Wilson and van Keken
Progress in Earth and Planetary Science (2023) 10:68
https://doi.org/10.1186/s40645-023-00588-6

Progress in Earth and Planetary Science

REVIEW

Open Access

An introductory review of the thermal structure of subduction zones: II—numerical approach and validation

Cian R. Wilson[1] and Peter E. van Keken[1*]

Abstract

The thermal structure of subduction zones is fundamental to our understanding of the physical and chemical processes that occur at active convergent plate margins. These include magma generation and related arc volcanism, shallow and deep seismicity, and metamorphic reactions that can release fluids. Computational models can predict the thermal structure to great numerical precision when models are fully described but this does not guarantee accuracy or applicability. In a trio of companion papers, the construction of thermal subduction zone models, their use in subduction zone studies, and their link to geophysical and geochemical observations are explored. In this part II, the finite element techniques that can be used to predict thermal structure are discussed in an introductory fashion along with their verification and validation.

Keywords Geodynamics, Plate tectonics, Finite element methods, Subduction zone metamorphism, Arc volcanism

1 Introduction to part II

This paper is a companion to van Keken and Wilson "An introductory review of the thermal structure of subduction zones: I—motivation and selected examples" (van Keken and Wilson 2023a, hereafter called Part I) and van Keken and Wilson "An introductory review of the thermal structure of subduction zones: III—comparison between models and observations" (van Keken and Wilson 2023b, hereafter referred to as part III).

Combined these articles provide an introduction to the use of thermal models and observational constraints to aid our understanding of the dynamics, structure, and evolution of subduction zones from a geophysical, geochemical and petrological perspective. In Part I, we provided the motivation for these studies, fundamental constraints on subduction zone geometry and thermal structure, and a limited overview of existing thermal models. In this article, we will provide a discussion of the use of the finite element method to discretize partial differential equations needed for subduction zone modeling, present open-source software, and discuss validation and verification approaches to understand the reliability of the thermal models.

Our approach will be similar to that in part I—we strive to make this introduction accessible to advanced undergraduates, graduate students, and professionals from outside geodynamics. This will, hopefully, make the reader able to establish a fundamental understanding of what is required for numerical modeling of the thermal structure of subduction zones.

While we focus on the use of finite element methods to solve the governing equations, we acknowledge that significant and important studies have been published that use finite difference (FD) or finite volume (FV) methods. An introduction to the use of FD methods in geodynamical applications is provided by Gerya (2019). A broader overview of computational methods for geodynamics

*Correspondence:
Peter E. van Keken
pvankeken@carnegiescience.edu
[1] Earth and Planets Laboratory, Carnegie Institution for Science, 5241 Broad Branch Road NW, Washington, DC 20015, USA

including FV is in Ismail-Zadeh and Tackley (2010). A useful overview of the use of finite element methods specifically for mantle convection modeling with a comparison to FD and FV methods is in Zhong et al. (2015). As we will see, finite element methods can be used to discretize complex geometries, which provides a significant advantage for subduction zone modeling over FD and FV methods.

In Sect. 2, we first describe how finite element approaches to solve common linear partial differential equations such as the Poisson and Stokes equations are constructed. We then apply this to dynamical models that rely on solving the Stokes and heat equations, which include a standard convection benchmark and a new simplified subduction zone benchmark. The latter will be used to quantify the precision with which we can predict the subduction zone thermal structure using a kinematic–dynamic approach.

2 Finite element modeling

2.1 General formulation of the finite element solution of partial differential equations

The goal of the numerical models discussed here is to find the approximate solutions of partial differential equations (PDEs) in a spatial domain denoted by Ω, with boundaries $\partial\Omega$, representing some part of the Earth, say, a cross-section through a subduction zone. These PDEs can be time dependent, nonlinear, or nonlinearly coupled to other PDEs. To sketch out how we can discretize the PDEs with finite elements, we will first assume that we have linear PDEs of the general form

$$L(u) = f \qquad \text{in } \Omega \tag{1}$$

where L is a linear differential operator, f some right-hand side function and $u = u(\vec{x}, t)$ the solution we seek to approximate over space $\vec{x}$ and time t. In addition to (1), we require boundary conditions of the form

$$J(u) = g \qquad \text{on } \partial\Omega \tag{2}$$

where J is a linear differential operator and g is a function describing how u and/or its derivatives behave on the boundary. Efficient computer-based solution of the linear differential problem (1) and (2) relies on discretizing the domain Ω into a set of degrees of freedom (DOFs) or values at "nodal" points in the domain at which the approximate solution is sought. This discretization facilitates the translation of the governing equations from differential to algebraic matrix–vector form. Discretization schemes differ in how they organize and distribute the degrees of freedom onto a mesh or grid of points across the domain.

Finite difference methods distribute DOFs at points in Ω and construct approximate derivatives by taking the differences between the values of neighboring points (along connecting lines in a mesh of points). This is made easier if the DOFs are organized in a regular or structured grid. Finite volume methods construct control volumes around the degrees of freedom, and rather than approximating the derivatives, they consider the fluxes through the control volume boundaries between neighboring degrees of freedom. This means that the DOFs can be distributed in an unstructured way, but achieving higher orders of accuracy with FV methods is easier on structured meshes. The finite element method (FEM), on the other hand, tessellates the domain with polygonal elements and then distributes DOFs relative to these elements. The order of accuracy is then controlled by the number and the distribution of DOFs within an element, which can themselves be arranged in an unstructured pattern.

Formally, the FEM approximates u by $\tilde{u}$, the solution's representation in a function space on the mesh where

$$\tilde{u}(\vec{x}, t) = \sum_j \phi_j(\vec{x}) u_j(t) \tag{3}$$

Here, u_j are coefficients that as indicated can be time dependent but do not depend on space. The shape functions ϕ_j are a function of space but generally independent of time. The index j indicates the number of the shape function on the mesh and is associated with the number of the nodal point. In this manuscript, we will principally discuss so-called Lagrange shape functions which define ϕ_j as a polynomial over an element with a value of 1 at a single nodal point and a value of 0 at all other points associated with the degrees of freedom such that $\sum_j \phi_j$=1 (see Fig. 1). The shape functions can be of arbitrary order and can have various conditions on their continuity across or between elements. We will focus principally on linear Lagrange shape functions (denoted by P1) and quadratic Lagrange shape functions (denoted by P2) that are continuous between mesh elements. Our choice of Lagrange shape functions means that u_j are the actual values of the solution in (3). With some other forms of the shape function, u_j are instead interpolation weights that are used to construct the solution values. The split of temporal and spatial dependence above is typical in geodynamic applications but not required. Given the "trial" solution function (3), finite element methods pose (1) as a residual $R(\tilde{u})$:

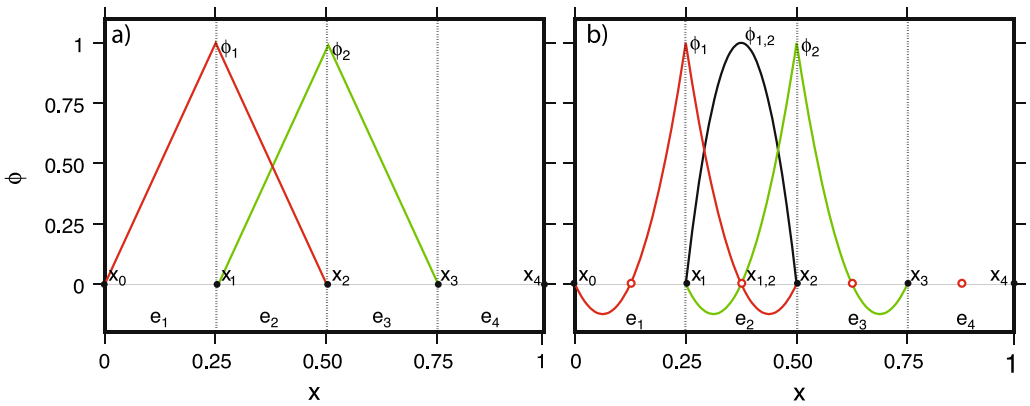

Fig. 1 **a** Illustration of the discretization of the 1D unit domain into four elements e_k with five nodal points x_i. The two linear (P1) Lagrange shape functions ϕ_i are shown that are nonzero in element e_2. **b** Illustration of quadratic (P2) shape functions that are nonzero on element e_2. The mesh still has four elements, but each element now has internal nodal points (indicated by open red circles)

$$R(\tilde{u}) = L(\tilde{u}) - f \tag{4}$$

The residual is minimized in a weighted average sense by multiplying the residual with a weighting test function, $\tilde{u}_t$, integrating over the domain of interest and setting this to zero:

$$\int \tilde{u}_t R(\tilde{u}) \mathrm{d}\Omega = 0 \tag{5}$$

The test functions $\tilde{u}_t$ can be independent of the functions ϕ_j that span the function space of the trial function, but in the widely used Galerkin approach the test functions are restricted to be in the same function space such that

$$\tilde{u}_t(\vec{x}, t) = \sum_i \phi_i(\vec{x}) u_{ti}(t) \tag{6}$$

Since the method is valid for all $\tilde{u}_t$, we can dispense with the test function values at the DOFs, u_{ti}, and the minimization function can be written as

$$\int \phi_i R(\tilde{u}) \mathrm{d}\Omega = 0 \quad \text{for all } i \tag{7}$$

Given a domain with n DOFs such that i, j=1, ..., n, combining (7) with (3) results in a matrix–vector system of the form

$$\mathbf{Su} = \mathbf{f} \tag{8}$$

where $\mathbf{S}$ is a $n \times n$ matrix, $\mathbf{f}$ is the right-hand side vector of length n and $\mathbf{u}$ is the solution vector of values or weights at the DOFs

$$\mathbf{S} = S_{ij} = \int \phi_i L(\phi_j) \mathrm{d}\Omega \tag{9}$$

$$\mathbf{f} = f_i = \int f \phi_i \mathrm{d}\Omega \tag{10}$$

$$\mathbf{u} = u_j \tag{11}$$

where we can move the solution values out of the integral in (7) due to the linear nature of L. For elliptic problems, $\mathbf{S}$ is sometimes called the stiffness matrix and $\mathbf{f}$ the load vector because the finite element method was initially used in structural problems where $\mathbf{u}$ typically represents a displacement. It expresses how for a given load $\mathbf{f}$ the stiffness of the structure, as expressed by the coefficients in the stiffness matrix $\mathbf{S}$, limits the displacement $\mathbf{u}$ of nodes in a structure. Note that in the above summary we have glossed over the imposition of boundary conditions (2), which must be incorporated into the residual (4), trial (3) and test (6) functions. Assuming that the boundary conditions are correctly implemented, that problems (1) and (2) are well posed, and that the discretization is adequate, then the discrete approximate solution $\mathbf{u}$ (11) can be found through direct or iterative solution of (8).

The ease with which finite elements can be used on an unstructured mesh gives them one of their primary advantages for subduction zone modeling—being able to tessellate complex geometries. This is of particular importance when, for example, explicitly discretizing the subducting slab surface, surface topography, or crustal interfaces in the overriding plate. In addition, grid refinement can be used where strong gradients in solutions exist (such as at the top of the slab when it gets in contact with the hot mantle wedge; see Figure 1b in part I) and coarse grids can be used where the solutions are relatively constant, leading to improved overall computational efficiency compared to methods that require a structured discretization

of space. Another advantage of the finite element method, that we will see below, is the natural way in which boundary conditions can be implemented.

For Lagrange bases increasing the order of the polynomial of ϕ_j increases the number of DOFs per element (see Fig. 1) and increases the order of accuracy of the solution. The shape functions may be continuous or discontinuous between elements but each ϕ_j ideally has compact support, meaning that the basis function associated with a degree of freedom only has nonzero values in the elements immediately surrounding the DOF. It is this property that ensures the matrix $\mathbf{S}$ (9) is sparse in the final discrete system of equations (8).

We provide practical examples that show how to construct (8) using finite elements. Our goal is to demonstrate the flexibility and power of the FEM without giving an exhaustive introduction or rigorous mathematical derivation of the method. Practical introductions to the FEM can be found in Johnson (1987) and Logan (2017). More mathematically founded descriptions of the FEM can be found in Oden and Reddy (1976), Hughes (1987), and Strang and Fix (2008). Some of these texts are available in affordable Dover reprints.

2.2 Construction of finite element models

2.2.1 Examples of partial differential equations solved by the FEM

The exact set of equations that needs to be solved to make predictions of the thermal structure of subduction zones using a kinematic–dynamic approach is provided in Sect. 2.3.1. These are derived from the fundamental equations governing the conservation of mass, momentum, and thermal energy. The conservation of mass and momentum leads, under a number of simplifying assumptions (that we will not discuss in detail but that can be found in fundamental textbooks such as Turcotte and Schubert 2002), to the nondimensional Stokes equation and the condition of incompressibility

$$-\nabla\cdot\left(2\eta\frac{\nabla\vec{v}+\nabla\vec{v}^{T}}{2}\right)+\nabla P = \vec{f}_B \tag{12}$$

$$\nabla\cdot\vec{v} = 0 \tag{13}$$

Given a viscosity, η, and a buoyancy force, $\vec{f}_B$, that can depend on temperature and composition, the Stokes equation balances viscous, pressure, and buoyancy forces. Further imposition of the incompressibility constraint (13) allows us to find the velocity, $\vec{v}$, and pressure, P. The conservation of thermal energy leads to the nondimensional heat advection–diffusion equation

$$\rho c_p\left(\frac{\partial T}{\partial t}+\vec{v}\cdot\nabla T\right) = \nabla\cdot(k\nabla T)+H \tag{14}$$

which, given the density, ρ, heat capacity, c_p, and thermal conductivity, k, balances the transport of heat by diffusion and advection with heat production, H. The heat equation can be modeled to be stationary (by assuming $\frac{\partial T}{\partial t}$=0) and the Stokes equation can be nonlinear due to the dependence of the viscosity on stress. The Stokes equation with the incompressibility constraint are generally nonlinearly coupled with the heat advection–diffusion equation.

In this section, rather than immediately solving the full nonlinear set of equations, we will provide examples of how to solve (12)–(14) one by one, under various simplifying assumptions, before embarking on a fully coupled problem. We will start with a simple worked-out example of a 1D Poisson equation which is arguably the simplest form of (14) under the assumption of zero velocity, which also eliminates (12) and (13) entirely. This will include the generation of shape functions, construction of the matrix–vector system, solution on a coarse mesh, comparisons between linear and quadratic elements, and convergence tests. This section is particularly intended for those new to finite element methodology and nomenclature. Those comfortable with basic FEM concepts but interested in the weak form formulation of PDEs and their FEM solution can skip forward to Sect. 2.2.3 where we describe the FEM implementation and software availability. This is followed by the extension of the Poisson heat diffusion problem to more than one dimension and the solution of the linear Stokes equation for a traditional corner-flow problem, neglecting temperature effects. We then combine the heat and Stokes equation in coupled problems using a standard mantle convection benchmark before focusing on simplified models of subduction zones. Unless explicitly mentioned otherwise, we will assume in all examples below that the equations are in nondimensional form.

Section 2.3 derives (12)–(14) from their dimensional form and discusses how they are used in kinematic–dynamic subduction zone models. Readers who are more interested in understanding how different modeling approaches for subduction zone thermal structure compare or how the models compare to observations are invited to skip forward to part III.

2.2.2 1D Poisson

As an introductory and simplified example, we will solve the Poisson equation on a 1D domain of unit length, $\Omega = [0, 1]$. This can be derived from the steady-state form of (14) by assuming zero velocity and a constant thermal conductivity, and seeking the approximate solution of

$$-\frac{\mathrm{d}^2 T}{\mathrm{d}x^2} = f \tag{15}$$

where we choose for this example $f = \frac{H}{k} = \frac{1}{4}\pi^2 \sin\left(\frac{\pi x}{2}\right)$. At the boundaries, $x = 0$ and $x = 1$, we apply as boundary conditions (2)

$$T = 0 \quad \text{at } x = 0 \tag{16}$$

$$\frac{\mathrm{d}T}{\mathrm{d}x} = 0 \quad \text{at } x = 1 \tag{17}$$

The first boundary condition is an example of an essential or Dirichlet boundary condition where we specify the value of the solution. The second boundary condition is an example of a natural or Neumann boundary condition that can be interpreted to mean that the solution is symmetrical around $x = 1$. We will return to the various types of boundary conditions and their implementation in a later section. The analytical solution to (15) with given boundary conditions (16) and (17) is simply

$$T = \sin\left(\frac{\pi x}{2}\right) \tag{18}$$

Minimization of the residual $R(\tilde{T})$ following (4) and (7) leads to

$$-\int_0^1 \phi_i \frac{\mathrm{d}^2 \tilde{T}}{\mathrm{d}x^2} \mathrm{d}x = \int_0^1 \phi_i f \,\mathrm{d}x \qquad i = 1, \ldots, n \tag{19}$$

By integrating the first term by parts, we find

$$\int_0^1 \frac{\mathrm{d}\phi_i}{\mathrm{d}x}\frac{\mathrm{d}\tilde{T}}{\mathrm{d}x}\mathrm{d}x - \left[\phi_i \frac{\mathrm{d}\tilde{T}}{\mathrm{d}x}\right]_0^1 = \int_0^1 \phi_i f \,\mathrm{d}x \qquad i = 1, \ldots, n \tag{20}$$

where the second term can be dropped because at $x = 1$ we require $\frac{\mathrm{d}\tilde{T}}{\mathrm{d}x} = 0$ and the solution at $x = 0$ is known, $\tilde{T} = 0$, so can be lifted from the resulting matrix equation.

