Modeling Mantle Flow and Melting Processes at Mid-Ocean Ridges and Subduction Zones

—

Development and Application of Numerical Models

Dissertation

Zur Erlangung des Doktorgrades der Naturwissenschaften im
Department Geowissenschaften der Universität Hamburg

vorgelegt von

Jörg Hasenclever

aus Altenkirchen

Hamburg

2010

Als Dissertation angenommen vom Fachbereich Geowissenschaften
der Universität Hamburg

Auf Grund der Gutachten von Prof. Dr. Matthias Hort
und Prof. Dr. Jason Phipps Morgan

Hamburg, den ....

Prof. Dr. Jürgen Oßenbrügge
Leiter des Department Geowissenschaften

Zusammenfassung

Schmelzprozesse und viskoses Fließen im oberen Erdmantel sind die Ursache für viele geologische Prozesse, die die Erdoberfläche prägen. Die Untersuchung der Vorgänge im Erdinneren wird dadurch erschwert, dass es unmöglich ist zu den Orten vorzudringen, an denen sie stattfinden. Um die geodynamischen Zusammenhänge dennoch zu untersuchen haben sich numerische Modelle als hilfreiche Werkzeuge erwiesen. Numerische Modelle kombinieren die Beobachtungen und Ergebnisse anderer geowissenschaftlicher Disziplinen und führen sie in einem physikalisch konsistenten Gerüst zusammen.

In dieser Arbeit habe ich numerische Modelle entwickelt, um das viskose Kriechen des Erdmantels sowie seine Temperaturentwicklung und partielle Aufschmelzung zu untersuchen. Die Modelle $\mathbf{M3_{tri}}$ und $\mathbf{M3_{tet}}$ beschreiben diese Prozesse im zwei- bzw. dreidimensionalem Raum. Das viskose Flussfeld, beschrieben durch die Stokes Gleichungen, und die Energiegleichung werden mittels der Finite Elemente Methode (FEM) approximiert. Diese Methode erlaubt die Verwendung unstrukturierter Gitter, mithilfe derer lokal eine hohe räumliche Auflösung erzielt werden kann.

Bei großskaligen 3-D Problemstellungen können keine direkten Lösungsalgorithmen zur Lösung der Matrixgleichungen verwendet werden, die aus der FEM hervorgehen. Stattdessen werden iterative „Solver“ benötigt, deren Performance möglichst unabhängig von den Viskositätskontrasten sein sollte — letztere werden durch Veränderungen in Temperatur, Druck sowie Zusammensetzung des Erdmantels hervorgerufen. In dieser Arbeit habe ich verschiedene eigenständige Lösungsalgorithmen hinsichtlich ihrer Performance verglichen und schließlich kombiniert, um von ihren individuellen Stärken zu profitieren. Ich habe einen Lösungsalgorithmus entwickelt, der aus einem Konjugierte Gradienten Verfahren besteht, welches mittels eines Multigrid-Algorithmus (einzelner V-Zyklus) präkonditioniert wird. Auf dem gröbsten Gitter wird ein direkter Lösungsalgorithmus (Cholesky Faktorisierung) angewendet. Zur Lösung des gekoppelten Geschwindigkeit-Druck-Problems zeige ich unterschiedliche Formulierungen und vergleiche sie hinsichtlich ihrer Performance in Kombination mit dem oben genannten Krylov-Unterraum Algorithmus.

Weiterhin habe ich in dieser Arbeit eine neue Formulierung für Schmelzprozesse entwickelt, welche ein aus mehreren Lithologien bestehendes Mantelgestein sowie den Wassergehalt der Gesteine berücksichtigt. Diese Methode wird mit Hilfe eines 1-D Modells erläutert, welches das Schmelzen eines heterogenen Erdmantels unter Druckentlastung beschreibt. Chemische und rheologische Konsequenzen dieser Schmelzprozesse an Rückenachsen werden untersucht. Es stellt sich heraus, dass ein höherer Wassergehalt im Mantelgestein nur eine sehr geringe Auswirkung auf die produzierte Gesamtmenge an Schmelzen hat. Die starke Fraktionierung von Wasser in die Gesteinsschmelze führt zu sehr

geringen Aufschmelzraten im „nassen“ Umfeld, und hohe Schmelzraten werden erst nach Überschreiten des „trockenen“ Solidus erreicht.

Anschließend werden die 1-D Resultate mit 2-D und 3-D numerischen Modellierungen verglichen, in denen Mantelströmung und Schmelzprozesse gemeinsam betrachtet werden. Hier stelle ich fest, dass der mit der Dehydrierung einhergehende Anstieg der Gesteinsviskosität nicht notwendigerweise die effektive Viskosität des Gesamtgesteins erhöht. Erst wenn die letzte lithologische Einheit zu schmelzen beginnt und ihren Wassergehalt reduziert, steigt die Viskosität des Gesamtgesteins an und bildet eine kompositionelle Lithosphäre. Unterhalb dieser Lithosphäre kann eine ca. 30–50 km mächtige Zone erniedrigter Viskosität entstehen, falls die Gesteine durch Schmelzeinschlüsse geschwächt werden. Konvektive Instabilitäten, die in dieser Zone entstehen könnten, wären eine Erklärung für den auffällig regulären Abstand vulkanischer Zentren an langsam spreizenden Rücken.

In einer Fallstudie wird eine Schmelzanomalie am Mittel-Atlantischen Rücken nahe der Ozeaninsel Ascension untersucht. Zwei mögliche Erklärungen werden gegenübergestellt: ein schwacher Mantelplume einerseits und eine chemisch weniger verarmte Heterogenität im Erdmantel andererseits. Um die beobachteten Krustendicken mit dem numerischen Modell beschreiben zu können, darf der Plume nicht mehr als 60°C heißer sein als das umgebende Gestein und nicht weniger als 100 km östlich der Rückenachse aufsteigen. Eine Mantelheterogenität aus weniger verarmtem Peridotit könnte sehr ähnliche Krustenanomalien produzieren und scheint eine realistischere Erklärung zu sein, da ein schwacher Mantelplume nur sehr langsam aufsteigt und währenddessen einen Großteil seiner thermischen Energie an das umgebende Mantelgestein abgeben würde.

Eine weitere Anwendung der 2-D und 3-D Modelle beschäftigt sich mit Mantelströmungen an Subduktionszonen. Freigesetzte wasserhaltiger Fluide aus der abtauchenden ozeanischen Platte können die Zusammensetzung und Dichte der Mantelgesteine verändern, durch die sie migrieren. Eine wasserhaltige Grenzschicht könnte sich oberhalb der abtauchenden Lithosphärenplatte („Slab“) bilden, aus der sich Rayleigh-Taylor-Instabilitäten entwickeln. Die Studie zeigt, dass sich verschiedene Diapirtypen entwickeln können, die eine Alternative zu dem „Hot Fingers“-Model darstellen, welches die Gruppierung vulkanischer Zentren in manchen Vulkanbogen zu erklären sucht. Startzeitpunkt und Position der ersten Instabilität wird von 2-D und 3-D Modellen ähnlich beschrieben. Nachfolgende Instabilitäten sowie der Aufstieg der Diapire selbst verlangen jedoch nach einem 3-D numerischen Modell. Eine hohe numerische Auflösung im Bereich der Grenzschicht ist von größter Bedeutung, da ein unzureichend auflösendes numerisches Model die Diapirbildung sowohl zeitlich als auch räumlich inkorrekt beschreibt.

Abstract

Melting processes and solid-state flow of the rocks in the Earth's uppermost mantle are responsible for many geological processes that shape the Earth's surface. Studying these processes is complicated by our lack of direct access to the regions where they take place. Numerical models have become a very helpful tool to study the interactions of these different processes — they also allow researchers to better synthesize and understand observations and interpretations obtained by other geophysical, geological and geochemical disciplines.

In this thesis, I have developed numerical models (named $\mathbf{M3_{tri}}$ and $\mathbf{M3_{tet}}$) that allow studying the thermal evolution of the mantle as well as its viscous creep in two and three dimensions, respectively. Mantle flow (described by the Stokes equations) and the energy equation are approximated using the finite element method (FEM). This approach was chosen because it allows to use unstructured meshes that are locally refined in critical regions of interest.

Direct solvers cannot be used to solve the matrix equations that arise from the FEM applied to large 3-D problems. Instead an iterative solver is required, whose performance is desired to be largely independent of the viscosity contrasts that arise from temperature, pressure, and compositional variations in the Earth's mantle. Different stand-alone solution algorithms such as multigrid and the conjugate gradient algorithm were tested and combined to best take advantage of their distinct strengths. The solution algorithm developed in this thesis uses a conjugate gradient algorithm that is preconditioned by a multigrid V-cycle with a direct solver (Cholesky factorization) on its coarsest level. Different formulations to address the coupled velocity-pressure problem are presented and compared to each other with emphasis being on performance in combination with the above mentioned iterative Krylov-subspace solver.

A new formulation for melting of a heterogeneous multi-component mantle is introduced using a simple 1-D model that has been developed in this thesis. It is used to study the 1-D decompression melting of a heterogeneous mantle and to so explore detailed chemical and rheological consequences of melting underneath a ridge axis. I find that an initial water content in the mantle rocks has a very small impact on the total melt production. While the onset of melting of a wet lithology is shifted to greater depths, the melting rates in this wet melting regime remain low because water efficiently partitions into the melt. Only when the dry solidus is crossed high melt productivities are observed.

The 1-D results are compared to 2-D and 3-D calculations of mid-ocean ridge mantle flow and melting. Here I find that the dehydration-related increase in viscosity of each lithology does not increase the effective (aggregate) mantle viscosity until the last (most depleted) lithology starts to melt and to dehydrate. Instead a low viscosity region forms if melts are assumed to weaken the mantle rocks. This low viscosity region is located underneath the base of the compositional lithosphere and has a vertical extension of about 30–50 km. Convective instabilities may arise in this region and could explain the observed regular spacing between volcanic centers at several slow-spreading ridges.

A case study focussing on a particular melting anomaly at the Mid-Atlantic ridge near Ascension Island further illustrates the applicability of the numerical models developed in this thesis. Here two potential scenarios are compared: a weak mantle plume and a fertile mantle heterogeneity. I find that in order to achieve similar crustal thicknesses in the numerical calculations as observed, a potential mantle plume has to have an excess temperature of less than 60°C and has to be located not more than 100 km East of the ridge axis. A mantle heterogeneity composed of more fertile mantle peridotite can produce a very similar crustal thickness anomaly. This seems to be a more realistic explanation, because a plume with only 50–60°C excess temperature ascends very slowly and can easily lose most of its thermal energy to the ambient mantle.

Another application of the 2-D and 3-D numerical models focusses on mantle flow at subduction zones. Aqueous fluids, released by the descending and dehydrating slab, have to migrate through the mantle wedge and are likely to change the composition and density of the mantle rocks that they pass into. These density reductions can give rise to Rayleigh-Taylor-like instabilities emerging from a wet boundary layer on top of the slab. The study shows that different types of diapirism can evolve that could be an alternative explanation to the "hot finger" model in order to explain the clustering of volcanic centers in some arcs. I find that the onset time and position of the diapirism is often very similar for 2-D and 3-D calculations when using the same set of parameters. However, modeling the ascent time and formation of secondary instabilities (triggered by the first diapirs) require a 3-D code. Of greatest importance is the numerical resolution in the region where the boundary layer forms, because a too low resolution leads to misleading onset times and locations of diapirism.

Contents

Table 1: *List of abbreviations.*

abbreviation	meaning
PDE	Partial Differential Equation
BT	back-tracking points
FEM	Finite Element Method
FDM	Finite Difference Method
SD	Steepest Descent
CG	Conjugate Gradients
PCG	Preconditioned Conjugate Gradients
SOR	Successive Overrelaxation
MG	(Geometrical) Multigrid
AMG	Algebraic Multigrid
CR	Crouzeix-Raviart
TH	Taylor-Hood
dof	degree of freedom
RHS	Right-Hand Side
SMP	Symmetric Multi-Processing
MOR	Mid-Ocean Ridge
MORB	Mid-Ocean Ridge Basalt
OIB	Ocean Island Basalt
TF	Transform Fault
MAR	Mid-Atlantic Ridge
AFZ	Ascension Fracture Zone
BVFZ	Bode Verde Fracture Zone
EPR	East Pacific Rise
DP	Depleted Peridotite
FP	Fertile Peridotite, also Pyrolite
PYX	Pyroxenite
ol	Olivine
opx	Orthopyroxene
cpx	Clinopyroxene
gt	Garnet
sp	Spinel
plg	Plagioclase
Fe	Iron
Mg	Magnesium

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Our modern view of the Earth

The largest geological features that dominate the face of our planet are lithospheric plates. They are in a permanent though slow motion so that their ”lively” character was recognized comparably late in the history of science. Pioneering studies by Alfred Wegener, Arthur Holmes, Chaim Leib Pekeris, and Harry Hammond Hess (and many others) have been major steps towards our current knowledge of Earth's internal structure and functioning (see Bercovici (2007) for a comprehensive historical review). Morphologic changes of the Earth's crust over a human lifetime are only observable in few places, for instance at active tectonic margins that extend to the surface such as the San Andreas fault in California, or at places of ongoing volcanic activity (e.g. Hawaii or Iceland). Occasionally large natural catastrophes such as the earthquake at Sumatra's subduction zone that led to the devastating tsunami in 2004 or the Haiti earthquake in 2010 remind mankind of Earth's dynamic nature.

Our modern picture of the dynamic state of the Earth is shaped through various modern technologies, all of which contribute to verification and quantification of the very slow changes inside the Earth, especially in its outer shell. These technologies span from detailed interpretations of geologic structures and the variability of geochemical data all the way to observations of seismic activity and ground deformations as well as the direct measurement of plate tectonics by employing very precise geodetic measurements (GPS or VLBI[1]).

Today we distinguish two types of lithospheric plates: continental and oceanic. The largest fraction of the continental plates, on the one hand, formed early in Earth's history, supposedly by large amounts melt of extraction from the initially much hotter planet. Accretion of new material to the continents, however, occurs continuously through volcanism,

[1] Global Positioning System and Very Long Baseline Interferometry, resp.

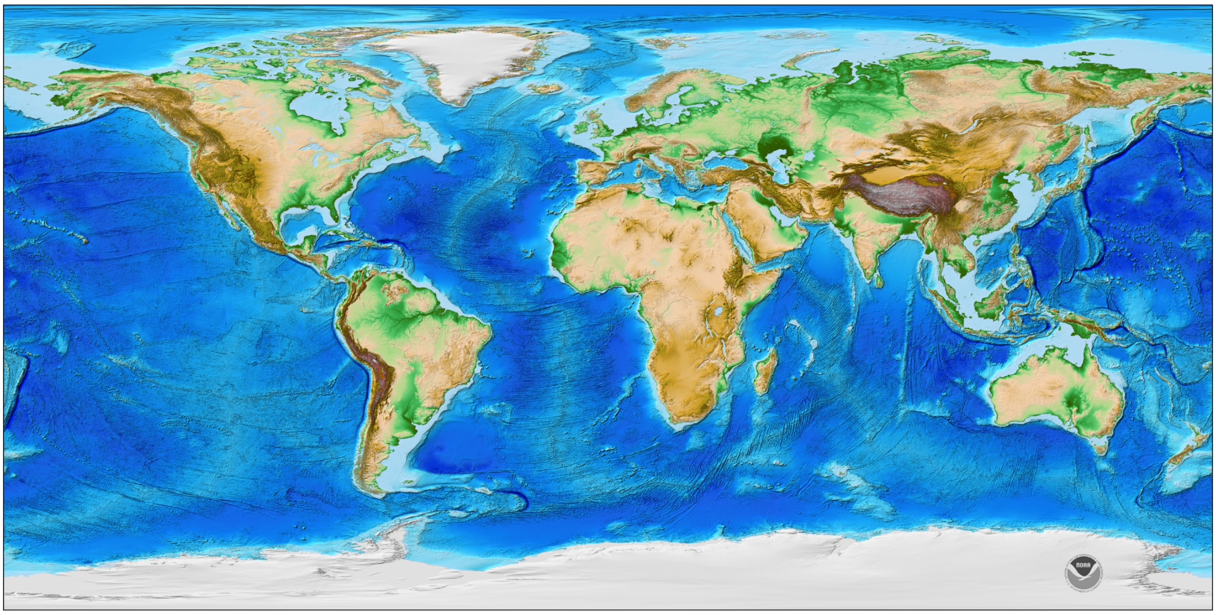


Figure 1.1: *Global relief model of the Earth (ETOPO1, Amante and Eakins, 2009). Spreading centers (mid-ocean ridges) appear as extensive mountain ranges within the oceans while the deep ocean trenches are related to subduction zones. Hot spots, likely to be the surface expression of mantle plumes, form linear chains of seamounts or oceanic Islands (e.g. Hawaiian – Emperor seamount chain).*

accretionary regions at some subduction zones, as well as conductive cooling at the base of the continental lithosphere. Among the processes that reduce the continental mass are erosion at the continent's surface but also crustal (brittle) or lithospheric (ductile) delamination at its base. While delamination processes itself are hidden in the Earth's interior, associated surface expressions can be observed and linked to the underlying processes. The origin of the Sierra Nevada mountains in the western USA, for example, is thought to be related to a delamination process (e.g. Manley et al., 2000; Lee et al., 2000).

Oceanic lithosphere, on the other hand, is mostly hidden from the human eye as it is covered by the oceans and only easily accessible in regions where tectonic forces have uplifted its uppermost part so that it is emplaced within continental crust (e.g. the Semail ophiolite in Oman or Troodos ophiolite in Cyprus). Oceanic lithosphere is essentially composed of a specific sequence of geologic units: A 4-8 km thick layer of basaltic rocks (the oceanic crust) which, in many regions, is covered by a few 10s to 100s of meters thick layer of sediments. Underneath the oceanic crust the so-called Mohorovicic discontinuity (commonly called *Moho* for short hand) marks the transition to the rocks of the Earth's uppermost mantle, from which the basaltic melts have been extracted. Mantle rocks are generally refereed to as *peridotites*, but they are further petrologically classified depending on their degree of melt extraction (i.e. the degree of depletion). With progressing melt extraction, an initially fertile peridotite (*lherzolite*) becomes a *harzburgite* and ultimately a *dunite*, from which no more basaltic melts can be extracted. Mid-ocean ridge (MOR) processes leading to both the extraction of melts and the generation of oceanic lithosphere are a major subject of this thesis.

Mid-ocean ridges are the most active volcanic areas on Earth. These spreading centers form a connected mountain range of about 80,000 km length (see Fig. 1.1) that processes about 20 km^3 of magma every year (i.e. 2.5–3 km^2 of new seafloor is created per year). The morphology of the seafloor created is related to the spreading rate (e.g. Small, 1998), which is the relative motion of the two diverging plates. While slow-spreading ridges such as the Mid-Atlantic ridge (MAR) typically show a pronounced axial valley on both sides of which oceanic crust rises hundreds of meters to few kilometers, fast-spreading ridges like the East Pacific Rise (EPR) reveal a considerably different morphology. Here, the ridge axis is located on a bathymetric high and seafloor subsides on either side. These differences are thought to result from the larger amount of thermal energy that is transported towards shallow depths beneath fast-spreading ridges, which leads to a higher and more continuous melt production and a thinner axial lithosphere.

Basaltic crust, created at the mid-ocean ridge, represents the upper part of the oceanic lithosphere. The latter is defined because of its deformation behavior rather than because it is a lithologically distinct unit. The lithosphere forms a strong and highly viscous boundary layer that overlies ductile mantle rocks. Its comparatively high strength results mainly from the conductive cooling at its surface (i.e. the seafloor) so that the thickness of the oceanic plate correlates well with its age within few 100 km of the spreading center (e.g. Johnson and Carlson, 1992).

In order to reveal the driving forces behind plate motion and melt production at MOR we have to look closely at the Earth's energy budget. The Earth's preferred way of losing its internal heat is thermal convection. As opposed to thermal conduction, which represents a diffusive transport of thermal energy without a transport of material, thermal convection is associated with the redistribution of material (i.e. mantle flow). Hotter material being less dense can overcome viscous resistance and will rise buoyantly until either a horizon of same density is reached or some mechanical barrier stops the upward motion. Two different convective regimes are considered to be important in the Earth's mantle: (1) a large scale convection that organizes into so-called convection cells (e.g. Hansen and Ebel, 1988; Stemmer et al., 2006) and (2) localized diapiric upwellings that transport mass and thermal energy on much shorter time scales. The latter geodynamic features have become generally known as mantle plumes (Morgan, 1971), and are causally linked to so-called hot spots (Wilson, 1963), which are assumed to be their associated surface expression. Mantle plumes that rise in the vicinity of a spreading center can interact with the mid-ocean ridge and cause variations in ridge morphology, crustal thickness and crust composition. Galapagos Island and Iceland are the most prominent examples for this so-called plume-ridge-interaction.

Since the Earth is neither shrinking nor expanding, the generation of new lithosphere at spreading centers must be balanced by a destructive process that removes oceanic lithosphere. The subduction of oceanic plates at convergent margins is this process. Here distinct lithological units — oceanic and continental sediments, basalts of the oceanic crust and ocean island basalts (OIB), as well as the depleted lithosphere mantle rocks — are recycled and transported back into the mantle from which most of they originated millions of years before. The competing processes of plate formation at MOR and subduction have reached a quasi-steady-state that continuously re-surfaces the oceanic part of the Earth.

All studies that seek detailed insights into processes beyond this rough classification are complicated by the inaccessibility of the regions in which melting processes and mantle flow take place. Indirect measurements of physical properties of the uppermost mantle can be achieved by means of active seismic, seismic tomography, gravimetry, geo-electric, geo-magnetic, electromagnetic and other methods. However, the acquisition of high-quality data and their interpretation remain challenging. The structural, mineralogical and geochemical analysis of rock samples, obtained from ophiolites or recovered by either dredging atop or drilling into the oceanic crust, also give further insights into the processes that led to the formation of these rocks.

Due to the direct inaccessibility of the deeper parts of the Earth's crust and mantle, numerical modeling of geodynamic processes has become an important tool for testing different ideas with respect to their applicability to the Earth. Simply speaking, in a numerical geodynamic model the processes detailed above are simplified (i.e. approximated) and put into a self-consistent physical relationship to one another. This highlights both the great power but also the crux of numerical modeling: on the one hand, numerical models link the mostly independent observations of the previously named methods into an interconnected framework. The different processes (e.g. heat conduction, the slow motion of mantle rocks, generation of melts, etc.) are related to one-another so that they can interact similarly to how they interact in the Earth's interior where they occur simultaneously. On the other hand, formulating the equations and deriving numerical solutions within a reasonable amount of time forces us to simplify (approximate) almost all of the individual processes.

Once a numerical model has been formulated, the outcome of an experiment[2] depends strongly on the chosen parameters and approximations that are used to describe each individual process. For instance, the assumption on how effectively the melt present at grain boundaries within a rock will change the rock's response to applied stresses is subject

[2] The expression *numerical experiment* rather than *numerical simulation* is used throughout this thesis to highlight the uncertainties that are inherent in numerical models of geodynamic problems. An experiment shows "how it could be for the parameters chosen and assumptions made" whereas a simulation implies to be "the correct answer to the given problem description"

to some uncertainty that, unfortunately, can have a considerable impact on the outcome of a numerical experiment. From the numerical modeler's point of view a solution for this issue is to conduct many experiments in which single parameters are varied systematically to estimate their ”importance” to the results obtained. While this approach is realizable for 1-D and (maybe) 2-D models, it represents a huge problem for time-consuming 3-D calculations. Chapter 3 will present some ideas how to use 1-D results to reduce the parameter space of similar 2-D problems. Both Chapter 3 and 4 will show how 2-D results can be advantageous prior to studying similar 3-D problems.

In any case, a reduced parameter space, even at the expense of additional approximations to be made, can help to derive more robust results than conducting only very few experiments to try to explore a large parameter space. In addition, this is a truly interdisciplinary field of research as results from mineral physics and fluid dynamics provide critical constraints at the same time. If, for example, geochemical analysis of mid-ocean ridge basalts (MORB) provides evidence that the mineral garnet was present during the melting process while laboratory studies on the stability of crystals at certain temperature-pressure conditions suggest that garnet is stable below only 70 km depth, the combination of these results (onset of melting at 70 km depth or lower) is an extremely important constraint for numerical models as it helps to narrow the parameter space.

1.2 Numerical modeling of geodynamic processes

Numerical modeling of mantle flow in the Earth's interior requires solutions for the thermal evolution and the steady state viscous flow (i.e. Stokes flow) of the mantle rocks. The rheological behavior of mantle rocks at 1000°C and above can be approximated as viscous creep of an incompressible material. The so-called Maxwell relaxation time, the ratio between viscosity and elasticity, is a characteristic timescale to describe the deformation behavior of a material. For hot mantle rocks, this timescale is on the order of few 100 to few 1000 years. Thus, on small time scales the mantle behaves like an elastic material, for example during the passage of seismic waves and during natural oscillations of the entire body of the Earth. On larger time scales, however, it deforms like a viscous fluid.

With the exception of earthquakes, most large-scale geodynamic processes within the Earth occur on time scales much longer than the above mentioned 1000 years. The geodynamic problems studied here usually take place over few hundred thousands to millions of years, and a typical time step in the numerical model is on the order of 5,000 to 50,000 years. To further illustrate the temporal and spatial dimensions consider a typical numerical domain that covers some 100 km in all spatial directions, and a mantle motion similar to the speed of the fastest moving lithospheric plate (10 cm/yr = 100 mm/yr =

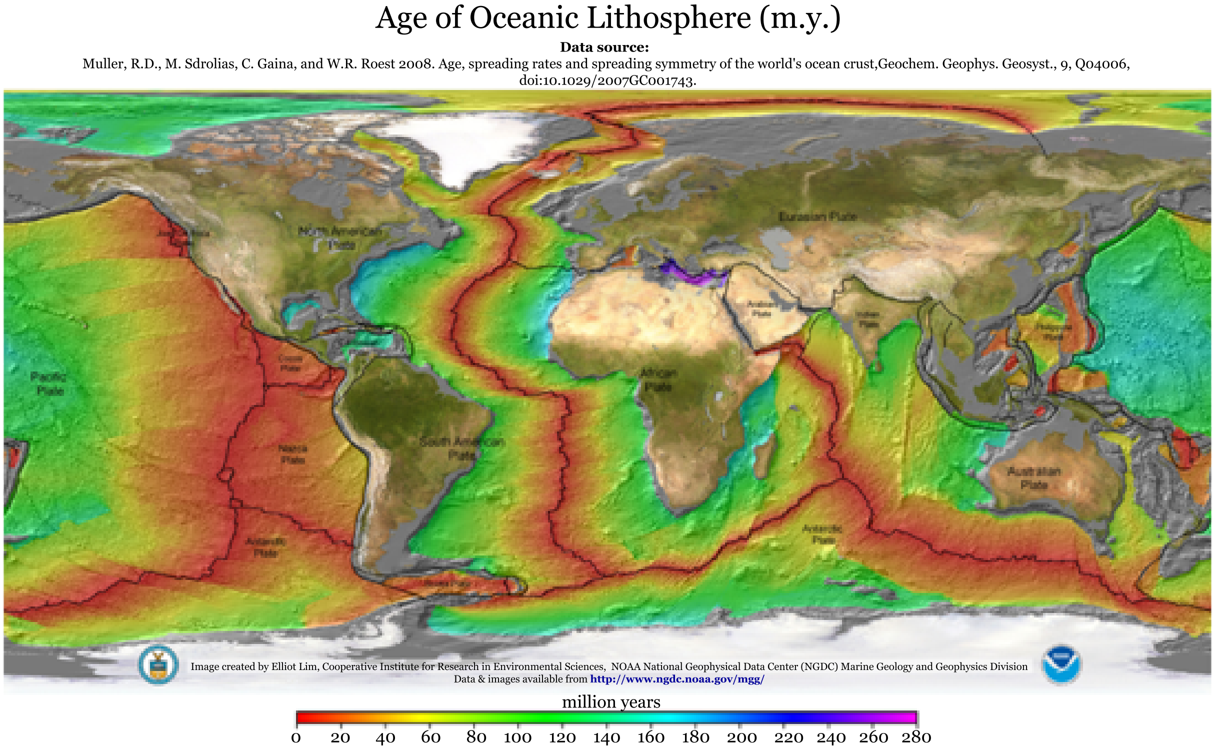

Age of Oceanic Lithosphere (m.y.)
Data source:
Muller, R.D., M. Sdrolias, C. Gaina, and W.R. Roest 2008. Age, spreading rates and spreading symmetry of the world's ocean crust,Geochem. Geophys. Geosyst., 9, Q04006, doi:10.1029/2007GC001743.
North American Plate
Eurasian Plate
Pacific Plate
Cocos Plate
Arabian Plate
Indian Plate
Philippine Plate
African Plate
Nazca Plate
South American Plate
Australian Plate
Antarctic Plate
Antarctic Plate
Scotia Plate
Image created by Elliot Lim, Cooperative Institute for Research in Environmental Sciences, NOAA National Geophysical Data Center (NGDC) Marine Geology and Geophysics Division
Data & images available from http://www.ngdc.noaa.gov/mgg/
million years
0
20
40
60
80
100
120
140
160
180
200
220
240
260
280

Figure 1.2: *Age of oceanic lithosphere on the Earth.*

100 km/Myr). Over a numerical time step of 5,000 years the transport distance would be 500 m, which is similar to the numerical discretization (i.e. finite element node spacing, see below) in the highest resolution regions.

In case of viscous flow, the dynamic viscosity is the parameter that describes how a fluid flows in response to applied forces. Viscosity strongly depends on both temperature variations as well as compositional changes. One of the most challenging aspects of numerically modeling mantle flow is to include realistically high viscosity variations and still be able to derive precise solutions for velocity and pressure within a reasonable amount of computation time. This is discussed in detail in Section 2.6.

Mantle flow is driven by buoyancy forces resulting from density variations. In numerical models, additional driving forces result from the motion of rigid lithospheric plates, which are often imposed as a fixed kinematic boundary condition at the domain top. Adjacent underlying mantle is forced to flow accordingly, which might be viewed as an artificial influence on the numerical experiment. However, plate motion on Earth is ultimately driven by density variations, too, so that kinematic boundary conditions account for buoyancy forces that occur outside of the numerical domain but still affect the flow inside the domain. These forces on oceanic lithosphere could, for example, result from pulling forces of a subducting old oceanic lithosphere or the viscous coupling of thicker continental lithosphere to convective mantle flow underneath.

Density variations result from temperature contrasts and changes of the mantle composition. Buoyant upwelling of mantle plumes is a consequence of their excess temperature with respect to surrounding mantle, whereas the lithosphere generated at mid-ocean ridges is buoyantly stable because of compositional changes. These changes in composition result mainly from melting processes, during which some chemical elements preferentially enter the melt phase, while others prefer to stay in the host rock. A significant, though still unknown fraction of the melt is extracted and migrates to the surface to form the oceanic crust. The about 60–120 km thick region of the mantle from which the melt has been extracted (exact numbers are still subjects of debate), has lost a fraction of certain chemical elements, and is called ”depleted in these elements” or just ”depleted mantle”. One of the elements that preferentially enter the melt phase is the comparably dense chemical element iron, which is the main reason why the mantle residue is less dense than the mantle rocks prior to melting (i.e. depletion buoyancy). As opposed to thermal buoyancy, compositional changes are essentially permanent and irreversible, because chemical diffusion (acting to re-homogenize the chemical composition) is much slower than thermal diffusion in mantle rocks. As the oceanic lithosphere ages and cools, negative thermal buoyancy works against the positive compositional buoyancy and eventually dominates after about 40–50 Myr (Oxburgh and Parmentier, 1977). The plate becomes heavier than underlying mantle and would sink during subduction. For this reason no oceanic plate (with very few exceptions) is older than about 200 Myr (Fig. 1.2).

1.3 Outline and objectives of this thesis

The topics covered in this thesis are grouped into three parts. Chapter 2 documents the development of two new numerical models ($\mathbf{M3_{tri}}$ and $\mathbf{M3_{tet}}$[3]) that can be used to study the major geodynamic processes that are mentioned above in two and three dimensions, resp. The physical equations that describe different aspects of the geodynamic problems will be discussed as well as their approximation by numerical methods. Efficient strategies for solving the arising matrix equations will be presented and explored in detail.

In Chapter 3, the numerical model is extended by a parameterization of the melting processes in the mantle. The formulation presented for the melting of a heterogeneous mantle represents a new combination of two previously published parameterizations. The advantages of this formulation developed in this thesis include (1) the ability to handle mantle rocks that are composed of different lithological units, (2) to consider the effects of different water contents in each lithology, and (3) to not require an iterative smoothing

[3] **M3** denotes **M**antle convection and **M**elting code written in **M**ATLAB (www.mathworks.com). Subscripts ”tri” and ”tet” refer to the triangular and tetrahedral elements that are used in the 2-D and 3-D version of the numerical code, resp.

when implemented in 2-D or 3-D mantle convection codes (which was required in previous formulations). Afterwards, selected key results of 2-D and 3-D numerical models on mantle flow and melting processes at mid-ocean ridges are presented.

While Chapter 3 focusses on the formation of oceanic lithosphere, Chapter 4 addresses processes that are related to the end of the lifecycle of an oceanic plate. As the former oceanic lithosphere undergoes subduction aqueous fluids are released into the rocks located in the so-called mantle wedge, that is the region between the underlying subducting slab[4] and the overlying volcanic arc. This so-called slab dehydration is conceptually supported by geochemical analyses of island arc lavas (Elliott, 2003). The change in mantle rock's composition as it absorbs these fluids may lead to convective instabilities that could explain the patterns in the distribution of volcanic centers in the volcanic arcs above subduction zones. The conditions under which the instabilities could emerge as well as their spatial and temporal patterns are studied in this last chapter.

[4] The subducting oceanic lithosphere is commonly referred to as *slab*.

Chapter 2

Development of 2-D and 3-D mantle convection models

2.1 Introduction

2.1.1 Numerical modeling of geodynamic processes

The mathematical formulation of the numerical models is based on the conservation laws for mass, momentum, and energy. The strong temperature dependence of viscosity, which controls the viscous flow of the mantle that in turn affects the thermal evolution through advection, makes these coupled equations strongly non-linear. A standard technique is therefore to march forward in time by successively solving for viscous flow for given viscosity and density fields (i.e. a given temperature field) at time t_1, and then use this flow field to calculate a new temperature field at time $t_2 = t_1 + \Delta t$. This new temperature field is used for the viscous flow calculation at t_2 and so forth. As mentioned above, the time step Δt is on the order of few 1,000 to few 10,000 years and an experiment usually covers 100,000 to few tens of million years.

In the next section (2.1.2), I will give a short introduction to the finite element method (FEM), which is used to derive the viscous flow and thermal diffusion solutions in all numerical models developed in this thesis. The mathematical description of the temperature advection-diffusion problem, its numerical approximation, and solution is discussed in Section 2.2. The viscous flow solution, which consumes considerably more computer time and memory than the solution for heat transport, is presented in Section 2.3. After an introduction to numerical solution techniques for solving the matrix equations arising from the finite element formulations of the heat conduction and viscous flow problem (Section 2.4), I will present a combination of these numerical solvers in Section 2.5 that takes advantage of their distinct strengths. This combined algorithm is used to solve the viscous flow problem, which is discussed in Section 2.6.

2.1.2 A short introduction to FEM

All numerical codes that are used to study the geodynamic problems in this thesis, have been developed by the author in collaboration with Prof. Jason Phipps Morgan. These codes are based on the finite element method (FEM), more precisely, on the Galerkin finite element method. Before discussing the mathematical description and numerical solution of the viscous flow and thermal advection-diffusion problems, I will give a brief introduction to the FEM. This introduction is based on chapter 1 in Hughes (2000). I adopted Hughes' more compact notation for partial derivatives:

$$u_{i,j} = \frac{\partial u_i}{\partial x_j} = \nabla u \tag{2.1}$$

In words, $u_{i,j}$ denotes the derivative of the i-component of u with respect to j, where the domain of definition for i and j will be given. Similarly

$$u_{,xx} = \partial^2 u / \partial x^2 \tag{2.2}$$

Suppose we want to solve a 1-D boundary value problem on a domain Ω ranging from $x = 0$ to $x = 1$ that is defined by the partial differential equation (PDE) and boundary conditions:

$$u_{,xx} + f(x) = 0 \quad \text{, with boundary conditions } u(1) = g \text{ and } -u_{,x}(0) = \psi. \tag{2.3}$$

Two types of boundary conditions are imposed here: a Dirichlet boundary condition $u(1) = g$ that prescribes the value of u at the point $x = 1$ and a Neumann boundary condition prescribing the first derivative of the solution vector at $x = 0$: $-u_{,x}(0) = \psi$.

As opposed to finite difference methods (FDM) that use the above *strong form* of the problem definition as their starting point, the FEM is based on the so-called *weak* or *variational form*. For deriving the weak form, we have to define two sets (or collections) of functions: (1) a collection $\mathscr{U}$ of trial solutions u that already satisfy the Dirichlet boundary conditions, and (2) a collection $\mathscr{W}$ of weighting functions w that are zero at points where Dirichlet boundary conditions are imposed.

$$\mathscr{U} = \{u | u \in H^1, u(x = 1) = g\} \tag{2.4}$$

$$\mathscr{W} = \{w | w \in H^1, w(x = 1) = 0\} \tag{2.5}$$

H^1 states that the derivatives of the functions must be square-integrable, that is $\int_0^1 (\partial w / \partial x)^2 \, \mathrm{d}x < \infty$. The first steps in deriving the weak form are pre-multiplying each term by weighting functions and integrating over the domain.

$$\int_0^1 w \, u_{,xx} \, \mathrm{d}x + \int_0^1 w f \, \mathrm{d}x = 0 \tag{2.6}$$

We can reduce the order of the derivative of the trial functions u in the first integral using integration by parts:

$$\int_0^1 (w\, u_{,x})_{,x}\, \mathrm{d}x - \int_0^1 w_{,x}\, u_{,x}\, \mathrm{d}x + \int_0^1 wf\, \mathrm{d}x = 0 \quad \text{(integration by parts)} \tag{2.7}$$

$$-w(0)u_{,x}(0) + w(1)u_{,x}(1) - \int_0^1 w_{,x}\, u_{,x}\, \mathrm{d}x + \int_0^1 wf\, \mathrm{d}x = 0 \tag{2.8}$$

Note that now a first derivative of the weighting functions ($w_{,x}$) is required. The first integral in (2.7) can be explicitly written as done in (2.8). These new terms are defined on the boundary of the domain and we know that

$$w(0)u_{,x}(0) = w(0)\psi \qquad \text{(Neumann boundary condition)} \tag{2.9}$$

$$w(1)u_{,x}(1) = 0 \qquad \text{(because } w(x=1) = 0\text{, see (2.5))} \tag{2.10}$$

Given f, g, and ψ as above, the weak form of (2.3) is

$$\int_0^1 w_{,x}u_{,x}\, \mathrm{d}x = \int_0^1 wf\, \mathrm{d}x + w(0)\psi \tag{2.11}$$

The weak form is equivalent to the strong form and has the same unique solution u for the defined problem. Neumann boundary conditions are also called natural boundary conditions, because they are implied by the variational equation and appear "naturally" on the right-hand side (RHS) when deriving the weak form.

The FEM solves the weak form (2.11) by defining approximate trial solutions $u^h \approx u$ and weighting functions $w^h \approx w$. The superscript "h" is used to indicate these functions as the discretized counterparts of the functions defined in (2.4) and (2.5), resp. The discretized functions are associated with a mesh in the numerical domain Ω, on which a solution for the problem at hand is sought. In the *Galerkin* FEM, the u^h are constructed with the help of functions v^h from the collection $\mathscr{W}$:

$$u^h = v^h + g^h \qquad \text{where } v^h \in \mathscr{W} \tag{2.12}$$
$$\text{and } g^h \in \mathscr{U} \text{ so that } g^h(x=1) = g$$

Using this definition, (2.11) becomes

$$\begin{aligned} \int_0^1 w^h_{,x}u^h_{,x}\, \mathrm{d}x &= \int_0^1 w^h f\, \mathrm{d}x + w^h(0)\psi \\ \Leftrightarrow \int_0^1 w^h_{,x}v^h_{,x}\, \mathrm{d}x &= \int_0^1 w^h f\, \mathrm{d}x + w^h(0)\psi - \int_0^1 w^h_{,x}g^h_{,x}\, \mathrm{d}x \end{aligned} \tag{2.13}$$

Equation (2.13) represents a coupled system of linear algebraic equations, in which all known terms have been moved to the RHS. We now have to define what the weighting functions ($w^h \cap v^h \in \mathscr{W}$) and trial solutions ($u^h \cap g^h \in \mathscr{U}$) actually are. Let us construct the weighting functions using all linear combinations of the functions N_A, where $A = 1, 2, ..., n$ and n is the number of unknowns in the 1-D problem:

$$w^h = \sum_{A=1}^{n} c_A N_A \text{ , where } c_A (A = 1, 2, ..., n) \text{ are constants} \tag{2.14}$$

$$v^h = \sum_{B=1}^{n} d_B N_B \text{ , where } d_B (B = 1, 2, ..., n) \text{ are constants} \tag{2.15}$$

The N_A's (and N_B's) are the so-called shape, basis or interpolation functions. For the definition of the trial solutions we need the above definition of v^h but also have to take into account the Dirichlet boundary condition (because $u^h(1) = g$ is required, whereas $v^h(1) = 0$).

$$\begin{aligned} u^h &= v^h + g^h \\ &= \sum_{B=1}^{n} d_B N_B + g N_{n+1} \quad \text{, with } N_{n+1}(x = 1) = 1 \end{aligned} \tag{2.16}$$

Substituting (2.14) and (2.16) into (2.13) gives

$$\begin{aligned} \int_0^1 \sum_{A=1}^{n} c_A N_{A,x} \sum_{B=1}^{n} d_B N_{B,x} \,\mathrm{d}x &= \int_0^1 \sum_{A=1}^{n} c_A N_A f \,\mathrm{d}x + \sum_{A=1}^{n} c_A N_A(0) \psi \\ &\quad - \int_0^1 \sum_{A=1}^{n} c_A N_{A,x} g N_{n+1,x} \,\mathrm{d}x \end{aligned} \tag{2.17}$$

Considering the bilinearity of the integrals in (2.17), the order of summation and integration can be changed, so that

$$0 = \sum_{A=1}^{n} c_A G_a \quad \text{, where} \tag{2.18a}$$

$$G_A = \sum_{B=1}^{n} \int_0^1 N_{A,x} N_{B,x} d_B \,\mathrm{d}x - \int_0^1 N_A f \,\mathrm{d}x - N_A(0)\psi + \int_0^1 N_{A,x} N_{n+1,x} g \,\mathrm{d}x \tag{2.18b}$$

Equation (2.18) holds for all weighting functions $w^h \in \mathscr{W}$, that is, for all coefficients c_A ($A = 1, 2, ..., n$). Thus, in order for (2.18) to be true, G_A must be identically zero for each $A = 1, 2, ..., n$. Consequently:

$$\sum_{B=1}^{n} \int_0^1 N_{A,x} N_{B,x} d_B \,\mathrm{d}x = \int_0^1 N_A f \,\mathrm{d}x + N_A(0)\psi - \int_0^1 N_{A,x} N_{n+1,x} g \,\mathrm{d}x \tag{2.19}$$

Note that the coefficients c_A in the definition of the weighting functions ($w^h = \sum_{A=1}^{n} c_A N_A$) disappeared from the equation about to be solved, but the shape functions N_A remain.

Equation (2.19) represents a set of n linear algebraic equations that can be written in matrix form.

$$\sum_{B=1}^{n} K_{AB} d_B = F_A \tag{2.20}$$

$$[K]_{AB} = \int_0^1 N_{A,x} N_{B,x} \,\mathrm{d}x \tag{2.21}$$

$$(F)_A = \int_0^1 N_A f \,\mathrm{d}x + N_A(0)\psi - \int_0^1 N_{A,x} N_{n+1,x} g \,\mathrm{d}x \tag{2.22}$$

$$\Rightarrow \mathbf{K}\, d = F \tag{2.23}$$

$[K]_{AB}$, and $(F)_A$ denote single elements of the matrix $\mathbf{K}$ and vector F, resp. The physical meaning of matrix $\mathbf{K}$ and vector F depends on the type of problem that the PDE describes. In elasticity or viscous flow, $\mathbf{K}$ is called stiffness matrix and F force vector, whereas in thermal diffusion problems, $\mathbf{K}$ would be the conductivity matrix and F a vector associated with heat flow.

The solution $d = \mathbf{K}^{-1} F$ of the matrix equation is the coefficients for the shape functions N_B that construct the trial solutions. The idea behind the FEM may therefore be described in one sentence: Given a set of basis functions N, the FEM solves for the coefficients that "best" fit these N's to the solution of the PDE. Using the coefficients and the N's (i.e. the definition of the trial functions in (2.16)), the finite element approximation to the solution of the PDE at any point x inside Ω is given by $u^h(x) = \sum_{A=1}^{n} d_A N_A(x) + g N_{n+1}(x)$. This points out a fundamental characteristic of the FEM: The choice for N has a major impact on the quality of the solution, because these functions ultimately approximate the solution of the PDE.

The quality of a FEM solution depends substantially on the number of discrete points in the mesh and the definition of the shape functions N. The discrete points in the FEM are called *nodes*, and each one is connected to other nodes in its neighborhood by so-called *elements*. Each shape function is defined over the entire domain, but usually in such a way that $N_A = 1$ at node A and $N_A = 0$ at all nodes $B \notin A$. It has the advantage that the shape functions have a zero value everywhere except in the neighborhood of their associated node, more specifically, within the elements that connect to this node. This is illustrated for sections of 1-D and 2-D meshes in Fig. 2.1.

Another property of standard shape functions is that $\sum_A N_A(x) = 1$ at any $x \in \Omega$, which qualifies them to be easily used for interpolation purposes. The number of nodes in an element defines the polynomial order of the shape functions. A 2-node element in 1-D has linear shape functions, so that the trial and weighting functions are constructed from piecewise linear approximations. On the other hand, a 3-node 1-D element has quadratic shape functions, so that potentially a more accurate approximation to the solution of the

PDE can be achieved for the same number of unknowns. It is therefore important to consider the properties of the PDE at hand before selecting the type of element. Most important are the orders of the partial derivatives in the PDE, where one should use shape functions of same or higher order, so that the finite element solution can capture the properties of the PDE solution. This is discussed in Section 2.6, where the numerical formulation of the viscous flow problem is presented.

As mentioned above, solution d of (2.23) is the coefficients for the trial solutions. However, because each shape function has a value of one at a single node and is zero at all other nodes, these coefficients also represent the solution of the PDE at each node. To illustrate this, assume we have a solution for d and want to calculate the value of u at node B:

$$u^h(x_B) = \sum_{A=1}^{n} d_A N_A(x) + gN_{n+1}(x) \qquad \text{, Eq. (2.16)} \tag{2.24}$$

$$= d_B N_B(x_B) \qquad \text{, because } N_A = 0 \text{ for all } A \neq B \tag{2.25}$$

$$= d_B \cdot 1 \tag{2.26}$$

Nevertheless, (2.16) is required if one is interested in the finite element solution of u at locations between nodes, for instance, if a variable has to be determined at a back tracking point (see Section 2.2). Quadratic shape functions should be used with caution in interpolations because they can produce over- and undershoots if nodal values show strong gradients.

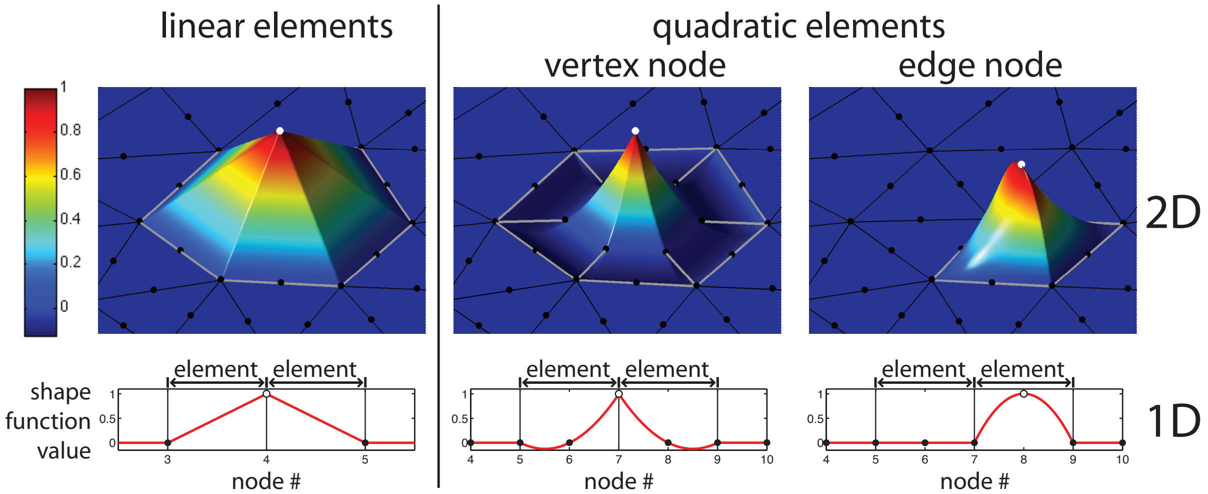


Figure 2.1: *Elements with linear (left column) and quadratic shape functions (mid and right column) in one and two dimensions. Each shape function N_A has a value of one at its associated node A (white dot) and is zero at all other nodes (black dots). N_A has non-zero values only within elements connecting to node A (these are enclosed by grey lines in the 2-D case). The quadratic elements in 2-D have two types of shape functions: those associated with a vertex (corner) node and those corresponding to a node on the element's edge. In 1-D, the latter is located within an element and only connects to the nodes at the ends of the element. Functions of this type are also called "bubble" functions.*

2.2 Thermal advection and diffusion

2.2.1 Mathematical formulation

The time dependent temperature evolution depends on advection and diffusion of heat as well as the generation (e.g. radioactive decay) or consumption of heat (for example by latent heat cooling). The temperature field in two dimensions is described using the equation for energy conservation

$$\frac{\partial T}{\partial t} + \left(v_x \frac{\partial T}{\partial x} + v_z \frac{\partial T}{\partial z} \right) - \kappa \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial z^2} \right) + \Psi = 0 \tag{2.27}$$

or, using vector notation:

$$\frac{\partial T}{\partial t} + \vec{v} \nabla T - \kappa \nabla^2 T + \Psi = 0 \tag{2.28}$$

Here, v_x and v_z are the velocity components of the vector field $\vec{v}$, x and z are spatial coordinates, T is the temperature field, t is time, κ is thermal diffusivity, and Ψ is a source term. κ is assumed to be constant throughout the domain and thus can be moved in front of the two spatial derivatives. We use operator splitting, that is we solve for advection and diffusion separately rather than simultaneously. The advection part is done by a semi-Lagrange method, while the diffusion part is solved for using a Galerkin finite element formulation. The numerical formulation of both parts will be discussed next. The advantages of the operator splitting approach and some tests using other formulations are discussed at the end of this section.

2.2.2 Numerical formulation

Advection of temperature

The simplest way to advect a field defined at discrete points on a finite element mesh is to advect the nodes along the characteristics of the flow field. This full-Lagrange method has been tested, and its advantages and disadvantages will be discussed below. Due to some technical difficulties that arise in this formulation, a similar method is used that does not require the mesh to move and deform. This semi-Lagrange method (as well as the full-Lagrange method), however, require the separation of diffusion and advection, which is a common practice generally known as operator splitting. This has the benefit of avoiding instabilities in the temperature field resulting from a combined advection-diffusion formulation, which requires additional stabilization terms. An example for a method that solves simultaneously for advection and diffusion is the Streamline-Upwind-Petrov-Galerkin method (SUPG). It will also be discussed below.

The idea behind the semi-Lagrange method is to find the coordinates $\hat{x}_A$ of the material that will arrive at node A (located at x_A) at the end of a time step. $\hat{x}_A$ is called the back-tracking point of node A, and its location depends on the characteristics of the flow field and the length of the time step. The quantity to be advected is interpolated at $\hat{x}_A$, and this value becomes the new nodal value at x_A. This approximates a transport of material from $\hat{x}_A$ to x_A. Potential inaccuracy in the semi-Lagrange method results from two operations that are relatively easy for the user to control, namely: (1) finding the location of the back-tracking points and (2) accurately interpolating the quantity at the back-tracking points.

Finding the coordinates $\hat{x}$ requires a back-tracking step that follows the trajectories backwards in time. The simplest (and least accurate) way to calculate the coordinates of the back-tracking point is $\hat{x}_A = x_A - \Delta t \cdot \vec{v}_A$, where $\vec{v}_A$ is the velocity vector at node A and Δt is the length of the time step. The inaccuracy here results from only using the nodal velocity vector for the entire back-tracking step. However, the velocity field generally varies between back-tracking point and node so that including more velocity information as we go back in time leads to more accurate results. Higher order schemes such as predictor-corrector method or Runge-Kutta methods can be used to integrate along the characteristics, but their better accuracy comes at the cost of increased computational effort.

For the laminar flow fields in mantle convection, the 2nd order accurate predictor-corrector scheme is a good compromise between accuracy and performance. This scheme first evaluates the coordinates of a back-tracking point located half-way back, i.e. $\hat{x}_{1/2} = x - \frac{\Delta t}{2} \cdot \vec{v}$, then interpolates the velocity field $\vec{v}_{1/2}$ at $\hat{x}_{1/2}$, and uses this velocity for the full back-tracking step starting again at the node: $\hat{x}_A = x_A - \Delta t \cdot \vec{v}_{1/2}$. I see no significantly improved accuracy when using higher order schemes, given that the length of the time step is bounded by the diffusion problem (see below).

The second source of inaccuracy in the semi-Lagrange method are interpolation errors at the back-tracking points. This problem is much more serious and more difficult to get a grip on. Each interpolation is based on the nodal values surrounding the back-tracking point, and the interpolated values become the new nodal values for the next time step. Therefore, interpolation errors accumulate quickly and result in a numerical diffusion that smooths out steep gradients in the advected field in all spatial directions. This is especially an issue in 3-D, where more surrounding nodes contribute to an interpolation value.

I use a recently developed spline-like cubic interpolation scheme (Shi and Phipps Morgan, 2010) that operates on the unstructured meshes in 2-D (triangular elements) and 3-D (tetrahedral elements). In addition to the nodal value of the scalar field, it takes into account the spatial derivatives of the field at the nodes surrounding the interpolation point. Using this additional information, variations in the scalar field that lie in-between nodes can be considered during the interpolation. Compared to a linear interpolation (e.g. a classical upwind-like scheme) that smoothes gradients in the field similar to a diffusion operator, the cubic interpolation maintains steep gradients for many more interpolation steps. A comparison between the cubic interpolation and a linear interpolation in a semi-Lagrange method are presented in the next section. Also part of the comparison are a full-Lagrange approach and a coupled advection-diffusion solver.

In reality, advection and diffusion happen simultaneously, whereas operator splitting treats these processes sequentially: The field is diffused first and the resulting field is advected afterwards. One can slightly improve the accuracy by placing the diffusion in the middle of the advection step, that is, advect for half of the time step, diffuse over the whole time step, then advect the diffused field over the second half of the time step. However, a better approximation is to calculate the diffusive change in temperature at the back tracking point at the start of diffusion $(\partial\hat{T}_0/\partial t)_{\text{diff}}$ as well as at the node at the end of the time step $(\partial T_1/\partial t)_{\text{diff}}$, and use the average of both during the advection step. This can be written as

$$T_1 = \hat{T}_0 + 0.5 \cdot \left(\left(\frac{\partial T_1}{\partial t} \right)_{\text{diff}} + \left(\frac{\partial \hat{T}_0}{\partial t} \right)_{\text{diff}} \right) \cdot \Delta t \tag{2.29}$$

Another advantage of this formulation is that a source term can be avoided in the diffusion equation but be fully included during the semi-Lagrange advection step. If, for instance, decompression melting of material moving from the back tracking point to the node consumes the latent heat ΔH, this negative heat can be added to the RHS of equation (2.29). The approximation implied by this method is that the diffusion of the heat ΔH is considered in the next time step, not in the current time step. On the other hand, including the heat as a source term in the diffusion operator would essentially postpone its advection to the next time step. I prefer the former method compared to the more common source term in the diffusion solver for two reasons: 1) A spatially and temporarily changing source term can lead to numerical oscillations in the diffusion solver that downgrade the quality of the evolving temperature field; this kind of source term emerges during experiments with transient melting processes. 2) Since advective heat transport dominates over diffusive transport in the Earth's mantle, considering the advection of new latent heat immediately seems to be more important than including its immediate diffusion.

Thermal diffusion

Based on the energy equation (2.28), a time-dependent diffusion problem including a possible source term can be formulated on a domain Ω. To do so, boundary conditions on the domain edge Γ as well as an initial state T_0 of the temperature field have to be defined.

$$\begin{aligned}
\dot{T} - \kappa\nabla^2 T &= \Psi && \text{on } \Omega && \text{(Eq. (2.28) without advection term)} && (2.30a)\\
T(t_0) &= T_0 && \text{at time } t_0 = 0 && && (2.30b)\\
T &= T_D && \text{on } \Gamma_D && \text{(Dirichlet boundary condition)} && (2.30c)\\
-\vec{n}\kappa\nabla T &= q && \text{on } \Gamma_N && \text{(Neumann boundary condition)} && (2.30d)
\end{aligned}$$

$\dot{T}$ denotes the temperature time derivative $\partial T/\partial t$, $\vec{n}$ is the unit outward normal vector to the boundary Γ and q denotes a heat flux. The boundary Γ is divided into a part Γ_D, on which the temperature is prescribed (i.e. Dirichlet boundary condition), and a part Γ_N, on which $q = -\kappa\nabla T$ is defined (i.e. Neumann boundary condition). For using the FEM, the weak or variational form of (2.30) has to be derived . This is done by the following steps:

$$\int_\Omega w\dot{T}\,\mathrm{d}\Omega - \int_\Omega w\kappa\nabla^2 T\,\mathrm{d}\Omega = \int_\Omega w\,\Psi\,\mathrm{d}\Omega \qquad (2.31a)$$

$$\int_\Omega w\dot{T}\,\mathrm{d}\Omega - \int_\Omega \nabla\cdot(w\kappa\nabla T)\;\mathrm{d}\Omega + \int_\Omega (\nabla w)\cdot(\kappa\nabla T)\,\mathrm{d}\Omega = \int_\Omega w\,\Psi\,\mathrm{d}\Omega \qquad (2.31b)$$

$$\int_\Omega w\dot{T}\,\mathrm{d}\Omega - \int_\Gamma w\,(\vec{n}\kappa\nabla T)\;\mathrm{d}\Gamma + \int_\Omega (\nabla w)\cdot(\kappa\nabla T)\,\mathrm{d}\Omega = \int_\Omega w\,\Psi\,\mathrm{d}\Omega \qquad (2.31c)$$

The above operations include pre-multiplying each term in the PDE (2.30a) by weight functions w and integrating the terms over the numerical domain →(2.31a), and using integration by parts →(2.31b) and divergence (Gauss) theorem →(2.31c) to reduce the second order spatial derivative of T to first order derivatives of w and T. Note that in (2.31) (and from now on), T is associated with trial solutions for temperature (i.e. functions that will eventually describe the temperature solution; see Section 2.1). The weighting functions w have the property to be zero on Γ_D, where the temperature solution is prescribed (see definition of w in (2.5)). Accounting for $w = 0$ on Γ_D and substituting the Neumann boundary condition leads to the weak or variational form of the thermal diffusion problem defined in (2.30):

$$\int_\Omega w\dot{T}\,\mathrm{d}\Omega + \int_\Omega (\nabla w)\cdot(\kappa\nabla T)\,\mathrm{d}\Omega = \int_\Omega w\,\Psi\,\mathrm{d}\Omega - \int_{\Gamma_N} wq\,\mathrm{d}\Gamma_N \qquad (2.32)$$

Next, the Galerkin FEM is used to approximate (2.32) by discretizing the weight functions w and the temperature trial solutions T using the finite element shape functions N. The Galerkin approximations are

$$w^h = \sum_{A\notin A_D} N_A(x)\,w_A \qquad (2.33)$$

$$T^h = \sum_{B\notin B_D} N_B(x)\,T_B + \sum_{B\in B_D} N_B(x)\,T(x_B) \qquad (2.34)$$

In the Galerkin FEM, the same shape functions N are used in the definition of w^h and T^h, i.e. $N_A = N_B$ if $A = B$. In this definition T_B denotes the coefficient for the shape function N_B, which is equal to the temperature at the node B. T^h is set of a continuous functions over the domain Ω that will eventually approximate the temperature solution. In other words, the functions T^h are linear combinations of all shape functions N, and the unknown coefficients T_B have to be evaluated in order to find the solution. The contributions to T^h are separated into a known part at nodes B_D where a Dirichlet boundary condition is applied, and an unknown part $B \notin B_D$ that we are solving for. Since all w^h are zero at B_D, the sum in (2.33) only includes "free" nodes without Dirichlet boundary conditions. Substituting (2.33) and (2.34) into the weak form (2.32) and moving the prescribed temperatures to the RHS yields

$$\begin{aligned}&\int_\Omega \sum_{A\notin A_D} N_A w_A \sum_{B\notin B_D} N_B \dot{T}_B \;\mathrm{d}\Omega + \int_\Omega \sum_{A\notin A_D} \nabla N_A w_A \sum_{B\notin B_D} \kappa \nabla N_B T_B \;\mathrm{d}\Omega \\ &\quad= \int_\Omega \sum_{A\notin A_D} N_A w_A \,\Psi \;\mathrm{d}\Omega - \int_{\Gamma_N} \sum_{A\notin A_D} N_A w_A \, q \;\mathrm{d}\Gamma \\ &\qquad - \int_\Omega \sum_{A\notin A_D} N_A w_A \sum_{B\in B_D} N_B \dot{T}_B \;\mathrm{d}\Omega - \int_\Omega \sum_{A\notin A_D} \nabla N_A w_A \sum_{B\in B_D} \kappa \nabla N_B T_B \;\mathrm{d}\Omega \end{aligned} \tag{2.35}$$

If *nnod* denotes the number of nodes in the mesh and n_D the number of prescribed temperatures, (2.35) represents a set of $n = nnod - n_D$ equations that is true for any weighting function w^h. The bilinearity of the integrals allows to change the order of integration and summation so that (2.35) is equivalent to

$$0 = \sum_{A\notin A_D} w_A G_A \quad \text{, where} \tag{2.36a}$$

$$\begin{aligned} G_A = &- \sum_{B\notin B_D} \left(\int_\Omega N_A N_B \dot{T}_B \;\mathrm{d}\Omega + \int_\Omega (\nabla N_A)\cdot(\kappa \nabla N_B T_B) \;\mathrm{d}\Omega \right) \\ &+ \int_\Omega N_A \,\Psi \;\mathrm{d}\Omega - \int_{\Gamma_N} N_A \, q \;\mathrm{d}\Gamma \\ &- \sum_{B\in B_D} \left(\int_\Omega N_A N_B \dot{T}_B \;\mathrm{d}\Omega + \int_\Omega (\nabla N_A)\cdot(\kappa \nabla N_B T_B) \;\mathrm{d}\Omega \right) \end{aligned} \tag{2.36b}$$

In order for (2.36) to be true for all w_A $(A = 1, 2, ..., n)$, each $G_A = 0$ for all A. Thus, for each A the following equation holds

$$\begin{aligned}&\int_\Omega N_A N_B \dot{T}_B \;\mathrm{d}\Omega + \int_\Omega (\nabla N_A)\cdot(\kappa \nabla N_B T_B) \;\mathrm{d}\Omega \\ &\quad= \int_\Omega N_A \,\Psi \;\mathrm{d}\Omega - \int_{\Gamma_N} N_A \, q \;\mathrm{d}\Gamma \\ &\qquad - \int_\Omega N_A N_{B_D} \dot{T_{B_D}} \;\mathrm{d}\Omega - \int_\Omega (\nabla N_A)\cdot(\kappa \nabla N_{B_D} T_{B_D}) \;\mathrm{d}\Omega \end{aligned} \tag{2.37}$$

This can be written in a matrix form: The terms multiply T_B and $\dot{T}_B$ are combined and form the so-called conductivity matrix $\mathbf{C}$ and heat capacity or "mass" matrix $\mathbf{M}$, resp. All terms on the RHS of (2.37) are known and combined in a vector F.

$$
\begin{aligned}
[M]_{AB} &:= \int_\Omega N_A N_B \,\mathrm{d}\Omega && (2.38)\\
[C]_{AB} &:= \int_\Omega (\nabla N_A)\cdot(\kappa \nabla N_B)\,\mathrm{d}\Omega && (2.39)\\
(F)_A &:= \int_\Omega N_A\,\Psi\,\mathrm{d}\Omega \\
&\quad -\int_{\Gamma_N} N_A\,q\,\mathrm{d}\Gamma \\
&\quad -\int_\Omega N_A\,N_{B_D}\,\dot{T_{B_D}}\,\mathrm{d}\Omega \\
&\quad -\int_\Omega (\nabla N_A)\cdot(\kappa\nabla N_{B_D}\,T_{B_D})\,\mathrm{d}\Omega && (2.40)
\end{aligned}
$$

The Galerkin finite element approximation can then be written as a matrix equation, which is equivalent to (2.37)

$$\mathbf{M}\,\dot{T} + \mathbf{C}\,T = F \tag{2.41}$$

The second term in F describes the Neumann boundary condition, that is, the heat flux q in or out through the domain boundary. In geodynamic problems one often requires insulating boundary conditions, for example on walls where symmetry-plane velocity boundary conditions are imposed. Insulating boundary conditions are enforced, if the second term is zero (i.e. no heat flux in or out of the domain).

