interpolation to collocation points has to be included for staggered grid methods. When splitting the wave operator, one has to take care of proper collocation interpolation when using a staggered grid. Thus the interpolation issue can complicate the primary goal of adapting derivative operators to the quantities. Splitting of the wave operator also affects the numerical stability and numerical dispersion of the algorithm. One can apply Neumann-style analysis to obtain stability conditions and dispersion curves. Each of the two terms of the split wave operator each influences stability such that the overlap of those two is the region where the algorithm is stable and non dispersive.

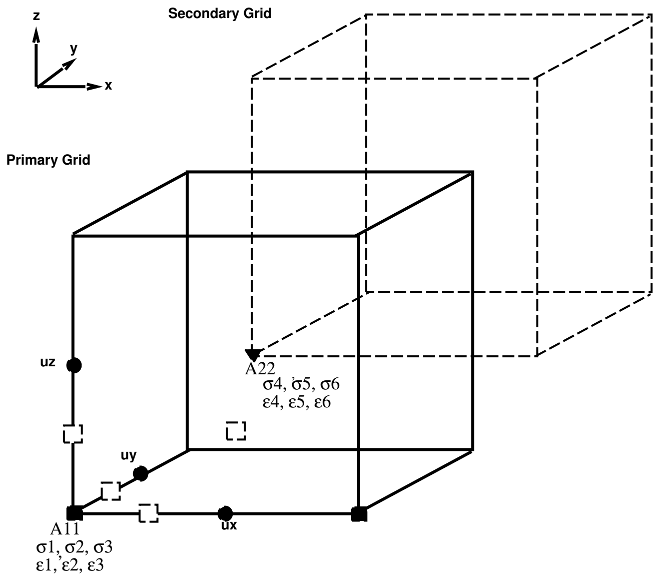


FIG. 5.12. The primary grid and secondary grid as usually defined for staggered grid applications for elastic wave modeling. The primary cube center coincides with the secondary cube edge. The positions of stress, strain and stiffness components are indicated (black triangles), together with the evaluation points of the split wave operator scheme (empty boxes). genmod-stagsplit [NR]

5.6.4 Equivalence of modeling equations

First I show that equation (5.18) and (5.17) are numerically equivalent and produce the same results, as predicted analytically. I propagate a wave field in a model that shows a step discontinuity but is otherwise homogeneous. The elastic wave propagates across a discontinuity in v_p and v_s. Figure 5.13 shows a single trace comparison of two modeling responses. One result is calculated using equation (5.18) and one result is obtained using equation (5.17). For both components the resulting responses overlie each other and indicate that both modeling equations are also numerically equivalent.

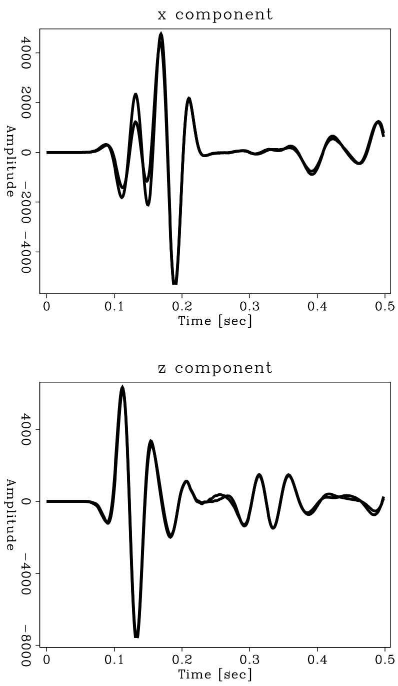


FIG. 5.13. Trace 90 is taken from two wave fields. The results from the two different modeling equations are overlaid and confirm that they are not only analytically but also numerically equivalent. The modeling was carried out on a non-staggered grid. The source is located on top of the step and the receivers are located below. genmod-equivalns [CR]

5.6.5 A long and short operator combination

Figure 5.14 is using the same model as in Figure 5.13. However, the difference lies in the types of operators used for calculating the wave field. For derivating the field quantities, a 6-point operator was used, while for derivating the medium parameters, a 16-point wide operator was used. The 16 point operator calculates the derivative of the step function more accurately and is less local than the shorter 6-point operator. The shorter operator is adequate for the wave field, since it is a smoother function than the medium and it is not necessary to use the full 16 point operator. The results of the split operator, the composite 16-point and the composite 6-point operator, are overlaid in Figure 5.14. One can see that the results from the composite 16-point and the split operator are hardly distinguishable, but that the result from the composite 6-point operator clearly shows differences and cannot adequately propagate the wave field. The step discontinuity has a very high frequency content and one prefers an accurate computation of the derivative of that discontinuity.

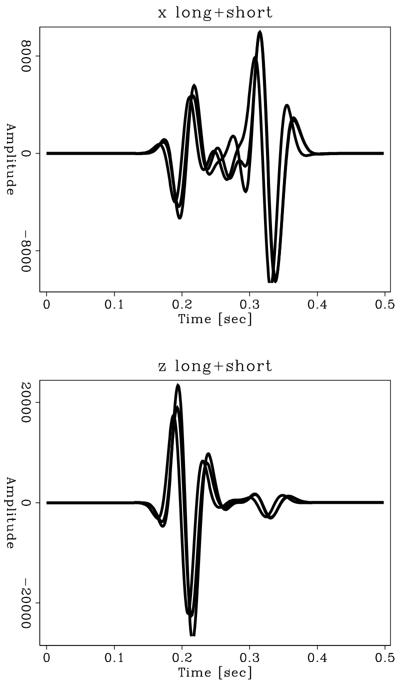


FIG. 5.14. A trace is taken from three wave fields, using three different modeling operators. The composite 16-point operator and a split operator, which match well and a composite 6-point operator which shows differences. are over lain. The source is located above a step discontinuity and the receivers are located below. $\boxed{\text{genmod-mlong+fshort}}$ [CR]

5.6.6 Summary

The wave operator can formally and numerically be split into parts that involve taking derivatives of either model parameters or propagating field quantities. For identical derivative operators this is the same as the conventional method that takes nested derivatives of those quantities. However, the advantage of using the split operators is the possibility to treat the two parts differently. I have given the splitting for completely anisotropic media. If one assumes different a priori properties of the medium, the operators can be adjusted to match the medium. Another advantage is the increase in efficiency that a certain optimum operator combination can achieve given a desired accuracy. So far adaptation to particular media have not been used much; a possible topic of future research is the use of wavelet transform for derivative calculations for medium and field variables and their proper back interpolation into the primary grid.

5.7 Nonlinear tensor wave propagation modeling

The interaction of a seismic source device with the free surface of the earth makes the radiation characteristic different from that of the equivalent body point source. Usual seismic sources are built to produce enough energy to penetrate a few kilometers into the earth. There is no guarantee that when the source is activated, the surface material behavior will stay within a region that can be appropriately described using infinitesimal stress strain relationships. In this section I use finite-differences to model radiation characteristics and wave propagation when the equations of motion are no longer linear. I extend a nonlinear scalar equation that is normally used in one dimension into a three component, three-dimensional setting.