We can find the solution at the DOFs, T_j, from the discrete $n \times n$ matrix–vector system (8) where now

$$\mathbf{S} = S_{ij} = \int_0^1 \frac{\mathrm{d}\phi_i}{\mathrm{d}x}\frac{\mathrm{d}\phi_j}{\mathrm{d}x}\,\mathrm{d}x \tag{21}$$

$$\mathbf{f} = f_i = \int_0^1 \phi_i f \,\mathrm{d}x \tag{22}$$

$$\mathbf{u} = \mathbf{T} = T_j \tag{23}$$

where $\mathbf{T}$ has components T_j that define the continuous approximate solution

$$\tilde{T}(x) = \sum_{j=1}^{n} \phi_j(x) T_j \tag{24}$$

and $T_0 = 0$.

The domain is divided into n_e elements of equal length, $\Delta x = \frac{1}{n_e}$, with elements e_i and degrees of freedom T_i ordered from $x = 0$ to $x = 1$. This introduces nodal points x_i, $0 \le i \le n$ (see Fig. 1a). A simple assumption for the Lagrange shape functions ϕ_i is that the shape functions are linear within the elements. Such functions within a given element e_i $(x_{i-1} \le x \le x_i)$, $1 \le i \le n_e$, are

$$\lambda_{i-1} = \frac{x_i - x}{\Delta x}, \qquad \lambda_i = \frac{x - x_{i-1}}{\Delta x} \tag{25}$$

The functions λ_j are zero for all elements except e_j and e_{j+1} $(\forall e_i \notin \{e_j, e_{j+1}\})$. Since they fit the definition of linear Lagrange functions, we can write $\phi_i = \lambda_i$. Within a given element e_i, we can construct the interpolated approximate solution for $\tilde{T}$ from $\mathbf{T}$ using

$$\tilde{T}(x) \;=\; T_{i-1}\phi_{i-1}(x) + T_i \phi_i(x) \tag{26}$$

The expression is compact because all shape functions other than ϕ_{i-1} and ϕ_i are zero within this element. Note that the derivatives of the shape functions in this element are simply

$$\frac{\mathrm{d}\phi_{i-1}}{\mathrm{d}x} = -\frac{1}{\Delta x}, \qquad \frac{\mathrm{d}\phi_i}{\mathrm{d}x} = \frac{1}{\Delta x} \tag{27}$$

which allows for easy evaluation of the matrix coefficients.

Evaluation of the integrals in (21) and (22) allows us to construct (8) as

$$\frac{1}{\Delta x^2}\begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 2 & -1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 1 \end{pmatrix}\begin{pmatrix} T_1 \\ T_2 \\ \vdots \\ T_{n-1} \\ T_n \end{pmatrix} = \begin{pmatrix} \int f\phi_1 \mathrm{d}x \\ \int f\phi_2 \mathrm{d}x \\ \vdots \\ \int f\phi_{n-1}\mathrm{d}x \\ \int f\phi_n \mathrm{d}x \end{pmatrix} \tag{28}$$

The integral in the right-hand side vector $\mathbf{f}$ can be found analytically or through numerical integration. The matrix may look familiar to those acquainted with finite difference approximations to the 1D Poisson equation where $\mathrm{d}^2T/\mathrm{d}x^2$ is approximated by second-order central finite differences (for a derivation see, e.g., Cuvelier et al. 1986, their section 2.2.1). The matrix rows repeat triples (−1, 2, −1) to form a tridiagonal symmetric matrix for which (very) efficient solution methods exist.

Implementation While writing out the system of equations is instructive, and solutions can be constructed by manual Gaussian elimination for a small number of degrees of freedom n, solution of the equations governing subduction zone thermal structure requires significantly more involved code. Modern software design approaches have become available that allow us to develop numerical code using a relatively simple syntax in which the developer describes the problem in terms of the differential equation and boundary conditions, specifies the coefficients, the geometry and its discretization, and solution methods. We will provide a few examples of high-level syntax (written in python) that can be used with the open-source FEniCS software (Logg et al. 2012) to produce a finite element code. We will first introduce this syntax and provide a more complete description of the approach that we use in Sect. 2.2.3.

The one-dimensional heat diffusion problem (15)–(17) can be solved using FEniCS with the python function `solve_poisson_1d` (Listing 1). Lagrange polynomials of order one (defined by the keyword argument `p` on line 17, which defaults to 1) are used to define a function space (V). Test (`T_t`) and trial (`T_a`) functions are defined on this function space, before being used to describe the integrals defining $\mathbf{S}$ and $\mathbf{f}$. The Dirichlet boundary condition at $x = 0$ is then declared as `bc` before being passed to a function `solve` that assembles the matrix–vector system, manipulates it to ensure satisfaction of the essential boundary condition and solves for `T_i`, the function containing the vector of values of $\tilde{T}$ at the DOFs T_j. Finally, the solution is returned.

Higher-order elements We will use this simple example further to show that we can construct shape functions of higher order that allow us to find solutions that are (in general) more accurate with the same number of nodal points compared to solutions with lower order shape functions. We will construct quadratic Lagrange shape functions on the elements as shown in Fig. 1b. Note that each element now has an internal nodal point such that the number of nodal points for the fixed number of elements increases by nearly a factor of two compared to the linear P1 function space (Fig. 1a). Within an element e_i $(x_{i-1} \le x \le x_i)$, there are three shape functions that are of quadratic form

$$\phi_{i-1} = \frac{2}{\Delta x^2}(x - x_i)(x - x_{i-1,i}) = 2\lambda_{i-1}(\lambda_{i-1} - \tfrac{1}{2}) \tag{29}$$

$$\phi_{i-1,i} = \frac{-4}{\Delta x^2}(x - x_{i-1})(x - x_i) = 4\lambda_{i-1}\lambda_i \tag{30}$$

$$\phi_i = \frac{2}{\Delta x^2}(x - x_{i-1})(x - x_{i-1,i}) = 2\lambda_i(\lambda_i - \tfrac{1}{2}) \tag{31}$$

with λ_i and λ_{i-1} defined in (25). We have used the notation $\phi_{i-1,i}$ to identify the internal Lagrange polynomial centered in element e_i on the new internal nodal point $x_{i-1,i}$. This also makes explicit the relation between the P1 nodal points and the edge nodal points (also called vertices) of the P2 elements and clarifies the relationship between P1 and P2 shape functions through (29)–(31). Note that the nonzero values of a quadratic Lagrange shape function may extend beyond the neighboring DOFs and they can be positive or negative depending on where its nodal point is located within an element. Note also that the shape functions now connect more nodal points to the central nodal point—which suggests matrix (28) changes form to have more entries per row than in the case of the P1 based matrix. In addition the matrix will have more rows since there are more nodal points for the same number of elements. Clearly the use of higher order elements comes at a greater computational cost since it is more expensive to solve a larger algebraic system.

Calling the python function `solve_poisson_1d` with a second keyword argument `p=2` allows us to solve the system with quadratic Lagrange shape functions. The script shows that only the definition of the `FunctionSpace` is changed by setting `p=2`. Figure 2 shows the approximate solution for linear and quadratic elements on a coarse grid compared to the analytical solution.

```python
# Import the dolfin library (a component of FEniCS)
from dolfin import *
def solve_poisson_1d(ne, p=1):
  """
  A python function to solve a one-dimensional Poisson problem
  on a unit interval domain.
  Parameters:
    * ne - number of elements
    * p  - polynomial order of the solution function space
  """
  # Describe the domain (a one-dimensional unit interval)
  # and also the tessellation of that domain into ne
  # equally spaced elements
  mesh = UnitIntervalMesh(ne)
  # Define the solution function space using Lagrange polynomials
  # of order p
  V = FunctionSpace(mesh, "Lagrange", p)

  # Define the trial and test functions on the same function space (V)
  T_a = TrialFunction(V)
  T_t = TestFunction(V)

  # Define the location of the boundary, x=0
  def boundary(x):
      return x[0] < DOLFIN_EPS
  # Specify the value and define a boundary condition (bc)
  gD = Constant(0.0)
  bc = DirichletBC(V, gD, boundary)

  # Define the right hand side function, rhsf
  x = SpatialCoordinate(mesh)
  rhsf = (pi**2)*sin(pi*x[0]/2)/4

  # Define the integral to be assembled into the stiffness matrix
  S = inner(grad(T_t), grad(T_a))*dx
  # Define the integral to be assembled into the forcing vector
  f = T_t*rhsf*dx

  # Define the solution and compute it (given the boundary condition,
    bc)
  T_i = Function(V)
  solve(S == f, T_i, bc)

  # Save solution to disk in XDMF format
  ofile = XDMFFile("poisson_{}_{}.xdmf".format(ne,p,))
  ofile.write(T_i)
  ofile.close()

  # Return the solution
  return T_i
```

Listing 1 FEniCS example for the 1D Poisson FEM solution (see XDMF.org for a description of the XDMF file format)

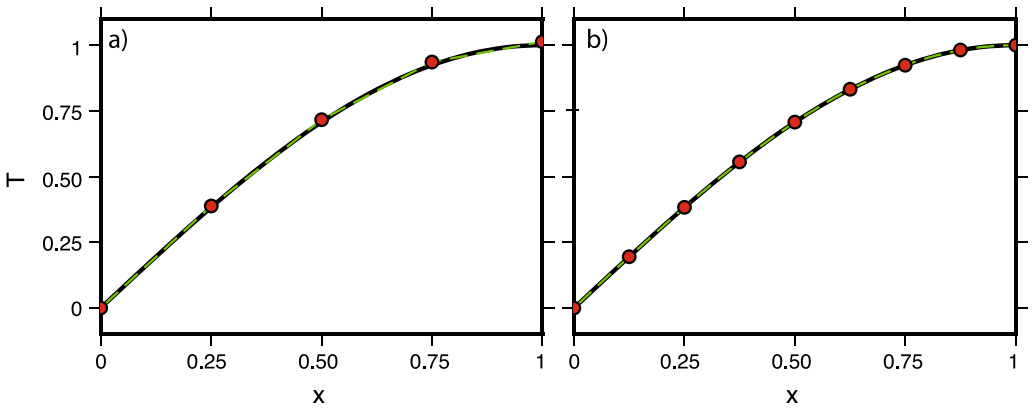

Fig. 2 FEM solution to 1D Poisson equation. **a** Approximate FEM solution obtained using a mesh of four P1 elements compared to analytical solution (black). Discrete values from the solution vector **T** are shown in red circles. The interpolated approximate solution is shown by the green dashed line. **b** Same as a) but now for four P2 elements

Note that the P2 solution stays closer to the analytical solution than the P1 solution.

Convergence analysis Repeating the numerical experiments with increasing `ne` allows us to test the convergence of our approximate finite element solution to the known analytical solution (18). A key feature of any discretization technique is that with an increasing number of DOFs these solutions should converge, i.e. the error in our approximation should decrease. As an error metric, we will use the L^2 norm of the difference between the approximate, $\tilde{T}$, and analytical, T, solutions

$$e_{L^2,P} = \sqrt{\int_\Omega \left(\tilde{T} - T\right)^2 \mathrm{d}x} \tag{32}$$

where the subscript P stands for Poisson. The rate at which this decreases is known as the order of convergence. Numerical analysis predicts a certain order depending on the type of the polynomials used as finite element shape functions and other constraints related to the well-posedness of the problem. For piecewise-linear shape functions, we expect second-order convergence, that is that the error decreases as h^2 where h is the nodal point spacing. With piecewise-quadratic elements, we expect to see third-order convergence. These expectations are met by the actual numerical experiments (Fig. 3). Convergence analysis is an essential way to test the accuracy of a numerical model, but it relies on having a known analytical solution and the ability to represent it and its boundary conditions in a discrete function space. We will discuss this issue in the context of other examples with increasing complexity below.

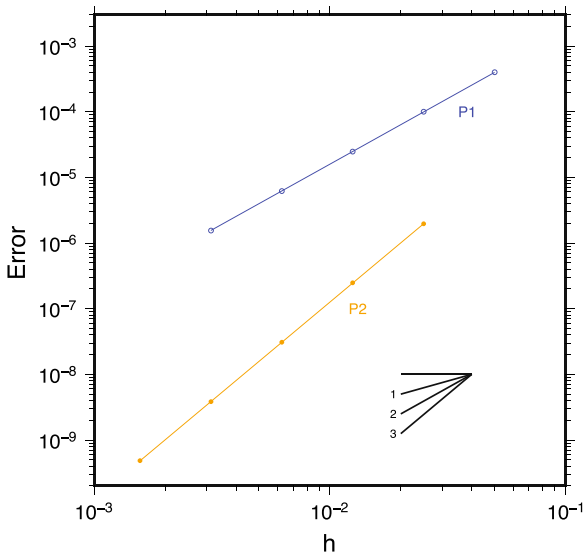

Fig. 3 Convergence analysis for the 1D Poisson problem with P1 (blue) and P2 (orange) elements. The error following metric (32) is shown as a function of nodal point spacing h. Symbols represent individual experiments—the lines show the trend. With P1 we find second-order convergence, whereas with P2 we find a smaller error at a given h and a faster, third-order, convergence rate

2.2.3 Practical approaches, software availability and comparison

Traditionally, finite element methods have been implemented using Fortran or C/C++ based codes that, at the core, build the matrix–vector system (8) by numerical integration of (9) and (10) after which this system is solved by linear algebraic solvers. Most FEM codes provide options for time dependence and the ability to solve nonlinear and nonlinearly coupled systems of PDEs. Examples of such codes that have been used in geodynamical applications including subduction zone modeling are ConMan (King et al. 1990), Sopale (Fullsack 1995), Underworld (Moresi et al. 2007), CitcomS (Zhong et al. 2008), MILAMIN (Dabrowski et al. 2008), ASPECT (Kronbichler et al. 2013), Sepran (van den Berg et al. 2015), Fluidity (Davies et al. 2011), and

Rhea (Burstedde et al. 2013). A number of these are distributed as open-source software and many among those are currently maintained through the Computational Infrastructure for Geodynamics (geodynamics.org). These implementations can be shown to be accurate using intercomparisons and benchmarks (e.g., King et al. 2010; van Keken et al. 2008; Euen et al. 2022; Davies et al. 2011) and make use of advances in parallel computing and efficient linear algebra solver techniques. Yet, modifications to the existing code requires deep insight into the structure of the Fortran/C/C++ code which is not trivial for experienced, let alone beginning, users.

In recent years, an alternative approach for FEM has become available which elevates the user interface to simply specifying the FEM problem and solution method with the high-level approach of which an example is shown in Listing 1. The python code is used to automatically build a finite element model that can be executed in a variety of environments ranging from Jupyter notebooks (jupyter.org) and desktop computers to massively parallel high performance computers. Two prominent examples of this approach are Firedrake (www.firedrakeproject.org) and FEniCS (www.fenicsproject.org). Examples of the use of these two approaches in geodynamical applications are in Davies et al. (2022) and Vynnytska et al. (2013).

We will focus on the use of the FEniCS ("Finite Elements in Computational Sciences"; Alnæs et al. 2015) approach to solving finite element equations. FEniCS is a suite of open-source numerical libraries for the description of finite element problems. Most importantly, it provides a high-level, human-readable language for the description of equations in python (the "Unified Form Language" (UFL); Alnæs et al. 2014, an example of which we provided in Listing 1) and a compiler (the "FEniCS Form Compiler" (FFC); Kirby and Logg 2006) to write fast code to assemble the resulting discrete matrix–vector system. We will specifically use FEniCS within TerraFERMA (the "Transparent Finite Element Rapid Model Assembler;" Wilson et al. 2017). TerraFERMA provides a graphical user interface (using the "System for Problem Description" (SPuD); Ham et al. 2009) that allows users to describe the geometry, variables and boundary conditions of their problem and construct physics-based solvers using PETSc (the "Portable Extensible Toolkit for Scientific computation;" Balay et al. 2023).

TerraFERMA aims to increase transparency in modeling by exposing all options, including the equations, in a single options file that can be validated and automatically updated, which increases reproducibility. We provide all options files used in the following sections in a repository and in a docker image (see material contained in the zenodo repository referenced in the data availability statement) for readers to try. In addition to results from TerraFERMA, we compare some solutions with the aforementioned finite element package Sepran which has been used extensively in subduction zone modeling (e.g., Syracuse et al. 2010; van Keken et al. 2011). Sepran is not an open-source code but allows for direct comparisons between independent finite element methods to establish their relative precision.

2.2.4 The Poisson equation beyond 1D

We can generalize (and formalize) the description of the Poisson equation using the steady-state heat diffusion equation in multiple dimensions, where (14) becomes

$$-\nabla \cdot (k\nabla T) = H \quad \text{in } \Omega \tag{33}$$

after assuming zero velocity. T is the temperature solution we are seeking, k is the thermal conductivity, and H is a heat source. If k is constant in space, we can simplify (33) to

$$-\nabla^2 T = f \quad \text{in } \Omega \tag{34}$$

where $f = \frac{H}{k}$.