As explained in Section 2.1, the shape functions N have zero values everywhere in the numerical domain except in the neighborhood of their associated node. Thus, they have non-zero contributions to the integrals in (2.38)-(2.40) only within the elements connecting to conjoint nodes. An efficient way to evaluate the above integrals is therefore to perform the integrations over each element and then accumulate the results at the nodes connected to the element. In other words, an integral of shape function N_A over the entire domain Ω can be evaluated by integrating N_A over the few elements that are connected to node A and summing up their contributions at node A.

In this so-called assembly process, the element capacity matrix $\mathbf{M^e}$, element conductivity matrix $\mathbf{C^e}$, and element RHS-vector F^e are calculated for each element using numerical integration. A mapping from local (element) node numbers to global node numbers is required, in order to assemble the element tensors into the global counterparts. This mapping information is stored in the so-called *connectivity matrix*, which provides the global node number for each node in each element. In the finite element codes developed in this thesis, **M**, **C** and F in (2.41) are calculated by

$$\mathbf{M} = \sum_{e=1}^{nel} \mathbf{M^e} \quad , \quad \mathbf{C} = \sum_{e=1}^{nel} \mathbf{C^e} \quad , \quad F = \sum_{e=1}^{nel} F^e \quad \text{, where} \tag{2.42a}$$

$$[M]^e_{ab} = \int_{\Omega^e} N_a N_b \, \mathrm{d}\Omega^e \tag{2.42b}$$

$$[C]^e_{ab} = \int_{\Omega^e} (\nabla N_a) \cdot (\kappa \nabla N_b) \, \mathrm{d}\Omega^e \tag{2.42c}$$

$$(F)^e_a = \int_{\Omega^e} N_a \, \Psi \, \mathrm{d}\Omega^e + \int_{\Gamma^e_f} N_a \, q \, \mathrm{d}\Gamma - \sum_{\substack{b=1 \\ b \in B_D}}^{npe} \left([C]^e_{ab} \, T_b + [M]^e_{ab} \, \dot{T}_b \right) \tag{2.42d}$$

The lowercase letters a or b denote node numbers within an element and range from 1 to npe, with $npe = 6$ for quadratic-order Taylor-Hood elements in 2-D (triangles) and $npe = 10$ in 3-D (tetrahedra); see pp. 167 in Hughes (2000). The total number of elements in the mesh is denoted nel and the sum symbol is used to indicate the assembly process involving the connectivity matrix.

Equation (2.41) states a time dependent problem, whereas the above Galerkin method solely represents a spatial discretization. In order to step forward in time, a temporal discretization is required as well. Here the so-called alpha-family of methods is used, where the parameter α can be understood as the position between the current point in time t_n and the future time t_{n+1}, towards which we aim to step. Following Hughes (2000, p. 460), the time approximation can be formulated as

$$\mathbf{M} \, \dot{T}_{n+1} + \mathbf{C} \, T_{n+1} = F_{n+1} \tag{2.43}$$

$$T_{n+1} = T_n + \Delta t \, \dot{T}_{n+\alpha} \tag{2.44}$$

$$\dot{T}_{n+\alpha} = (1-\alpha)\dot{T}_n + \alpha \dot{T}_{n+1} \tag{2.45}$$

Subscript n and $n+1$ indicate values at time t_n and $t_{n+1} = t_n + \Delta t$, resp., where Δt is the step size as we march forward in time. Applying this scheme to (2.41) leads to a new matrix equation

$$(\mathbf{M} + \alpha \, \Delta t \, \mathbf{C}) \, T_{n+1} = (M - (1-\alpha)\Delta t \, \mathbf{C}) \, T_n + \Delta t (\alpha F_{n+1} + (1-\alpha) F_n) \tag{2.46}$$

$$\hat{\mathbf{C}} \qquad T = \tilde{F} \tag{2.47}$$

The time approximation can be done either on the element level prior to assembly (2.42) or, as suggested here, on the global matrices after the assembly. The latter has the advantage of accelerating the calculation of the element matrices. All terms on the RHS in (2.46) are known and combined in $\tilde{F}$. This matrix equation is solved iteratively using a preconditioned conjugate gradient (PCG) algorithm (see Section 2.4.1).

The parameter α controls the accuracy and stability of the time stepping process: $\alpha = 0$ leads to an explicit scheme (also called forward Euler method), in which the time step only depends on the values of T and $\dot{T}$ at $t = t_n$. The stability of this scheme depends on the size of the time step and the spatial resolution of the mesh. $\alpha = 1$ defines an implicit scheme (backward Euler method) that only depends on T and $\dot{T}$ at $t = t_{n+1}$. It is unconditionally stable. The best accuracy (in theory) is achieved for values of $\alpha = 0.5$ (Crank-Nicolson scheme). In the geodynamic problems studied in this thesis, alpha is varied between 0.5 and 1.

The time stepping method described by (2.43)-(2.45) is equivalent to a finite difference approach, since time-derivatives are approximated by discrete temporal differences. The time discretization can also be done using the FEM, which leads to the consistent Finite Element Galerkin formulation (e.g. Donea and Huerta, 2003, pp. 96). In the case of a uniform finite element time discretization, α naturally takes a value of 2/3.

2.2.3 Numerical implementation

The performance of the FEM depends partly on the element assembly, during which the integrals have to be evaluated and accumulated into the global matrices and the RHS vector. The numerical implementation of this essential part is briefly discussed next.

The evaluation of integrals over each element is done by numerical integration, which requires a loop over a certain number of integration points (Gaussian Quadrature; see for example Hughes (2000, pp. 141 & pp. 171) or Zienkiewicz and Taylor (1989, pp. 175)). Within each element, the value of the integrand has to be calculated at the integration points. If, for instance, the product of shape function N_A and temperature trial solution T^h needs to be integrated, the product $N \cdot T^h$ has to be evaluated at the integration points, multiplied by integration weights, and summed up. While the shape functions are mathematically defined so that the function value of N_A can be calculated analytically everywhere inside the elements, the trial solution T^h has to be interpolated at the integration points using the nodal values. This interpolation can be done with the help of shape functions, which is why they are also called interpolation functions.

In standard FEM, the required shape function values at the integration points as well as their spatial derivative (required for the viscous flow problem discussed below), are pre-calculated for an idealized undeformed master element. These values can be used for all elements in the mesh and do not need to be evaluated over and over again for each element. The spatial derivatives of the shape functions, however, depend on the deformation of an element with respect to the master element on which the derivatives have been pre-calculated. Thus, the shape function derivatives require a mapping from the master element into each element in the mesh. This mapping represents a scaling into each of the n_{dim} spatial dimensions and is expressed by the so-called Jacobi matrix $\mathbf{J}$, which has dimensions [n_{dim} x n_{dim}]. In fact, the inverse of $\mathbf{J}$ is required for every element in the mesh.

The assembly procedure in standard FEM leads to a loop over all elements, within which a loop over all integration points is located (i.e. nested loop). Within the innermost loop, relatively small matrix operations take place: (1) calculation and inversion of $\mathbf{J}$, (2) mapping of shape function derivatives into the element coordinates using $\mathbf{J^{-1}}$, (3) multiplications involving shape functions and their derivatives, (4) construction of the element matrices $\mathbf{M^e}$ and $\mathbf{C^e}$ (both [6x6] in 2-D and [10x10] in 3-D, resp.), and (5) calculation of the RHS-vector. In compiler based languages like *C* or *Fortran*, a good compiler "unrolls" these loops to achieve larger mathematical operations at a time. This leads to an improved performance of the numerical code because the data transfer between memory and CPU cache is more efficient (fewer large data packages are transferred instead of many small ones). Furthermore, mathematical libraries such as "BLAS" (Basic Linear Algebra Subprograms, see *www.netlib.org/blas*) can be used for all vector and matrix operations. These libraries perform well for large vector-matrix operations but rather poorly for small calculations (partly because each call of the library is associated with an overhead).

All codes developed in this thesis are written in MATLAB (*www.mathworks.com*), so that this vectorization has to be done in the source code itself. This requires restructuring of the numerical code to gain larger vector-matrix operations, which greatly improves its performance. I follow the concept suggested by Dabrowski et al. (2008) and assemble blocks of elements at once, which requires a restructuring of the element assembly procedure. The element block size can be varied to find the best performance for a given hardware. Without this block-wise assembly, the time for assembling the global matrix equation of a large numerical problem easily exceeds the time for solving the matrix equation itself. The slowdown is especially dramatic in 2-D, because here meshes usually contain more elements than in 3-D, and element matrices are smaller. However, using the block element assembly overcomes this issue. Another, smaller, speed-up results from calculating $\mathbf{J}$ only once in each element, that is, only in the first cycle of the loop over integration points. In most FEM examples, $\mathbf{J}$ is calculated for each integration point, but it is actually constant in triangular and tetrahedral elements as long as the elements have straight edges.

The resulting global matrices are very large but sparse, that is, most of their entries are zero. Only nodes that are connected by elements can have non-zero entries in the respective rows and columns. The Conjugate Gradient method (CG; algorithm (2.124) on page 49) with a Jacobi preconditioner (diagonal scaling) is used to solve the matrix equation in 2-D and 3-D. This simple preconditioner proved to be sufficient, because the constant thermal diffusivity makes the conductivity matrix in (2.46) well conditioned.

More advanced preconditioners result in fewer CG iterations, but also require more computational effort than the Jacobi preconditioner. Diagonal preconditioning also has the advantage of simplifying the parallelization of this part of the convection code (discussed in Section 2.5.2).

2.2.4 Discussion of alternative numerical implementations

As described above, we use operator splitting and solve for advection and diffusion of temperature separately. The diffusion part is formulated using the Galerkin FEM, while the semi-Lagrange method with a highly accurate interpolation scheme performs the advection. The reasons for not including the advection term in the Galerkin formulation will be briefly summarized next. For more details see Donea and Huerta (2003, pp. 33).

Using the FEM to solve for the advection of a temperature field requires the derivation of the variational form of the advection term on the left-hand side of equation (2.28). The Galerkin approximation of the variational form and the resulting element advection matrix $\mathbf{D^e}$ are:

$$\text{variational form of } \vec{v}\,\nabla T: \qquad \int_\Omega w\,\vec{v}\,\nabla T\,\mathrm{d}\Omega \tag{2.48}$$

$$\text{Galerkin approximation}: \qquad \int_\Omega \sum_A N_A w_A\,\vec{v}\,\nabla \sum_B N_B\,T_B\,\mathrm{d}\Omega \tag{2.49}$$

$$\text{element advection matrix}: \qquad [D]^e_{ab} = \int_{\Omega^e} N_a\,\vec{v}\,\nabla N_b\,\mathrm{d}\Omega^e \tag{2.50}$$

$$\text{global advection matrix}: \qquad \mathbf{D} = \sum_{e=1}^{nel} \mathbf{D^e} \tag{2.51}$$

The first drawback of this approach is that the advection term results in an non-symmetric matrix $\mathbf{D}$ that needs to be added to the conductivity matrix $\mathbf{C}$ in equation (2.41) in order to solve for advection and diffusion simultaneously: $\mathbf{M}\dot{T} + (\mathbf{C} + \mathbf{D})T = F$. The asymmetry disqualifies fast and memory efficient iterative solvers like the Conjugate Gradient algorithm for solving the matrix equation, as they rely on the symmetry of the matrix. Instead, solvers that are more costly in both CPU time and memory usage like GMRES (Generalized Minimal RESidual method, (Saad and Schultz, 1986)) or Bi-CGSTAB (biconjugate gradient stabilized method, (van der Vorst, 1992)) have to be used for this asymmetric problem.

The second and much more critical drawback of the advection term in the FEM is a lack of stability when advection dominates diffusion. Unfortunately, this is the case in most regions of the Earth's mantle due to the low thermal conductivity of the rocks. The instabilities appear as oscillations in the temperature field (see, for instance the stability analysis in Donea and Huerta (2003, pp. 50)) that can easily lead to artifacts in the melting formulation, but also feed back into the flow field if viscosity is strongly temperature dependent. To overcome this issue, stabilization terms have to be added by modifying the weighting functions, that is, N_a in equation (2.50). The functions constructing T^h are thus no longer those that construct w^h (i.e. N_A in Eq. (2.33) $\neq N_B$ in Eq. (2.34)). The stabilization terms have the effect of an additional balancing diffusion but only in the direction of flow, so that the method is known as Streamline-Upwind-Petrov-Galerkin (SUPG). While the SUPG-method effectively reduces the oscillations in the temperature field, it leads to a ”too diffusive” solution with a smearing-out of steep temperature gradients in the upwind direction.

Transport terms in numerical formulations on Eulerian grids generally cause problems and also finite difference methods (FDM) require stabilization efforts to avoid oscillations. Probably the most accurate stabilization technique is the MP-DATA algorithm for the FDM (Smolarkiewicz, 1984). In Fig. 2.2, two step-like temperature anomalies are advected around a 180°corner in a steady velocity field using the above methods. Note that the examples show a pure advection problem without solving for any diffusion.

The full Lagrange formulation (Fig. 2.2b) represents a very accurate alternative to the semi-Lagrange method: If the FEM nodes are defined in a Lagrangian coordinate system that follows the characteristics of the flow field, the advection-diffusion process is approximated by solving for diffusion only using the Galerkin FEM and then advecting the resulting field by moving the nodes along the characteristics.

Although the implementation of the Lagrange method is straightforward, problems emerge for flow fields that have strong velocity gradients (i.e. shearing) or eddies. In these regions, the finite element mesh quickly gets distorted to a degree that affects the quality of the mesh. Potential risks of bad-quality meshes include an emerging linear dependence of the equations, which can lead to a singular matrix and no convergence of the numerical solver in the worst case. As an example, consider a triangular element that is flattened so that one node is located near the line connecting the two other nodes. Round-off in the numerical solution algorithm can make the equation associated with the middle node become a linear combination of the two equations associated with the enclosing nodes. In this case the system of linearly independent equation becomes smaller than the number of unknowns and the solution algorithm fails.

To overcome this problem, the numerical mesh has to be recovered (re-meshed) as soon as the deformation becomes too strong. Unfortunately, nodes in elements that are located in domain corners are limited in their freedom to move. In the presence of a corner flow, these elements ”collapse” after very few time steps and would require a re-meshing. Every re-meshing, however, goes along with an interpolation of the advected field to the

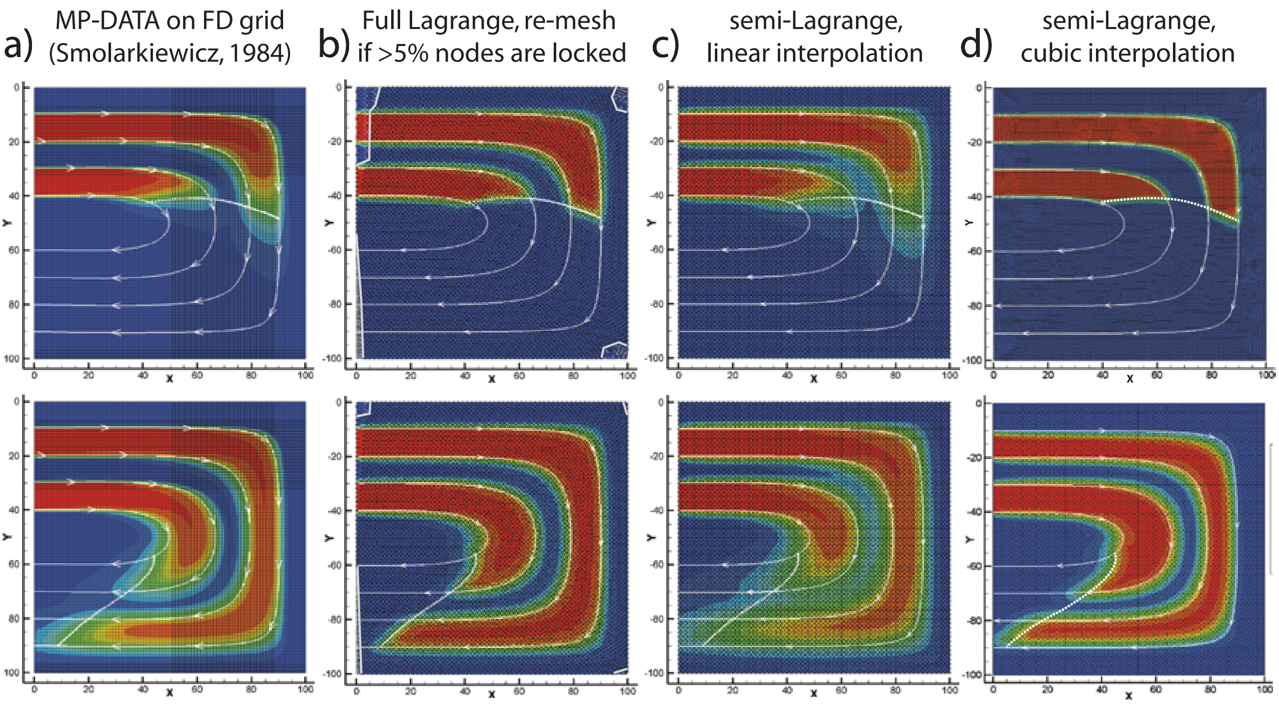


Figure 2.2: *Pure advection (no diffusion!) of two step-shaped temperature anomalies within a steady state flow field (indicated by the white stream lines). In each figure, the dotted white line shows how far the field has been advected; two snap shots are shown for each method. The temperature difference between hot (red) and cold back ground (blue) is 100 °C . (a) Eulerian grid, MP-DATA algorithm (Smolarkiewicz, 1984), implemented in a FDM code. (b) full-Lagrange FEM, i.e. nodes move within the flow field; nodes are locked when adjacent elements become too distorted (regions enclosed by gray lines); re-meshing if more than 5% of all nodes are locked. (c) semi-Lagrange method with linear interpolation. (d) semi-Lagrange method with bi-cubic interpolation (Shi and Phipps Morgan, 2010), this method is used in all finite element codes developed within this thesis.*

new location of the nodes and can introduce numerical diffusion (the degree of which depending on the quality of the interpolation scheme). In order to allow more time steps until a complete re-meshing is required, nodes connected to elements that have become too distorted are ”locked”. For these nodes, I switch to an Eulerian frame and use the semi-Lagrange method. Regions with locked nodes are encompassed by gray lines in the Lagrange example in Fig. 2.2b. A global re-meshing is performed, if a certain percentage of the nodes is locked (usually 5%).

In spite of the excellent results of the Lagrange method, I decided to use the semi-Lagrange method for two reasons: (1) In typical geodynamic problem geometries, the ”corner elements” are located within a region of interest (for example, the axis of a mid-ocean ridge). Thus, the semi-Lagrange method would be used here most of the time anyways. The same problem exists for domain boundaries that are open for in- and out flow. These are frequently used in favor of symmetry planes to minimize boundary effects. (2) The multigrid method used to precondition the viscous flow solver (see Section 2.5) has a much better performance if the fine mesh is nested within the coarser mesh. That is,

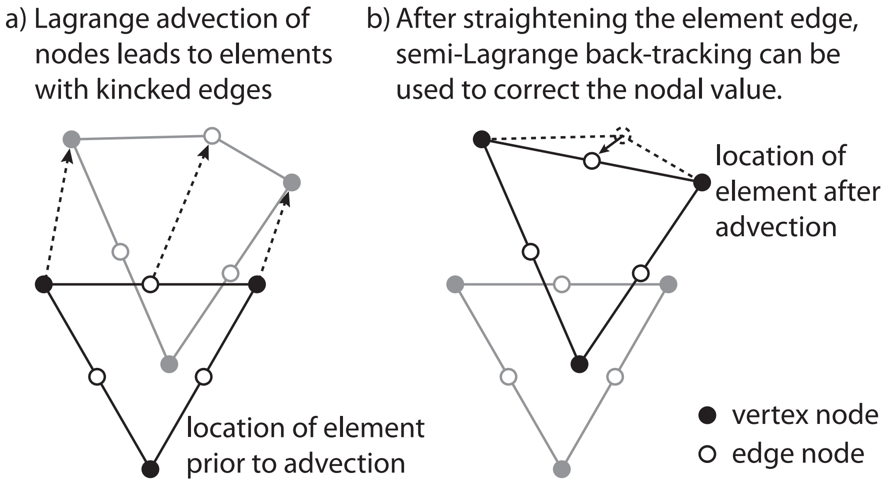

a) Lagrange advection of nodes leads to elements with kincked edges
b) After straightening the element edge, semi-Lagrange back-tracking can be used to correct the nodal value.
location of element after advection
location of element prior to advection
● vertex node
○ edge node

Figure 2.3: *Schematic advection and distortion of a triangular element in the Lagrangian frame. Numerical algorithms such as point-search routines for triangular meshes or multigrid (see Section 2.4.2) perform best if elements keep their straight edges. After the Lagrangian advection step (a), semi-Lagrange back tracking can be used to interpolate the edge node value at the location central between the vertices. Element distortion thus only results from the motion of the vertex nodes. This method has been tested but is not implemented in the 2-D and 3-D codes developed in this thesis; the semi-Lagrange method with cubic interpolation is used instead.*

patches of elements on the fine mesh should lie within a single coarser mesh element. This requirement strongly limits the possible displacements of nodes: Edge nodes, for instance, must always be located on the line connecting the two vertex nodes. It is possible to force edge nodes to always be in this centered position, and correct their variable values accordingly (Fig. 2.3). However, this again requires the semi-Lagrange method with a back tracking vector that describes the difference between the characteristics at the edge node and it's actual displacement. This could be a promising way, and more tests should be conducted in the future.

However, at present, the greatest solution difficulties reside in the viscous flow subproblem, hence this is where I devoted the greatest part of my thesis work to develop, test, and implement improved methods for solving variable viscosity Stokes flow. This is the subject of the next section.

2.3 Viscous flow I: finite element formulation

2.3.1 Introduction

This introduction to the FEM formulation and solution of the viscous flow problem is based on Hughes (2000) and Donea and Huerta (2003) and the reader is encouraged to consult these books for more details on elasticity, viscous flow, and the FEM solution of these problem types in general. As in the previous section, I will use the compact notation for derivatives ($u_{i,j} = \partial u_i / \partial x_j$).

The solution for a given viscous flow problem are a velocity and a pressure field. In the case of slowly creeping flow (Stokes flow), a steady state solution is derived. That is, the flow field only depends on viscous stresses, buoyancy forces, and boundary conditions but is independent of time. In other words, a new flow field does not depend on previous flow fields except for the advection history of density and viscosity fields. This approximation in the Stokes flow is justified, if the inertia of the material in motion can be neglected, since the time dependence solely results from inertial effects. In case of mantle flow, the moving masses are large (domain sizes of several 100 km extension are considered, densities range from 3,000–4,000 kg/m^3), but their speed is extremely slow: a ”fast” moving plate covers a distance of about 8 cm per year, which translates into $2.5 \cdot 10^{-9}$ m/s. The kinematic energy of a 2000 by 2000 km large portion of this plate (assumed to be 100 km thick and to have a density of 3,000 kg/m^3) is comparably to that of a 1000 kg car at a speed of 10 km/h (the energy of both is about 3.86 kJ). In this regard the Stokes approximation is spectacularly good for mantle flow. Essential for the mathematical description of Stokes flow are velocity gradients that control viscous stresses, which is where the mathematical description will begin.

2.3.2 Mathematical formulation

The physical description of viscous flow is based on the conservation of momentum, that is, body forces and viscous forces have to be in equilibrium. Viscous forces, on the other hand, are related to the velocity field that we aim solving for. Of particular importance are velocity gradients,

$$\begin{aligned} \nabla v &= v_{i,j} \qquad (2.52) \\ &= \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} \end{bmatrix} \text{(in 2-D)} \end{aligned}$$

The 2nd-order tensor (2.52) can be decomposed into its symmetric part that describes the strain rate in the fluid, and its non-symmetric part that describes the vorticity or rate of rotation in the fluid.

$$v_{i,j} = \frac{1}{2}\left[v_{i,j} + v_{j,i}\right] \qquad \text{(symmetric part; referred to as} = v_{(i,j)} \text{ below)} \qquad (2.53)$$

$$+\frac{1}{2}\left[v_{i,j} - v_{j,i}\right] \qquad \text{(non-symmetric or skew-symmetric part)} \qquad (2.54)$$

This decomposition is useful, because only the symmetric part is required for the definition of the strain rate tensor. The rotation of a fluid does not induce any strain rates so that this part of the velocity field can be neglected. Note, however, that in transient problems (physically described by the Navier-Stokes equations) this part cannot be ignored, due to the momentum associated with "rotating" part of the fluid. The strain rate tensor is defined as

$$\dot{\epsilon}_{ij} = v_{(i,j)} = \frac{v_{i,j} + v_{j,i}}{2} \qquad \text{(strain rate tensor)} \qquad (2.55)$$

$$= \begin{bmatrix} v_{x,x} & \frac{v_{x,y}+v_{y,x}}{2} \\ \frac{v_{y,x}+v_{x,y}}{2} & v_{y,y} \end{bmatrix} \qquad \text{(in 2-D)} \qquad (2.56)$$

In a fluid in motion, non-zero strain rates lead to viscous stresses that act to minimize differential motion and, thus, often decelerate the fluid, i.e. they counteract the forces driving the flow (such as buoyancy forces). It is convenient to decompose the viscous stresses into two parts: (1) an isotropic part σ^n_{ij} that describes the pressure acting to compress the fluid, and (2) deviatoric stresses σ^d_{ij} that act to shear and deform (squeeze) the fluid but preserve its volume.

$$\sigma^n_{ij} = -p\delta_{ij} \qquad \text{(isotropic stress tensor)} \qquad (2.57)$$

$$\text{where} \quad p = -\frac{1}{3}\sigma_{ii} \qquad \text{(pressure in a fluid in motion)} \qquad (2.58)$$

$$\sigma^d_{ij} = C_{ijkl}\,\dot{\epsilon}_{kl} \qquad \text{(deviatoric stress tensor)} \qquad (2.59)$$

$$\sigma_{ij} = \sigma^n_{ij} + \sigma^d_{ij} \qquad \text{(total stress tensor)} \qquad (2.60)$$

The isotropic stress (2.57) is a function of pressure p, which is defined by (2.58) at any point within a moving fluid. If the fluid is at rest, (2.58) is equal to the static pressure resulting from the mass of the overburden (i.e. lithostatic pressure). The deviatoric stresses (2.59) include all remaining, non-isotropic stresses that are related to the strain rates by a 4-th rank tensor C_{ijkl} that describes the material properties (e.g. anisotropy). The total stresses are the sum of isotropic and deviatoric stresses. For a compressible isotropic fluid, the constitutive tensor reduces to a simpler relation because of symmetries in its components:

$$\sigma^d_{ij} = \eta 2\dot{\epsilon}_{ij} + \lambda v_{k,k}\delta_{ij} \qquad (2.61)$$

The first term in (2.61) describes the shear stresses resulting from strain rates, whereas the second term accounts for "squeezing" due to net normal stresses (the part of the normal stresses that is equal in magnitude is included in the isotropic stress tensor). The dynamic viscosity η is the fluid's resistance to shearing and the bulk viscosity λ is the fluid's resistance to compression (also called 2nd coefficient of viscosity).

The compressibility of the mantle rocks is often neglected in numerical models, especially in those that cover a limited depth (pressure) range of the Earth rather than the entire upper and lower mantle. In this case, the mantle may be described as an incompressible fluid, which is mathematically expressed by

$$\nabla \cdot v = v_{i,i} = 0 \qquad \text{(incompressibility constraint)} \qquad (2.62)$$

Eq. (2.62) claims that the inflow into an infinitesimal volume is equal to the flow out of this volume. This can only be true if volume is preserved, i.e. the fluid is incompressible. $v_{i,i}$ is sometimes referred to as volumetric strain rate ϵ_v.

When substituting (2.62) into (2.61), the 2nd term becomes zero and the dynamic viscosity η remains as the only scaling factor between the strain rate and the deviatoric stresses. The stress-strain rate relation for an incompressible isotropic fluid

$$\sigma_{ij} = -p\delta_{ij} + \eta 2\dot{\epsilon}_{ij} \qquad \text{(incompressible isotropic stress tensor)} \qquad (2.63)$$

describes the fluid's motion under applied forces. It is frequently assumed that mantle rocks behave as a Newtonian fluid, that is, the viscosity η depends on neither stresses nor strain rate (e.g. Schubert et al., 2001). In this case, η is not a function of $\dot{\epsilon}$, thus independent of the unknown velocity field. In this case (2.63) describes a linear stress-strain rate relation. However, viscosity of mantle rocks is dependents on composition and temperature of the mantle, which makes η a function that strongly varies in space (usually several orders of magnitude over the numerical domain). These viscosity variations are the crux of the matter for the numerical solution process.

The most important contribution to the forces driving mantle flow are buoyancy forces f resulting from density variations with respect to a reference density ρ_0.

$$f = (\rho - \rho_0)g \qquad \text{(buoyancy force)} \qquad (2.64)$$

With the above definitions and relations, a Stokes flow boundary value problem in n_{dim} dimensions (indicated by indices $i, j = 1, ..., n_{dim}$) may be defined as follows: Given the buoyancy forces f_i, Dirichlet (velocity) boundary conditions v_{Di}, Neumann (traction) boundary conditions t_i, and a constitutive law (2.63), determine the velocity flow field v and the pressure p such that

$$\begin{aligned}
\sigma_{ij,i} + f_i &= 0 && \text{(force equilibrium)} && (2.65a)\\
v_{i,i} &= 0 && \text{(incompressibility constraint)} && (2.65b)\\
v_i &= v_{Di} && \text{on } \Gamma_{Di} \text{ (Dirichlet boundary condition)} && (2.65c)\\
\sigma_{ij} n_j &= t_i && \text{on } \Gamma_{Ni} \text{ (Neumann boundary condition)} && (2.65d)
\end{aligned}$$

The next section will deal with the numerical approximation and solution of (2.65).

2.3.3 Numerical formulation

The FEM requires the conversion of the strong or differential form of the Stokes boundary value problem (2.65) into an equivalent weak or variational formulation. As opposed to the thermal diffusion problem discussed earlier, two unknowns have to be determined here: a velocity field v and a pressure field p. Furthermore, the former is a vector field with n_{dim} components per node. Thus, two sets of trial solutions and weighting functions have to be defined:

$$v \in \mathscr{U} \qquad \text{(velocity trial solutions)} \tag{2.66}$$

$$u, w \in \mathscr{W} \qquad \text{(velocity weighting functions)} \tag{2.67}$$

$$p \in \mathscr{P} \qquad \text{(pressure trial solutions)} \tag{2.68}$$

$$q \in \mathscr{P} \qquad \text{(pressure weighting functions)} \tag{2.69}$$

The weighting functions have the property to vanish where Dirichlet boundary conditions are imposed. However, because no explicit boundary conditions are imposed on the pressure field the functions p and q are actually from the same functional space $\mathscr{P}$. The variational form of (2.65) is achieved by the following operations

$$\int_\Omega w\, \sigma_{ij,i} + \int_\Omega w f_i \, \mathrm{d}\Omega = 0 \tag{2.70a}$$

$$\int_\Omega w_{,i}\, \sigma_{ij,i} \, \mathrm{d}\Omega - \int_\Omega w_{,i} \sigma_{ij} \, \mathrm{d}\Omega + \int_\Omega w f_i \, \mathrm{d}\Omega = 0 \tag{2.70b}$$

$$\int_\Gamma w\, \vec{n} \sigma_{ij} d\Gamma - \int_\Omega w_{,i} \sigma_{ij} \, \mathrm{d}\Omega + \int_\Omega w f_i \, \mathrm{d}\Omega = 0 \tag{2.70c}$$

The steps are multiplying (2.65a) by w and integrating over the domain Ω →(2.70a), partial integration of the stress term →(2.70b), and applying the divergence theorem →(2.70c). Re-ordering, considering $w = 0$ on Γ_D, and substituting the Neumann boundary condition (2.65d) leads to the variational form of (2.65a)

$$\int_\Omega w_{,i}\, \sigma_{ij} \, \mathrm{d}\Omega = \int_\Omega w f_i \, \mathrm{d}\Omega + \int_{\Gamma_N} w\, t_i \, d\Gamma \tag{2.71}$$

The variational form of (2.65b) is

$$\int_\Omega q\, v_{i,i} \, \mathrm{d}\Omega = 0 \tag{2.72}$$

Adding both, (2.71) and (2.72), gives the variational form of Stokes problem in (2.65)

$$\int_\Omega w_{,i}\, \sigma_{ij} \, \mathrm{d}\Omega - \int_\Omega q\, v_{i,i} \, \mathrm{d}\Omega = \int_\Omega w f_i \, \mathrm{d}\Omega + \int_{\Gamma_N} w\, t_i \, d\Gamma \tag{2.73}$$

The first term in (2.73) includes the stress tensor σ_{ij} and consequently depends on both pressure (isotropic stresses) and velocity field (deviatoric stresses, which depend on the symmetric part of ∇u). Decomposition of this term into its pressure and velocity contributions yields

$$\int_{\Omega} w_{,i}\, \sigma^{d}_{ij}\ \mathrm{d}\Omega - \int_{\Omega} w_{,i}\, p\ \mathrm{d}\Omega - \int_{\Omega} q\, v_{i,i}\ \mathrm{d}\Omega = \int_{\Omega} w f_i\ \mathrm{d}\Omega + \int_{\Gamma_N} w\, t_i\, d\Gamma \tag{2.74}$$

It is convenient to write the strain rate tensor $\dot{\epsilon}_{ij}$ (a n_{dim} x n_{dim} matrix) in a vector form by taking advantage of its symmetry. This so-called Voigt notation reduces the order of the tensor and leads to the strain rate vector

$$\dot{\epsilon} = \begin{pmatrix} v_{i,i} \\ v_{j,j} \\ v_{i,j} + v_{j,i} \end{pmatrix} \tag{2.75}$$

This allows to rewrite (2.74) in terms of the strain-rate vector

$$\int_{\Omega} \dot{\epsilon}(w)^{T}\, \mathbf{C}_{\eta}\, \dot{\epsilon}(v)\, \mathrm{d}\Omega - \int_{\Omega} w_{,i}\, p\ \mathrm{d}\Omega - \int_{\Omega} q\, v_{i,i}\ \mathrm{d}\Omega = \int_{\Omega} w f_i\ \mathrm{d}\Omega + \int_{\Gamma_N} w\, t_i\, d\Gamma \tag{2.76}$$

where $\dot{\epsilon}^{T}$ means transpose of $\dot{\epsilon}$ and $\mathbf{C}_{\eta}$ is the reshaped constitutive matrix ($\mathbf{C_{ijkl}}$) relating the stresses and strain rates.

Since we use the Galerkin FEM, the trial solutions v and p are constructed with the help of weighting functions (see discussion in 2.1). The superscript "h" is used to indicate these functions as discretized counterparts of the ones defined in (2.66)-(2.69). Because we solve for $i = 1, ..., n_{dim}$ components of the velocity field at each node A, a subscript i appears in the definition of the velocity weighting functions and trial solutions. However, there is no summation on i. Velocity weighting functions are defined as

$$w_i^h = \sum_{B \notin B_{Di}} N_B c_{iB} \qquad \text{(velocity weighting functions)} \tag{2.77}$$

The velocity trial solutions v_i^h are constructed using weighting functions u_i^h and the functions u_{Di}^h that satisfy the Dirichlet boundary conditions.

$$v_i^h = u_i^h + u_{Di}^h \qquad \text{(velocity trial solutions)} \tag{2.78a}$$

where

$$u_i^h = \sum_{A \notin A_{Di}} N_A u_{iA} \tag{2.78b}$$

$$u_{Di}^h = \sum_{A \in A_{Di}} N_A u_{Di}(x_A) \tag{2.78c}$$

It is useful to introduce unit vectors e_i that define the canonical basis of the $\mathbb{R}^{n_{dim}}$, that is, for the $\mathbb{R}^2$: $e_1 = \left\{ \begin{smallmatrix} 1 \\ 0 \end{smallmatrix} \right\}$ and $e_2 = \left\{ \begin{smallmatrix} 0 \\ 1 \end{smallmatrix} \right\}$. (2.78b) and (2.77) may then be written in a vector notation

$$u^h = \sum_{i=1}^{n_{dim}} u_i^h e_i = \sum_{i=1}^{n_{dim}} \sum_{A \notin A_{Di}} N_A u_{iA} e_i \qquad \text{, vector version of (2.78b)} \tag{2.79}$$

$$u_D^h = \sum_{i=1}^{n_{dim}} u_{Di}^h e_i = \sum_{i=1}^{n_{dim}} \sum_{A \in A_{Di}} N_A u_{Di}(x_A) e_i \qquad \text{, vector version of (2.78c)} \tag{2.80}$$

$$w^h = \sum_{i=1}^{n_{dim}} w_i^h e_i = \sum_{i=1}^{n_{dim}} \sum_{B \notin B_{Di}} N_B w_{iB} e_i \qquad \text{, vector version of (2.77)} \tag{2.81}$$

As mentioned above, the trial solutions and weighting functions for pressure are of the same functional space, due to the lack of boundary conditions on pressure. The pressure shape functions $\hat{N}$ usually differ from those defined to approximate the velocity problem. Pressure nodes are denoted $\hat{A}$ and $\hat{B}$.

$$p^h = \sum_{\hat{A}} p_{\hat{A}} \hat{N}_{\hat{A}} \qquad \text{(pressure trial solutions)} \tag{2.82}$$

$$q^h = \sum_{\hat{B}} d_{\hat{B}} \hat{N}_{\hat{B}} \qquad \text{(pressure weighting functions)} \tag{2.83}$$

The next steps are analogous to the operations that led from (2.35) to (2.36) in the thermal diffusion problem. Substituting the Galerkin approximations (2.79)-(2.83) into (2.76), changing the order of summation and integration, and collecting terms gives

$$0 = \sum_{A \notin A_D} w_A Q_{iA} + \sum_{\hat{A}} q_{\hat{A}} \hat{Q}_{\hat{A}} \tag{2.84a}$$

$$\begin{aligned} Q_{iA} = & \sum_{j=1}^{n_{dim}} \left\{ \sum_{B \notin B_{Di}} \int_\Omega \dot{\epsilon} \left(N_A e_i\right)^T \mathbf{C}_\eta \, \dot{\epsilon} \left(N_B e_j\right) \mathrm{d}\Omega \; u_{jB} \right\} \\ & - \sum_{\hat{A}} \int_\Omega \nabla \left(N_A e_i\right) \; \hat{N}_{\hat{A}} \, \mathrm{d}\Omega \; p_{\hat{A}} \\ & - \int_\Omega N_A e_i \; f_i \, \mathrm{d}\Omega + \int_{\Gamma_N} N_A e_i \; t_i \, \mathrm{d}\Gamma_N \\ & + \sum_{j=1}^{n_{dim}} \left\{ \sum_{B \in B_{Di}} \int_\Omega \dot{\epsilon} \left(N_A e_i\right) \mathbf{C}_\eta \, \dot{\epsilon} \left(N_B e_j\right) \mathrm{d}\Omega \; u_{Di} \right\} \end{aligned} \tag{2.84b}$$

$$\begin{aligned} \hat{Q}_{\hat{A}} = & \sum_{i=1}^{n_{dim}} \left\{ \sum_{A \notin A_{Di}} \int_\Omega \hat{N}_{\hat{A}} \; \nabla \left(N_A e_i\right) \, \mathrm{d}\Omega u_{iA} \right\} \\ & - \sum_{i=1}^{n_{dim}} \left\{ \sum_{A \in A_{Di}} \int_\Omega \hat{N}_{\hat{A}} \; \nabla \left(N_A e_i\right) \, \mathrm{d}\Omega u_{Di} \right\} \end{aligned} \tag{2.84c}$$

(2.84b) represents a set of $n = n_{dim} \cdot nnod - \sum_{i=1}^{n_{dim}} n_{Di}$ linearly independent equations (n_{Di} is the number of Dirichlet boundary conditions in spatial direction i). The number of equations in (2.84c) is equal to the number of pressure nodes $nPnod$. In order for (2.84) to be true for any velocity weighting function w_A and any pressure weighting function q_A, every equation in both, (2.84b) and (2.84c), has to be equal to zero, i.e. $Q_{iA} = 0$ for all $A \notin A_D$ and all $i = 1, ..., n_{dim}$ as well as $\hat{Q}_{\hat{A}} = 0$ for all $\hat{A}$. Upon setting (2.84b) and (2.84c) equal to zero and moving all known terms to the RHS, (2.84) can be written as a matrix equation

$$\begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \tilde{\mathbf{G}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} F \\ H \end{pmatrix} \tag{2.85}$$

The sub-matrices and RHS-vector of the matrix equation are

$$[K]_{iAjB} := \int_\Omega \dot{\epsilon}(N_A e_i)^T \, \mathbf{C}_\eta \, \dot{\epsilon}(N_B e_j) \, \mathrm{d}\Omega \qquad \text{(stiffness matrix)} \tag{2.86}$$

$$[G]_{iA\hat{A}} := -\int_\Omega \nabla N_A e_i \, \hat{N}_{\hat{A}} \, \mathrm{d}\Omega \qquad \text{(gradient matrix)} \tag{2.87}$$

$$\left[\tilde{G}\right]_{iA\hat{A}} := -\int_\Omega \hat{N}_{\hat{A}} \, \nabla N_A e_i \, \mathrm{d}\Omega \qquad \text{(divergence matrix)} \tag{2.88}$$

$$\begin{aligned} (F)_{iA} := & \int_\Omega N_A e_i \, f_i \, \mathrm{d}\Omega \\ & + \int_{\Gamma_N} N_A e_i \, t_i \, \mathrm{d}\Gamma_N \\ & - \int_\Omega \dot{\epsilon}\left(N_A e_i\right) \mathbf{C}_\eta \, \dot{\epsilon}\left(N_B e_j\right) \, \mathrm{d}\Omega \, u_{Di} \end{aligned} \qquad \text{(force vector)} \tag{2.89}$$

$$(H)_{\hat{A}} := \int_\Omega \hat{N}_{\hat{A}} \, \nabla N_A e_i \, \mathrm{d}\Omega \, u_{Di} \qquad \text{(dilatation vector)} \tag{2.90}$$

As mentioned above, pressure trial solutions p are constructed from pressure weighting functions q and, because no pressure boundary conditions are imposed, both are of the same functional space $\mathscr{P}$. That means $q = p$, except for the coefficients $p_{\hat{A}}$ and $d_{\hat{B}}$ (see definitions in (2.82)-(2.83)). The coefficients, however, are not part of the matrices $\tilde{\mathbf{G}}$ and $\mathbf{G}$, resp., but only the pressure shape functions $\hat{N}$. Thus, $\tilde{\mathbf{G}} = \mathbf{G}^{\mathbf{T}}$ and

$$\mathbf{A} = \begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \tilde{\mathbf{G}} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{G}^{\mathbf{T}} & \mathbf{0} \end{bmatrix} \tag{2.91}$$

is a symmetric matrix. Mathematically, (2.85) represents a saddle point problem, whose properties are discussed comprehensively in the review article by Benzi et al. (2005). Usually the vector H is completely zero, except for boundary condition terms. If the entire domain boundary has prescribed velocities and no Neumann boundary condition is imposed, these velocities have to satisfy the $u_{i,i} = 0$ constraint — otherwise the Stokes flow problem has no solution. Before presenting some strategies for solving the above matrix equation, I will discuss the numerical implementation of the FEM. Here I will focus on the efficient evaluation of the integrals in the programming language *MATLAB*.

2.3.4 Numerical implementation

Element assembly

As discussed in the heat diffusion problem, the sub-matrices **K** and **G** as well as the force vector F are assembled on an element level. This is efficient, because the integrals are zero almost everywhere within Ω. For instance, the shape functions N_A for velocity or $\hat{N}_{\hat{B}}$ for pressure are non-zero only within elements that connect to velocity node A and pressure node $\hat{B}$, resp. Therefore, the global matrices and vectors are obtained from the assembly of element contributions.

$$\mathbf{K} = \sum_{e=1}^{nel} \mathbf{K^e} \quad , \quad \mathbf{G} = \sum_{e=1}^{nel} \mathbf{G^e} \quad , \quad F = \sum_{e=1}^{nel} F^e \tag{2.92}$$

A standard method to calculate the products of functions for the stiffness matrix is to re-order the spatial derivatives of the velocity shape functions and store them in a matrix **B**. The constitutive matrix $\mathbf{C}_\eta$ reduces to a 3x3 matrix (in 2-D) or a 6x6 matrix (in 3-D), resp. The following matrix multiplication then yields the integrand for the stiffness matrix (Hughes, 2000):

$$\dot{\epsilon}(N_a e_i)^T \, \mathbf{C}_{\eta ab} \, \dot{\epsilon}(N_b e_j) = \mathbf{B}_a^T \, \mathbf{D} \, \mathbf{B}_b \qquad \text{, where} \tag{2.93}$$

$$\mathbf{B_a} = \begin{bmatrix} N_{a,x} & 0 \\ 0 & N_{a,y} \\ N_{a,x} & N_{a,y} \end{bmatrix} \quad \text{and} \quad \mathbf{D} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.94}$$

For a 3-D example see Donea and Huerta (2003, p. 282). An alternative formulation for the above **D**

$$\mathbf{D^{(d)}} = \begin{bmatrix} \frac{4}{3} & -\frac{2}{3} & 0 \\ -\frac{2}{3} & \frac{4}{3} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.95}$$

is used if the dilatation is non-zero somewhere within the numerical domain, for example, if the mass transfer associated with melt extraction underneath a ridge and accumulation in the oceanic crust is considered. [1]

[1] Coupled mantle flow and melt migration are rarely modeled as a coupled system, because both happen on very different time scales. There are several orders of magnitude between the estimated

Table 2.1: *Properties of element and global stiffness matrices and code performance for both* $\mathbf{D}$*-matrices in* (2.95)*. Results for two test cases in 2-D are shown. Test problem: 1000x1000 km box, a 300 km thick weak layer on top with 250-fold lower viscosity than underlying material and 0.9% less density, forced plate motion at the top (30 km/Myr), symmetry plane b.c. everywhere else.*

test problem	Test problem 1		Test problem 2	
# velocity unknowns	254 k		32 k	
matrix version	$\mathbf{D}$	$\mathbf{D^d}$	$\mathbf{D}$	$\mathbf{D^d}$
# non-zeros in $\mathbf{K^e}$	96-106	108-118	96-106	108-118
# non-zeros in $\mathbf{K}$	4,278 k	4,271 k	528 k	530 k
number of pressure iterations	23	38	26	44
number of inv($\mathbf{K}$) iterations	450	662	477	712
time for solution (sec)	177	272	23	37

Both versions of $\mathbf{D}$ lead to global stiffness matrices $\mathbf{K}$ with similar sparse structure. However, there are few different connections between degrees of freedom in the resulting $\mathbf{K}$ in that the diagonal $\mathbf{D}$ leads to about 0.25% more non-zero entries in $\mathbf{K}$ (Tab. 2.1). Despite this slightly denser stiffness matrix, the convergence rate of the multigrid-preconditioned Conjugate Gradient solver (see 2.5.1) is significantly faster (a factor of 1.5 to 1.6) compared to runs where $\mathbf{D}^{(d)}$ is used. This has been identified for different mesh sizes, mesh structures and test problems. I therefore recommend using the diagonal $\mathbf{D}$ whenever a dilatation-free problem needs to be solved.

Another way to improve the performance of a FEM code in general without affecting the quality of the solution, is to ”clean-up” the stiffness matrix after or during the assembly. Round-off during the numerical integration leads to small ($< 10^{-12}$) values in components of the element matrices, where the correct value would be zero. These noise-values are assembled and enter the global stiffness matrix, where they are frequently used during the iterative solution process. Although these tiny values do not affect the quality of the final solution, they should be avoided for two reasons: 1) even though the values are tiny, they require space in the memory, because the matrix is stored as a sparse matrix, and 2) every calculation involving the tiny numbers is wasted computational time, since the results of these calculations have no effect on the solution process. To illustrate this ”noise”-problem in the finite element matrices ($\mathbf{G}$ is affected in the same way), I compared single solutions for the same test problem, in which I remove entries in the global stiffness matrix that are below a certain threshold. Of course, once the threshold is chosen too high, the physical meaning of the stiffness matrix is destroyed and the velocity/pressure

average migration speed of melts (on the order of about 40 m/yr (Connolly et al., 2009), in agreement with geochemical data, (Stracke et al., 2006; Rubin et al., 2005)) and the viscous flow of mantle (on the order of 0.04 m/yr). Therefore, melt extraction (i.e. transport from depth to the surface) is often assumed to happen instantaneously. The dilatation, if related to the melt production rate, allows to account for the mass transfer from melting region (sink) to the top of the domain (source), where the melts accumulate to form an oceanic crust. This extra mass transport has the effect to slightly enhance mantle flow into the melting region.

Table 2.2: *Reducing the number of non-zeros (nnz) in the stiffness matrix* $\mathbf{K}$ *by removing entries smaller then a certain threshold (first column). Very small entries (*$< 10^{-12}$*) result from round-off during the numerical quadrature and have no effect on the solution process. However, they require memory and computer time, if they are not removed during or after the assembly. The biggest improvement results form removing very small numbers (*$< 10^{-12}$*), where* $\mathbf{K}$ *is reduced to about 84% of its original size. Choosing a too large threshold (*10^{-4} *and larger) changes the physical meaning of* $\mathbf{K}$ *and leads to a wrong velocity/pressure solution, as indicated by an increasing iteration count and changing properties of* $\mathbf{K}$ *(smallest eigenvalue and condition number).*

threshold	nnz(K)	% of **K**	iterations inv(**K**)	eig_S	eig_L	cond(**K**)
-	717 k	100	712	3.39E-03	1.84E+03	5.43E+05
1.00E-20	717 k	99.7	713	3.39E-03	1.84E+03	5.43E+05
1.00E-16	685 k	97.1	713	3.39E-03	1.84E+03	5.43E+05
1.00E-12	530 k	84.3	712	3.39E-03	1.84E+03	5.43E+05
1.00E-08	530 k	84.2	713	3.39E-03	1.84E+03	5.43E+05
1.00E-07	530 k	84.2	713	3.39E-03	1.84E+03	5.43E+05
1.00E-06	528 k	84.0	712	3.39E-03	1.84E+03	5.43E+05
1.00E-05	528 k	83.9	712	3.39E-03	1.84E+03	5.43E+05
1.00E-04	527 k	83.8	714	3.39E-03	1.84E+03	5.43E+05
1.00E-03	525 k	83.3	722	3.37E-03	1.84E+03	5.49E+05
1.00E-02	518 k	82.3	777	1.91E-03	1.84E+03	9.67E+05

solution becomes erroneous. However, this simple ”clean-up” of the matrices by removing all entries with absolute values smaller than 10^{-12} reduces the size of $\mathbf{K}$ to about 84% (Tab. 2.2) and the size of $\mathbf{G}$ to about 66% of their original sizes. The solution process (in terms of iteration count and all calculations from beginning to the end) is not affected, but the computational time is reduced roughly proportional to the decrease of $\mathbf{K}$ in memory. This is simply explained by $\mathbf{K}$ usually being the largest mathematical object in a Stokes flow problem so that all operations involving this matrix benefit from its smaller number of non-zeros.

The assembly of $\mathbf{G}$ and the first two terms in F (Neumann boundary condition and body force) is straightforward and follows the concepts discussed in Section 2.2.3. The Dirichlet b.c. is moved to the RHS after the assembly. A fast execution of the assembly process in the programming language MATLAB requires a vectorization, which is based on the block-wise element assembly suggested by Dabrowski et al. (2008).

The discussion of strategies for decomposing the saddle point problem into subproblems with positive-definite matrices is discussed in Chapter 2.6 after introducing algorithms to solve matrix equations.

2.4 Solving sparse linear systems

The finite element method in the thermal diffusion problem and the Stokes flow problem lead to matrix equations that have to be solved numerically to derive the solutions for temperature and pressure/velocity, resp. In case of the viscous flow problem, the saddle point matrix is split into two systems of equations, in which only sparse, symmetric, positive-definite matrices appear (see Section 2.6). For these large systems of equations, it is important to use efficient techniques to solve them. The choices can be grouped into direct methods and iterative methods. In the following section I will present selected solution algorithms (referred to as solvers in short hand form) and discuss their properties and applicability to the above numerical problems. After this introduction, I will present the solution algorithm that is implemented in the 2-D and 3-D numerical codes developed in this thesis. This algorithm represents a combination of iterative and direct solvers to benefit from the distinct advantages of each ”stand-alone” algorithm. The sections are structured in the following way: I will first introduce so-called ”Krylov-subspace” methods, which are among the most powerful iterative solvers nowadays, and discuss what needs to be done to accelerate convergence of these algorithms. Afterwards, I will briefly discuss basic iterative schemes like the Jacobi method, and what needs to be done to include these into a multigrid algorithm. Multigrid is perhaps the most powerful iterative tool for very large systems of equations, because the number of iterations until convergence scales very favorable with the size of the linear system. I will discuss a few direct solvers and show how they can be used to support the above mentioned iterative algorithms. Finally, I will present a combination of the above solvers that takes advantage of each algorithm’s strengths. For a large system of equations (especially in 3-D) this combination significantly outperforms each of the solvers on its own.

2.4.1 Krylov subspace methods

This introduction is mostly based on Shewchuk (1994). For the benefit of a clearer reading I will skip citations to this paper in this section. If not mentioned otherwise, all equations and descriptions are taken from Shewchuk (1994).

Quadratic from

A matrix equation

$$\mathbf{A}x = b, \tag{2.96}$$

where b is a known vector, $\mathbf{A}$ is a known matrix and x is an unknown solution vector, can be written as the minimization of the so-called quadratic form, which is a scalar, quadratic function of x:

$$f(x) = \frac{1}{2}x^T\mathbf{A}x - b^Tx + c \tag{2.97}$$

For a symmetric, positive-definite matrix $\mathbf{A}$, the vector x that minimizes (2.97) is also the solution vector in the matrix equation (2.96). Thus, the solution can be found by using the gradient $f'(x)$ of the quadratic form:

$$f'(x) = \frac{1}{2}\mathbf{A}^Tx + \frac{1}{2}\mathbf{A}x - b \quad \text{,which (if A is symmetric) reduces to} \tag{2.98a}$$

$$f'(x) = \mathbf{A}x - b \tag{2.98b}$$

Setting (2.98b) equal to zero and solving for x will give the solution vector for (2.96). Examples for a quadratic form $f(x)$ and its gradient $f'(x)$ are illustrated in Fig. 2.4.

The shape of the gradient function depends on the properties of the matrix $\mathbf{A}$. From the examples shown in Fig. 2.5 it becomes clear that equations involving positive-definite and negative-definite matrices can be solved by locating the (unique) extremum of their associated quadratic form, while equations with singular or saddle point matrices (as, for instance, in the Stokes flow problem) require more advanced solution strategies.

