ELSEVIER

Applied Numerical Mathematics 39 (2001) 151–171

APPLIED NUMERICAL MATHEMATICS

www.elsevier.com/locate/apnum

Use of coordinate-free numerics in elastic wave simulation

Helmer André Friis [a,*], Tor Arne Johansen [b], Magne Haveraaen [c], Hans Munthe-Kaas [c], Åsmund Drottning [d]

[a] *RF-Rogaland Research, P.O. Box 8046, 4068 Stavanger, Norway*
[b] *Institute of Solid Earth Physics, University of Bergen, N-5020 Bergen, Norway*
[c] *Institute of Informatics, University of Bergen, N-5020 Bergen, Norway*
[d] *UniGEO a.s., N-5008 Bergen, Norway*

Abstract

The modeling of elastic waves in solids and fluids is widely applied in physics and geophysics and often requires large computer codes for its realization. Despite the fact that different wave propagation problems have many similarities from the point of view of the abstract mathematical formulation, such codes are usually hard to adapt to new problems and changing requirements.

In this paper we discuss a so-called coordinate-free approach for the general computer implementation of tensor-field equations. We show how it applies to the modeling of seismic waves in isotropic and anisotropic structures, sonic waves in boreholes and ultrasonic waves in poro-elastic fluid-filled materials. The results clearly indicate that the coordinate-free approach is very flexible for the implementation of various wave propagation problems. © 2001 IMACS. Published by Elsevier Science B.V. All rights reserved.

Keywords: Seismic modeling; Coordinate-free numerics; Anisotropy; Poro-elastic modeling; Object-oriented numerics

1. Introduction

Schemes for accurate and fast numerical simulation of elastic wave propagation in earth materials are being used in 2D/3D seismic imaging for recovering subsurface oil and gas reservoir properties. Seismic wave simulation is performed by solving the elastodynamic wave equation, considering an earth model with spatial distributions of density and elastic properties. Several solution techniques may be used, dependent on the model complexity. Two of the most used solution methods are Finite Difference (FD) techniques and Asymptotic Ray Theory (ART).

The ART solutions are fast, but less applicable to complicated and strongly heterogeneous materials. The main advantage of the FD techniques over ART solvers is the ability to simulate wave propagation in

* Corresponding author.
E-mail address: HelmerAndre.Friis@rf.no (H.A. Friis).

PII: S0168-9274(01)00096-4

strongly heterogeneous materials. Increasing the spatial resolution of the reservoir, e.g., due to hypotheses of spatial variabilities of, e.g., lithology, porosity, permeability and pore fluid properties, may lead to the associated seismic models being highly heterogeneous. Increasing the spatial resolution may also yield the necessity of incorporating higher order material complexities of the elastic properties, i.e., to include seismic velocity anisotropy. Acoustic experiments are also performed in boreholes and in the laboratory in order to retrieve the internal properties of rocks. In such cases the illumination frequencies are in the sonic to ultrasonic range, which may infer the need to apply the more complicated elastodynamic relations of poro-elastic materials.

The development of computer codes for FD simulation of elastic wave problems is no trivial task. Depending on the problem complexity, a traditional program development within this area may be rather time consuming, and an existing program may be hard to adapt to new problems. One of the main reasons for this is that various concerns (e.g., discretization methods, coordinate systems, type of geophysical model) usually are mixed together in the program structure, allowing little reuse of code. For much of the same reason, it is also a time consuming task to obtain a parallel program version. It would definitely be advantageous to have a faster program development process in order to study more complicated wave modeling problems, and to port these programs onto high performance computers with little additional effort. In this paper we propose one such programming approach which is clearly superior to traditional programming approaches, and show how it applies to elastic wave simulation. Our methodology is heavily based on the notion of tensors and on formal methods from computer science.

The theory of differential equations have diverged in two different directions in the 20th century; the applied mathematical coordinate based presentation which has its origin in hand calculations, and the pure mathematical presentation based on coordinate-free formulations, commutative diagrams and global analysis. Here we want to show that much can be gained by employing 'abstract' coordinate-free formulations also for seismic modeling applications.

The origin of this line of work was the design of a software system for numerical solution of tensor field equations on parallel computers using formal methods from pure computer science [1,14,17,20]. It was discovered that the pure abstract mathematical definitions were better suited for software design than the concrete coordinate based definitions, and that modern computer languages enable us to express and program various numerical algorithms directly within a coordinate-free language. The basis for this is the recognition that "Computer programming is *what* + *how*", i.e., we should keep in mind the difference between an *abstract specification* of tasks to be done and the actual algorithms/data structures used in a *concrete implementation.* The abstract specification defines the external interfaces of software modules, and the concrete implementation is hidden within the modules, and thus not accessible to the rest of the world. The main benefits are modularity (i.e., change of algorithms/data structures within a given module without doing anything outside). Thus we can easily change, e.g., discretization technique, coordinate system, type of geophysical properties (isotropic vs. anisotropic), with a program development methodology based on these concepts. Using the concepts of coordinate-free mathematics for numerical problems is termed *coordinate-free numerics* and we have based the Sophus software library (see Appendix A) on these notions.

Here we show the approach of coordinate-free numerics applied to various problems in elastic wave modeling. We demonstrate that elastic wave modeling problems differing in, e.g., material complexity and boundary conditions, can all be formulated within the same framework, having almost all of their code in common. The various program versions can also easily be parallelized with only minor additional effort. The main benefits of our methodology are thus the ability to experiment with wave propagation

models and their corresponding numerical solution strategies at a much lower cost compared to traditional approaches.

This paper is organized as follows: Section 2 gives an introduction to the concepts of coordinate-free numerics. The next section sketches how these notions have been implemented in the Sophus software library for a collection of problems in elastic wave modeling. In Section 4 we give some numerical examples demonstrating the capabilities of this abstract programming approach. In particular, we focus on modeling of elastic waves in isotropic and anisotropic media, waves in boreholes, and waves in poro-elastic materials. Finally, we give our main conclusions in Section 5.

2. Concept of coordinate-free numerics

A solver for a computational model described by a partial differential equation (PDE) must embody the discretization method, the coordinate system, as well as the actual solver algorithm with time integration. Each of these are complex by themselves, and keeping track of their interaction when developing computational software is an overwhelming and error-prone task. Techniques to improve stability, accuracy and convergence of the solver further increase this complexity. Using abstract mathematical notions to control this complexity was suggested in [20]. Specifically we focused on the use of coordinate-free mathematics as guidelines in the structuring of numerical software, yielding a notion of coordinate-free numerics. Coordinate-free mathematics is based on the notion of tensors, which is much used, e.g., in theoretical physics. An introduction can be found in [22]. How to combine these notions with the aim of developing flexible software is studied in [14,17], and the case studies [13,16] present the benefits of using coordinate-free numerics.

The idea underlying coordinate-free mathematics can be explained by scrutinizing the concepts of a vector. Traditionally a vector is considered as numbers in an array structure with one index. The numbers represent quantities of the physical phenomena we are studying, and they have to be given concrete values, measured in some coordinate system with some unit of measurement. Hence the choice of coordinate system will influence the formulation of the physical laws. This is not an ideal situation, since there does not exist a coordinate system being ideally suited for all purposes. (Some physical domains are of such a nature that there does not even exist a universal coordinate system covering all of the domain.) In coordinate-free mathematics, vectors, as well as scalars and other notions, are considered as algebraic structures which are characterized by the operations that may be applied on them and the laws these operations obey, not by their values in a coordinate system.

From the abstract viewpoint we identify the notions *scalar space* and *vector space*. A scalar space has two associative binary operations termed addition and multiplication, the first of which is commutative, both with neutral elements, zero and one respectively, such that multiplication distributes over addition. A vector space over a given scalar space has a commutative, associative operation (vector addition) with a neutral element (the zero vector), and a scalar multiplication (multiplication of scalars with vectors) which distributes over vector addition.

We then introduce the notion of linear mappings (matrices), which may map a vector value from a vector space into another. Simple observations now show that the scalar spaces themselves may be considered as vector spaces, and also that all linear mappings between two vector spaces themselves form a vector space. But, as in the traditional coordinate-dependent approach, we still know that the “proper” vectors we started with could be characterized using arrays with one index, while linear mappings are

characterized using 2 indices. However, since linear mappings form a vector space, we also have linear mappings between linear mappings, and so forth. Such structures may require 4 or more indices, while the scalar space is characterized with 0 indices.