Boundary conditions We supplement (34) with some combination of the boundary conditions (2)

$$T = g_D \quad \text{on } \partial\Omega_D \subset \partial\Omega \tag{35}$$

$$\nabla T \cdot \hat{\vec{n}} = g_N \quad \text{on } \partial\Omega_N \subset \partial\Omega \tag{36}$$

$$aT + \nabla T \cdot \hat{\vec{n}} = g_R \quad \text{on } \partial\Omega_R \subset \partial\Omega \tag{37}$$

where $\partial\Omega_D$, $\partial\Omega_N$ and $\partial\Omega_R$ are segments of the domain boundary that do not overlap ($\partial\Omega_D \bigcap \partial\Omega_N = \emptyset$, $\partial\Omega_D \bigcap \partial\Omega_R = \emptyset$, $\partial\Omega_N \bigcap \partial\Omega_R = \emptyset$) and that together span the entire boundary ($\partial\Omega_D \bigcup \partial\Omega_N \bigcup \partial\Omega_R = \partial\Omega$). The unit outward-pointing normal to the boundary $\partial\Omega$ is denoted by $\hat{\vec{n}}$, and $g_D = g_D(\vec{x}, t)$, $g_N = g_N(\vec{x}, t)$ and $g_R = g_R(\vec{x}, t)$ are known functions of space and time. Equation (35) is known as a Dirichlet boundary condition and specifies the value of the solution on $\partial\Omega_D$. Equation (36) is a Neumann boundary condition and specifies the value of the flux through $\partial\Omega_N$. Finally, Equation (37) is a Robin boundary condition, which describes a linear combination of the flux and the solution on $\partial\Omega_R$.

Weak form The first step in the finite element discretization of (34) is to transform it into its weak form. Following (7), this requires multiplying the equation by a test function, T_t, and integrating over the domain Ω

$$-\int_\Omega T_t \nabla^2 T \, \mathrm{d}x = \int_\Omega T_t f \, \mathrm{d}x \tag{38}$$

After integrating the left-hand side by parts

$$\int_\Omega \nabla T_t \cdot \nabla T \, \mathrm{d}x - \int_{\partial\Omega} T_t \nabla T \cdot \hat{\vec{n}} \, \mathrm{d}s = \int_\Omega T_t f \, \mathrm{d}x \tag{39}$$

we can see that we have reduced the continuity requirements on T by only requiring its first derivative to be bounded across Ω (see Hughes 1987, for a more formal discussion of the requirements on the solution). Integrating by parts also allows Neumann and Robin boundary conditions to be imposed "naturally" through the second integral on the left-hand side since this directly incorporates the flux components across the boundary. In this formulation, Dirichlet conditions cannot be imposed weakly and are referred to as essential boundary conditions, that are required of the solution but do not arise naturally in the weak form. The weak form therefore becomes: find T such that $T{=}g_D$ on $\partial\Omega_D$ and

$$\begin{aligned}&\int_\Omega \nabla T_t \cdot \nabla T \,\mathrm{d}x - \int_{\partial\Omega_N} T_t g_N \,\mathrm{d}s \\ &- \int_{\partial\Omega_R} T_t \left(g_R - aT\right) \,\mathrm{d}s = \int_\Omega T_t f \,\mathrm{d}x\end{aligned} \tag{40}$$

for all T_t such that $T_t = 0$ on $\partial\Omega_D$.

Discretization The weak (40) and strong (34)–(37) forms of the problem are equivalent so long as the solution is sufficiently smooth. We make our first approximation to the solution by seeking the trial function $\tilde{T}$ such that $\tilde{T} = g_D$ on $\partial\Omega_D$ where

$$T \approx \tilde{T} = \sum_j \phi_j T_j \tag{41}$$

for all test functions $\tilde{T}_t$ where

$$T_t \approx \tilde{T}_t = \sum_i \phi_i T_{ti} \tag{42}$$

noting again that $\tilde{T}_t = 0$ on $\partial\Omega_D$. The finite element shape functions ϕ_j are as discussed earlier. Assuming these are continuous across elements of the mesh, (41) and (42) can be substituted into (40) to yield

$$\begin{aligned}&\sum_i \sum_j T_{ti} T_j \sum_k \int_{e_k} \nabla\phi_i \cdot \nabla\phi_j \,\mathrm{d}x \\ &+ \sum_i \sum_j T_{ti} T_j \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i a \phi_j \,\mathrm{d}s \\ &- \sum_i T_{ti} \sum_k \int_{\partial e_k \cap \partial\Omega_N} \phi_i g_N \,\mathrm{d}s \\ &- \sum_i T_{ti} \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i g_R \\ &= \sum_i T_{ti} \sum_k \int_{e_k} \phi_i f \,\mathrm{d}x\end{aligned} \tag{43}$$

where we are integrating over the whole domain by summing the integrals over all the elements e_k ($\int_\Omega \mathrm{d}x = \sum_k \int_{e_k} \mathrm{d}x$). Note that in practice, because the shape functions are zero over most of the domain, only element integrals with nonzero values need be included in the summation. The element boundaries, ∂e_k, are only of interest (due to the assumed continuity of the shape functions between the elements) if they either intersect with $\partial\Omega_N$, $\partial e_k \cap \partial\Omega_N$ or $\partial\Omega_R$, $\partial e_k \cap \partial\Omega_R$. Since the solution of the now discretized weak form should be valid for all $\tilde{T}_t$, we can drop T_{ti} from (43)

$$\begin{aligned}&\sum_j T_j \sum_k \int_{e_k} \nabla\phi_i \cdot \nabla\phi_j \,\mathrm{d}x + \sum_j T_j \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i a \phi_j \,\mathrm{d}s \\ &- \sum_k \int_{\partial e_k \cap \partial\Omega_N} \phi_i g_N \,\mathrm{d}s - \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i g_R \,\mathrm{d}s = \sum_k \int_{e_k} \phi_i f \,\mathrm{d}x\end{aligned} \tag{44}$$

This represents a matrix–vector system of the form of (8) with

$$\mathbf{S} = S_{ij} = \sum_k \int_{e_k} \nabla\phi_i \cdot \nabla\phi_j \,\mathrm{d}x + \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i a \phi_j \,\mathrm{d}s \tag{45}$$

$$\begin{aligned}\mathbf{f} = f_i = &\sum_k \int_{e_k} \phi_i f \,\mathrm{d}x + \sum_k \int_{\partial e_k \cap \partial\Omega_N} \phi_i g_N \,\mathrm{d}s \\ &+ \sum_k \int_{\partial e_k \cap \partial\Omega_R} \phi_i g_R \,\mathrm{d}s\end{aligned} \tag{46}$$

$$\mathbf{u} = \mathbf{T} = T_j \tag{47}$$

The compact support of the shape functions $\phi_{(i,j)}$, which limits their nonzero values to the elements immediately neighboring DOF i or j, means that the integrals in (45) and (46) can be evaluated efficiently by only considering shape functions associated with an element e_k. It also means that the resulting matrix $\mathbf{S}$ is sparse, with most entries being zero. These properties can be seen by considering a one-dimensional version of (34) as discussed in Sect. 2.2.2.

For an example of the implementation of the 2D Poisson problem on a unit square see Listing 2 with convergence tests and solution in Fig. 4. In this case, we use a manufactured solution (that is, one that is not necessarily an example of a solution to a PDE representing a naturally occurring physical problem) where we take a known analytical solution $T(x, y)$ and substitute this into (34) to find f and then use this as the right-hand side in our numerical test. We choose $T(x, y) = \exp\left(x + \frac{y}{2}\right)$ which is the solution to

$$-\nabla^2 T = -\tfrac{5}{4} \exp\left(x + \tfrac{y}{2}\right) \tag{48}$$

```
from dolfin import *
def solve_poisson_2d(ne, p=1):
  """
  A python function to solve a two-dimensional Poisson problem
  on a unit square domain.
  Parameters:
    * ne - number of elements in each dimension
    * p  - polynomial order of the solution function space
  """
  # Describe the domain (a unit square)
  # and also the tessellation of that domain into ne
  # equally spaced squares in each dimension which are
  # subdivided into two triangular elements each
  mesh = UnitSquareMesh(ne, ne, diagonal='right')
  # Define the solution function space using Lagrange polynomials
  # of order p
  V = FunctionSpace(mesh, "Lagrange", p)

  # Define the trial and test functions on the same function space (V)
  T_a = TrialFunction(V)
  T_t = TestFunction(V)

  # Define the location of the boundary condition, x=0 and y=0
  def boundary(x):
      return x[0] < DOLFIN_EPS or x[1] < DOLFIN_EPS
  # Specify the value and define a Dirichlet boundary condition (bc)
  gD = Expression("exp(x[0] + x[1]/2.)", degree=p)
  bc = DirichletBC(V, gD, boundary)

  # Get the coordinates
  x = SpatialCoordinate(mesh)
  # Define the Neumann boundary condition function
  gN = as_vector((exp(x[0] + x[1]/2.), 0.5*exp(x[0] + x[1]/2.)))
  # Define the right hand side function, rhsf
  rhsf = -5./4.*exp(x[0] + x[1]/2.)

  # Get the unit vector normal to the facets
  n = FacetNormal(mesh)
  # Define the integral to be assembled into the stiffness matrix
  S = inner(grad(T_t), grad(T_a))*dx
  # Define the integral to be assembled into the forcing vector,
  # incorporating the Neumann boundary condition weakly
  f = T_t*rhsf*dx + T_t*inner(gN, n)*ds

  # Define the solution and compute it (given the boundary condition,
    bc)
  T_i = Function(V)
  solve(S == f, T_i, bc)

  # Save solution to disk in XDMF format
  ofile = XDMFFile("poisson_{}_{}.xdmf".format(ne,p,))
  ofile.write(T_i)
  ofile.close()

  # Return the solution
  return T_i
```

Listing 2 FEniCS script for 2D Poisson problem

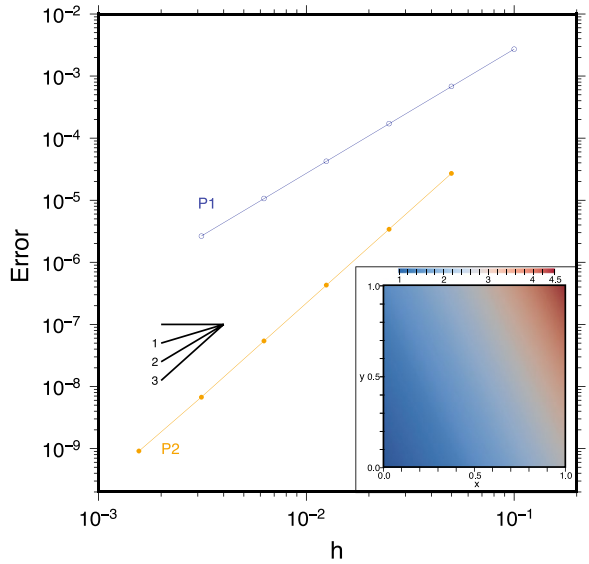

Fig. 4 Poisson 2D example. Convergence of the error as a function of nodal point spacing h for P1 and P2 elements. Inset: solution field T

Solving (48) numerically in a unit square, $\Omega = [0,1] \times [0,1]$, for the approximate solution $\tilde{T} \approx T$, we impose the boundary conditions

$$\tilde{T} = \exp\left(x + \tfrac{y}{2}\right) \quad \text{on } \partial\Omega \text{ where } x = 0 \text{ or } y = 0 \tag{49}$$

$$\nabla\tilde{T} \cdot \hat{\vec{n}} = \exp\left(x + \tfrac{y}{2}\right) \quad \text{on } \partial\Omega \text{ where } x = 1 \tag{50}$$

$$\nabla\tilde{T} \cdot \hat{\vec{n}} = \tfrac{1}{2}\exp\left(x + \tfrac{y}{2}\right) \quad \text{on } \partial\Omega \text{ where } y = 1 \tag{51}$$

where (49) represents an essential Dirichlet condition on the value of $\tilde{T}$ and (50)–(51) are natural Neumann conditions on $\nabla\tilde{T}$.

Listing 2 shows an implementation of this problem using FEniCS, which returns the approximate solution $\tilde{T}$. Comparison of this to the analytical solution T using the metric (32) gives the expected order of convergence for the P1 and P2 elements (see Fig. 4).

2.2.5 Batchelor cornerflow problem

The solid flow in a subduction zone is primarily driven by the motion of the downgoing slab entraining material in the mantle wedge and dragging it down with it setting up a cornerflow in the mantle wedge (see, e.g., Figure 1a in part I). This effect can be simulated by imposing the motion of the slab as a kinematic boundary condition at the base of the dynamic mantle wedge, allowing us to drop the buoyancy term from (12), $\vec{f}_B = 0$. With the further assumption of an isoviscous rheology, $2\eta = 1$, the momentum and mass equations simplify to

$$-\nabla \cdot \left(\frac{\nabla\vec{v} + \nabla\vec{v}^T}{2}\right) + \nabla P = 0 \quad \text{in } \Omega \tag{52}$$

$$\nabla \cdot \vec{v} = 0 \quad \text{in } \Omega \tag{53}$$

Here, $\vec{v}$ is the velocity of the mantle in the subduction zone wedge, Ω, and P is the pressure. Imposing isothermal conditions means that (14) has been dropped altogether. With these simplifications, we can test our numerical solution to (52) and (53) against the analytical solution provided by Batchelor (1967).

Analytical solution To more easily describe the analytical solution, we consider the cornerflow geometry in Fig. 5a where we have rotated the mantle wedge by 90° counterclockwise and assumed a 90° angle between the wedge boundaries. In this geometry Eqs. (52) and (53) can be transformed into a biharmonic equation for the stream function, ψ,

$$\nabla^4\psi = 0 \tag{54}$$

where $\psi = \psi(r,\theta)$ is a function of the radius, r, and angle from the x-axis, θ, related to the velocity, $\vec{v} = \vec{v}(x,y)$ by

$$\vec{v} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \frac{1}{r}\frac{\partial\psi}{\partial\theta} \\ -\frac{\partial\psi}{\partial r} \end{pmatrix} \tag{55}$$

With semi-infinite x and y axes, a rigid boundary condition, $\vec{v} = \mathbf{0}$, along the y-axis (the rotated "crust" at the top of the wedge), and a kinematic boundary condition on the x-axis (the "slab" surface at the base of the wedge), $\vec{v} = (\mathrm{U}, 0)^T$, the analytical solution is found as

$$\psi(r,\theta) = \frac{r\mathrm{U}}{\frac{1}{4}\pi^2 - 1}\left(-\frac{1}{4}\pi^2\sin\theta + \frac{1}{2}\pi\theta\sin\theta + \theta\cos\theta\right) \tag{56}$$

Discretization Since it is not possible with our numerical approach to solve the equations in a semi-infinite domain, we discretize (52) and (53) in a unit square domain with unit length in the x and y domains, as in Fig. 5b. We choose different function spaces, with different shape functions, $\vec{\omega}_j(x)$ and $\chi_j(x)$ for the approximations of $\vec{v}$ and P, respectively, such that

$$\vec{v} \approx \tilde{\vec{v}} = \sum_j \omega_j^k v_j^k \tag{57}$$

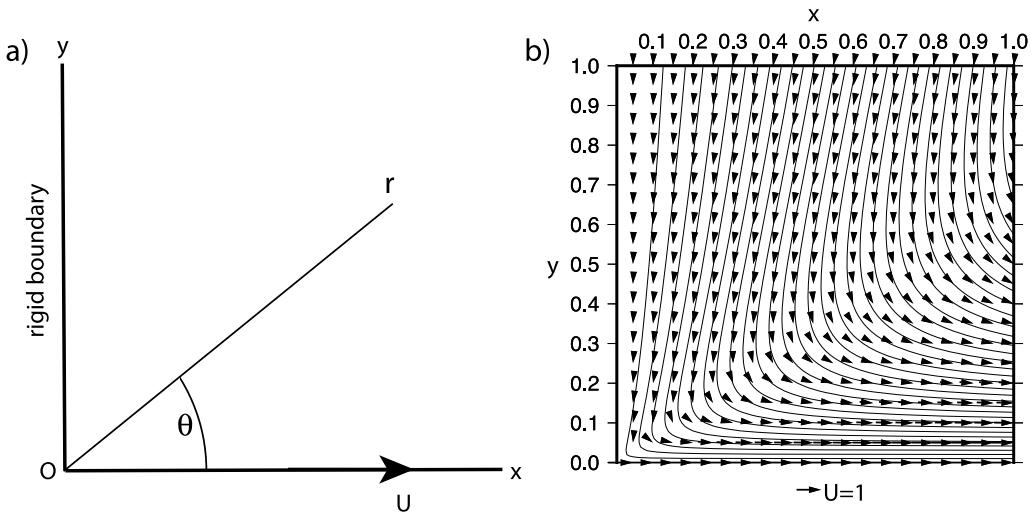

Fig. 5 Batchelor cornerflow geometry and example model solution. **a** Specification of Cartesian (x, y) and polar (r, θ) coordinate systems as well as boundary conditions. **b** Solution for ψ and $\vec{v}$ on geometry $\Omega = [0, 1] \times [0, 1]$ with U = 1. Stream function contours are at arbitrary intervals

$$P \approx \tilde{P} = \sum_j \chi_j P_j \tag{58}$$

where v_j^k and P_j are the values of velocity and pressure at node j, respectively, and the superscript k represents the spatial component of $\vec{v}$. The discrete test functions $\tilde{\vec{v}}_t$ and $\tilde{P}_t$ are similarly defined. We will discuss the choice of $\vec{\omega}_j = \omega_j^k$ and χ_j later but simply assume that they are continuous across elements of the mesh in the following.

Boundary conditions To match the analytical solution (56), we apply essential Dirichlet conditions on $\tilde{\vec{v}}$ on all four sides of the domain

$$\tilde{\vec{v}} = (0, 0)^T \quad \text{on } \partial\Omega \text{ where } x = 0 \tag{59}$$

$$\tilde{\vec{v}} = (\mathrm{U}, 0)^T \text{ on } \partial\Omega \text{ where } y = 0 \tag{60}$$

$$\tilde{\vec{v}} = \vec{v} \quad \text{on } \partial\Omega \text{ where } x = 1 \text{ or } y = 1 \tag{61}$$

Note that the first two conditions imply a discontinuity in the solution for $\tilde{\vec{v}}$ at $(x, y) = (0, 0)$. The last boundary condition simply states that we apply the analytical solution (obtained from (56) via (55)) at the boundaries at $x = 1$ and $y = 1$. One consequence of applying essential boundary conditions on $\vec{v}$ on all sides of the domain is that P is unconstrained up to a constant value as only its spatial derivatives appear in the equations. The ability to add an arbitrary constant to the pressure is referred to as the pressure containing a null space. This makes it impossible to find a unique solution to (52) and (53) with (59)–(61) since an infinite number of pressure solutions exist. There are a number of ways to select an appropriate pressure solution. Here, we arbitrarily choose one such solution by adding the condition that

$$\tilde{P} = 0 \quad \text{at } (x, y) = (0, 0) \tag{62}$$

which will allow a unique solution to the discrete equations to be found.