5.7.1 Nonlinearities in a seismic experiment

When a time dependent forcing function is generated, nonlinear behavior can occur in the electronic amplification part of the system. These nonlinearities can be avoided by the use of feedback mechanisms (Baeten et al., 1989). Another nonlinear mechanism is the transfer of the desired signal into the ground. Most of the signal distortion occurs at this instance with high momentum devices by impact or vibration. The subsurface volume in which such mechanisms dominate is rather small, yet such nonlinear behavior shows up as an effective alteration of the signal spectrum and depends on radiation direction. It is this interaction of the device with the earth's surface that cannot be described adequately by the use of infinitesimal displacements or stresses. When one describes nonlinearities two effects come into play: the finiteness of the variables and the possibility of entering a nonlinear region in the stress-strain relationship. In general, the function describing the stress-strain dependency, (cf. Figure 5.15) may be arbitrary as long as some underlying conservation principles (energy, momentum) are satisfied. It is always possible to expand this function around an equilibrium point. A conventional elastic method would only use the linear term of such a Taylor series expansion. This works well if the magnitude of the variables is small. If the magnitude exceeds a certain limit, one has to use higher order terms in the approximation to describe the medium accurately. It is seldom the case that these higher order stiffness constants are measured. For this reason, this section mainly deals with effects introduced by the finiteness of the variables, but not with the nonlinear stress-strain relationship. Fortunately, the finiteness of displacements introduces no new material parameters that must be known or estimated. Allowing the displacements to be non-infinitesimal merely means that the volume is deformed by finite displacements. This is equivalent to allowing higher order terms in the Taylor expansion of the displacement field. This section does not deal with receiver properties, although the same framework can be applied to modeling and estimating receiver functions. I assume that receiver properties can be effectively described using their transfer function. Most of the external effects are likely to be caused by ground coupling. Stresses and displacements at receivers are generally many orders of magnitudes smaller than at sources.

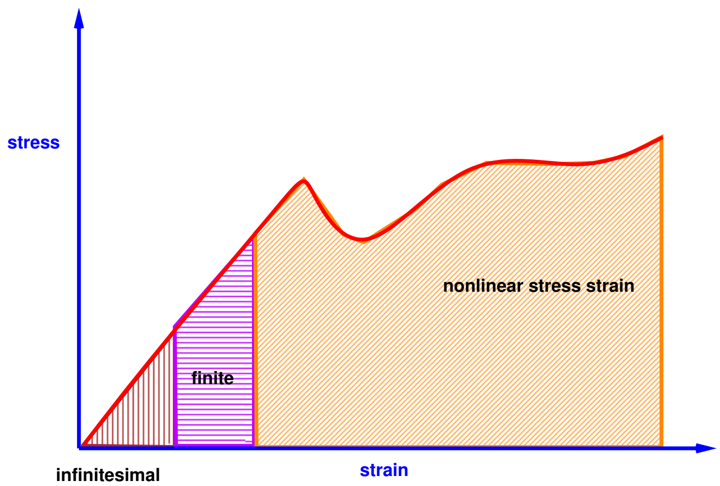


FIG. 5.15. Stress-strain relationships can be arbitrary, as long as some underlying conservation principle holds. The region of infinitesimal strain is at around 10^{-6}, while the geometric nonlinearity allows strain in the magnitude of 10^{-3}. Above that value the stress-strain relationship enters the nonlinear-medium region. genmod-stress-strain [NR]

5.7.2 Abandoning linearity

Geometric nonlinearity

If the deformation is less than 10^{-3} (Fung, 1965) in an elastic region (cf. Figure 5.15), nonlinearity is assumed to result from finite geometry and not from a nonlinear stress-strain relation. However, the line between geometric and material nonlinearity cannot be drawn clearly. Following Biot (1965), it is possible to compute the material deformation to second order as

$$\epsilon_{kl} = e_{kl} + 1/2(e_{k\mu}\omega_{\mu l} + e_{l\mu}\omega_{\mu k}) + 1/2(\omega_{k\mu}\omega_{l\mu}) \tag{5.20}$$

$$e_{kl} = 1/2(\frac{\partial u_k}{\partial l} + \frac{\partial u_l}{\partial k}) \qquad \text{and} \qquad \omega_{kl} = 1/2(\frac{\partial u_k}{\partial l} - \frac{\partial u_l}{\partial k}) \quad . \tag{5.21}$$

Neglecting the rotational tensor components ω in (5.20) results in the usual definition of the elastic symmetric strain tensor e. Following Fung (1965), we have a similar formulation for the Eulerian strain tensor:

$$\epsilon^E_{kl} = 1/2(\frac{\partial u_k}{\partial l} + \frac{\partial u_l}{\partial k}) - \frac{\partial u_m}{\partial k}\frac{\partial u_m}{\partial l} \quad . \tag{5.22}$$

Both equation (5.20) and (5.22) describe the deformation of a medium to a higher order than the usual elastic strain tensor e. The deformation in Figure 5.16 is a pure geometrical property and not a material property. Consequently, the linear Hooke's law

$$\sigma_{ij} \;=\; b_{ijkl} \;\; \epsilon_{kl} \tag{5.23}$$

remains unmodified. The relation between stress and strain components is still linear. It is the computation of ϵ_{kl} that changes in each case. When considering the displacement gradients, we can see that higher order gradient components are related to stress components nonlinearly. However, the parameters which link them, the stiffness coefficients b_{ijkl}, are still linear elastic parameters. All previous equations lead to an elastic wave equation of the form

$$\nabla \; \underset{\sim}{\mathbf{b}} \; \nabla^t \; \mathbf{u} - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} = \mathbf{f} \;, \tag{5.24}$$

or, in the geometrically nonlinear case, to a slight modification of the previous expression:

$$\nabla_g \; \underset{\sim}{\mathbf{b}} \; \nabla_g^t \; \mathbf{u} - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} = \mathbf{f} \;, \tag{5.25}$$

where ∇_g is the geometrically nonlinear derivative operator.

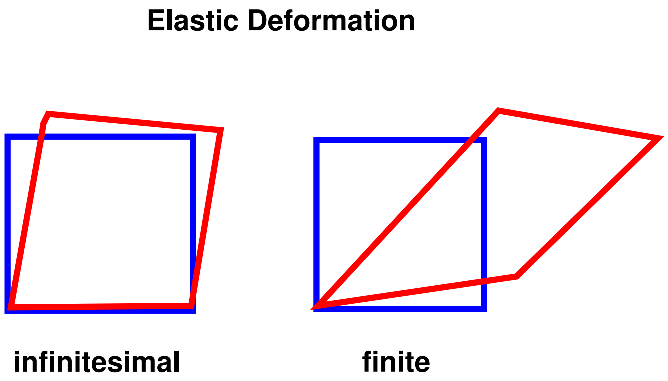


FIG. 5.16. Finite elastic deformation is a purely geometric effect. The strain does not yet enter the region where the medium itself behaves nonlinearly. Stress and strain components are still related through a linear relationship. genmod-finite [NR]

Material nonlinearity

The next higher level of complexity involves the use of nonlinear relations between stress and strain components. Even when displacement gradients are computed to first order, only the higher order material relationship introduces higher order effects in the differential equation. Material nonlinearity might easily be coupled with geometric nonlinearity; drawing a clear boundary between them might be hard in many cases. This kind of nonlinearity introduces new parameters into the differential equation, namely, new material constants (higher order elastic parameters). In many practical cases one might not have knowledge of these parameters, or they might be hard to estimate.

Omission of the second and higher order cross terms ensures that a wave equation is linear. If these terms are neglected in the Taylor expansion of $d\mathbf{u}$, one is still confined to examining only a small neighborhood around a reference point. The assumption that the products of the displacement gradients are small assures that the principle of superposition remains valid.