We then form the (coordinate-free) notion of tensor spaces T_n as a generalization of these vector space notions, where n, for $0 \leqslant n$, denotes the rank of the tensor, 0 for scalars, 1 for vectors, 2 for matrices, etc. The algebra of tensors has operations $\otimes : T_m \times T_n \to T_{n+m}$ for the outer product between tensors, $@ : T_{n+k} \times T_k \to T_n$ for applying a tensor as a linear mapping to another tensor, $\circ : T_{m+k} \times T_{k+n} \to T_{m+n}$ for composition of tensors as linear mappings, $+ : T_n \times T_n \to T_n$ for addition of tensors (where $\otimes$, $@$ and $\circ$ distribute over $+$), and so forth. If one of the arguments to the multiplication operators $\otimes$, $@$ or $\circ$ is from T_0, i.e., is a scalar space, these operations will coincide, and we will normally drop the multiplication symbol. The operations on tensors are defined independently of any coordinate system, and hence all our mathematical formulas and solver codes built on top of these notions will be coordinate-free. We still need the coordinate representations to implement the tensors, but the coordinate system will no longer interfere with our description of the problem and its solution.

The physical phenomena we want to study are not characterized by simple scalar and vector values. We need the notion of values that change continuously across the physical domain, the *manifold*, we are studying. Such values are called scalar fields (and vector fields etc.), and can be considered as continuous functions from the spatial coordinates into scalar values (vector values etc.). Now a scalar field satisfies all the properties of being a scalar space, and thus we may build vector fields and tensor fields just by using the appropriate scalar field as the underlying scalar space. With the notion of scalar field also comes the notion of derivatives, in particular the partial derivatives along the vectors spanning out the physical domain. Likewise, we get differentiation operators on the tensor fields. These include the gradient $\nabla : T_n \to T_{n+1}$, the divergence $\nabla\cdot : T_{n+1} \to T_n$ and the Lie derivative $\mathcal{L}_{\boldsymbol{u}} : T_n - T_n$ for every vector $\boldsymbol{u} \in T_1$. The tensor field derivatives may be implemented using the partial derivatives of the underlying scalar field. This allows us to implement the discretization method only in the scalar field, thus effectively decoupling the interaction between coordinate system and discretization method.

We have implemented the notions from coordinate-free numerics in the Sophus software library. See Appendix A for a brief overview. Using this library we may freely change discretization methods by choosing appropriate scalar field implementations, change coordinate systems (for instance, between Cartesian and axisymmetric) by choosing appropriate tensor implementations, and implement various solvers independently of these choices by formulating the algorithms using the tensor notation. This is made possible by ensuring that all implementations have the same interface, with operators satisfying the same algebraic laws, independently of what data structures and algorithms are chosen.

3. Implementation for elastic wave modeling

Here we will demonstrate how the programming concepts, described in the previous section, were applied in order to develop a set of programs for elastic wave simulation. Our focus will be on program structure and not on specific details connected to the numerical implementation. Due to the advanced structure of the Sophus library, these wave simulation programs share most of their components, and since the components are highly independent, we obtain a very flexible program development process.

We consider the elastic wave equation on a domain Ω, written in a coordinate-free form as

$$\rho \frac{\partial^2 \boldsymbol{u}}{\partial t^2} = \nabla \cdot \sigma + \boldsymbol{f}(t), \qquad \sigma = \Lambda @ e, \quad e = \mathcal{L}_{\boldsymbol{u}}(g). \tag{1}$$

Here $\boldsymbol{u} \in T_1$ is the particle displacement vector, $\sigma \in T_2$ and $e \in T_2$ are the stress and strain tensors, respectively, $\rho \in T_0$ is the material density, and $\Lambda \in T_4$ is the stiffness tensor (Hooke's tensor), describing the elasticity of the material. We note that due to physical properties of the problem, σ, e and Λ exhibit several symmetry properties. Furthermore, $\boldsymbol{f} \in T_1$ is the external source, $g \in T_2$ is the metric tensor and $\mathcal{L}_{\boldsymbol{u}}$ is the Lie derivate along $\boldsymbol{u}$. The coordinate-free formulation (1) contains a large set of practically interesting wave propagation problems ranging from large to small scale seismics with various material models. It may be subject to various boundary conditions depending on the concrete problem to be solved.

In order to develop programs for different types of elastic wave modeling problems, we decided on the following main software modules within the framework of the Sophus library (see Appendix A):

- MANIFOLDS AND SCALAR FIELDS
  - TN: classes defining the notion of an n-dimensional torus manifold (domain information), the point type on a torus manifold, and the scalar field on a torus (no boundaries).
  - BN: classes defining the extent of an n-dimensional bounded submanifold of $\mathbb{R}^n$ (domain information), the point type on a bounded submanifold of $\mathbb{R}^n$, and the scalar field on a bounded manifold.
- TENSOR: classes defining tensors with multiplication, addition, differentiation, etc., operators.
- SOLVERS
  - SEISMOD: the solver algorithm in coordinate-free form.

Now consider solving Eq. (1) numerically using the finite difference method. The various tasks that are required for the implementation of an FD discretization may now be separated into the above modules. The time discretization is naturally placed in the SEISMOD module. Using the abstract language of the Sophus library, the time stepping algorithm may be written in the following way, when using a second order accurate centered finite difference scheme, given Lambda (the stiffness tensor containing the geophysical material properties) and initial values for *unm*1 and *un* (displacement vectors):

```
WHILE (t < stoptime)
  e = Lie(un, g);
  sigma = apply(Lambda, e);
  a = div(sigma)/rho + fsource;
  unp1 = 2 * un − unm1 + a * (Δt * Δt);
  unm1 = un;
  un = unp1;
END WHILE
```
(2)

where Δt is the time step and *unm*1, *un* and *unp*1 is the displacement vector at the time step level $n-1$, n and $n+1$, respectively. Furthermore, *sigma* is the stress tensor, *Lie* is the Lie derivative operation, *apply* is the tensorial application operation, *div* is the divergence operation, *rho* is the material density and *fsource* is the external source. The remaining symbols used in (2) should be evident from Eq. (1).

The basic algorithm (2) is only slightly expanded with the introduction of appropriate boundary conditions. The tensorial operations which must be coded in the TENSOR module includes the Lie

derivation, the tensor application and the divergence operator. In practice these operations must be expressed in a given coordinate system inside the TENSOR module. This is discussed in more detail in Section 3.1.

The stiffness tensor is generally a fourth rank tensor T_4, i.e., $\Lambda = c_{ijkl}$, but has at most 21 independent elastic coefficients in case of *full anisotropy* in 3D. In the following, we use the Voigt notation [3] for the elastic stiffness tensor. This gives an array C with two indices I (which encodes the index pair ij) and J (which encodes the index pair kl). In 3D, when i, j, k and l range from 1 to 3, the encoded indices I and J range from 1 to 6, as given by the following conversion table.

$$\begin{array}{cccccccc} ij \text{ or } kl: & 11 & 22 & 33 & 32=23 & 31=13 & 12=21 \\ \updownarrow \qquad \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\ I \qquad\quad J & 1 & 2 & 3 & 4 & 5 & 6 \end{array} \tag{3}$$

In other words, $C_{11} = c_{1111}$, $C_{12} = c_{1122}$, $C_{13} = c_{1133}$, $C_{14} = c_{1123} = c_{1132}$, $C_{15} = c_{1113} = c_{1131}$, $C_{16} = c_{1112} = c_{1121}$, and so forth. The encoded matrix C is symmetric and thus has at most 21 independent components.

The simplest fully elastic medium is isotropic, and given by two distinct, non-zero coefficients only (see Eq. (5)). Most sandstones, e.g., reservoir sands, and igneous rocks are considered to be isotropic. Transverse isotropic (TI) materials have 5 independent elastic coefficients, and they have one axis of rotational elastic symmetry. For a TI material with an axis of symmetry parallel to the z-axis, the following elastic stiffnesses are nonzero:

$$C_{11} = C_{22},\ C_{33}; \qquad C_{12},\ C_{13} = C_{23}; \qquad C_{44} = C_{55}; \qquad C_{66} = \tfrac{1}{2}(C_{11} - C_{12}). \tag{4}$$

Shales and materials with an internal layering/bedding are considered to behave transversely isotropic, with an axis of symmetry perpendicular to the layering plane. Accurate modeling of shales is important both due to its occurrence as a source rock for hydrocarbons, a cap rock preventing the hydrocarbons to migrate into overburden formations, or as infill material in the reservoir sand. We see that the stiffness tensor has different forms depending on the type of properties of the medium, s.a. isotropy versus transverse isotropies. Programs that solve Eq. (1) for various types of material properties need only differ in the TENSOR module, see Section 3.2.