Weak form Multiplying (52) by $\vec{v}_t$ and (53) by P_t, integrating (by parts) over Ω, and discretizing the test and trial functions allows the discrete matrix–vector system of the form of (8) to be written as

$$\mathbf{S} = \begin{pmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{pmatrix} \tag{63}$$

$$\mathbf{K} = K_{i_1 j_1} = \sum_k \int_{e_k} \left(\frac{\nabla \vec{\omega}_{i_1} + \nabla \vec{\omega}_{i_1}^T}{2} \right) : \left(\frac{\nabla \vec{\omega}_{j_1} + \nabla \vec{\omega}_{j_1}^T}{2} \right) \mathrm{d}x \tag{64}$$

$$\mathbf{G} = G_{i_1 j_2} = -\sum_k \int_{e_k} \nabla \cdot \vec{\omega}_{i_1} \chi_{j_2} \mathrm{d}x \tag{65}$$

$$\mathbf{D} = D_{i_2 j_1} = -\sum_k \int_{e_k} \chi_{i_2} \nabla \cdot \vec{\omega}_{j_1} \mathrm{d}x \tag{66}$$

$$\mathbf{u} = (\mathbf{v}, \mathbf{P})^T = \left(\vec{v}_{j_1}, P_{j_2}\right)^T \tag{67}$$

$$\mathbf{f} = f_i = 0 \tag{68}$$

Note that in (64)–(66) all surface integrals around $\partial\Omega$ arising from integration by parts have been dropped because the velocity solution is fully specified on all boundaries. Additionally, when integrating (64) by parts we have used the fact that $\nabla\vec{\omega}_{i_1} : \left(\frac{\nabla\vec{\omega}_{j_1}+\nabla\vec{\omega}_{j_1}^T}{2}\right) = \left(\frac{\nabla\vec{\omega}_{i_1}+\nabla\vec{\omega}_{i_1}^T}{2}\right) : \left(\frac{\nabla\vec{\omega}_{j_1}+\nabla\vec{\omega}_{j_1}^T}{2}\right)$ to demonstrate the symmetry of $\mathbf{K}$. In fact, $\mathbf{S}$ has been made symmetric by integrating the gradient of pressure term, ∇P, by parts in (65) and negating (53) in (66) such that $\mathbf{G} = \mathbf{D}^T$. This symmetry property can be exploited when choosing an efficient method of solving (8).

As before, the weak form of (63) may be described using UFL with rather simple python code shown in Listing 3. For the sake of brevity, we have assumed that the test and trial functions `v_t`, `p_t`, `v_a` and `p_a` have been declared. Additional code is also required to fully describe the boundary conditions and solve the resulting system. The full example is provided in the material contained in the zenodo repository referenced in the data availability statement as a TerraFERMA input file.

An important aspect of $\mathbf{S}$ is that it describes a so-called saddle point system. The lower right block is zero, which indicates that pressure is acting in this system as a Lagrange multiplier, enforcing the constraint that the velocity is divergence free but not appearing in (53) itself. Such systems require special consideration of the choice of shape functions for the discrete approximations of velocity and pressure to ensure the stability of the solution, $\mathbf{u}$. Several choices of so-called stable element pairs, $(\vec{\omega}_j, \chi_j)$ are available in the literature (e.g., Auricchio et al. 2017). Here we select the frequently used lowest order Taylor–Hood element pair, in which $\vec{\omega}_j$ are piecewise-quadratic and χ_j are piecewise-linear polynomials, referred to on triangular (and tetrahedral in 3D) meshes as P2P1. This fulfills a necessary (but not sufficient) criterion for stability that the velocity has more DOFs than the pressure. Solving (63)–(68) subject to the conditions (59)–(62) on a series of successively finer meshes and comparing the resulting solution to the analytical result given by (56) and (55) using the error metric

$$e_{L^2,B} = \sqrt{\int_\Omega (\tilde{\vec{v}} - \vec{v}) \cdot (\tilde{\vec{v}} - \vec{v}) \mathrm{d}x} \tag{69}$$

(where B stands for Batchelor) shows linear rather than quadratic convergence. We encourage the readers to convince themselves of this by running the example. This first-order convergence rate is lower than would be expected for piecewise-quadratic velocity functions. This drop in convergence is caused by the boundary conditions at the origin being discontinuous, which cannot be represented in the selected function space and results in a pressure singularity at that point. This is an example where convergence analysis demonstrates suboptimal results due to our inability to represent the solution in the selected finite element function space.

2.2.6 Blankenbach thermal convection benchmark

Before discussing the solution of the full governing equations for subduction zone thermal structure, we will explore solving the equations governing a buoyancy-driven convection model in a square domain following the steady-state mantle convection benchmarks from Blankenbach et al. (1989). This example allows us to couple a steady-state advection–diffusion equation

```
K = inner(sym(grad(v_t)), sym(grad(v_a)))*dx
G = -div(v_t)*p_a*dx
D = -p_t*div(v_a)*dx
S = K + G + D
```

Listing 3 Weak form of Stokes system

for temperature to the Stokes and mass conservation equations we have already discussed. This also provides an example of solving a nonlinearly coupled system and will show how we can test a model for which no analytical solution exists.

The flow in the box is driven by heating from below and cooling from above (Fig. 6). We solve (12) and (13)

$$-\nabla\cdot\left(2\eta\frac{\nabla\vec{v}+\nabla\vec{v}^T}{2}\right)+\nabla P=-\mathrm{Ra}T\hat{\vec{g}}\quad\text{in }\Omega \tag{70}$$

$$\nabla\cdot\vec{v}=0\quad\text{in }\Omega \tag{71}$$

where variable rheology is permitted through the inclusion of the viscosity η and the buoyancy force vector has been defined as $\vec{f}_B=-\mathrm{Ra}T\hat{\vec{g}}$, using the temperature T, nondimensional Rayleigh number, Ra, and unit vector in the direction of gravity, $\hat{\vec{g}}$. The Rayleigh number arises from the nondimensionalization of the governing equations and is a ratio that balances factors that enhance convective vigor (e.g., thermal expansivity, gravity) with those that retard convective vigor (e.g., viscosity). In general, convective vigor increases with increasing Ra when it exceeds a critical value for the Rayleigh number (see, e.g., Turcotte and Schubert 2002). The heat equation (14), under the assumptions of steady state ($\frac{\partial T}{\partial t}=0$), constant material properties ($k=1$) and zero internal heating ($H=0$), reads

$$\vec{v}\cdot\nabla T=\nabla^2 T\quad\text{in }\Omega \tag{72}$$

Boundary conditions We discretize the trial function spaces for temperature ($T\approx\tilde{T}$), velocity ($\vec{v}\approx\tilde{\vec{v}}$) and pressure ($P\approx\tilde{P}$) as before using (41), (57) and (58), with similarly defined discrete test functions, $\tilde{T}_t$, $\tilde{\vec{v}}_t$ and $\tilde{P}_t$. For the Stokes problem, we assume free-slip boundaries. These are formed by the combination of a Dirichlet boundary condition of zero normal velocity ($v_n=\tilde{\vec{v}}\cdot\hat{\vec{n}}=0$) and a Neumann zero tangential stress condition ($\tau_t=(\boldsymbol{\tau}\cdot\hat{\vec{n}})\cdot\hat{\vec{t}}=0$). Here, $\hat{\vec{n}}$ is the unit normal to the boundary, $\hat{\vec{t}}$ is the unit tangent on the boundary (see Fig. 6a), and $\boldsymbol{\tau}$ is the deviatoric stress tensor

$$\boldsymbol{\tau}=2\eta\frac{\nabla\tilde{\vec{v}}+\nabla\tilde{\vec{v}}^T}{2}=2\eta\begin{bmatrix}\frac{\partial\tilde{v}_x}{\partial x} & \frac{1}{2}\left(\frac{\partial\tilde{v}_x}{\partial y}+\frac{\partial\tilde{v}_y}{\partial x}\right)\\ \frac{1}{2}\left(\frac{\partial\tilde{v}_x}{\partial y}+\frac{\partial\tilde{v}_y}{\partial x}\right) & \frac{\partial\tilde{v}_y}{\partial y}\end{bmatrix} \tag{73}$$

This set of velocity boundary conditions once again results in a pressure null space. We arbitrarily choose to impose the extra condition that $\tilde{P}(0,0)=0$ to force a unique solution to exist. For the heat equation, the side boundaries are insulating (imposed by the Neumann boundary condition $\partial\tilde{T}/\partial x=0$) with Dirichlet boundary conditions for the top boundary ($\tilde{T}=0$) and bottom boundary ($\tilde{T}=1$).

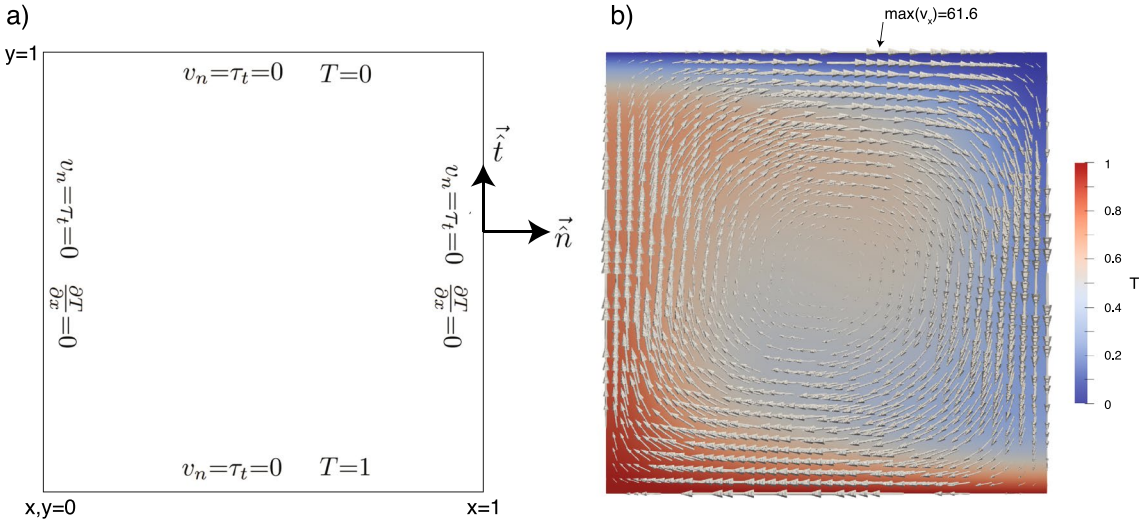

Fig. 6 **a** Thermal convection benchmark description. **b** Select model solution for case 1a from Blankenbach et al. (1989)

Nonlinearity Unlike the previous examples, which were linear problems of their solution variables, (70)–(72) are nonlinear. For an isoviscous rheology, the equations are individually linear but the buoyancy contribution to (70) and the advective component in (72) mean that the coupled system of equations is nonlinear, with $\vec{v}$ depending on T and vice versa. For non-Newtonian rheologies, where $\eta{=}\eta(\vec{v})$, (70) itself becomes nonlinear too. Because of this, rather than immediately defining the weak forms of the linear operator $\mathbf{S}$ we begin by considering the weak form of the nonlinear residual, $\mathbf{r}$. This is derived in exactly the same manner as before by multiplying (70) by $\vec{v}_t$, (71) by P_t and (72) by T_t, discretizing the functions, integrating (by parts) over the domain Ω, dropping the resulting surface integrals (either to enforce the weak boundary conditions or because they are unnecessary due to the essential boundary conditions), and defining the discrete weak forms as

$$\begin{aligned}\mathbf{r}_{\vec{v}} = r_{\vec{v}_{i_1}} := \sum_k \int_{e_k} \Bigg[\left(\frac{\nabla\vec{\omega}_{i_1} + \nabla\vec{\omega}_{i_1}^T}{2} \right) \\ : 2\eta \left(\frac{\nabla\tilde{\vec{v}} + \nabla\tilde{\vec{v}}^T}{2} \right) \\ -\nabla\cdot\vec{\omega}_{i_1}\tilde{P} + \vec{\omega}_{i_1}\cdot\hat{\vec{g}}\mathrm{Ra}\tilde{T} \Bigg] \mathrm{d}x = 0\end{aligned} \tag{74}$$

$$\mathbf{r}_P = r_{P_{i_2}} := -\sum_k \int_{e_k} \chi_{i_2} \nabla\cdot\tilde{\vec{v}} \mathrm{d}x = 0 \tag{75}$$

$$\mathbf{r}_T = r_{T_{i_3}} := \sum_k \int_{e_k} \left[\phi_{i_3}\tilde{\vec{v}}\cdot\nabla\tilde{T} + \nabla\phi_{i_3}\cdot\nabla\tilde{T} \right] \mathrm{d}x = 0 \tag{76}$$

Here, $\mathbf{r} = (\mathbf{r}_{\vec{v}}, \mathbf{r}_P, \mathbf{r}_T)^T = \left(r_{\vec{v}_{i_1}}, r_{P_{i_2}}, r_{T_{i_3}} \right)^T$ is a residual vector, the root of which must be found in order to find an approximate solution to (70)–(72). Finding the exact root is not generally possible. Instead, we aim to find $\mathbf{r}{=}\mathbf{0}$ within some tolerance. For example, we can use an L^2 norm and an absolute $||\mathbf{r}||_2 = \sqrt{\mathbf{r}\cdot\mathbf{r}} < \epsilon_{\mathrm{atol}}$, or relative, $\frac{||\mathbf{r}||_2}{||\mathbf{r}^0||_2} = \frac{\sqrt{\mathbf{r}\cdot\mathbf{r}}}{\sqrt{\mathbf{r}^0\cdot\mathbf{r}^0}} < \epsilon_{\mathrm{rtol}}$, tolerance, where $\mathbf{r}^0$ is the residual evaluated using the initial guess at the solution. We will briefly discuss two commonly used approaches to approximately finding the residual root.

Newton's method To find the root, $\mathbf{u}^{i+1}{=}\left(\vec{\mathbf{v}}^{i+1}, \mathbf{P}^{i+1}, \mathbf{T}^{i+1}\right)^T = \left(\vec{v}_{j_1}^{i+1}, P_{j_2}^{i+1}, T_{j_3}^{i+1}\right)^T$, we can expand the residual in a Taylor series around the current best guess at the solution $\mathbf{u}^i{=}\left(\vec{\mathbf{v}}^i, \mathbf{P}^i, \mathbf{T}^i\right)^T{=}\left(\vec{v}_{j_1}^i, P_{j_2}^i, T_{j_3}^i\right)^T$ such that

$$\begin{aligned}\mathbf{r}\left(\mathbf{u}^{i+1}\right) = \mathbf{r}\left(\mathbf{u}^i\right) + \mathbf{r}'\left(\mathbf{u}^i\right)\left(\mathbf{u}^{i+1} - \mathbf{u}^i\right) \\ + \mathbf{r}''\left(\mathbf{u}^i\right)\left(\mathbf{u}^{i+1} - \mathbf{u}^i\right)^2 + \ldots = \mathbf{0}\end{aligned} \tag{77}$$

where $\mathbf{r}'\left(\mathbf{u}^i\right)$ and $\mathbf{r}''\left(\mathbf{u}^i\right)$ represent the first- and second-order derivatives of the residual with respect to the solution variables, evaluated at $\mathbf{u}^i$. Dropping terms with orders higher than first, defining the Jacobian $\mathbf{J}\left(\mathbf{u}^i\right) = \mathbf{r}'\left(\mathbf{u}^i\right)$ and $\delta u = \mathbf{u}^{i+1} - \mathbf{u}^i$, and rearranging results in the matrix equation

$$\mathbf{J}\left(\mathbf{u}^i\right)\delta\mathbf{u} = -\mathbf{r}\left(\mathbf{u}^i\right) \tag{78}$$

which can be solved for $\delta\mathbf{u}$ and used to find $\mathbf{u}^{i+1} = \mathbf{u}^i + \delta\mathbf{u}$. Since we have dropped terms from the Taylor expansion, $\mathbf{u}^{i+1}$ will only be a first-order approximation of the root of $\mathbf{r}$. So long as the initial guess $\mathbf{u}^i$ is close enough to the final solution and (78) is solvable, then $\mathbf{u}^{i+1}$ should give a better estimate of $\mathbf{r} = \mathbf{0}$, in the sense that $\mathbf{r}\left(\mathbf{u}^{i+1}\right) < \mathbf{r}\left(\mathbf{u}^i\right)$. Repeatedly solving (78) and at each iteration updating $\mathbf{u}^{i+1} \rightarrow \mathbf{u}^i$ will then result in a final solution where $\mathbf{r}$ approaches $\mathbf{0}$ in some norm and to some tolerance.

For highly nonlinear problems, the Jacobian matrix, $\mathbf{J} = \mathbf{r}'$, can be complicated and difficult to derive, let alone to code. Fortunately, modern finite element libraries, like FEniCS, that provide the symbolic and human-readable representation of weak forms, seen above through UFL, allow the Jacobian to be automatically evaluated and assembled. For (74)–(76), this results in the code snippet in Listing 4.