Steepest descent

Some definitions are required before proceeding:

$$e_i = x_{(i)} - x \qquad \text{error vector (unknown)} \tag{2.99a}$$

$$r_i = b - \mathbf{A}x_{(i)} \qquad \text{residual vector (known, equal to } -f'(x)) \tag{2.99b}$$

$$r_i = -\mathbf{A}e_i \qquad \text{follows from (2.99a)-(2.99b) and (2.96)} \tag{2.99c}$$

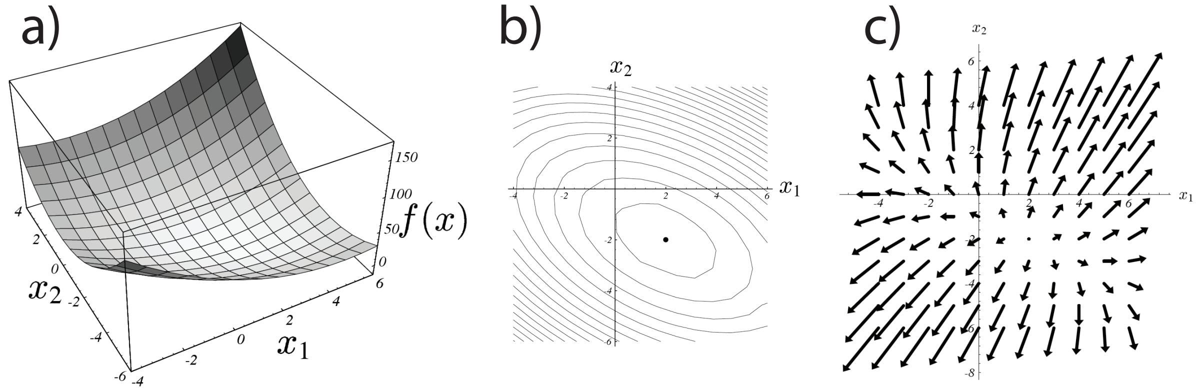


Figure 2.4: *The quadratic form of the matrix equation $\mathbf{A}x = b$, where $\mathbf{A} = \left[\begin{smallmatrix}3 & 2\\ 2 & 6\end{smallmatrix}\right]$ and $b = \left(\begin{smallmatrix}2\\ -8\end{smallmatrix}\right)$, is shown as a surface (a) and as contours (b). If the matrix is positive-definite, the quadratic form has a unique minimum location (point in b), which can be found by locating the point of a zero gradient of the quadratic form (c). Figure taken from Shewchuk (1994).*

The ”Method of Steepest Descent” (SD) makes use of the gradient $f'(x)$, given by the current residual vector (2.99b). Starting at an arbitrary point x_0 that represents an initial guess for the solution x, SD ”follows” the steepest gradient of f, which is given by the residual vector $r_{(0)}$. The step length $\alpha_{(0)}$ into this direction is chosen such that f is minimized in direction $r_{(0)}$, i.e. the new guess $x_{(1)}$ is located at the minimum along the line $r_{(0)}$. This point coincides with the location, where the gradient f' is perpendicular to $r_{(0)}$, so that the point $x_{(1)}$ is reached when $r^T_{(1)}r_{(0)} = 0$ (see Fig. 2.6). This constraint

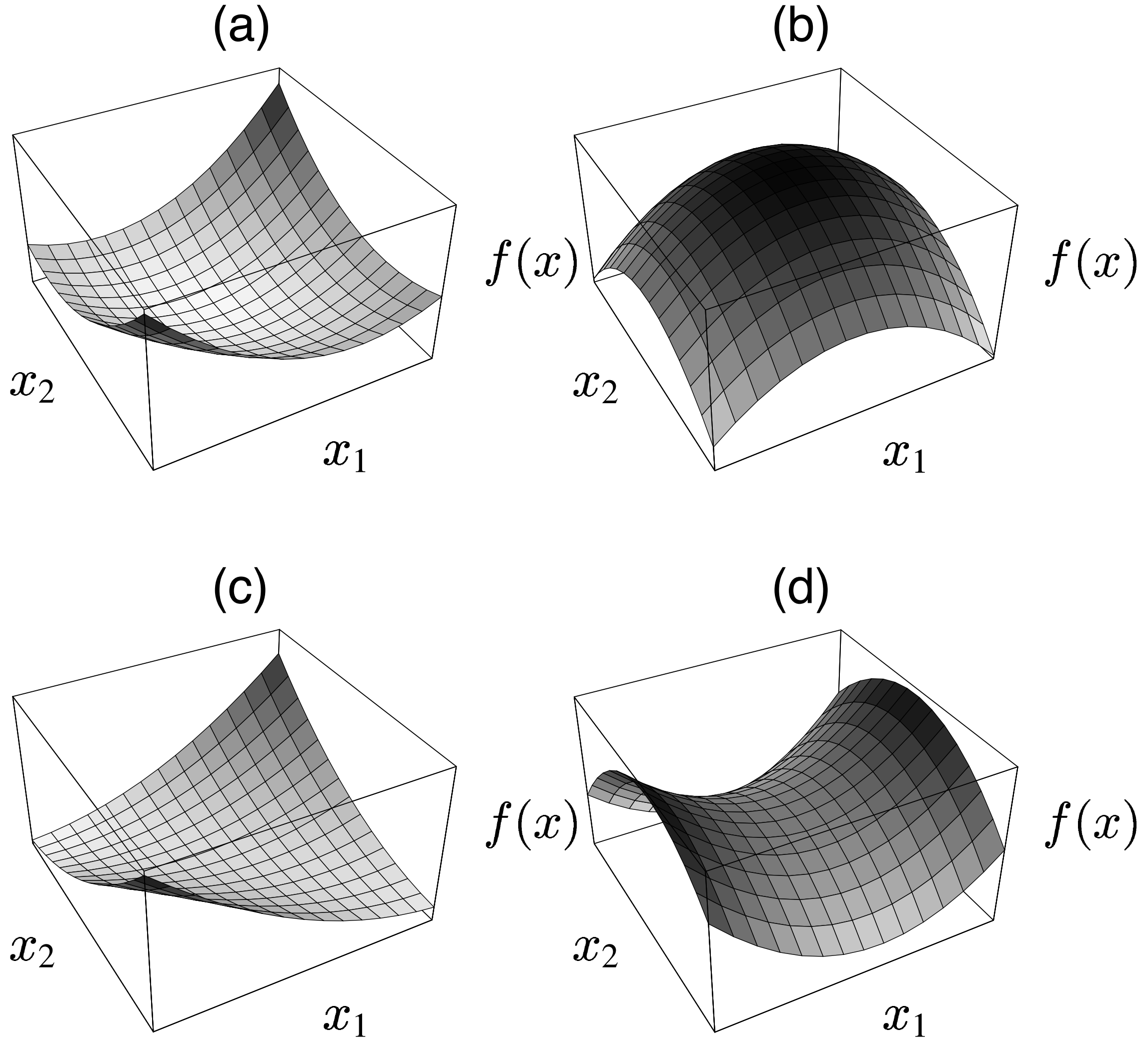


Figure 2.5: *Examples of quadratic forms corresponding to a (a) positive-definite matrix, (b) negative-definite matrix, (c) singular (positive-indefinite) matrix, and (d) indefinite matrix (saddle point matrix). (a) and (b) have unique solutions that can be found using the gradient of the quadratic form. (c) has a set of possible solutions that lie in the minimum valley rather than in a minimum point. Saddle point matrices (d) require a different solution strategy because the minimum (if it exists) is not the solution vector x. Figure taken from Shewchuk (1994).*

defines the distance $\alpha_{(0)}$ to go in the direction $r_{(0)}$

$$\begin{aligned}
r_{(1)}^T r_{(0)} &= 0 \\
\left(b - \mathbf{A}x_{(1)}\right)^T r_{(0)} &= 0 \\
\left(b - \mathbf{A}\left(x_{(0)} + \alpha r_{(0)}\right)\right)^T r_{(0)} &= 0 \\
\left(b - \mathbf{A}x_{(0)}\right)^T r_{(0)} - \alpha \left(\mathbf{A}r_{(0)}\right)^T r_{(0)} &= 0 \\
\left(b - \mathbf{A}x_{(0)}\right)^T r_{(0)} &= \alpha \left(\mathbf{A}r_{(0)}\right)^T r_{(0)} \\
r_{(0)}^T r_{(0)} &= \alpha r_{(0)}^T \left(\mathbf{A}r_{(0)}\right) \\
\alpha &= \frac{r_{(0)}^T r_{(0)}}{r_{(0)}^T \mathbf{A} r_{(0)}} \qquad (2.100)
\end{aligned}$$

As opposed to the error vector $e_{(i)}$, the current residual $r_{(i)}$ can always be calculated during the solution process so that the direction for SD is known. The SD algorithm can then be written as

$$r_{(i)} = b - \mathbf{A}x_{(i)} \qquad \text{calculate residual} \qquad (2.101a)$$

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T \mathbf{A} r_{(i)}} \qquad \text{step size into direction } r_{(i)} \qquad (2.101b)$$

$$x_{(i+1)} = x(i) + \alpha_{(i)} r_{(i)} \qquad \text{new } x \text{ at the end of the step} \qquad (2.101c)$$

For symmetric, positive-definite matrices $\mathbf{A}$, the above algorithm converges towards the solution x. Most of the computational time is consumed by the multiplications of $\mathbf{A}$ times a vector. These two multiplications can be reduced to a single one: Premultiplying

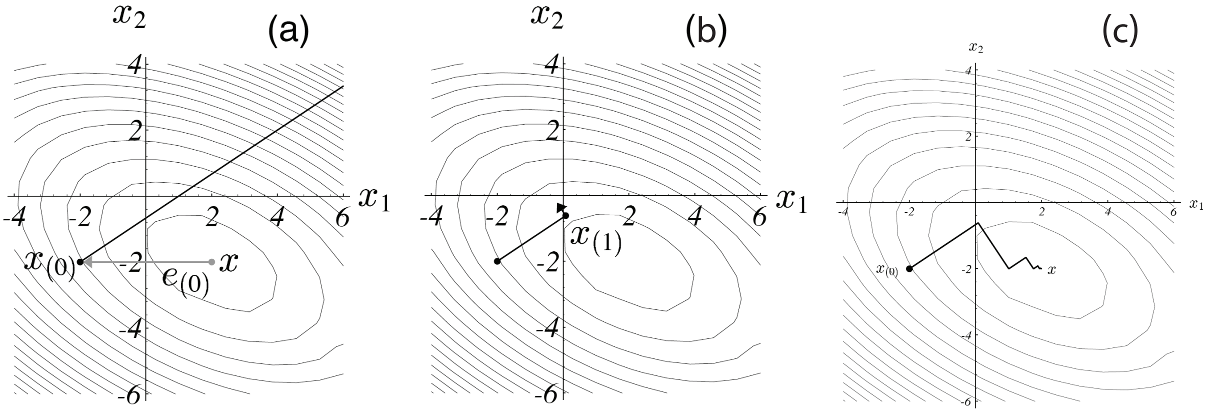


Figure 2.6: *Method of Steepest Descent; elliptic lines in all pictures are the contours of the quadratic form f that has a minimum at x. (a) Starting at a first guess $x_{(0)}$, march into the direction in which the residual vector $r_{(0)}$ points (black line). The error $e_{(0)}$ (grey line) of $x_{(0)}$ is unknown. (b) Stop at the point $x_{(1)}$, where the gradient of f is perpendicular to this direction (i.e. stop when $r_{(1)}^T r_{(0)} = 0$). This step length is called $\alpha_{(0)}$. (c) March into the new direction give by $r_{(1)}$ with a step length $\alpha_{(1)}$ after which $r_{(2)}^T r_{(1)} = 0$. Repeat this cycle until arriving at the solution x. Figures taken from Shewchuk (1994).*

(2.101c) by $-\mathbf{A}$ and adding b gives an expression for the new residual $r_{(i+1)}$:

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)}\mathbf{A}r_{(i)} \tag{2.102}$$

After calculating a first residual $r_{(0)}$ using (2.101a), the residual is updated during the iterations using (2.102). The potential danger in this computational speed-up is that accumulation of roundoff can cause a convergence of SD to a point near x rather than at x. This is because x and r evolve without feedback when using (2.102), so that roundoff will not be corrected for unless a correct residual is calculated using (2.101a) from time to time.

Eigenvectors and eigenvalues

Before continuing with the ideas behind the Conjugate Gradient Method (CG) it is useful to introduce the concept of eigenvalues and eigenvectors of a matrix. Following Shewchuk (1994), an eigenvector $v \in \mathbb{R}^n$ of a matrix $\mathbf{B} \in \mathbb{R}^{nxn}$ is a "nonzero vector that does not rotate when $\mathbf{B}$ is applied to it." This means that v may change its length or reverse its direction, but it does not shift in the n-dimensional space. Accordingly, an eigenvector v and it's corresponding eigenvalue λ are defined as

$$\mathbf{B}v = \lambda v \tag{2.103}$$

where v is a nonzero vector and λ is a scalar constant. Eigenvectors can be scaled, so that for any scalar constant c, the vector $c\,v$ is also an eigenvector of $\mathbf{B}$. In general, eigenvectors are defined to have length 1.

Eigenvectors are important because iterative solvers multiply repeatedly a matrix by a vector. If v is multiplied by $\mathbf{B}$ over and over again, the result either vanishes (if $|\lambda| < 1$) or goes to infinity (if $|\lambda| > 1$). A full-rank symmetric matrix $\mathbf{B} \in \mathbb{R}^{nxn}$ has n linearly independent eigenvectors denoted $v_1, v_2, ..., v_n$, but some of the corresponding eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$ may be equal to each other. Since the v's are linearly independent, each vector in $\mathbb{R}^n$ can be constructed from a linear combination of eigenvectors of $\mathbf{B}$. Hence, each multiplication of $\mathbf{B}$ to a vector x can be expressed as the linear combination of $\mathbf{B}$ times each of its eigenvectors. The k-th multiplication can be written as:

$$B^k x = \sum_{i=1}^{n} B^k \xi_i v_i = \sum_{i=1}^{n} \lambda_i^k \xi_i v_i \tag{2.104}$$

Whether the repeated multiplications converge to zero or diverge to infinity for $k \to \infty$ depends on the spectral radius ρ of matrix $\mathbf{B}$, which is defined as the eigenvalue with the largest magnitude. The condition number of a matrix is defined as the ratio between the largest and the smallest eigenvalue.

$$\rho(\mathbf{B}) = max|\lambda_i| \qquad \text{(spectral radius)} \tag{2.105}$$

$$\kappa(\mathbf{B}) = \lambda_{max}/\lambda_{min} \qquad \text{(condition number)} \tag{2.106}$$

Furthermore, all eigenvalues of a positive-definite matrix are positive. From the definition of positive-definite ($v^T B v > 0$) and Eq. (2.103) follows $v^T B v = \lambda v^T v$, hence λ must be positive.

Conjugate Directions

The method of Steepest Descent often takes steps into the same direction as earlier steps (see Fig. 2.6c). An improvement to the algorithm would be, to step into each direction only once and with a correct step size, so that we never have to go into this direction again. For the SD example above, such an algorithm would converge in two steps (Fig. 2.7a). The correct step size $\alpha_{(0)}$ in this case depends on the orientation of the error vector $e_{(1)}$. Using search directions $q_{(0)}, q_{(1)}, ..., q_{(n-1)}$ instead of the residual vectors $r_{(i)}$, a step size α and a new x after each step would be given by

$$x_{i+1} = x_{(i)} + \alpha_{(i)} q_{(i)} \qquad \text{new } x \text{ after the step} \tag{2.107a}$$

$$\begin{aligned} q_{(i)}^T e_{(i+1)} &= 0 \\ q_{(i)}^T \left(e_{(i)} + \alpha_{(i)} q_{(i)}\right) &= 0 \\ \alpha_{(i)} &= \frac{q_{(i)}^T e_{(i)}}{q_{(i)}^T q_{(i)}} \qquad \text{step size into direction } q_{(i)} \text{ (insolvable)} \end{aligned} \tag{2.107b}$$

Unfortunately, the error vector $e_{(i)}$ is unknown. However, $\mathbf{A} e_{(i)} = r_{(i)}$ is known:

$$\alpha_{(i)} = \frac{q_{(i)}^T \mathbf{A} e_{(i)}}{q_{(i)}^T \mathbf{A} q_{(i)}} = \frac{q_{(i)}^T r_{(i)}}{q_{(i)}^T \mathbf{A} q_{(i)}} \qquad \text{step size into direction } q_{(i)} \text{ (solvable)} \tag{2.107c}$$

In this formulation, the step size $\alpha_{(i)}$ is constraint by $q_{(i)}$ and $e_{(i)}$ being "A-orthogonal" rather than orthogonal (see Fig. 2.7b). To complete the algorithm, a set of A-orthogonal search directions $\{q_{(i)}\}$ is needed. These can be constructed from a set of n linearly independent vectors $u_0, u_1, ..., u_{n-1}$ from which recursively any component is subtracted out that is not A-orthogonal to previous q vectors. This is the so-called Gram-Schmidt process: Set $q_{(0)} = u_0$ and for $i > 0$ set

$$q_{(i)} = u_i + \sum_{k=0}^{i-1} \beta_{ik} q_{(k)} \tag{2.108a}$$

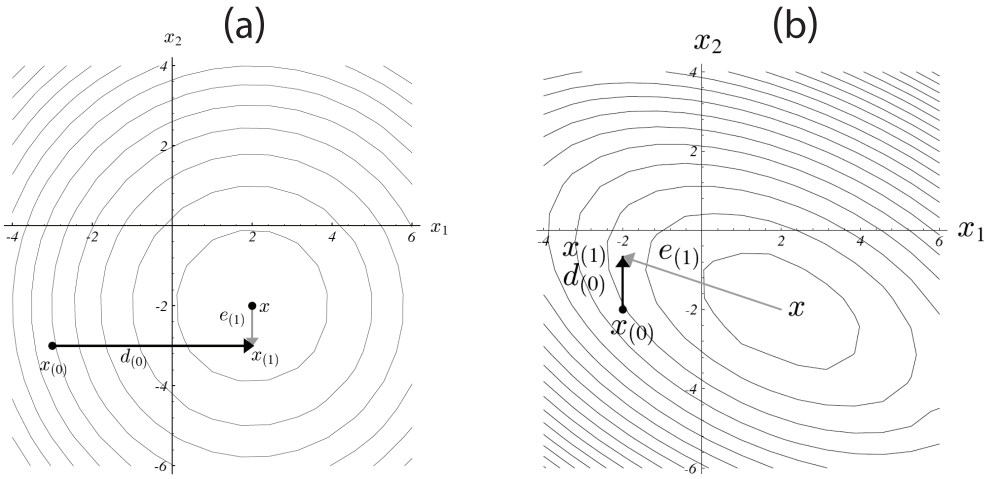


Figure 2.7: *(a) Hypothetical Method of Orthogonal Directions; the error $e_{(1)}$ is as unknown as the solution x. (b) Method of Conjugate Directions. Instead of using the unaccessible constraint $r^T_{(i)}e_{(i+1)} = 0$, the step size $\alpha_{(i)}$ is constraint by $r^T_{(i)}\mathbf{A}e_{(i+1)} = r^T_{(i)}r_{(i+1)} = 0$. The latter leads to $r_{(i)}$ and $e_{(i+1)}$ being A-orthogonal rather than orthogonal as in (a). The difference between orthogonal and A-orthogonal is best pictured by the contours: A-orthogonal means stretched (or scaled) by A. Figures taken from Shewchuk (1994).*

where coefficients β_{ik} are defined for $i > k$ as

$$\beta_{ik} = -\frac{u_i^T \mathbf{A} q_{(k)}}{q_{(k)} \mathbf{A} q_{(k)}} \tag{2.108b}$$

The above Method of Conjugate Direction (CD) has the major drawback that all old search vectors $q_{(j)}, j < i$, have to be stored in order to calculate the new search direction $q_{(i)}$. The "classical" Gram-Schmidt process (2.108) also suffers from the accumulation of roundoff that can lead to a linear dependence among the q's as the number of vectors increases. The "modified" Gram-Schmidt process (Saad, 2003, p. 11) overcomes the latter problem. However, a lot of computational effort goes into the A-orthogonalization of the search directions, which is why CD has received little attention.

To proof that CD converges within n iterations (in the absence of roundoff errors), the initial error $e_{(0)}$ is expressed as a linear combination of all search directions:

$$e_{(0)} = \sum_{j=0}^{n-1} \delta_j q_{(j)} \tag{2.109a}$$

The coefficients δ are determined by multiplying (2.109a) by $q_{(k)}^T\mathbf{A}$:

$$\begin{aligned}
q_{(k)}^T\mathbf{A}e_{(0)} &= \sum_{j=0}^{n-1}\delta_j q_{(k)}^T\mathbf{A}q_{(j)} \\
&= \delta_k q_{(k)}^T\mathbf{A}q_{(k)} && \text{step (i)} \\
\Rightarrow \delta_{(k)} &= \frac{q_{(k)}\mathbf{A}e_{(0)}}{q_{(k)}^T\mathbf{A}q_{(k)}} \\
&= \frac{q_{(k)}\mathbf{A}\left(e_{(0)}+\sum_{i=0}^{k-1}\alpha_{(i)}q_{(i)}\right)}{q_{(k)}^T\mathbf{A}q_{(k)}} && \text{step (ii)} && (2.109b) \\
&= \frac{q_{(k)}\mathbf{A}e_{(k)}}{q_{(k)}^T\mathbf{A}q_{(k)}} && \text{step (iii)} && (2.109c)
\end{aligned}$$

Step (i) and (ii) use that all q's are A-orthogonal: $q_{(k)}\mathbf{A}q_i = 0$, for all $i \neq k$. Step (iii) is true because $x_{(i+1)} = x_{(i)} + \alpha_{(i)}q_{(i)}$ (2.107b) and $x_{(i)} = x + e_{(i)}$, thus, $e_{(k)} = e_{(k-1)} + \alpha_{(k-1)}q_{(k-1)}$). A comparison of the step size $\alpha_{(i)}$ in (2.107c) and $\delta_{(k)}$ in (2.109c) shows that $\alpha_{(i)} = -\delta_{(i)}$. From this analogy follows

$$\begin{aligned}
e_{(i)} &= e_{(0)} + \sum_{j=0}^{i-1}\alpha_{(j)}q_{(j)} \\
&= \sum_{j=0}^{n-1}\delta_j q_{(j)} - \sum_{j=0}^{i-1}\delta_{(j)}q_{(j)} && \text{using } \alpha_{(i)} = -\delta_{(i)} \\
&= \sum_{j=i}^{n-1}\delta_{(j)}q_{(j)} && (2.110)
\end{aligned}$$

The error $e_{(i)}$ remaining at iteration i only contains components of upcoming search direction vectors $q_{(j)}$, where $j = i, ..., n$. Every iteration $j = 1, ..., i$ has removed a search vector component form the error so that CD will converge within n iterations because $e_n = 0$, thus, $x_n = x$. Hence, being able to find search vectors that have large components in the error is essential for a fast convergence to a point very close to x. This conclusion is important for CG, but especially for the preconditioning of CG, because an iterative solver requiring n iterations is impractical for large linear system.

Conjugate Gradients

The very popular Method of Conjugate Gradients (CG) (Hestenes and Stiefel, 1952) emerges from a small but essential modification of the Method of Conjugate Directions (CD): The vector $u_{(i)}$ in (2.108a), required to construct the new search direction $q_{(i)}$, is replaced by the residual $r_{(i)}$. The reason for this choice is that each residual vector is orthogonal to all previous search directions

$$\begin{aligned} e_{(j)} &= \sum_{k=j}^{n-1} \delta_{(k)} q_{(k)} && \text{by Eq. (2.110)} \\ -q_{(i)}^T \mathbf{A} e_{(j)} &= -\sum_{k=j}^{n-1} \delta_{(k)} q_{(i)}^T \mathbf{A} q_{(k)} && \text{multiplied by } -q_{(i)}^T\mathbf{A} \\ q_{(i)}^T r_{(j)} &= 0 \quad \text{, for} \quad i < j && \text{A-orthogonality of q's} \end{aligned} \tag{2.111}$$

The implication of this finding is that new A-orthogonal search directions q can be calculated without the computationally very expensive orthogonalization to all previous search directions. Instead it is sufficient to make the new residual A-orthogonal to the last one. The CG algorithm can be written as

$$q_{(0)} = r_{(0)} = b - \mathbf{A}x_{(0)} \qquad \text{initial residual and first search direction } q_{(0)} \tag{2.112a}$$

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{q_{(i)} \mathbf{A} q_{(i)}} \qquad \text{step size constraint} \tag{2.112b}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} q_{(i)} \qquad \text{new approximate solution after the step} \tag{2.112c}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} \mathbf{A} q_{(i)} \qquad \text{new residual after the step} \tag{2.112d}$$

$$\beta_{(i+1)} = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} \qquad \text{find part of } r_{(i+1)} \text{ that is not A-orth. to } r_{(i)} \tag{2.112e}$$

$$q_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} q_{(i)} \qquad \text{new q (is A-orth. to all previous q)} \tag{2.112f}$$

The step size α is chosen such that the search direction becomes A-orthogonal to the error (see Eq. (2.107c) and Fig. 2.7b). In other words, the length of the new error vector is minimized within the A-orthogonal space, i.e. the energy norm of the error $||e_{(i)}||_A^2$ is minimized. Using (2.110) the energy norm of $e_{(i)}$ can be written as a summation

$$\begin{aligned} ||e_{(i)}||_A^2 &= e_{(i)}^T \mathbf{A} e_{(i)} \\ &= \sum_{j=i}^{n-1} \delta_{(j)}^2 q_{(j)}^T \mathbf{A} q_{(j)} \end{aligned} \tag{2.113}$$

Conjugate Gradients is one of the so-called Krylov subspace methods. The subspace $\mathscr{Q}$ spanned by the search directions is constructed by repeated multiplications of $\mathbf{A}$ times a vector (in this case the residual vector). A subspace created in this way is called a Krylov subspace:

$$\begin{aligned} \mathscr{Q} &= span\left\{r_{(0)}, \mathbf{A}r_{(0)}, \mathbf{A}^2 r_{(0)}, ..., \mathbf{A}^{i-1} r_{(0)}\right\} \\ &= span\left\{\mathbf{A}e_{(0)}, \mathbf{A}^2 e_{(0)}, \mathbf{A}^3 e_{(0)}, ..., \mathbf{A}^i r_{(0)}\right\} \end{aligned} \tag{2.114}$$

In every iteration i, CG finds a solution within the subspace $\mathscr{Q}_{(i)}$ defined by the search directions $q_{(i)}$. Considering (2.114), the error at iteration i can be expressed as

$$e_{(i)} = \left(\mathbf{I} + \sum_{j=1}^{i} \psi_{(j)} \mathbf{A}^j \right) e_{(0)} \tag{2.115}$$

where the coefficients $\psi_{(j)}$ depend on $\alpha_{(i)}$ and $\beta_{(i)}$ in (2.112). The expression in parentheses in (2.115) represents a polynomial, so that $e_{(i)} = P_i(\mathbf{A})e_{(0)}$, where $P_i(\mathbf{A})$ is a polynomial of degree i and $P_i(0) = 1$. Using the definitions of eigenvectors (2.103) we can write

$$P_i(\mathbf{A})v = P_i(\lambda)v \tag{2.116}$$

If the initial error is expressed as a linear combination of the eigenvectors, i.e.

$$e_{(0)} = \sum_{j=1}^{n} \xi_j v_j \tag{2.117}$$

it follows from (2.116) and (2.117) that

$$\begin{aligned} e_{(i)} &= \sum_{j=1}^{i} \xi_j P_i(\lambda_j) v_j \\ \mathbf{A}e_{(i)} &= \sum_{j=1}^{i} \xi_j P_i(\lambda_j)\lambda_j v_j \\ ||e_{(i)}||_A^2 &= \sum_{j=1}^{i} \xi_j^2 P_i(\lambda_j)^2 \lambda_j \end{aligned} \tag{2.118}$$

The eigenvectors disappear from (2.118), because their inner product is either zero or one because they are orthogonal and have a length of one by definition. Since CG minimizes (2.118) at every iteration, the convergence rate depends on the eigenvalues λ of $\mathbf{A}$, more precisely, on their distribution and the condition number of $\mathbf{A}$. See pp. 35 in Shewchuk (1994) and pp. 203 in Saad (2003) for a detailed analysis of CG convergence.

The number of iterations until convergence also depends on the quality of the initial guess. The methods of Steepest Descent and Conjugate Gradient are compared in Fig. 2.8a–b using a 1-D heat conduction problem and two initial guesses for the solution (which is zero everywhere in this example). In both experiments the good convergence of SD in the beginning stalls after about 10 iterations. From then on, the flat average slope leads to a very large number of iterations (9591 and 10705, resp.) until convergence, even though the problem is very small (n = 63 unknowns). Poor SD convergence can be a consequence of an unfavorable shaped quadratic form, as sketched in Fig. 2.8c for a problem with two unknowns: SD always follows the gradient, which can lead to an

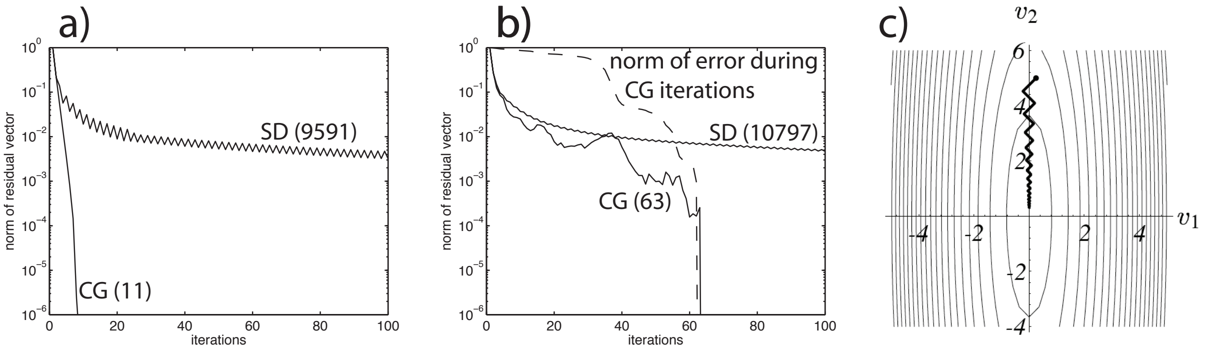


Figure 2.8: *Convergence, defined as $||r_{(i)}||$, of SD and CG for a 1-D steady state heat diffusion problem with constant conductivity (problem size: 63 unknowns). The solution to the problem is $x = 0$ everywhere. The numbers in parenthesis are the iterations required to reach $||r_{(i)}|| < ||r_{(0)}|| \cdot 10^{-6}$. (a) Convergence with a starting guess composed of 7 sine functions. (b) same as (a) but random noise added to the starting guess. The dashed line shows the norm of the error $||e_{(i)}||$. (c) Illustration to explain the slow SD convergence. Figure (c) taken from Shewchuk (1994).*

zig-zag path between the flanks of the valley with an extremely slow progress towards the minimum.

In contrast, CG does not follow the gradient (except for its first step) but uses search directions. In the first experiment (Fig. 2.8a) CG converges very fast within 11 iterations. The initial guess here is composed of 7 sine functions with different wave length. Thus, the initial error has a limited number of components, which CG effectively removes in each step. Adding white noise to the initial guess (Fig. 2.8b) results in as many error components as there are unknowns. Since CG takes one step into each of the linearly independent search directions, it must converge within 63 iterations (which is observed). The second experiment also shows a typical CG convergence path, during which the norm of the residual vector can grow for several iterations before converging again. This is a consequence of removing search direction components in the error completely rather than depending on gradient of the quadratic form (i.e. the residual) as SD does. While the residual occasionally increases, the norm of the error is alway decreasing monotonically, because every iteration removes one of its components. This can be zero in the worst case, but this would not increase the error.

Although much more efficient than SD, a Conjugate Gradient algorithm that converges in n iterations is impractical for large systems, where $n = 10^6$ or larger. The technique of preconditioning, which is discussed next, aims to overcome this problem.

Preconditioned Conjugate Gradient

The last ingredient needed to make CG a very powerful iterative solver is the technique of preconditioning the matrix equation. Preconditioning is a way to improve the condition number of $\mathbf{A}$ and thereby accelerate the convergence rate by potentially orders of magnitude.

Suppose a symmetric, positive-definite matrix $\mathbf{M}$ can be found that approximates $\mathbf{A}$ but is easier to invert. In this case $\mathbf{A}x = b$ can be transformed into

$$\mathbf{M}^{-1}\mathbf{A}x = \mathbf{M}^{-1}b \tag{2.119}$$

The problem here is that $\mathbf{M}^{-1}\mathbf{A}$ is not necessarily either symmetric or definite (even if $\mathbf{M}$ and $\mathbf{A}$ are). One way to maintain symmetry is to calculate a factorized matrix $\mathbf{L}$ such that $\mathbf{L}\,\mathbf{L}^T = \mathbf{M}$, and apply this to $\mathbf{A}x = b$:

$$\mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-T}\hat{x} = \mathbf{L}^{-1}b \quad , \hat{x} = \mathbf{L}^T x \tag{2.120}$$

This approach, however, is rarely used, since the factorization of $\mathbf{M}$ can be very time consuming, especially if $\mathbf{A}$ describes a large linear system. It also requires that $\mathbf{M}$ is an explicitly formed matrix, which I will show soon, is not always the case.

A more popular way to precondition CG is given by Saad (2003, pp.263, algorithm 9.1). As mentioned above, substituting $\mathbf{M}^{-1}\mathbf{A}$ into the CG algorithm (2.112) will likely lead to a non-converging algorithm, because

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{q_{(i)}^T \mathbf{M}^{-1}\mathbf{A} q_{(i)}} \tag{2.121}$$

is not necessarily a symmetric operation. Symmetry of the inner product in (2.121) can be re-established by replacing the Euclidean inner product (denoted $x^T y$) by an "$\mathbf{M}$-inner product", which is defined as

$$(x^T y)_M \equiv (\mathbf{M}x)^T y = x^T(\mathbf{M}y) \tag{2.122}$$

This will lead to symmetry in the algorithm, because the operator $\mathbf{M}^{-1}\mathbf{A}$ is symmetric with respect to the $\mathbf{M}$-inner product:

$$((\mathbf{M}^{-1}\mathbf{A}x)^T y)_M = (\mathbf{A}x)^T y = x^T(\mathbf{A}y) = x^T(\mathbf{M}(\mathbf{M}^{-1}\mathbf{A})y) = (x^T\mathbf{M}^{-1}\mathbf{A}y)_M \tag{2.123}$$

Replacing all Euclidean inner products in (2.112) by $\mathbf{M}$-inner products leads to the commonly used preconditioned conjugate gradient algorithm (PCG):

$$r_{(0)} = b - \mathbf{A}x_{(0)} \quad \text{initial residual} \tag{2.124a}$$

$$d_{(0)} = \mathbf{M}^{-1}r_{(0)} \quad \text{preconditioned } r_{(0)} \tag{2.124b}$$

$$q_{(0)} = d_{(0)} \quad \text{use } d_{(0)} \text{ as the first search direction} \tag{2.124c}$$

FOR $i = 0, 1, ...,$until convergence

$$\alpha_i = \frac{r_{(i)}^T d_{(i)}}{q_{(i)}^T \mathbf{A} q_{(i)}} \qquad \text{step size into direction } q_{(i)} \tag{2.124d}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} q_{(i)} \qquad \text{new approx. solution} \tag{2.124e}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} \mathbf{A} q_{(i)} \qquad \text{new residual} \tag{2.124f}$$

$$d_{(i+1)} = \mathbf{M}^{-1} r_{(i+1)} \qquad \text{preconditioned } r_{(i+1)} \tag{2.124g}$$

$$\beta_{(i)} = \frac{r_{(i+1)}^T d_{(i+1)}}{r_{(i)}^T d_{(i)}} \qquad \text{part of } r_{(i+1)} \text{ that is not A-orthogonal} \tag{2.124h}$$

$$q_{(i+1)} = d_{(i+1)} + \beta_{(i)} q_{(i)} \qquad \text{new search direction; now A-orthogonal} \tag{2.124i}$$

END

The advantage of (2.124) over (2.120) is that the factorization of $\mathbf{M}$ is avoided, but the action of the operator $\mathbf{M}^{-1}\mathbf{A}$ is kept symmetric. Note that $r^T d = r^T \mathbf{M}^{-1} r$ is the $\mathbf{M}$-inner product corresponding to $(r^T r)_2 = (r^T \mathbf{M}^{-1} r)_M$.

A few comments on (2.124):

- Setting $d = r$ (i.e. $\mathbf{M} = \mathbf{M}^{-1} = \mathbf{I}$) in the preconditioning steps reduces the algorithm to the un-preconditioned CG method presented in (2.112).

- $\mathbf{M}^{-1}\,r$ is not necessarily a matrix-vector multiplication; it can be anything that approximates the action of $\mathbf{A}^{-1}$ on r, for example, an incomplete Cholesky factorization or another iterative solver

- Whatever choice is made for $\mathbf{M}^{-1}$, it has to be equivalent to a symmetric matrix — otherwise the $\mathbf{M}$-inner product will not be symmetric and PCG is not guaranteed to converge

- In situations, where $\mathbf{M}^{-1}\,r_{(i)}$ is not an exactly symmetric operation (but close to one), the PCG algorithm can fail to converge. Using an alternative formulation to calculate β makes the algorithm more robust in this regard (discussed below).

- All operations within the PCG-loop, specifically in the preconditioning step, have to be restricted with regard to roundoff. Accumulated roundoff will eventually lead to linearly dependent search directions (q) and/or re-introduced error components corresponding to previous search directions (which will not be chosen by CG again). In this case, a restart of PCG with the current x as starting guess is the best option.

The technique of preconditioning has been introduced as a way to improve the condition number of $\mathbf{A}$. Another, maybe more intuitive way, is to think of (2.124b) and (2.124g) as providing a good estimate for the unknown error $e_{(i)}$ of the current solution $x_{(i)}$. Knowing the error exactly (i.e. $\mathbf{M} = \mathbf{A}$) would lead to a convergence to the solution in one iteration, because the search direction $q_{(i)}$ would cross x. Since the step size $\alpha_{(i)}$ is chosen to match the point where the remaining error $e_{(i+1)}$ is A-orthogonal to $q_{(i)}$, we would stop next to the solution x. The next two sections will briefly review basic (stationary) iterative solvers and direct solvers. These, in combination with multigrid, can be used to provide a very good error guess.

2.4.2 Basic iterative methods and multigrid

I will focus on the Jacobi iterative method as an example of a basic iterative method, because this method is used in parts of the numerical codes. It has the advantage that its parallelization is simple compared to other, stationary iterative methods such as Gauss-Seidel, successive overrelaxation (SOR) or block-relaxation schemes. The latter algorithms are not covered in this short review — see (Saad, 2003, chapter 4) for more information on this topic.

In the Jacobi method, a matrix equation $\mathbf{A}x = b$ is solved by splitting matrix $\mathbf{A}$ into its diagonal part $\mathbf{D}$ and all remaining, off-diagonal elements $\mathbf{E}$, so that $\mathbf{A} = \mathbf{D} + \mathbf{E}$. The matrix equation then becomes

$$\begin{aligned}
\mathbf{A}x &= b \\
(\mathbf{D} + \mathbf{E})x &= b \\
\mathbf{D}x &= -\mathbf{E}x + b \\
\mathbf{D}x &= \mathbf{D}x - \mathbf{E}x - \mathbf{D}x + b \\
\mathbf{D}x &= \mathbf{D}x + (b - \mathbf{A}x) \\
x_{(i+1)} &= x_{(i)} + \mathbf{D}^{-1}\, r_{(i)} \qquad (2.125)
\end{aligned}$$

The multiplication in (2.125) is equivalent to solving each equation in $\mathbf{A}x = b$. The components of the solution vector x are connected to each other by the non-zero entries in the coefficient matrix $\mathbf{A}$. Here, these dependencies are accounted for by using the x values from the previous iteration so that they can be moved to the right-hand-side. Given an initial guess $x_{(0)}$, equation (2.125) can be used to iteratively obtain the solution x.

A more general formulation includes a weighting factor ω that results in a weighted average between the previous solution $x_{(i)}$ and the new solution $x_{(i+1)}$.

$$x_{(i+1)} = x_{(i)} + \omega\, \mathbf{D}^{-1}\, r_{(i)} \qquad (2.126)$$

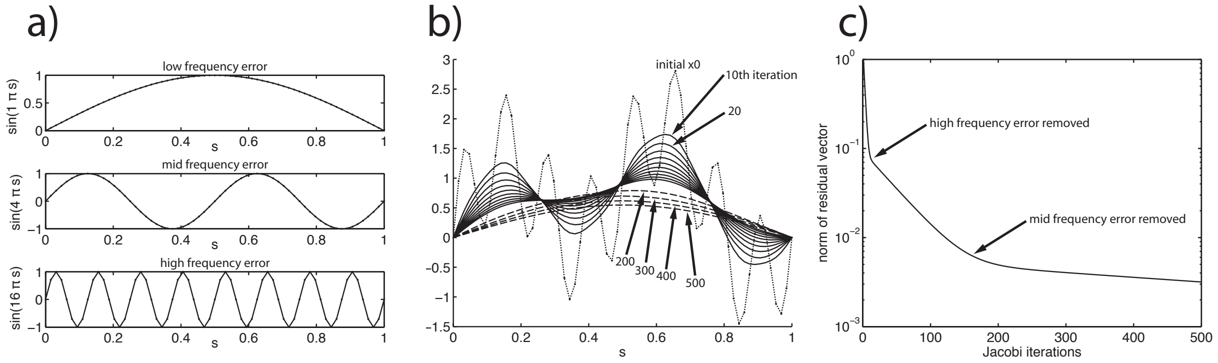


Figure 2.9: *Convergence of the Jacobi iterative method for a steady state heat diffusion problem with the solution $x = 0$. This choice means, the value of x during the iterations shows the error of the current solution. The mesh has 65 nodes and linear 1-D elements. (a) The initial guess is composed of the superposition of three sine functions of magnitude 1 and of different wave lengths. (b) Evolution of x during the Jacobi iterations; initial guess (dotted), x in steps of 10 iterations (solid) and x after 100, 200, etc iterations (dashed) are shown. (c) The norm of the residual vector ($||r_{(i)}||$) during the iterations. The steep slope in the beginning corresponds to the removal of the high frequency part of the error, intermediate slope to removal of the mid-frequency error. The low frequency error persists for several hundred iterations and is inefficiently reduced by the Jacobi method.*

For $\omega < 1$, (2.126) is referred to as a *damped* or *weighted* Jacobi iteration. The convergence behavior of the undamped algorithm is nicely illustrated using a 1-D finite element formulation for the steady state heat conduction problem (Fig. 2.9). The boundary conditions at the ends of the 1-D domain are both equal to zero and neither advection nor source terms are included. Thus, the correct solution of the problem is zero everywhere. By choosing different sine functions for the initial guess $x_{(0)}$, the evolution of x during the iterations shows, which part of the error is removed quickly and which part survives many iterations. Clearly, the Jacobi method performs well on the short-wave length error but fails to efficiently reduce the long-wave length error.

The convergence rate is controlled by the eigenvalues of the iteration matrix $\mathbf{G}$, which in case of the Jacobi method is

$$\mathbf{G}_J = \left(\mathbf{I} - \omega \mathbf{D}^{-1} \mathbf{A}\right) \tag{2.127}$$

This operator is repeatedly multiplied to a vector, and it can be shown that no convergence will be achieved for $\omega < 0$ and $\omega > 1$ (Saad, 2003). Between these bounds, the fastest overall convergence (i.e. complete removal of the error) is observed for $\omega = 1$, which represents the standard Jacobi method as a stand-alone solver. If the error of the initial guess $x_{(0)}$ is expressed as a linear combination of the eigenvectors of the iteration matrix, it can further be shown that components corresponding to larger eigenvalues converge slower than those associated with small eigenvalues (Saad, 2003). This explains, why short wave length error components decay much faster than long wave length components.

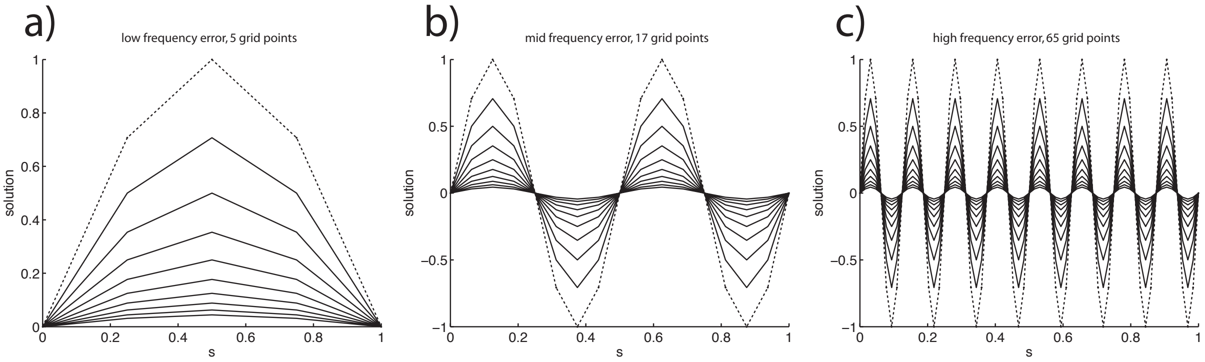


Figure 2.10: *Convergence of the Jacobi iterative method for the same heat conduction problem as in Fig. 2.9. Solid lines show x for each of the first 10 iterations. (1) The low frequency sine function error discretized on a coarse mesh (5 nodes); (b) mid frequency error on a mesh with 17 nodes; (c) high frequency error on a mesh with 65 nodes. All errors are reduced efficiently during the first 10 iterations shown.*

The convergence behavior can also be understood from a less mathematical point of view. Each equation is solved separately and each equation only relates adjacent unknowns to each other, so that a disequilibrium in the solution between neighbors is efficiently corrected. A disequilibrium between nodes a and b that are separated by many other nodes, is only corrected indirectly: The information has to slowly propagate (at a speed of one node per iteration in the given example) through all nodes in-between a and b. At the time the information from node b reaches the distant node a (and vice versa), both have changed their values so that the correction is not optimal.

Fig. 2.10 shows the convergence of the Jacobi method for the three sine functions but discretized separately on meshes with a different number of nodes. Obviously, now the Jacobi method performs well on each of the initial errors. The finding that the spatial discretization affects the convergence behavior has motivated the development of the *geometrical multigrid method* (MG).[2] A correction to the long wave length error is evaluated best on a coarse mesh, where the long wave length error on the fine mesh appears as a short wave length error so that the Jacobi iterations perform efficiently. If several meshes of different node-spacing are employed, and the corrections obtained from all are accumulated, every component of the error is reduced efficiently. These meshes with different spatial resolution are also referred to as MG levels. Only a few Jacobi iterations are required on each mesh to get a good approximation of the respective short wave length error. In this case, the Jacobi iterations are used as a smoother rather than a solver, and $\omega = 1$ is no longer the best choice. Optimal damping factors depend on the

[2] Geometric multigrid is used in all applications covered in this thesis. A different type of multigrid (algebraic multigrid, AMG) does not require the definition of coarser meshes, but removes systematically selected rows and columns from matrix **A** to derive "coarser" approximations to **A** (e.g. Briggs et al., 2000). Geometric multigrid usually performs better than AMG, but requires higher effort for its implementation.

structure of the multigrid but usually are chosen such that $\frac{2}{3} \geq \omega \geq \frac{4}{5}$.

To set up a multigrid algorithm, operators that transfer between the MG levels have to be constructed. They are called restriction matrix R (for transfers to the next coarser level) and interpolation matrix I for the opposite direction. A multigrid algorithm solving $\mathbf{A}x = b$ may then be formulated as:

1. calculate the residual $r^1 = b - \mathbf{A}x$ on the finest mesh h^1 (the mesh on which the problem at hand was discretized in the first place)
2. perform a few damped Jacobi iterations (relaxations) on $\mathbf{A}e^1 = r^1$ to get an approximation to the high frequency part of the error (e^1)
3. restrict the remaining residual to the next coarser mesh h^2 using the restriction operator: $r^2 = R_{1\to 2}\, r^1$
4. relax a few times on level h^2 to get an approximation to the "high frequency" part of the error on this coarser mesh (e^2)
5. continue until the coarsest level h^n has been reached
6. interpolate the error e^n evaluated on the coarsest level to the next finer level using an interpolation operator: $\hat{e}^{n-1} = I_{n\to n-1}\, e^n$
7. add the interpolated error ($\hat{e}^{n-1}$) to the error evaluated on level h^{n-1} (e^{n-1})
8. preform few Jacobi iterations using the remaining residual and interpolate the accumulated error to the next finer mesh
9. continue until the accumulated error is interpolated on the fine mesh
10. correct the current solution using the accumulated error and continue with step 1

The relaxations on the coarser meshes require a restricted version of $\mathbf{A}$. It can be calculated by using restriction and interpolation operators

$$\mathbf{A}^2 = R_{1\to 2}\, \mathbf{A}^1\, I_{2\to 1} \tag{2.128}$$

Fig. 2.11 shows the MG method in practice. Different numbers of multigrid levels are used to solve the same sample problem as in Fig. 2.9. The convergence of the Jacobi method is included in Fig. 2.11a for a better comparison. The more MG levels used, the better the reduction of all components of the error. The 2-level MG, for instance, is not able to remove the long-wave length error, whereas the 6-level MG algorithm does (its coarsest mesh only includes five elements with three free degrees of freedom). The

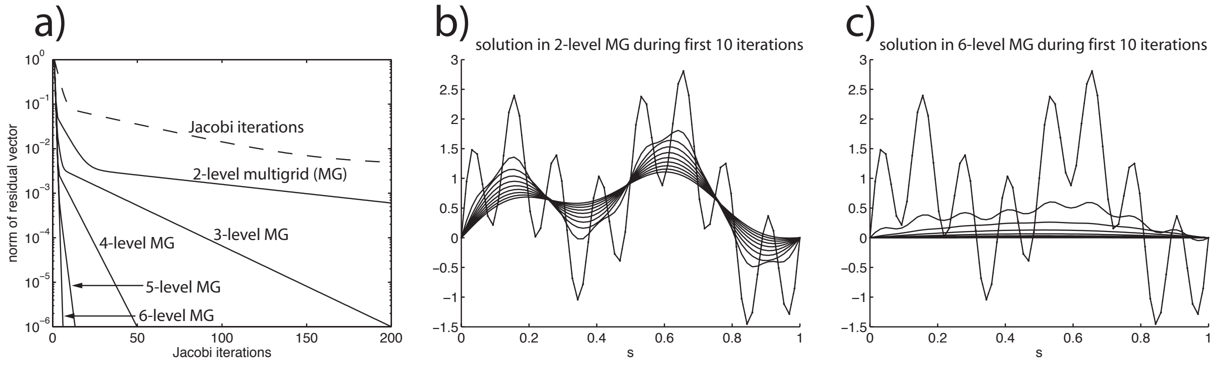


Figure 2.11: *Convergence of the multigrid method for the same sample problem as in Fig. 2.9. multigrid methods using 2, 3, 4, 5, and 6 levels are shown. On each level, two relaxation steps with a $\omega = rac23$ are used. The number of nodes on each level are 5, 9, 17, 33, and 65, resp. (a) norm of the residual during the iterations for the five multigrid runs; the Jacobi convergence is shown again for a better comparison. (2) The solution (=error) during the first 10 iterations of the 2-level multigrid. Mid- and large-wave length part of the error remain. (c) The first 10 steps of the 6-level multigrid; all wave lengths of the error are removed within a few steps. This is also indicated by the convergence in (a): The 6-level MG has no kink in the residual convergence but converges at a constant rate.*

number of relaxations on each level and the damping factor ω have also a great influence on the convergence behavior. Tab. 2.3 summarizes a set of 1-D experiments (on the same problem), in which the number of multigrid levels, the number of relaxation on each level, and the weighting factor is varied. The results can be summarized as

1. the more MG levels used, the better the convergence

2. the number of MG cycles reduces by a factor 2 when using 2 relaxations instead of 1; using 3 relaxations instead of 2 reduces the iterations by another factor of 1.5

3. the optimum weighting factor depends on the number of MG levels, but is independent of the number of relaxations

4. the fewer MG levels employed, the higher the optimal ω

5. for the maximum number of MG levels, the optimal ω is in the range of 2/3 to 4/5 (in agreement with recommendations in Saad (2003) and Briggs et al. (2000))

The observations can be interpreted as follows: When few MG levels are employed, the long wave length part of the error is not reduced as efficiently. In this case, the algorithm is comparable to a standard Jacobi solver, which performs best for $\omega = 1$ (Jacobi iterations on the coarsest MG level have to solve for a long wave length error, for which the mesh is not suited). In experiments with 6 MG levels, all wave lengths of the error are equilibrated efficiently, because each error component is turned into a short wave length error on one of the meshes. In this case, two relaxations on each mesh are

Table 2.3: *Number of outermost MG iterations to solve the problem in Fig. 2.9 to reduce the norm of the residual by a factor of 1e-8. A maximum of 1000 iterations was allowed. Results for different damping factors ω are shown for 2-6 MG levels and 1-3 relaxations on each level.*

ω	0.5	2/3	.7	.8	.85	.9	.95	.97	.98	.99	1
1 relaxation											
2-MG levels	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
3-MG levels	1000	855	814	712	670	633	599	588	**585**	606	1000
4-MG levels	282	211	201	175	165	156	**151**	168	226	425	1000
5-MG levels	72	54	51	45	**43**	47	89	146	216	418	1000
6-MG levels	21	**16**	**16**	22	29	44	88	145	215	417	1000
2 relaxations											
2-MG levels	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
3-MG levels	571	428	408	357	336	317	301	295	**293**	304	1000
4-MG levels	142	106	101	88	83	79	**77**	85	114	214	1000
5-MG levels	37	28	26	23	**22**	24	45	74	109	210	1000
6-MG levels	12	**10**	**10**	12	16	23	45	73	108	210	1000
3 relaxations											
2-MG levels	1000	1000	1000	1000	969	915	867	849	840	832	1000
3-MG levels	382	287	273	239	225	213	202	198	**197**	204	1000
4-MG levels	96	72	69	60	57	54	**52**	57	77	143	1000
5-MG levels	26	20	19	**17**	**17**	18	31	50	73	141	1000
6-MG levels	11	**10**	**10**	**10**	12	16	30	49	73	141	1000

sufficient to approximate all error components. Any further relaxations start to reduce longer wave length components, but those are already taken care of on the next coarser level. Thus, there is almost no improvement when using more than 2 relaxations in the 6 level MG. When using ”too few” levels, the additional relaxation helps to reduces the long wave length error components, for which no appropriate mesh is available. However, these experiments perform more poorly than the ones that include all MG levels.

The set of experiments clearly shows the importance of having a good coarse mesh solution in order to achieve a good rate of convergence. However, in large-scale 2-D and 3-D models this would lead to a very large number of multigrid levels, which often is not practical (for example in unstructured meshes and unevenly shaped domains). A coarse mesh solution will also be less practical, if the problem defined on the fine grid has non-uniform material properties that cannot be captured by the coarsest grids. In this case the iterations on these coarse levels will actually solve a different problem so that the obtained error components might not help improve the solution on the fine mesh.

A way to circumvent these issues is to use a direct solver on the coarsest mesh that is employed. Direct solvers, which will be briefly introduced in the next section, solve the given problem exactly (except for round-off). Every component of the error corresponding to the restricted residual will be evaluated. Since this error will include all long wave lengths, no coarser mesh than the one on which the direct solver is located, is needed.

2.4.3 Direct solvers

Direct solvers are advantageous when the problem at hand is small enough so that the factorization of the coefficient matrix is not too expensive in either memory usage or computation time. In this case, a direct solver will be faster than an iterative solver, especially if multiple solutions are needed for the same coefficient matrix with different right-hand sides. I will show soon that the ”problem size” is not just a question of the number of unknowns but also a question of whether a 2-D or a 3-D problem needs to be solved. Another reason to choose a direct solver is if the condition number of the matrix to be inverted is very large, in which case any iterative solver will require a very large number of iterations.

All problems covered in this thesis require the solution of symmetric positive-definite matrix equations. For this kind of matrix the Cholesky factorization (e.g. Press et al., 1992) is usually the best choice. It is related to the better known LU-decomposition, which aims to decompose the coefficient matrix $\mathbf{A}$ into a lower triangular matrix $\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$ so that $\mathbf{L}\,\mathbf{U} = \mathbf{A}$. If such a decomposition can be computed, the matrix equation $\mathbf{A}x = b$ can be solved easily:

$$\mathbf{L}\,y = b \qquad \text{(quickly solved by forward substitution)} \tag{2.129}$$

$$\mathbf{U}\,x = y \qquad \text{(quickly solved by backward substitution)} \tag{2.130}$$

If matrix $\mathbf{A}$ is symmetric, a more memory efficient and less roundoff error sensitive variant of the LU-decomposition can be used. Here, $\mathbf{A}$ is decomposed into a single triangular matrix $\mathbf{L}$ such that $\mathbf{L}\,\mathbf{L}^T = \mathbf{A}$. The solution then uses the same forward-backward substitution as in the LU-decomposition.

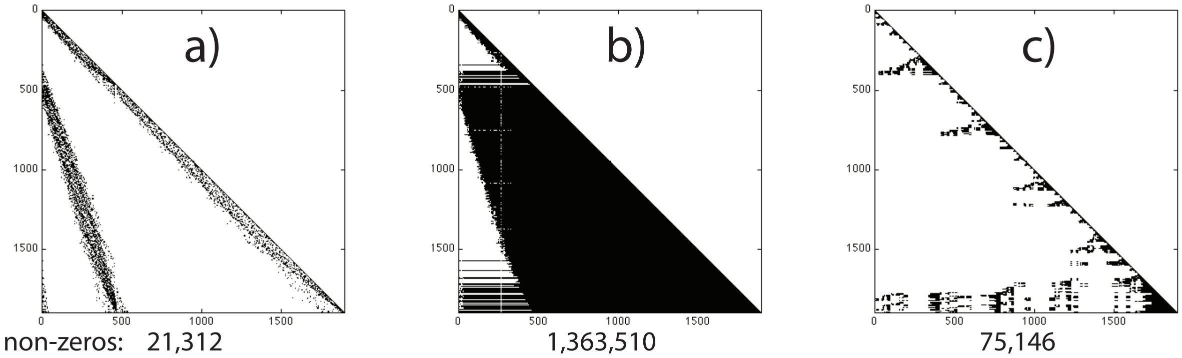


Figure 2.12: *Sparse pattern (locations of non-zeros) in the stiffness matrix for a small 2-D viscous flow problem using FEM with quadratic Taylor-Hood elements. (a) sparse pattern of the lower triangle of the symmetric stiffness matrix* $\mathbf{K}$*; (b) sparse pattern of the Cholesky factorization of* $\mathbf{K}$ *(64-times more entries than in* $\mathbf{K}$*); (c) sparse pattern of the Cholesky factorization of* $\mathbf{K}$ *when using a good variable permutation during the factorization (3.5-times more entries than in* $\mathbf{K}$*). The Cholesky factorization are done using the ”chol” command built into MATLAB)*

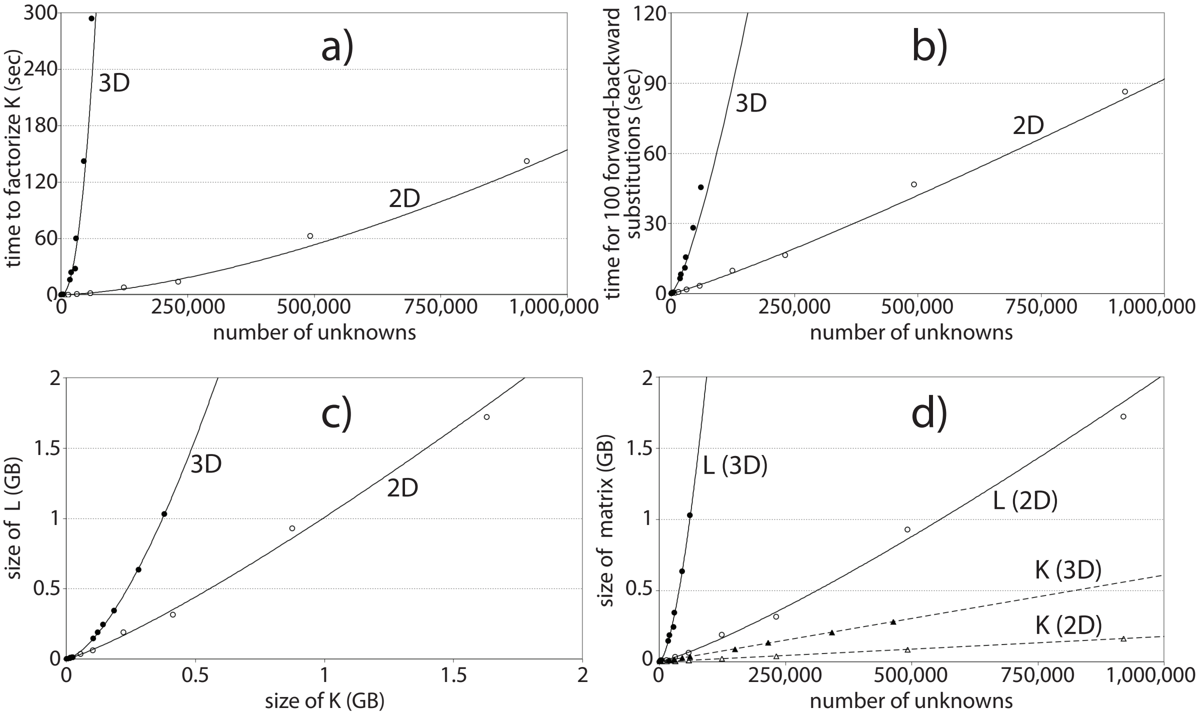


Figure 2.13: *Times to factorize stiffness matrix* **K** *and size of the resulting matrix* **L** *for viscous flow problems in 2-D and 3-D (Taylor-Hood elements, i.e. 6-node triangles and 10-node tetrahedra). Circles are measurements, lines are fitted and extrapolated. (a) Time to compute* **L** *as a function of the number unknowns. (b) Time for 100 forward-backward substitutions using* **L**. *(c) Memory storage required for* **L** *compared to the memory needed to store* **K**. *The coefficients for the exponential increase are ∼1.2 for 2-D and ∼1.5 for 3-D. (d) Memory needed to store* **K** *and* **L** *as a function of the number of unknowns. Memory storage always includes MATLAB's storage for indices pointing to the non-zero locations.*

There are two limiting criteria that restrict using the Cholesky factorization for every numerical problem that one is confronted with: the memory requirements to store **L** and the time to compute **L**. Both are related to each other. Fig. 2.12a shows the stiffness matrix **K** for a small 2-D viscous flow problem with 1,900 unknowns, for which the matrix has about 21,000 non-zero entries. Fig. 2.12b shows the factorized matrix **L**, which has 1.36 million entries and is 64-times larger in memory than **K**. The huge fill-in of non-zeros into the zero-regions of **K** can be significantly reduced with a fill-in-reducing permutation of the rows and columns of **K** during the factorization process (Fig. 2.12c). The resulting matrix **L** has about 75,000 entries and, thus, grew by only a factor of 3.5. While this increase seems to be tolerable, it becomes larger with the number of unknowns and with the number of non-zeros in the matrix that needs to be factorized. The latter has dramatic consequences for 3-D problems.