It is clear that the differential operators at the TENSOR level may be expressed by combinations of partial derivatives. In the Sophus library the partial differential operators are implemented at the scalar field level. The actual FDM discretization is thus confined to a separate module, in this case the TN module. The corresponding numerical grid is implemented at the discrete level in the MESH module.

Boundary conditions may in principle be handled in two ways, either by extending the PDE with extra equations defining the boundary properties, or by having scalar field operators handle boundary properties. The latter is common for FD methods, and in our case the BN module is a boundary sensitive scalar field using the TN module for its implementation. The PDE handler SEISMOD must be upgraded to call the boundary correction operations, which have to be passed down to the BN module via the tensor abstraction. A change in the boundary conditions then imply a small change in the BN module, and perhaps some minor alterations in the SEISMOD and TENSOR modules to activate the appropriate handlers.

3.1. Implementing for different coordinate systems

Eq. (1) is valid independently of the choice of coordinate system. Coordinate systems are handled at the tensor level, i.e., in the TENSOR implementation, in this way keeping the implementation of the scalar field independent of such choices. There are in principle two ways to handle arbitrary coordinate systems. The most general approach is to use the metric tensor $g \in T_2$ [22] which encodes the information about the chosen coordinate system. Then one may use any coordinate system by changing the data for the metric tensor appropriately. Alternatively, one may tailor the implementation of TENSOR to the chosen coordinate system. This requires reprogramming when a new coordinate system is to be used, but normally much better run-time efficiency, as special properties of the coordinate systems may be exploited.

In the common case of 3D Cartesian coordinates x, y and z, the derivation operations divergence $\nabla\cdot$ and Lie derivative $\mathcal{L}$ simplify dramatically from the general case. The divergence $\boldsymbol{w} = \nabla\cdot\sigma$ on symmetric $\sigma \in T_2$ and result $\boldsymbol{w} \in T_1$, is encoded by operations on their scalar field components,

$$
\begin{aligned}
w_x &= \frac{\partial\sigma_{xx}}{\partial x} + \frac{\partial\sigma_{xy}}{\partial y} + \frac{\partial\sigma_{zx}}{\partial z},\\
w_y &= \frac{\partial\sigma_{xy}}{\partial x} + \frac{\partial\sigma_{yy}}{\partial y} + \frac{\partial\sigma_{zy}}{\partial z},\\
w_z &= \frac{\partial\sigma_{zx}}{\partial x} + \frac{\partial\sigma_{zy}}{\partial y} + \frac{\partial\sigma_{zz}}{\partial z}.
\end{aligned}
$$

The Lie derivative $e = \mathcal{L}_{\boldsymbol{u}}(g)$ on identity $g \in T_2$ (for Cartesian geometry) and vector $\boldsymbol{u} \in T_1$ giving the symmetric tensor $e \in T_2$ as result, is encoded by

$$
\begin{aligned}
e_{xx} &= \frac{\partial u_x}{\partial x}, \qquad e_{yy} = \frac{\partial u_y}{\partial y}, \qquad e_{zz} = \frac{\partial u_z}{\partial z},\\
e_{xy} &= e_{yx} = \frac{1}{2}\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right),\\
e_{xz} &= e_{zx} = \frac{1}{2}\left(\frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x}\right),\\
e_{yz} &= e_{zy} = \frac{1}{2}\left(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\right).
\end{aligned}
$$

All these expressions are easily implemented using the partial derivative and arithmetic scalar field operations. Using, e.g., cylindrical coordinates r, θ and z, will change these formulations, but the affected code in TENSOR is small. Thus the cost of changing coordinate system is small, even if reimplementation of the TENSOR code is needed.

In the important case of retrieving formation parameters from acoustic measurements in boreholes, we will often have axial symmetry. Using cylindrical coordinates we may easily exploit this fact in our tensor implementation. The vector $\boldsymbol{w} \in T_1$ may then be reduced to a 2D data set with only r and z directions, as it will be constant in the angular θ direction. The divergence $\boldsymbol{w} = \nabla\cdot\sigma$ on symmetric $\sigma \in T_2$ and result $\boldsymbol{w} \in T_1$ is now encoded by

$$
w_r = \frac{\partial\sigma_{rr}}{\partial r} + \frac{\partial\sigma_{zr}}{\partial z} + \frac{\sigma_{rr}}{r} - \frac{\sigma_{\theta\theta}}{r}, \qquad w_z = \frac{\partial\sigma_{zz}}{\partial z} + \frac{\partial\sigma_{rz}}{\partial r} + \frac{\sigma_{rz}}{r},
$$

where r is a scalar field measuring the distance from the axis. The Lie derivative $e = \mathcal{L}_{\boldsymbol{u}}(g)$ on $g \in T_2$ (for cylindrical geometry), vector $\boldsymbol{u} \in T_1$ and tensor $e \in T_2$ with only four distinct, non-zero components as result, may be encoded by

$$e_{rr} = \frac{\partial u_r}{\partial r}, \qquad e_{\theta\theta} = \frac{u_r}{r}, \qquad e_{rz} = e_{zr} = \frac{1}{2}\left(\frac{\partial u_z}{\partial r} + \frac{\partial u_r}{\partial z}\right), \qquad e_{zz} = \frac{\partial u_z}{\partial z}.$$

Some of the terms for the divergence and Lie derivative apparently have singularities at the axis where r is 0, but the PDE itself is not singular at these points, and the problem may be cured by using, e.g., L'Hospital's rule, yielding special formulations at the axis,

$$w_r = 2\frac{\partial \sigma_{rr}}{\partial r} + \frac{\partial \sigma_{zr}}{\partial z} - \frac{\partial \sigma_{\theta\theta}}{\partial r}, \qquad w_z = \frac{\partial \sigma_{zz}}{\partial z} + 2\frac{\partial \sigma_{rz}}{\partial r}, \qquad e_{\theta\theta} = e_{rr}.$$

Again we are using the scalar field operators from the BN module, and these are implemented independently of coordinate systems.

It must be emphasized that the creation of a new TENSOR version from an already existing one (e.g., moving from Cartesian to cylindrical coordinates) normally requires only a very limited amount of work, since the operations at the TENSOR level are implemented using relatively few code lines. Now having different TENSOR versions for different coordinate systems, the use of a particular coordinate system in a program is only a matter of program configuration, see [16].

3.2. Implementing for different types of material properties

Eq. (1) is valid independently of the elastic complexity of the materials considered, as the particular kind of material properties is defined through the stiffness tensor. In our approach, the implementation of the stiffness tensor is confined to the TENSOR module. We will consider three material models which covers a wide range of practically interesting cases. As discussed above, the stiffness tensor in 3D consists at most of 21 independent coefficients. In case of isotropy, there are only two independent coefficients, and the stiffness tensor is thus completely constructed from the Lame constant λ and the shear modulus μ (where $\lambda = C_{33} - 2C_{44}$ and $\mu = C_{44}$). In a Cartesian coordinate system, the stress–strain relation $\sigma = \Lambda \,@\, e$ in Eq. (1), is then written as, using the Voigt notation and recalling that both the stress tensor $\sigma \in T_2$ and the strain tensor $e \in T_2$ are symmetric,

$$\begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{zy} \\ \sigma_{zx} \\ \sigma_{xy} \end{pmatrix} = \begin{pmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{pmatrix} \begin{pmatrix} e_{xx} \\ e_{yy} \\ e_{zz} \\ 2e_{zy} \\ 2e_{zx} \\ 2e_{xy} \end{pmatrix}. \tag{5}$$

Considering a TI material with an axis of elastic symmetry parallel to the z-axis, the stress–strain relation reads

$$\begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{zy} \\ \sigma_{zx} \\ \sigma_{xy} \end{pmatrix} = \begin{pmatrix} C_{11} & C_{11}-2C_{66} & C_{13} & 0 & 0 & 0 \\ C_{11}-2C_{66} & C_{11} & C_{13} & 0 & 0 & 0 \\ C_{13} & C_{13} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{pmatrix} \begin{pmatrix} e_{xx} \\ e_{yy} \\ e_{zz} \\ 2e_{zy} \\ 2e_{zx} \\ 2e_{xy} \end{pmatrix}. \tag{6}$$

Both Eqs. (5) and (6) are easily implemented in the TENSOR module using operations provided by the BN module, since the different terms in these equations are scalar field expressions. Similar stress–strain relations may also be written for other coordinate systems (e.g., cylindrical coordinates).