For the sake of brevity, we have assumed that the most recent iterated solutions, `v_i`, `p_i` and `T_i`, and test

```
rv = (inner(sym(grad(v_t)), 2*eta*sym(grad(v_i))) - div(v_t)*p_i \
      + inner(v_t, gravity)*Ra*T_i)*dx
rp = -p_t*div(v_i)*dx
rT = (T_t*inner(v_i, grad(T_i)) + inner(grad(T_t), grad(T_i)))*dx
r = rv + rp + rT
J = derivative(r, u_i, u_a)
```

Listing 4 Weak form of Stokes thermal convection system

functions, `v_t`, `p_t` and `T_t`, have been declared. The individual solutions are part of a larger system solution, `u_i`=(`v_i`, `p_i`,`T_i`), and a trial function for the system also exists, `u_a`=(`v_t`, `p_t`, `T_t`). Additionally, the unit vector in the direction of gravity, `gravity`, the Rayleigh number, `Ra`, and the viscosity, `eta`, have been declared with the latter either being 1 in the isoviscous case or a function of temperature, `T_i`, in the temperature-dependent case. In either case, the Jacobian matrix, `J`, is easily obtained using the `derivative` function. Using this and the residual `r` allow (78) to be repeatedly solved for `u_i` until convergence is achieved and the root of the residual found.

Picard's method Convergence of the Newton iteration method depends on having a good initial guess, which is not always possible, especially when solving steady-state problems like (70)–(72). In this case, an alternative approach is to use a Picard iteration. This splits the equations into multiple linearized subsets and solves them sequentially and repeatedly, updating the nonlinear terms at each iteration, until convergence is achieved. Equations (70)–(72) can be split into two systems of the form of (8), the first for the Stokes system

$$\mathbf{S}_s = \begin{pmatrix} \mathbf{K}_s & \mathbf{G}_s \\ \mathbf{D}_s & \mathbf{0} \end{pmatrix} \tag{79}$$

$$\mathbf{K}_s = K_{s_{i_1 j_1}} = \sum_k \int_{e_k} \left(\frac{\nabla\vec{\omega}_{i_1} + \nabla\vec{\omega}_{i_1}^T}{2} \right) : 2\eta \left(\frac{\nabla\vec{\omega}_{j_1} + \nabla\vec{\omega}_{j_1}^T}{2} \right) \mathrm{d}x \tag{80}$$

$$\mathbf{G}_s = G_{s_{i_1 j_2}} = -\sum_k \int_{e_k} \nabla\cdot\vec{\omega}_{i_1}\chi_{j_2}\,\mathrm{d}x \tag{81}$$

$$\mathbf{D}_s = D_{s_{i_2 j_1}} = -\sum_k \int_{e_k} \chi_{i_2}\nabla\cdot\vec{\omega}_{j_1}\,\mathrm{d}x \tag{82}$$

$$\mathbf{u}_s = (\mathbf{v},\mathbf{P})^T = \left(\vec{v}_{j_1}, P_{j_2}\right)^T \tag{83}$$

$$\mathbf{f}_s = f_{s_{i_1}} = -\sum_k \int_{e_k} \vec{\omega}_{i_1}\cdot\hat{\tilde{g}}\mathrm{Ra}\tilde{T}\,\mathrm{d}x \tag{84}$$

and the second for the temperature equation

$$\mathbf{S}_T = S_{T_{ij}} = \sum_k \int_{e_k} \left(\phi_i \tilde{\tilde{\nu}}\cdot\nabla\phi_j + \nabla\phi_i\cdot\nabla\phi_j \right) \mathrm{d}x \tag{85}$$

$$\mathbf{u}_T = \mathbf{T} = T_j \tag{86}$$

$$\mathbf{f}_T = f_{T_i} = 0 \tag{87}$$

For UFL code snippets of (79) and (84), see Listing 5 and for (85), see Listing 6.

The full system solution vector remains $\mathbf{u} = (\mathbf{u}_s, \mathbf{u}_T)^T = (\vec{\mathbf{v}}, \mathbf{P}, \mathbf{T})^T$, and the best guess at the solution is $\mathbf{u}^i$. $\mathbf{S}_s(\mathbf{u}^i)\mathbf{u}_s^{i+1} = \mathbf{f}_s(\mathbf{u}_s^i)$ is solved for $\mathbf{u}_s^{i+1}$, which is used to update $\mathbf{u}$ such that $\mathbf{u}_s^{i+1} \rightarrow \mathbf{u}_s^i$ before solving $\mathbf{S}_T(\mathbf{u}^i)\mathbf{u}_T^{i+1} = \mathbf{0}$ for an updated solution for temperature, $\mathbf{u}_T^{i+1}$. Repeating this iteration will generally find the root of the residuals (74)–(76) and once again the iteration is repeated until $\mathbf{r} = \mathbf{0}$ in some norm and to some tolerance.

```
Ks = inner(sym(grad(v_t)), 2*eta*sym(grad(v_a)))*dx
Gs = -div(v_t)*p_a*dx
Ds = -p_t*div(v_a)*dx
Ss = Ks + Gs + Ds
fs = -inner(v_t, gravity)*Ra*T_i*dx
```

Listing 5 Weak form of Stokes system

```
ST = (T_t*inner(v_i, grad(T_a)) + inner(grad(T_t), grad(T_a)))*dx
```

Listing 6 Weak form of Stokes thermal convection system

If the initial guess is sufficiently good, then Newton should converge quadratically while a Picard iteration will converge at a lower rate. However neither convergence nor the convergence rate of either method is guaranteed. Various methods are available for solutions that do not converge. These include finding a better initial guess (e.g., a solution from a case with lower convective vigor), "relaxing" the solution by only applying a partial update at each iteration, or linearizing terms in the Jacobian matrix. It should also be noted that, if applied to the linear problems discussed in previous sections, any nonlinear iteration should converge in a single iteration.

Diagnostics The geometry and expressions for the boundary conditions for the selected Blankenbach et al. (1989) cases are shown in Fig. 6a, and a converged model solution for temperature and velocity obtained for Ra=10^4 (benchmark case 1a from Blankenbach et al. 1989) is shown in Fig. 6b. To quantify the precision with which the governing equations can be solved, we focus on two measures of convective vigor. The first is the Nusselt number Nu which is the integrated nondimensional surface heatflow

$$\mathrm{Nu} \;=\; -\int_{x=0}^{x=1} \frac{\partial T}{\partial y}(x, y=1)\,\mathrm{d}x \tag{88}$$

The second is the root-mean-square velocity V_{rms} defined as

$$V_{\mathrm{rms}} \;=\; \sqrt{\frac{\int_\Omega \vec{v}\cdot\vec{v}\mathrm{d}x}{\int_\Omega \mathrm{d}x}} \tag{89}$$

Table 9 in Blankenbach et al. (1989) specifies their best estimates for various quantities of the benchmark. We will focus on Nu and V_{rms} and show results for their steady-benchmarks 1a–1c (isoviscous, $\eta = 1$, with Ra increasing from 10^4 to 10^5 and 10^6) and benchmark 2a which has Ra = 10^4 and a temperature-dependent viscosity $\eta(T) = \exp(-bT)$ with $b = \ln(10^3)$ (see Table 1).

Discretization For the Stokes equation, TerraFERMA uses the P2P1 Taylor–Hood Lagrange element pair for the shape functions $(\vec{\omega}_j, \chi_j)$ (as in Sect. 2.2.5) and P2 elements for the heat equation (ϕ_j). The choice of elements here can be tersely described as P2P1P2. In TerraFERMA, we apply a Newton iteration to cases 1a–c with a harmonic perturbation to the conductive state $T(x, y) = 1 - y + 0.1\cos\pi x \sin\pi y$ as an initial guess for temperature and the solution to $\mathbf{S}_s\mathbf{u}_s = \mathbf{f}_s$ given the initial T as a first guess for velocity and pressure. We use a Picard iteration and an isoviscous initial velocity and pressure guess for case 2a owing to the difficulty getting Newton to converge without a better initial guess. Both are solved to a relative tolerance, ϵ_{rtol}, of 10^{-9}.

We also show results obtained with Sepran using the same P2P1P2 discretization as in TerraFERMA. The same initial guess is used for case 1a, but for 1b and 2a we use the final solution from 1a as an initial guess and for 1c we use the final solution of 1b. Picard iteration is used for all cases to a relative tolerance of 10^{-9}.

Results We obtain results for grids with 32, 64, 128, and 256 elements on a side. The TerraFERMA results have grid refinement toward the edges of the domain to allow for better resolution of the thermal boundary layer at a lower number of grid points. The Sepran results are obtained on equidistant meshes where the computation of (88) is improved following the method of Ho-Liu et al. (1987). We follow Blankenbach et al. (1989) in using Richardson extrapolation to attempt to find the "best" estimate as shown in comparison with theirs in Table 1. We make estimates from the modeling approaches independently and average them to find the "new" results. A brief inspection suggests that the estimates made in 1989 were clearly rather precise!

Figure 7 shows how our model predictions trend toward our average extrapolated values. Note that these are not convergence plots like those used previously when we compared the approximate solution to

Table 1 Best values from Blankenbach et al. (1989) and our averaged extrapolated values from current models for selected benchmark values (see text)

Case	Ra	η	Blankenbach et al. (1989)		Updated estimates	
			Nu	V_{rms}	Nu	V_{rms}
1a	10^4	1	4.884409	42.864947	4.88440907	42.8649484
1b	10^5	1	10.534095	193.21454	10.53404	193.21445
1c	10^6	1	21.972465	833.98977	21.97242	833.9897
2a	10^4	$e^{-\ln(10^3)T}$	10.0660	480.4334	10.06597	480.4308

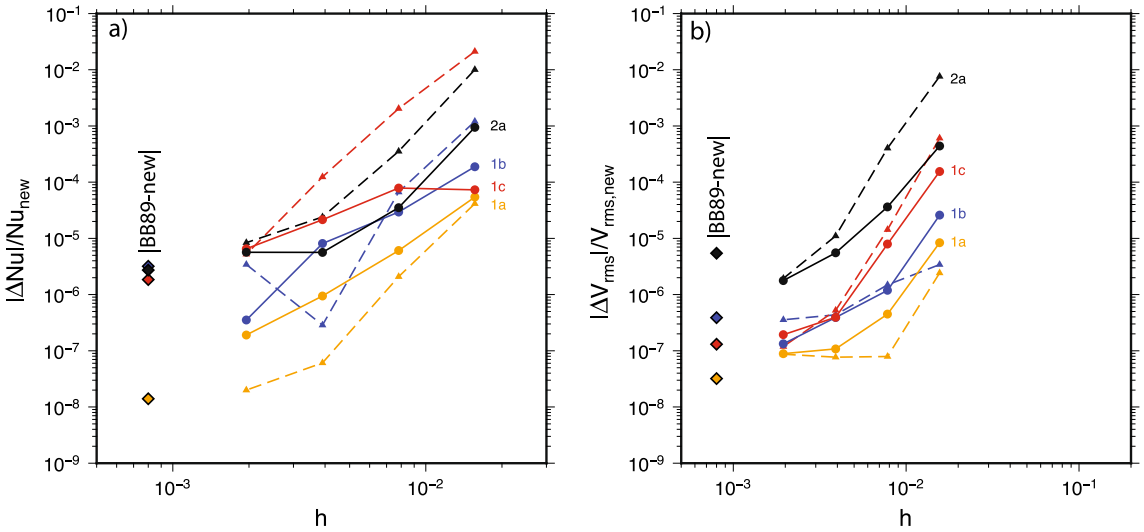

Fig. 7 Convergence characteristics for *Nu* (frame **a**) and V_{rms} (frame **b**) for the thermal convection benchmarks 1a–1c and 2a. Dashed lines with triangular symbols denote Sepran results. TerraFERMA results are shown by the solid lines with circles. The difference between the original Blankenbach "best" estimates and our new extrapolated results is shown by the diamonds at an arbitrary point on the x-axis

the analytical solution. Here, the best estimates do not represent metrics obtained from an analytical solution. Some of the flattening or 'V'-ing in the curves is due to the change in sign of the difference between the modeled and extrapolated values. In general, the difference between approximate solution and extrapolated value is smaller at lower convective vigor (compare 1a and 1c) and larger with stronger nonlinearities (compare 1a and 2a).

2.3 FEM determination of SZ thermal structure

2.3.1 Recap of the governing equations

While we already encountered examples of solution of the governing equations (12)–(14), we will formulate the full set of equations for subduction zone thermal structure below for clarity and completeness' sake. We will set up the parameters and equations in a general form that we will use in part III for a global suite of models (similar to those in Syracuse et al. (2010) and Wada and Wang (2009)) but restrict ourselves in this part to applying them to a simplified benchmark problem. The equations will be introduced in dimensional form before nondimensionalizing them in Sect. 2.3.2. All dimensional variables will be indicated by a superscript *. Dimensional reference values will be indicated by the subscript $_0$. We assume a 2D Cartesian coordinate system with coordinates $\vec{x}^* = (x_1^*, x_2^*)^T = (x^*, y^*)^T = (x^*, -z^*)^T$ where z^* is depth.

Conservation of mass under the assumption that the fluid is incompressible leads to

$$\nabla^* \cdot \vec{v}^* = 0 \tag{90}$$

where, in two-dimensions, $\vec{v}^* = (v_1^*, v_2^*)^T = (v_x^*, v_y^*)^T$ is the velocity vector. Assuming all flow is driven by a kinematic boundary condition, conservation of momentum leads to the dimensional Stokes equation without buoyancy forces

$$-\nabla^* \cdot \boldsymbol{\tau}^* + \nabla^* P^* = 0 \tag{91}$$

where P^* is the dynamic pressure and $\boldsymbol{\tau}^*$ is the deviatoric stress tensor given by

$$\boldsymbol{\tau}^* = 2\eta^* \dot{\boldsymbol{\epsilon}}^* \tag{92}$$

Here, η^* is dynamic viscosity and $\dot{\boldsymbol{\epsilon}}^*$ is the deviatoric strain rate tensor with components

$$\dot{\epsilon}_{ij}^* = \frac{1}{2}\left[\frac{\partial v_i^*}{\partial x_j^*} + \frac{\partial v_j^*}{\partial x_i^*}\right] \tag{93}$$

The time-dependent dimensional heat equation is given by

$$\rho^* c_{\mathrm{p}_0}\left(\frac{\partial T^*}{\partial t^*} + \vec{v}^* \cdot \nabla^* T^*\right) = \nabla^* \cdot \left(k^* \nabla^* T^*\right) + H^* \tag{94}$$

while, in cases where a steady state is assumed ($\frac{\partial T^*}{\partial t^*} = 0$) temperature is governed by

$$\rho^* c_{\mathrm{p}_0} \vec{v}^* \cdot \nabla^* T^* = \nabla^* \cdot \left(k^* \nabla^* T^*\right) + H^* \tag{95}$$

where ρ^* is density, c_{p_0} is the heat capacity at constant pressure (assumed constant), T^* is temperature, k^* is thermal conductivity, and H^* is volumetric heat production. In this paper, we will assume that the viscosity η^* is either constant, $\eta^* = \eta_0$, or is a function of temperature and strain rate following a simplified creep law for dislocation creep in dry olivine from Karato and Wu (1993)

$$\eta^*_{\rm disl} = A^*_\eta \exp\left(\frac{E^*}{nR^*(T^* + T^*_a)}\right)\dot\epsilon^{*\frac{1-n}{n}}_{II} \tag{96}$$

where A^*_η is a prefactor, E^* is the activation energy, R^* is the gas constant, n is a power-law index, T^*_a a linear approximation of an adiabatic temperature using a gradient of 0.3°C/km with $T^*_a = 0$ at the top of the model (which may not be at $z^* = 0$ due to assumptions of ocean bathymetry as we will see in Sect. 2.3.3) and $\dot\epsilon^*_{II}$ is the second invariant of the deviatoric strain rate tensor (also known as the effective deviatoric strain rate)

$$\dot\epsilon^*_{II} = \sqrt{\frac{1}{2}\dot{\boldsymbol\epsilon}^* : \dot{\boldsymbol\epsilon}^*} \tag{97}$$

Since the dynamical range of the viscosity (96) is large over the temperature contrast across subduction zones, it is common practice to cap the viscosity at some arbitrary maximum $\eta^*_{\rm max}$ so that in the variable viscosity case

$$\eta^* = \left(\frac{1}{\eta^*_{\rm disl}} + \frac{1}{\eta^*_{\rm max}}\right)^{-1} \tag{98}$$

2.3.2 Nondimensionalization

It is attractive to nondimensionalize the equations such that most quantities are scaled to be close to 1. This provides simple scaling arguments to allow for understanding which terms in the equations are dominant, avoids computer algebra that mixes very large and very small numbers, and provides for the formation of a matrix–vector system where the condition number of the matrix (Golub and Van Loan 1989) is more optimal.