Fig. 2.13a shows the time required to factorize stiffness matrices of resulting from 2-D and 3-D viscous flow problems for different numbers of unknwons, when using an "optimal" permutation algorithm determined by *AMD* (Davies, 2006) or *METIS*[3]. The steep increase in computation time for the 3-D problem makes the Cholesky factorization impractical for problem sizes exceeding about 75,000 unknowns, while 2-D problems with one million and more unknowns are still solvable. A similar (but not as dramatic) discrepancy can be seen for the forward-backward substitution once the factorization has been accomplished (Fig. 2.13b; times for 100 solutions using **L**). The memory requirements to store **L** are shown in Fig. 2.13c–d in comparison to the size of the original matrix. Although not as critical as the computation time, the applicability of the Cholesky solver would be limited to 3-D problems with less than 100,000–150,000 unknowns, depending on the hardware. In 2-D, factorized matrices for problems with a million unknowns still fit into the memory of modern PCs. Unfortunately, 100,000 unknowns in 3-D is equivalent to a grid with 32x32x32 nodes, which is too coarse to solve almost any of the current geodynamic problems.

The reason for these difficulties in 3-D arise from denser matrices, i.e. matrices that have a higher percentage of non-zero content. Each non-zero entry in the stiffness matrix represents the connection between degrees of freedom (dof). In a 3-D volume more connections exist than in a 2-D plane, so that the number of non-zeros is much higher in 3-D for the same number of unknowns. In an unstructured 3-D finite element mesh with quadratic Taylor-Hood elements (i.e. tetrahedra), a vertex node can easily connect to 80 and more other nodes, and an edge node to 30 and more neighbors. The average number of connections was found to be around 40. In unstructured 2-D meshes with quadratic Taylor-Hood elements (triangles), the number of connections are about 20 (vertex nodes), 8 (edge nodes) and 12 on average. This leads to a larger percentage of fill-in during the factorization in 3-D, because the work of the permutation algorithm is hindered by the larger number of connections.

Variations of the Cholesky factorization exist that limit the amount of fill-in (e.g. incomplete Cholesky factorization) or even can avoid it (zero fill-in incomplete Cholesky factorization). However, these operations go along with a change in the physical meaning to the matrix, because information is lost. These factorizations can only be used as approximations, for instance for preconditioning purposes.

Although impractical to solve mid- to large-size 3-D problems, the Cholesky factorization can still be used to support the work of iterative solvers. In a multigrid algorithm, the number of unknowns reduces during each restriction, which makes direct solvers attractive for replacing the relaxations on the coarsest level. A direct solver at the coarsest level of a MG cycle has two advantages: (1) It solves exactly for the error, so that no coarser mesh is required beyond the level where the direct solver is positioned. (2) The factorization

[3] http://glaros.dtc.umn.edu/gkhome/views/metis

of the matrix is the most time-consuming part, while each forward-backward substitution is very fast in comparison (Fig. 2.13a–b). Since a solution for the matrix equation on the coarsest level will be required for each MG cycle, the efficiency of the direct solver increases the more frequently the factorized matrix is used.

2.5 Combining solvers for best performance

2.5.1 Strategy

The basic messages from the previous sections are

- The method of Conjugate Gradients (CG) is a very intelligent algorithm that requires efficient preconditioning to display its full potential (it works best with a good estimate for the error)
- The multigrid method (MG) can approximate all components of the error; its number of outer iterations is independent of the problem size if all wave lengths of the error are furnished with a suitable mesh; this is less practical for the coarsest levels in large scale problems and if varying material properties on the fine mesh prevents the coarser levels from capturing the problem at hand
- Direct solvers perform great if the problem size is small enough and if the factorized matrix can be used for several solutions involving different right-hand-side vectors

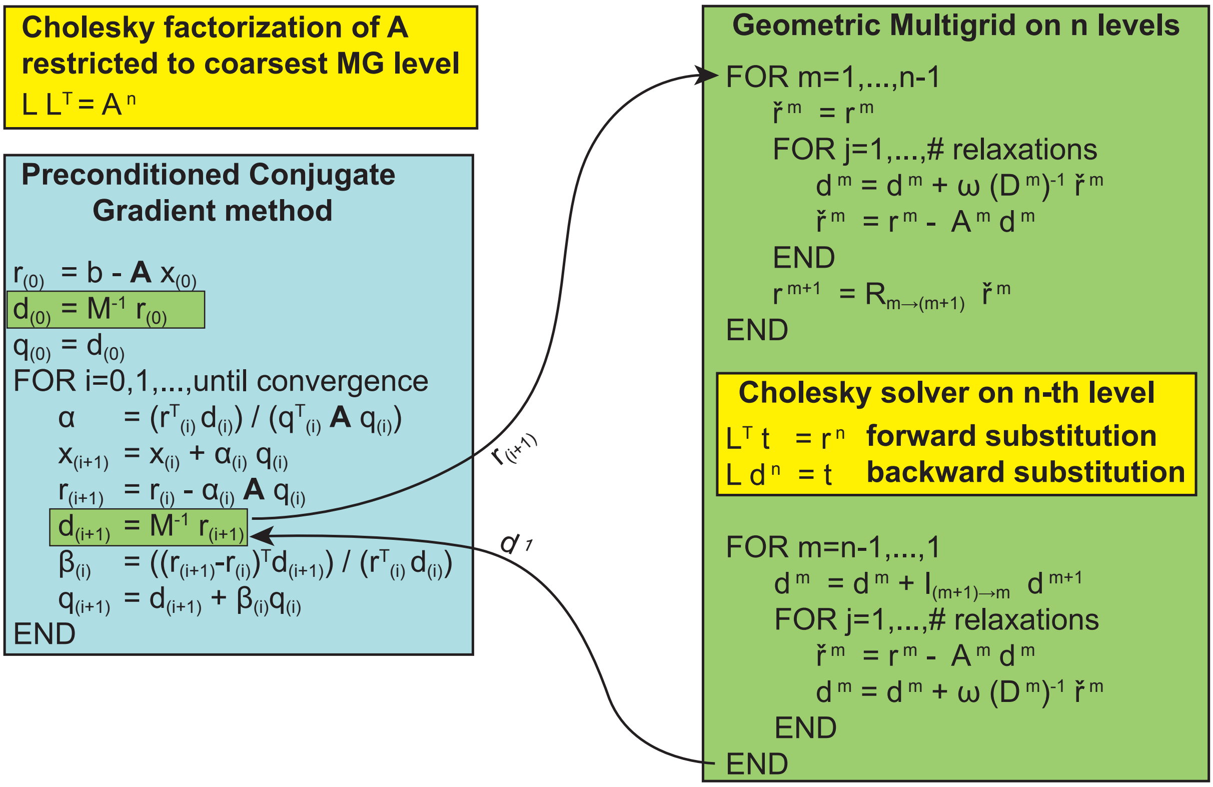
These results have motivated the combination of the three ”stand-alone” solvers to a single solution algorithm (Fig. 2.14) that is capable of solving large systems of equations efficiently. The outer solver is the CG algorithm in Eq. (2.124), with the difference that β is calculated differently (the reason for this choice is discussed below). The CG is preconditioned by a single V-cycle of a geometric multigrid solver (i.e. $\mathbf{M}^{-1}$ in Eq. (2.124) is the green box in Fig. 2.14). On the coarsest MG-level, a forward-backward substitution is performed using the factorized coarse-mesh approximation to matrix $\mathbf{A}$.

The calculation of β in the standard CG algorithm is the so-called *Fletcher-Reeves* formulation

$$\beta_{(i)} = \frac{r_{(i+1)}^T d_{(i+1)}}{r_{(i)}^T d_{(i)}} \tag{2.131}$$

It has been replaced by the *Polak-Ribière* formulation

$$\beta_{(i)} = \frac{\left(r_{(i+1)} - r_{(i)}\right)^T d_{(i+1)}}{r_{(i)}^T d_{(i)}} \tag{2.132}$$



Cholesky factorization of A restricted to coarsest MG level

```
$L L^T = A^n$
```

Preconditioned Conjugate Gradient method

```
$r_{(0)} = b - \mathbf{A}\, x_{(0)}$
$d_{(0)} = M^{-1} r_{(0)}$
$q_{(0)} = d_{(0)}$
FOR i=0,1,...,until convergence
    $\alpha = (r^T_{(i)} d_{(i)}) / (q^T_{(i)} \mathbf{A}\, q_{(i)})$
    $x_{(i+1)} = x_{(i)} + \alpha_{(i)} q_{(i)}$
    $r_{(i+1)} = r_{(i)} - \alpha_{(i)} \mathbf{A}\, q_{(i)}$
    $d_{(i+1)} = M^{-1} r_{(i+1)}$
    $\beta_{(i)} = ((r_{(i+1)}-r_{(i)})^T d_{(i+1)}) / (r^T_{(i)} d_{(i)})$
    $q_{(i+1)} = d_{(i+1)} + \beta_{(i)} q_{(i)}$
END
```

Geometric Multigrid on n levels

```
FOR m=1,...,n-1
    $\check{r}^m = r^m$
    FOR j=1,...,# relaxations
        $d^m = d^m + \omega (D^m)^{-1} \check{r}^m$
        $\check{r}^m = r^m - A^m d^m$
    END
    $r^{m+1} = R_{m\rightarrow(m+1)} \check{r}^m$
END
```

Cholesky solver on n-th level

```
$L^T t = r^n$    forward substitution
$L d^n = t$      backward substitution
```

```
FOR m=n-1,...,1
    $d^m = d^m + I_{(m+1)\rightarrow m}\, d^{m+1}$
    FOR j=1,...,# relaxations
        $\check{r}^m = r^m - A^m d^m$
        $d^m = d^m + \omega (D^m)^{-1} \check{r}^m$
    END
END
```

Figure 2.14: *The solution algorithm that is used to solve the velocity subproblem in the Stokes flow problem. A conjugate gradient algorithm (blue) is preconditioned by a single V-cycle of a geometric multigrid algorithm (green box). On its coarsest level, a Cholesky forward-backward substitution avoids the need for more coarser meshes. See text for details.*

This change is recommended, because CG "expects" $\mathbf{M}^{-1}$ to be the same operator during all its iterations. By using an inexact MG-solver (only one V-cycle), the operator between approximate error d and residual r is not exactly the same in each iteration. The Polak-Ribière formula, usually used in combination with the Nonlinear Conjugate Gradient Method (Shewchuk, 1994), is more robust in this regard without having any noticeable disadvantages apart from storing the additional vector $r_{(i)}$ that normally gets overwritten by $r_{(i+1)}$.

On all MG-levels but the coarsest one, two Jacobi relaxations are done before restricting r (on the downward path) and after interpolating e (on the upward path). As shown in Tab. 2.3, the best damping factor in the 1-D problem is $2/3 \le \omega \le 4/5$. In 2-D, but especially in 3-D, a better smoothing result is often achieved by using a different damping factor in each of the two relaxation sweeps. The best choice for ω depends on the particular matrix equation to be solved, but a damping factor of $\omega = 1/3$ in the first and $\omega = 1$ in the second relaxation seems to be a good choice for a wide range of viscous Stokes flow problems (see also Fig. 2.30 on p. 99). In order to keep the entire multigrid algorithm symmetric, the order of these factors is reversed on either the upward or the downward path – no difference in performance was found between the two choices.

A coarse version of $\mathbf{A}$ is required on every level during the relaxations. These matrices are calculated recursively using the restriction and interpolation operators:

```
FOR m=2,...,n
        $\mathbf{A}^m = R_{(m-1)\to m}\,\mathbf{A}^{m-1}\,I_{m\to(m-1)}$        (2.133)
END
```

The restriction operator $R_{(m-1)\to m}$ is chosen to be the transpose of $I_{m\to(m-1)}$, so that every matrix constructed by (2.133) is symmetric.

The assembly of coarser $\mathbf{A}$ matrices, as opposed to the above restriction, has also been tested. To do so, the average viscosity over patches of elements that are nested within a single element on the next coarser mesh, is used to construct coarse element matrices, which are then assembled to the $\mathbf{A}$ matrix on the coarse mesh. However, the assembled coarse matrices are not as good an approximation to the original $\mathbf{A}$ and result in lower quality error approximations in the MG-cycle (i.e. more CG iterations). The reason is that a matrix restricted using (2.133) can better approximate spatial viscosity variations within each element patch, whereas the assembled coarse $\mathbf{A}$ cannot, because it is based on a single average viscosity in each patch.

Another reason for using the mesh transfer operators to calculate the coarser matrices can be seen from the equations: Solving $\mathbf{A}e = r$ using a coarse-mesh approximation e^m to the error e on the fine mesh can be written as

$$\mathbf{A}\,e = r \tag{2.134a}$$

$$\mathbf{A}(I_{m\to 1}\,e^m) \approx r \tag{2.134b}$$

$$(R_{1\to m}\,\mathbf{A}\,I_{m\to 1})\,e^m \approx R_{1\to m}\,r \tag{2.134c}$$

Equation (2.134b) does not represent a symmetric operation, whereas (2.134c) does if $R_{1\to m} = I^T_{m\to 1}$. Therefore restricting $\mathbf{A}$ using (2.133) is a symmetric operation as required by CG for the operator $\mathbf{M}^{-1}$.

Multigrid is more often used in combination with finite difference codes, where a restriction to the next coarser level is done by simply using the value at every second grid point in each spatial direction. Restriction in the FDM is equivalent to skipping grid points. The interpolation operator copies the coarse grid solution to the grid points that are in the same location on the finer mesh, and linearly interpolates the values at the grid points in-between. This restriction/interpolation procedure is a non-symmetric operation, because $R_{(m-1)\to m} \neq (I_{m\to(m-1)})^T$. This represents no problem in a pure multigrid solver, but would be an issue if the result was used to precondition a CG algorithm.

Multigrid in combination with finite elements is different, because the FEM evaluates coefficients for the trial solutions rather than nodal solution values (see section 2.1, p. 13). The trial solutions that approximate the error on a coarser mesh have to be transferred to the fine mesh, so that they can update (improve) the coefficients of the trial solutions on the fine mesh. For this reason, shape functions on a coarse level have to be expressible in terms of fine mesh shape functions. This can be achieved by making fine mesh elements be nested within elements on the coarser mesh, as will be shown in the next section (cf. Fig. 2.15 and 2.16). Providing this property ensures that the coarse mesh approximation can be converted into a correction to the trial solution coefficients on the fine mesh (this is discussed in detail in Briggs et al. (2000), chapter 10). If this property is not given, the multigrid algorithm might still converge, but the introduced conceptual error might cause a sub-optimal convergence rate. Briggs et al. (2000) also show that the construction of the interpolation operator, as well as choosing its transpose for the restriction operator, emerges naturally in the Galerkin finite element formulation. Restriction and interpolation in the FEM is therefore always a symmetric operation and allows us to use multigrid to precondition CG.

The number of MG-levels depends on (1) the problem size (number of unknowns on the finest mesh) and (2) the number of unknowns for which the Cholesky solver can be used efficiently. In 3-D problems, the limit for the Cholesky solver is about 30,000–40,000 unknowns, whereas in 2-D problems 800,000 unknowns and more are possible. Depending on the resolution required on the finest mesh, the number of MG-levels is usually between 3 and 5 in the applications presented in this thesis (see Chapter 3 and Chapter 4).

2.5.2 Implementation and parallelization

Generation of multigrid meshes

The Taylor-Hood elements are split recursively to generate the multigrid meshes (Fig. 2.15 and Fig. 2.16). The software *GiD* (http://gid.cimne.upc, Version 8, 2007) is used to generate the coarsest mesh with a number of nodes that will allow an efficient factorization of the associated stiffness matrix. In 2-D each 6-node triangle is split into four elements, each of which has one quarter of the area of the ”parent”-element. The vertex nodes of the new elements are the six nodes of the parent element, whereas new edge nodes have to be generated after the split. Depending on the element connectivity within the unstructured meshes, this leads to an increase in the number of nodes by a factor of 3.5-4 for each additional multigrid level. The angles in all elements are preserved during the splits so that the mesh quality is independent of the number of MG-levels (i.e. no flattening of elements during the recursive splitting procedure). The same strategy is used in 3-D

(Fig. 2.16): Each 10-node tetrahedron is split into 8 tetrahedra, each of which has 1/8 of the parent element's volume. Here the number of nodes increases by a factor of about 7.3-8 for each new MG-level.

The nodes that are generated for every new mesh, are appended to the list of existing nodes. Consequently, nodes with the same number are in the same place on all MG meshes, which has some advantages for the book-keeping (nodes in Fig. 2.15a are in the same position as in Fig. 2.15b). The resulting wide-spread, non-zero entries in the stiffness matrix have been checked to have no negative influence on the speed of matrix-vector multiplications in MATLAB.

Generation of subdomains

When running in parallel, the coarsest mesh is divided into n_{SD} non-overlapping regions, where n_{SD} is the number of subdomains (SD). Fig. 2.17 shows an example for a small 2-D mesh. For splitting the mesh, I have developed an algorithm that proceeds similarly to a nested dissection algorithm. First, the domain is cut into two halves such that each part has the same number of elements. If, for instance, the split is perpendicular to the x direction, the center coordinates of all elements are sorted such that the x-coordinate increases. Elements in the first half of this list are assigned to one SD, those in the second half to the other. This logic is repeated within the two SDs that have been created and in all following ones, until the required number of subdomains has been reached.

There are multiple choices for the spatial direction of all bisections. The best sequence of bisections is one that minimizes (1) the number of shared nodes (i.e. minimum overhead)

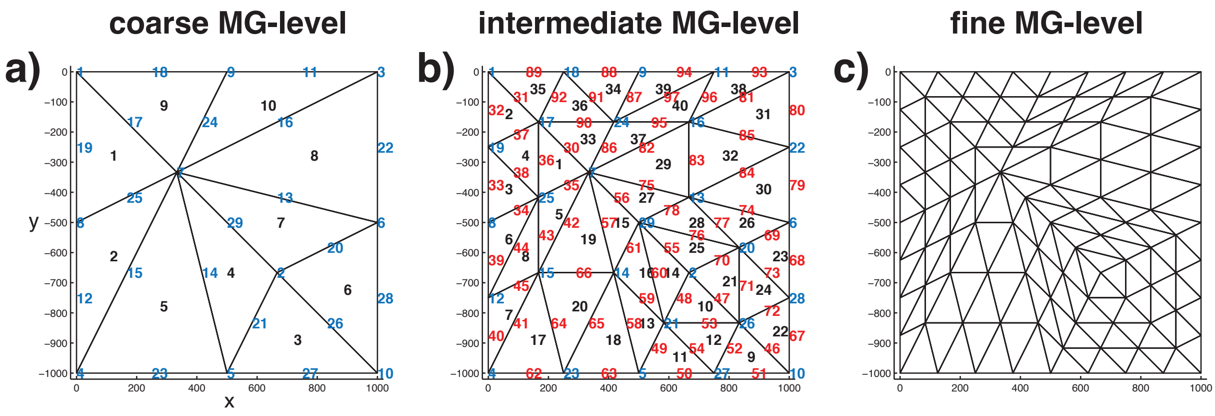


Figure 2.15: *Generation of multigrid meshes in 2-D: Starting with the coarsest mesh (a), a mesh for the next finer MG-level is calculated by splitting each element into four triangles and generating new edge nodes (b). Another split leads to a 3rd MG-level (c). Numbers are the nodes on the coarsest mesh (blue), new edge nodes on the intermediate mesh (red), and element numbers (black). Nodes with the same number remain in the same location on every MG-level.*

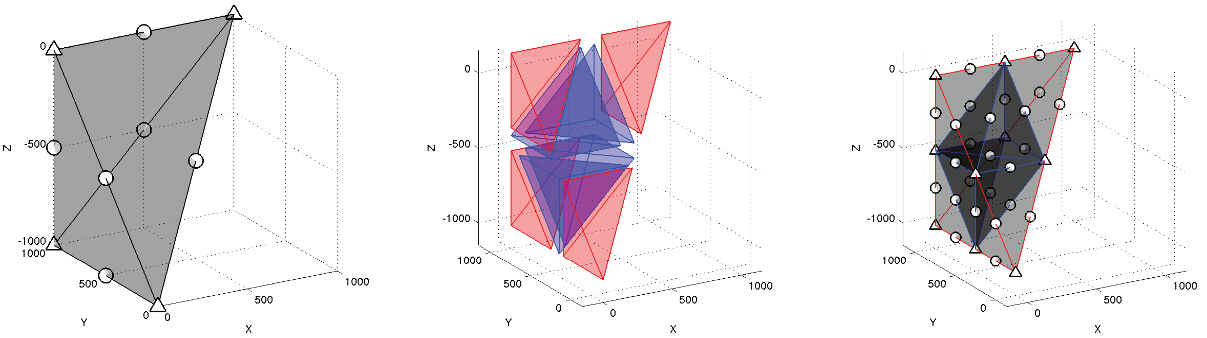


Figure 2.16: *Generation of multigrid meshes in 3-D: Each 10-node tetrahedron (a) has 4 vertex nodes (drawn as triangles) and 6 edge (drawn as circles). The vertices are re-connected to form eight new elements (b). New edge nodes are generated afterwards to obtain eight quadratic-order Taylor-Hood elements (c).*

as well as (2) the maximum number of neighbors of the SDs (i.e. minimum number of messages during communication). The optimum sequence depends on the structure of the mesh, particularly on the location of regions with higher spatial resolution, and on the

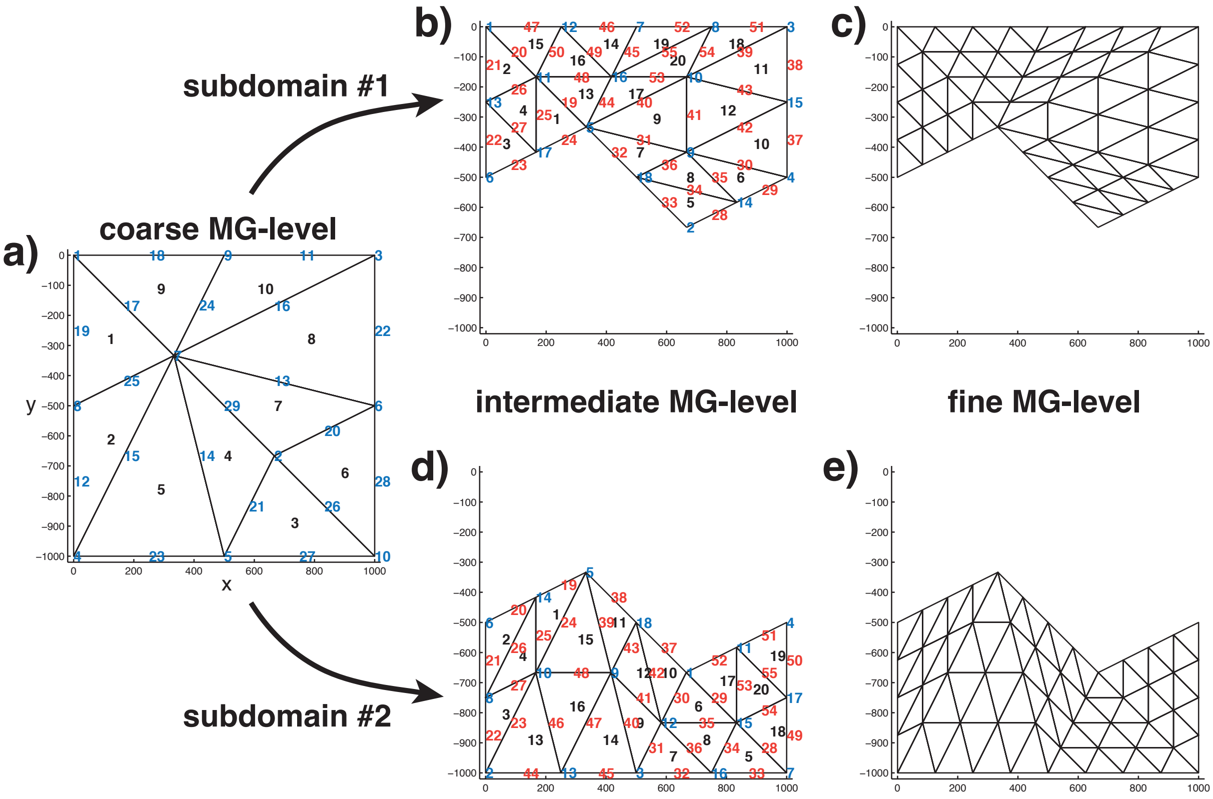


Figure 2.17: *Example of a small 2-D mesh (a) that is split into two subdomains. Each subdomain recursively splits its part of the mesh to generate the multigrid levels: (b-c) for subdomain 1 and (d-e) for subdomain 2, resp. All subdomains keep mesh (a), as it is needed for the direct solve over the entire domain on the coarsest MG-level.*

Table 2.4: *Splitting of a cube-shaped mesh with 20,000 nodes into 16 subdomains. I have developed an algorithm that calculates all possible combinations of bisections in the three spatial directions. The configuration with the smallest number of shared nodes and the fewest maximum number of neighbors is selected. Here, the split 4x1x4 was chosen as a compromise: 4 subdomains in the x-direction, 4 in the z-direction and only 1 (i.e. no split) in the y-direction.*

# splits in (x,y,z)	# shared nodes	% shared nodes	max # neighbors
1 x 1 x16	15346	75.86	6
1 x 2 x 8	8274	40.90	6
1 x 4 x 4	6440	31.83	8
1 x 8 x 2	8209	40.58	7
1 x16 x 1	15111	74.70	6
2 x 1 x 8	8190	40.48	6
2 x 2 x 4	5320	26.30	11
2 x 4 x 2	5289	26.14	11
2 x 8 x 1	8185	40.46	6
4 x 1 x 4	6308	31.18	8
4 x 2 x 2	5289	26.14	11
4 x 4 x 1	6333	31.31	8
8 x 1 x 2	8222	40.64	6
8 x 2 x 1	8202	40.54	6
16 x 1 x 1	15042	74.36	6

spatial extension of the domain. To find a good subdomain configuration, I use a straightforward approach: All possible combinations of bisections in all spatial dimensions that lead to the required number of SDs are tested, and the best decomposition is selected. The calculation of these recursive bisections has been vectorized and operates very fast, even for a larger number of subdomains. A result is obtained within few seconds.

Tab. 2.4 shows an example, in which a 3-D cube-shaped domain with 20,000 nodes is divided into 16 subdomains. Since both of the above criteria (minimum overhead and minimum number of neighbors) are important for the performance of the code, weighting factors are used to define the relative importance of the two aspects. When running on a hardware with comparably slow network connections, for instance, the minimization of the communication may have priority.

This simple method has the advantage of providing a number of nodes in each subdomain that is close to the optimum. Alternatively, the nested dissection algorithm *nesdis* can be used, which is built into *CHOLMOD* (Chen et al., 2006). While providing good domain decompositions in 2-D, I find suboptimal results from *nesdis* for 3-D meshes: The load balance between SDs is not optimal (Fig. 2.18) and the maximum number of neighbors (defining the number of communication sequences) is often larger than when using the self-developed algorithm.

Once the subdomains have been generated, a communication scheme is constructed to allow a pairwise (send-receive) message exchange between the subdomains without

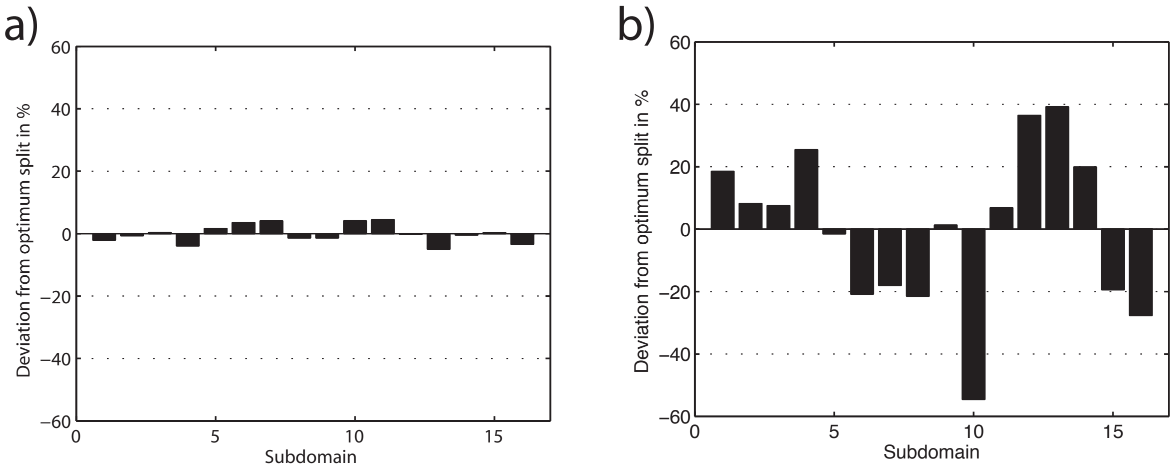


Figure 2.18: *(a) Load balance (here: number of nodes) for the subdomain configuration "4x1x4" in Tab. 2.4. The optimum is the zero line, in which case a subdomain has exactly* 1/16*-th of the total node number. (b) Load balance after using nesdis to calculate the 16 subdomains.*

producing a deadlock. This task is more complicated when using unstructured meshes, because no systematic order for communication can be defined a priori.[4] The algorithm for constructing this scheme has been developed by myself, and it ensures that (1) pairs of SDs communicate in every step, and (2) a minimum number of communication steps is required. The achievable minimum is defined by the SD with the largest number of neighbors. This communication scheme is calculated once in the beginning and used for all message exchange, except for broadcasts that are received not just by neighbors but by all subdomains.

The scheme corresponding to the best decomposition (4x1x4) in Tab. 2.4 is shown in Tab. 2.5. Each row in the latter table defines the communication partner for each of the 16 SDs (a zero means no communication during a squence). After 8 communications, all SDs have talked to all their neighbors.

The subdomains generated by mesh-splitting overlap only at the nodes that are shared by neighbors, i.e. they are not surrounded by so-called halos or ghost-nodes. All information that is required for a parallel run, are the following for each subdomain SD:

1. the column in the communication scheme (Tab. 2.5) belonging to the subdomain; it contains all neighbors and the order how to communicate to them

2. a list of nodes that are shared with each neighboring SD

[4] In structured grids the following logic can be used without producing a deadlock: send-receive communication between SDs in the (1) positive x-direction; (2) negative x-direction; (3) positive y-direction etc.

Table 2.5: *Communication scheme for the subdomain configuration "4x1x4" in Tab. 2.4. A pair-wise (send-receive) communication between each subdomain and all its neighbors is conducted during 8 cycles, which is equal to the maximum number of neighbors of a subdomain. A zero means that the subdomain has no communication in this cycle.*

comm-	subdomain															
cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	2	1	4	3	6	5	11	12	13	14	7	8	9	10	16	15
2	5	3	2	8	1	7	6	4	10	9	15	16	14	13	11	12
3	0	5	7	0	2	10	3	0	14	6	12	11	0	9	0	0
4	0	6	0	0	9	2	8	7	5	11	10	15	0	0	12	0
5	6	7	0	0	10	1	2	11	0	5	8	0	0	15	14	0
6	0	0	8	7	0	11	4	3	0	15	6	0	0	0	10	0
7	0	0	6	0	0	3	10	0	0	7	16	0	0	0	0	11
8	0	0	0	0	0	9	12	0	6	13	14	7	10	11	0	0

3. a list of "unique" nodes, that is, a list that contains each node within the domain only once; in other words, if all SDs would merge their unique nodes, no node in the entire domain would be either counted twice or missing

In case of the viscous flow problem, equivalent lists containing degrees of freedom (dofs) for the velocity unknowns are required. Using this information, the conjugate gradient and multigrid algorithm in the combined solver (Fig. 2.14) can run in parallel as described in the next paragraph.

Parallelization of the iterative solver

In a parallel run, matrix **A** is assembled separately in each SD so that some dof-to-dof connections in **A** are missing or incomplete: For instance, the unknowns associated with node 17 in subdomain SD 1 in Fig. 2.15 are actually the same as the ones associated with node 14 in SD 2. Since SD 2 does not know node 11 in SD 1, it has an incomplete equation for the unknowns associated with node 14. Missing connections cannot be corrected for without introducing subdomain halos and thereby increasing the dimensions of the matrix. We seek to avoid this, for reasons discussed below.

The connection between nodes 14 and 19 in SD 2 is described by $\mathbf{A}_2(14, 19)$ and $\mathbf{A}_2(19, 14)$ (subscripts indicate the number of the subdomain). These entries are incomplete, because the relation between 17 and 24 in SD 1 (components $\mathbf{A}_1(17, 24)$ and $\mathbf{A}_1(24, 17)$) also describes this connection and would have to be added. These incomplete components at subdomain boundaries can be corrected by summing up the shared components, so that all subdomains have the complete values afterwards. In case of a vector this step is simple, since neighboring subdomains have an ordered list of nodes they share with each neighbor.

Two mathematical operations in a CG algorithm require a special treatment in parallel: the inner products (e.g. calculation of β) and the matrix-vector multiplications (e.g. calculation of α). The inner product of two vectors that are separated into overlapping parts, can be done in three steps:

1. sum up the shared components of each vector so that all components are complete
2. calculate the inner product in each SD but only for the unique components
3. broadcast the result and accumulate (requires broadcasting a scalar value)

Parallel matrix-vector multiplications can be done although the matrices have incomplete components on the boundary:

1. multiply the incomplete matrix to a complete vector in each subdomain
2. sum up the resulting incomplete vector at all subdomain boundaries to obtain the result

With these modifications, the CG algorithm running in parallel produces the exact same values (in all iterations) as the serial CG for the same equation (except for round-off). The parallel run, however, requires a communication between neighboring subdomains after every inner product and every matrix-vector multiplication within a CG iteration.

The above concept is also used in the MG solver, where the new residual has to be summed up at SD boundaries before continuing, because it results from the multiplication of a complete vector by the incomplete matrix $\mathbf{A}^m$. The restriction process requires special attention: The vector to be restricted has to be summed up at SD boundaries and set to zero at components that are not in the list of the unique components. Otherwise the restricted values can receive multiple contributions from the same component on SD boundaries. The interpolation process does not require this step. No modification needs to be done to the restriction and interpolation operators itself. All matrices $\mathbf{A}^m$ (where $m = 1, ..., n-1$ and n is the number of MG-levels) are restricted inside each subdomain using (2.133). They are not corrected at subdomain boundaries, but remain incomplete on all MG-levels except the coarsest one.

On the coarsest (n-th) level, a Cholesky forward-backward substitution is used to obtain a global solution. This global solution requires that $\mathbf{L}$ is the factorization of the global matrix $\mathbf{A}^n$. However, $\mathbf{A}^n$ has to be calculated by restricting $\mathbf{A}^{n-1}$, which only exists in subdomains. Thus, all subdomain matrices $\mathbf{A}^n$ are merged (by superposition) after the restriction to form the global counterpart, which is very similar to the element assembly procedure in that the global (domain) dofs of the components of each subdomain matrix have to be known. This information is stored in a second connectivity matrix, pointing from SD dofs to domain dofs. Given this pointer for each subdomain, the global matrix is formed by superposition of the subdomain pieces.

The calculation of the global $\mathbf{A}^n$ by each subdomain would require the broadcast of each SDs $\mathbf{A}^n$, i.e. $n_{SD} \cdot (n_{SD} - 1)$ messages containing matrices. I improved the performance of this part of the code by calculating only one Cholesky factorization in a multiprocessor hardware with shared memory (SMP; Symmetric MultiProcessing) rather than on each CPU. This allows to take advantage of the SMP-parallelization of MATLAB's Cholesky factorization and also reduces the number of messages, because only one CPU in each SMP hardware has to receive the SD matrices.

For illustration, consider a run with 32 CPUs, conducted on 4 SMP machines with 8 CPUs each: One CPU in each SMP machine receives and superimposes all subdomain $\mathbf{A}^n$-matrices (this requires $4 \cdot (n_{SD} - 1)$ messages). The factorization of the merged $\mathbf{A}^n$ uses the power of all 8 CPUs, and $\mathbf{L}$ is then sent to the other 7 CPUs within the SMP system, which does not cause network traffic.

Once the factorized matrix is available for all subdomains, only the SD-parts of the residual vectors in the MG cycles have to be merged on the coarsest level to form a global RHS for the forward substitution. After the global (coarse) error has been calculated, only the part belonging to each subdomain is interpolated to the next finer MG-level, where all operations continue as outlined above. I do not see any improvement when using the above strategy to calculate only one forward-backward substitution per SMP and distributing the results. The computations for solving the triangular system cost much less than those required for the factorization, so that the time required for distributing the results outweighs what is gained from the parallel computation.

Some tests have been conducted with an iterative (CG) solver on the coarsest level, but I found a much better performance of the Cholesky solver, because its forward-backward substitution is very fast once $\mathbf{L}$ has been calculated. Further tests using parallel direct solvers that run on distributed memory systems (e.g. multi frontal algorithm ”MUMPS” (Amestoy et al., 2003); http://graal.ens-lyon.fr/MUMPS) or combinations of direct and iterative solvers are planned.

Comparison to other parallelization techniques

Alternative parallelization methods can be grouped into those that require a halo or ghost nodes around each subdomain and those that do not. Halos in the former group typically extend over a distance of one element so that the equations for the unknowns on the SD boundary can be formulated. That is, the values at the outermost nodes in the halos (so-called ghost nodes) are fixed and serve as boundary conditions for each of the subdomain solutions. A solution within each subdomain is calculated independently, and then matched (averaged) in the overlapping regions. It usually requires a certain number of subdomain solutions, until the SD solutions match across subdomains so that a continuous global solution is obtained.

This method has the advantage of significantly reducing the number of messages, because communication is only required after a SD solution has been derived. However, using halos also has some disadvantages:

1. Halos lead to a larger total problem size, because not only SD boundaries but also the nodes within the halos (i.e. the ghost nodes) are shared duplicates.
2. It is more difficult to achieve a good load balance, which is not necessarily equivalent to having the same number of unknowns in each subdomain. If, for instance, one subdomain has to solve a "harder" part of the global problem than others (e.g. a strong viscosity contrast that leads to a larger condition number of one subdomain's **A**), this subdomain will need more iterations until convergence.
3. The halo-method can converge slowly (or even fail to converge), if a hard part of the problem happens to be at the boundary between two SDs (i.e. within the halo). In this case, no matching solution may be obtained for many iterations, because none of the subdomains can completely cover this critical region.

None of the above problems arises in the method chosen here: After the summation at subdomain boundaries, all variables (e.g. residuals, search directions, solution vectors, etc.) are identical to the ones in a serial run. Thus, all SDs are solving the global problem rather than several sub-problems. The subdomain distribution and the problem geometry are therefore independent and have no influence on the convergence rate, which makes the halo-free method very robust. Furthermore, the load balance in this method is equivalent to the number of unknowns (i.e. the size of matrices and vectors), so that an almost perfect load balance can be achieved.

Using the halo-free method has also advantages during developing and debugging the numerical code, because variable values during all CG iterations (e.g. step size α or search direction q) have to be identical in serial and parallel runs, if they are summed up correctly.

For completeness I would like to mention that alternative halo-free methods exist as well. For instance, a subdomain solution can be obtained by defining alternating Dirichlet and Neumann boundary conditions at the nodes on the SD boundary. The solution value that is calculated where Neumann boundary conditions are imposed, becomes a Dirichlet boundary condition in the next iteration. Accordingly, the solution value is now calculated at nodes where Dirichlet boundary conditions have been imposed in the previous iteration. A comprehensive overview on parallelization methods is given in (Smith et al., 2004).

2.5.3 Performance

In this section the performance of the combined solver presented in Fig. 2.14 is compared against the performance of other standard iterative solvers. Given the pressure solution, the velocity solution u for a so-called ”sinker problem” (factor 10^4 viscosity contrast, see p. 84 for details) is calculated by solving the matrix equation $\mathbf{K}u = F$, where $\mathbf{K}$ is the stiffness matrix and vector F contains body forces, pressure forces, and boundary conditions (see Eq. (2.85) on p. 34). This problem is solved in 2-D (Fig. 2.19a,b; about 109,000 unknowns) as well as in 3-D (Fig. 2.19c,d; about 834,000 unknowns). The iterations of the solvers are stopped once the residual norm has been reduced below a defined tolerance. The evolution of the residual norm during the iterations and as a function of computation time are shown. Five solution algorithms are compared here:

- CG preconditioned by diagonal scaling (Jacobi preconditioner, solid black line)
  This simplest preconditioning technique requires many iterations. The preconditioning costs are very low but in each iteration a matrix-vector multiplication is required, which (for large problems) slows down the solver.

- CG with symmetric (forward-backward) Gauss-Seidel preconditioner (dashed black)
  Although reducing the number of iterations by a factor of ∼2.5 compared to the Jacobi-preconditioned CG, this solver is only few seconds faster than the Jacobi-preconditioned CG, because the preconditioning comes at a higher computational cost and does not vectorize as well as the Jacobi method.

- CG preconditioned by zero fill-in incomplete Cholesky factorization[5] (green)
  This solver performs similar to the Gauss-Seidel preconditioned CG. However, it has the disadvantage that $\tilde{\mathbf{L}}$ has to be calculated first, which consumes a significant amount of time in 2-D (see Tab. 2.6) and is impossible to calculate in 3-D (which is why this solver is not shown in the 3-D comparison).

[5] The incomplete Cholesky factorization calculates a matrix $\tilde{\mathbf{L}}$, which is an approximation to matrix $\mathbf{L}$ that would be obtained from a ”normal” Cholesky factorization (see p. 57). Hence, $\tilde{\mathbf{L}}\tilde{\mathbf{L}}^T \approx \mathbf{K}$. The quality of $\tilde{\mathbf{L}}$ depends on the degree of fill-in of non-zeros that is allowed during the factorization. Common variations are the ”zero fill-in” factorization ($\tilde{\mathbf{L}}$ and $\mathbf{K}$ have the same sparse patterns) and factorizations that define a drop level: New entries smaller than this threshold are ignored. Using forward-backward substitutions, $\tilde{\mathbf{L}}$ can be used to approximate $\mathbf{K}^{-1}$.

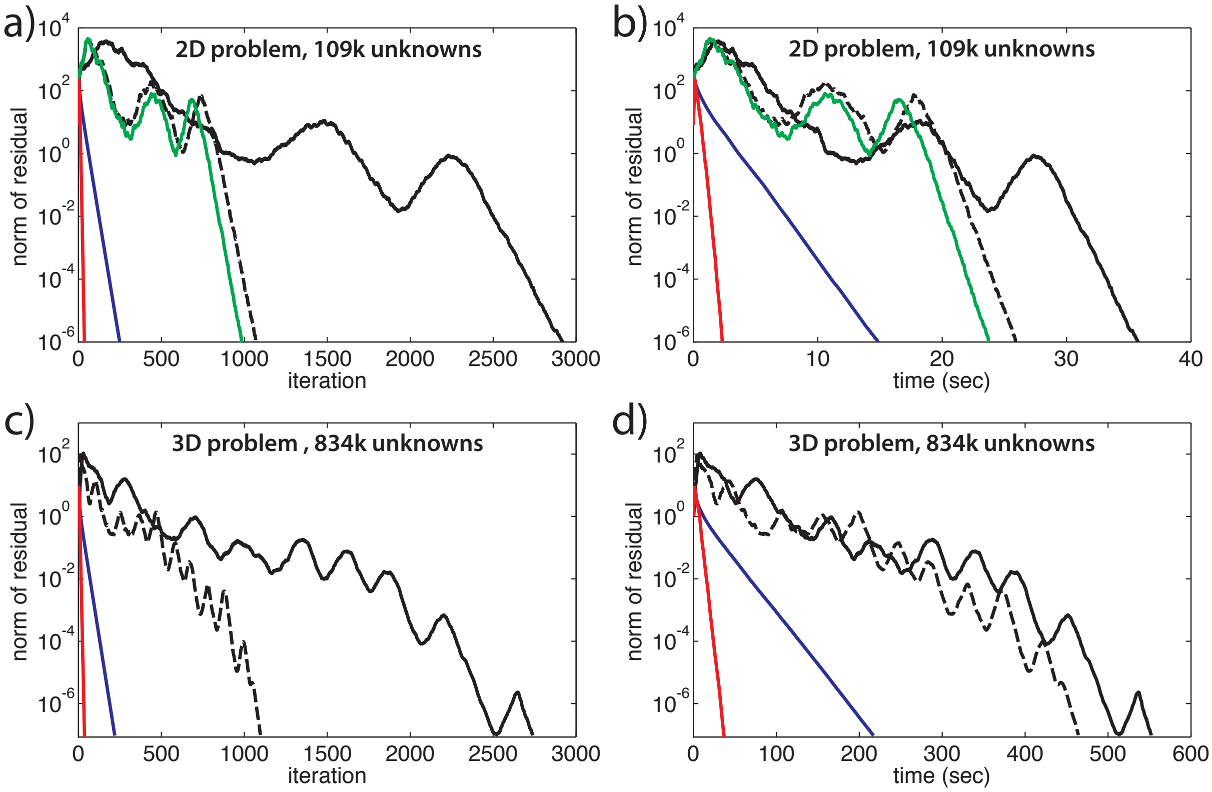


Figure 2.19: *Performance of the combined solver (algorithm in Fig. 2.14) in comparison to other iterative solvers. The equation solved is part of a so-called sinker problem (see p. 84) with a viscosity contrast of* 10^4*. A 2-D problem (panels a and b) and a 3-D problem (panels c and d) are solved. The evolution of the norm of the residual vector is shown in each iteration and as a function of computational time. The iterative solvers are: CG with Jacobi preconditioner (solid black), CG with symmetric Gauss-Seidel preconditioner (black dashed), CG preconditioned by a zero fill-in incomplete Cholesky factorization (green), MG algorithm (four levels in 2-D, 3 levels in 3-D; blue), and a CG algorithm using a single V-cycle of the same MG algorithm for preconditioning. Hardware: Intel Xeon 2.8 GHz, 8 MB Cache, 40 GB memory. Software: MATLAB R2009b (64-bit), SuiteSparse (http://www.cise.ufl.edu/research/sparse/SuiteSparse). See text and Tab. 2.6 for more information.*

- MG with a Cholesky direct solver on the coarsest mesh (blue)
  Four MG levels are used in the 2-D problem and three MG-levels in 3-D. This method requires much fewer iterations than the above CG algorithms. While the CG convergence shows the typical ups and downs, the MG always converges along a straight line (if implemented properly).

- CG preconditioned by a single V-cycle of the above MG algorithm (red)
  In this combined solver, the CG takes major advantage of the comprehensive error estimate provided by the single MG-cycle. Compared to the pure MG algorithm the number of iterations required to solve the problem is reduced by a factor of 7.2 (2-D) and 5.9 (3-D). The computation time is reduced by about the same factor, because the computational work is essentially the same as in the MG-solver.

Table 2.6: *Performance of the MG preconditioned CG in comparison to other iterative solvers (see Fig. 2.19 for convergence behavior). The startup times include:* [1]*extracting the diagonal of* **K**, [2]*extracting the diagonal and the lower triangular part of* **K**, [3]*calculating the zero fill-in factorization of matrix* **K** *using MATLAB's "cholinc", and* [4]*restriction of matrix* **K** *to coarser MG levels and factorizing* **K** *on the coarsest mesh.*

Solver	startup (sec)	solving (sec)	iterations
2-D sinker problem, 109k unknowns			
CG(Jacobi)	$< 0.1^1$	35.9	2929
CG(Gauss-Seidel)	$< 0.1^2$	26.0	1071
CG(incChol-0)	403^3	23.8	986
MG-4	0.3^4	15.1	251
CG(MG-4)	0.3^4	2.4	35
3-D sinker problem, 834k unknowns			
CG(Jacobi)	$< 0.5^1$	552.2	2739
CG(Gauss-Seidel)	$< 0.5^2$	464.3	1097
MG-3	9.2^4	217.2	219
CG(MG-3)	9.2^4	37.7	37

In large 3-D problems, most of the computational work is consumed by multiplying matrix **K** to a vector. In case of the multigrid algorithm, only matrix-vector multiplications on the finest mesh are important as the problem size rapidly decreases towards coarser meshes. Therefore using MG to precondition a CG algorithm does not significantly increase the computational work per iteration compared to a standard MG algorithm: In each iteration of both methods a new residual needs to be calculated (which requires one matrix-vector multiplication) and relaxations have to be performed (in the example shown here: two relaxations before restricting the residual and two after interpolating the error). Hence, the reduction in number of iterations immediately translates to a faster computation (cf. Tab. 2.6).

The fast convergence of the MG preconditioned CG algorithm is of greatest importance for all applications of the geodynamic code, because the example illustrated in Fig. 2.19 only represents an update of the velocity solution during the pressure iterations (see the next chapter on velocity-pressure formulations) — the "$\mathbf{K}u = F$"-problem has to be solved repeatedly during solving the coupled pressure-velocity problem (i.e. to obtain a single pressure-velocity solution). Furthermore, 3-D problems are easily larger than the one presented here, which was reduced in size in order to include the poorly parallelizing Gauss-Seidel method into the competition. Applications of the 3-D code presented in Chapter 3 and Chapter 4 have up to 12 million unknowns, for which only the MG preconditioned CG is a suitable algorithm.

2.6 Viscous flow II: velocity-pressure formulation

2.6.1 Solution strategies for coupled velocity-pressure problems

The Stokes flow problem discussed in Section 2.3 leads to a matrix equation, whose solution vector describes the velocity and pressure fields. This equation is repeated here for clarity:

$$\mathbf{A} \quad x = b \tag{2.135a}$$

where $\mathbf{A}$, x and b are defined as

$$\mathbf{A} = \begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{G^T} & \mathbf{0} \end{bmatrix} \quad , \quad x = \begin{pmatrix} u \\ p \end{pmatrix}, \text{ and } \quad b = \begin{pmatrix} F \\ H \end{pmatrix} \tag{2.135b}$$

$\mathbf{K}$ is the stiffness matrix, $\mathbf{G}$ the gradient matrix (which is equal to the transpose of the divergence matrix; see Eq. (2.91) on p. 34), and vector F includes the body forces, traction boundary conditions, and velocity boundary conditions u_D. Vector H includes the dilatation, which is only non-zero if sources and/or sinks[6] exist in the flow field. H also includes the values resulting from $\mathbf{G^T} u_D$ (see second term in (2.84c) on p. 33 and Eq. (2.88)). In a dilatation-free Stokes flow problem, velocity boundary conditions have to be chosen such that they allow for an incompressible flow field in order to obtain a solution (i.e. no net flow of material into or out of the numerical domain). Hence, for dilatation-free flow fields, $\mathbf{G^T}(u + u_D) = 0$ so that $H = -\mathbf{G^T} u_D$. To somewhat simplify the equations, H will be ignored from now on, as its non-zero part comes only from $-\mathbf{G^T} u_D$.

Mathematically, equation (2.135) represents a so-called saddle point problem, that is, the solution cannot be obtained by locating the minimum of the quadratic form $\mathbf{A}x = b$ (see Fig. 2.5 on p. 40). The major difficulty for solving this equation numerically arises from the zero-block on the diagonal of the "saddle point matrix" $\mathbf{A}$, which makes this matrix positive semi-definite. Several standard tools for solving symmetric, positive-definite matrix equations, such as Conjugate Gradients (CG) or Cholesky factorization cannot be used. Stationary iterative solvers that make use of the reciprocal diagonal of $\mathbf{A}$, also fail to solve (2.135). One way to overcome this issue is to decomposed Eq. (2.135) into the two inherent subproblems.

The problem specified in (2.135) comprises two matrix equations

$$\mathbf{K}u + \mathbf{G}p = F \tag{2.136a}$$

$$\mathbf{G^T}u = 0 \tag{2.136b}$$

[6] A sink in this context defines a region in which material is removed from the domain, e.g. to account for the volume lost by melt extraction. Accordingly, a source defines a region of material accretion.

which offer additional ways to solve the Stokes flow problem. The solution strategies for solving the Stokes flow problem can be subdivided into two categories: 1) coupled methods that compute the unknown vectors u and p simultaneously by solving (2.135), and 2) segregated methods that solve u and p separately by combining the two equations in (2.136) in some way. The latter group can be further subdivided into methods that eliminate p from (2.136a) and solve for u first, and those with the opposite pattern for reducing the equations.

The two strategies that I will present are both segregated methods: In the *penalty method* (e.g. Engelman et al., 1982; Hughes et al., 1979), pressure is eliminated from (2.136a) so that a solution for u can be evaluated. *Patera's algorithm* (cf. Maday and Patera, 1989) forms the Schur complement of $\mathbf{A}$. It then solves for p using an iterative solver within which the velocity solution is corrected using an inner (nested) solver.

For the sake of completeness I want to mention that the coupled problem (2.135) can be solved using Krylov-subspace solvers like *MinRes* (e.g. Paige and Saunders, 1975). They require the same types of preconditioning strategies for pressure and velocity as will be presented below, but furthermore require more storage for their implementation without an apparent gain in efficiency. This was found in tests using MinRes to solve the coupled problem, before deciding to focus on segregated methods.

2.6.2 Penalty method

Theory and numerical fomulation

In this method the incompressibility constraint is relaxed to allow a slightly compressible behavior of the fluid. Namely, $\nabla \cdot v = 0$ is replaced by $\nabla \cdot v = p/\gamma$, where γ is a large scalar parameter (typically $\gamma \approx 10^7 - 10^8$; (Hughes et al., 1979)). Pressure is thus redefined as

$$p = -\gamma \nabla \cdot v \qquad \text{(definition of pressure in penalty method)} \tag{2.137}$$

Substituted in the definition of the isotropic part of the stress tensor (Eq. (2.57) on p. 29), this can be viewed as approximating a very large but not infinite bulk modulus (the latter defines the fluid's resistance to uniform compression).

Using the new definition of p (2.137), the Stokes problem is re-formulated as a problem with a single unknown vector v

$$\sigma^{(\gamma)}_{ij,i} + f_i = 0 \qquad \text{force equilibrium, with} \tag{2.138a}$$

$$\sigma^{(\gamma)}_{ij} = -p\delta_{ij} + \eta 2 v_{(i,j)} \qquad \text{and } p \text{ given by (2.137)} \tag{2.138b}$$

$$v_i = v_{Di} \qquad \text{on } \Gamma_{Di} \text{ (Dirichlet boundary condition)} \tag{2.138c}$$

$$\sigma^{(\gamma)}_{ii} n_j = t_i \qquad \text{on } \Gamma_{Ni} \text{ (Neumann boundary condition)} \tag{2.138d}$$

The weak form of (2.138) is given by

$$\int_\Omega \dot{\epsilon}(w)^T\, \mathbf{C}_\eta\, \dot{\epsilon}(v)\, \mathrm{d}\Omega - \gamma \int_\Omega w_{,i}\, v_{,j}\ \mathrm{d}\Omega = \int_\Omega w f_i\ \mathrm{d}\Omega + \int_{\Gamma_N} w\, t_i\, d\Gamma \tag{2.139}$$

with $\dot{\epsilon}$ and $\mathbf{C}_\eta$ denoting the strain rate vector and constitutive matrix as introduced in Section 2.3.2. Substituting the Galerkin approximations for velocity trial solutions v (2.66) and weighting functions w (2.67) (see p. 31) leads to the following matrix equation:

$$[\mathbf{K} + \mathbf{K}^\gamma]\ u = f \tag{2.140a}$$

where

$$[K]_{iAjB} := \int_\Omega \dot{\epsilon}(N_A e_i)^T\, \mathbf{C}_\eta\, \dot{\epsilon}(N_B e_j)\, \mathrm{d}\Omega \qquad \text{(stiffness matrix)} \tag{2.140b}$$

$$[K^\gamma]_{iAjB} := \gamma \int_\Omega \nabla(N_A e_i)\ \nabla(N_B e_j)\, \mathrm{d}\Omega \qquad \text{(penalty matrix)} \tag{2.140c}$$

$$\begin{aligned}(F)_{iA} := &\int_\Omega N_A e_i\ f_i\, \mathrm{d}\Omega \\ &+ \int_{\Gamma_N} N_A e_i\ t_i\, \mathrm{d}\Gamma_N \\ &- [K + K^\gamma]_{Dj\,iA}\ u_{Dj}\end{aligned} \qquad \text{(force vector)} \tag{2.140d}$$

$\mathbf{K}^\gamma$ is the so-called penalty matrix, which has the same dimensions as $\mathbf{K}$. In the Stokes flow problem, $\mathbf{K}$ is proportional to the viscosity η (which enters $\mathbf{K}$ through $\mathbf{C}_\eta$), whereas $\mathbf{K}^\gamma$ is proportional to the penalty parameter γ. Since γ has to be chosen large enough to enforce a reasonable incompressibility, $\mathbf{K}^\gamma$ may dominate over $\mathbf{K}$, which can lead to a zero-velocity solution (="locked"). This happens, if too many incompressibility constraints (expressed by $\mathbf{K}^\gamma$) compared to velocity unknowns are specified (see discussions in Hughes (2000); Donea and Huerta (2003)). In other words, there are not enough free velocity degrees of freedom to describe a flow field that is as incompressible as enforced by $\mathbf{K}^\gamma$, so that the only possible solution is a zero flow everywhere (which then of course is incompressible). In case of non-zero velocity boundary conditions, no solution at all might be obtained.

There are two ways to overcome this so-called "locking" of the finite element mesh:

1. *under-integration* of the penalty terms (Malkus and Hughes, 1978)
   The integrals for constructing the element matrices are calculated by numerical integration (Gaussian quadrature), which requires a certain number of integration points in order to exactly evaluate the integral of a function with a given polynomial degree. By using fewer integration points, a less accurate, lower order integral is evaluated. Hughes et al. (1979) showed that by using an integration order high enough to exactly evaluate the terms in $\mathbf{K}$, but a lower order integration scheme for the terms in $\mathbf{K}^\gamma$, the number of incompressibility constraints is effectively reduced.

2. *consistent penalty method* (Engelman et al., 1982)
   Here the penalty terms are projected onto a pressure functional space with fewer basis functions than the velocity space, which again leads to fewer incompressibility constraints and avoids locking. For an artifact-free velocity solution, the pressure basis functions need to satisfy the LBB-condition (discussed below).

For specific combinations of velocity and pressure functional spaces, both methods are equivalent (Engelman et al., 1982). However, if not equivalent, that same study finds the consistent penalty method to be more accurate, so that this method has been chosen. It requires the assembly of a new matrix $\mathbf{M}$, sometimes referred to as the pressure mass matrix. The consistent penalty formulation for the Stokes problem is then given by

$$[\mathbf{K} + \mathbf{K}^{\gamma}]\ u = F \qquad \text{where} \tag{2.141a}$$

$$\mathbf{K}^{\gamma} = \gamma \mathbf{G} \mathbf{M}^{-1} \mathbf{G}^{\mathbf{T}} \tag{2.141b}$$

and, once u is obtained, pressure is calculated using

$$p = -\gamma \mathbf{M}^{-1} \mathbf{G}^{\mathbf{T}} u \tag{2.141c}$$

The stiffness matrix $\mathbf{K}$, gradient matrix $\mathbf{G}$ and the force vector F are the same as in (2.140). The components of the pressure mass matrix $\mathbf{M}$ are given by

$$[M]_{\hat{A}\hat{B}} := \int_{\Omega} \hat{N}_{\hat{A}}\ \hat{N}_{\hat{B}}\, \mathrm{d}\Omega \qquad \text{(pressure mass matrix)} \tag{2.142}$$

As before (page 31), $\hat{N}_{\hat{A}}$ denotes the pressure shape function corresponding to pressure nodes $\hat{A}$, etc. Pre-multiplying the divergence of the velocity field ($\mathbf{G}^{\mathbf{T}}u$) by $\mathbf{M}^{-1}$ is equivalent to projecting $\nabla \cdot u$ onto the pressure space, on which $\mathbf{M}$ is defined. The consistent penalty method gives a non-trivial solution for the velocity field as long as velocity and pressure spaces are chosen to fulfill the so-called Babuska-Brezzi condition, that is, the elements selected have to be LBB-stable (see Zienkiewicz and Taylor (1989, pp. 324) or Hughes (2000, pp. 207) and references therein).

A simple and fairly effective way to estimate whether or not a velocity-pressure element is LBB-stable, is to calculate the ratio between free velocity degrees of freedom and pressure (incompressibility) constraints. In general, this ratio needs to be less than one pressure dof per velocity node in 2-D and 3-D. A list of 2-D elements with their so-called constraint ratio is given in Hughes (2000, p. 211) for discontinuous pressure elements and continuous pressure elements [p. 215]. The former element type is characterized by pressure shape functions that vanish on element boundaries, i.e. the pressure nodes are located within elements and their associated shape functions are nonzero only within this element. Continuous pressure shape functions are defined on the edges of elements so that all elements connecting to a pressure node have non-zero contributions from the associated shape function (see the examples in Fig. 2.1 on p. 14). The pressure field in this case is continuous across element boundaries.

Since the solution algorithm ultimately has to perform well in large 3-D problems, we rely on iterative solvers such as CG and multigrid. It is well-known that the convergence rate of CG is related to the condition number of the matrix (see the introduction to CG in Section 2.4). The high values for γ lead to a poorly conditioned matrix $\mathbf{K}^{\gamma}$, which no iterative solver can efficiently deal with. Therefore a lower γ is used in combination with an iterative scheme commonly known as ”Uzawa iterations” or ”method of multipliers”. It was developed independently by Arrow and Hurwicz (1958), Hestenes (1969) and Powell (1969) for different purposes. The scheme works as follows:

1. calculate a velocity field u by solving (2.141a) with a small value for γ
2. calculate a <u>correction</u> δp to the pressure using (2.141c)
3. calculate the forces resulting from the gradients of the pressure correction using the gradient matrix (i.e. $\delta F_p = \mathbf{G}\,\delta p$)
4. correct the forces F in (2.141a) by adding δF_p and solve for a new u

Steps 1-4 are repeated until convergence. The iterative algorithm successively increases the incompressibility of the flow field by calculating corrections to the pressure field and updating the forces accordingly. The penalty parameter is usually scaled by the maximum value of viscosity η in the domain (i.e. $\gamma = 10$ means: $\mathbf{K}^{\gamma} = 10 \max(\eta)\, \mathbf{G}\mathbf{M}^{-1}\mathbf{G}^{\mathbf{T}}$). In all following descriptions and discussions, the number given for the penalty parameter is scaled by the maximum viscosity in the algorithm.