One needs to use equations (5.20) and (5.22) as soon as one considers finite displacements of the medium. It is practical to start with such a nonlinear description since it introduces only small modifications of existing modeling programs, while admitting some degree of nonlinearity. Since algorithms that are based on linear assumptions fail in this case, other methods have to be used to solve the problem. The wave equation, therefore, will be nonlinear in the spatial domain; in the time domain, the principle of superposition is still valid. In order to investigate the impact of nonlinearity on radiation patterns and signal wave forms, one can calculate numerical examples in which nonlinear wave propagation is modeled using finite differences in time and space. Such a model should serve as a lower limit of what we can expect from nonlinearities.

5.7.3 Modeling with finite-differences

Many geophysical numerical algorithms assume linearity. Thus, they will work only on equations that have a linear behavior. For modeling nonlinear phenomena, the choice of algorithms becomes rather restrictive. In most cases, an analytical solution is not possible or would involve some sort of linearization. That, in turn, would defeat the original purpose of investigating nonlinear effects. For modeling nonlinearities finite difference algorithms have the advantage of not having to make linearity assumptions in order to model nonlinear wave propagation. However, one problem with the use of finite difference methods remains: it is now much harder to calculate stability criteria properly, since many rigorous stability criteria are based on linearity. The usual methods use linear transforms to produce an easily calculated estimate of stability and dispersion of the algorithm. In a recent paper Kosik (1993) makes use of a Crank-Nicholson FD scheme that is implicit and is thus guaranteed to exhibit stability for a certain range of parameters, but how much such an implicit method averages out nonlinear effects is debatable.

Applicability

For investigating surface source behavior, the previously described notion of finite stresses and strains in a medium can be readily applied to surface source of different types. Multi-component impulsive or vibratory sources could generate a region in the subsurface in which such nonlinear behavior is appropriate. However, such a region, as shown in Figure 5.17, might be relatively small compared to the source wavelength. For data processing, a far-field description with an effective source is more efficient. It allows a different parameterization of source behavior that depends on the experiment type and the data itself. However, when modeling source behavior for the purpose of testing source equalization schemes, such as those described in chapter 6, it is valuable to have numerical algorithms that allow modeling of nonlinear source behavior.

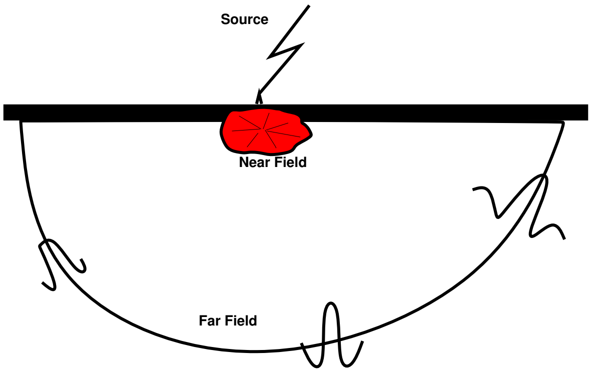


FIG. 5.17. Nonlinear near-field source behavior generates far-field effects, such as amplitude variations and wavelet frequency content that can vary with angle. For seismic data processing an effective far-field source behavior has to be determined. genmod-region [NR]

5.7.4 Chaotic sources

The above paragraph describes the nonlinear source behavior of an elastic medium that obeys the "usual elastic" anisotropic wave equation. However, an investigation by Walker (1993) shows that the time dependence of source-surface interactions of vibrators can be effectively described using a nonlinear "chaotic" wave equation. Walker uses the well-known Duffing equation to parameterize vibratory sources and nonlinear propagation in the subsurface. The one dimensional Duffing equation is given as

$$\alpha u + \frac{\partial^2 u}{\partial t^2} + \delta \frac{\partial u}{\partial t} + \gamma u^2 + \beta u^3 = F \cos(\omega t) \quad , \tag{5.26}$$

where α, β, γ and δ are adjustable parameters that govern the nonlinear behavior, u is a scalar quantity, F is the strength of the periodic forcing function, and ω the temporal frequency.

The time dependent solution $u(t)$ of the Duffing equation exhibits typical behavior for a wide range of parameters. When the source-surface system is at a high energy level, the amplitudes are large. Harmonics that alter the high frequency part of the spectrum of the original source wavelet are then generated. When the amplitudes are low, the nonlinear behavior creates subharmonics that introduce low frequency peaks. Such a period-doubling phenomenon is typical of chaotic equations. The typical chaotic behavior, however, occurs in band-limited intervals of the forcing function's spectrum. Outside those intervals, the equation behaves pseudo-regularly. The following figures show some of that behavior. Unless otherwise noted, the parameters are set to zero. In practice, only a of the effects that are possible, are indeed observed with surface seismic data: top- and sub-harmonics, as illustrated in the third and fourth plot from the top in Figure 5.18. The top plot in Figure 5.18 shows a constantly growing signal, resulting from a system that is not damped or limited in any way. The second plot from the top shows a damped linear system. The phase space (displacement plotted versus velocity) is a spiral approaching a constant orbit. The time domain response is well behaved and the spectrum shows a single peak developed around the center frequency of the forcing function. Some harmonic peaks, in the third plot from the top, are generated in the spectrum beyond the peak of the forcing function. High amplitudes arise by means of an emphasis on the third order term in the equation. As the fourth plot in Figure 5.18 illustrates, subharmonic peaks are generated in the spectrum below the peak of the forcing function. Relatively low amplitudes are generated here, in contrast to the amplitudes for topharmonics. Nonlinear behavior has developed in the signal response (shown in the fifth plot from the top in Figure 5.18) at this parameter setting and the signal is apparently erratic in the phase space. The spectrum of the forcing function is changed in a non-systematic way. Compare this to the structured modifications that generate subharmonics and topharmonics. The last two plots in Figure 5.18 show slightly different parameter settings and illustrate the unstable character of this equation. Small changes in parameters produce large effects in the solution behavior.

5.7.5 Duffing extended for three components and three dimensions

The original Duffing equation describes a scalar chaotic process and has been used in the past to model and to analyze the effective time behavior of a source-surface system. I extend the scalar equation and use it for multi-component sources in an anisotropic 3-D medium. Equation (5.26) is a harmonic oscillator equation with additional nonlinear terms added. I construct a similar equation for the case in which the wave field is of vectorial nature and extends in three dimensions. Starting with the general elastic wave equation (5.1), the first two terms in (5.27) correspond to the harmonic oscillator analog of equation (5.26).

$$\nabla \underset{\sim}{\mathbf{b}} \nabla^t \mathbf{u} - \rho \frac{\partial^2 \mathbf{u}}{\partial t^2} + \nabla \underset{\sim}{\mathbf{c}} \nabla \frac{\partial \mathbf{u}}{\partial t} + \underset{\sim}{\gamma} ||\mathbf{u}||^2 + \underset{\sim}{\beta} \mathbf{u} ||\mathbf{u}||^2 = \mathbf{F}(\mathbf{x}, t) \tag{5.27}$$

I extended the terms to allow for three space dimensions and tensorial stress and strain quantities instead of a one-dimensional displacement; thus I also allow for fully anisotropic behavior. The third term $\nabla \underset{\sim}{\mathbf{c}} \nabla \frac{\partial \mathbf{u}}{\partial t}$ adds a viscous damping factor; in equation (5.26) this