Table 2 provides a list of dimensional reference values, dimensional parameters and their nondimensional equivalents. For the nondimensionalization of (90)–(98) we use the diffusional time scaling with nondimensional time defined as $t = t^*\kappa_0/h_0^2$ where h_0 is the reference length scale and κ_0 is the reference thermal diffusivity. With $\vec{x} = \vec{x}^*/h_0$ it follows $\vec{v} = \vec{v}^* h_0/\kappa_0$, $\dot\epsilon = \dot\epsilon^* h_0^2/\kappa_0$ and $\nabla = \nabla^* h_0$. We further introduce $T = (T^* - T^*_s)/T_0$, $k = k^*/k_0$, $\rho = \rho^*/\rho_0$, $P = P^* h_0^2/(\kappa_0\eta_0)$ and $H = H^* h_0^2/(\rho_0 c_{p_0} T_0 \kappa_0)$. Note that our choices of T_0 and h_0 in Table 2 cause the numerical values of dimensional position (in km) and temperature (in °C) to have the same magnitude as the corresponding nondimensional quantities. Substitution of the nondimensional variables and constants leads to the following set of nondimensional equations for pressure and velocity

$$\nabla \cdot \vec{v} = 0 \tag{99}$$

Table 2 Nomenclature and reference values

Quantity	Symbol	Nominal value	Nondimensional value
Reference temperature scale	T_0	1 K = 1°C	–
Surface temperature	T^*_s	273 K = 0°C	T_s=0
Mantle temperature	T^*_m	1623 K = 1350°C	T_m=1350
Surface heat flow [c]	q^*_s	[§] W/m^2	q_s [§]
Reference density	ρ_0	3300 kg/m^3	–
Crustal density [c]	ρ^*_c	2750 kg/m^3	$\rho_c = 0.833333$
Mantle density	ρ^*_m	3300 kg/m^3	$\rho_m = 1$
Reference thermal conductivity	k_0	3.1 W/(m K)	–
Crustal thermal conductivity [c]	k^*_c	2.5 W/(m K)	$k_c = 0.8064516$
Mantle thermal conductivity	k^*_m	3.1 W/(m K)	$k_m = 1$
Volumetric heat production (upper crust) [c]	H^*_1	1.3 μW/m^3	$H_1 = 0.419354$
Volumetric heat production (lower crust) [c]	H^*_2	0.27 μW/m^3	$H_2 = 0.087097$
Age of overriding crust [o]	A^*_c	[§] Myr	A_c [§]
Age of subduction [t]	A^*_s	[§] Myr	A_s [§]
Age of subducting slab	A^*	[§] Myr	A [§]
Reference length scale	h_0	1 km	–
Depth of base of upper crust [c]	z^*_1	15 km	$z_1 = 15$

Table 2 (continued)

Quantity	Symbol	Nominal value	Nondimensional value
Depth of base of lower crust (Moho)	z_2^*	[§] km	z_2 [§]
Trench depth	z_{trench}^*	[§] km	$z_{\text{trench}}^{\S}$
Position of the coast line	x_{coast}^*	[§] km	$x_{\text{coast}}^{\S}$
Wedge inflow/outflow transition depth	z_{io}^*	[§] km	$z_{\text{io}}^{\S}$
Depth ofdomain	D^*	[§] km	$D^{\S}$
Width of domain	L^*	[§] km	$L^{\S}$
Depth of change from decoupling to coupling	d_c^*	80 km	$d_c = 80$
Reference heat capacity	c_{p_0}	1250 J/(kg K)	–
Reference thermal diffusivity	κ_0	0.7515×10^{-6} m^2/s	–
Activation energy	E	540 kJ/mol	–
Power-law exponent	n	3.5	–
Pre-exponential constant	A_η^*	28968.6 Pa s$^{1/n}$	–
Reference viscosity scale	η_0	10^{21} Pa s	–
Viscosity cap	η_{max}^*	10^{25} Pa s	–
Gas constant	R^*	8.3145 J/(mol K)	–
Derived velocity scale	v_0	23.716014 mm/yr	–
Convergence velocity	V_s^*	[§] mm/yr	V_s [§]

[c] Ocean–continent subduction only

[o] Ocean–ocean subduction only

[t] Time-dependent simulations only

[§] Varies between models

$$-\nabla\cdot\left(2\eta\frac{\nabla\vec{v}+\nabla\vec{v}^T}{2}\right)+\nabla P=0 \tag{100}$$

and either a time-dependent equation for temperature

$$\rho\left(\frac{\partial T}{\partial t}+\vec{v}\cdot\nabla T\right)=\nabla\cdot(k\nabla T)+H \tag{101}$$

or its equivalent when a steady-state solution is assumed

$$\rho\vec{v}\cdot\nabla T=\nabla\cdot(k\nabla T)+H \tag{102}$$

The viscosity η is either constant 1 or follows from the dislocation creep formulation (96) with cap (98) as

$$\eta=\frac{\eta^*}{\eta_0} \tag{103}$$

Note that for simplicity as well as clarity we form the viscosity function (98) in dimensional form and nondimensionalize the viscosity with the reference viscosity η_0.

2.3.3 Geometry, boundary conditions, and initial conditions

A simplified version of the typical geometry used in 2D subduction zone modeling with a kinematically prescribed slab is shown in Fig. 8a. The model is a 2D Cartesian box of width L and depth D. We picture a model with a straight slab surface here but it can also be constructed from a natural spline through a set of control points as in Syracuse et al. (2010) or connected linear segments with different angles with respect to the horizontal as in Wada and Wang (2009). In the models following the geometries of Syracuse et al. (2010) described in part III this simplified geometry is modified by including a curved slab and a coastline. At $x = 0$, the top of the model is at $(0, z_{\text{trench}})^T$, for a given depth of the trench, z_{trench}. Between $x = 0$ and $x = x_{\text{coast}}$, the presumed horizontal position of the coast, the top of the model shallows linearly to $(x_{\text{coast}}, 0)^T$. For $x > x_{\text{coast}}$, the top of the model is at $z = 0$. Actual choices for these parameters are provided in the zenodo archive linked to in the data availability statement. The kinematic slab approach requires at a minimum that the slab surface velocity with magnitude V_s is prescribed. The velocity in the slab, $\vec{v}_s$, can be determined from the solution of (99) and (100) in the slab (resulting in an extra Stokes equation owing to the discontinuity in velocity and pressure required across the slab above the coupling depth). Alternatively, the velocity in the slab can also be simply prescribed by defining the internal slab velocity to be parallel to and of same

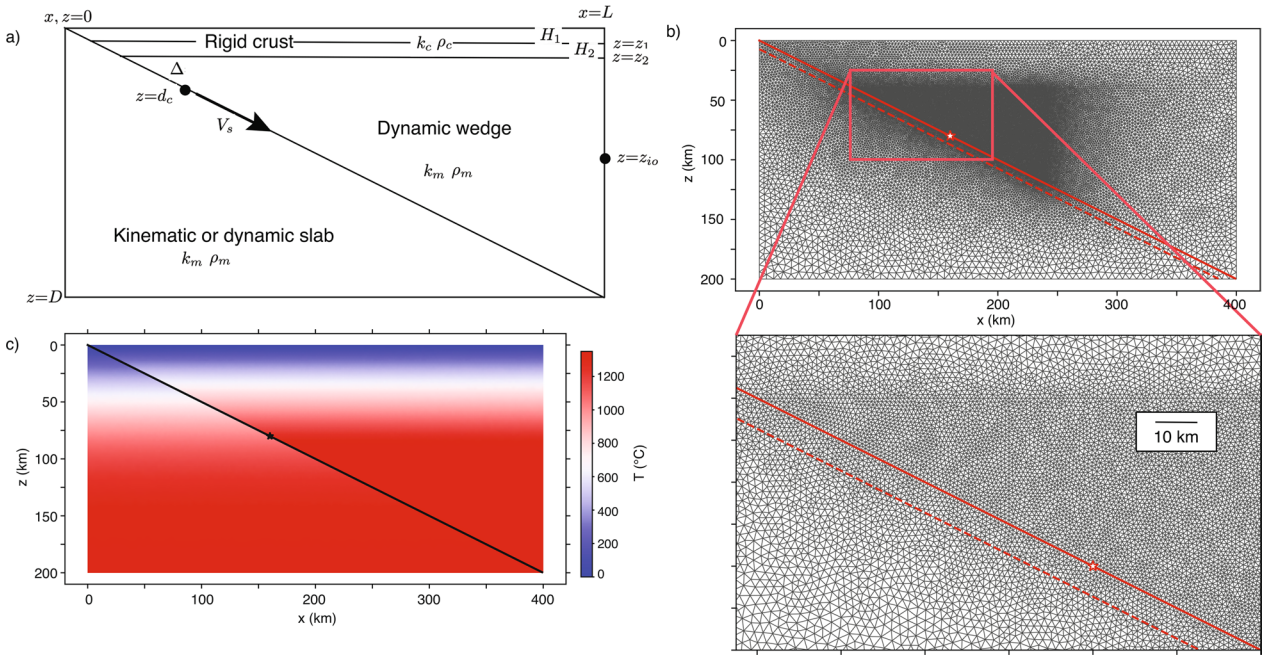

Fig. 8 **a** Geometry and coefficients for a simplified 2D subduction zone model specifically for the proposed new benchmark. All coefficients and parameters in the graph are nondimensional. The decoupling point is indicated by the star. **b** Example mesh (upper frame) constituted of triangles for the new benchmark geometry with zoom in (lower frame). This particular example is for TerraFERMA with 83,935 degrees of freedom in the heat equation. Size of the finite elements ranges from 1 km near the coupling point where the solution gradients are highest to up to 6 km away from thermal boundary layers. Red solid line is the top of the slab. Dashed red line is the slab Moho. c) Initial condition for the time-dependent benchmark problem

magnitude as that of the point on the slab surface closest to the point internal to the slab. For a straight-dipping slab, we have found that either approach leads to very similar temperature solutions; for a curved slab, the use of temperature-dependent viscosity also yields very similar temperature solution at the top of the slab for these two approaches. Here, we take the approach of solving for the velocity in the slab, solving (101) for temperature T in the whole domain and two Stokes equations (99) and (100), one in the wedge for $\vec{v}$ and P and one in the slab for $\vec{v} = \vec{v}_s$ and $P = P_s$. The velocity in the overriding plate, above the slab and down to $z = z_2$, is always prescribed as $\vec{v} = 0$ and the Stokes equation is not solved here.

We use an unstructured mesh of triangular elements to discretize the domain. A typical example, with 1 km element resolution in the region with the most activity is shown in Fig. 8b. On this mesh, we define approximate discrete solutions for velocity, pressure and temperature as

$$\vec{v} \approx \tilde{\vec{v}} = \sum_j \omega_j^k v_j^k \tag{104}$$

$$P \approx \tilde{P} = \sum_j \chi_j P_j \tag{105}$$

$$T \approx \tilde{T} = \sum_j \phi_j T_j \tag{106}$$

with similarly defined discrete test functions, $\tilde{\vec{v}}_t$, $\tilde{P}_t$ and $\tilde{T}_t$ using the same shape functions $\vec{\omega}_j = \omega_j^k$, χ_j and ϕ_j for velocity, pressure and temperature at each DOF j, respectively. In the results presented using TerraFERMA, we use a P2P1P2 discretization where $\vec{\omega}_j$ are piecewise-quadratic, χ_j are piecewise linear and ϕ_j are piecewise-quadratic continuous Lagrange functions. The results from Sepran use either the same P2P1P2 discretization (indicated by TH) or a penalty function method (indicated by PF) with quadratic P2 Crouzeix–Raviart (rather than Lagrange) shape functions for the velocity ($\vec{\omega}_j$). In this method, the dynamic pressure is eliminated from the Stokes equation (70) by a perturbation of the incompressibility constraint, that is, $\nabla \cdot \vec{v} = \epsilon_P P$ where ϵ_P is a small number. We use $\epsilon_P = 10^{-6}$ here; see Cuvelier et al. (1986) or King et al. (1990) for details on the elimination process. This method leads to a smaller stiffness matrix compared to that when using Taylor–Hood elements since the pressure unknowns are eliminated. It also results in a positive definite matrix for which more efficient direct solution methods exist. For the temperature shape functions (ϕ_j) Sepran also uses quadratic Lagrange

polynomials (resulting in a combined P2P2 discretization). In the penalty function approach, pressure is eliminated from the equations so χ_j are not used.

For the heat equation (101), we assume homogeneous natural (or Neumann) boundary conditions along the geometry where the velocity vector points out of the box (i.e., an outflow boundary). At the trench inflow boundary, we assume a half-space cooling model $T_{\text{trench}}(z)$ given by

$$\tilde{T}(x=0,z) = T_{\text{trench}}(z) = T_{\text{s}} + (T_{\text{m}} - T_{\text{s}})\text{erf}\left(\frac{z-z_{\text{trench}}}{z_d}\right) \tag{107}$$

where T_{s} is the nondimensional surface temperature, T_{m} is the nondimensional mantle temperature, z_{trench} is the nondimensional depth of the trench, and the nondimensional scale depth z_d is proportional to the dimensional age of the incoming lithosphere A^* via $z_d = 2\frac{\sqrt{\kappa_0 A^*}}{h_0}$.

Details of the backarc temperature depend on whether we are modeling ocean–continent or ocean–ocean subduction. In the ocean–continent case, we assume a constant surface heat flow q_s and radiogenic heat production H. We use a two-layer crustal model with density $\rho = \rho_c$, thermal conductivity $k = k_c$ and heat production $H = H_1$ from depth 0 to z_1 and heat production $H = H_2$ between depths z_1 and z_2, where z_1 and z_2 vary between subduction zones. The mantle portion of the model (in both the slab and the wedge) is assumed to have density $\rho = \rho_m$, conductivity $k = k_m$ and zero heat production $H = 0$. At the backarc, the wedge inflow boundary condition on temperature is chosen to be a geotherm $T_{\text{backarc,c}}(z)$ consistent with these parameters, that is,

$$\tilde{T}(x=L,z) = T_{\text{backarc,c}}(z) = \begin{cases} T_s - \frac{H_1 z^2}{2k_c} + \frac{q_s}{k_c} z & : 0 \le z \le z_1 \\ T_{\text{backarc,c}}(z=z_1) - \frac{H_2 (z-z_1)^2}{2k_c} + \frac{q_1}{k_c}(z-z_1) & : z_1 < z \le z_2 \\ \min(T_m, T_{\text{backarc,c}}(z=z_2) + \frac{q_2}{k_m}(z-z_2)) & : z_2 < z \le z_{io} \end{cases} \tag{108}$$

The discrete heat flow values q_i are the heat flow at the crustal boundaries at depth $z = z_i$ that can be found as $q_1 = q_s - H_1 z_1$ and $q_2 = q_1 - H_2(z_2 - z_1)$. In the ocean–ocean case we use a one-layer crustal model (z_1 is not defined), heat production is zero ($H = 0$) and the density and thermal conductivity are set to, respectively, $\rho = \rho_m$ and $k = k_m$ everywhere. The wedge inflow boundary condition on temperature down to z_{io} is then

$$\tilde{T}(x=L,z) = T_{\text{backarc,o}}(z) = T_{\text{s}} + (T_{\text{m}} - T_{\text{s}})\text{erf}\left(\frac{z}{z_c}\right) \tag{109}$$

where z_c is related to the dimensional age of the overriding plate A_c^* minus the age of subduction A_s^* via $z_c = 2\frac{\sqrt{\kappa_0 (A_c^* - A_s^*)}}{h_0}$. Below z_{io} we assume again a homogeneous Neumann boundary condition for temperature.

For the two Stokes equations, we assume homogeneous (zero stress) Neumann boundary condition on $\tilde{\vec{\nu}}$ and $\tilde{P}$ for the wedge in and outflow and on $\tilde{\vec{\nu}}_s$ and $\tilde{P}_s$ for the slab in and outflow. The top of the wedge at $z = z_2$ is a rigid boundary, $\tilde{\vec{\nu}} = 0$, consistent with the imposition of zero flow in the overriding plate. The wedge flow, $\tilde{\vec{\nu}}$, is driven by the coupling of the slab to the wedge below a coupling depth. This is implemented by a Dirichlet boundary condition along the slab surface. Above the coupling depth we impose zero velocity. Below the coupling depth the velocity is parallel to the slab and has magnitude V_s. It has been found that a smooth transition from zero to full speed over a short depth interval enhances the accuracy of the Stokes solution (see discussion in van Keken et al. (2002) and equations (13)–(15) in van Keken et al. (2008)) so here coupling begins at $z = d_c$ and ramps up linearly until full coupling is reached at $z = d_c$+2.5. For improved numerical accuracy, we specify nodal points at these depths in all models presented here and in part III. At the top of the wedge we imposed a rigid Dirichlet boundary condition at the base of the Moho on the wedge velocity, $\vec{\nu} = 0$. The slab flow, $\tilde{\vec{\nu}}_s$, is driven by the imposition of a Dirichlet boundary condition parallel to the slab with magnitude V_s along the entire length of the slab surface, resulting in a discontinuity between $\tilde{\vec{\nu}}$ and $\tilde{\vec{\nu}}_s$ above $z = d_c$+2.5.

In the case of time-dependent simulations, we require an initial condition T^0. We use an initial condition where the temperature on the slab side is given by T_{trench} (107). Above the slab we use $T_{\text{backarc,c}}$ (108) for ocean–continent subduction or $T_{\text{backarc,o}}$ (109) for ocean–ocean subduction. Figure 8c shows the initial condition used in the time-dependent benchmark comparison below.