Finally, we consider a TI medium with an arbitrary axis of symmetry (TIA) (i.e., the direction of the axis of symmetry is itself a vector field). This case is very close to the case of full anisotropic media. The representation of Hooke's law will therefore contain a dense matrix (Hooke's tensor) with coefficients dependent on the space coordinates. Instead of expressing this equation in the global coordinate system, it is more convenient to use a local coordinate system with z-axis parallel to the local axis of symmetry at a given point. Transition matrices $P \in T_2$, very similar to the metric tensor concept, encode the coordinate geometry and are used to translate between the tensor components in the local and global coordinate systems. In the local coordinate system, Hookes law is expressed by Eq. (6). Now, let the quantities with a prime represent the tensor components in the local coordinate system. The basic algorithm for modeling stress–strain relations then is:

$$e' = P^{-1} \circ e \circ P, \qquad \sigma' = \Lambda' @ e', \qquad \sigma = P \circ \sigma' \circ P^{-1}. \tag{7}$$

Here P is the transition matrix from the local coordinate basis to the basis of the global coordinate system. (Notice that P actually may change from point to point in the physical model.) $\sigma' = \Lambda' @ e'$ represents Hooke's law in the local coordinate system. It has precisely the same structure as Eq. (6). The Iso, TI and TIA models are all implemented in the Sophus library. Choosing between the different versions is only a matter of program configuration.

3.3. Implementing for poro-elastic wave propagation problems

Finally, we show that our approach also easily allows other types of wave propagation problems to be implemented within the same framework. As an example, we here consider the modeling of elastic waves in poro-elastic materials. Poro-elastic materials consist of, at least, one solid component and one fluid component. Porous fluid-filled rocks are a class of such materials. The theory of elasticity in this case implies that the solid and the fluid may have different displacement vectors, which leads to a viscous drag between the fluid and the solid. The viscous drag causes energy dissipation to occur, resulting in wave energy attenuation. The theoretical foundations were originally formulated in [5,6], but have been discussed in several papers since, among these [7,19].

Poro-elastic relations are specifically applicable for studying details of the wave propagation field in reservoir rocks. Now, the motions of the solid and fluid are defined through a coupled set of differential equations, which, in a coordinate-free formulation, are

$$\begin{aligned}
&\rho_{11}\frac{\partial^2 \boldsymbol{u}}{\partial t^2} + \rho_{12}\frac{\partial^2 \boldsymbol{U}}{\partial t^2} = \nabla \cdot \sigma + (1-q)\boldsymbol{f}(t) + b @ \left(\frac{\partial \boldsymbol{U}}{\partial t} - \frac{\partial \boldsymbol{u}}{\partial t}\right),\\
&\rho_{12}\frac{\partial^2 \boldsymbol{u}}{\partial t^2} + \rho_{22}\frac{\partial^2 \boldsymbol{U}}{\partial t^2} = \nabla s \cdot q\boldsymbol{f}(t) - b @ \left(\frac{\partial \boldsymbol{U}}{\partial t} - \frac{\partial \boldsymbol{u}}{\partial t}\right),\\
&\sigma = \Lambda @ e + Q \otimes \tau,\\
&s = Q @ e + R \otimes \tau,\\
&e = \mathcal{L}_{\boldsymbol{u}}(g), \quad \tau = \nabla \cdot \boldsymbol{U}.
\end{aligned} \tag{8}$$

Here $\boldsymbol{u} \in T_1$ and $\boldsymbol{U} \in T_1$ are the particle displacement vectors in the solid and fluid parts, respectively, $\sigma \in T_2$ and $e \in T_2$ are the stress and strain tensors in the solid part, whereas $s \in T_0$ and $\tau \in T_0$ represent

the pressure and strain in the fluid part. Furthermore, $\Lambda \in T_4$, $R \in T_0$ and $Q \in T_2$ are stiffness tensors representing the stiffness of the solid and fluid parts and their interaction, respectively. $\rho_{11}, \rho_{12}, \rho_{22} \in T_0$ are dynamic density coefficients and $b \in T_2$ is a second rank tensor representing the viscous drag between the fluid and solid parts. The quantities R, Q, ρ_{11}, ρ_{12}, ρ_{22} and b are all defined in [5]. Finally, $\boldsymbol{f} \in T_1$ is the external source, $g \in T_2$ is the metric tensor and $\mathcal{L}_{\boldsymbol{u}}$ is the Lie derivate along $\boldsymbol{u}$, ∇ is the gradient operator and $\nabla\cdot$ is the divergence operator, q is an arbitrary scalar quantity ($0 < q < 1$) which determines the force distribution, i.e., the energy splitting between the solid and fluid part. A natural choice is to put q equal to the porosity in place, but other choices may give more transparent wave fields for certain wave components.

The implementation of Eq. (8) follows the same lines as for Eq. (1). Basically, we need to write a new SEISMOD module for Eq. (8) and the chosen time stepping scheme. We thus observe that in the coordinate-free numerics approach, the difference between implementing (1) and (8) (which are quite different PDEs) lays in one single module (SEISMOD), a relatively small module containing the abstract solver algorithm. Different poro-elastic program versions for different coordinate systems and material complexity are now obtained using exactly the same principles as explained in Sections 3.1 and 3.2. This clearly demonstrates the advantages of this abstract programming approach.

4. Numerical examples

In this section we demonstrate the usefulness of our so called coordinate-free numerics approach in elastic wave simulation, by applying it to a range of different wave propagation problems all relevant to exploration geophysics. We emphasize that the different computer codes used in the examples, which require, e.g., different material complexity and PDEs, usually would have implied the need for rather different computer codes using a traditional programming approach. In the examples below, we first focus on the modeling of the seismic responses of a gas–oil reservoir where the overlaying rocks are anisotropic. Secondly, we model the acoustic wavefield within and in the vicinity of a fluid-filled borehole. Finally, we apply our system for simulation of the seismic wavefield within porous fluid-filled materials using the relations describing poro-elasticity.

The PDE's above are all solved by an FD method on a staggered grid. The spatial finite difference stencil are very close to an 8-point centered stencil of Holberg type [18], and the numerical treatment of the absorbing boundary conditions are discussed in [24], which is a modification of the technique presented by [9]. The equations are integrated in time by a standard second order accurate centered finite difference approximation.

4.1. Seismic reflections from hydrocarbon reservoirs with anisotropic overburden

Our first example focuses on the modeling of P and S wave data from a hydrocarbon reservoir model, where the overlaying formation occur both isotropic and anisotropic. In case of anisotropy we limit ourselves to TI media, which may be described using the stiffness coefficients of Eq. (4). Though, here we use the vertical P and S wave velocities (i.e., $V_{\mathrm{Pv}} = \sqrt{C_{33}/\rho}$ and $V_{\mathrm{Sv}} = \sqrt{C_{44}/\rho}$), together with the Thomsen parameters [25], which read

$$\varepsilon = \frac{C_{11} - C_{33}}{2C_{33}}, \qquad \gamma = \frac{C_{66} - C_{44}}{2C_{44}}, \qquad \delta = \frac{(C_{13} + C_{44})^2 - (C_{33} - C_{44})^2}{2C_{33}(C_{33} - C_{44})}. \tag{9}$$

In case of isotropy these three parameters vanish.

Table 1 defines a model of plane and horizontal layers defined by their vertical velocities, the average density and the Thomsen parameters. Layers 6, 7 and 8 contain gas (G), oil (O) and water (W), respectively, and we define them as the target zone for our modeling. Their seismic properties (V_p and V_s) were defined using a differential effective medium model [4,23] for the dry rock frame properties and computing the fluid effect using the Gassmann equation [11].

Table 2 defines the dry rock frame and fluid properties of the target zone layers. Here ϕ denotes the fractional porosity, K_f the bulk modulus of the pore fluid, ρ_f the fluid density, K_s the bulk modulus of the rock frame, μ_s the shear modulus of the rock frame and ρ_s is the grain density.