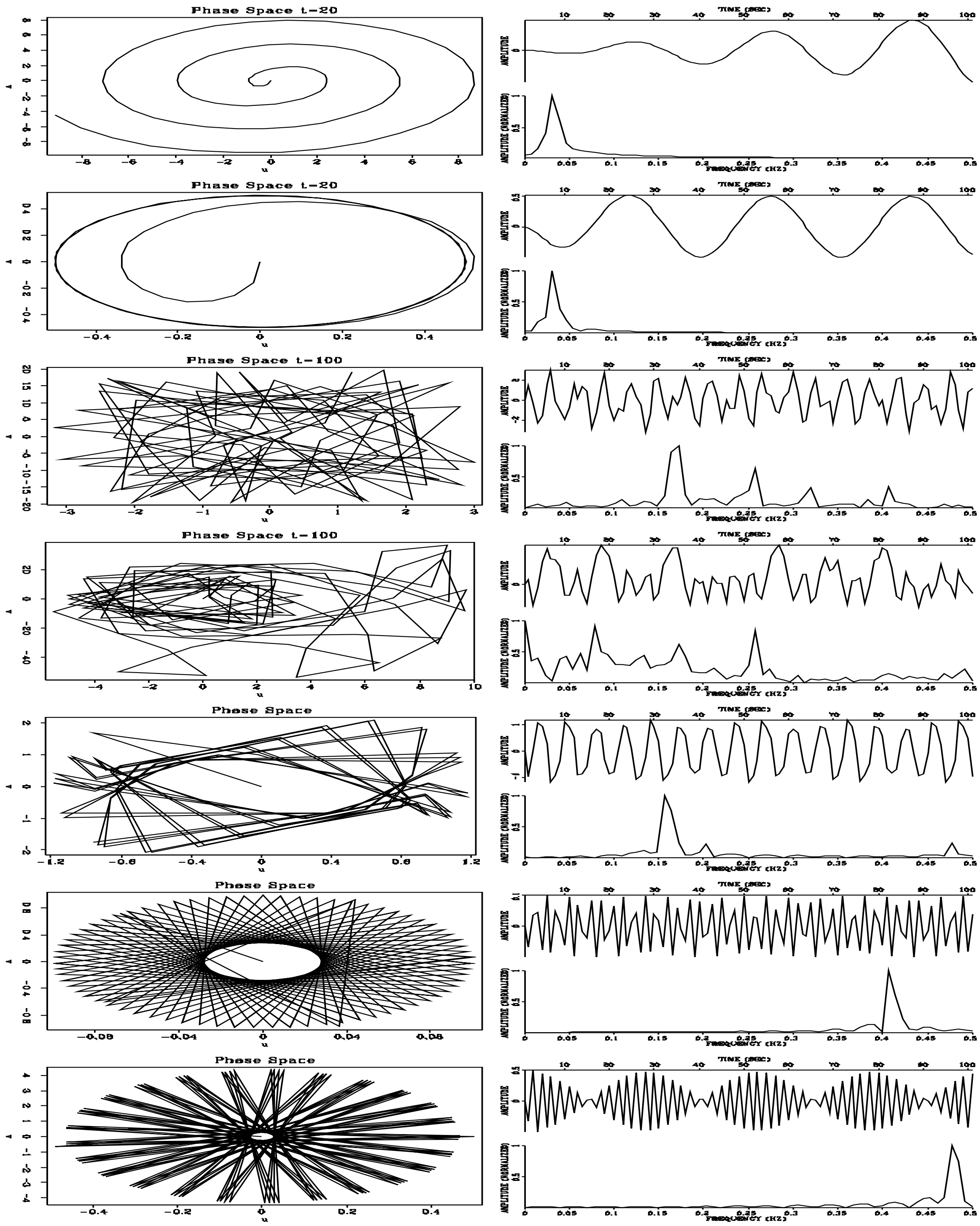


FIG. 5.18. Different parameter settings result in completely different behavior of the equation. Phase space plots are on the left and signal and spectral representations are on the right. For seismic waves, generated under realistic conditions by vibrators, only the third to fifth plot (top-harmonics, subharmonics and stable) from the top are significant.
genmod-chaos [CR]

is only a scalar parameter, while in equation (5.27) it becomes a matrix of viscous moduli. The fourth term, $\underset{\sim}{\gamma}||\mathbf{u}||^2$, augments the equation by a force that is proportional to the squared magnitude of the displacement. In both equations the total displacement governs the influence of this term. In equation (5.27) the parameters can be of tensorial nature, affecting each displacement vector component differently compared to the original equation (5.26). The fifth and last term, $\underset{\sim}{\beta}\mathbf{u}||\mathbf{u}||^2$, adds an additional force to the equation that is proportional to the third power of the displacement components. This is similar to a hard spring force taking effect when amplitudes become large, because of the parameters' tensorial nature, each of the tensor components can be affected differently. It seems that there are many additional parameters that one needs to know in order to use this equation, but one can restrict the number of unknown parameters by limiting $\underset{\sim}{\gamma}$ and $\underset{\sim}{\beta}$ to being scalars or tensors that have the same relative tensor component magnitude as the elastic stiffness tensor.

In Figure 5.19 and 5.20 I modeled 2D wave propagation using equation (5.27) in a homogeneous isotropic medium to highlight some effects. All the plots consist of sections cut through a fully modeled z-component wave field cube. Each section corresponds to a different face of the cube. The cube has three axes: distance, depth and, propagation time. The front panels show a wave field snapshot taken at about 0.755 seconds propagation time. The top panels correspond to seismograms recorded at the surface. The side panels display the recordings in a bore hole that is located at a 2.8 km distance. In all of the experiments shonw, the source was an identical vertical force acting in about 0.4 km depth. Since the top surface of the model simulated a free surface boundary condition, the upward propagating wave field reflects and follows the primary wave front down into depth with its converted waves.

The top plot in Figure 5.19 shows the propagation of a wave front using a pure visco-elastic medium. In other words, medium parameters $\underset{\sim}{\gamma}$ and $\underset{\sim}{\beta}$ are zero while parameter $\underset{\sim}{\mathbf{c}}$ is given as a small fraction of the elastic moduli $\underset{\sim}{\mathbf{b}}$. The wave field shows a lower frequency content than the purely elastic modeling in the bottom plot. The primary function of the viscosity terms is to disperse the wave field and to damp off high frequencies as propagation continues. In the top plot of Figure 5.20 I activated all the terms in equation (5.27). I compared the higher-order modeling equation to visco-elastic modeling and the difference is a low frequency distortion of the wave field. It shows the strongest effect where the amplitudes in the visco-elastic modeling are high. This is in agreement with the analytical prediction, as the additional forces only depend on the magnitude of the displacement components and thus only indirectly on the medium. A very low frequency tail is visible around the source location. The source component was strongest in the vertical direction at that location and this has a far-reaching effect on the wave field. It is the interplay of a hard spring force and the magnitude force that interacts with the near-source region as long as amplitudes are high. In the bottom plot of Figure 5.20 I model propagation with one of the higher-order terms turned on while the other one is turned off. To better compare the wave fields, I compute the difference of the two displacement fields. The biggest differences are around the high amplitude portions of the wave field; but low-frequency broad-range differences also appear at low amplitudes in the form of expanding circular wave fronts.

5.7.6 Summary

When extending linear wave propagation to a nonlinear regime one can start with a purely geometric parameterization to enter the region of finite stresses and stains: this is a first step that has the advantage of requiring no knowlegde of new material parameters. I extend the one-dimensional Duffing equation to three-component 3-D wave fields and show some characteristic differences between purely elastic and visco-elastic and nonlinear wave propagation. The nonlinear propagation is only important when amplitudes are high, such as in the near source region.