2.3.4 Solution strategy

Sections 2.3.2 and 2.3.3 describe a set of nonlinear, potentially time-dependent equations and boundary conditions for the temperature, velocity and dynamic pressure in a subduction zone. To find their solution, we wish to find the root of the residual $\mathbf{r} = \mathbf{r}_{\vec{\nu}} + \mathbf{r}_P + \mathbf{r}_{\vec{\nu}_s} + \mathbf{r}_{P_s} + \mathbf{r}_T$, where

$$\mathbf{r}_{\vec{v}} = r_{\vec{v}_{i_1}} := \int_{\Omega_{\text{wedge}}} \left[\left(\frac{\nabla\vec{\omega}_{i_1} + \nabla\vec{\omega}_{i_1}^T}{2} \right) : 2\eta \left(\frac{\nabla\tilde{\vec{v}} + \nabla\tilde{\vec{v}}^T}{2} \right) - \nabla\cdot\vec{\omega}_{i_1}\tilde{P} \right] \mathrm{d}x = 0 \tag{110}$$

$$\mathbf{r}_P = r_{P_{i_2}} := -\int_{\Omega_{\text{wedge}}} \chi_{i_2} \nabla\cdot\tilde{\vec{v}} \mathrm{d}x = 0 \tag{111}$$

$$\mathbf{r}_{\vec{v}_s} = r_{\vec{v}_{si_3}} := \int_{\Omega_{\text{slab}}} \left[\left(\frac{\nabla\vec{\omega}_{i_3} + \nabla\vec{\omega}_{i_3}^T}{2} \right) : 2\eta \left(\frac{\nabla\tilde{\vec{v}}_s + \nabla\tilde{\vec{v}}_s^T}{2} \right) - \nabla\cdot\vec{\omega}_{i_3}\tilde{P}_s \right] \mathrm{d}x = 0 \tag{112}$$

$$\mathbf{r}_{P_s} = r_{P_{si_4}} := -\int_{\Omega_{\text{slab}}} \chi_{i_4} \nabla\cdot\tilde{\vec{v}}_s \mathrm{d}x = 0 \tag{113}$$

and, in the time-dependent case

$$\begin{aligned}\mathbf{r}_T = r_{T_{i_5}} := &\int_{\Omega_{\text{wedge}}} \left[\phi_{i_5}\rho\frac{\partial\tilde{T}}{\partial t} + \phi_{i_5}\tilde{\vec{v}}\cdot\nabla\tilde{T} + \nabla\phi_{i_5}\cdot k\nabla\tilde{T} \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{slab}}} \left[\phi_{i_5}\rho\frac{\partial\tilde{T}}{\partial t} + \phi_{i_5}\tilde{\vec{v}}_s\cdot\nabla\tilde{T} + \nabla\phi_{i_5}\cdot k\nabla\tilde{T} \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{crust}}} \left[\phi_{i_5}\rho\frac{\partial\tilde{T}}{\partial t} + \nabla\phi_{i_5}\cdot k\nabla\tilde{T} - \phi_{i_5}H \right] \mathrm{d}x = 0\end{aligned} \tag{114}$$

Here, Ω_{wedge}, Ω_{slab} and Ω_{crust} are subsets of the domain corresponding to the mantle wedge, slab and overriding crust, respectively. We have yet to discretize the time derivative $\frac{\partial\tilde{T}}{\partial t}$ in (114). Here, we choose to do this using finite differences, approximating the derivative by the difference between two discrete time levels

$$\frac{\partial\tilde{T}}{\partial t} \approx \frac{\tilde{T}^{n+1} - \tilde{T}^n}{\Delta t^n} \tag{115}$$

where $\Delta t^n = t^{n+1} - t^n$ is the time-step, the difference between the old and new times, and $\tilde{T}^{n+1}$ and $\tilde{T}^n$ represent the solution at these time levels. It then only remains to define at what time level the other coefficients in (114) are evaluated and we do this using a "theta" scheme such that

$$\begin{aligned}\mathbf{r}_T = r_{T_{i_5}} := &\int_{\Omega_{\text{wedge}}} \left[\phi_{i_5}\rho\left(\frac{\tilde{T}^{n+1} - \tilde{T}^n}{\Delta t^n} \right) + \phi_{i_5}\tilde{\vec{v}}^\theta\cdot\nabla\tilde{T}^\theta + \nabla\phi_{i_5}\cdot k\nabla\tilde{T}^\theta \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{slab}}} \left[\phi_{i_5}\rho\left(\frac{\tilde{T}^{n+1} - \tilde{T}^n}{\Delta t^n} \right) + \phi_{i_5}\tilde{\vec{v}}_s^\theta\cdot\nabla\tilde{T}^\theta + \nabla\phi_{i_5}\cdot k\nabla\tilde{T}^\theta \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{crust}}} \left[\phi_{i_5}\rho\left(\frac{\tilde{T}^{n+1} - \tilde{T}^n}{\Delta t^n} \right) + \nabla\phi_{i_5}\cdot k\nabla\tilde{T}^\theta - \phi_{i_5}H \right] \mathrm{d}x = 0\end{aligned} \tag{116}$$

where $\tilde{\vec{v}}^\theta = \theta_v\tilde{\vec{v}}^{n+1} + (1-\theta_v)\tilde{\vec{v}}^n$, $\tilde{\vec{v}}_s^\theta = \theta_v\tilde{\vec{v}}_s^{n+1} + (1-\theta_v)\tilde{\vec{v}}_s^n$ and $\tilde{T}^\theta = \theta\tilde{T}^{n+1} + (1-\theta_v)\tilde{T}^n$, and θ_v, $\theta \in [0,1]$ are parameters controlling what time level the coefficients are evaluated at. The parameter θ controls the stability and accuracy of the time integration scheme. Common choices are $\theta = 0$ (explicit Euler), $\theta = 1$ (implicit Euler) and $\theta = 0.5$ (Crank–Nicolson).

At each time level, (110)–(113) and (116) represent a nonlinear problem, which we solve using a Picard iteration, first solving (116), then solving (110)–(113) using the most up to date temperature, $\tilde{T}^{n+1}$, and repeating until the root of the residual, $\mathbf{r}$, is found to some tolerance. The time level and all solution variables are then updated and a new time level and new Picard iteration commenced. The time-step Δt^n is chosen such that the maximum Courant number, $c^n_{\max} = \max\left(\frac{\max\left(\tilde{\vec{v}}^n\right)\Delta t^n}{h_e}, \frac{\max\left(\tilde{\vec{v}}_s^n\right)\Delta t^n}{h_e} \right)$, where h_e is a measure of the local element size, does not exceed some critical value, $c^n_{\max} \le c_{\text{crit}}$. This procedure is repeated until the final time (the age of subduction, A_s^*) is reached.

If we are seeking the steady-state solution ($\frac{\partial T}{\partial t} = 0$), we solve (110)–(113) but (114) becomes

$$\begin{aligned}\mathbf{r}_T = r_{T_{i_5}} := &\int_{\Omega_{\text{wedge}}} \left[\phi_{i_5}\tilde{\vec{v}}\cdot\nabla\tilde{T} + \nabla\phi_{i_5}\cdot k\nabla\tilde{T} \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{slab}}} \left[\phi_{i_5}\tilde{\vec{v}}_s\cdot\nabla\tilde{T} + \nabla\phi_{i_5}\cdot k\nabla\tilde{T} \right] \mathrm{d}x \\ &+ \int_{\Omega_{\text{crust}}} \left[\nabla\phi_{i_5}\cdot k\nabla\tilde{T} - \phi_{i_5}H \right] \mathrm{d}x = 0\end{aligned} \tag{117}$$

where a theta-scheme approach is no longer required because no time levels exist. A Picard iteration is used to approximately find $\mathbf{r} = \mathbf{0}$, this time solving (110)–(113) first followed by (117). At the beginning of the simulation, we find an isoviscous ($\eta = 1$) solution to (110)–(113) to initialize the velocity and pressure.

2.3.5 An optimized subduction zone benchmark

The community subduction zone benchmark in van Keken et al. (2008) provides a set of simplified models well suited to test the accuracy of the solution of the governing equations that are relevant for subduction zones.

Unfortunately, the model geometry and assumptions that were chosen at the time are such that they introduce a few artifacts that do not occur, as best as we know, in any subduction zone on Earth. These artifacts include a slab that dips at a constant angle of 45° to 600 km depth, an overriding plate that excludes continental heat production, and the imposition of slab-wedge coupling at 50 km rather than at 75–80 km depth. The lack of crustal heating and the large width of the model, combined with the assumption of steady state, lead in the cases with temperature-dependent rheology to a very thick top boundary layer. This is caused by the cooling in the lithosphere, which results in a gradual thickening of the overriding lid in regions of the model that are far away from the arc-side boundary condition. While this is less of a problem in time-dependent problems (where time may not be sufficient for significant growth of the boundary layer), it shows up dramatically as a "viscous belly" in steady-state cases when the model domain is large (as it was in van Keken et al. 2008). In time-dependent models, it can show up if integration time is very long compared to the typical age of subduction zones (Hall 2012). The models in Syracuse et al. (2010) avoided this issue by using time integration to only $\sim$ 20-40 Myr. The models in Wada and Wang (2009) avoided it using steady-state models in a domain that is both narrower and shallower than that of the van Keken et al. (2008) benchmark.

To mitigate the artifacts of the previous benchmark, we propose a new benchmark model. Modifications include a more shallowly dipping slab that only extends to a depth of 200 km, the incorporation of radiogenic heating in the overriding crust and a deeper slab-wedge coupling point. We will also replace some of the requested model outputs from van Keken et al. (2008) with proper integrals. We will use the simplified geometry as in Fig. 8 with constant slab dip $\Delta = \tan^{-1}(1/2) = 26.56505°$ with respect to the horizontal. The maximum depth $D = 200$ defines $L = 400$. Crustal depths z_1 and z_2 are chosen as 15 and 40, respectively. $z_{\rm io}$ depends on wedge geometry and rheology and is therefore variable between models. To find this, we performed a simple iteration in the modeling by setting $z_{\rm io}$ first to a constant value, finding the solution to the nonlinear system, determining the actual value of $z_{\rm io}$ from the wedge flow and then imposing this value in a subsequent solution of the nonlinear system. While this approach guarantees appropriate implementation of the switch from Dirichlet to Neumann boundary condition for the heat equation as stated above, we have found that as long as $z_{\rm io}$ is larger than the depth where the actual switch between inflow and outflow occurs nearly identical solutions are obtained.

We will assume the reference values in Table 2 with case-specific parameters given in Table 3. The benchmark assumes ocean-continent subduction with heat production in a two-layer crust with crustal density and thermal conductivity (ρ_c and k_c, respectively) distinct from the mantle (ρ_m and k_m) and a backarc boundary condition on temperature given by $T_{\rm backarc,c}(z)$ (108). We will solve (110)–(113) either with constant viscosity ($\eta = 1$, case 1) or with temperature- and strain-rate-dependent viscosity following (103) (case 2). The heat equation will be solved under the assumption of steady state (117) for the benchmark, but we will also discuss some time-dependent results below. For the incoming lithosphere, we will assume $z_d = 97.397$ (corresponding to a dimensional age of the incoming lithosphere $A^* = 100$ Myr) and convergence speed $V_s = 4.2166$ (corresponding to a dimensional speed of 10 cm/yr).

2.3.6 Benchmark comparison TerraFERMA–Sepran

In the benchmark comparison, we focus on dimensional metrics representing the averaged thermal and velocity structures near the coupling point where gradients in velocity and temperature are high. The first metric is the slab temperature at 100 km depth, $T^*_{(200,-100)}$

$$T^*_{(200,-100)} \;=\; T_0\tilde{T}(x=200, y=-100) \tag{118}$$

The second metric is the average integrated temperature $\overline{T}^*_s$ along the slab surface between depths $z_{s,1} = 70$ and $z_{s,2} = 120$, that is,

$$\overline{T}^*_s \;=\; T_0\frac{\int_{s_1}^{s_2}\tilde{T}\,\mathrm{d}s}{\int_{s_1}^{s_2}\mathrm{d}s} \tag{119}$$

Table 3 Benchmark parameter values

Case	Type	η	q_s^* (W/m^2)	q_s	A^* (Myr)	z_2	$z_{\rm io}$	$z_{\rm trench}$	$x_{\rm coast}$	D	L	V_s
1	c	1	0.065	20.96774	100	40	139	0	0	200	400	4.2166
2	c	η^*/η_0	0.065	20.96774	100	40	154	0	0	200	400	4.2166

[c] Ocean–continent subduction

where s is distance along the slab top from the trench and $s_1 = \sqrt{5z_{s,1}^2} = 156.5248$ and $s_2 = \sqrt{5z_{s,2}^2} = 268.32816$. The third metric is the volume-averaged temperature $\overline{T}_w^*$ in the mantle wedge corner below the Moho, $z = z_2$ and above where the slab surface, $z = z_{\rm slab}(x)$, is between $z_{s,1}$ and $z_{s,2}$ as defined above

$$\overline{T}_w^* = T_0 \frac{\int_{x=140}^{x=240}\int_{z=z_2}^{z=z_{\rm slab}(x)} \tilde{T}\,{\rm d}z{\rm d}x}{\int_{x=140}^{x=240}\int_{z=z_2}^{z=z_{\rm slab}(x)} {\rm d}z{\rm d}x} \tag{120}$$

where $z_{\rm slab}(x) = x/2$. The final metric is the root-mean-squared averaged velocity $V_{{\rm rms},w}^*$ in the same volume as the third metric, that is,

$$V_{{\rm rms},w}^* = v_0 \sqrt{\frac{\int_{x=140}^{x=240}\int_{z=z_2}^{z=z_{\rm slab}(x)} \left(\tilde{\vec{v}}\cdot\tilde{\vec{v}}\right){\rm d}z{\rm d}x}{\int_{x=140}^{x=240}\int_{z=z_2}^{z=z_{\rm slab}(x)} {\rm d}z{\rm d}x}}. \tag{121}$$

Figure 9 shows the temperature fields obtained with TerraFERMA and temperature differences between the TerraFERMA and Sepran models. Convergence behavior on a series of finer meshes as a function of the number of degrees of freedom in the heat equation using metrics (118)–(121) is shown in Tables 4 and 5.

Note that even on the coarser grids the metrics are generally within less than 1% from those at the finest grids. The TerraFERMA and Sepran results tend to converge toward the same limit to reasonable precision for case 1. There seems to be a slight, but systematic difference particularly for $\overline{T}_w^*$ and $V_{{\rm rms},w}^*$ for case 2. Inspection of Fig. 9e shows the likely reason for the differences—a systematic bubble shows in ΔT right above the coupling point. We attribute this to how the two methods treat pressure and we will see more examples of this in part III.

2.3.7 Comparison of the time-dependent solution to that assuming steady state

Solving for the time-dependent solution given the same geometry, boundary conditions and parameters demonstrates how similar the steady-state and time-dependent solutions are after sufficient time in this optimized benchmark. The time-dependent slab top temperature evolution until $t^* = A_s^* = 25$ Myr is shown in Fig. 10a and that at the Moho is in Fig. 10b. In both cases, we plot the temperature to the depth that the subducting slab has reached after a given time interval. The temperature curves show a gradual

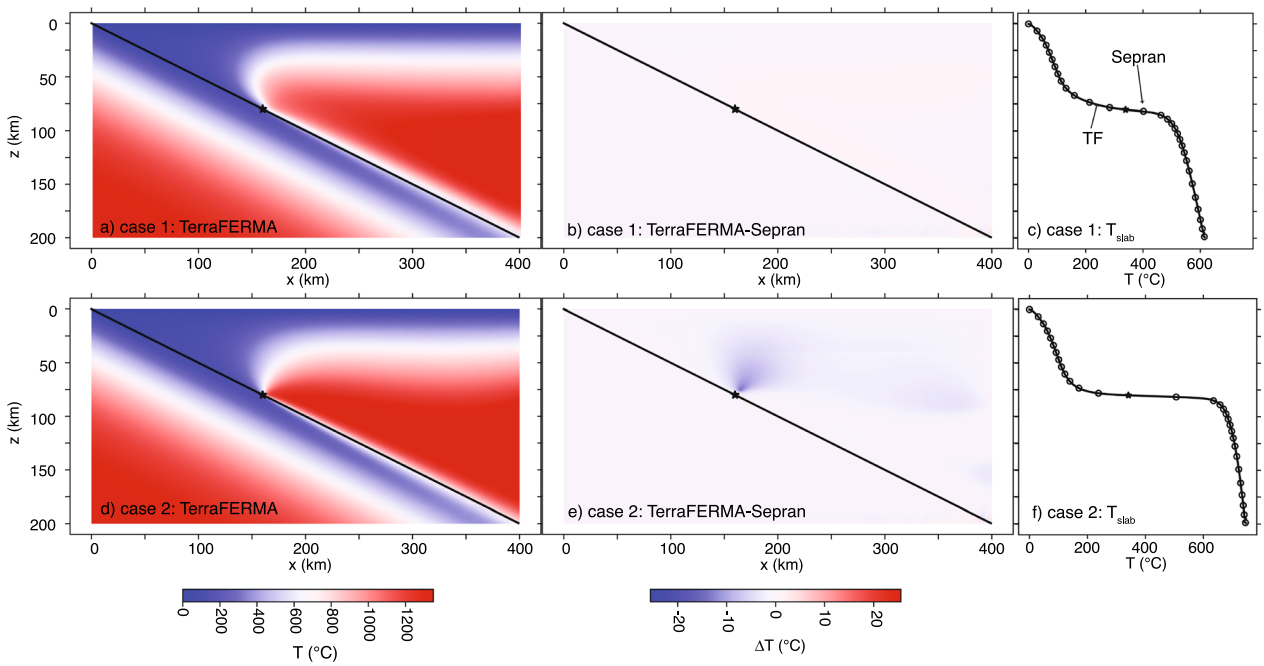

Fig. 9 Steady-state thermal structure for the updated subduction zone benchmark. **a** Temperature predicted by TF for case 1; **b** temperature difference between TF and Sepran using the penalty function (PF) method for case 1 at $f_m = 1$ where f_m represents the smallest element sizes in the finite element grids near the coupling point; **c** slab top temperature comparison for case 1; (**d**–**f**). As **a**–**c** but now for case 2. The star indicates the position or temperature conditions at the coupling point