The explosion source is situated 20 m below the surface of the upper layer, which is defined by the physical properties of water. Thus, the source generates P waves only. In order to also record energy from S waves (P-to-S converted waves), the geophones are placed on the top of layer 2 (the ocean bottom). The receiver line consists of 31 geophones with the first geophone situated vertically below the source point and the remaining geophones are equispaced 100 m along the line.

In the following example we compare modeled seismic data considering both the anisotropic overburden (layers 3–5) in Table 1, and an isotropic overburden where the according Thomsen parameters are set to zero.

Fig. 1 shows the modeled seismograms of the horizontal and vertical displacements, particularly focused on the PP reflections from the target zone for the two overburden models. Furthermore, kinematic

Table 1
Seismic model parameters (isotropic and anisotropic)

Layer	Depth	V_{Pv}	V_{Sv}	ρ	ε	δ	γ
1	0.000	1.481	0.000	1.030	0.000	0.000	0.000
2	0.200	1.921	0.849	1.672	0.000	0.000	0.000
3	0.500	2.300	0.840	2.142	0.250	0.250	0.000
4	0.800	2.461	0.909	2.212	0.250	0.250	0.000
5	1.100	2.609	1.324	2.313	0.250	0.250	0.000
6	1.300	2.422	1.466	1.995	0.000	0.000	0.000
7	1.400	2.630	1.472	2.083	0.000	0.000	0.000
8	1.500	2.840	1.473	2.166	0.000	0.000	0.000

Table 2
Medium parameters of the target zone

	Layer	ϕ	K_f	ρ_f	K_s	μ_s	ρ_s
Gas zone	6	0.25	0.067	0.172	20.00	10.00	2.550
Oil zone	7	0.25	0.887	0.682	20.00	10.00	2.550
Brine zone	8	0.25	2.773	1.014	20.00	10.00	2.550

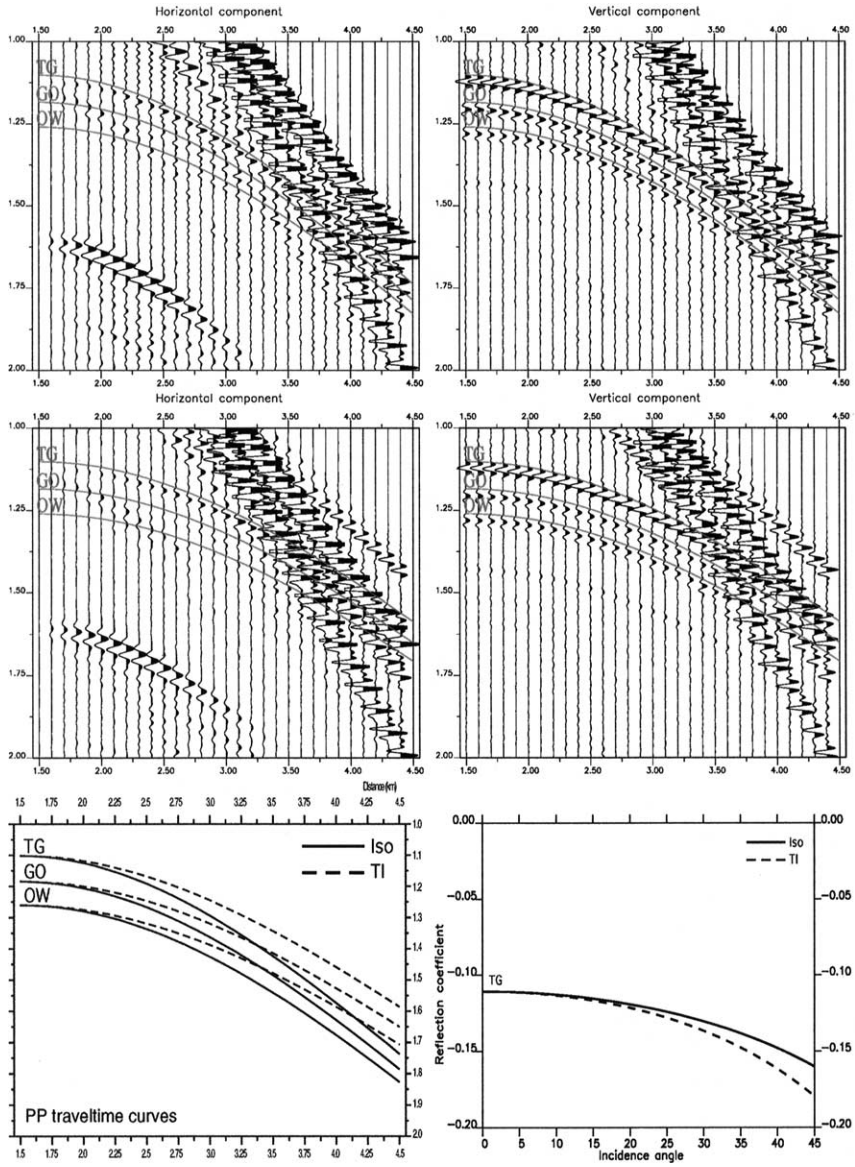

Fig. 1. P–P reflection properties from target zone of the geomodels defined in Table 1. Superimposed on the synthetic seismics are traveltime curves representing the TG, GO and OW reflectors. Top row: synthetic seismograms from isotropic model. Middle row: synthetic seismograms from TI model. Bottom row: traveltime curves (left) and theoretical reflection coefficients of the TG boundary (right), where solid curves are derived from isotropic model and dashed curves from anisotropic model. The horizontal and vertical axes are in kilometers and seconds, respectively (except in the bottom right subfigure).