I have tested the Uzawa-consistent penalty method with two types of triangular elements: The Crouzeix-Raviart element with quadratic velocity shape functions and linear, discontinuous pressure shape functions (referred to as ”CR-element” from now on) and the Taylor-Hood element with quadratic velocity shape functions and linear, continuous pressure shape functions (referred to as ”TH-element” below). The reason for choosing quadratic velocity/linear pressure elements instead of less computationally costly linear velocity/constant pressure is that the latter can lead to spurious pressure modes. These modes appear as oscillating pressure solutions (the so-called checkerboard pattern; see Hughes et al. (1979), but also the detailed analysis in Sani et al. (1981)a,b), which require smoothing and downgrade the velocity/pressure solution.

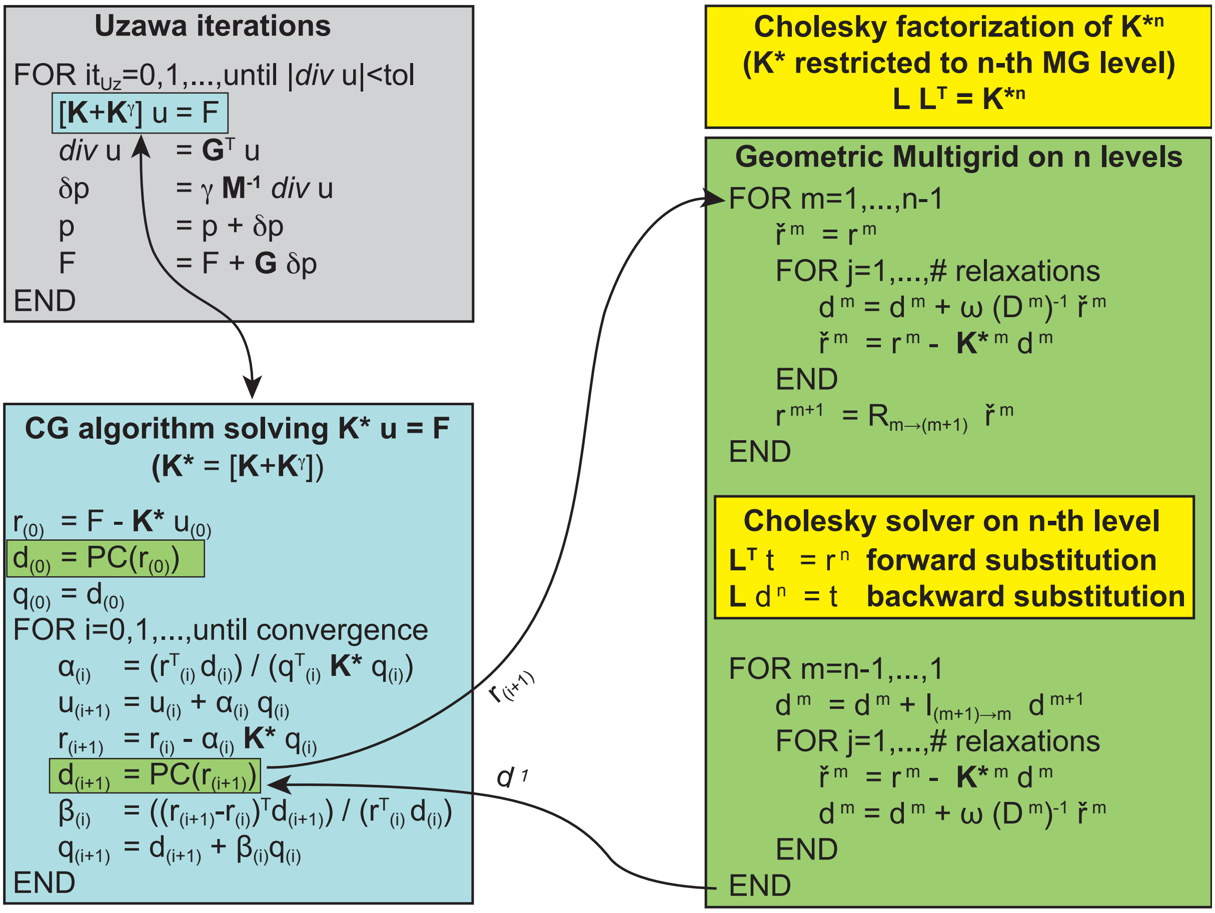
Another reason for not using constant pressure elements is related to the lithostatic pressure inherent to all large-scale geodynamic calculations. The lithostatic pressure in a material with constant density is a linearly increasing function of depth. Constant pressure elements cannot capture this linear function and compute a depth-averaged constant pressure inside each element. This can trigger artificial flow (see Fig 7.6 in Fortin and Fortin (1985) and discussions in Pelletier et al. (1989)) that is solely driven by a badly approximated lithostatic pressure – the resulting flow is non-physical, because lithostatic pressure should not inherently drive any flow. Usually density varies in space, however, constant density within each element can be assumed, similarly to using a constant, average viscosity within each element to improve accuracy (Deubelbeiss and Kaus, 2008). In this case, linear pressure shape functions fully capture the lithostatic pressure and do not introduce artificial flow.

Consistent penalty method with Crouzeix-Raviart elements

The CR-element is the standard element used in combination with the consistent penalty method, because the inversion of $\mathbf{M}$ can be done on the element level. Since its pressure is discontinuous, each element's pressure mass matrix (2.142) can be evaluated completely and inverted, because it is independent of all pressure shape functions in other elements. In 2-D, the element $\mathbf{M}^{-1}$-matrices form small 3x3 blocks around the diagonal of the global $\mathbf{M}^{-1}$, because only the three pressure nodes within an element are connected to each other. Steps 2 and 3 in the Uzawa scheme are therefore conducted on the element level. The second advantage of CR-elements is that the sparsity of $\mathbf{K}$ and $\mathbf{K}^{\gamma}$ are identical: $\mathbf{M}$ only connects the pressure nodes inside an element, thus, $\mathbf{G}\,\mathbf{M}^{-1}\,\mathbf{G}^{\mathbf{T}}$ relates velocity degrees of freedom (dofs) that are connected by elements, just like $\mathbf{K}$. The memory requirements to store $\mathbf{K}$ are the same as for storing $\mathbf{K}^{*} = \mathbf{K} + \mathbf{K}^{\gamma}$. The algorithm is shown in Fig. 2.20 and contains the following pieces

- the outermost loop represents the Uzawa iterations (gray box)
- a CG algorithm is used to solve $\mathbf{K}^{*}u = F$, where $\mathbf{K}^{*} = \mathbf{K} + \mathbf{K}^{\gamma}$ (blue box)
- the CG is preconditioned by a single V-cycle of geometric multigrid (green box)
- in the beginning, matrix $\mathbf{K}^{*}$ is recursively restricted to all MG levels to obtain the matrices $\mathbf{K}^{*m}, m = 2, ..., n$
- on the coarsest (n-th) MG-level, $\mathbf{K}^{*n}$ is factorized, so that a direct solution for $\mathbf{K}^{*n}d^{n} = r^{n}$ can be obtained by forward-backward substitutions

When using a small penalty number, the first few Uzawa iterations produce flow fields that are very compressible. During this initial phase, corrections to pressure and the related forces can be obtained from less accurate flow fields. Thus, the number of CG



Uzawa iterations

```
FOR $it_{Uz}=0,1,\dots,$until $|div\ u|<tol$
    $[K+K^\gamma]\ u = F$
    $div\ u = G^T u$
    $\delta p = \gamma\, M^{-1}\, div\ u$
    $p = p + \delta p$
    $F = F + G\,\delta p$
END
```

Cholesky factorization of K^{*n}
(K^* restricted to n-th MG level)
$L\,L^T = K^{*n}$

Geometric Multigrid on n levels

```
FOR m=1,...,n-1
    $\check{r}^m = r^m$
    FOR j=1,...,# relaxations
        $d^m = d^m + \omega\,(D^m)^{-1}\,\check{r}^m$
        $\check{r}^m = r^m - K^{*m}\,d^m$
    END
    $r^{m+1} = R_{m\rightarrow(m+1)}\ \check{r}^m$
END
```

Cholesky solver on n-th level
$L^T t = r^n$ **forward substitution**
$L\,d^n = t$ **backward substitution**

```
FOR m=n-1,...,1
    $d^m = d^m + I_{(m+1)\rightarrow m}\ d^{m+1}$
    FOR j=1,...,# relaxations
        $\check{r}^m = r^m - K^{*m}\,d^m$
        $d^m = d^m + \omega\,(D^m)^{-1}\,\check{r}^m$
    END
END
```

CG algorithm solving $K^*\,u = F$
($K^* = [K+K^\gamma]$)

```
$r_{(0)} = F - K^*\,u_{(0)}$
$d_{(0)} = PC(r_{(0)})$
$q_{(0)} = d_{(0)}$
FOR i=0,1,...,until convergence
    $\alpha_{(i)} = (r^T_{(i)}\,d_{(i)})\ /\ (q^T_{(i)}\,K^*\,q_{(i)})$
    $u_{(i+1)} = u_{(i)} + \alpha_{(i)}\,q_{(i)}$
    $r_{(i+1)} = r_{(i)} - \alpha_{(i)}\,K^*\,q_{(i)}$
    $d_{(i+1)} = PC(r_{(i+1)})$
    $\beta_{(i)} = ((r_{(i+1)}-r_{(i)})^T d_{(i+1)})\ /\ (r^T_{(i)}\,d_{(i)})$
    $q_{(i+1)} = d_{(i+1)} + \beta_{(i)}\,q_{(i)}$
END
```

Figure 2.20: *Algorithm for the consistent penalty method using Uzawa iterations (gray box) in combination with the Crouzeix-Raviart element (CR). In every iteration, $\nabla \cdot u$ and δp are calculated on the element level. The velocity solution is obtained using a CG algorithm (blue box) that is preconditioned using a single V-cycle of geometric multigrid (MG) on n levels (green box). $\mathbf{K}^*$ denotes the sum of stiffness matrix $\mathbf{K}$ and penalty matrix $\mathbf{K}^\gamma$. On the coarsest MG level, a Cholesky direct solver (yellow box) is employed to avoid further coarser meshes. Note that the factorized matrix $\mathbf{L}$ is only calculated once in the beginning, after $\mathbf{K}^*$ has been restricted to all MG-levels. The calculation of β in the CG is the Polak-Ribière-formulation, because MG represents a non-constant preconditioner (see discussion on p. 60).*

iterations can be significantly reduced by dynamically adjusting the tolerance for the CG depending on the divergence ($\mathbf{G^T}u$) of the flow field. No increase in the number of Uzawa iterations has been observed, if the CG tolerance is set to be one order of magnitude lower than the norm of the current divergence. This proves that the lower-quality flow fields during the beginning of the Uzawa iterations are sufficient to correct pressure and associated forces. Using the velocity field from the last Uzawa iteration as initial guess for the next u also reduces the number of CG iterations.

Consistent penalty method with Taylor-Hood elements

Several complications arise, when the TH-element is used in combination with the consistent penalty method. The continuous pressure shape functions (continuous between elements) preclude the inversion of element pressure mass matrices, since they are incomplete. An inversion of the assembled $\mathbf{M}$ is impractical, because it would result in a huge fill-in of non-zeros in the sparse pattern of $\mathbf{M}$ and accordingly large memory requirements. Not being able to calculate $\mathbf{M}^{-1}$ also means that $\mathbf{K}^{\gamma}$ cannot be formed explicitly ($\mathbf{K}^{\gamma}$ would also be a full matrix anyway that could not be stored, even for comparably small 2-D problems). I have circumvented these problems by using an ”inner” conjugate gradient algorithm whenever the action of $\mathbf{M}^{-1}$ is required, namely, during the multiplications of $\mathbf{K}^{*}$ to a vector in the ”outer” CG iterations and during the relaxations on the MG levels. The algorithm for the Uzawa-consistent penalty method using Taylor-Hood elements is shown in Fig. 2.21.

The following modifications to the above standard Uzawa-consistent penalty method have been done:

- the outermost loop (Uzawa iterations, gray box) operates exclusively on global matrices and vectors
- the outer CG (blue box) solves $\mathbf{K}^{*}u = F$, but $\mathbf{K}^{*}$ cannot be formed explicitly
- every multiplication involving $\mathbf{K}^{*}$ is done in three steps (brown box):
(1) $y = \mathbf{G^T}q$
(2) an inner, multigrid-preconditioned CG algorithm solves $\mathbf{M}z = y$ (red box)
(3) the multiplication is completed by $\mathbf{K}^{*}q = \mathbf{K}q + \mathbf{G}z$
- two difficulties arise during the relaxations in the MG algorithm preconditioning the the outer CG: (1) matrix $\mathbf{K}^{*}$ cannot be formed on the fine grid, thus, cannot be restricted to coarser levels; (2) no simple smoother is available, since the diagonal of $\mathbf{K}^{*}$ (white box) is also unobtainable; different methods to approximate $\mathbf{K}^{*}$ and its diagonal on the MG levels are discussed below

As for the CR-element, the tolerance for the outer CG is dynamically adjusted to be one order of magnitude smaller than $\nabla \cdot u$ in each Uzawa iteration. The inner CG (solving the $\mathbf{M}^{-1}$ problem) requires a tolerance as high as the tolerance used for the outer CG (blue box) in every call. A lower tolerance would introduce roundoff errors in the $\mathbf{K}^{*}q$ multiplications, which lead to linearly dependent search directions of the outer CG and/or re-introduced error components in search directions that won’t be selected again. If this

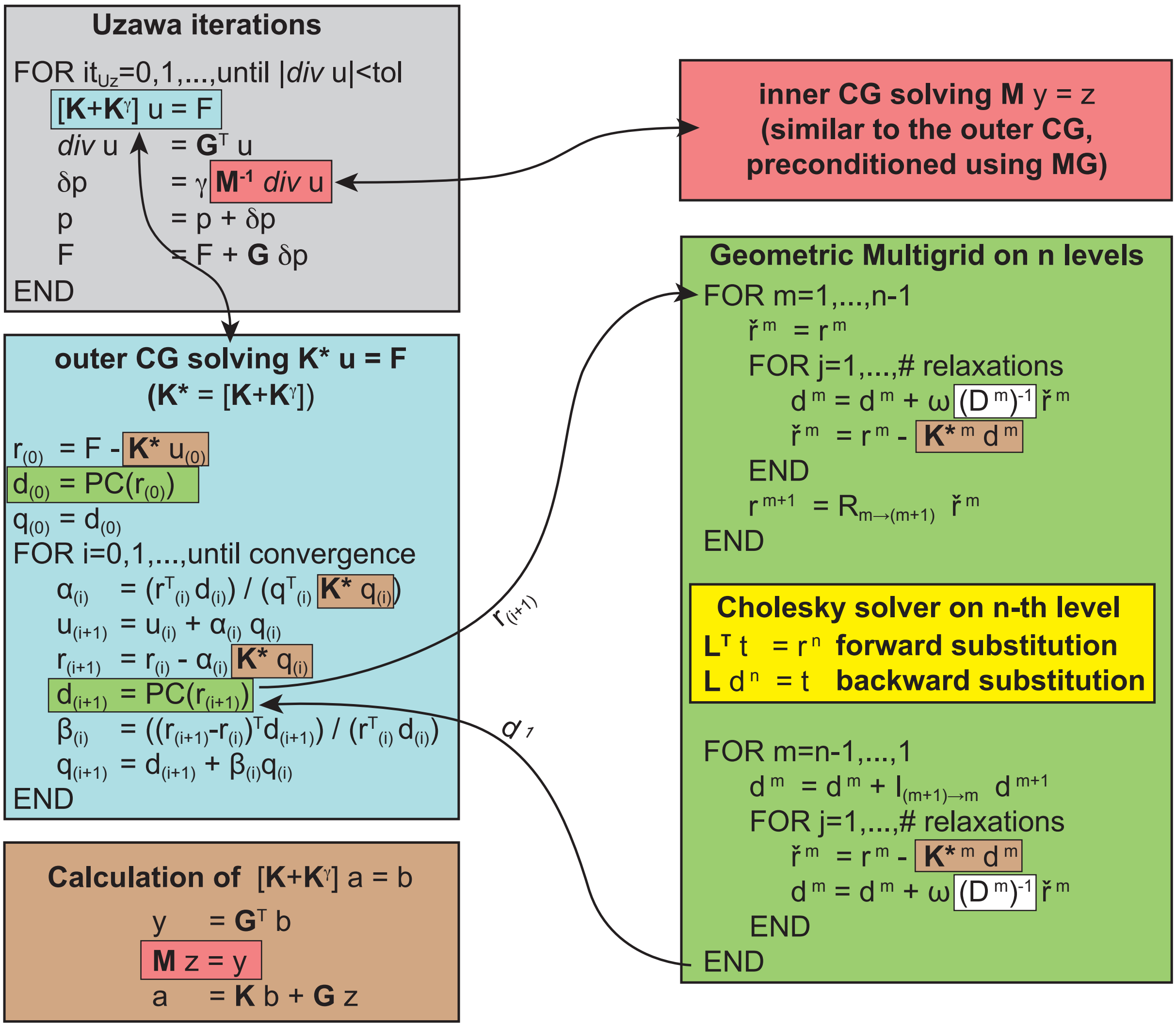

Uzawa iterations
FOR it_{Uz}=0,1,...,until |div u|<tol
[K+K$^\gamma$] u = F
div u = G^T u
δp = γ M^{-1} div u
p = p + δp
F = F + G δp
END
inner CG solving M y = z
(similar to the outer CG, preconditioned using MG)
outer CG solving K* u = F
(K* = [K+K$^\gamma$])
$r_{(0)}$ = F - K* $u_{(0)}$
$d_{(0)}$ = PC($r_{(0)}$)
$q_{(0)}$ = $d_{(0)}$
FOR i=0,1,...,until convergence
$\alpha_{(i)}$ = ($r^T_{(i)} d_{(i)}$) / ($q^T_{(i)}$ K* $q_{(i)}$)
$u_{(i+1)}$ = $u_{(i)}$ + $\alpha_{(i)} q_{(i)}$
$r_{(i+1)}$ = $r_{(i)}$ - $\alpha_{(i)}$ K* $q_{(i)}$
$d_{(i+1)}$ = PC($r_{(i+1)}$)
$\beta_{(i)}$ = (($r_{(i+1)}$-$r_{(i)}$)$^T d_{(i+1)}$) / ($r^T_{(i)} d_{(i)}$)
$q_{(i+1)}$ = $d_{(i+1)}$ + $\beta_{(i)} q_{(i)}$
END
$r_{(i+1)}$
d_1
Calculation of [K+K$^\gamma$] a = b
y = G^T b
M z = y
a = K b + G z
Geometric Multigrid on n levels
FOR m=1,...,n-1
$\check{r}^m$ = r^m
FOR j=1,...,# relaxations
d^m = d^m + ω $(D^m)^{-1}$ $\check{r}^m$
$\check{r}^m$ = r^m - K^{*m} d^m
END
r^{m+1} = $R_{m\rightarrow(m+1)}$ $\check{r}^m$
END
Cholesky solver on n-th level
L^T t = r^n forward substitution
L d^n = t backward substitution
FOR m=n-1,...,1
d^m = d^m + $I_{(m+1)\rightarrow m}$ d^{m+1}
FOR j=1,...,# relaxations
$\check{r}^m$ = r^m - K^{*m} d^m
d^m = d^m + ω $(D^m)^{-1}$ $\check{r}^m$
END
END

Figure 2.21: *Algorithm for the consistent penalty method using Uzawa iterations (gray box) in combination with the Taylor-Hood element.* $\mathbf{M}^{-1}$, *hence* $\mathbf{K}^\gamma$, *cannot be formed explicitly, because the continuous pressure elements would lead to a the large fill-in of non-zeros in both matrices.* $\mathbf{K}^*$ *therefore denotes the hypothetical sum of stiffness matrix* $\mathbf{K}$ *and penalty matrix* $\mathbf{K}^\gamma$*. All multiplications of* $\mathbf{K}^*$ *to a vector are done in 3 steps (brown box). An inner MG-preconditioned CG algorithm (red box) is used, whenever* $\mathbf{M}^{-1}$ *has to multiply a vector. Velocity updates are calculated by an outer CG algorithm (blue box) that is preconditioned by a single V-cycle MG (green box). The Polak-Ribière-formulation for* β *is used in both CG. A Cholesky direct solver is used on the coarsest MG level. Two different coarse approximations for* $\mathbf{K}^*$*, which are factorized to obtain* $\mathbf{L}$*, have been tested. Complications arise from finding a suitable smoother for the relaxations on each MG-level (white boxes). See text for details.*

happens, the outer CG algorithm fails to converge, as soon as it starts to operate at a tolerance higher than the one defined for the inner CG.

The calculation of β in all CG algorithms in Fig. 2.20 and 2.21 follows the *Polak-Ribière* formulation rather than *Fletcher-Reeves* formulation (Eq. (2.132) and (2.132), resp., on p. 60), because the multigrid preconditioner represents a non-constant operator during the CG iterations. This choice can also help to make CG less sensitive to roundoff error.

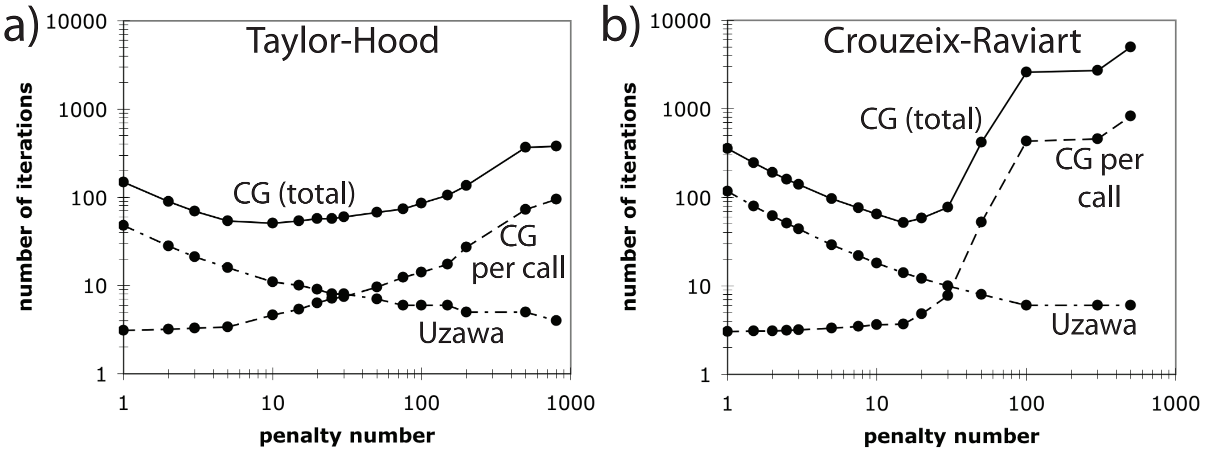


Figure 2.22: *Number of iterations required to solve a 2-D, isoviscous "sinker-problem" with 2,200 velocity unknowns. (a) Taylor-Hood element and algorithm shown in Fig. 2.21; (b) Crouzeix-Raviart element and algorithm shown in Fig. 2.20. A larger penalty term reduces the number of Uzawa iterations but increases the number of CG iterations, because the condition number of* $\mathbf{K}+\mathbf{K}^{\gamma}$ *becomes worse. Smaller penalty terms reduce the number of CG iterations but lead to more Uzawa iterations until the required incompressibility is achieved. In the isoviscous case, the total number of CG iterations has a minimum for* $\gamma = 10$ *and* $\gamma = 15$ *for Taylor-Hood and Crouzeix-Raviart element, resp.*

$\mathbf{M}$ is well conditioned, so that the inner CG solving the $\mathbf{M}^{-1}$ problem requires only 2-4 iterations, if preconditioned with a single V-cycle. Fewer MG levels than used for preconditioning the outer CG could be used, because $\mathbf{M}$ is much smaller than $\mathbf{K}^{*}$ so that a Cholesky factorization might be possible on a finer mesh.

Performance of the consistent penalty method

The performance of the CR-element and TH-element have been compared for a 2-D test problem called the "sinker-problem": A dense and highly viscous body is placed central in a squared domain with free-slip boundary conditions. Since the material surrounding the body is less viscous and less dense, the body will sink. The flow and pressure fields are only calculated for the first time step (starting with a zero-guess for both) so that no advection scheme is required. The finite element mesh has been constructed such that the boundary of the sinker does not cross element edges. Thus, elements with constant density and viscosity can be used everywhere, in favor of a more accurate solution (Deubelbeiss and Kaus, 2008).

For both elements, the combination of the penalty method with an iterative solver is a trade-off between the number of Uzawa iterations and the number of iterations that the iterative solver needs to evaluate u. This is shown for an isoviscous sinker problem[7] in

[7] The trade-off between number of Uzawa iterations and number of iterations of the solver, depending on the penalty number, are better visible in an isoviscous problem. Similar patterns are seen in problems

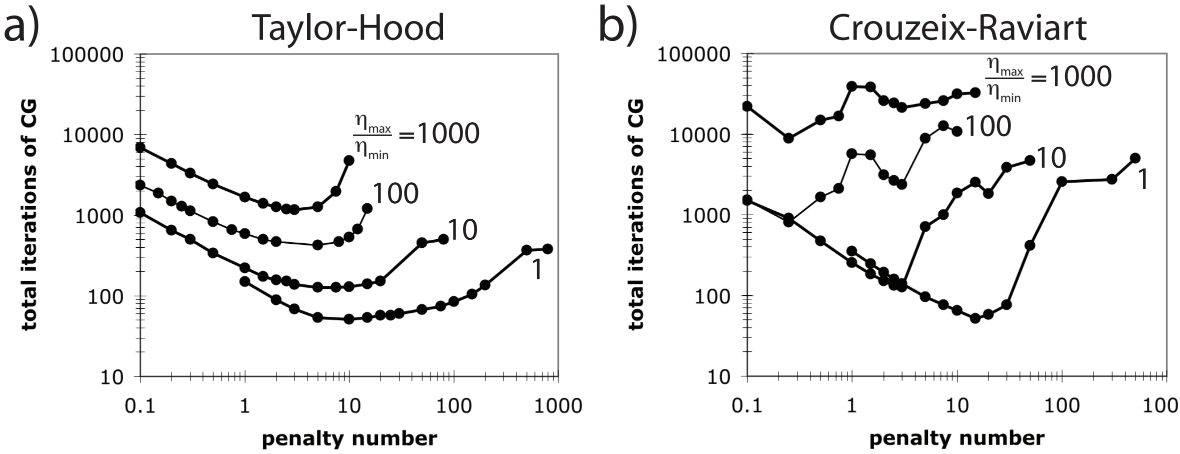


Figure 2.23: *Total number of CG iterations required to solve a 2-D "Sinker-problem" with 2,200 velocity unknowns, in which the viscosity contrast is successively increased. (a) Taylor-Hood element: The optimum penalty number shifts towards lower values ($\gamma = 3$ for the highest viscosity contrast considered) and the optimum range narrows. Number of iterations increases by roughly a factor of 3 per magnitude in viscosity contrast. (b) Crouzeix-Raviart element: With increasing viscosity contrasts, the optimum penalty number rapidly shifts towards $\gamma = 1$, which indicates that the CG solver cannot efficiently deal with the penalty terms. No clear minimum exists for larger viscosity contrasts, because $\gamma > 1$ results in many CG iterations and $\gamma < 1$ causes convergence problems in the Uzawa iterations.*

Fig. 2.22. On the one hand, a larger penalty number ($\gamma > 50$) leads to a good incompressibility within few Uzawa iterations, but the poorly conditioned $\mathbf{K}^{\gamma}$ causes more iterations per call of the iterative solver evaluating u. On the other hand, reducing γ improves the convergence rate of the iterative solver, but requires more Uzawa iterations and more calls to the iterative solver, until a good incompressibility is obtained.

The total number of CG iterations mainly controls the computation time, because the repeated multiplications by $\mathbf{K}^{*}$ represent the largest mathematical operation. A broad minimum range for $5 < \gamma < 15$ has been found for the TH-element (Fig. 2.22a). The number of Uzawa and CG iterations increases smoothly towards smaller and higher penalty numbers, respectively. The optimum range for the CR-element is very narrow ($\gamma \sim 15$) and higher values suddenly lead to many more CG iterations. This makes it difficult to find a problem-independent, optimum value for γ, when using the CR-element.

Fig. 2.23 shows experiments, in which the viscosity of the sinker has been successively increased by an order of magnitude. Note that the penalty numbers on the abscissa are scaled by the maximum viscosity value when they enter $\mathbf{K}^{\gamma}$, so that the penalty terms in $\mathbf{K}^{\gamma}$ increase in magnitude as the viscosity of the sinker is increased. In case of the TH-element (Fig. 2.23a), the number of CG iterations increases by about a factor of 3 per magnitude of viscosity contrast. The optimum range for the penalty number is shifted towards smaller numbers and becomes narrower, but an optimum value can still be

with varying viscosity, but they are overprinted by additional effects.

found. The CR-element (Fig. 2.23b) shows convergence problems for viscosity contrasts larger than 10, because the larger penalty terms cause serious problems in the CG solver. For larger viscosity contrasts, the best value for the CG solver would be $\gamma < 1$. However, this leads to hundreds of Uzawa iterations, during which incompressibility in the highly viscous regions is approached slowly or potentially never established. In these stronger regions, the incompressibility constraints become very weak for $\gamma < 1$, because the terms in $\mathbf{K}^\gamma$ decrease in magnitude relative to those in $\mathbf{K}$.

The performance of the Taylor-Hood element depends critically on two approximations that are necessary, because $\mathbf{K}^*$ cannot be formed explicitly. First, a smoother has to be defined (here, the reciprocal diagonal D^{-1} in the white boxes in Fig. 2.21), which in case of Jacobi relaxations should approximate the diagonal of $\mathbf{K}^* = \mathbf{K} + \mathbf{K}^\gamma$. Second, an approximation to the matrix $\mathbf{K}^*$ is required on each MG level to update the residual after each relaxation. The following approximations are found to perform best and have been used in the test problems shown in Fig. 2.22a and Fig. 2.23a:

$$\mathbf{K}^{*m} \approx \tilde{\mathbf{K}}^{*m} = R_{1\to m}\left[\mathbf{K} + \tilde{\mathbf{K}}^\gamma\right] \tag{2.143}$$

$$\text{where} \quad \tilde{\mathbf{K}}^\gamma = \mathbf{G}\; diag(\mathbf{M})^{-1}\; \mathbf{G}^\mathbf{T}$$

$$D \approx \tilde{D} = diag(\mathbf{K}) + I_{m\to 1}\; diag(\tilde{\mathbf{K}}^{\gamma m}) \tag{2.144}$$

In Eq. (2.143), I use the reciprocal diagonal of $\mathbf{M}$ to approximate $\mathbf{M}^{-1}$ on the fine mesh, and use it to form an approximate penalty matrix $\tilde{\mathbf{K}}^\gamma$. It is added to $\mathbf{K}$ and successively restricted to all MG levels. When using the reciprocal diagonal of $\mathbf{M}$, $\tilde{\mathbf{K}}^\gamma$ has about four times as many non-zero entries as $\mathbf{K}$ in 2-D. This is, because the matrix multiplications connect velocity dofs that are separated as far as two elements, if both pressure and velocity shape functions are continuous, while $\mathbf{K}$ connects velocity dofs only across one element. In 3-D, $\tilde{\mathbf{K}}^\gamma$ is a factor of about 8 larger than $\mathbf{K}$, which is likely to result in memory problems for larger problem sizes.

A smoother is constructed using Eq. (2.144): The diagonal of the penalty matrix on the coarsest mesh (resulting from the restriction of $\tilde{\mathbf{K}}^\gamma$) is interpolated "upwards" onto all finer meshes, where it is added to the diagonal of the restricted $\mathbf{K}$ matrix.

I have also tried to restrict matrices $\mathbf{M}$, $\mathbf{G}$ and $\mathbf{K}$ separately and form $\mathbf{K}^\gamma$ on each MG mesh. This has the great advantage of avoiding the 4-, resp. 8-times larger $\hat{\mathbf{K}}^\gamma$ matrix on the fine mesh, but unfortunately this does not perform as well as the method in (2.143).

Summary

The consistent penalty method shows an unfortunate increase of the number of total CG iterations with increasing viscosity contrasts. For viscosity contrasts exceeding $10^2 - 10^3$, this leads a very large number of CG iterations, even for small 2-D problems (about 2000 iterations when using the Taylor-Hood element, about 10,000 when using the Crouzeix-Raviart; see Fig. 2.23). This finding makes the consistent penalty method less practical for 3-D problems that require iterative solvers.

The Crouzeix-Raviart element in combination with the consistent penalty method should only be used if a direct solver is employed. In this case, a large penalty number has no effect on time required to solve the $\mathbf{K}^*$-inverse problem, and the Uzawa iterations will converge quickly within few cycles. The Uzawa iterations may even not be required, if γ is sufficiently large – however, since $\mathbf{K}^*$ has to be factorized only once, additional Uzawa iterations are computationally cheap. Since pressure is discontinuous in the CR-element, $\mathbf{K}$ and $\mathbf{K}^\gamma$ have the same sparsity, so that a factorization of the summed matrix is not more expensive than a factorization of $\mathbf{K}$. For 2-D problems and very small 3-D problems (<50,000 unknowns), this method might be the best choice.

The Taylor-Hood element is impossible to use in combination with a direct solver, because neither $\mathbf{M}^{-1}$ nor $\mathbf{K}^\gamma$ can be formed explicitly, but have to be ”simulated” using an iterative solver. The resulting algorithm (Fig. 2.21) is rather complex and suffers from many calls of the inner CG solving the $\mathbf{M}^{-1}$ problem. However, only very few iterations are required per call of the inner CG, if preconditioned using multigrid. The performance relies on a good approximation to $\mathbf{K}^\gamma$, which is required during the relaxations and for the coarse-mesh solution. The larger size of $\tilde{\mathbf{K}}^\gamma$, required only for the restriction of this matrix to all MG-levels and can be deleted afterwards, represents a major disadvantage, especially for 3-D applications.

In conclusion, the consistent penalty method with Crouzeix-Raviart elements is impractical in combination with an CG iterative solver. Using the Taylor-Hood element leads to a better performance, but is still less practical for viscosity contrasts larger than 100-1000. The next section will present a different approach to solve the Stokes flow problem.

2.6.3 Patera's algorithm

Theory and numerical formulation

An alternative to the above consistent penalty method is Patera's algorithm that was proposed by Patera (cf. Maday and Patera, 1989) to solve matrix equations arising from so-called spectral element formulations (these elements have shape functions of comparably high polynomial order). The algorithm is a specific Schur complement of the block-matrix $\mathbf{A}$ matrix in the saddle point problem in (2.135). The equations to be solved are repeated here for clarity:

$$\mathbf{K}u + \mathbf{G}p = F \tag{2.145a}$$

$$\mathbf{G^T}u = 0 \tag{2.145b}$$

While the penalty method aims to eliminate p from (2.145a) by expressing the divergence (2.145b) as a function of p, Patera's algorithm chooses the opposite approach. Solving (2.145a) formally for u

$$u = \mathbf{K}^{-1}F - \mathbf{K}^{-1}\mathbf{G}p \tag{2.146}$$

and substituting in (2.145b) gives

$$\mathbf{G^T}\left(\mathbf{K}^{-1}F - \mathbf{K}^{-1}\mathbf{G}p\right) = 0 \tag{2.147}$$

$$\left[\mathbf{G^T}\mathbf{K}^{-1}\mathbf{G}\right]p = 0 - \mathbf{G^T}\mathbf{K}^{-1}F \tag{2.148}$$

$$\mathbf{S}\,p = \hat{F} \tag{2.149}$$

Solving (2.149) to obtain p and substituting the result in (2.146) will give the velocity and pressure solutions for the Stokes flow problem.

Numerical implementation

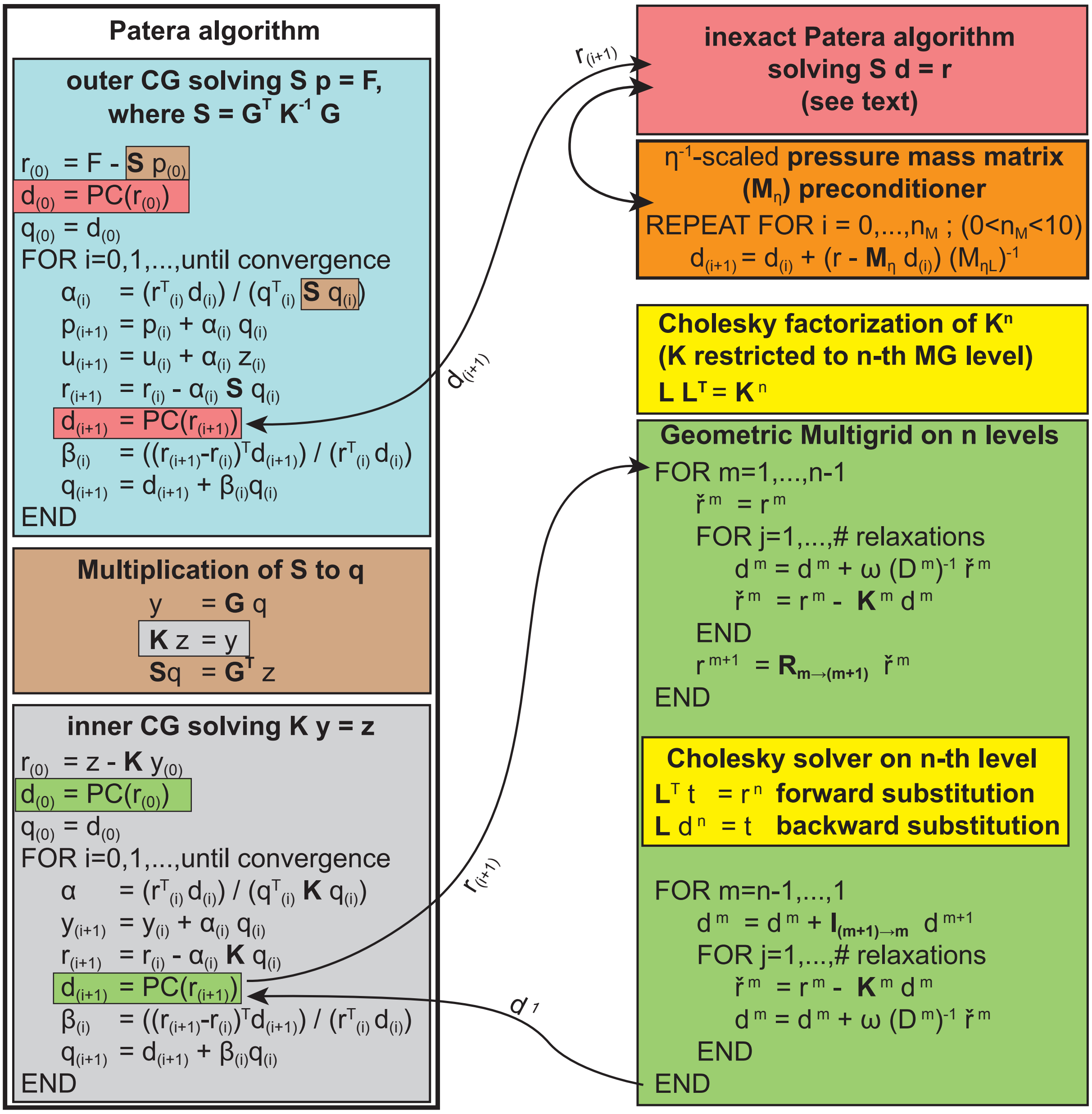
Similarly to the impossibility of calculating $\mathbf{K}^\gamma$ in the penalty method when using Taylor-Hood elements (because $\mathbf{M}$ is too expensive for inversion when pressure is continuous), matrix $\mathbf{S} = \mathbf{G^T}\mathbf{K}^{-1}\mathbf{G}$ cannot be formed explicitly: $\mathbf{K}^{-1}$ is unobtainable, because it would be a very large and full matrix. However, a CG algorithm solving $\mathbf{S}p = \hat{F}$ can be used, within which an inner CG solves the $\mathbf{K}^{-1}$ sub-problem in every outer iteration, i.e. whenever a multiplication of a vector by $\mathbf{S}$ is required. This algorithm is shown in Fig. 2.24. The action of $\mathbf{S}^{-1}$ is simulated by an outer CG algorithm (blue box), within which each multiplication by $\mathbf{S}$ is done in three steps (brown box). The second step would require $\mathbf{K}^{-1}$, but an inner CG (gray box) is used to perform the action of $\mathbf{K}^{-1}$. The inner CG is preconditioned using multigrid with a single V-cycle and a Cholesky solver on the coarsest mesh.

Once the solution for p is obtained, equation (2.146) can be used to solve for the flow field. However, each solution vector $z_{(i)} = \mathbf{K}^{-1}\mathbf{G^T}q_{(i)}$ from the inner CG represents a correction to the velocity field, which can be accumulated during the outer CG iterations, so that p and u are updated simultaneously. This is shown in the outer CG (blue box) in Fig. 2.24, namely:

$$u_{(i+1)} = u_{(i)} + \alpha_{(i)}z_{(i)} \tag{2.150}$$

where, $u_{(i)}$ is the velocity field from the previous pressure iteration and $\alpha_{(i)}$ is the step size of the outer CG into the search direction $q_{(i)}$.

Because no boundary conditions are imposed on the pressure field and the terms in $\mathbf{G}$



Patera algorithm

outer CG solving S p = F, where S = G^T K^{-1} G

```
$r_{(0)} = F - \mathbf{S}\, p_{(0)}$
$d_{(0)} = PC(r_{(0)})$
$q_{(0)} = d_{(0)}$
FOR i=0,1,...,until convergence
    $\alpha_{(i)} = (r^T_{(i)} d_{(i)}) / (q^T_{(i)} \mathbf{S}\, q_{(i)})$
    $p_{(i+1)} = p_{(i)} + \alpha_{(i)} q_{(i)}$
    $u_{(i+1)} = u_{(i)} + \alpha_{(i)} z_{(i)}$
    $r_{(i+1)} = r_{(i)} - \alpha_{(i)} \mathbf{S}\, q_{(i)}$
    $d_{(i+1)} = PC(r_{(i+1)})$
    $\beta_{(i)} = ((r_{(i+1)} - r_{(i)})^T d_{(i+1)}) / (r^T_{(i)} d_{(i)})$
    $q_{(i+1)} = d_{(i+1)} + \beta_{(i)} q_{(i)}$
END
```

Multiplication of S to q

```
$y = \mathbf{G}\, q$
$\mathbf{K}\, z = y$
$\mathbf{S}q = \mathbf{G}^T z$
```

inner CG solving K y = z

```
$r_{(0)} = z - \mathbf{K}\, y_{(0)}$
$d_{(0)} = PC(r_{(0)})$
$q_{(0)} = d_{(0)}$
FOR i=0,1,...,until convergence
    $\alpha = (r^T_{(i)} d_{(i)}) / (q^T_{(i)} \mathbf{K}\, q_{(i)})$
    $y_{(i+1)} = y_{(i)} + \alpha_{(i)} q_{(i)}$
    $r_{(i+1)} = r_{(i)} - \alpha_{(i)} \mathbf{K}\, q_{(i)}$
    $d_{(i+1)} = PC(r_{(i+1)})$
    $\beta_{(i)} = ((r_{(i+1)} - r_{(i)})^T d_{(i+1)}) / (r^T_{(i)} d_{(i)})$
    $q_{(i+1)} = d_{(i+1)} + \beta_{(i)} q_{(i)}$
END
```

inexact Patera algorithm solving S d = r (see text)

η^{-1}-scaled **pressure mass matrix (M$_\eta$) preconditioner**

```
REPEAT FOR i = 0,...,$n_M$ ; ($0<n_M<10$)
$d_{(i+1)} = d_{(i)} + (r - \mathbf{M}_\eta d_{(i)}) (M_{\eta L})^{-1}$
```

Cholesky factorization of K^n (K restricted to n-th MG level)

$\mathbf{L}\,\mathbf{L}^T = \mathbf{K}^n$

Geometric Multigrid on n levels

```
FOR m=1,...,n-1
    $\check{r}^m = r^m$
    FOR j=1,...,# relaxations
        $d^m = d^m + \omega (D^m)^{-1} \check{r}^m$
        $\check{r}^m = r^m - \mathbf{K}^m d^m$
    END
    $r^{m+1} = \mathbf{R}_{m\rightarrow(m+1)} \check{r}^m$
END
```

Cholesky solver on n-th level

```
$\mathbf{L}^T t = r^n$   forward substitution
$\mathbf{L}\, d^n = t$   backward substitution
```

```
FOR m=n-1,...,1
    $d^m = d^m + \mathbf{I}_{(m+1)\rightarrow m} d^{m+1}$
    FOR j=1,...,# relaxations
        $\check{r}^m = r^m - \mathbf{K}^m d^m$
        $d^m = d^m + \omega (D^m)^{-1} \check{r}^m$
    END
END
```

Figure 2.24: *Patera's algorithm to solve the viscous flow problem. Equation (2.145a) is formally solved for u and substituted into the incompressibility constraint (2.145b). The resulting equation $\mathbf{S}p = \hat{F}$ is solved using an outer CG algorithm (blue box). $\mathbf{S} = \mathbf{G}^\mathbf{T}\mathbf{K}^{-1}\mathbf{G}$ cannot be formed since it includes the unobtainable $\mathbf{K}^{-1}$. Every multiplication by $\mathbf{S}$ is done in three steps (brown box), during which a second (inner) CG algorithm simulates the action of $\mathbf{K}^{-1}$ (gray box). The inner CG is preconditioned by a single multigrid V-cycle (green box), which uses a direct (Cholesky) solver on the coarsest MG level (yellow box). The outer CG is preconditioned by an inexact Patera algorithm (red box; see text), which on its part uses Jacobi iterations on the inverse-viscosity scaled pressure mass matrix $\mathbf{M}_\eta$ for preconditioning (orange box; $\mathbf{M}_{\eta L}$ denotes the lumped $\mathbf{M}_\eta$).*

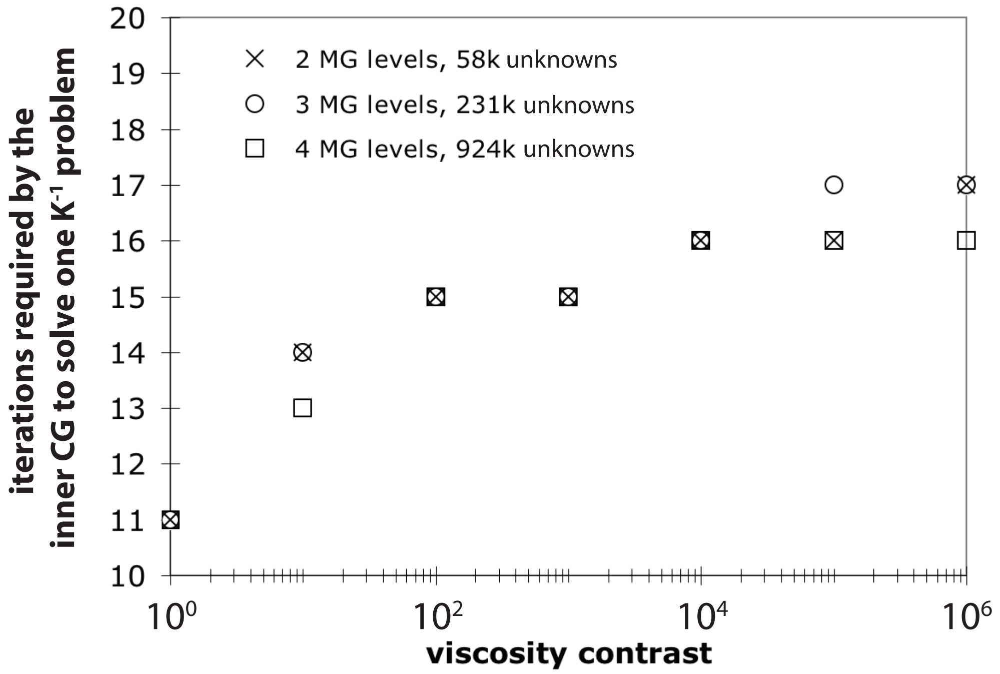


Figure 2.25: *Number of iterations of the inner CG to solve one* $\mathbf{K}^{-1}$ *problem, which is required in each outer CG iteration in the Patera algorithm. Sinker-problems are solved in 2-D, for different numbers of unknowns and different viscosity contrasts between the sinker and the surrounding material.*

only include spatial pressure derivatives, p is defined up to an arbitrary constant. This constant pressure represents a constant null space vector that does not affect the velocity solution, since only pressure gradients drive flow in incompressible media. Consequently, the Schur complement $\mathbf{S} = \mathbf{G^T K^{-1} G}$ contains a single zero eigenvalue and makes $\mathbf{S}$ a singular matrix.

Using an iterative CG solver requires the removal of the null space from the pressure field, because the initial CG convergence will eventually turn into a diverging behavior (van der Vorst, 2003). The null space can be removed in several ways: (1) The pressure can be anchored at one node by imposing a pressure boundary condition. We have tried this approach but observed a considerably reduced convergence rate of the CG — as reported by van der Vorst (2003). (2) Leaving a small region of the domain boundary open for an unconstraint flow-through also removes the null space, and is the recommended method (e.g. Bathe, 1996) for direct-solve formulations. These ”leaky nodes”, however, have the disadvantage of causing ups and downs during the pressure convergence, which also leads to a reduced convergence rate similar to (1). Instead, (3) the null space is removed by taking out the mean of the pressure solution in every CG iteration (May and Moresi, 2008). This operation is computationally cheap and has no negative influence on the CG convergence.

Performance and algorithm improvements

The finding that flow-through boundary conditions cause a sub-optimal CG convergence is very important for all applications: For best performance, boundary conditions for all velocity components normal to the domain boundaries have to be defined. These values have to be chosen such that a divergence-free flow field can exist within the domain. Defining these velocity boundary conditions can be challenging for complex geodynamic problems where material enters and leaves the domain in several regions. However, a simple solution exists: A flow-though boundary condition in some parts of the domain boundary can be used to obtain a divergence-free flow field in the first time step. This solution is from then on prescribed for the velocity components normal to the domain boundary. A slower CG convergence will be observed in the first time step, but an optimum convergence for all upcoming time steps is ensured.

Given that appropriate boundary conditions are imposed, the performance of the algorithm in Fig. 2.24 depends on two aspects: (1) the number of iterations that the inner CG requires to solve each K-inverse problem, and (2) the number of outer CG iterations, which defines the number of K-inverse problems that need to be solved.

The first problem is addressed using a multigrid algorithm to precondition the inner CG. CG in combination with multigrid performs much better here than in the penalty method, because no penalty terms increase the condition number of $\mathbf{K}$. The number of CG iterations for solving a single $\mathbf{K}^{-1}$-problem in 2-D is shown in Fig. 2.25. The test problem is the sinker-problem, in which the viscosity contrast and number of unknowns have been varied. The number of iterations shows nearly no dependence of the number of unknowns (which is very important for solving large-scale problems), and a tolerable dependency on the viscosity contrast.

The harder task is to find a preconditioner for the outer CG, because here the matrix $\mathbf{S}$ is essentially unknown. Recall that an efficient preconditioner $\hat{\mathbf{S}}^{-1}$ for the problem $\mathbf{S}p = \hat{F}$ has to be a good approximation to $\mathbf{S}^{-1}$. Only if this is achieved, a good estimate for the error $d_{(i)}$ of the pressure solution $p_{(i)}$ in iteration i can be calculated, by solving the equation $d_{(i)} = \hat{\mathbf{S}}^{-1}r_{(i)}$, where $r_{(i)} = \hat{F} - \mathbf{S}p_{(i)}$.

For isoviscous problems, the pressure mass matrix, defined by (2.142) on page 78, can serve as a reasonable preconditioner: $\hat{\mathbf{S}}_a^{-1} = \eta_0 \mathbf{M}^{-1}$, where η_0 is the viscosity of the fluid (Verführt, 1984). In problems with varying viscosity, however, the quality of this preconditioner is significantly reduced and can even lead to no convergence in the outer CG. A simple way to improve $\hat{\mathbf{S}}_a^{-1}$, is to scale each element's pressure mass matrix by the element's average inverse viscosity before assembling. The resulting global mass matrix $\mathbf{M}_\eta$ then includes the information on the viscosity field and can be used as a preconditioner, i.e. $\hat{\mathbf{S}}_b^{-1} = \mathbf{M}_\eta{}^{-1}$.

Before continuing the discussion and performance analysis of the above preconditioner, I would like to add a few remarks on other possible preconditioners for $\mathbf{S}p = \hat{F}$. Tests with the so-called *BFBt*-preconditioner $\hat{\mathbf{S}}_c = \mathbf{G^T G}$ (Elman et al., 2006) did not lead to viable results in combination with quadratic-order Taylor-Hood elements. The performance of this preconditioner, which corresponds to the discrete Laplacian defined on the pressure functional space, depends on the element type (May and Moresi, 2008) and seems to work better, when using discontinuous pressure elements. More tests with the *BFBt*-preconditioner in combination with Crouzeix-Raviart elements could be worthwhile. Another preconditioner, suggesting itself, results from using the reciprocal diagonal of $\mathbf{K}$ to approximate $\mathbf{K}^{-1}$, and use it to form an approximate $\mathbf{S}$: $\hat{\mathbf{S}}_d = \mathbf{G^T} diag(\mathbf{K})^{-1}\mathbf{G}$. Neither using the reciprocal diagonal of $\hat{\mathbf{S}}_d$ nor applying Jacobi iterations as described above led to satisfying results. Even actually inverting $\hat{\mathbf{S}}_d$ for a small Stokes flow problem was found to perform poorly in combination with Taylor-Hood elements. I will therefore focus on the the inverse-viscosity scaled mass matrix $\mathbf{M}_\eta$ in the following.

Once $\mathbf{M}_\eta$ is obtained, its inversion can still be too laborious in large-scale problems. The following approximate solutions can be used instead of ${\mathbf{M}_\eta}^{-1}$:

1. either the reciprocal diagonal of $\mathbf{M}_\eta$ or the inverse lumped mass matrix $\mathbf{M}_{\eta_L}^{-1}$ can be used to approximate ${\mathbf{M}_\eta}^{-1}$ (a lumped matrix is a vector containing the row-sum of the matrix, see Hughes (2000, pp. 444)).

2. a CG algorithm can be used to calculate the action of ${\mathbf{M}_\eta}^{-1}$ on the residual r; this would be very similar to the multiplications of $\mathbf{K} + \mathbf{K}^\gamma$ to a vector in the penalty method with Taylor-Hood elements (Fig. 2.21)

3. Jacobi iterations (see Eq. (2.125) on p. 51) can be used on the equation $\mathbf{M}_\eta\, d = r$, where r is the residual in the outer CG and d is the unknown approximate error for the current pressure solution. Either the diagonal of $\mathbf{M}_\eta$ or the lumped mass matrix $\mathbf{M}_{\eta_L}$ can be used to update d in each iteration. Starting with $d_{(0)} = r_{(0)}\mathbf{M}_{\eta_L}^{-1}$, repeat for a few iterations: $d_{(i+1)} = d_{(i)} + (r - \mathbf{M}_\eta\, d_{(i)})\, \mathbf{M}_{\eta_L}^{-1}$

All three methods have been tested, as well as actually inverting $\mathbf{M}_\eta$, and I find that the Jacobi iterations are the best option. It is important to keep in mind that ${\mathbf{M}_\eta}^{-1}$ is only used to approximate $\mathbf{S}^{-1}$, so that extensive computational efforts to calculate ${\mathbf{M}_\eta}^{-1}$ as precisely as possible are misspent. This is contrary to the role of $\mathbf{M}^{-1}$ in the penalty method, where the terms in $\mathbf{K}^\gamma$ are exactly defined and a precise evaluation of $\mathbf{M}^{-1}$ is required. Therefore, inverting $\mathbf{M}_\eta$ for preconditioning purposes is computationally too expensive. Using CG for only very few iterations on the problem $\mathbf{M}_\eta\, d = r$ has the disadvantage that the local error of the solution may not decrease: As opposed to Jacobi

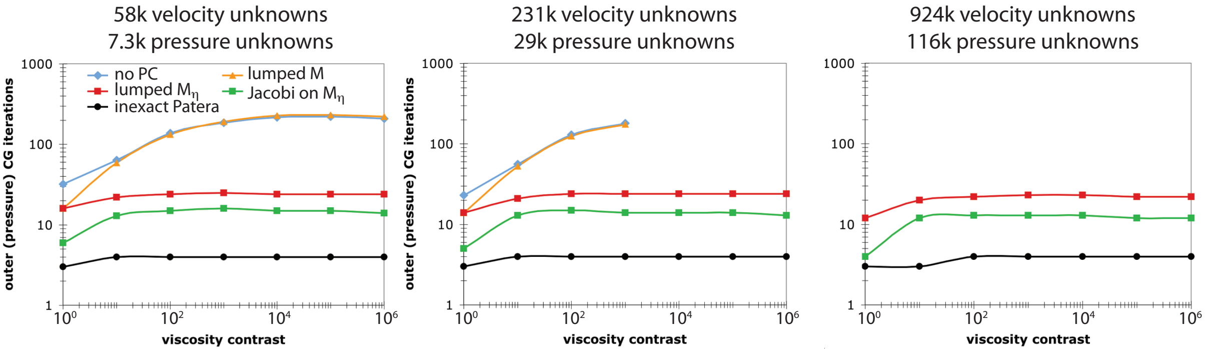


Figure 2.26: *Number of outer CG iterations to solve a sinker-problem for different number of unknowns and viscosity contrasts. The preconditioners approximating* $\mathbf{S} = \mathbf{G^T K^{-1} G}$ *are: "no PC" – no preconditioner* $(d = r)$*; "lumped* $\mathbf{M}$*" – lumped mass matrix, not including viscosity variations; "lumped* $\mathbf{M}_\eta$*" – lumped mass matrix scaled by inverse viscosity; "Jacobi on* $\mathbf{M}_\eta$*" - 5 Jacobi iterations on* $\mathbf{M}_\eta\, d = r$*; "inexact Patera" – calling the Patera algorithm to solve* $\mathbf{S}d = r$ *to a low tolerance (see text).*

iterations that smooth and reduce local disequilibria, CG iterations tend to reduce the global error (see sections 2.4.1 and 2.4.2). By doing so, local imbalances in the solution can increase temporarily so that, if stopped at a "bad moment", the incomplete CG solution can potentially be worse than the initial guess. Furthermore, $\mathbf{M}_\eta$ has a much larger condition number than $\mathbf{M}$, because it includes viscosity variations. Thus, 2–4 iterations as in the penalty method are not enough to evaluate ${\mathbf{M}_\eta}^{-1}$.

Of the two remaining options (inverse lumped mass matrix and Jacobi iterations on $\mathbf{M}_\eta\, d = r$), I favor the Jacobi iterations, because they tend to reduce the number of outer CG iterations more effectively than the lumped mass matrix for all Stokes flow problems tested so far. The computational effort for performing 3–5 Jacobi iterations is marginal compared to the computational costs of performing a single additional iteration of the outer CG (blue box in Fig. 2.24) – the latter involves solving a $\mathbf{K}$-inverse problem, which requires 10–20 multiplications by $\mathbf{K}$ (note that, in addition, $\mathbf{K}$ has considerably more nonzero terms than $\mathbf{M}$).

Fig. 2.26 shows the performance of the Patera algorithm when using some of the above preconditioners, in comparison to runs without any preconditioning and a preconditioning technique referred to as "inexact Patera", which will be introduced soon. Clearly, preconditioning of the outer CG is required and must include information on the viscosity field. Otherwise the number of outer iterations scales unfavorably with the number of unknowns and the viscosity contrast. The Jacobi iterations on the inverse-viscosity-scaled mass matrix are computationally inexpensive and perform slightly better than the inverse-viscosity scaled lumped mass matrix.

A characteristic of the Patera algorithm allows to further speed up its convergence by using "inexact Patera iterations". Recall that during each outer pressure iteration, an inner $\mathbf{K}^{-1}$-problem has to be solved. The solution of the latter is an essential part of the multiplication of $\mathbf{S}$ to a vector. Any roundoff error introduced at this point will eventually lead to convergence problems in the outer CG or, even more fatally, cause a convergence to erroneous flow and pressure fields.

These roundoff errors are only avoided, by solving the $\mathbf{K}^{-1}$ problem to an accuracy of at least the tolerance that is defined for the outer CG. The relative tolerance between the inner and outer CG depends among other things on the viscosity contrasts, in that stronger viscosity contrasts might require even higher relative accuracy for the $\mathbf{K}^{-1}$ calculations. This high accuracy is required in every iteration of the outer CG, because once roundoff errors have entered the pressure iterations, it cannot be removed, because CG choses each search direction only once (see Section 2.4.1).