5.8 Summary

In this chapter I have described the design and implementation of a modeling tool box mathematically and algorithmically. My goal is to have a portable rapid prototyping environment for new modeling and imaging algorithms that can be designed for any dimension and with a variety of underlying mathematical equations. I succeeded in implementing basic building blocks and algorithms in the Ratfor90 (Ratfor + Fortran90, HPF) programming language. I chose this language beacuse of the compute-intensive nature of seismic wave propagation in realistic complex earth models. I hope that in the future this numerical tool box will be converted to an object-oriented frame work such as C++ and achieve the same performance level. In the demonstrated modeling algorithms I use primarily finite-difference approximations for partial derivative calculation. The Marmousi velocity

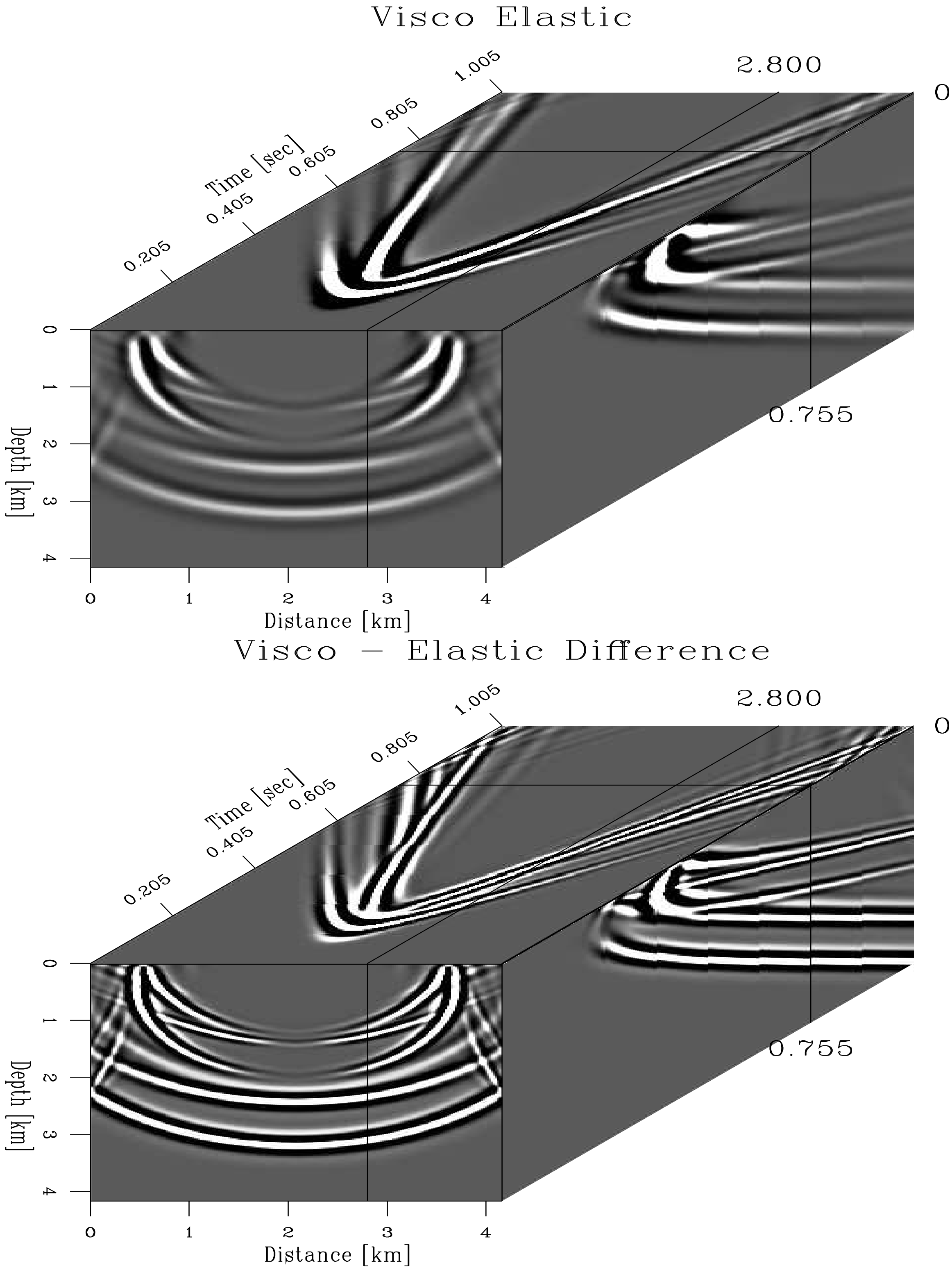
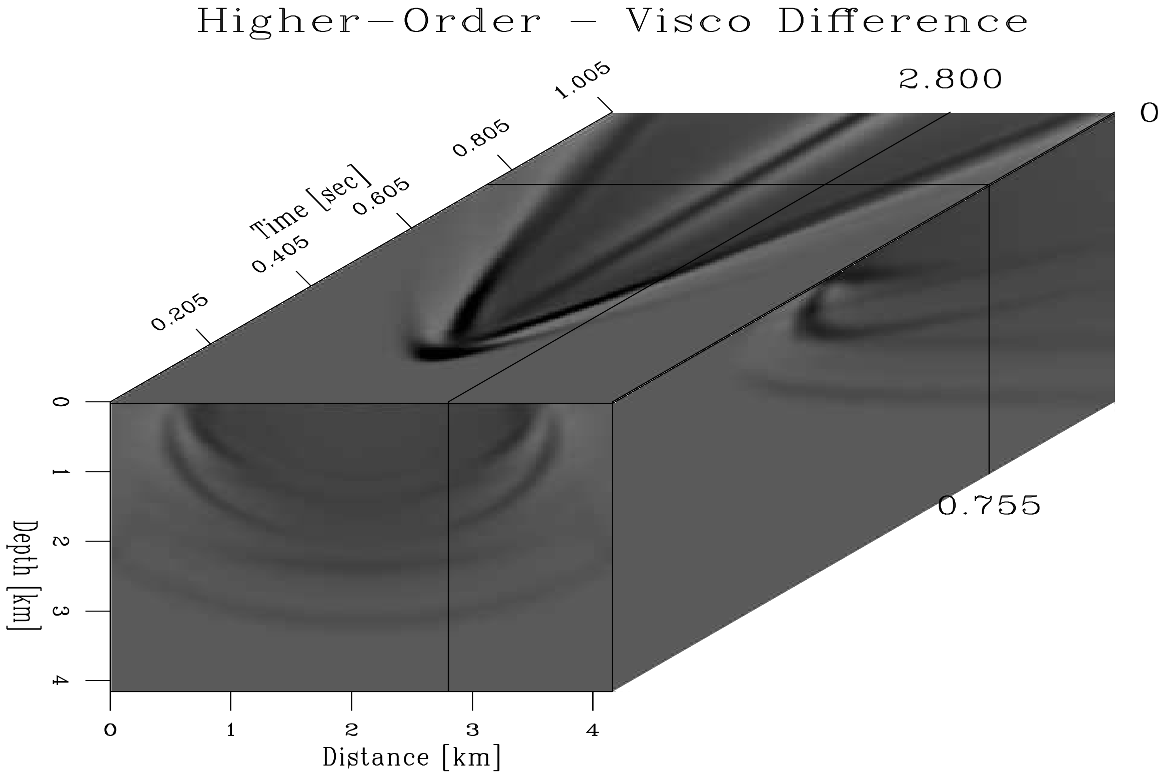


FIG. 5.19. The top shows the visco-elastic modeling. The bottom shows the difference between elastic and visco-elastic modeling. genmod-elvis [CR]