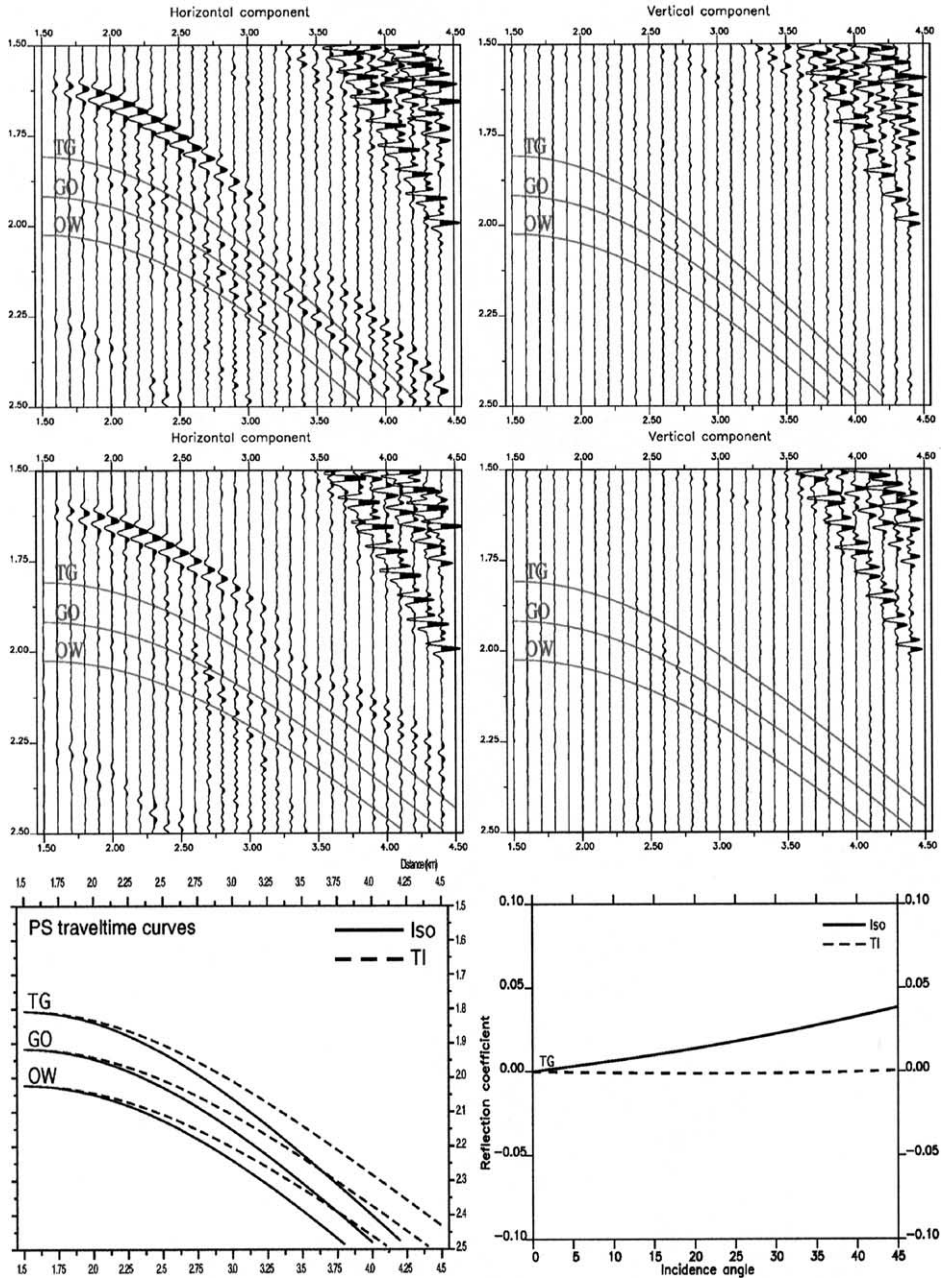

Fig. 2. P–S reflection properties from target zone of the geomodels defined in Table 1. Superimposed on the synthetic seismics are traveltime curves representing the TG, GO and OW reflectors. Top row: synthetic seismograms from isotropic model. Middle row: synthetic seismograms from TI model. Bottom row: traveltime curves (left) and theoretical reflection coefficients of the TG boundary (right), where solid curves are derived from isotropic model and dashed curves from anisotropic model. The horizontal and vertical axes are in kilometers and seconds, respectively (except in the bottom right subfigure).

data, i.e., traveltime curves, for the top gas (TG) reflector together with the gas–oil (GO) and oil–water (OW) contact interfaces, have been modeled separately using NORSAR ARM software [21], and are shown both individually and superimposed on the seismograms. The plots clearly reveal the relative strong impact of the TI model on the normal moveout (NMO) of the reflections. Also seen is how the P–P events mainly occur on the vertical displacement component for near normal incidence waves. Though, the P waves become more exposed on the horizontal displacement component as the incidence angle increases. Another important seismic attribute to consider is the characteristics of the reflection amplitude versus offset (AVO) of the target reflectors [2,8].

The bottom right subfigure in Fig. 1 shows the theoretically derived AVO curves of the TG reflector for both the isotropic and the anisotropic case using the equations given in [12]. For the chosen anisotropy model the differences in AVO are seen to be quite small and both curves show increased absolute value of the reflection coefficient with increasing incidence angle.

The similar data focusing on the P–S conversions are shown in Fig. 2. Here we consider the P-to-S conversions occurring at the reflection boundary, i.e., a downgoing P wave being reflected as an upgoing S wave. These events are much weaker than the P–P waves, and also, no P-to-S wave conversions occur for normal incident P waves. Accordingly, the P–S events are more strongly exposed on the horizontal component. As seen in the figure, the P-to-S conversions occur to be very weak in this example. This is also revealed by the modeled AVO curves of the TG interface. While the AVO of the anisotropic model is constant and close to zero, the isotropic model gives a slight increase in amplitude with offset. This is further revealed in the horizontal component of the modeled data.

4.2. Acoustic waves in boreholes

Acoustic surveying in boreholes is performed for measuring P and S wave velocities on a scale of a few decimeters. The velocities are derived from P and S head waves traveling along the borehole wall, and mainly occurring as refractions from P waves incident to the borehole wall from the fluid. Thus, nowadays also the use of dipole sources may generate S waves directly in the borehole wall in cases of slow formations, i.e., when the S velocity in the formation is less than the P velocity of the borehole fluid, which unables the generation of P–S refractions.

Modeling of the accoustic wavefield within the borehole may be used to enhance information about formation properties in the close vicinity to the borehole. In this process, modeled data are, in some manner, compared to full waveform acoustic logs. In particular, such studies can reveal local heterogeneities and cracked zones.

In the following example, again in order to better illustrate the various wave modes, we consider a simple model of a fluid-filled borehole within a homogeneous and fully elastic isotropic material. The

Table 3
Seismic parameters of the borehole simulations model

	V_P	V_S	ρ
Borehole	1.500	0.000	1.000
Formation	2.500	1.443	2.200

relevant model parameters, compressional velocity V_P, shear velocity V_S and density ρ are listed in Table 3. Fig. 3 shows three snapshots of the radial (u_r) and the vertical (u_z) displacement components of the wavefield generated from a P wave source placed at the center of the borehole. The borehole wall is located at 0.3 m. Various wavemodes have been indicated in the plots. In particular, the P–S refractions (Rs) are identified specifically by studying the u_r-component; the P–S amplitude in the formation (P–S)

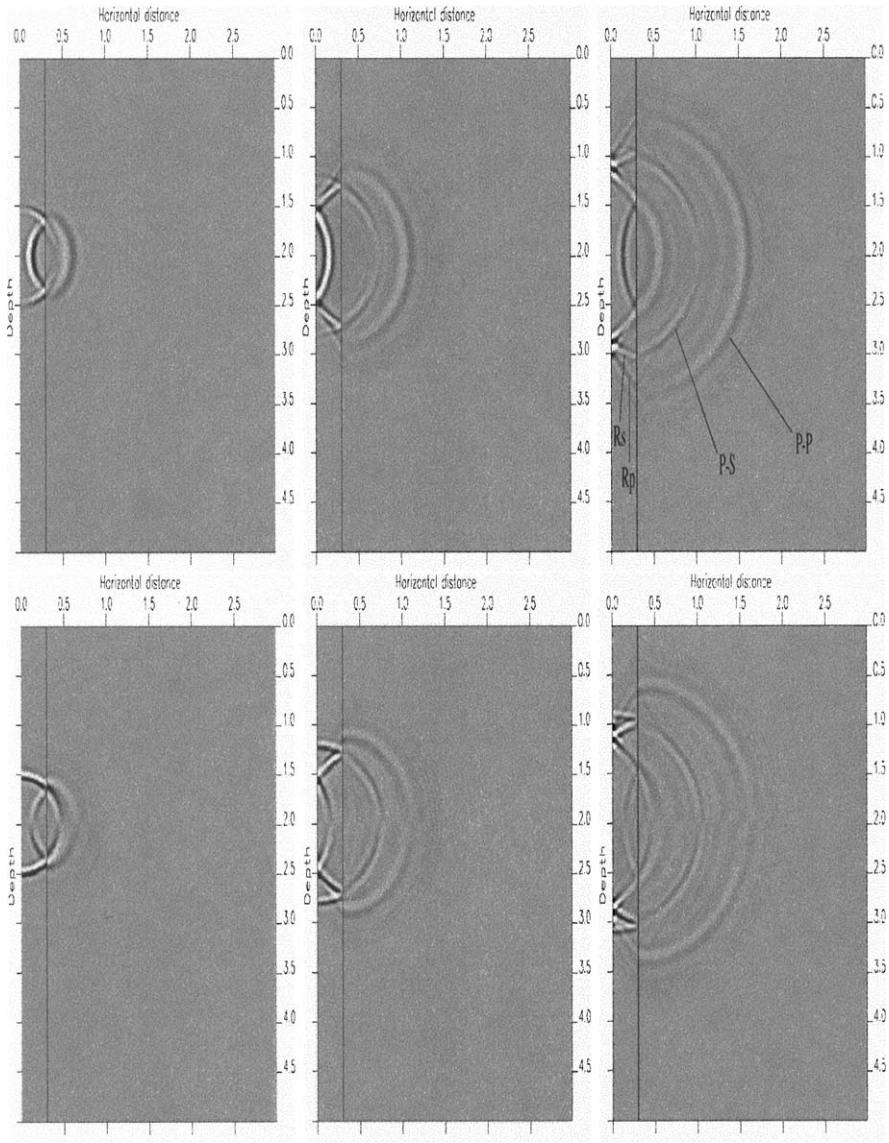

Fig. 3. Snapshots of wave propagation in borehole model at $t = 0.4$, 0.6 and 0.8 msec, radial (u_r) component (top) and vertical (u_z) component (bottom). Horizontal and vertical axis are in meters. The borehole wall is located at 0.30 meters and the source is located at the borehole center, vertically 2.0 m below the top of the model. P–P and P–S are the transmitted P and S waves, whereas Rp and Rs denote the refracted P and S waves, respectively.