The required high accuracy makes the inner CG very costly, especially during the beginning of the pressure iterations, when the corrections to the velocity field are larger and consequently more iterations are required until convergence. Fig. 2.27a shows the convergence of the residual r_p in the pressure iterations (outer CG). Each iteration requires a solution from the inner CG, whose convergence for every call is shown in Fig. 2.27b. Once the norm of r_p is below the defined tolerance (dotted line in Fig. 2.27a), the pressure solution is obtained, as well as the velocity field if updated using (2.150). In this example, the 19 outer iterations require a total of 227 iterations of the inner CG.

Clearly, if the number of outer CG iterations can be significantly reduced, the number of inner iterations will be reduced accordingly. To do so, a very exact approximation for $\mathbf{S}^{-1}$ is required so that $d = \mathbf{S}^{-1} r_p$ is very accurate. This can be achieved by calling the Patera algorithm itself to solve the problem $\mathbf{S}\, d = r_p$ to a tolerance that is about 1–2 orders of magnitude lower than the norm of the current pressure residual (I will refer to this preconditioning algorithm as the "inexact Patera" algorithm). In other words, given $||r_{(i)}|| = ||\hat{F} - \mathbf{S} p_{(i)}||$ in the i-th iteration of the outer CG, the inexact Patera algorithm is asked to solve the problem $\mathbf{S}\, d_{(i)} = r_{(i)}$ to a tolerance of $10^{-2} \cdot ||r_{(i)}||$. Using the resulting error estimate $d_{(i)}$ in the outer CG will correct the pressure such that the new residual $r_{(i+1)}$ in the next iteration is reduced by a factor of 10^{-2}, i.e. $||r_{(i+1)}|| = 10^{-2} \cdot ||r_{(i)}||$.

In the extreme case, where the tolerance for the inexact Patera is set to be the final tolerance of the outer CG, the error $d_{(i)}$ will correct pressure $p_{(i)}$ as precisely as required by the tolerance of the outer CG – it will converge in the next iteration. However, nothing would be gained from this, since the computational work has simply been moved from the outer CG to the preconditioning algorithm. Recall that solving $\mathbf{S}\, p = \hat{F}$ is equivalent to solving for the error d of p using $\mathbf{S}\, d = r$, where $r = \hat{F} - \mathbf{S}\, p$ (see Eq. (2.99a)-(2.99b) on page 39).

What is the advantage of calling an algorithm recursively to precondition itself, if

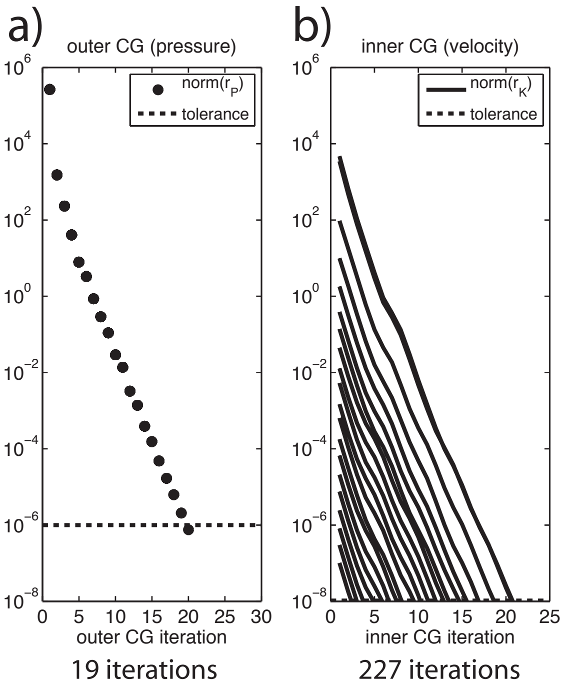


Figure 2.27: *(a): Convergence of the pressure residual (black dots are $||r_p|| = ||\hat{F} - \mathbf{S}\,p||$) in the outer CG. The dashed line shows the tolerance defined for the pressure solution. The CG is preconditioned using 5 Jacobi iterations on $\mathbf{M}_\eta\, d = r_p$. (b): Each pressure iteration requires the solution of a $\mathbf{K}^{-1}$ problem in the inner CG. The black lines show the convergence of the residual $r_K = y - \mathbf{K}z$ (see Fig. 2.24), i.e. there's a line for each dot in (a). A higher tolerance is required here (dashed line) to ensure a precise solution in the outer CG.*

the problem stays essentially the same? The answer is that in the Patera algorithm the accuracy required for the inner CG solution is directly related to the final accuracy defined for the outer CG. If called to solve the problem $\mathbf{S}d = r$ to a low tolerance (to precondition the algorithm solving $\mathbf{S}p = \hat{F}$), the inner CG in the inexact algorithm does not have to operate as accurately as required for the inner CG in the algorithm solving $\mathbf{S}p = \hat{F}$. Figure 2.28 shows this idea schematically. Of particular importance are the tolerances for the CG solutions.

The best performance is achieved, if the fewest total number of iterations for solving $\mathbf{K}^{-1}$ problems can be realized (sum in both exact and inexact Patera algorithms is meant here). The important parameter is ϵ, which defines the tolerance of the inexact algorithm relative to the current norm of the pressure residual. I find the best performance for $10^{-3} < \epsilon < 10^{-1}$. Using smaller values ($\epsilon < 10^{-3}$) causes the inexact algorithm to solve more or less the entire problem and nothing is gained, while $\epsilon > 10^{-1}$ results in very few CG iterations of the inexact Patera algorithm. In the latter case, the CG algorithm cannot display its full potential of solving the global problem rather than equilibrating locally.

The convergence when using the inexact Patera preconditioner is shown in Fig. 2.29.

The viscous flow problem solved is the same as in Fig. 2.27. The panels show the convergence of the norm of the residual in (a) the outer CG solving $\mathbf{S}p = \hat{F}$, (b) the inner CG solving a $\mathbf{K}^{-1}$ problem in every iteration of (a), (c) the outer CG in the inexact Patera algorithm solving $\mathbf{S}d = r_p$, and (d) the inner CG in the inexact algorithm called in every iteration of (c). The dotted lines show the tolerances defined for each of the four CG algorithms. A total number of 161 iterations is required to solve the 24 $\mathbf{K}^{-1}$ problems (4 in the exact outer CG, 20 in the inexact outer CG). Although more iterations are required in all outer CGs (24 instead of 19 iterations in Fig. 2.27), 66 fewer inner CG iterations are needed. These are saved in the inexact algorithm, due to lower tolerances in the beginning of the pressure iterations. Since the largest mathematical operation are multiplications by $\mathbf{K}$, the algorithm used in Fig. 2.29 is about 40% faster than the one in Fig. 2.27.

I want to add some final remarks on the Patera algorithm. The inexact Patera preconditioning represents a non-constant operator between d and r, so that I recommend again the *Polak-Ribière* formulation for β (2.132) to avoid convergence problems. The same is true for the inexact algorithm, that uses Jacobi iterations on its residual for preconditioning. Using the *Polak-Ribière* formulation seems to also make the outer CG algorithm more robust with respect to less accurate inner CG solutions. The relative accuracy between

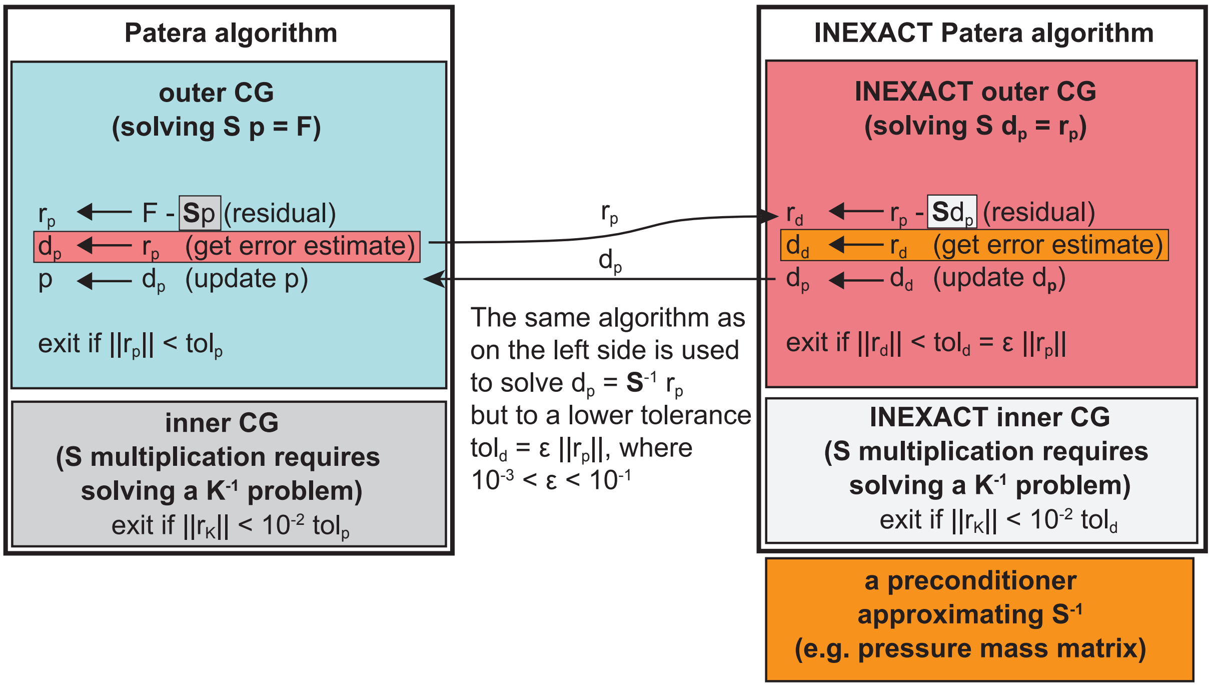


Figure 2.28: *Schematics of the recursive call of the Patera algorithm when used as an inexact preconditioner. The outer CG (blue box) requires high tolerance* $\mathbf{S}$*-matrix multiplications in every iteration. The number of outer CG iterations can be reduced, by calculating good approximations for the error* d_p *of the current pressure. The same algorithm is called in a recursion to evaluate* $\mathbf{S}\,d_p = r_p$ *to a tolerance, which is dynamically adjusted as the pressure solution converges.*

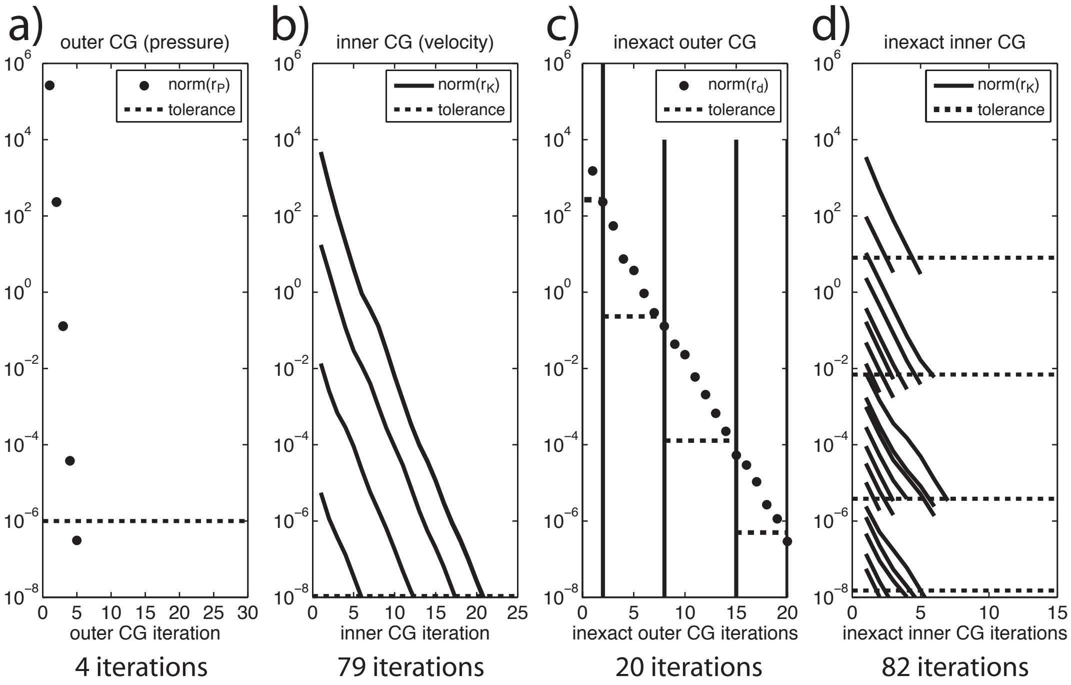


Figure 2.29: *(a): Convergence of the outer CG when using the inexact Patera algorithm to derive low-tolerance solutions for* $\mathbf{S}\,d = r$*. Fewer pressure iterations, thus, fewer high-tolerance inner CG solutions (b) are required. (c) The convergence in the inexact Patera algorithm, which is called four times in this example (separated by vertical lines). The tolerance (dashed lines) for each solution increase as the pressure residual* r_p *decreases. The corresponding* $\mathbf{K}^{-1}$ *solutions (d) can be less accurate as well, because they need a high accuracy with respect to the tolerances in (c). Instead of 227* $\mathbf{K}^{-1}$*-iterations (Fig. 2.27), only 161 are required here. The inexact Patera algorithm (c) uses the Jacobi-mass matrix preconditioner that was used in Fig. 2.27.*

the pressure problem and the velocity sub-problem is a critical parameter in the Patera algorithm. On the one hand, it should be as low as possible in order to keep the number of inner CG iterations at a minimum, because about 90-95% of the computational work is consumed by repeated **K**-multiplications. Saving 10% of inner CG iterations is almost equivalent to speeding-up the entire solver by 10%. On the other hand, using a too low accuracy in an attempt to improve performance will have negative consequences:

- Small roundoff errors, introduced during $\mathbf{S}\,q$-multiplications, can slow down the convergence of the outer CG. This leads to more outer iterations and more calls of the inner CG. Thus, trying to improve the performance by reducing the tolerance for the inner CG can easily result in the opposite effect: a slower convergence and more $\mathbf{K}^{-1}$ problems that need to be solved.
- Roundoff introduces error components into search directions that the outer CG has already worked on – in this case, convergence in the pressure iterations will stop and a restart of the outer CG is the best option (and using the last pressure solution as initial guess). Since all previous search directions are lost in a restart, the total

number of iterations of a repeatedly restarted CG is always larger than the number of iterations that a (roundoff-save) CG needs that keeps all information on previous search directions. Consequently, restarts cause more outer iterations and also slow done the overall performance.

- The initial residual $r = \hat{F} - \mathbf{S}p_{(0)}$ is equal to the divergence of the associated flow field $u_{(0)} = \mathbf{K}^{-1}F - \mathbf{K}^{-1}\mathbf{G}p_{(0)}$. If no roundoff error enters the outer CG, the residual is equal to the divergence of the flow field during all iterations, i.e. $\mathbf{G^T}u_{(i)} = r_{(i)} = \hat{F} - \mathbf{S}p_{(i)}$. If, however, inaccurate solutions for the $\mathbf{S}\,q$ multiplications enter the solution process, the residual might still converge (maybe at a slower rate, which would correspond to the first point in this list), but it is no longer associated with the divergence of the flow field. This was found accidentally, by explicitly calculating the divergence of the flow field in each iteration and observing, that the residual converged, while the divergence did not below a certain tolerance. If this happens, the pressure and velocity solutions obtained do not have a quality (in terms of incompressibility) as enforced by the tolerance on p.

No ad hoc rule for the relative tolerance of the inner CG with respect to the outer can be formulated here, but tests should be conducted to evaluate the divergence of the flow field after a solution has been obtained. If the divergence is larger than the tolerance defined for the pressure CG, roundoff errors are likely to have entered the solution process.

The largest numerical operation in the viscous flow problem is the iterative solution for $\mathbf{K}^{-1}$ problems, within which the multiplications by $\mathbf{K}$ are the most time consuming lines in the code. These multiplications are required in the CG algorithm (once per iteration) and in the multigrid algorithm for smoothing (here only the relaxations on the finest mesh are important, because multiplications by restricted $\mathbf{K}$ matrices are very fast due to their reduced size). The performance of the CG algorithms depends mainly on the quality of the preconditioning multigrid algorithm, which in turn relies on effective relaxations on each mesh (except for the coarsest mesh, where a direct solver is employed). Currently, a simple Jacobi (diagonal) smoother is used, because it has advantages in the parallelization, in that fewer and smaller messages have to be exchanged between subdomains. The Jacobi relaxations are improved by using different damping factors during the relaxations. However, the Jacobi smoother performs better in 2-D than in 3-D so that (as a rule of thumb) about twice as many CG iterations are required in 3-D compared to 2-D, when solving comparably complex flow problems.

The optimum parameter setting for achieving the fewest $\mathbf{K}$ multiplications in the CG solver is a compromise: More relaxations require more $\mathbf{K}$ multiplications, but lead to a better error estimate in the multigrid algorithm. The more accurate error reduces the number of CG iterations, thus, the number of calls of the multigrid algorithm. Using fewer relaxations speeds up each multigrid cycle, but results in more CG iterations. This trade-off is summarized in Fig. 2.30, where a CG solution for one $\mathbf{K}^{-1}$ problem is calculated. Different Jacobi relaxations (number and damping factors are varied) and different multigrid schemes have been used to precondition the CG. Few CG iterations are not necessarily equivalent to the fastest solution (e.g. 1st bar (41 CG iterations and 18 sec) compared to 4th bar (32 CG iterations, 24 sec), because the total number of $\mathbf{K}$ multiplications is important. The results also show that the relaxations on the intermediate mesh are a minor part of the total computation.

Interestingly, a single relaxation with $\omega > 1$, although at the risk of worsen the result, can help to improve the multigrid convergence. This "anti-damping"-factor should always be used with care and in combination with 1 or 2 relaxations with $\omega < 1$ to correct the result if necessary.

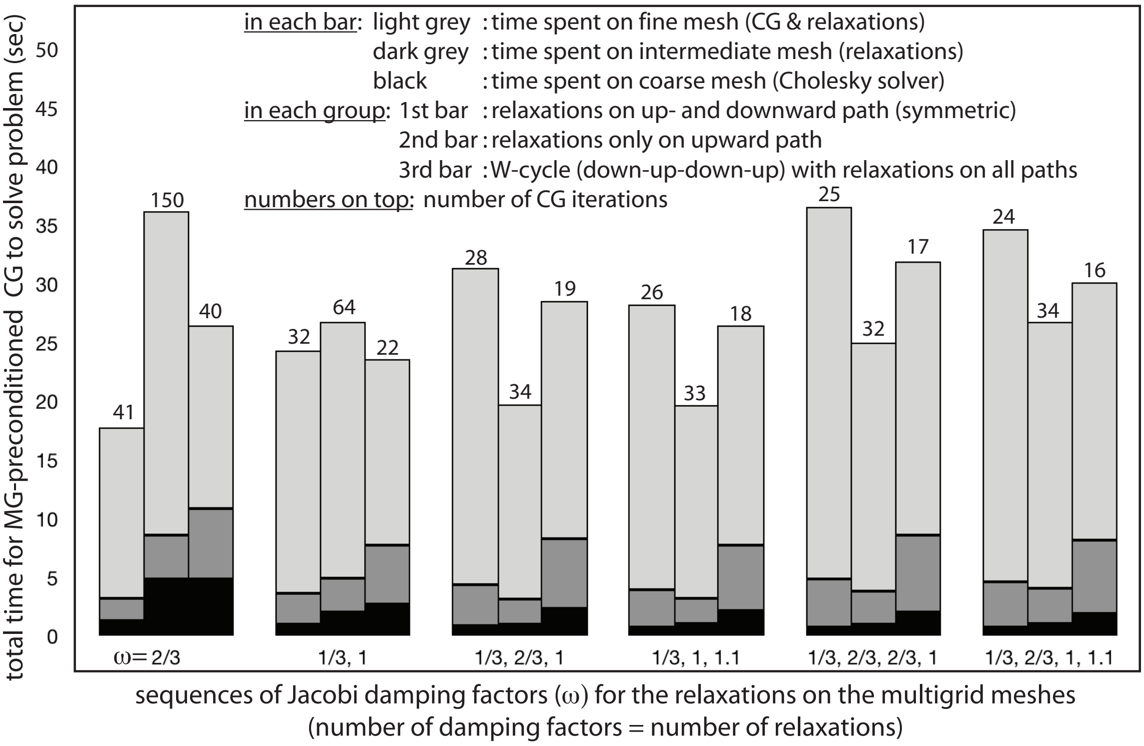


Figure 2.30: *Performance of the multigrid-preconditioned CG algorithm. Three MG levels are employed with a Cholesky solver on the coarse mesh. The solution of $\mathbf{K}u = F - \mathbf{G}p$ is calculated on a single CPU. The pressure solution p has been pre-calculated for this 3-D test problem, so that the velocity solution u can be calculated starting with a zero-guess (viscosity contrast=1000; number of velocity unknowns=824,000). 18 runs are compared, in which the number of relaxations and the Jacobi damping factors varied. The height of stacked bars is the total time for solving the problem; the numbers on top give the number of required CG iterations. The fastest solution is obtained for a single relaxation with $\omega = 2/3$. Asymmetric V-cycles (only relaxations on the upward path) require at least 3 relaxations to be comparable, so that they have no advantage over the symmetric V-cycle. A W-cycle reduces the number of CG iterations, but is comparably costly and always slower than the V-cycle.*

Similarly to the 1-D relaxation tests presented in Tab. 2.3 on page 56, more than 2-3 Jacobi relaxations do not considerably improve the quality of the MG result (number of CG iterations stays essentially the same). In contrast, more than 2 relaxations have the disadvantage to rapidly increase the total solution time, because they require more computations while not improving the quality of the error. This highlights the need for a more advanced smoother that equilibrates more velocity unknowns in each relaxation (e.g. block-diagonal smoothers in combination with Cornell-Macro elements; see below). Future work has to address this problem.

2.7 Summary

The finite element method was selected to solve the thermal diffusion as well as the Stokes flow problem, because it allows very flexible numerical meshes so that small-scale processes can be resolved in large-scale numerical domains. For instance, a ridge-melting process can require a spatial resolution of less than 1 km in order to correctly calculate the thermal evolution of the young lithosphere. The numerical domain, on the other hand, has to be large enough (few 100 km in all spatial directions) to reduce the influence of boundary conditions on the outcome of the numerical experiment.

The Galerkin finite element method results in symmetric, positive-definite matrix equations for the thermal diffusion problem and, when using segregated methods, for the velocity and pressure solutions in the Stokes flow problem. These matrix equations allow the usage of very efficient solvers that take advantage of the matrix symmetry, for instance, Conjugate Gradient algorithms or Cholesky direct solvers.

The consistent penalty method as well as Patera's algorithm have been tested for solving the velocity and pressure fields in viscous Stokes flow problems. The former method has been tested with two types of elements: Crouzeix-Raviart elements (discontinuous pressure shape functions), which do not perform well in combination with an iterative solver, and Taylor-Hood elements (continuous pressure shape functions), which require an approximation to the penalty matrix on all but the finest multigrid levels. Finding the optimum penalty number is difficult, as its value depends on the viscosity contrast and potentially on other parameters such as size, geometry and boundary conditions of problem the Stokes flow problem.

I therefore find Patera's algorithm to be better suited for solving large-scale problems in 2-D and 3-D. For all Stokes flow problems tested, it requires fewer total CG iterations for solving $\mathbf{K}^{-1}$ problems than the Taylor-Hood consistent penalty method. The performance can be improved by up to 50% when using an inexact Patera algorithm for preconditioning.

The latter is reasonably well preconditioned using Jacobi iterations on $\mathbf{M}_\eta d = r$, where $\mathbf{M}_\eta$ is the inverse-viscosity scaled pressure mass matrix.

The algorithm shown in Fig. 2.24 has been fully parallelized as described in sections 2.5.2 and 2.5.2, and is implemented in the 2-D and 3-D codes $\mathbf{M3_{tri}}$ and $\mathbf{M3_{tet}}$[8], respectively. These codes are used to investigate geodynamical problems in the next two chapters.

2.8 Outlook

First tests with so-called *Cornell-Macro-elements* (patent pending; developed by J. Phipps Morgan in collaboration with me during the code developments in this thesis) have been conducted. These macro elements combine a few Taylor-Hood elements into patches of a continuous velocity-continuous pressure (just like standard TH-elements). Between these macro elements, however, pressure is discontinuous, similar to pressure in the Crouzeix-Raviart elements. The macro elements have two major advantages in the Patera algorithm: First, the discontinuous pressure between macro elements increases the number of global pressure basis functions so that a better constraint ratio between pressure and velocity unknowns can be achieved. A higher constraint ratio (but not so high as to cause mesh locking!) leads to a more incompressible solution, because more incompressibility constraints are included. This ratio can be controlled by defining how many Taylor-Hood elements form a single Cornell-Macro element. Second, each macro-element's pressure mass matrix is complete, because it does not share pressure nodes with other macro-elements. This allows the inversion of the pressure mass matrix of each macro-element in the same way, as it is done for each Crouzeix-Raviart element. This also allows (1) the construction of better approximations to matrix $\mathbf{S}$, and (2) the development of block-smoothers for the multigrid in the $\mathbf{K}^{-1}$ problem: The Jacobi smoother could be replaced by an approximate inverse of a macro-element's stiffness matrix (or the inverse of the stiffness matrix for a group of macro elements). This would result in the equilibration of all velocities in a macro-element, rather than in single Taylor-Hood elements, in each relaxation.

[8] **M3** denotes **M**antle convection and **M**elting code written in **M**ATLAB (www.mathworks.com). Subscripts "tri" and "tet" refer to the triangular and tetrahedral elements that are used in the 2-D and 3-D version of the numerical code, resp.

Chapter 3

Mantle flow and melting at mid-ocean ridges

3.1 Introduction

In this chapter mantle flow and melting processes at mid-ocean ridges (MOR) are studied. First, a new formulation to parameterize melting of a multi-lithology mantle is introduced. The implications of this parameterization are tested using a 1-D decompression melting model that approximates the vertical upwelling of mantle rocks beneath the ridge axis.

This melting formulation is implemented in the 2-D and 3-D numerical models that have been described in the previous chapter. These models are used to study the feedbacks between mantle flow and melting at an idealized straight ridge axis as well as in the presence of transform faults (TF).

At the end of this chapter, a case study focussing on a particular melting anomaly at the Mid-Atlantic ridge near Ascension Island. Two possible origins of the melting anomaly are compared: (1) a weak mantle plume that interacts with the nearby ridge axis and (2) a heterogeneity in the mantle that leads to a high melt production as it is advected within the melting zone.

3.2 1-D model for pressure release melting of a multi-component mantle

3.2.1 Introduction

To gain basic insights into melting of a multi-component mantle beneath spreading centers, I first study a 1-D mantle upwelling and melting model that approximates a vertical profile underneath a mid-ocean ridge. It solves numerically for thermal advection and diffusion (using algorithms discussed in Section 2.2) as well as melting of a multi-lithology mantle (discussed below) and the advection of the compositional fields. Like all other codes developed in this thesis, the 1-D model is written in MATLAB.

The formulation of melting of a multi-component mantle is based on the thermodynamic relationships derived by Phipps Morgan (2001). The mantle structure is assumed to be composed of several lithologies (also referred to as mantle components) such as depleted peridotite (DP), fertile peridotite (FP, also referred to as Pyrolite), and pyroxenite (PYX) as a proxy for an enriched mantle component. The fertile and enriched components are assumed to be distributed as veins within a matrix of depleted peridotite, with the latter representing the largest lithological unit in the mantle. This ”marble-cake” (Allegre and Turcotte, 1986) or ”plum-pudding” (Phipps Morgan and Morgan, 1999) mantle composition is a likely candidate for the present day mantle composition: Continuous melt extraction at mid-ocean ridges leads to separate lithologies (mid-ocean ridge basalts and residual DP, depleted to different degrees) that are likely to not be re-blended by mantle convection. Together with smaller amounts of sediments (both oceanic and continental) and ocean-island basalts (OIB), these lithologies are permanently injected into the mantle during the subduction of oceanic lithosphere. Convective mantle stirring over millions of years might be effective in crushing and shearing these different lithologies, but it is unlikely that the mechanical re-homogenization process can lead to a single lithology with uniform composition down to mineral-scale (e.g. Schmalzl and Houseman, 1996; van Keken and Zhong, 1999). Instead, the mechanical mixing subsides as the lithological units become smaller. Chemical diffusion, on the other hand, is sufficiently slow in the mantle to preserve chemical disequilibrium between the mantle components for billions of years (Philpotts and Ague, 2009, p. 125). The interpretation of seismic scattering data supports the existence of mantle heterogeneities, and most of the scatterers are supposedly smaller than about 4 km (Helffrich and Wood, 2001).

Veins of distinct mantle components are assumed to be in thermal equilibrium with surrounding mantle rocks. That is, thermal equilibrium is assumed before and, most importantly, during the melting process. This idealization is reasonable as long as the time scale linked to the mantle upwelling rate is smaller than the time scale of thermal diffusion between veins. With the thermal diffusivity of mantle rocks ($\kappa = 10^{-6}\,\mathrm{m^2 s^{-1}}$) and a mantle upwelling rate beneath a slow-spreading ridge of ($u_z = 30\,\mathrm{mm\,yr^{-1}}$) the length scale, where thermal diffusion will be able to equilibrate neighboring material is

$$L = \frac{\kappa}{u_z} \approx 1\,km \tag{3.1}$$

Thus, as long as the veins are thinner than about 1 km, thermal diffusion will equilibrate temperature between veins and surrounding rocks, for instance, if the veins melt while the matrix is still in sub-solidus conditions. A chemical disequilibrium, however, persists during the melting process: For a chemical diffusivity of $10^{-12}\,\mathrm{m^2 s^{-1}}$ (an average chemical diffusion coefficient for iron, taken from the compilation of laboratory data in Philpotts and Ague (2009), p. 125), (3.1) implies a chemical equilibrium over 1 mm distance under conditions where there will be thermal equilibrium over 1 km distances. Hence each mantle lithology can be assumed to change its composition independently from its surrounding material as it depletes during melt extraction.

The vertical velocity on the 1-D profile is prescribed to be an idealized vertical mantle upwelling underneath the ridge axis. For a given half-spreading rate, the mantle upwelling speed underneath a ridge depends on rheological parameters (see the 2-D and 3-D numerical model calculation in Sections 3.3.3 and 3.3.4, resp.). These parameters control the shape of the lithosphere as well as the competing forces of buoyantly driven flow and viscous resistance. A good first approximation, however, is to assume that the average vertical velocity beneath the ridge is roughly the half-spreading rate of the ridge. As the mantle rises adiabatically and surrounding pressure drops, decompression melting will start as soon as a lithology crosses its solidus temperature. This is the starting point for the 1-D experiments.

Table 3.1: *List of variables, their units and values.*

variable	meaning, reference	value, units
x	spatial coordinates	km
t	time	Myr
u	velocity	$\mathrm{km\,Myr^{-1}}$
p	pressure	Pa
T	temperature	°C
T_0	reference temperature	1315°C
κ	thermal diffusivity	$10^{-6}\,\mathrm{m\,s^{-2}}$
Q	latent heat	J
τ	deviatoric stress tensor	Pa
g	gravitational acceleration	$\mathrm{m\,s^{-2}}$
e_z	unit vector in vertical direction	1
$\left(\frac{\partial T}{\partial t}\right)_{\mathrm{diff}}$	diffusive change in temperature	$°\mathrm{C\ Myr^{-1}}$
T^s	solidus temperature	°C
$\frac{\partial T^s}{\partial P}$	solidus-pressure (depth) dependence	$°\mathrm{C\ Pa^{-1}}$
$\frac{\partial T^s}{\partial F}$	solidus-depletion dependence	°C
$\frac{\partial F_i}{\partial P}$	melt productivity of component i (per unit of decompression)	$\mathrm{Pa^{-1}}$
ΔS	entropy change during solid-to-melt phase change	$\mathrm{J\ K^{-1}}$
Λ	parameter (Katz et al., 2003)	$43°\mathrm{C\ wt\%}^{-\gamma}$
γ	parameter (Katz et al., 2003)	0.75
D_{H_2O}	partition coefficient of water (Hirth and Kohlstedt, 1996)	0.01
η	viscosity (all are dynamic viscosities)	Pa·s
η_C	viscosity of lithology c	Pa·s
η_0	reference viscosity	$3{\cdot}10^{18}$–10^{19} Pa·s
E_A	activation energy	$400\,\mathrm{kJ\,mol^{-1}}$
V_A	activation volume	$4{\cdot}10^{-4}\,\mathrm{cm^3\,mol^{-1}}$
c_p	specific heat	$\mathrm{J\ K^{-1}}$
R	universal gas constant	$8.314\,\mathrm{J\,mol^{-1}\,K^{-1}}$
$T_{(K)}$	temperature	K
A_X	dehydration effect on viscosity	see (3.18b)
B	melt effect on viscosity	see (3.18c) and (3.18d)
$X0$	bulk water content in mantle	ppm
$X0_{ol}$	water content at which max. viscosity is reached	ppm, see (3.18b)
$\delta\eta_x$	max. viscosity increase due to dehydration	see (3.18b)
X_{ol}	water content in olivine	ppm

ρ	density	$\mathrm{kg\,m^{-3}}$
ρ_M	bulk density of mantle	$\mathrm{kg\,m^{-3}}$
ρ_C	density of lithology c	$\mathrm{kg\,m^{-3}}$
$\xi_{\#}$	volume fraction of mineral #	see Tab. 3.3
$\rho_{\#}$	density of mineral # ($\# = plg$, gt, sp, ol, opx, or cpx)	see Tab. 3.3
V_C	volume fraction of a lithology	1
C_C	mineral composition	see text
F_C	degree of melting of a mantle component (=depletion of a mantle component)	1
X_C	water content of a mantle component	ppm
X_{C0}	initial water content of a mantle component	ppm
ϕ	melt fraction in mantle rock (porosity)	1
Φ	melt flux in vertical direction	$\mathrm{km\ Myr^{-1}}$
α	thermal expansion coefficient	$2.5^{-5\circ}\mathrm{C}^{\ -1}$
β	depletion buoyancy parameter	0.3
M_B	bulk melting rate ($=dF/dt$)	$\mathrm{Myr^{-1}}$
M_{DP}	DP melting rate (same for other lithologies)	$\mathrm{Myr^{-1}}$
K	permeability	$\mathrm{m^2}$
z	vertical coordinate	km
w	Darcy velocity	$\mathrm{m\,s^{-1}}$
ρ_m	density of melt	$2970\,\mathrm{kg\,m^{-3}}$
η_m	viscosity of melt	$5\,\mathrm{Pa{\cdot}s}$
b	grain size in mantle	$0.3\,\mathrm{mm}$

3.2.2 Model description

Thermal evolution and melting

The energy conservation in 1-D is formulated as

$$\frac{\partial T}{\partial t} = \kappa\left(\frac{\partial^2 T}{\partial z^2}\right) - u_z\frac{\partial T}{\partial z} + Q \qquad (3.2)$$

with T denoting temperature, κ thermal diffusivity, z the vertical spatial coordinate and u_z the vertical velocity. Q is the latent heat that describes the energy consumed by the solid-to-liquid phase change. The terms on the right-hand side (RHS) are thermal diffusion, thermal advection, and the source term.

Based on the small-scale mantle heterogeneity model discussed above, the melt productivity ($\partial F/\partial P$), describing the change in degree of melting per increment of decompression, can be calculated for every mantle component. Following Phipps Morgan (2001), the melt productivity of lithology i, situated within other lithologies j that each have a mass fraction W_j, is described by

$$-\frac{\partial F_i}{\partial P} = \; = \frac{\frac{\partial T_i^s}{\partial P} + \left(\frac{\partial T}{\partial t}\right)_{\text{diff}} + \frac{T}{c_p}\sum_j W_j \Delta S_j \left(\frac{\partial T_i^s}{\partial P} - \frac{\partial T_j^s}{\partial P}\right)}{\frac{T}{c_p}\left[W_i \Delta S_j \left(\frac{\partial T_i^s/\partial F_i}{\partial T_j^s/\partial F_j}\right)\right] + \frac{\partial T_i^s}{\partial F_i}} \tag{3.3}$$

Eq. (3.3) has been modified from the eq. 29 in Phipps Morgan (2001) in two aspects: (1) The adiabatic term is missing, because all temperatures in the 1-D model (as well as in the 2-D and 3-D models) are potential mantle temperatures, and (2) the addition of heat from thermal diffusion $\left(\frac{\partial T}{\partial t}\right)_{\text{diff}}$ is included in the numerator. The other terms in (3.3) are the solidus-pressure dependence ($\partial T^s/\partial P$), the solidus-depletion dependence ($\partial T^s/\partial F$), and the change in entropy associated with the solid-to-melt phase change (ΔS). The values of the thermodynamic properties for each lithology considered in this study are listed in Tab. 3.2. Eq. (3.2) and (3.3) are related by the consumption of latent heat: The change in temperature during melting over a decompression interval dP is given by (Phipps Morgan, 2001)

$$\frac{dT}{dP} = \frac{\partial T_i^s}{\partial P} + \frac{dT_i^s}{dF_i}\frac{dF_i}{dP} \tag{3.4}$$

To include the water effect on the solidus function of each lithology, I have implemented a wet melting parameterization (Katz et al., 2003) by modifying the solidus depletion dependence ($\partial T^s/\partial F$) in Eq. (3.3). It is assumed, for simplicity, that peridotite as well as pyroxenite phases can be parameterized in the same way. The water content in each mantle component during melting is calculated based on (a) the partition coefficient of water between solid and melt $D_{H_2O} = 0.01$ (Hirth and Kohlstedt, 1996), (b) the initial water content X_{C0} of the mantle component, and (c) the current degree of melting F_C of the component. Assuming fractional melting the remaining water in the solid is given by (Shaw, 1970)

$$X_C = X_{C0} * (1 - F_C)^{\left[\frac{1}{D_{H_2O}} - 1\right]} \tag{3.5}$$

The derivative of (3.5) with respect to F_C gives the change in water content X_C with F_C (for the current degree of melting):

$$\frac{\partial X_C}{\partial F_C} = -X_{C0}\frac{1}{D_{H_2O} - 1} * (1 - F_C)^{\left[\frac{1}{D_{H_2O}} - 2\right]} \tag{3.6}$$

For each component the change in the solidus temperature ΔT_C^s in the presence of an amount of water X_C is approximated by (Katz et al., 2003)

$$\Delta T_C^s(H_2O) = \Lambda \cdot X_C^{\gamma} \tag{3.7}$$

$$\text{where} \quad \Lambda = 43\,^{\circ}C\, wt\%^{\,-\gamma} \quad \text{and} \quad \gamma = 0.75$$

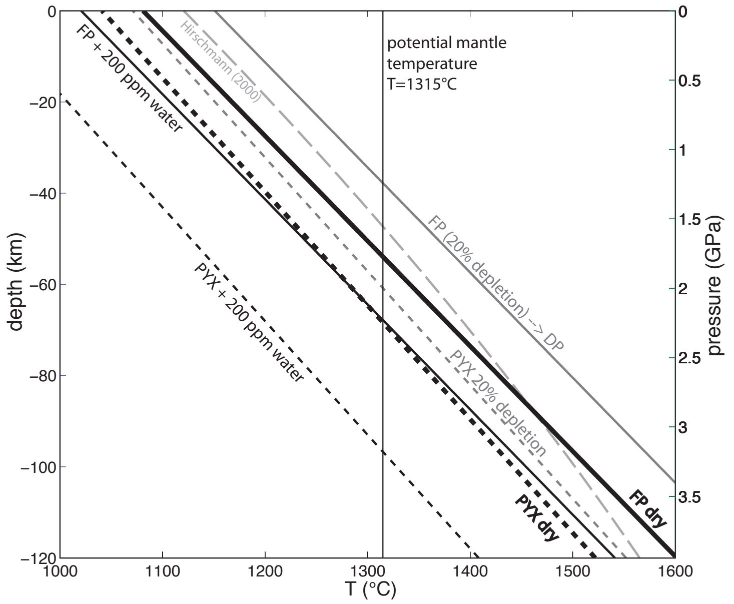


Figure 3.1: *Solidus functions used in this study. Short-dashes lines: pyroxenite (PYX) under dry and wet (200 ppm water) conditions, and if 20% depleted. Solid lines: fertile peridotite (FP) under dry and wet (200 ppm water) conditions, and depleted peridotite DP (residue after melting FP to 20%). The upper mantle-solidus parameterization by Hirschmann (2000), which is used frequently in numerical studies, is shown for comparison (long-dashed line).*

The derivative of (3.7) with respect to X_C is

$$\frac{\partial T^s_C}{\partial X_C} = \gamma \cdot \Lambda \cdot X_C^{\gamma - 1} \tag{3.8}$$

The term required to include the water effect in equation (3.3) has to have the form of a solidus-depletion dependence, which is obtained by the product of (3.8) and (3.6)

$$\left(\frac{\partial T^s_C}{\partial F_C}\right)_{\mathrm{H_2O}} = \frac{\partial T^s_C}{\partial X_C} \cdot \frac{\partial X_C}{\partial F_C} \tag{3.9}$$

This water-related ”depletion” dependence of the solidus is superimposed onto the composition-related depletion dependence for all components (the latter are listed in Tab. 3.2). Fig. 3.1 shows the solidus functions that are used in all following numerical calculations. Also shown are the solidus changes associated with the presence of water and increasing depletion.

The equation for energy conservation (3.2) is solved by operator splitting, that is, the advection and diffusion parts are solved separately during each time step. The finite element method with an implicit time stepping is used to solve for the thermal diffusion part of (3.2).

The advection of mantle components is done by advecting their respective volume

Table 3.2: *Mineral composition, thermodynamic properties and depth of gt-sp and sp-plg phase transition for all lithologies used in this study. FP=fertile peridotite (Pyrolite), DP=depleted peridotite, PYX=pyroxenite. For mineral properties see Tab. 3.3.*

lithology	associated rock	T^s(P=0) (°C)	$\partial T^s/\partial P$ (°C /GPa)	$\partial T^s/\partial F$ (°C)	*ol, opx, cpx, gt* (vol%)	*gt-sp* (km)	*sp-plg* (km)
FP	lherzolite	1081	132	350	56, 20, 12, 12	64	21
DP	harzburgite	1116	132	350	70, 24, 3, 3	79	21
PYX	basalt, eclogite	1041	122	150	1,1,49,49	49	21

fraction and depletion:

$$\frac{\partial V_C}{\partial t} = -u_z \frac{\partial V_C}{\partial z} \qquad \text{(advection of volume fraction)} \tag{3.10}$$

$$\frac{\partial F_C}{\partial t} = -u_z \frac{\partial F_C}{\partial z} \qquad \text{(advection of depletion/degree of melting)} \tag{3.11}$$

Thermal advection as well as Eq. (3.10) and (3.11) are determined using a semi-Lagrange method, which requires to evaluate the variables to be advected at so-called back-tracking points (BT) that are located between finite element nodes. A high-precision 1-D spline-interpolation (built into MATLAB) is used for this purpose, because accurate interpolation helps to reduce numerical diffusion to a minimum. This is of particular importance for the advection of non-diffusive properties such as depletion and volume fractions of the mantle components, because they strongly influence the melting process.

As discussed in Section 2.2.2, the source term Q in (3.2) is accounted for during the advection step rather than in the right-hand side of the thermal diffusion equation. This has the advantages of advecting latent heat immediately and avoiding instabilities in the thermal diffusion solution, if the source term varies over time and space.

The numerical calculations in a single time step are described by the following sequence:

1. Calculate the coordinates of the back-tracking points (BTs).

2. Solve implicitly for the diffusive change in temperature over the current time step, that is, the temperature time derivative $\left(\frac{\partial T}{\partial t}\right)^{n+1}_{\text{diff}}$ at the end of the time step. Note that this diffusion is static and does not include any advection.

3. The temperature time derivative associated with heat conduction at the beginning and the end of the time step, $\left(\frac{\partial T}{\partial t}\right)^{n}_{\text{diff}}$ and $\left(\frac{\partial T}{\partial t}\right)^{n+1}_{\text{diff}}$ resp., are interpolated at the BTs[1]. The average of both is used to approximate the diffusive change in temperature over the time step: $\left(\frac{\partial T}{\partial t}\right)_{\text{diff}} = 0.5 \left(\frac{\partial T}{\partial t}\right)^{n}_{\text{diff}} + 0.5 \left(\frac{\partial T}{\partial t}\right)^{n+1}_{\text{diff}}$.

[1] $\left(\frac{\partial T}{\partial t}\right)^{n}_{\text{diff}}$ has to be evaluated at the points, where the "nodal" material is located at the beginning of the time step. These are the BTs of all nodes. $\left(\frac{\partial T}{\partial t}\right)^{n+1}_{\text{diff}}$ for the end of the current time step is also interpolated at the BTs, because it was calculated statically using the nodal values in step (1). Since it represents values at the end of the time step, it has to be advected "away" from the nodes. This is done by interpolating $\left(\frac{\partial T}{\partial t}\right)^{n+1}_{\text{diff}}$ at the BTs.

4. Decompression melting of all lithologies is calculated using (3.3). $\left(\frac{\partial T}{\partial t}\right)_{\text{diff}}$ from step (3) is used as a source term that can enhance or reduce melt production by in- or outflow of heat, respectively. Volume fractions V and degree of melting F of all lithologies have to be evaluated at BTs in order to get their values at the beginning of the time step. The calculated values are defined at the nodes at the end of the time step.

5. The new nodal temperature depends on whether or not melt is produced: In case of melting and latent heat consumption along the advection path from BT to the node, the new temperature at the end of the time step is given by

$$T^{n+1} = T^n_{BT} + \frac{dT}{dP} \cdot dP \tag{3.12}$$

where (dT/dP) is given by (3.4). In the absence of melting, the new temperature depends solely on the thermal advection and diffusion:

$$T^{n+1} = T^n_{BT} + \left(\frac{\partial T}{\partial t}\right)_{\text{diff}} \cdot dt \tag{3.13}$$

In case of melt production along the advection path, the term $\frac{dT}{dP} \cdot dP$ includes the in- or outflow of heat, i.e. $\left(\frac{\partial T}{\partial t}\right)_{\text{diff}}$, through Eq. (3.3), which enters (3.4). All additional heat that flows into the mantle volume on its path from BT to the node will lead to more melt production. This heat is consumed by the melting process (e.g. transformed into latent heat of melting) and will not increase the mantle temperature at the end of the time step, as long as there is material left that is able to melt. At sub-solidus conditions, heat input between BT and node will increase the nodal temperature at the end of the time step. This has major consequences for shallow mantle temperatures and will be further discussed in the first 1-D experiments (see below).

Using this algorithm, no under-relaxed (damped) iterations between the solutions for thermal diffusion and melting are required to derive a consistent and oscillation-free solution for temperature and melting rate. This was a problem for the method introduced by Jha et al. (1994), especially for 2-D and 3-D coupled melting-thermal evolution problems (Phipps Morgan, *personal comm.*). The 2-D and 3-D numerical codes developed in this thesis use the same sequence as above to calculate the interaction of thermal evolution and melting. For this purpose, I have developed a vectorized, multi-dimensional extension of the melting formulation (3.3)-(3.4). To ensure accurate interpolations in 2-D and 3-D, a cubic spline-like interpolation scheme is used that operates on unstructured meshes (Shi and Phipps Morgan, 2010).

A simple mineralogical model

Melt production in the Earth's mantle depends on the thermal state and the decompression rate (see above), but is also strongly controlled by the mantle composition, namely by the abundance of minerals and their melting temperature. Because minerals differ in their physical properties (e.g. viscosity and density), they also affect viscous flow of the mantle, which in turn has a feedback on decompression melting. Small sub-pieces of the extremely complex mineralogical system can be approximated using thermodynamic codes like MELTS (Ghiorso and Sack, 1995) or PERPLEX (Connolly and Petrini, 2002). These codes can be used to describe the stability fields of the considered minerals as a function of pressure (P), temperature (T), and composition (C). Mantle convection models, in which bulk composition and P-T conditions are known at discrete points, can be coupled to look-up tables produced by these codes in order to estimate the mineral composition within the domain. Rheological parameters and the melting process can then be linked to the mineral composition to obtain a more consistent (though very complex) numerical model that covers the major processes in the Earth's mantle.

With the aim to prepare the newly developed 2-D and 3-D codes for a coupling to thermodynamic data in the future, a very simple mineralogical system is introduced next. It is assumed that each mantle lithology contains four minerals that represent the lion's share of minerals in the upper mantle: olivine (*ol*), orthopyroxene (*opx*), clinopyroxene (*cpx*) and an aluminum-bearing mineral, which can be either garnet (*gt*), spinel (*sp*), or plagioclase (*plg*), depending on the stability field. The mantle lithologies are assumed to differ in their initial mineral budget: A pyrolite-like mineral composition (see Tab. 3.2) is used for the fertile peridotite (FP). The initial mineral fractions for depleted peridotite (DP) and pyroxenite (PYX) are calculated by ”pseudo-melting” the FP, as described next.

Typical mid-ocean ridge basalts (MORB) contain large amounts of *cpx* and *plg*, because these minerals (or their high-pressure cousins) melt first in a composite of the above four minerals. Every increment of melt extracted from FP is therefore strongly enriched in these minerals. In the ”pseudo-melting” calculation, the minerals *ol*, *opx*, *cpx*, and *gt* are extracted from FP in an arbitrarily defined ratio (1:1:49:49). By removing increments of melt with this mineral composition from the FP and accumulating them, a crude approximate composition of the melt and the residual peridotite (DP) can be calculated.

The melt composition provides the mineral ratio in PYX, because it is representative for a former MORB that re-entered the mantle by subduction. The mineral content in the residue is used as the initial composition of DP. Using this very simple melt extraction parameterization, PYX will always have mineral ratios of 1:1:49:49 (since this is the melt composition), until all *gt* and *cpx* has been removed from the FP.

Note that the sole purpose of the above ”pseudo-melting” is to calculate the initial

Table 3.3: *Mineral densities and water partition coefficients with respect to olivine. sp*: modified spinel density to include ol to opx reaction at gt to sp phase transition. References: Ox=Oxburgh and Parmentier (1977), Ph09=Philpotts and Ague (2009), Ba95=Bass (1995)*

mineral	abbreviation	density (kg/m^3) and reference	H_2O partition coefficient (relative to OL)
olivine	*ol*	3320 (Ox77)	1
orthopyroxene	*opx*	3300 (Ox77)	0.2
clinopyroxene	*cpx*	3250 (Ox77)	0.1
garnet	*gt*	3670 (Ox77)	1
spinel	*sp*	3578 (Ba95)	1
spinel*	*sp**	3260 (see text)	1
plagioclase	*plg*	2700 (Ph09)	1

mineral composition of the mantle lithologies. The reason for calculating the mineral composition rather than defining them arbitrarily for each component is to maintain the same pyrolite-like bulk composition of the mantle, independent of the number of lithologies that are considered in an experiment. In other words, the bulk composition of FP is identical to that of a two-component mantle with DP+PYX, and also identical to that of a three-lithology mantle composed of DP+PYX+FP. If an initial bulk composition differs from a pyrolite, it will be explicitly mentioned in the text.

Including minerals in the melting process allows to account for the so-called *cpx*-out effect. If a mantle component runs out of *cpx*, the solidus-depletion gradient suddenly increases (e.g. Hirschmann et al., 1998), because the component is exhausted in the major mineral that is easiest to melt. At this point the lherzolite turns into a harzburgite, which is much harder to melt (it is also called a refractory peridotite in literature). For all lithologies that reach this point, the $(\partial T_C^s/\partial F_C)$-term is increased by a factor of 10. The same factor of 10 increase is considered, if a lithology enters the *plg*-stability field. Here, *sp* is transformed into *plg*, which is less fusible.

The above modifications of the solidus-depletion gradient are approximations to the thermodynamic effects that occur in the sub-ridge melting region. A more accurate parameterization would require a thermodynamic code. MORB usually contains large amounts of *cpx* and *plg*, but only small amounts of *opx* and almost no *ol* — this is crudely approximated by only melting sp/gt and *cpx*, while *ol* and *opx* remain in the solid. If all *cpx* is melted, the lithology becomes refractory and melting rates drop (i.e. *cpx*-out effect). Furthermore, MORB does not show a strong europium anomaly that would exist, if much melting was happening in the *plg*-lherzolite stability field (Philpotts and Ague, 2009, pp. 356).

The mineral contents of the lithologies, together with the mineral densities (see Tab. 3.3), allow us to calculate the compositional density of each mantle component. The mantle density changes at phase transitions, and also as mantle components evolve and change their mineral composition during melt extraction. While the upwelling velocity is prescribed in the 1D model (see above) so that density changes have no effect on the mantle flow, the compositional densities can be included in the 2D and 3D models to study these potential effects on the mantle flow (see the discussion in the subsequent sections).

Two phase transitions in the uppermost mantle are included that involve some of the minerals above. It is assumed that *gt* breaks down at about 64 km depth and reacts with *ol* to form *opx* and *sp*. This phase transition is of interest because the products of the reaction are less dense then the reactants. The actual amount of *ol* that participates in the reaction depends on the exact mineral composition and a precise evaluation would require a thermodynamic solution. For simplicity, I assume that all *gt* is converted into *sp*. In order to account for the additional density reduction of the *ol* to *opx* reaction, the *sp* density has been modified (see "modified *sp* density" in Tab. 3.3).

In reality, the depth of the *gt*-*sp* phase change depends on the surrounding mantle temperatures (cf. Asimow et al., 2004), an effect that is neglected here for simplicity so that the phase change occurs at a prescribed depth. However, the depth of the phase change also depends on the iron content of the lithology (Phipps Morgan, *personal comm.*), because iron tends to stabilize garnet, so that its breakdown would be delayed to a depth. As a consequence, iron-rich lithologies such as the FP and enriched PYX are likely to have shallower *gt*-*sp* transitions. The depth of the *gt*-*sp* phase change is chosen such that it reflects the fertility of the component (see Tab. 3.2).

The second phase transition is defined at 21 km depth. Here, *sp* and *cpx* react to form *plg* and *ol*. Again the reaction is simplified in that all *sp* present at this depth is turned into *plg*, because an exact solution for how much *cpx* and *ol* are involved would require a thermodynamic treatment. The *sp*-*plg* phase transition is of great importance for the melting process, because *sp*-lithologies have lower solidus temperatures than *plg*-lithologies. Thus, this phase transition has a similar impact on the melt productivity to the *cpx*-out effect. The latent heat cooling of the two above endothermic phase transitions is neglected in the energy equation (3.2), because the cooling effects are demonstrated to have only a minor influence on the overall melt productivity ($< 2\%$ change in melt production; Phipps Morgan, 2001).

As described above, the effect of water on lowering the solidus temperature of each mantle component is considered. The initial water content of a lithology is defined using its initial mineral content. First, a bulk water content of the mantle is defined (that is, the water content of the pyrolite-like fertile peridotite). Using the partition coefficients of water between the four minerals suggested by Hirth and Kohlstedt (1996) (see also Tab. 3.3), the water concentration within each mineral can be calculated. A chemical equilibrium in water content between the minerals is established when the water concentrations in the minerals are inversely proportional to the partition coefficients, e.g. if *cpx* stores a 10-fold higher amount of water compared to *ol*.[2] Together with the mineral composition of each mantle component we can calculate the water content of each lithology. As a consequence of the above formulation, a pyroxenite equilibrated in terms of water content with surrounding peridotite, will have a higher net water content than its neighboring peridotite (because PYX contains a higher fraction of *cpx*, which - at chemical equilibrium - stores more water than *ol*). It also allows to calculate a water-dependent rheology for each lithology, which will be used later to determine an effective aggregate rheology for the multi-component mantle.

The assumption that water concentrations are initially equilibrated is reasonable, because hydrogen is an exception to the otherwise extremely low diffusivity of elements in the sub-solidus mantle. While most elements in mantle rocks have chemical diffusion coefficients of $10^{-10} - 10^{-20}\,\mathrm{m^2s^{-1}}$, hydrogen has a diffusion coefficient of about $10^{-7} - 10^{-8}\,\mathrm{m^2s^{-1}}$ (see the compilation of laboratory data from various studies in Philpotts and Ague (2009), p. 125). The different mantle lithologies (as well as their individual minerals) are thus likely to have equilibrated in water content during the more than hundreds of million years prior to entering a mid-ocean ridge melting zone (cf. equation (3.1)). However, during the sub-ridge upwelling and melting process, which takes place on a much shorter time-scale, water diffusivity is too slow to re-equilibrate between lithologies. While keeping track of the water content of each lithology, interesting (though feasible) scenarios can evolve, during which wet melting of a peridotite could coexist next to dry melting of pyroxenite.

The water content of each lithology decreases as soon as melting starts, because water is preferentially partitioned into the melt similar to an incompatible element (partition coefficient $D_{H_2O} = 0.01$, (Hirth and Kohlstedt, 1996)). Fractional melting is assumed to account for the rapid transport of melt from the melting region to the surface that is indicated by geochemical data (Stracke et al., 2006; Rubin et al., 2005). In other words, melts are assumed to not equilibrate chemically with surrounding rocks but to rapidly leave the mantle. Fractional melting is approximated numerically by neglecting any existing melt in the melting model. This simplification does not allow to account for some processes that are potentially of importance in reality: For instance, melts ascending from an enriched vein could react corrosively when in contact with a peridotite at shallower depth and generate further peridotite melt (Phipps Morgan, 2001). However, modeling this scenario would require (a) modeling of melt migration, (b) estimating the contact area between wall rock and melt, (c) estimating the thickness of the wall rock that

[2] This is in analogy to thermal equilibria: Materials with different heat capacities that equilibrate in temperature store different amounts of heat.

equilibrates with the melt chemistry, and (d) a thermodynamic code that then calculates the new composition of both melt and wall rock. This dramatically increasing complexity is beyond the scope of this study.

Summary

A 1-D model has been constructed that solves for the thermal evolution (3.2) as well as advection (3.10)-(3.11) and melting (3.3)-(3.4) of a multi-component mantle. The vertical upwelling speed (i.e. decompression rate) is prescribed. The effect of water on each mantle lithology's solidus function is included by modifying the solidus-depletion dependence (3.9). Each mantle lithology is composed of four minerals, which can affect the melting behavior (e.g. *cpx*-out effect) as well as the physical properties of the mantle (e.g. density and viscosity; only important for the 2-D and 3-D models). During the melting process, the composition and water content of each lithology are tracked. Melt is instantaneously removed in order to approximate fractional melting.

3.2.3 Initialization and boundary conditions

The initial mantle composition for each numerical experiment is calculated using the above framework. The following steps initialize the 1D experiments:

1. define the number of mantle components: either 1 (FP), 2 (DP+PYX) or 3 (DP+PYX+FP)

2. define the initial depletion F_{DP_0} of the DP (FP and PYX start with $F_{FP_0} = F_{PYX_0} = 0$)

3. define the volume fraction of the FP in the assemblage (V_{FP}); to achieve a pyrolite composition of the mantle, the volume ratio between DP and PYX is defined by F_{DP_0}: $V_{DP} = 1 - V_{FP} - V_{PYX}$, where $V_{PYX} = F_{DP_0}$

4. define the bulk water content of the mantle, e.g. $X_0 = 200\,\text{ppm}$

5. calculate the DP and PYX mineral composition by "pseudo-melting" the FP up to the degree F_{DP_0}

6. calculate the water content in each mineral using the partition coefficients in Tab. 3.3

7. calculate the bulk water content in each mantle component X_{c_0} (with c=DP, FP, or PYX) based on its mineral fractions and the water content of the minerals

8. calculate the solidus function for each mantle component based on F_{c_0} and X_{c_0}, initialize sub-solidus temperatures along the 1-D profile, and start the experiment

During the experiments we keep track of the degree of melting F_c, the mineral composition C_c and the water content X_c of each lithology, so that their solidus functions can be updated at every time step. The 1D model represents a vertical profile that extends from below the onset-depth of melting up to the ridge axis. The boundary conditions are 0 °C at the top and mantle temperature $T_M = 1315°$C at the lower boundary. The adiabatic increase in temperature with depth is not included, so that all temperatures are potential mantle temperatures. A vertical upwelling rate of 20 km/Myr, representative for a slow-spreading ridge, is used in the 1-D experiments.

All runs start with the initial volume faction and depletion of all mantle lithologies along the entire profile. The initial mantle temperature profile is chosen to be slightly below the lowest solidus temperature at any depth to avoid perturbations from an otherwise excessive melt production during the first time step. The mantle is advected upwards at a prescribed speed, eventually crosses the solidus function(s), starts to melt and changes its composition. The calculations continue until a steady state in all variables is reached, which happened in all experiments shortly after advecting over a distance corresponding to the length of the 1D profile.

3.2.4 1-D results

Before discussing the model calculations that include all of the above mechanisms, it may be instructive to look at some temperature profiles first. Fig. 3.2 shows the steady state temperatures paths in the 80 km below the ridge axis for five 1-D model calculations. If no melting is considered, the hot mantle with a potential temperature of 1315°C rises to very shallow depth until the conductive heat loss at the top rapidly cools it within the uppermost 10 km (model 1, dashed line). If melting of a homogeneous, dry mantle composition is considered, melting starts at about 54 km depth, where the mantle temperature intersects the dry solidus. The thermal energy required for melting is taken from the "overheated" mantle rocks and cools them (the so-called latent heat effect). Consequently the mantle temperature must stay at the solidus, because all additional temperature is used to produce more melt. The mantle temperature path follows the solidus until melting stops at about 8 km depth (model 2, dash-dotted line), where conductive cooling from the top leads to sub-solidus temperatures.