Table 4 Convergence of various metrics describing the solution to the new subduction zone benchmark as a function of degrees of freedom in the heat equation T_{ndof}. The employed meshes have grid refinement in the wedge above and near the coupling point. The factor f_m is representative of the element size near the coupling point. *TH* Taylor–Hood, *PF* penalty function method. P2P1P2 indicates a discretization that has quadratic shape functions (P2) for velocity and temperature and linear shape functions for pressure (P1). P2P2 is for velocity and temperature only because pressure is eliminated from the Stokes equation in the penalty function method (Cuvelier et al. 1986). In this case, $z_{IO} = 139$

f_m	T_{ndof}	$T^*_{(200,-100)}$ (°C)	$\overline{T}^*_s$ (°C)	$\overline{T}^*_w$ (°C)	$V^*_{rms,w}$ (mm/yr)
TerraFERMA TH P2P1P2					
2.0	21403	517.17	451.83	926.62	34.64
1.0	83935	516.95	451.71	926.33	34.64
0.5	332307	516.86	451.63	926.15	34.64
Sepran TH P2P1P2					
2.0	17585	514.83	450.74	925.47	34.29
1.5	30851	515.37	451.07	925.71	34.36
1.0	68633	516.08	451.31	926.34	34.45
0.75	121366	516.24	451.31	926.30	34.50
0.5	270348	516.47	451.40	926.30	34.54
Sepran PF P2P2					
2.0	17585	515.07	450.92	926.03	34.28
1.5	30851	515.54	451.20	926.11	34.35
1.0	68633	516.17	451.37	926.56	34.45
0.75	121366	516.29	451.34	926.44	34.50
0.5	270348	516.48	451.40	926.37	34.54

Table 5 As Table 4 but now for case 2 with stress- and temperature-dependent viscosity. In this case, $z_{IO} = 154$

f_m	T_{ndof}	$T^*_{(200,-100)}$ (°C)	$\overline{T}^*_s$ (°C)	$\overline{T}^*_w$ (°C)	$V^*_{rms,w}$ (mm/yr)
TerraFERMA TH P2P1P2					
2.0	21403	683.05	571.58	936.65	40.89
1.0	83935	682.87	572.23	936.11	40.78
0.5	332307	682.80	572.05	937.37	40.77
Sepran TH P2P1P2					
2.0	17581	681.28	570.26	935.47	41.05
1.5	30947	682.48	570.73	937.11	40.91
1.0	68713	683.07	571.23	940.47	40.92
0.75	121574	682.97	571.62	941.23	41.00
0.5	270668	682.92	572.04	941.28	41.06
Sepran PF P2P2					
2.0	17585	682.38	567.96	936.52	40.67
1.5	30851	683.67	569.60	942.73	40.63
1.0	68633	683.61	571.86	941.18	40.87
0.75	121366	683.03	571.77	940.32	40.98
0.5	270348	682.38	571.44	939.73	41.06

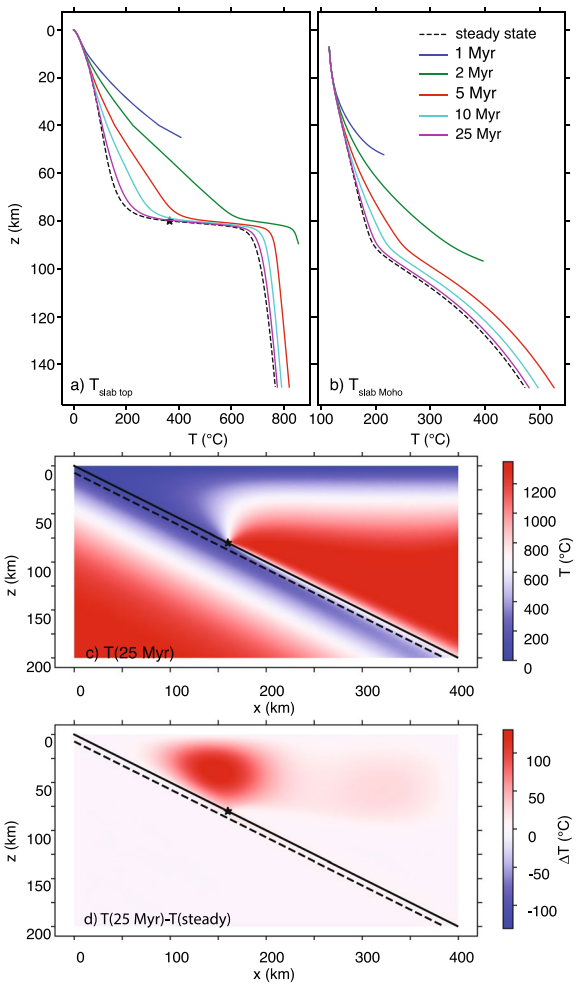

Fig. 10 Time-dependent example based on the new subduction zone benchmark. **a** Evolution of slab top temperature as a function of time—curves are plotted only to the depth that the slab tip has reached at each time; **b** As **a** but now for the slab Moho; **c** temperature at 25 Myr (compare to steady-state thermal structure in Fig. 9d); **d** difference between temperature of the time-dependent solution after 25 Myr and the steady-state solution—while the slab thermal structure is nearly identical, the cold corner is still evolving at 25 Myr toward the steady-state structure. Star indicates the location of the coupling point or its steady-state temperature

convergence to the steady-state solution (the dashed line). The temperature at 25 Myr is given in Fig. 10c (compare with Fig. 9d) and the temperature difference between that at 25 Myr and the steady-state case is shown in Fig. 10d—clearly the forearc thermal structure is the slowest part of the model to adjust to steady state.

The benchmark has been designed to give a near-steady-state solution close to the time-dependent solution after 25 Myr. However, this similarity is not generally the case in other geometries so time-dependent solutions remain necessary when considering a larger suite

of models and therefore form the bulk of the results presented in part III. Due to the slow evolution of the subduction system, we found in the time-dependent version of the benchmark that fully converging the residual, **r**, was not necessary for an accurate solution, making extremely minor differences after 25 Myr of evolution. Linearizing the problem and only taking a single Picard iteration at each time level represent a considerable computational cost saving so we adopt that approach in part III. TerraFERMA results are presented using $\theta_v = \theta = 0.5$. Sepran uses $\theta_v = \theta = 1$ and both use $c_{\text{crit}} = 1$ in all time-dependent results shown there.

3 Conclusions

By constructing a series of demonstration problems, we have shown how finite element models can be constructed, tested, and validated. Once validated, these simpler systems of equations can be used as building blocks to develop a kinematic–dynamic model of subduction zone thermal structure. We propose a new benchmark problem for subduction zones that incorporates more of the physical complexity associated with their thermal structure while avoiding some of the pitfalls associated with nonphysical geometries and assumptions of the original van Keken et al. (2008) benchmark. This has been demonstrated with two independent finite element approaches (TerraFERMA and Sepran) that also use different discretization strategies. In part III, we will use these models and apply the discretization and solution strategies described here to a global suite of subduction zones. We will discuss where they agree and disagree, both with each other and with published observations of subduction zone thermal structure.

Abbreviations

DOFs	Degrees of freedom
FEM	Finite element method
SZ	Subduction zone
TF	TerraFERMA
UFL	Unified Form Language

Acknowledgements

We thank editor Magali Billen and two anonymous reviewers for many helpful and constructive comments and questions that allowed us to improve the manuscript. We thank Scott King for comments on an earlier version of the manuscript.

Author contributions

Both authors conceived of the approach to the review paper. CW provided the main modeling using TF. PvK provided the Sepran-based models. Both authors contributed to writing this paper.

Funding

This work was supported in part by the National Science Foundation grants 1850634 and 2021027.

Availability of data and materials

The modeling data shown in the figures are provided in the zenodo repository available at doi.org/10.5281/zenodo.7843967. The TF modeling data can be independently reproduced using the input files provided in https://github.com/cianwilson/vankeken_wilson_peps_2023, which can be run using the docker images contained in https://github.com/users/cianwilson/packages/container/package/vankeken_wilson_peps_2023. The zenodo repository contains all files and information referenced in the data availability statement.

Declarations

Competing interests

The authors declare that they have no competing interest.

Received: 24 April 2023 Accepted: 5 September 2023
Published online: 30 November 2023

References

Alnæs MS, Logg A, Ölgaard KB, Rognes ME, Wells GN (2014) Unified form language: a domain-specific language for weak formulations of partial differential equations. ACM Trans Math Softw 40:1–37. https://doi.org/10.1145/2566630

Alnæs MS, Blechta J, Hake J, Johansson A, Kehlet B, Logg A, Richardson C, Ring J, Rognes ME, Wells GN (2015) The FEniCS project version 1.5. Arch Num Softw 3:9–23. https://doi.org/10.11588/ans.2015.100.20553

Auricchio F, Beirão da Veiga L, Brezzi F, Lovadina C (2017) Mixed finite element methods. In: Stein E, Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics, 2nd edn. John Wiley & Sons Ltd, Chichester, pp 1–53. https://doi.org/10.1002/9781119176817.ecm2004

Balay S, Abhyankar S, Adams MF, Benson S, Brown J, Brune P, Buschelman K, Constantinescu EM, Dalcin L, Dener A, Eijkhout V, Faibussowitsch J, Gropp WD, Hapla V, Isaac T, Jolivet P, Karpeev D, Kaushik D, Knepley MG, Kong F, Kruger S, May DA, McInnes LC, Mills RT, Mitchell L, Munson T, Roman JE, Rupp K, Sanan P, Sarich J, Smith BF, Zampini S, Zhang H, Zhang H, Zhang J (2023) PETSc Web page. https://petsc.org/

Batchelor GK (1967) An introduction to fluid dynamics. Cambridge University Press, Cambridge

Blankenbach B, Busse F, Christensen U, Cserepes L, Gunkel D, Hansen U, Harder H, Jarvis G, Koch M, Marquart G, Moore D, Olson P, Schmeling H, Schnaubelt T (1989) A benchmark for mantle convection codes. Geophys J Int 98:23–38. https://doi.org/10.1111/j.1365-246X.1989.tb05511.x

Burstedde C, Stadler G, Alisic L, Wilcox LC, Tan E, Gurnis M, Ghattas O (2013) Large-scale adaptive mantle convection simulation. Geophys J Int 192:889–906. https://doi.org/10.1093/gji/ggs070

Cuvelier C, Segal A, van Steenhoven AA (1986) Finite element models and the Navier-Stokes equations. Reidel, Dordrecht

Dabrowski M, Krotkiewski M, Schmid DW (2008) MILAMIN: MATLAB-based finite element method solver for large problems. Geochem Geophys Geosyst 9:Q04030. https://doi.org/10.1029/2007GC001719

Davies DR, Wilson CR, Kramer SC (2011) Fluidity: a fully unstructured anistropic adaptive mesh computational modeling framework for geodynamics. Geochem Geophys Geosyst 12:Q06001. https://doi.org/10.1029/2011GC003551

Davies RD, Kramer SC, Ghelichkhan S, Gibson A (2022) Towards automatic finite-element methods for geodynamics via Firedrake. Geosci Model Dev 15:5127–5166. https://doi.org/10.5194/gmd-15-5127-2022

Euen GT, Liu S, Gassmöller R, Heister T, King SD (2022) A comparison of 3-D spherical shell thermal convection results at low to moderate Rayleigh number using ASPECT (version 2.2.0) and CitComS (version 3.3.1). Geosci Model Dev Discuss. https://doi.org/10.5194/gmd-2022-252

Fullsack P (1995) An arbitrary Lagrangian-Eulerian formulation for creeping flows and its application in tectonic models. Geophys J Int 120:1–23. https://doi.org/10.1111/j.1365-246X.1995.tb05908.x

Gerya T (2019) Introduction to numerical geodynamical modelling, 2nd edn. Cambridge University Press, Cambridge

Golub GH, Van Loan CF (1989) Matrix computations, 2nd edn. Johns Hopkins University Press, Baltimore

Hall PS (2012) On the thermal evolution of the mantle wedge at subduction zones. Phys Earth Planet Int 198–199:9–27. https://doi.org/10.0116/j.pepi.2012.03.004

Ham DA, Farrell PE, Gorman GJ, Maddison JR, Wilson CR, Kramer SC, Shipton J, Collins GS, Cotter CJ, Piggott MD (2009) Spud 1.0: generalizing and automating the user interfaces of scientific computer models. Geosci Model Dev 2:33–42. https://doi.org/10.5194/gmd-2-33-2009
Ho-Liu P, Hager BH, Raefsky A (1987) An improved method of Nusselt number calculation. Geophys J Int 88:205–215. https://doi.org/10.1111/j.1365-246X.1987.tb01375.x
Hughes TJR (1987) The finite element method. Prentice-Hall Inc, Englewood Cliffs
Ismail-Zadeh A, Tackley P (2010) Computational methods for geodynamics. Cambridge University Press, Cambridge
Johnson C (1987) Numerical solution of partial differential equations by the finite element method. Cambridge University Press, Cambridge
Karato S-I, Wu P (1993) Rheology of the upper mantle: a synthesis. Science 260:771–778. https://doi.org/10.1126/science.260.5109.771
King SD, Raefsky A, Hager BH (1990) ConMan: vectorizing a finite element code for incompressible two-dimensional convection in the Earth's mantle. Phys Earth Planet Inter 59:195–207. https://doi.org/10.1016/0031-9201(90)90225-M
King SD, Lee C, van Keken PE, Leng W, Zhong S, Tan E, Tosi N, Kameyama MC (2010) A community benchmark for 2-D Cartesian compressible convection in the Earth's mantle. Geophys J Int 180:73–87. https://doi.org/10.1111/j.1365-246X.2009.04413.x
Kirby RC, Logg A (2006) A compiler for variational forms. ACM Trans Math Softw 32:417–444. https://doi.org/10.1145/1163641.1163644
Kronbichler M, Heister T, Bangerth W (2013) High accuracy mantle convection simulation through modern numerical methods. Geophys J Int 191:12–29. https://doi.org/10.1111/j.1365-246X.2012.05609.x
Logan DL (2017) A first course in the finite element method, 6th edn. Cengage, Boston
Logg A, Mardal K-A, Wells G (2012) Automated solution of differential equations by the finite element method. Springer, Berlin
Moresi L, Quenette S, Lemiale V, Mériaux C, Appelbe B, Mühlhaus H-B (2007) Computational approaches to studying non-linear dynamics of the crust and mantle. Phys Earth Planet Inter 163:69–82. https://doi.org/10.1016/j.pepi.2007.06.009
Oden JT, Reddy JN (1976) An introduction to the mathematical theory of finite elements. John Wiley and Sons, New York
Strang G, Fix G (2008) An analysis of the finite element models, 2nd edn. Wellesley Cambridge Press, Wellesley
Syracuse EM, van Keken PE, Abers GA (2010) The global range of subduction zone thermal models. Phys Earth Planet Int 183:73–90. https://doi.org/10.1016/j.pepi.2010.02.004
Turcotte D, Schubert G (2002) Geodynamics, 2nd edn. Cambridge University Press, Cambridge
van den Berg AP, Segal G, Yuen DA (2015) SEPRAN: a versatile finite-element package for a wide variety of problems in geosciences. J Earth Sci 26:89–95. https://doi.org/10.1007/s12583-015-0508-0
van Keken PE, Wilson CR (2023a) An introductory review of the thermal structure of subduction zones: I—motivation and selected examples. Prog Earth Planet Sci 10:42. https://doi.org/10.1186/s40645-023-00573-z
van Keken PE, Wilson CR (2023b) An introductory review of the thermal structure of subduction zones: III—comparison between models and observations. Progr Earth Planet Sci 10:57. https://doi.org/10.1186/s40645-023-00589-5
van Keken PE, Kiefer B, Peacock SM (2002) High-resolution models of subduction zones: implications for mineral dehydration reactions and the transport of water to the deep mantle. Geochem Geophys Geosyst 3:1056. https://doi.org/10.1029/2001GC000256
van Keken PE, Currie C, King SD, Behn MD, Cagnioncle A, He J, Katz RF, Lin S-C, Parmentier EM, Spiegelman M, Wang K (2008) A community benchmark for subduction zone modeling. Phys Earth Planet Int 171:187–197. https://doi.org/10.1016/j.pepi.2008.04.015
van Keken PE, Hacker BR, Syracuse EM, Abers GA (2011) Subduction factory: 4. Depth-dependent flux of H_2O from subducting slabs worldwide. J Geophys Res Solid Earth 116:B01401. https://doi.org/10.1029/2010JB007922
Vynnytska L, Rognes ME, Clark SR (2013) Benchmarking FEniCS for mantle convection simulations. Comput Geosci 50:95–105. https://doi.org/10.1016/j.cageo.2012.05.012
Wada I, Wang K (2009) Common depth of slab-mantle decoupling: reconciling diversity and uniformity of subduction zones. Geochem Geophys Geosyst 10:Q10009. https://doi.org/10.1029/2009GC002570
Wilson CR, Spiegelman MS, van Keken PE (2017) TerraFERMA: The Transparent Finite Element Rapid Model Assembler for multiphysics problems in Earth sciences. Geochem Geophys Geosyst 18:769–810. https://doi.org/10.1002/2016GC006702
Zhong SJ, Yuen DA, Moresi LN, Knepley MG (2015) Numerical methods for mantle convection. In: Schubert G (ed.) Treatise on geophysics (2nd ed), Volume 7 "Mantle Dynamics" (Bercovici, D (ed.)), pp 197–222. Elsevier, Amsterdam. https://doi.org/10.1016/B978-0-444-53802-4.00130-5
Zhong S, McNamara A, Tan E, Moresi L, Gurnis M (2008) A benchmark study on mantle convection in a 3-D spherical shell using CitcomS. Geochem Geophys Geosyst 9:Q10017. https://doi.org/10.1029/2008GC002048

Publisher's Note