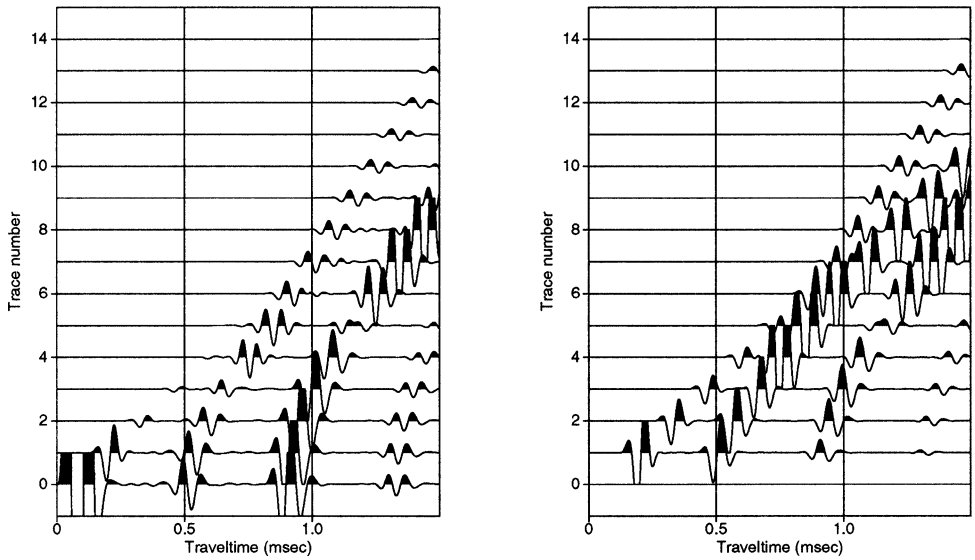

Fig. 4. Synthetic seismograms of the radial (u_r) component (left) and the vertical (u_z) component (right) from the borehole measurements. Receivers are placed regularly (with a distance of 0.2 m apart) along a section of the borehole axis, starting at a depth of 2.0 m and continuing below.

occurs weak along a horizontal line from the source point, while it increases with increasing incidence angle. In Fig. 4, seismograms of the displacement components are shown as they would be observed by a borehole acoustic experiment. Both the reflected and refracted waves are clearly seen in the seismograms.

4.3. Fast and slow P waves in poro-elastic materials

The final example shows simulation results of wave propagation in a two layered model consisting of poro-elastic materials. The layer parameters have been compiled from [10] and they are listed in Table 4. Here V_P is the compressional velocity, V_S is the shear velocity, V_{slow} is the velocity of the slow P wave, ρ_s is the density of the solid phase, ρ_f is the density of the fluid phase, ϕ is the fractional porosity, Q and R are Blot's elastic constants, b represents the resistive damping caused by the fluid motion relative to the solid (the resistive damping in this example is isotropic inside each layer and the tensor b in Eq. (8) is thus described by one coefficient only), and α is the tortuosity. The quantities ρ_{11}, ρ_{12} and ρ_{22} in Eq. (8) are constructed from ρ_s, ρ_f, ϕ and α (see, e.g., [5]). For further description of the model parameters, we refer to [10].

The two poro-elastic layers are both solid-like and an exploding source is placed within the upper layer. This modeling thus implies displacements of both the solid rock (u) and the fluid (U). The P wave source generates P waves in the solid and fluid phase. The P wave propagating along the rock skeleton, but modified by viscous fluid, is denoted the fast P wave. Analogously, the P wave propagating in the pore fluid, but modified by the surrounding solid, is referred to as the slow P wave.

The snapshots of Fig. 5 show the vertical displacements of the solid (u_z) and the fluid (U_z). While the solid displacement is seen to occur mainly as a result of the fast P wave, the fluid displacement is affected by both waves. Particularly is revealed the conversion of the fast P wave to a reflected slow P

Table 4
Seismic parameters of the poro-elastic model (compiled from [10])

Layer	V_P	V_S	V_{slow}	ρ_s	ρ_f	ϕ	Q	R	b	α
1	2.301	1.270	1.087	2.211	1.040	0.010	0.0013	0.0246	1.0	2.0
2	2.386	1.229	0.773	2.650	1.040	0.335	1.3510	0.7285	1680.0	2.0

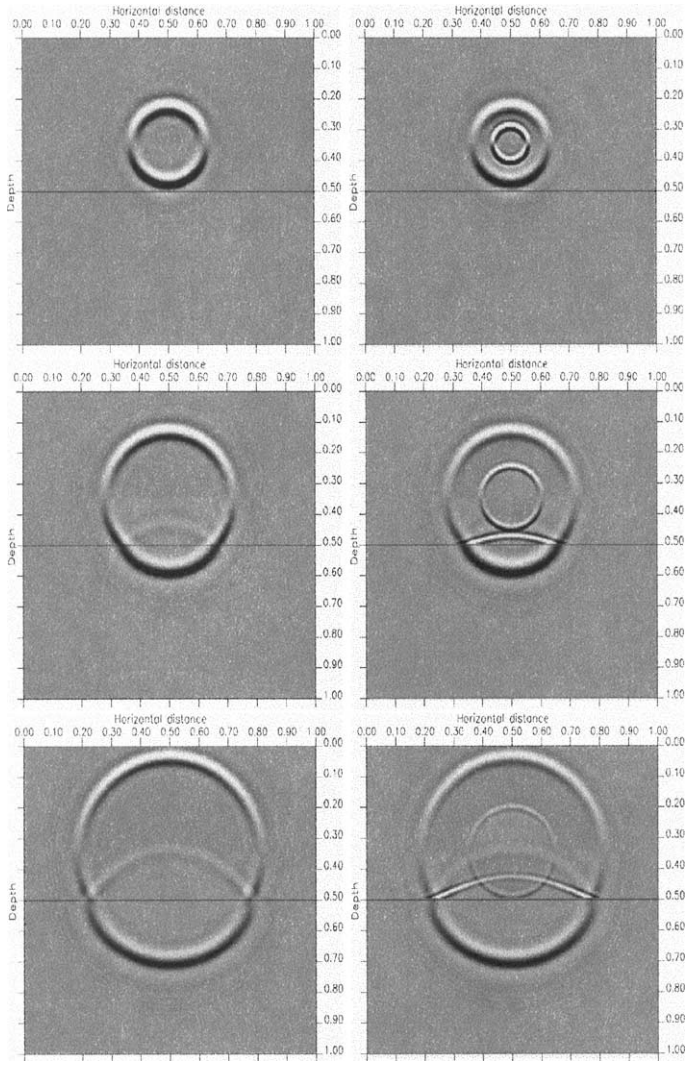

Fig. 5. Snapshots of wave propagation in layered poro-elastic model at $t = 0.08$, 0.12 and 0.16 sec. Left column is vertical displacement in solid (u_z) and right column is vertical displacement in fluid (U_z). Horizontal and vertical axis are in kilometers. The source is located (horizontaly) in the middle of the upper layer and (vertically) 0.35 km below the top of the model.

wave in layer 1. The layer velocities have been chosen in order not to expose too much energy of P–S waves converted at the layer boundary.

5. Discussion and conclusions

The modeling of elastic waves in solids and fluids is widely applied in physics and geophysics. All simulations reside on solutions of the stress–strain relations, resulting in some kind of wave phenomena. Such simulations usually have much in common and the differences of the various problem areas are mainly due to the geometry of the object to be illuminated, the wave frequencies, the recording apparatus, and the complexity of the medium properties. Clearly, it is advantageous to be able to exploit the various resemblances when implementing wave propagation problems on a computer. In order to achieve this goal, it is useful to employ modern object-oriented programming technology.

In this paper we have applied our recently developed object-oriented software development methodology, aimed at reducing software development costs for computational modeling, and demonstrated its capabilities within the field of elastic wave modeling. Our approach, which we have termed coordinate-free numerics, is based on notions from coordinate-free mathematics and applies formal techniques from computer science. The concepts from coordinate-free numerics have been implemented in the Sophus software library. A computer program for computational modeling developed within this abstract framework is easily modified and extended to satisfy new requirements on various levels.

In this work we have focused on the development of a collection of programs for elastic wave simulation. Since the Sophus library contains concepts from coordinate-free mathematics (e.g., tensors, differential operators) as programming entities, it was possible to achieve a highly flexible and efficient program development process. Among others we showed that an isotropic elastic wave simulator could be extended to tackle the situation of almost full anisotropy (TIA) with only a very limited amount of work. We also showed that a poro-elastic wave-simulator could be built upon this same structure only requiring a relatively small new solver module (i.e., SEISMOD). Numerical examples, ranging from the computation of seismic responses in a gas–oil reservoir with anisotropic overburden to the computation of the seismic wavefield within porous fluid-filled materials, were then presented in order to illustrate the ability and usefulness of our approach in the study of (geo)physical wave propagation problems.