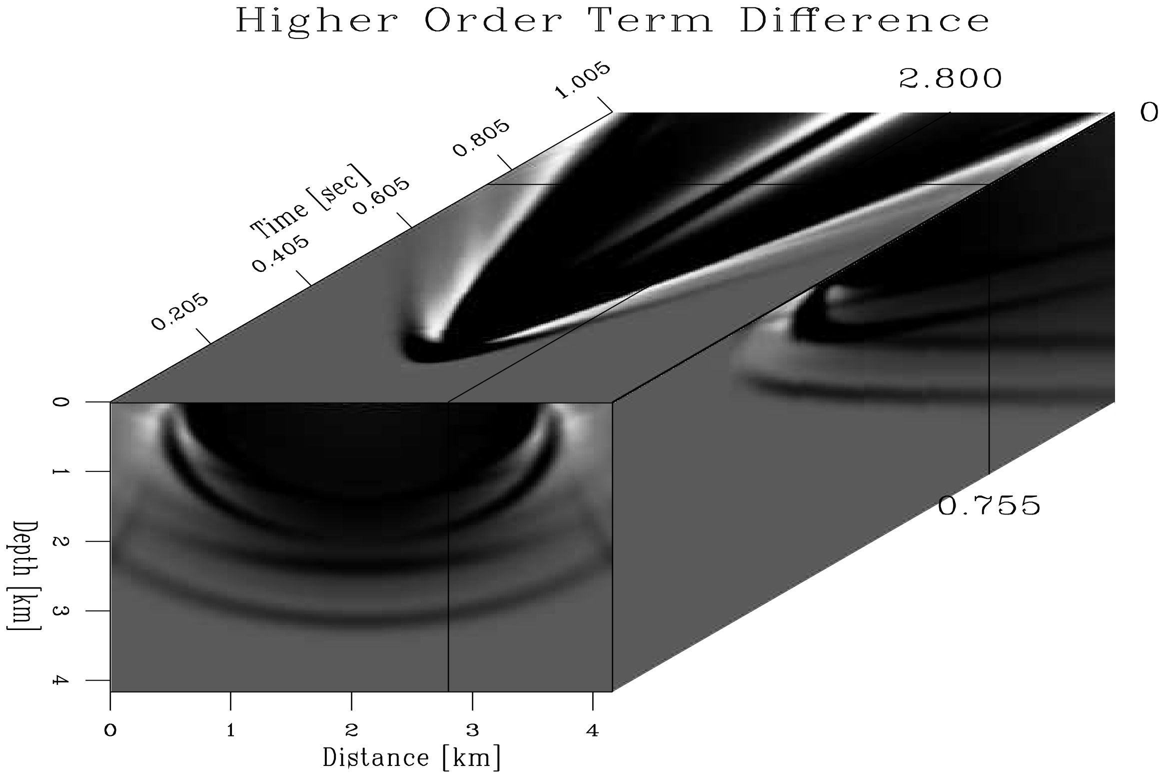


FIG. 5.20. The top shows the difference between visco-elastic modeling and modeling with equation (5.27). The bottom shows differences caused only by alternately switching on one higher order term. genmod-diffvisduff3 [CR]

and density model is a data set and subsurface model designed to test acoustic imaging and velocity estimation algorithms. I use that original model and create a new "elastic" subsurface model assuming a given Poisson's ratio. Modeling wave propagation acoustically and elastically using identical algorithms reveals differences in the recorded wave field that are due to energy conversions caused by elastic propagation. Thus, the acoustically modeled Marmousi dataset is a good test data set for algorithms that were created under the assumption of pure acoustic propagation. However, wave propagation in the real earth is more complicated and acoustic algorithms would experience more difficulties when dealing with more realistic, elastically-modeled seismic data.

Since true amplitude modeling is important for material parameter estimation, it is important to confirm how closely the numerical amplitude behavior agrees with the analytical prediction. I show an example where the reflection response from a single interface agrees well with the analytical Zoeppritz plane wave reflection coefficients. Obviously, if the numerical grid is made very small compared to the wave length of the scattering medium, the finite-difference algorithm will reproduce analytical behavior well. The problem lies in trying to maximize the coarseness of the computational grid so as to minimize computation time and computational resources, but to still get an accurate solution to the problem at hand. Many factors come into play when one tries to achieve this goal. I try to solve one of them by splitting the wave operator using the chain rule of derivatives and implementing a medium and wave field adaptive derivative. I can thus model wave propagation with an optimal derivative operator for both the wave field and the medium parameters.

When one models wave propagation in the near field of seismic sources, the normal linear wave equation fails to adequately describe the wave propagation effects. I use the well known finite-stress-strain description to augment the infinitesimal, linear elastic wave equation to regimes of finite stresses and strains using modifications of the usual linear derivative operator. The advantage of this approach is that it does not require an estimation of new material parameters. In a second approach I use the one-dimensional Duffing equation and create an anisotropic, three-component, three-dimensional nonlinear wave equation. The Duffing equation is often used to demonstrate chaotic behavior of a physical system when it is periodically driven. Therefore such an extension might be well suited to a seismic vibrator source. Nonetheless, I use the same equation for impulsive sources and show characteristic differences in wave fields produced by propagating in a purely elastic, visco-elastic, and nonlinear homogeneous medium.

The application of the wave propagation tool box to a variety of problems confirms the versatility that this frame work offers for turning mathematical concepts into numerical algorithms. Easy design, implementation and prototyping are possible through the use of a general object-oriented approach.

5.9 Appendix: Maximizing self-similarity of convolutional operators

Many geophysical problems involve the repeated application of identical operators to some data. A few example can be found in recursive schemes, recursive wave extrapolations, or multi-grid methods. For numerical implementation an approximation to an operator is usually used, which deviates from the true desired behavior of the operator. What happens, then, if the approximate operator is repetitively applied multiple times ? The answer lies in the existence of a Central Limit Theorem for that particular operator. Muir and Dellinger (1986) describe a way for recursive NMO application in the paper *How to Beat the Central Limit Theorem.* I illuminate this idea further and contrast it with other optimal design criteria for approximate operators.

Repeated convolution with identical operators

A smoothing filter, as shown in Figure 5.22, for example, can be represented as a convolutional operation with a three-point filter of the form (0.5,1,0.5). Let the operator that convolves it with some arbitrary input vector $\mathbf{x}$ be called $\mathbf{A}$:

$$\mathbf{A}\,\mathbf{x} \;=\; \mathbf{y} \tag{5.28}$$

The n-fold application of the operator $\mathbf{A}$

$$\mathbf{A}^n\mathbf{x} = \mathbf{y}_{(n)} \tag{5.29}$$

results in the signal $\mathbf{y}_{(n)}$. In the limit of large n the composite operator $\mathbf{G} = \mathbf{A}^n$ can converge to some stable operator $\mathbf{G}$. This is the essence of the central limit theorem, which is illustrated in Figure 5.22. The 400-fold application of the smoothing operator tends to the equivalent of the single application of the limit-operator G, which has the shape of a Gaussian function. The result has been obtained by simply allowing x to be

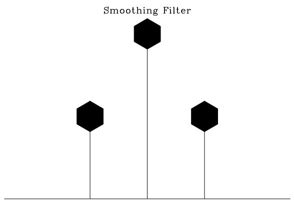


FIG. 5.21. Triangular smoothing filter. genmod-smoothing [CR]