Mantle rocks contain a variety of chemical elements that differ in their compatibility with respect to the mineral structure of the rock. When melting starts, very incompatible elements go into the melt immediately, while more compatible elements prefer to

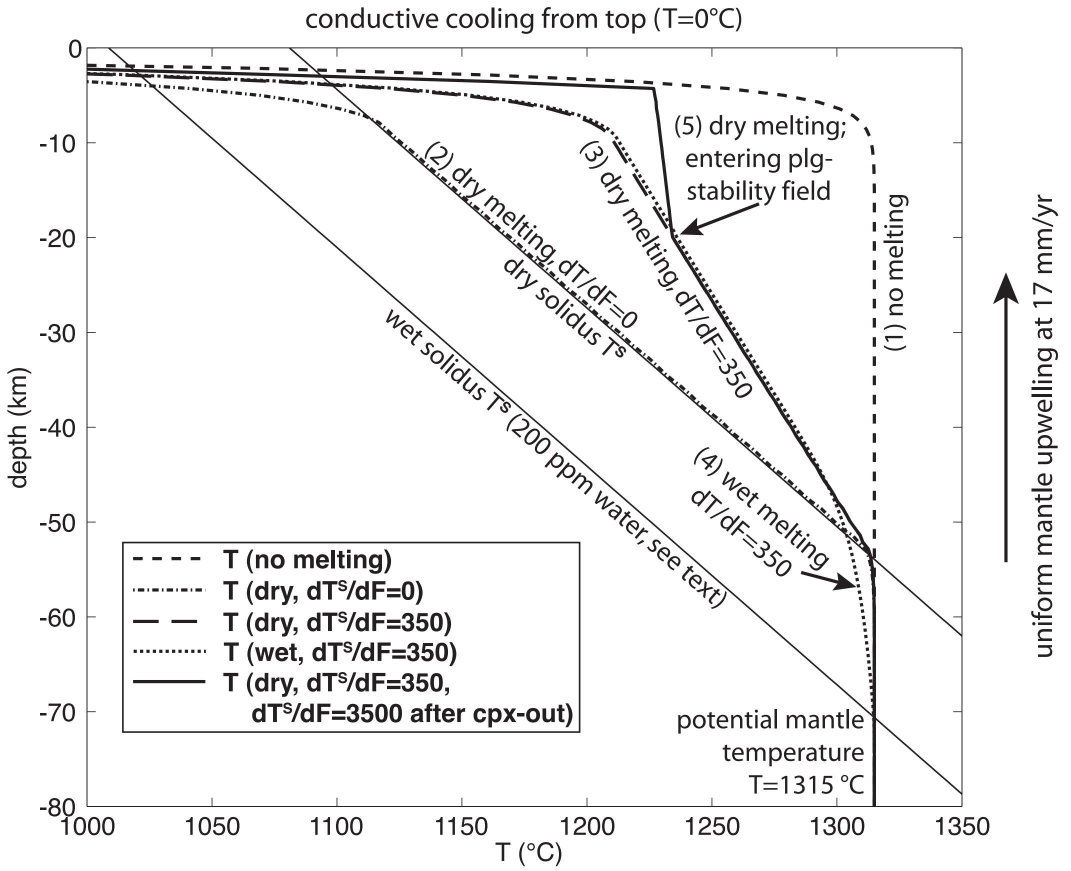


Figure 3.2: *1D temperature profiles below a mid-ocean ridge for different models. 1) no melting, only advective transport in the upward direction and conductive cooling from the top; 2) dry melting with no solidus-depletion dependence ($dT^s/dF = 0$); 3) dry melting with a solidus-depletion dependence of $dT^s/dF = 350$ (peridotite value); 4) wet melting of a mantle with 200 ppm water; 5) dry melting with a factor of 10 increase in dT^s/dF once the mantle enters the plg-stability field. The reduced melt productivity above this point results in less latent heat cooling and upwelling of hotter mantle to shallower depth. Clinopyroxene exhaustion (cpx-out) at greater depth would have the same effect in our calculation.*

stay in the solid. Consequently the rock changes its composition continuously as it melts and, with increasing degree of melting, it becomes more difficult to melt. This depletion-dependence of the solidus (dT^s/dF) is included in model 3 (Fig. 3.2, long-dashed path). Instead of following the dry solidus of the initial composition, the temperature at any depth lies at the solidus of the evolved mantle composition at that depth. Because composition changes with degree of melting, the temperature path diverges from the original solidus and is continuously shifted towards higher temperatures as soon as the melt extraction starts.

Water in the mantle behaves like an incompatible element and its presence increases the potential for upwelling mantle to melt. Adding 200 ppm water to the mantle lowers the solidus temperature by about 80°C . During melt extraction, water partitions into the melt, so that the residue quickly dries out and its solidus shifts towards the dry solidus

(Fig. 3.2, model 4, dotted path). Once the wet mantle has dried out and reaches the same depletion as the initially dry mantle, the temperatures of both calculations follow the same path (long-dashed and dotted lines above 48 km depth). Note, however, that the temperatures of the dry and wet calculation merge a few km above the initial dry solidus, because of the small increase in depletion during the wet melting below the dry solidus. Model 5 (Fig. 3.2, solid line) is similar to model 4 but includes an increase in dT^s/dF by a factor of 10, once the component enters the *plg*-stability field at 21 km depth. Melting rates drop here and less latent heat is consumed, so that the mantle stays hotter during the final ascent. A similar effect occurs, if a component runs out of *cpx* before reaching the *sp*-*plg* phase transition.

The calculations presented next are similar to model 5, in that all of the effects discussed above are included. Fig. 3.3 shows several variables at steady state during the decompression melting of a homogeneous, fertile peridotite (FP) under dry (top row) and wet conditions including 200 ppm water (bottom row). As the dry mantle crosses its solidus temperature at about 54 km depth (panel A) the melt productivity dF/dz reaches a steady 0.55 $\frac{\%}{\text{km}}$ (panel B). Depletion (panel C) increases linearly until the *sp*-*plg* phase transition is reached, where the solidus depletion dependence suddenly increases. At this point melt productivity decreases and only a small amount of melt is produced up to the depth where conductive cooling from the top intersects. The mineral composition of the peridotite (panel D) changes during melting, because only *cpx* and *sp* are assumed to melt so that *ol* and *opx* become larger fractions in the residue (all mineral fractions are rescaled to 100%). In the dry experiment no melt is produced in the garnet stability field (below the gray solid line in Fig. 3.3). Panels (E) and (F) show density and viscosity calculated using Eq. (3.17) and (3.18) in Section 3.3, resp. The most obvious features in the density profile are the phase transitions and the density increase at the top, caused by conductive cooling. The viscosity profile shows the pressure dependence (decrease towards shallower depth), on which the weakening effect of melt and the temperature dependence are superimposed. These effects are discussed later in the in the section on 2-D models and are shown here for comparison to 2-D and 3-D models.

Panels G-L in Fig. 3.3 show the same calculation as above but under wet conditions. The initial water concentration in the fertile peridotite is defined to be 200 ppm. In contrast to the dry case, melting starts deeper at around 70 km and within the *gt*-stability field. The *gt*-signature in the melt, however, is likely to be very small because the melt production at this depth is low. Productivities as large as in the dry case are observed not until the dry solidus is crossed. The total melt production is very similar to the dry case, because the small additional melt production at great depth comes at the cost of latent heat cooling. Thus, subsequent dry melting occurs at a slightly lower rate than in the initially dry case, so that both scenarios lead to very similar total productivity. The

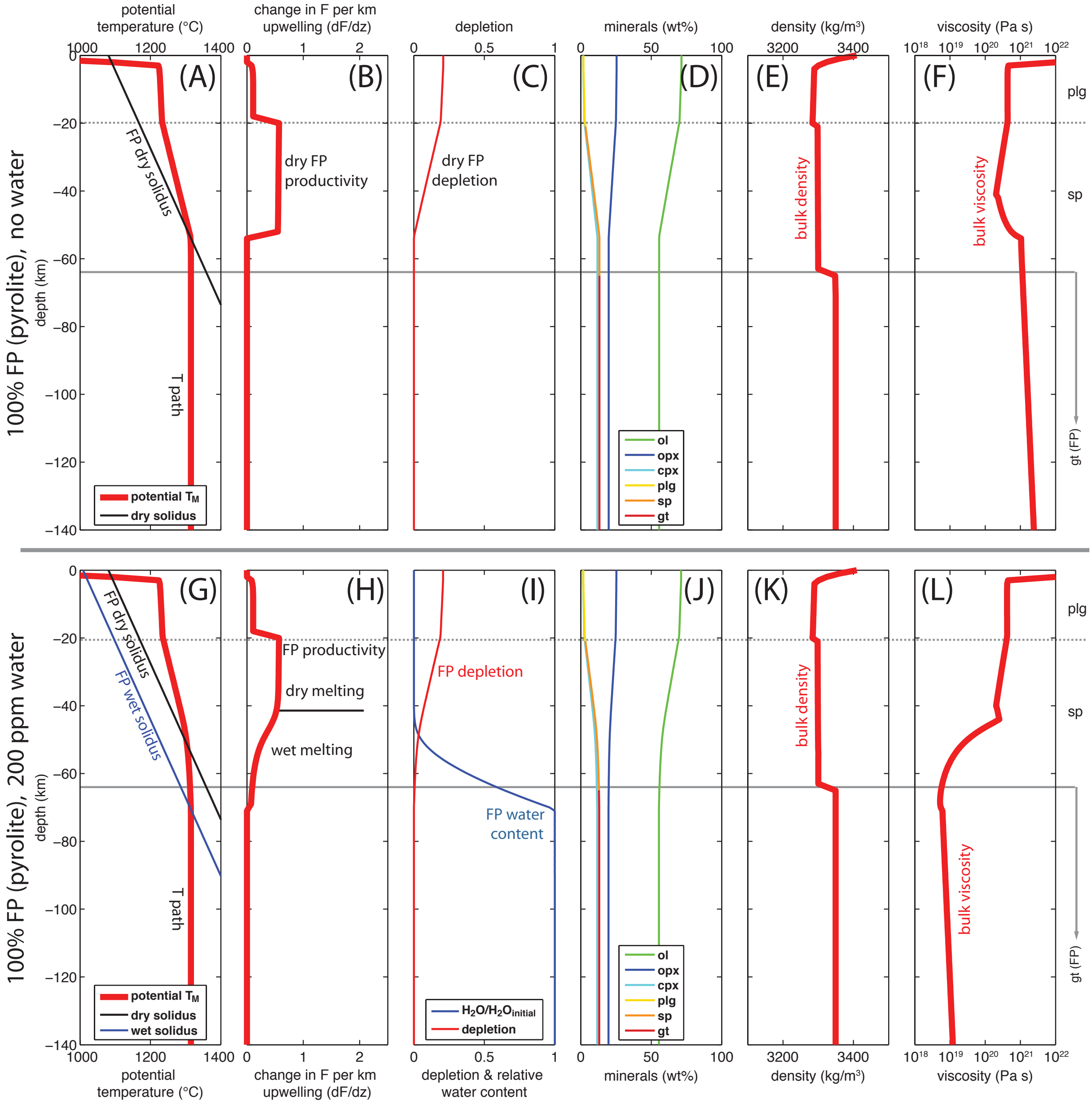


Figure 3.3: *1-D decompression melting of a homogeneous fertile peridotite (FP) under dry (A-F) and wet (G-L) conditions. Top row: A) T-path (red) and dry FP solidus. B) melt productivity in dF/dz. C) depletion. D) changing mineral content in FP as cpx and sp are removed by melt extraction. E) density of the mantle using including T, composition, depletion, and melt fraction (discussed in Section 3.3). F) viscosity including T, P, and melt fraction (discussed in Section 3.3). Bottom row: Same calculation but with under wet conditions. G) T-path, dry and wet FP solidus. H) melt productivity in wet and dry melting regimes. I) depletion and relative change in water content (blue). J) mineral composition. K) density as in E). L) viscosity as in F but including the dehydration-related increase. Mantle density and viscosity have no influence in this 1-D model. Gray horizontal lines (solid and dashed) mark the depths of the gt-sp and sp-plg phase transitions defined for FP, resp.*

signatures of the melts in both experiments, however, could vary, because different depths are sampled during the melting process.

While the density profile is very similar to the dry condition run, the viscosity profile shows a much lower viscosity at greater depth. Water has a strong weakening effect on olivine (Karato and Wu, 1993; Hirth and Kohlstedt, 1996) and, as the water partitions into the melt, the extraction of partial melt results in a much more viscous residue (the so-called dehydration-related increase in viscosity). This effect dominates over the melt weakening and leads to the formation of a ∼50 km thick compositional lithosphere by mid-ocean ridge melting.

Fig. 3.4 shows the melting of a two-component system composed of a depleted peridotite (DP, 10% depletion with respect to FP and a volume fraction of 0.9) and an enriched pyroxenite (PYX, volume fraction of 0.1). The bulk composition of this mantle is identical to the single-component calculation in Fig. 3.3. Shown are the dry case (Fig. 3.4, upper row) as well as the wet case with 200 ppm bulk water content (lower row). Because their mineral composition differs, the pyroxenite stores 366 ppm and the peridotite 181 ppm of the water (see discussion above).

PYX starts to melt at greater depth than DP at about 68 km. Since it is the only melting component at this depth but only makes up 10% of the mantle rock, its productivity is enhanced by the heat that flows from the non-melting DP into the PYX veins (panel B, black dash-dotted line). The thermal energy stored in DP is available for melting PYX, because all lithologies are assumed to be in thermal equilibrium. The bulk productivity of the mantle (panel B, red line), however, is relatively low, because only 10% of the rock mixture is producing melt. As the volume fraction of PYX decreases towards 40 km depth, PYX productivity increases but the bulk productivity decreases. At 40 km depth DP crosses its solidus and suddenly the productivity of PYX drops, because heat flow from DP into PYX stops. From now on, the temperature is mainly controlled by the heat consumed by DP melting. Note that the onset of DP melting is delayed compared to Fig. 3.3, because the peridotite is more depleted initially and PYX melting has consumed latent heat, thus cooled the mantle. Once the entire rock melts, the bulk productivity increases to the highest value in this experiment. Both lithologies suddenly decrease in productivity when they enter the *plg*-stability field at 21 km depth. The higher melting rate of PYX compared to DP when both melt simultaneously is explained by the lower solidus-depletion dependence assumed for PYX (see Tab. 3.2).

The wet melting example shows all the above effects, too, but the onset of melting of each component is ”smoothed-out” by the low-degree melting between wet and dry solidus (panel G and H). Slightly above the depth where PYX crosses its dry solidus (around 60 km), wet melting of the DP sets in and consumes a small fraction of the thermal energy (see the small kink in PYX productivity once wet DP melting starts). Towards shallower depth, this fraction becomes larger as DP dries out and increases in productivity. PYX productivity drops gradually, while the bulk productivity increases. Before entering the *plg*-stability field both lithologies have lost their water to the melts

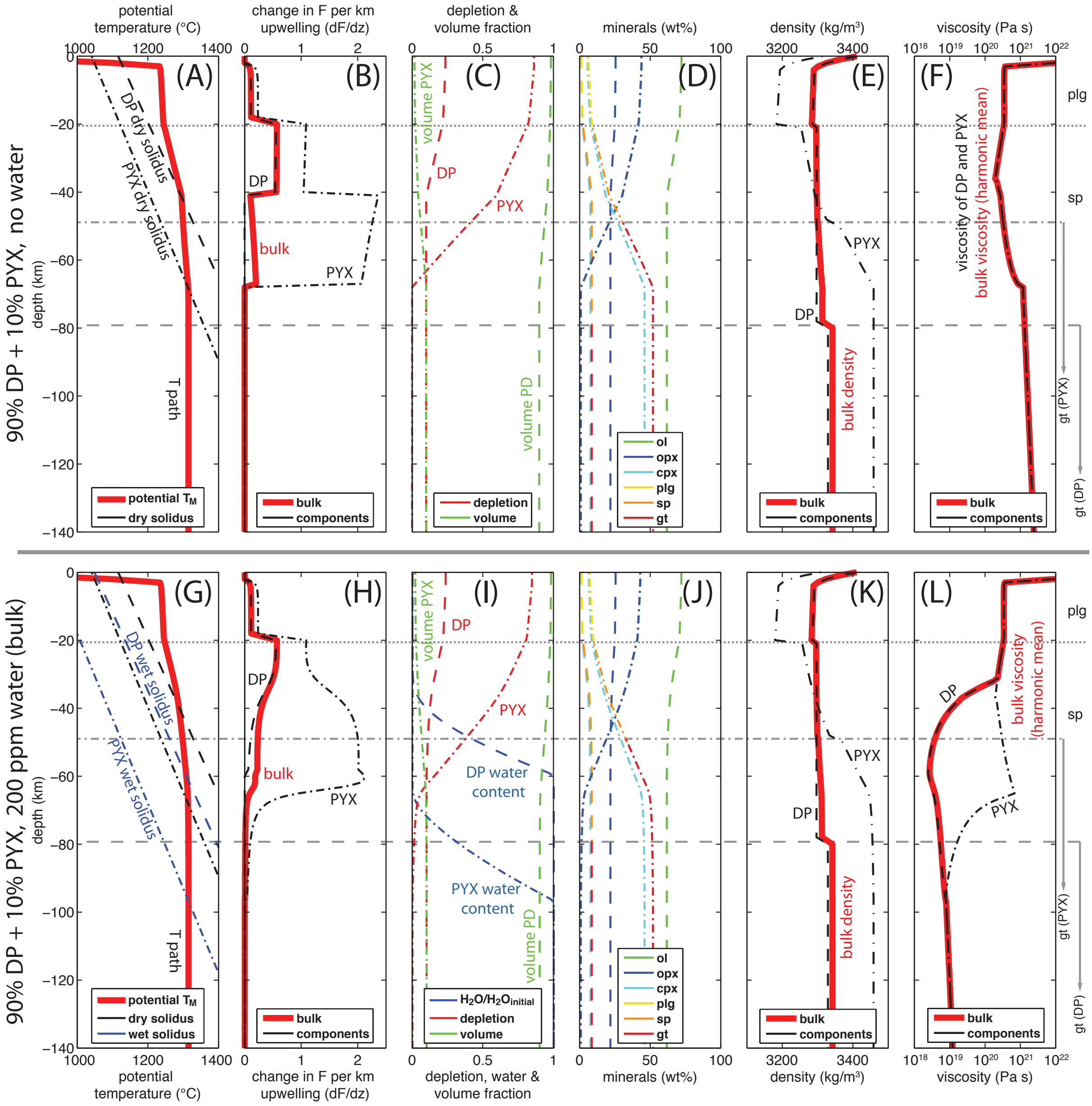


Figure 3.4: *1D decompression melting of a two-lithology mantle composed of 90% depleted peridotite and 10% enriched pyroxenite. The bulk composition of this mantle is identical to the fertile peridotite shown in Fig. 3.3. Panels (A-F) show a calculation under dry conditions, (G-L) show melting in the presence of 200 ppm water. Gray dotted horizontal lines mark the depth of the sp-plg phase transitions; gray dashed and dash-dotted lines mark depths of gt-sp phase change defined for DP and PYX, resp. See Fig. 3.3 and text for further explanation.*

and show the same productivities as in the dry case (panel B).

Both experiments show the stronger density reduction of PYX compared to DP towards shallower depth. The difference results from melting and extracting the dense mineral *gt*, which is more abundant in PYX. Above the *gt*-*sp* phase transition in PYX at 49 km depth (Tab. 3.3), the density reduction becomes less pronounced. The effect on the mantle bulk density is comparably small, because PYX represents a small fraction of the rock that

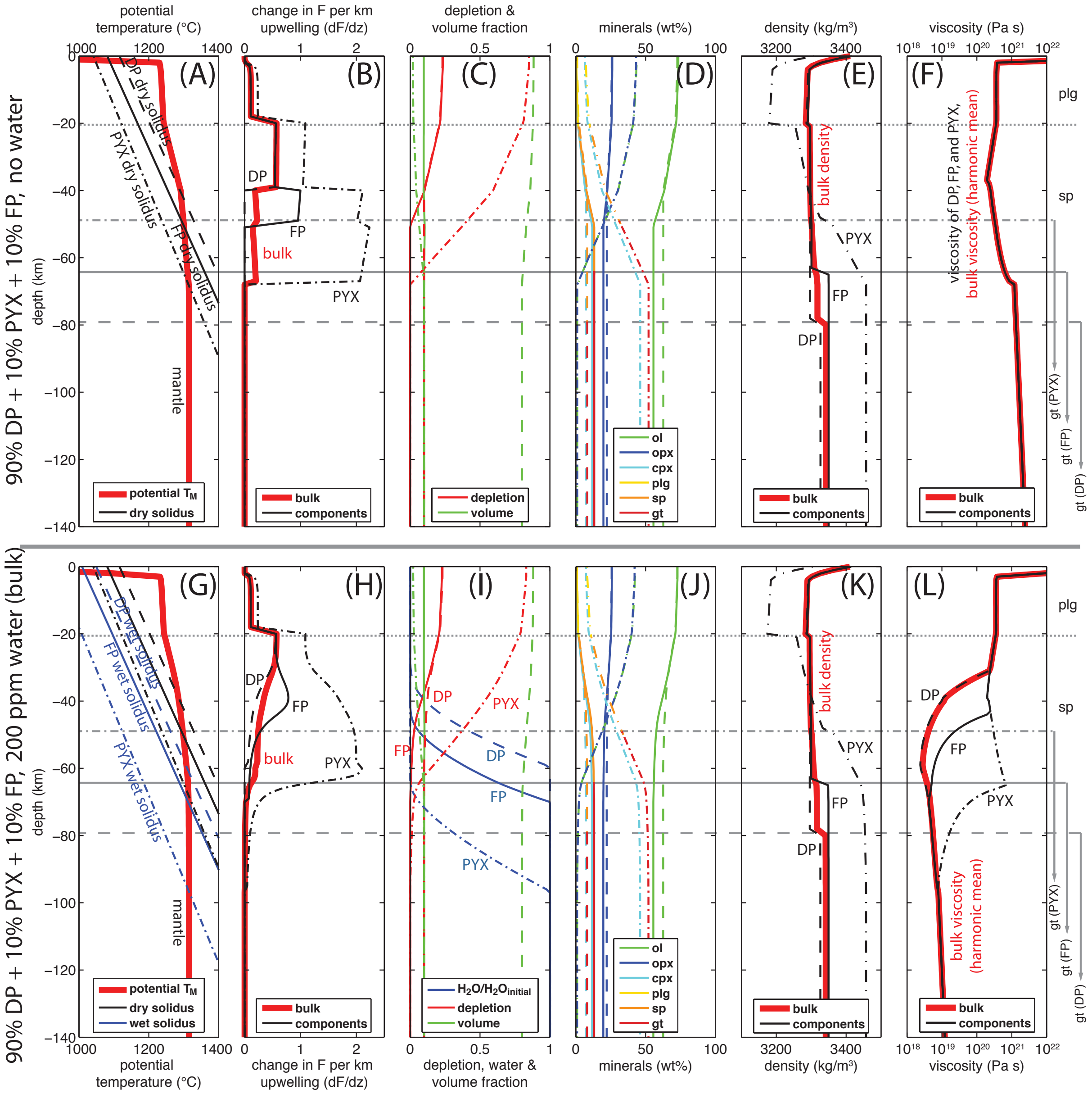


Figure 3.5: *1D decompression melting of a three-component mantle composed of 80% depleted peridotite, 10% fertile peridotite, and 10% enriched pyroxenite. The mantle bulk composition is identical to the calculations in Fig. 3.3–3.4. Panels (A-F) show a calculation under dry conditions, (G-L) show melting in the presence of 200 ppm water. Gray lines (dotted, dashed, dash-dotted) as in Fig. 3.4; solid gray line as in Fig. 3.3. See Fig. 3.3 and text for further explanation.*

becomes even smaller during PYX-melting. As opposed to the wet melting experiment with a single component, the low viscosity region extends to shallower depth (compare Fig. 3.3L and Fig. 3.4L). This is, because the aggregate viscosity of the mantle mixture remains low as long as DP contains most of its water and stays weak. DP dehydrates at about 40 km depth, which marks the dehydration-related increase in viscosity of the mantle. The dehydration and stiffening of PYX has almost no effect, because the effective rheology of a mixture of strong and weak components is mainly controlled by the low viscosity material (a volume-weighted harmonic mean of the lithology viscosities is used; see (3.18e) on page 129).

Decompression melting of a three-lithology system is shown in Fig. 3.5. It includes 80% DP (10% depletion), 10% FP, and 10% PYX. Again the bulk mantle composition is the same as in the above experiments. Panels A-F show dry melting and G-L wet melting (200 ppm bulk water content in the mantle). DP, FP, and PYX store 179, 200, and 366 ppm water, resp., due to their different mineral composition.

PYX melts first at a high rate but low bulk productivity (A). Heat flow from both DP and FP into PYX veins enhances its melting. Next FP crosses its solidus and PYX productivity slightly drops. At the point where FP is depleted to the same degree as DP, both lithologies have become the same material (see the mineral compositions of each component in panel D). Thus, they have the same solidus function and continue to melt at the same rate as a single lithology – the experiment appears to be identical to the 2-component calculation at shallow depth. However, the thermal energy consumed by melting FP (so that it becomes DP in the first place) is not available for the PYX melting. As a result, the maximum depletion of PYX in the 3-component calculation is about 2% lower than in the 2-component case.

The wet melting scenario appears again as a ”smoothed” version of the dry calculation but does reveal some particular characteristics. For instance, when DP first starts to melt it is not the same material as the progressively melted FP. DP is in the wet melting regime, while FP has already lost most of its water. Only after DP has entered the dry melting regime (above 30 km depth) both DP and FP have the same productivity. The compositional lithosphere is about 30–40 km thick, similar to its thickness in the 2-component experiment.

3.2.5 Discussion of the 1-D results

The results of the above 1-D decompression experiments with 1, 2, and 3 lithological units (Fig. 3.3-3.5) can be summarized as follows

- The melting rates of a lithology (i.e. its melt productivity) is highest once the component has dried out, i.e. wet melting decreases pressure-release productivity.
- The addition of water shifts the onset of melting to greater depth, but only low degrees of melting are observed between wet and dry solidus.
- The maximum degree of melting as well as the total amount of melt produced are very similar in calculations that only differ in the mantle’s water content (see Fig. 3.6) – this is contrary to the conclusions in Bonatti (1990)

- Enriched components like PYX begin to melt at greater depths than the more depleted matrix in which they are embedded. They show enhanced melting rates as long as the matrix is not also melting due to heat flowing from the non-melting matrix into the melting lithologies.
- The bulk productivity of the rock remains low until the entire rock melts (Phipps Morgan, 2001)
- The aggregate rheology of the mantle is controlled by the weakest lithology in the mantle mixture. The base of the compositional lithosphere is defined by the onset of melting and dehydration of the most depleted component.

The experiments show that the presence of water causes an onset of melting at greater depth, which results from the shift of the solidus functions towards lower temperatures (i.e. melting at a given mantle temperature starts at greater depth). The addition of 200 ppm water, for instance, lowers the solidus by about 80°C. Depending on the slope of the solidus, this corresponds to an almost 20 km deeper base of the melting zone than if the mantle would contain no water.

The results also show that the melt production within these additional 20 km is fairly low and does not increase the total melt production (see also Fig. 3.6). The low melt production at the onset of wet melting is a result of the strong partitioning of water into the melt. The large amount of water leaving the mantle rock rapidly shifts its solidus towards higher temperatures and chokes the melt production. Another reason for the similar total melt production in dry and wet melting experiments is the law of energy conservation. Melting consumes energy and cools the mantle rocks, so that an earlier onset of melting cools the mantle during its ascent and reduces melting towards shallower depth, i.e. in the dry melting regime.

Recent studies on melt extraction at mid-ocean ridges favor fast melt percolation and transport times as low as a few decades (Rubin et al., 2005; Stracke et al., 2006). This does not allow for a chemical equilibration of deep, water-rich melts with wall rocks at shallower depth, which is why fractional melting is assumed in the above calculations. As a result of this formulation, water that partitioned into the melt will leave the mantle with the melt – in the presented model as soon as melt is formed, because melt is removed instantaneously. However, the very first melts produced are likely to be immobile, because melt migration depends on permeability, which in turn requires a certain amount of melt along grain boundaries to form interconnected channels (e.g. Kelemen et al., 1997). These melts might stay in contact with wall rock long enough, so that diffusion of water into adjacent rocks could trigger additional melt production. This could lead to a slightly larger melt production at the base of the melting column than predicted in the above

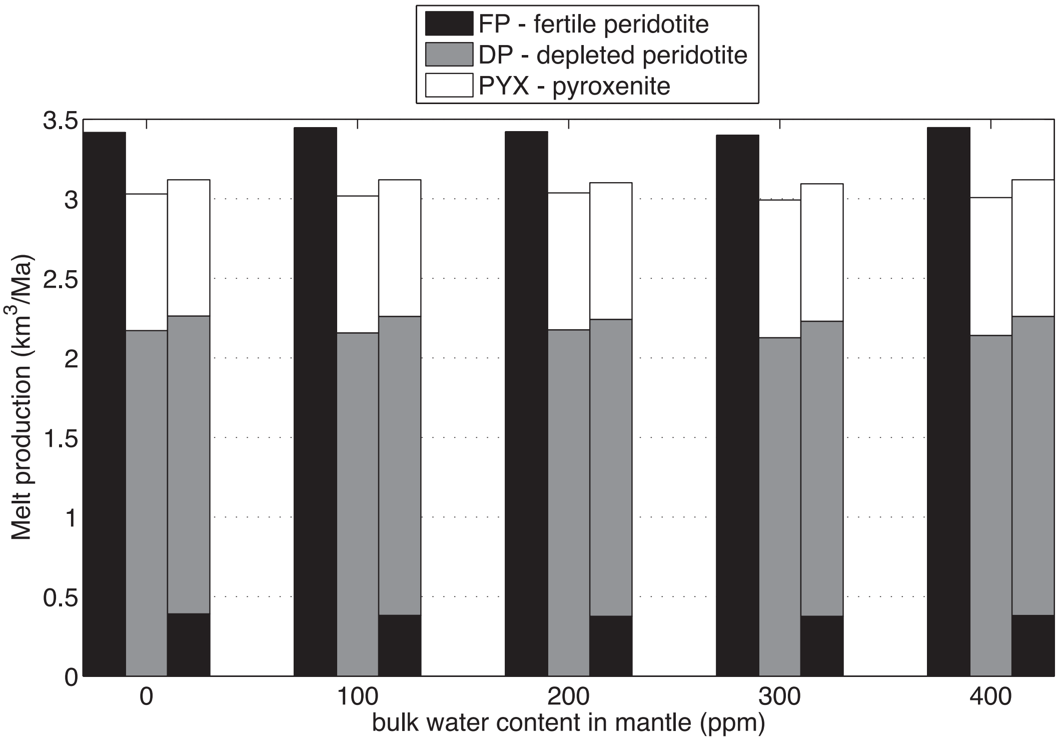


Figure 3.6: *Integrated melt production for experiments with 1, 2, and 3 lithologies and different bulk water contents. The total melt production does not depend on the initial bulk water content in the mantle. A systematically higher melt production is observed for the single-component experiments (100% FP, first bar in each group). This is explained by the solidus function selected for PYX, which has a slightly steeper slope (Fig. 3.1).*

models, but it is unlikely that this would fundamentally change the conclusions listed above.

The experiments with a single lithology (FP) show a higher total productivity than the ones with two (DP+PYX) or three (DP+FP+PYX) components (see Fig. 3.6). This occurs because the slope of the solidus function chosen for PYX differs from the slope of the DP solidus function (see Fig. 3.1 and parameters in Tab. 3.2). The solidus functions are taken from different publications and are not coupled to each other in a thermodynamic sense. Numerical experiments using the above 1-D model, in which several peridotites with different depletion define the initial mantle rock mixture, lead to the same total melt production. This is also shown in Phipps Morgan (2001), for a peridotite composed of layers with different initial depletion. While the total (integrated) melt production is the same in this case, the melt productivity as a function of depth is different, because it depends on the fraction of mantle rock that actively melts (cf. PYX productivity vs. bulk productivity in Fig. 3.4, panel B).

The 1-D models are idealized vertical mantle upwelling scenarios without any feedback between mantle flow and melt production. Inherent in these models is the assumption of axial symmetry (symmetry in all horizontal directions). Even an idealized straight mid-ocean ridge far away from segment edges can only assumed to be symmetric with respect to the vertical plane underneath the ridge axis. Here, a 2-D model is required to account for the heat flow in and out of the melting region due to lateral temperature gradients and laterally varying mantle upwelling rates. If no symmetry at all exists, for instance if the ridge axis is unevenly shaped, spreading is oblique, or transform faults are near, 3-D models are required. Although geometrically simplified, the above 1-D models provide insight into the basic relations among the various parameters and allow to quickly scan the parameter space. Furthermore, the steady state solutions for volume fraction and depletion of each lithology for a given mantle composition serve as a good initialization of the compositional fields in 2-D and 3-D models.

3.3 2-D and 3-D models for mantle flow and melting at ridges

The 1-D results in the previous section have identified some differences between melting of a multi-component and a homogenous mantle, even if both have the same bulk composition. The dissimilarities are primarily depth-dependent, in that the onset of melting of depleted components is delayed, whereas more fertile or enriched components start to melt at greater depth. When considering the water content of each lithology, this is likely to have consequences for the mantle rock rheology and, thus, on mantle flow. Other factors that affect the mantle rock rheology are temperature and melt fraction within the rock. All these quantities are related to the melting process and to the compositional structure of the mantle.

The numerical models $\mathbf{M3_{tri}}$ and $\mathbf{M3_{tet}}$ that have been developed in this thesis (Chapter 2) are used to study mantle flow and melting in 2-D and 3-D, resp. These models combine three major computational units: (1) the viscous flow solver, which is described in detail in Section 2.6.3, (2) the thermal advection/diffusion solver, which is discussed in Section 2.2, and (3) the melting model that has been introduced in connection with the 1-D model in the previous section.

In the following sections I will present sample applications, in which the above models are used. First, I will present selected results of 2-D and 3-D experiments on mantle flow and melting at mid-ocean ridges. The 3-D experiments include the effects of long-offset transform faults. An additional example for an application of the 3-D model is presented at the end of this chapter, where a melting anomaly at the Mid-Atlantic ridge near Ascension Island is in the focus of a case study. The results are meant to illustrate the potential of the 2-D and 3-D mantle convection codes developed in this thesis, rather than completed studies on mid-ocean ridge processes and transform faults.

3.3.1 Model description

The 2-D and 3-D mantle convection codes $\mathbf{M3_{tri}}$ and $\mathbf{M3_{tet}}$ developed in this thesis are used to solve for viscous flow, thermal evolution and melting of the Earth's mantle in Cartesian coordinates. The mathematical foundation and numerical formulation of the different parts of the code are discussed in sections 2.6.3 (Stokes flow solver), 2.2.3 (solution for thermal advection-diffusion) and 3.2.2 (multi-component melting). The codes are completely written in MATLAB (*www.mathworks.com*) and are parallelized to run on distributed memory systems.

Viscous flow

The mantle is described as an incompressible, viscous fluid with infinite Prandtl number (Stokes flow). At every time step, a steady-state solution for velocity and pressure is calculated that depends on the given density and viscosity fields, and the imposed boundary conditions. The Boussinesq approximation is applied, that is, density differences only take action in the buoyancy force term and in no other term. The governing equations can be written as

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{3.14}$$

$$\frac{\partial p}{\partial x_i} = \frac{\partial \tau_{ij}}{\partial x_j} - \rho g \, e_z \tag{3.15}$$

$$\tau_{ij} = \eta \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{3.16}$$

Eq. (3.14) satisfies conservation of mass by imposing incompressibility, Eq. (3.15) describes the force balance to ensure conservation of momentum, and Eq. (3.16) represents the constitutive law. τ_{ij} denotes the deviatoric stress tensor, u velocity, x physical coordinates, p pressure, g gravitational acceleration, ρ density, η viscosity and e_z the unit vector in the vertical direction. A complete list of variables, their meaning, units, and values is given in table Tab. 3.1.

Density ρ in Eq. (3.15) changes with temperature T, degree of melting (=depletion) F, mineral composition C, as well as the melt content ϕ of the mantle rock. These controlling factors are combined in the total density $\rho(T, F, C, \phi)$ using the following formulations.

$$\rho_C = \xi_{ol} \, \rho_{ol} + \xi_{opx} \, \rho_{opx} + \xi_{cpx} \, \rho_{cpx} + \xi_{gt} \, \rho_{gt} \tag{3.17a}$$

$$\rho_M = \sum_i^{nC} V_{ci} \, \rho_{ci} \tag{3.17b}$$

$$\rho(T, F, C) = (1 - \alpha(T - T_0) - \beta F) \, \rho_M \tag{3.17c}$$

$$\rho(T, F, C, \phi) = \phi \rho_m + (1 - \phi) \, \rho(T, F, C) \tag{3.17d}$$

Eq. (3.17a) describes the compositional density of each lithology based on its mineral content ξ and the mineral densities (see Tab. 3.3 on p. 112). The bulk density of the mantle is the density of each lithology, weighted by the lithology's volume fraction in the mantle →(3.17b) (nC is the number of lithologies considered). The bulk density is modified according to ambient temperature (α is the thermal expansion coefficient and T_0 is the reference temperature) and depletion F →(3.17c). The latter modification accounts for the preferential partitioning of iron (Fe) into the melt phase, which leaves the olivine in the residue relatively enriched in less dense magnesium (Mg) (Oxburgh and Parmentier,

1977) and (Phipps Morgan, 1997). β parameterizes the density decrease of the residue as it undergoes melting. This irreversible process leads to a compositional buoyancy equivalent to few hundreds of degrees thermal buoyancy (Yamamoto and Phipps Morgan, 2009). In the presence of melt, the mantle rock's bulk density is further reduced, because basaltic melt is buoyant at upper mantle pressures. The melt-induced buoyancy, accounted for in (3.17d), scales with the melt fraction ϕ and depends on the density of melt ρ_m.

Viscosity η in Eq. (3.16) is formulated as a function of temperature, depth, the water content of olivine, and the melt content of the mantle rock.

$$\eta_c = \eta_0 \cdot A_X \cdot B \cdot \exp\left(\frac{E_A + pV_A}{RT_{(K)}}\right) \tag{3.18a}$$

$$A_X = \frac{X0_{ol}}{max(X_{ol}, \frac{X0_{ol}}{\delta\eta_x})} \tag{3.18b}$$

$$B_{HK} = \exp\left(-45\phi\right) \tag{3.18c}$$

$$B_{TH} = \begin{cases} \exp(-400\phi) & , \text{if } \phi <= \phi_B \\ \exp(-400\phi_B) * \exp(-20(\phi - \phi_B) & , \text{if } \phi > \phi_B \end{cases} \tag{3.18d}$$

$$\text{where } \phi_B = 5 \cdot 10^{-3}$$

$$\eta = \left(\sum_i^{nC} \frac{V_i}{\eta_{ci}}\right)^{-1} \tag{3.18e}$$

Eq. (3.18a) is an Arrhenius-type law to describe a temperature and depth-dependent rheology: E_A denotes activation energy, V_A the activation volume, R the universal gas constant, $T_{(K)}$ mantle temperature in units of Kelvin, p pressure, and η_0 the reference viscosity. The pre-factors A_X and B parameterize the viscosity increase during the dehydration of olivine and the weakening effect of melt at grain boundaries, respectively. Factor A_X (3.18b) defines a viscosity increase by a factor of $\delta\eta_x$ as the water concentration in olivine (X_{ol}) decreases with respect to the initial water concentration $X0_{ol}$. This formulation is based on Hirth and Kohlstedt (1996), who report an increase in olivine viscosity that is inversely proportional to its decreasing water content.

Factor B parameterizes the viscosity reduction due to melt present along grain boundaries. Here, two different formulations have been tested: B_{HK}, defined by (3.18c), is the parameterization given by Hirth and Kohlstedt (2003) that results in a moderate decrease in viscosity as the rock's melt fraction increases. Recently, Takei and Holtzman (2009b) have estimated that the weakening effects of melt at grain boundaries might be much stronger, and that the weakening occurs almost instantaneously when melt is present (at melt fractions as low as $\phi = 10^{-4}$). To test the implications of the sudden viscosity drop, Fig. 11 in Takei and Holtzman (2009a) (a plot of normalized shear viscosity against melt fraction) is parameterized using the two exponential functions in (3.18d) to form B_{TH}. A

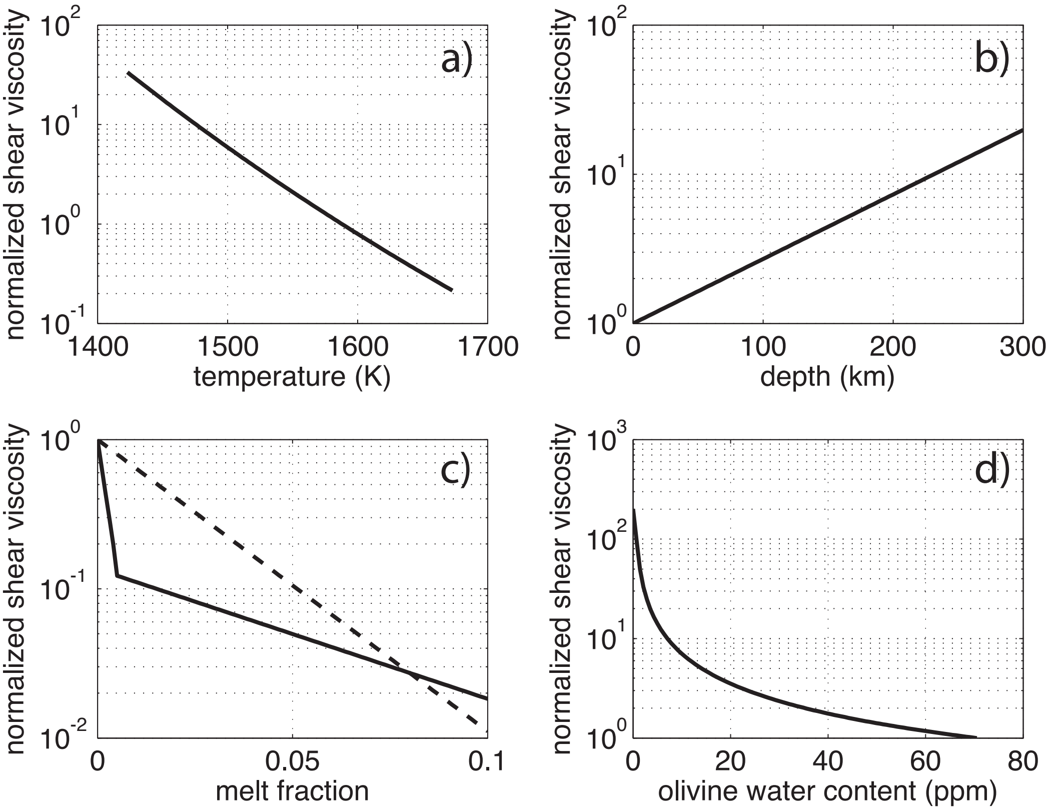


Figure 3.7: *Different factors controlling the viscosity of mantle rocks: (a) temperature-dependence ($E_A = 400kJmol^{-1}$); (b) depth-dependence ($V_A = 4 \cdot 10^{-4}cm^3mol^{-1}$); (c) melt weakening effect based on Hirth and Kohlstedt (2003) (dashed) and Takei and Holtzman (2009a) (solid); (d) water content of olivine (Hirth and Kohlstedt, 1996).*

threshold melt fraction ϕ_B is used to defined the transition in the viscosity-dependence that was shown in Takei and Holtzman (2009a). The formulation in (3.18c) is used in all calculations, if not mentioned otherwise. The parameters controlling viscosity are summarized in Fig. 3.7.

The viscosity of each lithology is calculated independently using (3.18a). All other controlling factors, namely T, p, and ϕ, are assumed to be the same for adjacent lithologies. To derive the bulk viscosity of the mantle rock, the viscosities of the lithologies are averaged using a volume fraction-weighted harmonic mean given by (3.18e). The density and viscosity fields change between time steps and a new solution for the steady state flow field is calculated by solving the system of equations (3.14)-(3.16) using the Stokes flow solver described in Section 2.6.3.

Thermal evolution and melting

The strategy to solve the thermal advection-diffusion problem and the multi-component melting, which are coupled through the consumption of latent heat, is discussed in Section 3.2.2. In particular, the sequence of numerical operations shown on page 109 is used. The applicability of this scheme to geodynamic problems with millions of unknowns was a key issue during its development. The scheme is parallelized and implemented in both the 2-D and the 3-D model.

Approximations to melt migration

The melt produced at depth underneath mid-ocean ridges rises through the mantle and forms the oceanic crust. This process is very complex and represents a separate field of research; see the recent review by Kohlstedt and Holtzman (2009). A very simple numerical formulation is used here, to not unnecessarily increase the complexity of the geodynamic model for the benefit of easier understanding and interpreting the numerical results. The melt migration model should basically have two characteristics: First, it should allow us to crudely approximate the melt fraction in the mantle rocks, given the melting rates everywhere in the domain. Second, it should be simple and robust, so that any feedbacks between melt fraction, melting rate and mantle flow can be understood.

I use the formulation to approximate melt migration considered by Jha et al. (1994), and I briefly summarize the essential assumptions. The melt is assumed to be transported by porous flow (e.g. Turcotte and Schubert, 2002, p. 376), with the melt's buoyancy being the sole driving mechanism (i.e. pure vertical rise of all melts). Assuming further that the melting rates represent a steady state solution for each time step (in particular, melting rates do not change in response to the melt migration), the melt migration can be described by combining the following equations:

$$\Phi = \int_Z M_B \, \mathrm{d}z \qquad \text{(melt flux in vertical direction)} \qquad (3.19)$$

$$w = \frac{K}{\eta_m}(\rho_m - \rho_M)g \qquad \text{(Darcy velocity)} \qquad (3.20)$$

$$K = \frac{b^2\phi^2}{72\pi} \qquad \text{(porosity-permeability relation)} \qquad (3.21)$$

Starting with (3.19), the bulk melting rate M_B is vertically integrated in every time step. This is done numerically by summing up the melting rates along vertical columns, which leads to a melt flux Φ at every depth.[3] The flux per unit area is the the Darcy velocity w. Providing the density ρ_m and viscosity η_m of the melt, (3.20) is used to calculate the permeability K. Finally, given the geometrical parameter b (which in this case is the grain size in the mantle), (3.21) is used to calculate the porosity ϕ, thus melt fraction, at any depth.

[3] Inherent are the two simplifications mentioned before: (a) all melt produced below a certain point has to cross this point during its rise (only buoyancy drives the flow of melt), and (b) the melting rates are steady for the duration of the melt migration (steady state during a time step).

Under the above assumptions, the melt fraction in the mantle can be estimated given the melting rates of all lithologies. The melt fraction enters the equations for density (3.17d) and viscosity (either (3.18c) or (3.18d)).

3.3.2 Boundary conditions and initialization

Boundary conditions

As discussed on page 90, the convergence rate of the viscous flow solver is considerably faster, if the entire domain is ”closed”. This does not mean that no in- or out-flow would be allowed (i.e. fixed walls or symmetry planes everywhere), but that all velocity components normal to the domain boundary have to be prescribed to avoid any unconstrained in- or outflow. In case of ridge geometries, the analytical corner flow solution (Batchelor, 1967) can be used if the material is isoviscous, but it is not correct in the presence of viscosity variations.[4] Nevertheless, using the corner flow solution has two advantages: Volume is preserved within the domain (as required by the Stokes flow problem), because the analytical solution represents an incompressible flow field everywhere, and (2) a stress-free boundary at the bottom can give rise to an unconstrained influx in case of buoyancy forces in the domain, because the viscous resistance of the mantle underneath the domain bottom is missing.

In 2-D, the above problem is addressed by increasing the domain size so that the mismatching velocities of the corner flow solution are far enough away to not affect the mantle flow near the ridge. Distances between the ridge axis and the domain boundaries should be at least 500 km (vertical direction) and 1000 km, (horizontal direction). Since symmetric spreading is assumed, only one half of the ridge needs to be included in the model. The vertical boundary below the ridge axis is defined to be the symmetry plane. Corner flow boundary conditions are imposed at the bottom and the vertical wall distant to the ridge. Plate motion is imposed at the top, a zero-velocity defines the ridge axis. The temperature boundary conditions are 0°C at the top, $T_M = 1315°\text{C}$ at the bottom, and insulating on both vertical boundaries.

Increasing the domain size in the 3-D model to compensate suboptimal boundary conditions is less practical, because it results in a much larger number of unknowns as actually required to study the problem at hand. Instead, the velocity solutions of appropriate 2-D models are used as boundary conditions. This will be discussed below.

[4] Horizontal velocities within the highly viscous lithosphere, for example, are very uniform, because it behaves similar to a rigid plate. This has the consequence that the material transport away from the ridge axis is larger than predicted by the corner flow solution, which in turn is balanced by faster mantle upwelling underneath the ridge.

Initialization

A sophisticated initialization of all time dependent variables is an efficient way to reduce the computational time of numerical experiments. While this may not be as critical for the 2-D calculations, which require about 2–3 hours to calculate 10–15 Myr of ridge melting and mantle flow, the initialization is of great importance for the 3-D calculations. Difficulties arise for a multi-component mantle, because depletion and volume fractions of all lithologies cannot be predicted a priori as a function of depth and distance to the ridge. If a fertile component has a too large volume fraction at a certain depth, its too high productivity in the initial stage of the experiment can result in unrealistically high melt fractions in the mantle rocks, which in turn affect mantle density and viscosity controlling the mantle flow. The latter feedbacks into the melting rate of all components (also at greater depth), so that this coupled system may need few million years to self-adjust. The same problem arises for the initial temperature field: If the initial temperature is too low, the lack of melt production below the ridge in the first time steps changes the mantle flow, hence, subsequent melting. Too high temperatures, on the other hand, result in an comparably extreme melting event at the beginning of the numerical calculation. The evolving highly depleted rocks have to advect out of the melting zone before a potential steady state solution can be reached.

The 1-D model presented in Section 3.2 offers the opportunity to quickly calculate compositional fields and temperature as a function of depth. These values are comparably close to a 2-D steady state solution underneath the ridge axis (if one exists for the problem at hand). The 1-D compositional fields can be used as a starting guess everywhere in the 2-D domain, since they are (by approximation) mainly a function of depth[5].

The 1-D temperature field is superimposed on a cooling half-space solution to account for the colder temperatures in the lithosphere with increasing distance from the ridge. This is required, because the solution for a cooling half space does not represent the correct steady state solution in the 2-D problems. First of all, the latent heat cooling associated with the melting process is not included in the analytical solution, which is compensated by superimposing the 1-D solution. Secondly, the analytical solution is defined for a purely lateral flow field (as it is based on a time-dependent 1-D solution), whereas mantle flow near the ridge has a strong vertical velocity component. This mismatch near the ridge is also partly corrected by the 1-D profile that includes the upward advection against the conductive cooling from the top.

Every 2-D calculation is therefore preceded by a (few seconds long) 1-D calculation with the same mantle composition. A similar initialization could be used for the 3-D

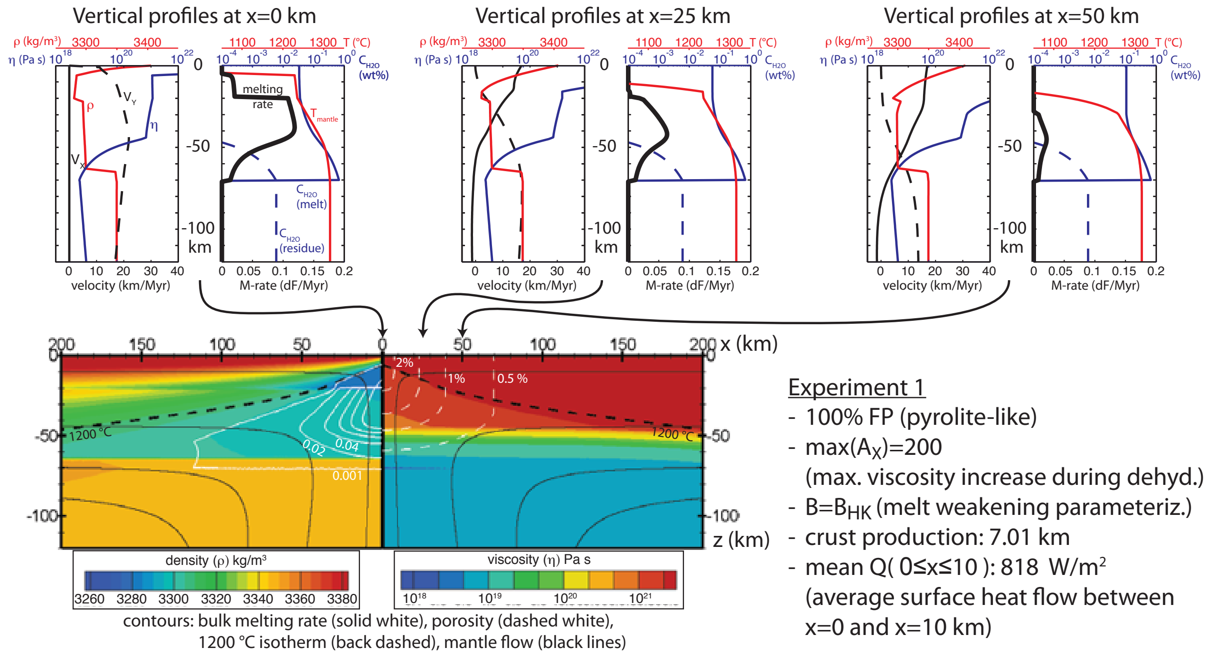
[5] Away from the ridge, mantle flow is approximately lateral, so that all rocks evolving during decompression melting remain at the depth, at which they are advected sideways out of the melting region.

models as well, however, the steady state 2-D results represent a better alternative, as they include a very exact temperature field as a function of both depth and distance to the ridge. If a steady state solution is sought in a 3-D geometry, a good initialization using previous 2-D results can easily save 80% or more of the computational time.

3.3.3 2-D experiments

The first set of 2-D experiments (Fig. 3.8–3.9) shows the response of mantle flow to two competing effects: The dehydration related increase in mantle viscosity and the weakening of mantle rocks with increasing melt content. A single mantle component with 200 ppm water is assumed in these experiments (i.e. a homogeneous, pyrolite-like mantle composition), the half-spreading rate is 17 mm/yr (=km/Myr).

In the first experiment (Fig. 3.8), the effect of melt on the rock viscosity is parameterized using the conservative estimate by Hirth and Kohlstedt (2003) (Eq. (3.18c) is substituted in (3.18a)). The mantle viscosity is assumed to increase by a factor of 200


Vertical profiles at x=0 km

Vertical profiles at x=25 km

Vertical profiles at x=50 km

contours: bulk melting rate (solid white), porosity (dashed white), 1200 °C isotherm (back dashed), mantle flow (black lines)

Experiment 1
- 100% FP (pyrolite-like)
- max(A_X)=200
  (max. viscosity increase during dehyd.)
- B=B_{HK} (melt weakening parameteriz.)
- crust production: 7.01 km
- mean Q(0≤x≤10): 818 W/m²
  (average surface heat flow between x=0 and x=10 km)

Figure 3.8: *Competing effects of melt weakening and dehydration stiffening; see also Fig. 3.9. Right and left half of the contour plot shows different variables of the same experiment (the calculation is symmetric around x=0). Left half: colors show density and isolines show melting rate in unit of "fraction per Myr"; right half: colors are viscosity, dashed white isolines are porosity. Mantle flow (solid black lines) and the 1200° C isotherm (dashed black) are shown in both panels. Vertical profiles at three distances from the ridge show the following variables: left graph in each group: density (red), viscosity (blue) and vertical speed (black, dashed); right graph: temperature (red), melting rate (thick black), water in residue (blue, dashed), water in pooled melt (blue, solid). Plate speed on top: 17 km/My; melt weakening parameterized using Eq. (3.18c). All figures on the following pages use the same notation. See text for a discussion of this experiment.*

as the water partitions into the melt and the rock dries out, within the range suggested by Karato and Wu (1993) and Hirth and Kohlstedt (1996). The dehydration stiffening starts immediately after the onset of melting at about 70 km depth and viscosities of $10^{21} - 10^{22}$ Pa·s are reached at 45 km depth. The upper 2/3rds of the melting zone become very stiff and the melt weakening has almost no effect on the viscosity. The resulting flow pattern is a passive mantle upwelling, driven solely by the divergence of the lithospheric plates. The vertical upwelling speed is close to the half-spreading rate over a large depth interval. A triangular shaped melting region forms as a result of the gradually decreasing decompression rate with increasing distance from the ridge. The *sp-plg* phase transition reduces the melting rate at 21 km depth, and melting stops completely where conductive cooling from the top controls the mantle's temperature. The discontinuities in the density field at 64 km and 21 km depths correspond to the density changes across the *gt-sp* and *sp-plg* phase transitions, resp. (see descriptions in Section 3.2.2).

The second experiment (Fig. 3.9, top) also includes a 200-fold viscosity increase, but the melt weakening effect is assumed to be stronger by using parameterization (3.18d). In particular, the stronger weakening effect of very small amounts of melt (also shown in Fig. 3.7c, solid line) leads to a ∼15 km thick low viscosity region at the base of the melting zone before the dehydration related increase in viscosity starts to dominate at shallower depths. The mantle upwelling remains passive, because viscosity at shallower depth is still comparably high (about 10^{20} Pa·s) and the low viscosity zone is too thin to allow for significant dynamic upwelling.

The same parameters as in experiment 2 are used in the third experiment (Fig. 3.9, bottom), except that the dehydration-related increase in viscosity is reduced to a factor of 2. The low viscosity zone now extends from the base of the melting region up to the conductive cooling front – a thermal lithosphere has formed as opposed to the compositional lithospheres in experiments 1 and 2. Because the mantle density is simultaneously lowered by the buoyancy effects of Fe-depletion and melt, an active upward flow initiates, so that the maximum upwelling speed is about twice as high as the half-spreading rate (see left vertical profile of experiment 3). Mantle flow is more focussed towards the ridge, melting rates are higher, and the melting region is narrower than in the passive flow scenarios.