We believe that this abstract programming approach is useful in computational modeling. Its nearness to the real mathematical language combined with its modularity, makes it relatively easy to experiment with new applications (e.g., PDEs) or with various numerical solution strategies (e.g., time and spatial discretizations). In this paper this has been shown when moving from elastic to poro-elastic wave simulation. Furthermore, the case of visco-elastic wave simulation could have been treated along similar lines. As another example, formulating (1) differently, e.g., by the velocity–stress formulation (see, e.g., [26]), which often may be useful for numerical reasons, is easily done basically by modifying the SEISMOD module. Experimentation with time discretization techniques is confined to the SEISMOD module, while spatial discretizations belong to the scalar field modules. Boundary conditions in the FD method basically affect the scalar field modules but have some minor implications for the SEISMOD and TENSOR modules.

Elastic wave simulation using, e.g., the finite element method (FEM) may thus in principle use the program structure described in this paper, only requiring a new SCALARFIELD module. However, moving to a method like the FEM also requires a reformulation of Eq. (1) to a variational from, thus requiring

a new SEISMOD module, as noted in the case study [13] taken from computational fluid dynamics. In our opinion this kind of flexibility makes this and other theoretically well funded object-oriented programming approaches highly valuable for scientific computing.

Acknowledgements

This research was supported in part by The Research Council of Norway (NFR), the European Union under ESPRIT Project 21871 (Scientific Computing and Algebraic Abstractions), Statoil, Saga Petroleum, and by computing resources grants from NFR's Programme for Supercomputing. Numerous people have contributed to the design and implementation of the Sophus library, of whom we will especially mention Victor Aarre, Kristin G. Frøysa, Helga Gunnarsli, Kristian Stewart and Steinar Søreide.

Appendix A. The Sophus library

The Sophus software library provides the notions from coordinate-free mathematics as programming entities. Each concept is specified to satisfy appropriate algebraic laws, such as those of scalar or tensor fields, while the implementations are based on conventional numerical algorithms and discretizations. Sophus is structured around the following concepts:

- Mesh structures M_n which are arrays with n indices. They include operations like $+, -$ and $*$ over all elements (much like Fortran-90 array operators). There are also operations for shifting the meshes along one of the indices. Parallel and sequential implementations of mesh structures can be used interchangeably, allowing easy porting between computer architectures of any program built on top of the mesh abstraction.
- Manifolds which define sets of points, embodying a notion of distance and direction. They represent the physical space where the problem to be solved takes place.
- Scalar fields on manifolds, which describe the measurable quantities of the physical problem to be solved. As the basic layer of "continuous mathematics" in the library, they provide the partial derivation and integration operations. Also, two scalar fields on the same manifold may be pointwise added, subtracted and multiplied. The different discretization methods, such as finite difference, finite element and finite volume methods, provide different designs for the implementation of scalar fields. Scalar fields are implemented using the mesh structures.
- Tensors T_n which provide the coordinate-free mathematics. The TENSOR module provides the general differentiation and integration operations, based on the partial derivatives and integrals of the scalar fields. Tensors also provide operations such as componentwise addition, subtraction and multiplication, as well as tensor product, composition and application. The implementation uses the meshes to hold the components, and is based on a choice of coordinate system, e.g., Cartesian, axisymmetric or general curvilinear.
- Solvers are written using the coordinate-free notation provided by the tensor level. Thus the choice of time integration technique and the use of advanced techniques for stability, error correction or convergence do not interfere with coordinate systems or discretization method.

A consequence of this structure is also that the choice of sequential or parallel solver is reduced to choosing appropriate mesh implementation. The parallelization technique is studied in detail in [15], while [16] presents some observations on the flexibility of this approach.

References

[1] K. Åhlander, M. Haveraaen, H. Munthe-Kaas, On the role of mathematical abstractions for scientific computing, in: R.F. Boisvert, P.T.P. Tang (Eds.), The Architecture of Scientific Software, Kluwer Academic Publishers, Boston, MA, 2001, pp. 145–158.
[2] J.L. Allen, C.P. Peddy, Amplitude Variation with Offset: Gulf Coast Case Studies, Geophysical Developments Series, Vol. 4, Society of Exploration Geophysicists, 1993.
[3] B. Auld, Acoustic Fields and Waves in Solids, Vol. 1, Wiley, New York, 1990.
[4] P.A. Berge, G.J. Fryer, R.H. Wilkens, Velocity–porosity relationships in the upper oceanic crust: Theoretical considerations, J. Geophys. Res. 97 (B11) (1992) 15239–15254.
[5] M.A. Biot, Theory of propagation of elastic waves in a fluid-saturated porous solid. I. Low-frequency range, J. Acoust. Soc. Am. 28 (1) (1956) 168–178.
[6] M.A. Biot, Theory of propagation of elastic waves in a fluid-saturated porous solid. II. Higher frequency range, J. Acoust. Soc. Am. 28 (2) (1956) 179–191.
[7] R. Burridge, J.B. Keller, Poroelasticity equations derived from microstructure, J. Acoust. Soc. Am. 70 (4) (1981) 1140–1146.
[8] J.P. Castagna, M. Backus, Offset-Dependent Reflectivity—Theory and Practice, Society of Exploration Geophysicists, 1993.
[9] C. Cerjan, D. Kosloff, R. Kosloff, M. Reshef, Short note: A nonreflecting boundary condition for discrete acoustic and elastic wave equations, Geophysics 50 (1985) 705–708.
[10] N. Dai, A. Vafidis, E.R. Kanasewich, Wave propagation in heterogeneous, porous media: A velocity–stress, finite-difference method, Geophysics 60 (2) (1995) 327–340.
[11] F. Gassmann, Elastic waves through a packing of spheres, Geophysics 16 (1951) 673–685.
[12] M. Graebner, Plane-wave reflection and transmission coefficients for a transversely isotropic solid, Geophysics 57 (11) (1992) 1512–1519.
[13] P. Grant, M. Haveraaen, M. Webster, Coordinate-free programming of computational fluid dynamics problems, Sci. Programming 8 (4) (2001).
[14] M. Haveraaen, Case study on algebraic software methodologies for scientific computing, Sci. Programming 8 (4) (2001).
[15] M. Haveraaen, Machine and collection abstractions for user-implemented data-parallel programming, Sci. Programming 8 (4) (2001).
[16] M. Haveraaen, H.A. Friis, T.A. Johansen, Formal software engineering for computational modelling, Nordic J. Comput. 6 (3) (1999) 241–270.
[17] M. Haveraaen, V. Madsen, H. Munthe-Kaas, Algebraic programming technology for partial differential equations, in: A. Maus, F. Eliassen et al. (Eds.), Norsk Informatikk Konferanse—NIK'92, Tapir, Norway, 1992, pp. 55–68.
[18] O. Holberg, Computational aspects of the choice of operator and sampling interval for numerical differentiation in large-scale simulation of wave phenomena, Geophys. Prospecting 35 (1987) 629–655.
[19] D.H. Johnston, Physical properties of shale at temperature and pressure, Geophysics 52 (10) (1987) 1391–1401.
[20] H. Munthe-Kaas, M. Haveraaen, Coordinate-free numerics—closing the gap between 'pure' and 'applied' mathematics?, Z. Angew. Math. Mech. 76 (1) (1996) 487–488.

[21] NORSAR Anisotropic Ray Modeling (ARM), Extension to NORSAR 2D Ray Modeling Software, NORSAR, Kjeller, Norway.
[22] B. Schutz, Geometrical Methods of Mathematical Physics, Cambridge University Press, Cambridge, 1980.
[23] P. Sheng, Effective-medium theory of sedimentary rocks, Phys. Rev. B 41 (7) (1990) 4507–4512.
[24] A.M. Skånøy, Finite difference modeling of elastic waves on parallel computers, Master's Thesis, Department of Informatics, University of Bergen, Norway, 1993.
[25] L. Thomsen, Weak elastic anisotropy, Geophysics 51 (10) (1986) 1954–1966.
[26] J. Virieux, P–SV wave propagation in heterogeneous media: Velocity–stress finite-difference method, Geophysics 51 (4) (1986) 889–901.