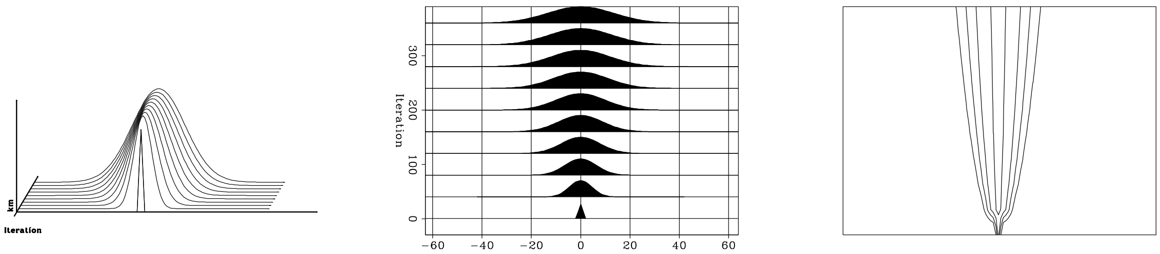


FIG. 5.22. The triangular smoothing was applied 400 times to a unit impulse; the equivalent composite convolutional operator tends to a bell-shaped Gaussian function. This is the essence of the central limit theorem. genmod-triangle [R]

an impulse function. This is an example of the well-known Central Limit Theorem that can be found in text books (Claerbout, 1992); the theorem itself has widespread use in many disciplines and is still an active research area (Yurinskii, 1977, Kerley, 1988). According to the Central Limit Theorem, the result of raising a function to a high power is dominated by the neighborhood of the highest value in the function. This statement holds true if the function displays a simple behavior, such as having only one dominant maximum and a non-zero second derivative at that maximum.

One can expand a convolutional operator $F(\omega)$ in the Fourier domain about its dominant frequency ω_0 into a power series in ω:

$$F(\omega) = 1 - A_1(\omega - \omega_0)^1 - A_2(\omega - \omega_0)^2 - A_3(\omega - \omega_0)^3 - \ldots . \tag{5.30}$$

Repeated convolution in time corresponds to repeated multiplication with this polynomial in frequency. After many multiplications with itself, the first nonzero term A_k in this expansion will dominate. Thus, the limit results in

$$\lim_{n\to\infty} F(\omega)^n = 1 - A_k(\omega - \omega_0)^k = e^{-A_k \omega^k} . \tag{5.31}$$

In particular, if the expansion point ω is at a maximum point ω_0, the first-order coefficient A_1 is zero. Thus, the second-order term $A_2(\omega - \omega_0)^2$ will dominate the result.

Hence, the above smoothing operator tends to a bell-shaped Gaussian $e^{-a\omega^2}$ in the frequency domain, which, in the original domain, is merely another Gaussian e^{-bt^2}.

Therefore, the Central Limit Theorem, as it stands above, is only applicable to a limited number of functions, namely, those that have a dominant maximum and a non-vanishing second derivative at that maximum.

An example of such a well-behaved filter is an operator that interpolates values between sample points. Such a filter is used frequently in, for example, multi-grid methods. In iterative schemes, this interpolation filter is applied many times and it too will tend to a certain Gaussian shape in the limit.

Figure 5.23 shows the application to a simple two-point interpolation filter $(0.5, 0.5)$, which evaluates the discrete function halfway between sampled points. Repeated

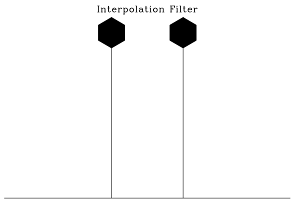


FIG. 5.23. Simple interpolation filter. $\boxed{\text{genmod-2point5}}$ [R]

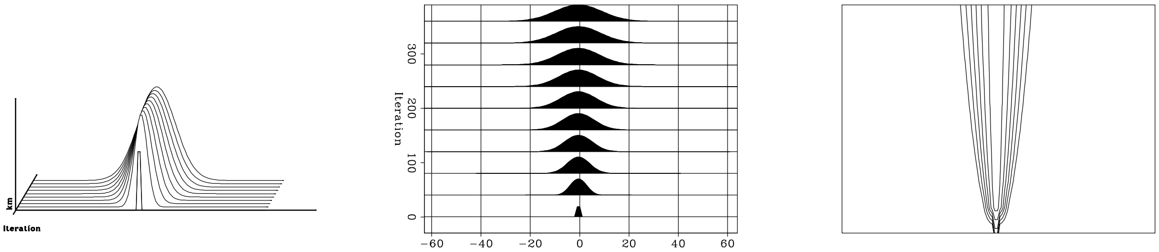


FIG. 5.24. Two point interpolation filter. $\boxed{\text{genmod-smear}}$ [R]

application of this operator tends to smear out the original signal, as we can see in Figure 5.24, where the operator has been applied 400 times. As, illustrated in Figure 5.22, the result is in the limit a Gaussian-shaped operator. A composite operator F^{400} would clearly smear the input signal over a considerable distance. Would it be possible to construct an operator that maintains its similarity under the operation of convolution ? The next sections show how to formalize such a design criterion.

Designing a self-similar convolutional operator

If one wants to interpolate a continuous, band-limited signal at some point Δ between the grid points with a convolutional operator, then the interpolation filter might have the $2N$ coefficients

$$f_{-N+1},\ f_{-N+2},\ \dots,\ f_0,\ f_1,\ \dots,\ f_N.$$

A normalization of the sample interval to 1, results in the interpolation distance ranging from 0...1, while the Fourier transform F of the interpolation filter has the form:

$$F(\omega) = \sum_{j=-N+1}^{N} f_j \mathrm{e}^{-i\omega(j-\Delta)}. \tag{5.32}$$

This expression results in a continuous spectrum, although the original filter time series was discretely sampled. The ideal interpolation operator, if one could construct it, would only shift the signal by the amount Δ and then evaluate it at that point. Given that the desired operator has the exact Fourier transform $e^{-i\omega\Delta}$, it is possible to find an operator that is optimum in some sense by minimizing

$$\min_{f_j} \ || \, [\sum_{j=-N+1}^{N} f_j \mathrm{e}^{-i\omega(j-\Delta)}] - e^{-i\omega\Delta} \ || \tag{5.33}$$

for a given range of frequencies. Note that the norm to be used is not specified, but a conventional choice is the L_2 norm, which minimizes the sum of the squared residuals. In that case, equation (5.33) has to be solved using non-linear least-squares solvers in order to obtain the best choice of coefficients f_j. The choice of the number of frequencies used in that optimization specifies if the system of equations is over-determined or completely determined. A number of frequencies that is less than the number of coefficients leaves the system with a non-unique solution.

However, such choice of optimization criterion does not consider a repeated application of the same filter to a given signal. In fact, if the optimization leaves a sizeable oscillating residual in the spectrum, the repeated application of the filter will accentuate those maxima and minima and, in the limit, these will dominate the filter behavior.

Alternatively, one can try to optimize for the self-similarity of the filter under repeated convolution. Since the repeated auto-convolution is governed by the Central Limit Theorem, one has to choose the filter coefficients f_j in such a way that the limiting shape coincides with the original shape as well as possible. Equation (5.32) can be rearranged in the form

$$F(\omega) = \sum_{k=0}^{\infty} A_k \omega^k \ , \tag{5.34}$$

where the coefficients A_k are given as

$$A_k = \left[\sum_{j=-N+1}^{N} f_j \frac{(-i(j-\Delta))^k}{k!} \right] . \tag{5.35}$$

To maximize self-similarity under repeated convolution for the interpolation filter, one wants to have as many coefficients A_k as possible, with the exception of $k = 0 : A_0 = 1$, being zero. This is because the first non-zero term in the expansion is $-A_{2N}\omega^{2N}$, whose limiting form under repeated convolution is $e^{-A_{2N}\omega^{2N}}$. The coefficients f_j can be found from solving a linear system of equations

$$\mathbf{A}\,\mathbf{f} \ = \ \mathbf{b} \tag{5.36}$$

where $\mathbf{A}$ is the matrix with the following coefficients

$$\begin{pmatrix} 1 & 1 & \dots & 1 & \dots & 1 & 1 \\ (-N+1-\Delta) & (-N+2-\Delta) & \dots & (-\Delta) & \dots & (N-1-\Delta) & (N-\Delta) \\ (-N+1-\Delta)^2 & (-N+2-\Delta)^2 & \dots & (-\Delta)^2 & \dots & (N-1-\Delta)^2 & (N-\Delta)^2 \\ (-N+1-\Delta)^3 & (-N+2-\Delta)^3 & \dots & (-\Delta)^3 & \dots & (N-1-\Delta)^3 & (N-\Delta)^3 \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots & \vdots \end{pmatrix} \tag{5.37}$$

Since $k!$ and $(-i)^k$ terms are constants across any row of the matrix $\mathbf{A}$, they have been canceled from the equations. The unknown vector is

$$f^T = (\, f_{-N+1} \quad f_{-N+2} \quad \dots \quad f_0 \quad \dots \quad f_{N-1} \quad f_N \,) \tag{5.38}$$

and the right-hand side $\mathbf{b}$ for the interpolation filter is

$$b^T = (\, 1 \quad 0 \quad \dots \quad 0 \quad \dots \quad 0 \quad 0 \,) . \tag{5.39}$$

Figures 5.25 and 5.26 show the example of the eight-point operator that was obtained by inverting the system of equations. Repeating the convolution with an initial spike 400 times results in no series loss of shape, although the amplitude decays very slowly. The contour plot on the right indicates hardly any shape change for large numbers of iterations.

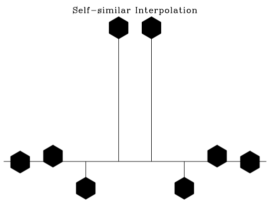


FIG. 5.25. Self-similar interpolation filter, obtained by solving system (5.37). $\boxed{\text{genmod-interp8}}$ [R]

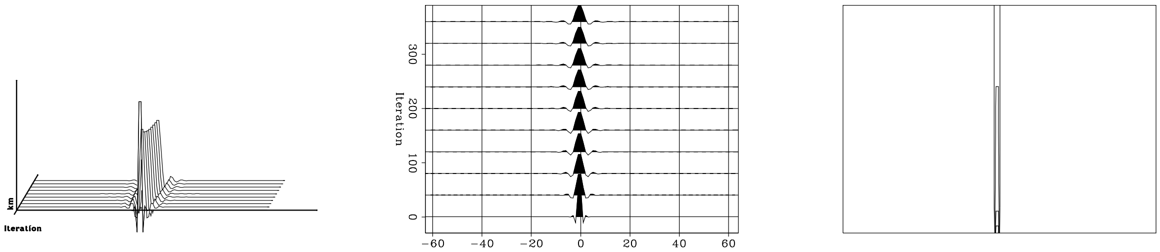


FIG. 5.26. Self-similar interpolation operator. No serious loss in shape has occured even after 400 application of the interpolation filter. $\boxed{\text{genmod-interpS8}}$ [R]

Designing self-similar differential operators

The previous example showed the case of an optimally self-similar interpolation filter. It was based on the fact that the power series expansion in the frequency domain was dominated by the "zero" frequency component and thus higher-order terms were purposely suppressed. However, what can be done if the filter has no "zero" frequency component and in fact the first term involving ω in the power series is the most important one, as is the case for the first-order differential operator ?

In this instance, the notion of The Central Limit has to be extended to mean something different from the above. Assume operator A is not as well behaved, as in the previous examples, but has multiple maxima and minima, or no region of dominance. Assume further operator A_* has an exact analytical expression and A is an approximation thereof. Then one can rephrase the Central Limit Theorem, so that the limiting operator

$$\lim_{n\to\infty} (A_*^{-1}A)^n = G \tag{5.40}$$

is given as a multiply-applied composite of the exact inverse of A_* and A. If $A_*^{-1}A = 1$, then it would not matter how often the operator pair is applied, the result would always be the same. In fact, when derivatives are computed in the Fourier domain, the data are transformed and multiplied by $-i\omega$. The inverse process is dividing by $-i\omega$. And for

$\omega \neq 0$, this process is exact (on the computer up to machine precision).

In contrast, if A is an approximation to the exact operator, the operator pair $A_*^{-1}A$ will not be identity, but, instead of having a constant spectrum of unit amplitude, the composite will deviate from unit amplitude. Typically these deviations occur for high frequencies. Repeated application of this defective operator will lead to growing errors that cause the operator shape in the limit of large n to differ from the outset.

Based on the designing a self-similar interpolation operator, one already has the necessary tools available to design a self-similar differential operator that will not change shape, as the Central Limit in equation (5.40) would dictate. For example, the first-order derivative operator is often used in wave propagation problems, where wave field components are derivated and implicitly integrated to advance the wave field in time or depth. The derivative operator has to be applied many times to the wave field. In the Fourier domain the mth-order derivative operator has exactly one term, namely

$$D_m(\omega) = A_m \omega^m \tag{5.41}$$

and all remaining terms are zero. Conversely, the approximate operator has the Fourier transform that is given in equation (5.32). To design an optimally self-similar approximation, one needs to find filter coefficients f_j in order to zero as many coefficients A_k as possible, except the coefficient A_m, which needs to be unity. This results in a right-hand side

$$\mathbf{b}^T = (\,0 \quad 0 \quad \ldots \quad 1 \quad \ldots \quad 0 \quad 0\,) \tag{5.42}$$

which has a 1 in the mth element, but is otherwise zero. The solution of equation (5.37) for an eight-point first-order operator is shown in Figure 5.27 and values are listed in table 5.1. The evaluation point of the derivative is between the original sample values; and in Figure 5.3 I compute a wave field using these derivative coefficients.

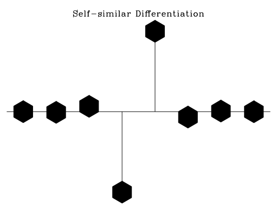


FIG. 5.27. Self-similar first-order derivative. genmod-deriv8 [R]

2pt	4pt	6pt	8pt
			6.97545E-04
		-4.68750E-03	-9.57032E-03
	4.16667E-02	6.51042E-02	7.97526E-02
-1	-1.12500	-1.17188	-1.19629
1	1.12500	1.17188	1.19629
	-4.16667E-02	-6.51042E-02	-7.97526E-02
		4.68750E-03	9.57031E-03
			-6.97544E-04
3pt	5pt	7pt	9pt
			-8.92857E-04
		5.55556E-03	1.26984E-02
	-4.16667E-02	-7.50000E-02	-1.00000E-01
0.5	0.666667	0.750000	0.800000
-1	-1.25000	-1.36111	-1.42361
0.5	0.666667	0.750000	0.800000
	-4.16667E-02	-7.50000E-02	-1.00000E-01
		5.55555E-03	1.26984E-02
			-8.92857E-04

Table 5.1. Self-similar derivative coefficients for the first (top) and second (bottom) order derivative.

Summary

Self-similarity can be an optimization criteria for convolutional operators where the full nonlinear problem of finding the optimum representation does not to be solved. Differential operators of any order can be designed with the use of the self-similarity criterion, which in turn uses a modified Central Limit Theorem. Application to interpolation and wave propagation problems show that the resulting operators perform as theory predicts.