The next set of 2-D experiments (Fig. 3.10) aims to study the potential changes in mantle flow, if the mantle is assumed to be composed of two and three lithologies instead of a single homogeneous lithology. All parameters are identical to those in experiment 2 (i.e. 200-fold increase in viscosity during dehydration and the melt weakening parameterization in Eq. (3.18d)), except that the number of mantle components is varied, while preserving the same bulk composition. As shown before in the 1-D experiments (Fig. 3.4 on page 121), the enriched pyroxenite starts to melt first at around 98 km depth. Melting rates are low, because PYX is in a wet melting regime and continuously loses its water to the

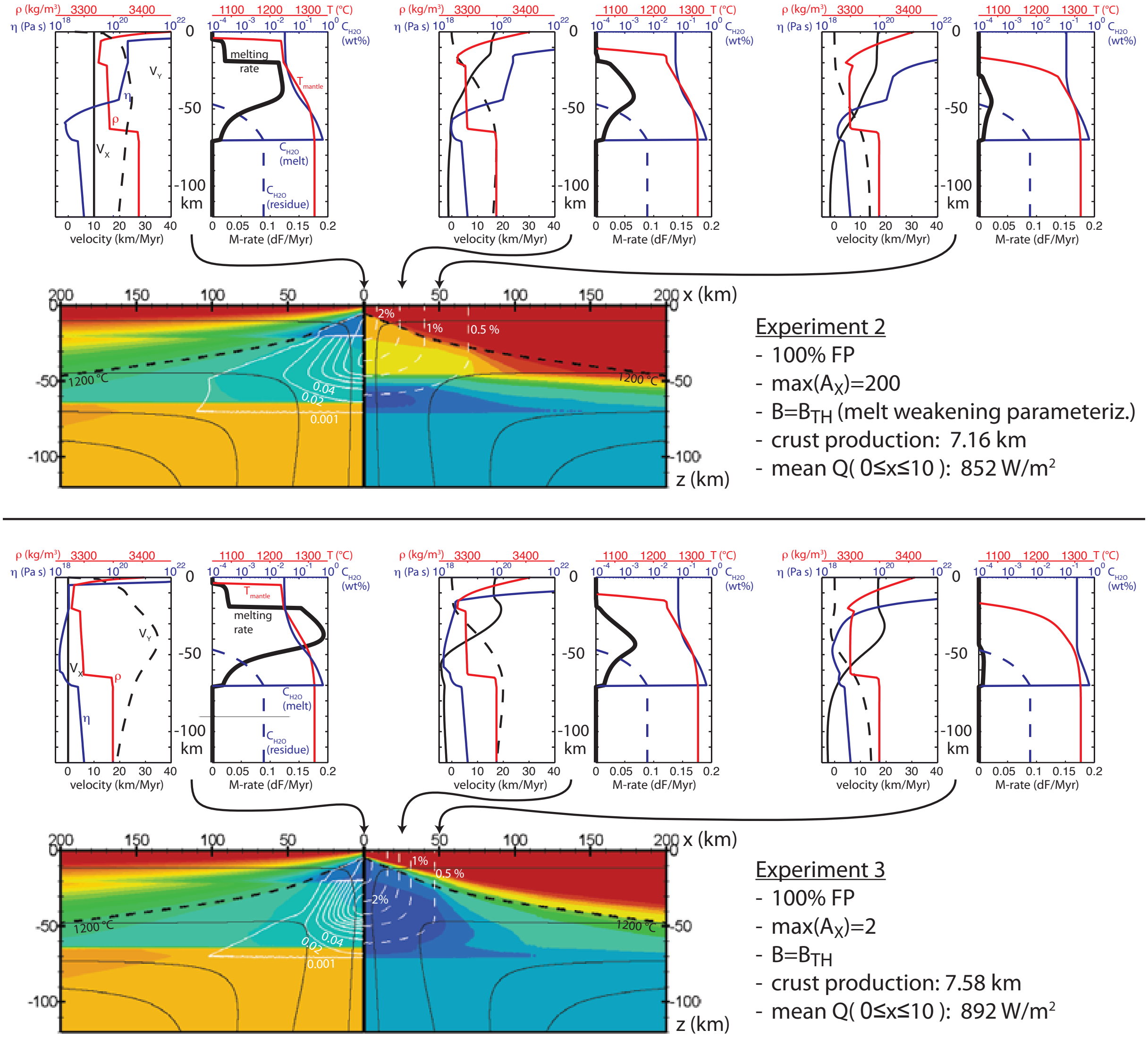
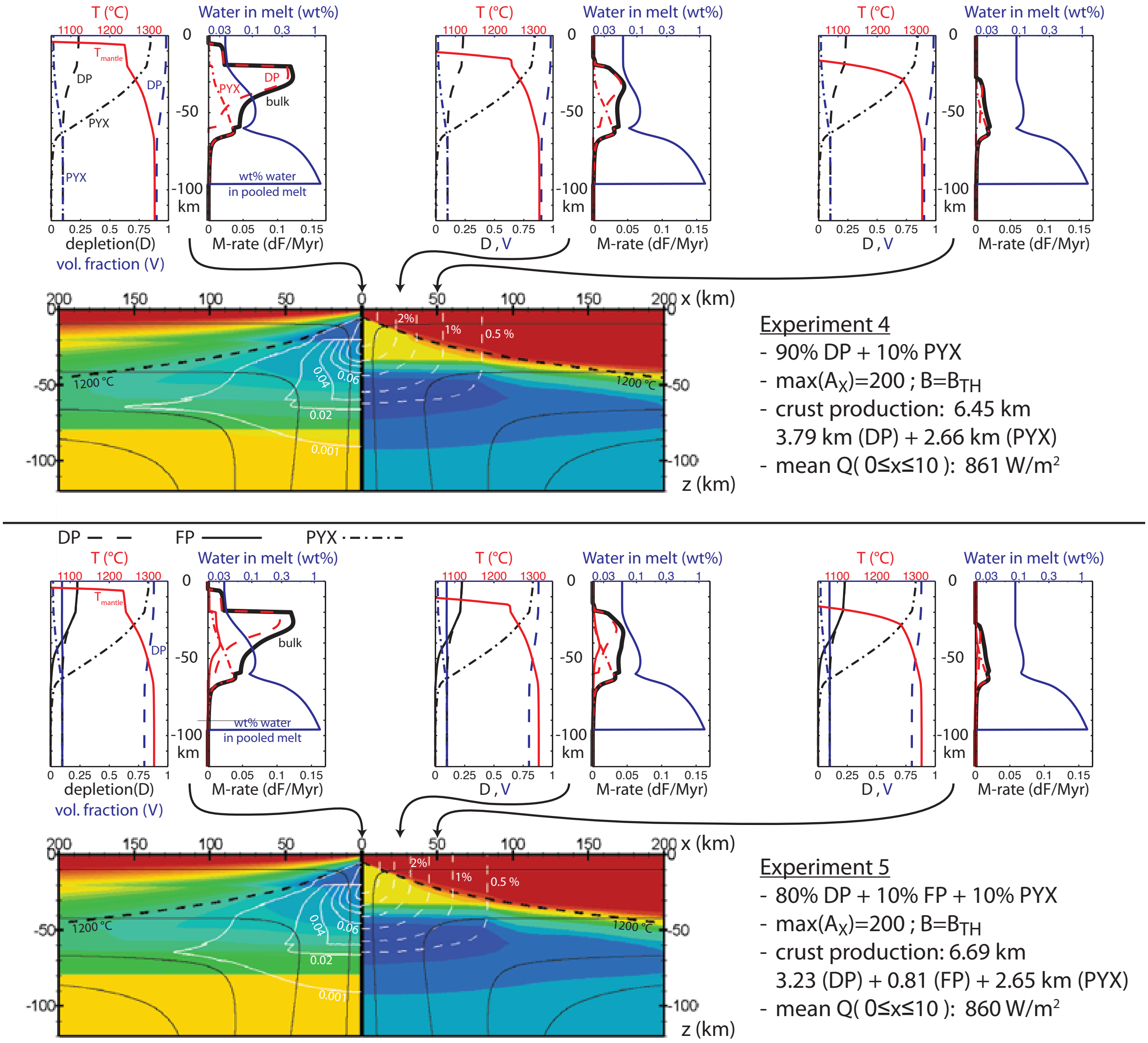


Figure 3.9: *Competing effects of melt weakening and dehydration stiffening. Melt weakening parameterized using Eq.* (3.18d)*. See Fig.* 3.8 *and text for more information.*

melt, which inhibits further melt production. As PYX gradually enters the dry melting regime around 65 km depth, its melt productivity increases.

Compared to experiment 2, melting of the peridotite starts at a much shallower depth. This delay has two reasons: (1) The peridotite in experiment 4 has a 10% initial depletion (in order for PD+PYX to have the same bulk composition as FP) and (2) the deep melting of PYX consumes latent heat, which further delays the onset of peridotite melting. While remaining in a sub-solidus state, the DP keeps its initial water content, so that 90% (and more) of the rock do not experience the dehydration related increase in viscosity[6].

[6] As outlined in Section 3.2 (see page 114), all lithologies are assumed to have equilibrated to the same chemical potential for hydrogen during the 100s of million years residence in the mantle before being advected into the melting region. Assuming a chemical diffusivity of $10^{-7} - 10^{-8}$ m^{2}s^{-1}, hydrogen can diffuse 10s of kilometers in 100 Myr to reach a initial chemical equilibrium between the lithologies. Since



Experiment 4
- 90% DP + 10% PYX
- max(A_X)=200 ; B=B_{TH}
- crust production: 6.45 km
  3.79 km (DP) + 2.66 km (PYX)
- mean Q(0≤x≤10): 861 W/m^2

Experiment 5
- 80% DP + 10% FP + 10% PYX
- max(A_X)=200 ; B=B_{TH}
- crust production: 6.69 km
  3.23 (DP) + 0.81 (FP) + 2.65 km (PYX)
- mean Q(0≤x≤10): 860 W/m^2

Figure 3.10: *Potential effects of a heterogeneous mantle composition on mantle flow at a ridge. All settings as in experiment 3, except that the mantle is composed of two lithologies (experiment 4) and three lithologies (experiment 5). See text for more information.*

In addition, PYX-melts further weaken the rock below 60 km depth, as they percolate upwards. The effective rock rheology remains at a low viscosity until DP starts to melt at about 60 km depth and viscosity gradually increases.

Experiment 5, including three lithologies, is very similar to experiment 4 with regard to the evolving viscosity structure. The onset of DP melting is slightly more delayed than in the previous example, which has also been observed in the 1-D experiments. This is explained by the latent heat cooling of FP melting. The viscosity and density fields of experiment 4 and 5 are very similar, in that a low viscosity layer forms between the onset

only the melting lithologies lose their water to the melt, the non-melting lithologies preserve their water content and remain rheologically weaker. The weakest volumetrically important lithology is assumed to control the effective viscosity of the aggregate →Eq. (3.18e).

of PYX-melting and the base of a thin compositional lithosphere, which is defined by the onset of melting of the most depleted mantle lithology.

Summary of the 2-D experiments

This set of calculations clearly demonstrates that the outcome of an experiment is controlled by the competing effects of increasing viscosity as the melting part of the rock dehydrates, and decreasing viscosity as the rock contains a larger melt fraction. The water-related change in rheology is directly linked to the depletion of the mantle rock and is therefore advected with the mantle flow field. This makes it comparably easy to include this effect in numerical mantle flow models, provided that a relation between viscosity and water content is provided.

The melt fraction in the mantle, however, depends on the melting process as well as on the migration of melts. Since a very simple approximation is used in the above models, the feedbacks between mantle flow and melt production are incompletely incorporated. If, for instance, melt migration is assumed to depend on lateral pressure gradients, this feedback will become more complex: The gradient in the dynamic pressure of the mantle flow field is related to the mantle's viscosity, which in turn depends on melt fractions, thus on the direction of melt migration.

I have also tested a different formulation for porous flow of melt that includes lateral pressure gradients as an additional driving force. The numerical formulation is based on Cordery and Phipps Morgan (1993) and assumes that the permeability is constant everywhere, thus, independent of the melt fraction. This formulation has been used during post-processing only, to estimate the contribution of off-axis melt production on the crustal accretion at the axis. Including this formulation in the numerical model may require iterations between mantle flow and melt migration, because both are strongly coupled by the viscosity.

Ultimately, a numerical solution for pressure driven porous flow is required to estimate melt ascent paths more realistically. In contrast to porous flow in hydrothermal systems, where the porosity and permeability of the matrix can be assumed (as an approximation) to be independent of the fluid passing through the pores, melt migration exhibits a major difficulty: In melt migration, the porosity is equal to the melt fraction, which in turn depends on where melts are formed and migrate. This feedback results in a strongly non-linear system of equations, which physically represents a so-called two-phase-flow problem. This could be addressed in future work.

The transition from passive to active mantle upwelling goes along with changes in the heat flow at the surface and the crustal thickness (these quantities are given in Fig. 3.8–3.10). However, the changes are comparably small in that crustal thickness as well as surface heat flow vary by less than 10%. Given that the differences in the density fields are also marginal between the experiments, quantifying the mantle viscosity would be the best way to distinguish between the different mantle flow patterns. Unfortunately, assessing quantitative values for the mantle viscosity is problematic using geophysical techniques.

The phase transitions, if prescribed to occur at specific depths, are found to have no influence on the mantle flow. Since no lateral density variations can result from this formulation, the phase transitions only affect the lithostatic pressure, which does not drive any flow. Only if a thermodynamic treatment of the phase transitions is considered (i.e. temperature and composition control the pressure at which the minerals transform), will lateral density differences arise that lead to changes in the mantle flow. The above implementation may therefore be viewed as a first step towards the full coupling of a thermodynamic code.

As shown in the 1-D experiments, adding water to the initial mantle composition does not increase its inherent melt productivity. In 2-D, changes in the melt productivity can potentially arise from the lower viscosity of a ”wetter” mantle underneath the ridge axis. Lowering the viscosity leads to an increasingly dynamic mantle upwelling, which in turn increases the melting rates by increasing the mass flux through the melting region. Comparing the 2-D experiments 1 and 3, however, shows that a significantly increased mantle upwelling is required to cause an obvious crustal thickness anomaly.

The low viscosity region underneath the compositional lithosphere leads to a slightly more focussed mantle flow until the base of the compositional lithosphere is crossed. This focussing leads to a narrower melting zone as in the homogeneous mantle scenario. The low density and low viscosity region supports active (buoyant) mantle upwelling, leading to higher melting rates right underneath the ridge axis. Similar flow fields to the above experiments have been reported by Braun and Sohn (2003), who suggest a different mechanism for the formation of a low viscosity region at this depth: a change in the creep mechanism at the onset of deep wet melting, which lowers the mantle viscosity. In the experiments presented above, the low viscosity zone evolves without any assumptions on changes in the creep mechanism.

While the changes in mantle upwelling are relatively moderate in the 2-D experiments (as they are forced to be a sheet-like upwelling), the same scenario in 3-D could potentially lead to convective instabilities within the low viscosity region. If Rayleigh-Taylor-like instabilities form in the along-ridge direction, their wavelength is likely to be similar to the vertical extension of the low viscosity region (∼30–50 km). This could explain the typical length-scale of segmentation at slow spreading ridges as well as the wavelength of ”bulls-eye” Bouguer anomalies derived from crustal thickness variations parallel to the ridge axis (J Lin et al., 1990; Kuo and Forsyth, 1988). This interesting question, however, requires more detailed exploration with 3-D experiments and this is beyond the scope of the work reported in this thesis.

3.3.4 3-D experiments

A 3-D numerical model allows to study changes in mantle flow and melt migration in the along-ridge directions. These variations may result from (1) the presence of transform faults that intersect and displace the ridge axis, (2) an oblique spreading direction, (3) changes in mantle composition or (4) mantle temperature in the along-axis direction, or (5) Rayleigh-Taylor-like gravitational instabilities in the weak and buoyant melting region, even if no variations in temperature, composition, and ridge axis exist along-axis. The 3-D experiments presented in this and the following section show selected experiments that address these potential sources for non-uniform melt production (i.e. a potential origin for observed along-axis variations in crustal thickness) beneath mid-ocean ridges.

3.3.5 Model setup and boundary conditions

The effect of a segmented spreading center on mantle flow and the melting processes is studied next. An idealized straight ridge axis is perpendicularly intersected by a single transform fault (TF). The ridge offset across the TF is varied between 50, 100, and 150 km. Fig. 3.11 shows the model geometry and describes the boundary conditions on velocity and temperature. Velocities on the bottom and the x-perpendicular walls are taken from preceding 2-D experiments with the same physical and petrological parameters.

The half-spreading rate is set to 17 km/My, similarly to the speed of plate motion at the Mid-Atlantic ridge near Ascension Island, which is subject to a case study in the subsequent section. The initial temperature field and the compositional fields (i.e. volume fraction and depletion of all lithologies considered) are also taken from the 2-D model that provides the velocity boundary conditions. The 2-D results represent the steady state solution for an uninterrupted ridge and do not fit as well in the vicinity of the transform fault. However, they help to avoid an unrealistically strong melting event in the beginning of the 3-D calculation.

A three-lithology mantle composition is assumed, consisting of 80% depleted peridotite, 10% fertile (pyrolite-like) peridotite, and 10% enriched pyroxenite. The more conservative parameterization by Hirth and Kohlstedt (2003) is used for the weakening effects of melt, given by (3.18c), with the intention to study a ”more passive” mantle upwelling first, and increase the feedback between melt production and mantle flow in subsequent model

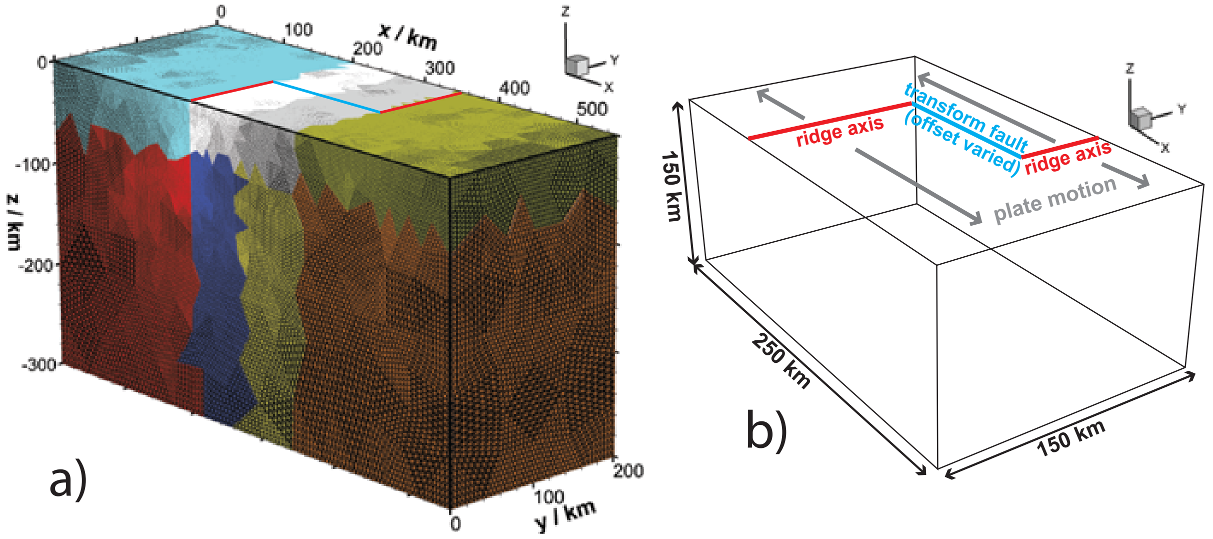


Figure 3.11: *Geometry and boundary conditions (bc) for the 3-D experiments to study the effects of transform faults (TF) on mantle flow and melting. (a) The domain covers 550x200x300 km in x-, y-, and z-direction, different mesh colors show how the model is distributed onto 8 CPUs. The red and blue lines on top show the fixed position of ridge axes and the transform fault, resp. Velocity bc: y-perpendicular walls are symmetry planes, plate motion of 17 km/Myr enforced on the top, velocity bc for bottom and x-perpendicular walls are taken from 2-D experiments with identical parameters. Ridge axes have a prescribed zero horizontal velocity, while velocities along the transform faults are horizontally unconstraint. Temperature bc:* $0°\,C$ *at the top,* $T_M = 1315°\,C$ *at the bottom, insulating everywhere else. (b): A section of the domain detailing the transform fault. The same view is used in Fig. 3.12.*

calculations. A 200-fold viscosity increase over the full dehydration is assumed, the initial bulk water content in the mantle is 200 ppm. Melt migration is approximated as discussed in Section 3.3.1, i.e. vertical, buoyancy driven porous flow.

Each 3-D experiment contains about 3 Mio velocity unknowns (i.e. about 1 Mio nodes) and has been calculated on 8 CPUs. The subdomain configuration (Fig. 3.12a) was generated using a self-developed nested bisection algorithm (see page 64). The experiments ran until 15 Myr of plate spreading was modeled, which was achieved after about 48 hours of computation time.

3.3.6 Results

Fig. 3.12 shows the 3-D flow field and the extension of the melting region in the mantle for three model calculations. The left column shows mantle flow streamlines (all models have reached steady state), which are colored by the bulk melting rate M_B of the mantle. M_B is the sum of the melting rates for each lithology, weighted by the volume fraction of each lithology, and hence a measure for the mantle rock productivity. The gray isosurface shows the extension of the melting region; it is defined by $M_\mathrm{B} = 0.01\frac{1}{Myr}$, i.e. on this surface 1% of the mantle rock would melt in one million years.[7]

The right column in Fig. 3.12 shows cross-sections below one ridge axis and the transform fault (M_B is color-coded). Also shown are the melting regions of each lithology: Two lines mark, where the melting rates are 10% and 90%, respectively, of the maximum melting rate for the lithology. The white lines show the PYX melting region, i.e. the outer line marks $M_\mathrm{PYX} = 0.1 \cdot max(M_\mathrm{PYX})$ and the inner $M_\mathrm{PYX} = 0.9 \cdot max(M_\mathrm{PYX})$. Grey lines correspond to FP and black lines to DP. This allows to visualize the extension of each lithology's melting region, as well as its region of major productivity.

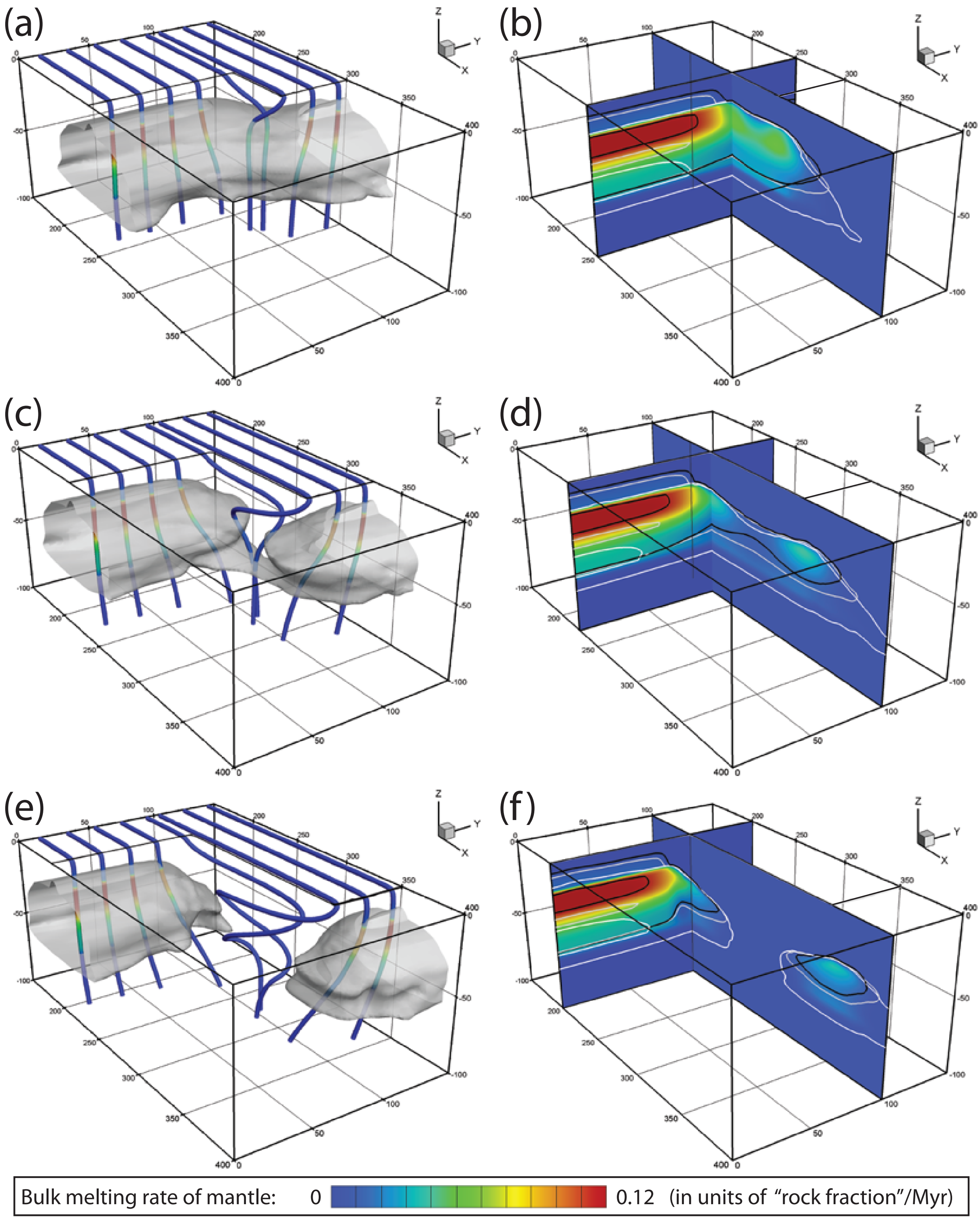
Only the length of the TF is varied in these experiment with all other parameters kept identical. In all experiments, melting at greater distance from the TF (e.g. near the wall at $y = 0$) is very similar to the 2-D model calculations with the same parameter settings. Melting rates decrease towards the segment edge at the TF. Below the 50 km long TF (Fig. 3.12a,b), melting rates are reduced by about 50%, but the melting zones of all lithologies remain interconnected. Mantle melting rates drop to about 10% central beneath the 100 km long TF (DP melting regions almost separate). A 150 km TF leads to separate melting regions underneath each ridge axis.

Although weakening of the mantle within the melting region is limited due to the chosen parameterization for the effect of melt on rheology, the diverging highly viscous lithosphere plates cause a ”suction” that attracts mantle from underneath the transform faults. This effect was also seen in the isoviscous experiments of Phipps Morgan and Forsyth (1988) and is indicated by the curled stream lines in Fig. 3.12a, c and e. The effect becomes stronger for longer TFs and can also be seen from the diagonally upwards directed mantle flow towards the right ridge in Fig. 3.12e.

Melting at greater depth seems to be less affected by the transform faults. Fig. 3.13 shows M_B and the melting rates of each lithology (white, gray and black lines) on horizontal slices at different depths in the 3-D box. The slices at 25 km depths show distinct melting regions, except for the 50 km long TF. Towards greater depth, the broadening melting regions start to partly overlap. Only the 150 km TF separates the melting regions of all lithologies when using the above definition for the outer edge of the melting region,

[7] In the center of the melting region, $M_\mathrm{B} \approx 0.12$. Since these rocks need about 2 Myr to cross the melting region, the maximum depletion is about 20–24%.

Figure 3.12 ***(facing page)*****:** *Left column: Mantle flow streamlines (colored by bulk melting rate M_B) and extension of the melting region(s) (gray surfaces, $M_\mathrm{B} = 0.01/Myr$). Right column: cross-sections below the ridge axis and the transform fault showing M_B (colored). The melting region(s) of each lithology are enclosed by white (PYX), gray (FP) and black lines (DP). Two lines are shown for each lithology: outer line = $M = 0.1 \cdot max(M)$ and inner line $M = 0.9 \cdot max(M)$. Three lengths of the TF are modeled: 50 km (a+b), 100 km (c+d), 150 km (e+f).*


(a)
(b)
(c)
(d)
(e)
(f)
Z
Y
X
0
50
100
200
250
300
350
400
-50
-100
Bulk melting rate of mantle: 0
0.12 (in units of "rock fraction"/Myr)

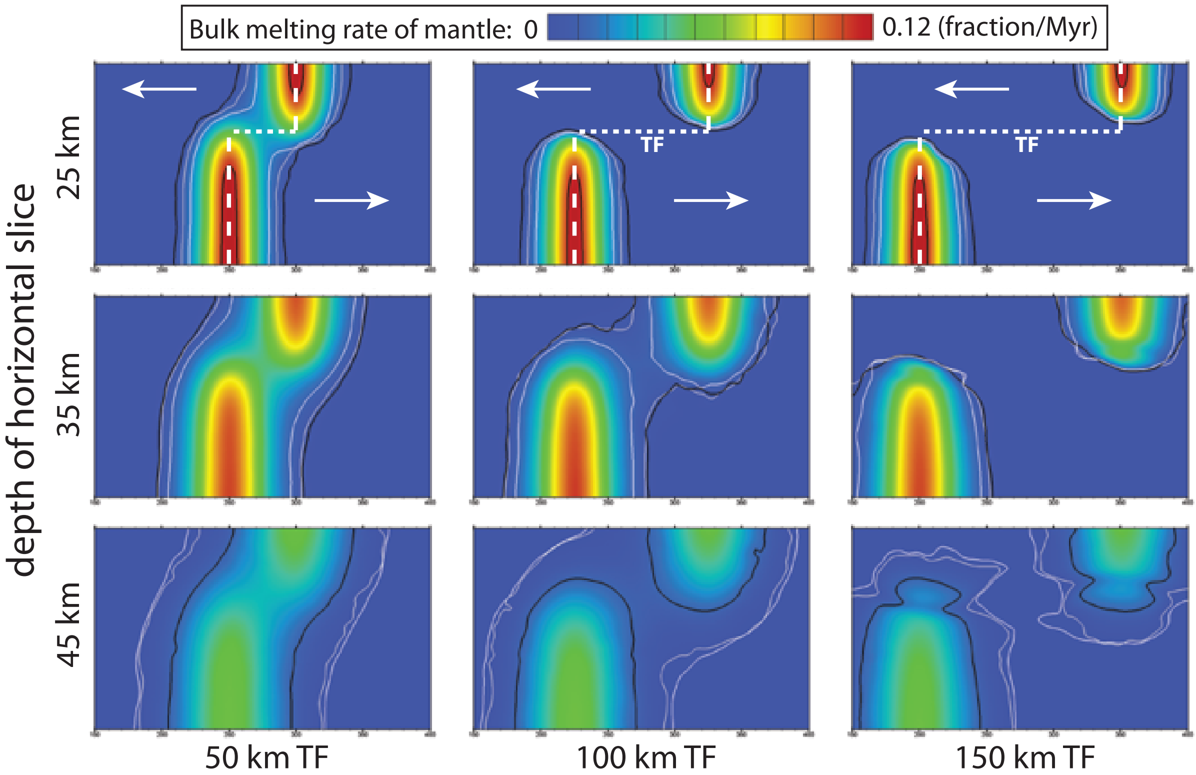


Figure 3.13: *Horizontal slices at three depths through the 3-D box. Three experiments are shown, in which the length of the TF has been varied. Colors show the bulk melting rate of the mantle M_B. Lines show the regions of 10% and 90% of the maximum melting rate for each component (explained in Fig. 3.12).*

although very small rates of melt production are still seen underneath the TF. Consequently, lithologies that melt at shallower depths are more affected by the transform fault than mantle components that start to melt at greater depth.

The patterns seen in Fig. 3.13 are partly explained by the thermal structure associated with TFs of different lengths. Fig. 3.14 shows vertical cross-sections underneath the ridge axis that continue into the older lithospheric plate on the other side of the TF. The longer the TF, the older (and colder) the plate that passes next to the tip of the ridge. Conductive cooling extends few 10 km into the melting region of the nearby ridge and reduces the melt production (Forsyth and Wilson, 1984; Phipps Morgan and Forsyth, 1988). The second factor that limits melt production near the TF is the reduced vertical upwelling of the mantle. In the presence of a compositional lithosphere near the ridge, mantle upwelling is very focussed towards the ridge axis as indicated by the streamlines in Fig. 3.12. In contrast, shallow mantle flow underneath long transform faults is mainly parallel to the plate motion.

Upwelling mantle transports thermal energy towards shallower depths. An indirect measure for the amount of thermal energy that is transported towards the seafloor is the heat flux at the seafloor. Fig. 3.15 shows the heat flow calculated for the three models.

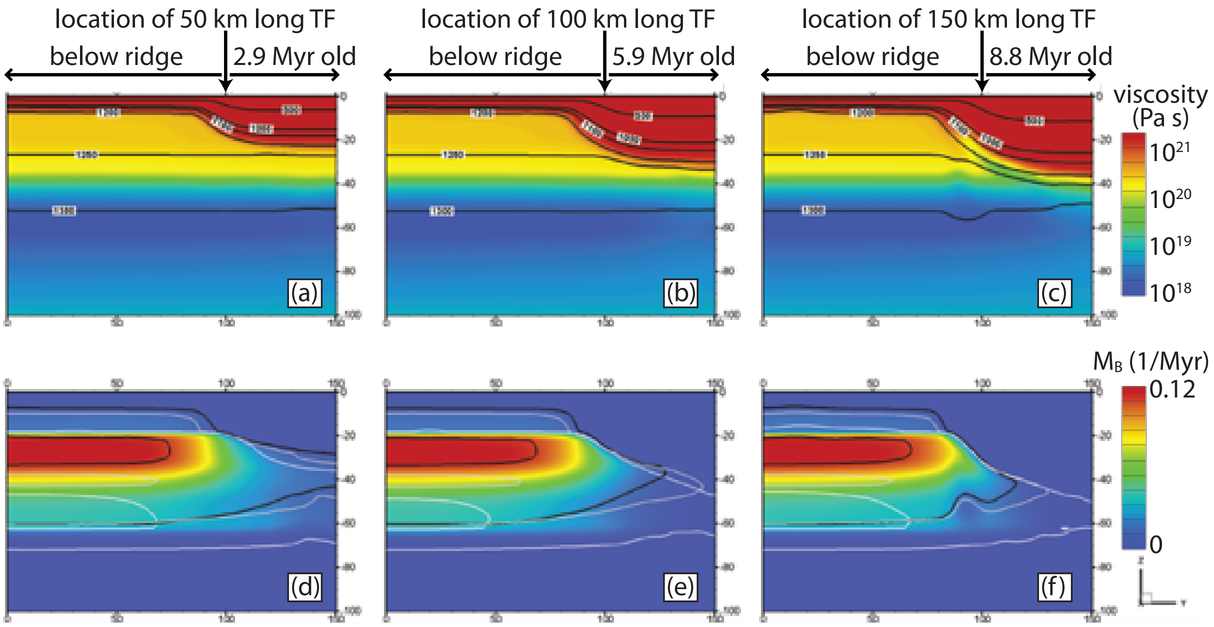


Figure 3.14: *Vertical cross-sections parallel to the ridge axis that extend into the older lithosphere beyond the transform fault. The same experiments as in the previous figures are shown. Top row: viscosity structure (colored) and isotherms. Bottom row: Bulk melting rate in the mantle, as well as melting regions for each lithology (explained in Fig. 3.12).*

Because the plate spreading rate is comparably slow (17 km/Myr), the thickness of the thermal lithosphere increases quickly with distance to the ridge axis. High heat fluxes are therefore concentrated at the ridge axes and fade as the age of the plate increases. Note that neither the thermal energy transported by ascending melts, nor the redistribution of thermal energy by hydrothermal convection are considered here. Both effects are likely to have a strong influence on the heat flux actually measured at the seafloor. The values shown here may be viewed as the amount of thermal energy that potentially is transported to the base of the oceanic crust.

Summary and outlook

Transform faults appear as colder regions in the shallow mantle, which is contrary to the results of Behn et al. (2007). In their study, brittle weakening of the rheology changes mantle flow near the TF, which makes a comparison with the results presented here very difficult.

If colder shallow temperatures are present near the TF (Fig. 3.14), the temperature drop ”chops off” the melting zone so that the relative fraction of deeper (wet) melts becomes larger in the vertically accumulated melts. Thus, melts with a higher water content are predicted to be found near the transform faults and especially within the TF. This is shown by means of vertical 1-D profiles at different locations in the ridge-

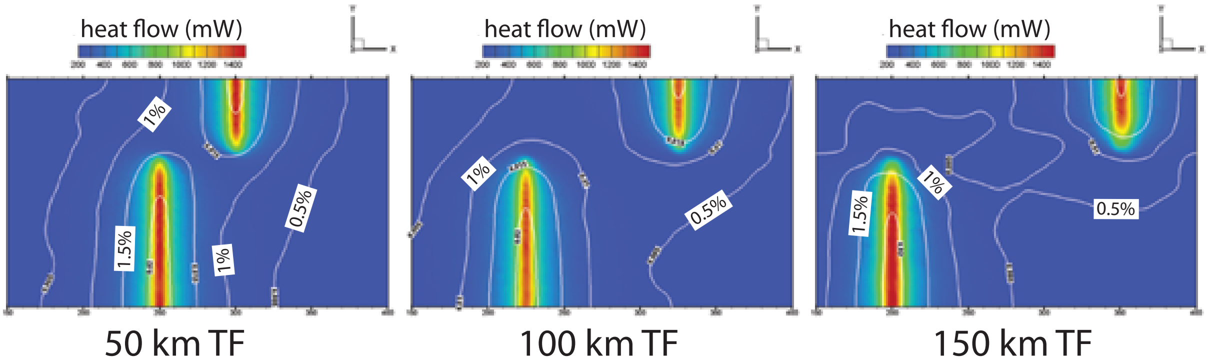


Figure 3.15: *Surface heat flux for the three experiments (colored) and porosity (white lines) in the uppermost mantle. The latter results from vertically integrating the melting rates as described in Section 3.3.1.*

transform fault system (Fig. 3.16). The presence of wetter melts near TFs is in agreement with a recent numerical study by Ligi et al. (2005), who calculated a stronger signature of deep melts near TFs using an analytical corner flow solution for the mantle flow. It is important to keep in mind, that no lateral migration of melts is considered in our model; this could allow water-rich melts to migrate into the ridge melting zone, within which dry melts dominate.

Not only does the water content in the melts vary if melting stops at greater depth, but also the melt composition itself will vary. Below the transform fault, lithologies with a deeper onset of melting (like PYX) contribute more to the pooled melts than they would in a ”normal” ridge melting zone (compare the melting rates of the lithologies in Fig. 3.16). This deep melting of more fertile or enriched mantle heterogeneities has been suggested by Phipps Morgan and Morgan (1999) as a mechanism for creating depleted MORB and more enriched OIB basalts from the same upwelling mantle mixture. (Ito and Mahoney, 2005) have further studied this idea using analytical flow fields. The 2-D and 3-D models developed in this thesis allow to test this ”two-stage-melting” process (Phipps Morgan and Morgan, 1999) in more realistic mantle flow fields, in which also the feedback between mantle flow and melt production can be considered.

In the 3-D calculations conducted so far, no convective instability in the lower, weak part of the melting region has been observed. This is interesting because much simpler rheological parameterizations with large regions of weak mantle within the uppermost ∼80 km beneath the ridge do generate diapiric instabilities for similar buoyancy effects when the sub-axial mantle is weaker than about $5 \cdot 10^{18}$ Pa·s (Parmentier and Phipps Morgan, 1990). The formation of along-axis variations in mantle flow (e.g. Rayleigh-Taylor-like instabilities) will be related to the surrounding viscosity, so that a stronger weakening effect of melts or a generally lower viscosity in the uppermost mantle can lead to along-axis variations.

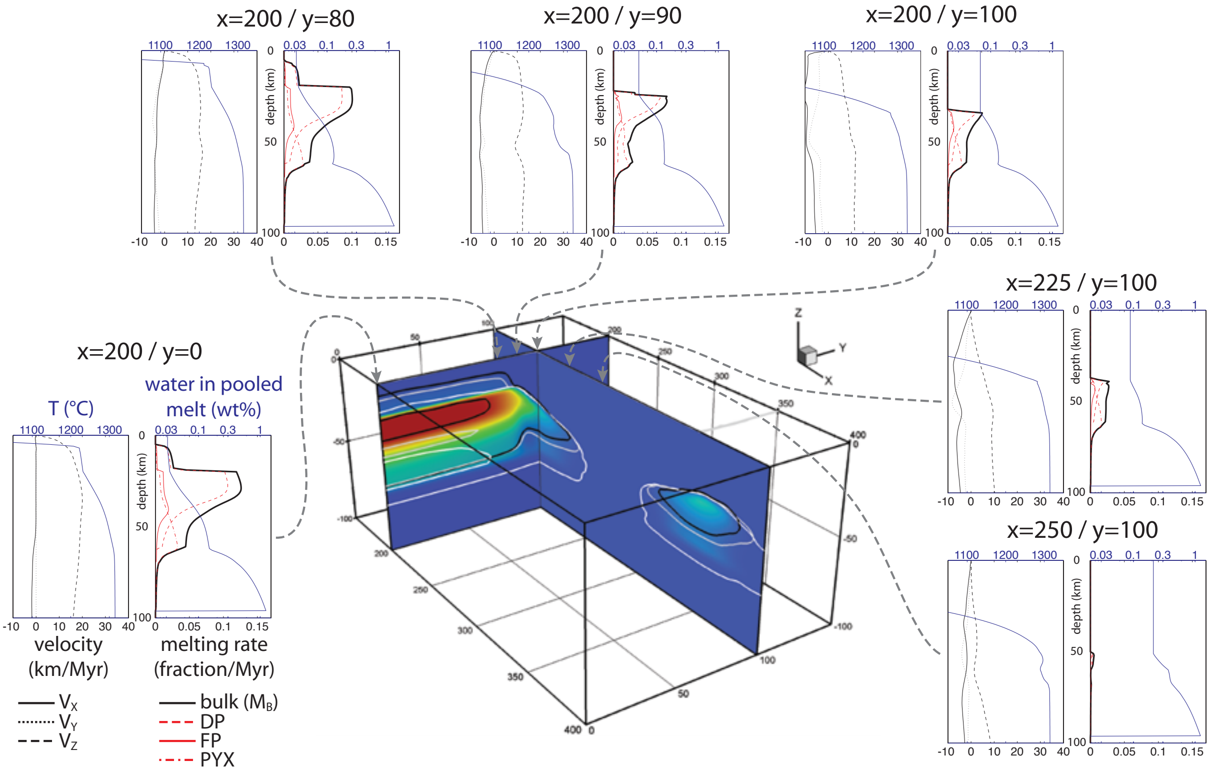


Figure 3.16: *1-D vertical profiles at different locations in the 150 km TF experiment. The left panel in each group shows the three velocity components and temperature, the right panel shows melting rates (bulk and for each component) and the water content in the (vertically) pooled melts. Towards the center of the transform fault, the shallowest mantle becomes colder and reduces the height of the melting column. The water content in the pooled melts increases as the deeper wet melts become a larger fraction in the pooled melt).*

Another requirement for diapiric instabilities is that sufficiently large lateral variations in buoyancy force are present near the base of the low viscosity region. These could arise from trapped melt in the mantle, prior to the onset of melt migration, but this effect is not modeled here. Another source of lateral density contrasts could result from the mineral phase transitions, if they are included as a function of temperature and composition. Fe-depletion buoyancy on the other hand, increases with the degree of melting (thus towards shallower depth) and could act to stabilize mantle upwelling. The fact that I do not find these instabilities with more ”realistic” mantle rheologies indicates that these features may not arise so much from the flow itself (as proposed by Sparks and Parmentier (1993); Jha et al. (1994)) but from focussing feedbacks between melt transport and mantle upwelling/melting processes. Once centers of upwelling have formed, they might be stable because of the positive feedback between faster mantle upwelling and higher melt production.

Even when using a less intense melt weakening formulation as done in the above experiments, a low viscosity region forms between the depth where the first melts are produced and the depth where the most depleted lithology starts to melt and dehydrate. The exact extension of this weak region also depends on the volume ratios of the lithologies and the initial water content of the mantle. A mantle mixture with lower initial water content should have a higher viscosity when entering the melting region, and the subsequent increase in viscosity should be more limited, so that a more uniform viscosity structure in the melting region might evolve.

An extreme situation could occur if a fraction of very depleted mantle rocks is advected into the melting region. Although strongly depleted and removed of their water during the last melting event, these rocks could have regained some fraction of water by diffusive equilibration to ambient more water rich lithologies as they reside in the mantle for millions of years. If so, these wet refractory rocks should have a comparably low viscosity, because they contain a large fraction of olivine (likely to be the weakest mineral in the upper mantle (Karato and Wu, 1993)), whose strength is additionally reduced by the presence of water. Since refractory rocks are unlikely to melt to a significantly larger degree, they would maintain their water content and lower the viscosity over the entire range of the melting column. This could give rise to buoyantly upwelling mantle flow. Furthermore, melting of ambient, more fertile rocks could be enhanced by (1) the heat stored in the depleted rocks (which is partly available for melting of neighboring, more fertile rocks), and (2) the buoyant upwelling that increases the mass flux through the melting region.

3.4 Case study: The Mid-Atlantic Ridge near Ascension Island

3.4.1 Introduction

This section presents selected results of a case study on mid-ocean ridge melting anomalies, which are observed at the Mid-Atlantic ridge (MAR) south of the equator. This part of the MAR is divided into several segments that are displaced by large-offset (200 km and more) as well as smaller transform faults (TF). The region between 2°–14°S is the study area of the German priority program SPP 1144, and has been the site of several ship cruises since 2004.

In this region, the MAR has a half-spreading rate of about 17 km/Myr (Bruguier et al., 2003), which classifies it as a slow spreading ridge. The morphology of such a spreading center is typically dominated by a pronounced axial valley (Small, 1998). However, ridge morphology, ridge axis bathymetry, and crustal thickness vary considerably along the MAR within this region, more precisely, between two large fracture zones named Ascension fracture zone (AFZ) at 7°S and Bode Verde fracture zone (BVFZ) at 12°S.

The most prominent geological features in this region are the active volcanic island of Ascension 80 km west of the MAR (7°55' S, 14°20' W) and a melting anomaly below the MAR at 9°30' S. The latter has caused an uplifted ridge axis without axial valley, which is more typical for a fast spreading ridge like the East-Pacific Rise (EPR). Above the melting anomaly, crustal thicknesses reach up to 10 km (Minshull et al., 1998) and large seamounts have formed, the largest of which (Grattan seamount) rises up to 72 m below sea level.

Possible explanations for these anomalies include enhanced melting of a compositional heterogeneity within the mantle (Minshull et al., 1998) or the influence of a hot mantle plume (Schilling et al., 1985; Bourdon and Hemond, 2001), which could be located either beneath the Circe seamount (450 km east of the MAR) or below the surface expression of the melting anomaly. Both scenarios, a weak mantle plume on the one hand and a mantle heterogeneity on the other hand, are modeled numerically to test their feasibility in terms of mantle flow, melting and crustal thickness. Below I will present some key results.

3.4.2 Model description

A bathymetry map of the MAR between 2°–14°S is shown in Fig. 3.17a. The about 1200 km long ridge axis (red line) is displaced by many smaller and larger transform faults. The position of the ridge axis within each segment (including some degree of

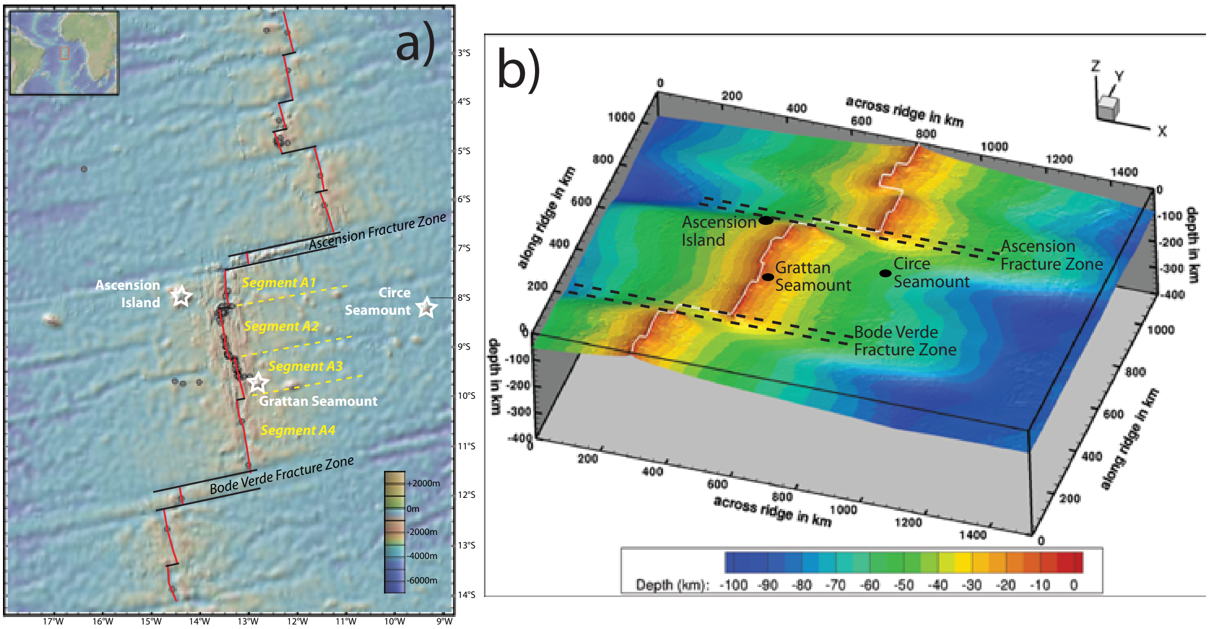


Figure 3.17: *(a) Bathymetry map of the complete study region [provided by C. W. Devey], in which the position of ridges (red), transform faults (black) and other geological features has been marked. (b) The same region in the 3-D numerical model (transformed into cartesian coordinates). The white line follows ridges and transform faults as prescribed in the model. The colored surface represents the 1200° C isotherm (depth is color-coded), after the model was run to steady state for a homogeneous mantle composition. The region between the two large transform faults is in the focus of the study (see also Fig. 3.18)*

curvature) and the transform faults has been extracted and transformed into a Cartesian coordinate system, within which the 3-D numerical model is defined. Fig. 3.17b shows the ridge axis and TFs (white line) in the 3-D numerical model. The colored surface shows the 1200°C isotherm, which is discussed below. The coordinate system has been rotated, so that plate motion can be imposed parallel to the x-coordinate, which considerably simplifies the boundary conditions on the y-perpendicular walls. Note that the spreading direction is oblique to the ridge axis in several regions.

The software *GiD* (http://gid.cimne.upc, Version 8, 2007) has been used to generate a high-resolution mesh near the ridge axis and transform faults, as well as in a ∼ 150 km wide and ∼ 100 km deep volume below these features. This ensures that the steep thermal gradients near the ridge axis as well as melting processes underneath are well resolved. The highest resolution (i.e. closest node spacing) is about 1 km near the points where ridge axis and TF intersect. With increasing distance from the ridge and towards greater depth, the node density is reduced to limit the computational work load. A maximum node spacing of about 200 km is reached near the lower right front of the domain (Fig. 3.17b). A total of about 4 million velocity unknowns is solved for on 8–16 CPUs.

The region between AFZ in the North and BVFZ in the South is in the focus of this numerical study, as the melting anomaly is located near Grattan seamount. The purpose of the large-scale model shown in Fig. 3.17b is to develop a regional mantle flow field, from which boundary conditions for on a local model, focusing on the region between the two large transform faults, can be extracted.

The position of the ridge axis is fixed during the experiments and a half-spreading rate of 17 km/Myr is prescribed on the top of the domain. Walls perpendicular to the y-direction are symmetry planes. Velocities at the x-perpendicular walls and the domain bottom are taken from 2-D calculations that ran to a steady state solution for the same parameter settings. The only exception is a 200 km wide region at the domain bottom that follows the ridge axis. Here, a flow-through boundary condition is used in the first time step, and the calculated velocities are from then on fixed for the rest of the experiment. This leads to a smoother variation of the bottom influx boundary condition compared to an interpolated 2-D result, which would cause discontinuities at the large transform faults.[8] During this first time step, all buoyancy sources below the ridge are neglected to avoid a ”too strong” influx, due to the missing viscous resistance below the domain.

Temperature boundary conditions are 0°C at the top, T_M =1315°C at the bottom and insulating everywhere else. Mantle temperature and composition are initialized using steady state 2-D calculations with the same physical and petrological parameter setting. This choice helps to reduce the computational time, since both, extreme melting events in the beginning of the calculation as well as an unnecessarily delayed onset of melting, are avoided. The model calculations cover 10–30 Myr of plate spreading and took between 3–10 days on 8 or 16 CPUs.

3.4.3 Selected results

The purpose of the regional experiment (Fig. 3.17b) is two-fold: First, it serves as a calibration of the various parameters that control the melt production at the ridge. The most important parameters are the mantle composition (mainly the solidus function that defines the productivity per unit of decompression) and the potential mantle temperature T_M. These parameters are adjusted such that a 6 km thick crust is produced in the center of the ridge segments. Second, the flow field calculated in the regional model is interpolated at the boundaries of the local model that focusses on the region between AFZ and BVFZ (Fig. 3.18). The reference model includes a single lithology (a peridotite depleted by 10%), a water content of 200 ppm and a 200-fold viscosity increase during complete dehydration are assumed. The melt weakening formulation of Hirth and Kohlstedt (2003) is used (Eq. (3.18c) on page 129).

[8] The 2-D velocities are interpolated with respect to the distance to the ridge axis. Where a 200 km long TF displaces the ridge axis, velocity boundary conditions at the bottom would be discontinuous across the TF.

Some characteristics can already be seen in the regional model. Transform faults appear as colder region in the shallow mantle as indicated by a deeper 1200°C isotherm. Mantle upwelling is significantly reduced here, so that conductive cooling near the TF penetrates deeper in to the mantle. Since the melt productivity depends on both decompression rate and thermal energy available for the solid-melt phase change, melting rates are reduced in the vicinity of TFs (see previous section on the effect of transform faults on mantle flow and melting).

The local model is bordered by the two large fracture zones (Fig. 3.18) and has a higher numerical resolution in this region compared to the regional model. The red isosurfaces enclose regions of partial melting: The outer transparent surface encloses the region where melt production is higher than 10% of the maximum melt production central underneath the ridge axis. The opaque red surface encloses regions, in which melt production exceeds 90% of the maximum. Melt production vanishes underneath both long TFs, but also smaller displacements of the ridge axis reduce the melt production (e.g. at $x = 530\,\mathrm{km}$ and $x = 320\,\mathrm{km}$). These variations in melt production are likely to be superimposed on any calculation in which the composition or temperature of the mantle is varied along the ridge axis.

The cross-sections in Fig. 3.18 show the viscosity structure in the uppermost mantle. Conductive cooling thickens the lithosphere as it ages, but if dehydration effects on viscosity are considered, a compositional lithosphere of about 40–50 km thickness forms next to the melting region. These strong diverging lithospheric ”walls” enhance the mantle upwelling below the ridge axis compared to model calculations, in which the dehydration effect on viscosity is neglected. Near the transform faults, conductive cooling leads to uniformly high viscosities in the upper 50 km of the mantle.

Two scenarios, a mantle heterogeneity on the one hand and a thermal (plume-like) anomaly on the other hand, are compared next. These model calculations are done in a simpler geometry with a straight ridge axis, so that symmetry across the vertical plane underneath the ridge can be assumed. This choice allows to more efficiently compare different mantle compositions and thermal structures in the first place.

The temporal evolution of a model calculation, in which a weak mantle plume with an excess temperature of about 60°C (at 100 km depth) rises 100 km off-axis, is shown in Fig. 3.19. The thermal anomaly ascends rather slowly, because the reductions in density and viscosity associated with the temperature increase are comparably small. Especially the lack of a significant thermal buoyancy results in a very slow upwelling within the background mantle flow field. The same mantle composition as above is used for the plume material.

The plume meets the compositional lithosphere about 80 km off-axis and is deflected

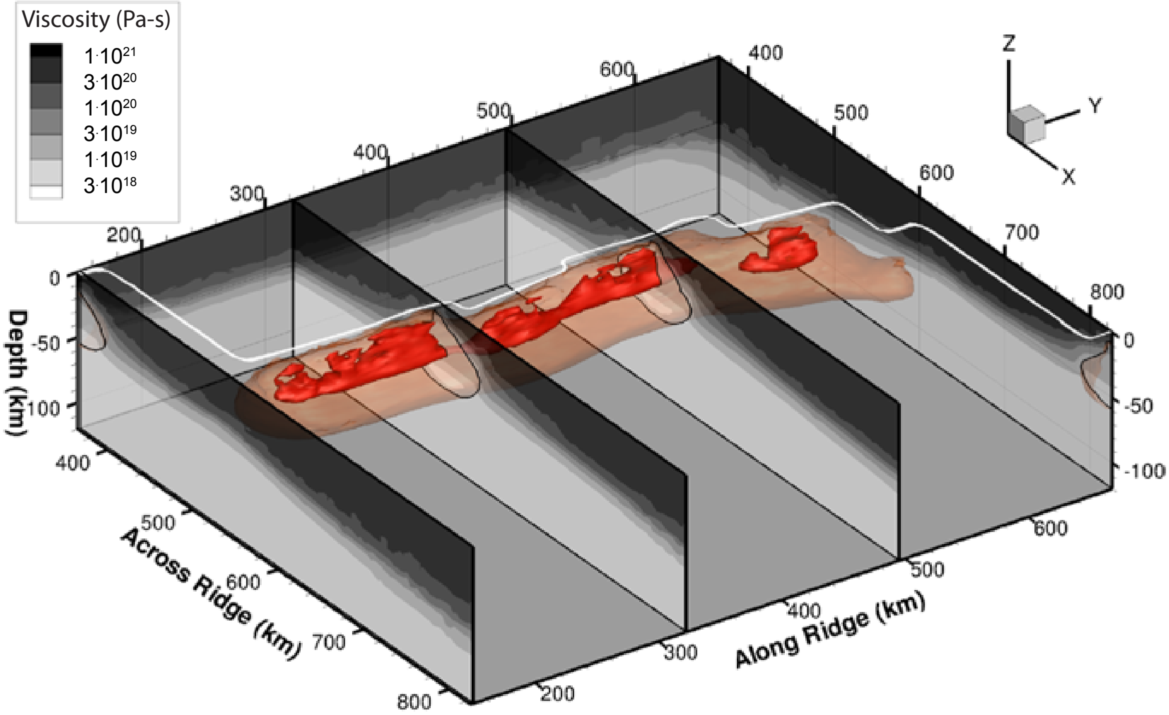


Figure 3.18: *A view of the region between the two large fracture zones (see also Fig. 3.17). Mantle rock viscosity is shown in greyscale on cross-sections, red iso-surfaces show regions with different melting rates (outside of transparent isotherm, melting rates are <10% of the maximum rate; the inner, opaque surface encloses the region, where melting rates are higher than 90%). The white line follows the ridge axis and transform faults, as prescribed in the numerical model. In this part of the MAR, the ridge axis is divided into four 2nd-order segments (e.g. Bruguier et al., 2003), which affects the vertical upwelling speed of the mantle and consequently the melting rates below each ridge axis displacement.*

and dragged away by the surrounding mantle flow Fig. 3.19c–d. The thermal anomaly associated with the plume results in an enhanced off-axis melt production so that the ridge melting zone is expanded towards the plume — partial melting starts about 30 km deeper and 80 km further away from the ridge axis. The crustal thickness, calculated by integrating the melt production vertically and advecting it with the plate, is shown in Fig. 3.19a–b. An elongated rise of 12 km thick oceanic crust forms perpendicular to the ridge axis.

An alternative model considers a compositional rather than a thermal anomaly in the mantle. A more fertile ”body” of cylindrical shape (no initial depletion, 40 km diameter and 80 km length, oriented parallel to the ridge axis) is assumed to be dragged into the ”normal” ridge melting zone. As opposed to the thermal anomaly discussed above, there is no differential speed between the compositional anomaly and surrounding mantle. The amount of excess melt production therefore depends strongly on the lateral distance of the body to the ridge axis, because the range of decompression melting is defined by the trajectories of the mantle flow. Only mantle that rises central underneath the ridge axis is decompressed over the full range and will experience the maximum degree of melting.

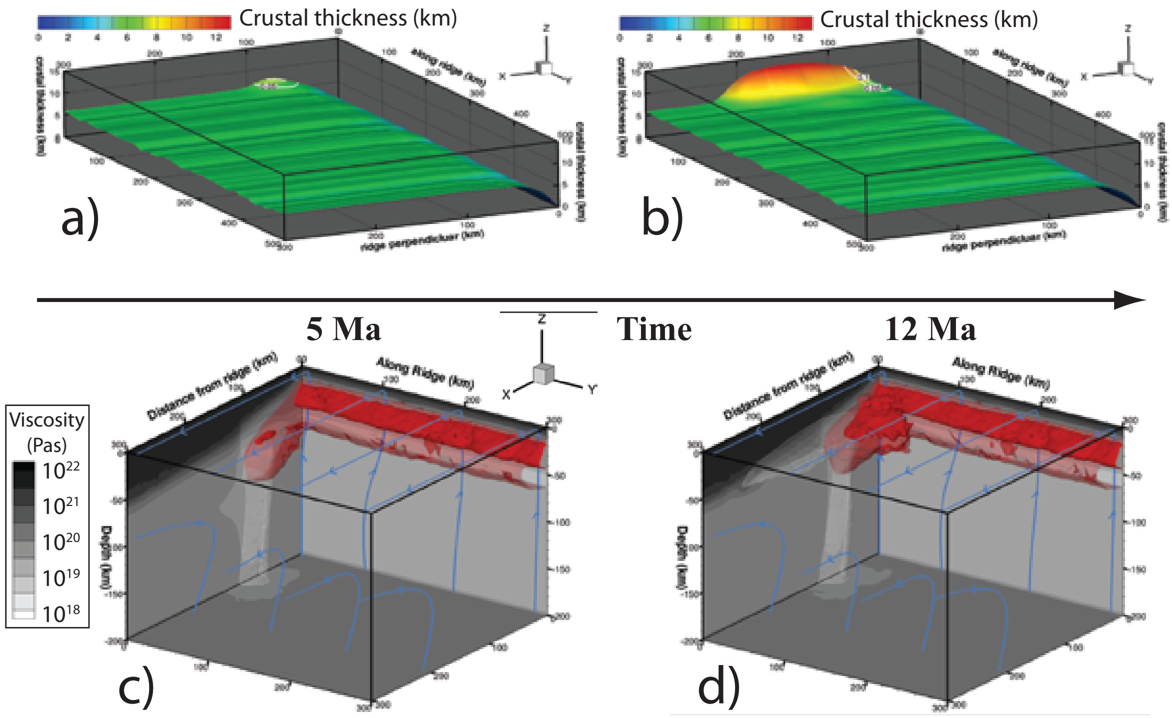


Figure 3.19: *Temporal evolution of the interaction of a mid-ocean ridge (parallel to y-direction at x=0) and a weak off-axis mantle plume (located at $x = 100\,km$ and $y = 0$). Symmetry is assumed in the ridge-parallel direction and only one-half of the plume is modeled. (a) and (b) show the crustal thickness predicted by the model calculation, which is shown underneath. The area where the plume's melts accrete to the crust is marked by white lines. Crustal thickness is calculated by vertically integrating all melting rates and adding them to the existing crust that is advected with the plate. (c)+(d): Mantle flow (blue streamlines), viscosity (grey scale on side walls and bottom), as well as the ridge melting zone (half-transparent surface at 10% $max(M_B)$, opaque surface at 90% $max(M_B)$). The plume is enclosed by a half-transparent white surface. While mantle flow is almost unaffected by the small density and viscosity anomalies associated with the plume, the ridge melting region is extended towards the plume and connects with its melting region.*

The importance of the lateral distance to the ridge axis is shown in Fig. 3.20. Crustal thickness calculations for four models are shown, in which only the distance of the mantle heterogeneity to the ridge axis is varied. While the heterogeneity rising 50 km off-axis causes a melting anomaly similar to the size of Grattan seamount, the same heterogeneity could lead to the formation of an ocean island (if placed at 25 km distance; Fig. 3.20d) or could be difficult to identify in terms of a crustal thickness anomaly (if located 100 km or more off-axis; Fig. 3.20a).

3.4.4 Discussion and outlook

A detailed numerical model of the MAR between 2°–14°S has been developed. It predicts lower temperatures in the uppermost mantle underneath the two long-offset transform faults. Very limited or no melt production is observed below these tectonic features. A model focussing on the region between the two TFs also shows the influence of comparably

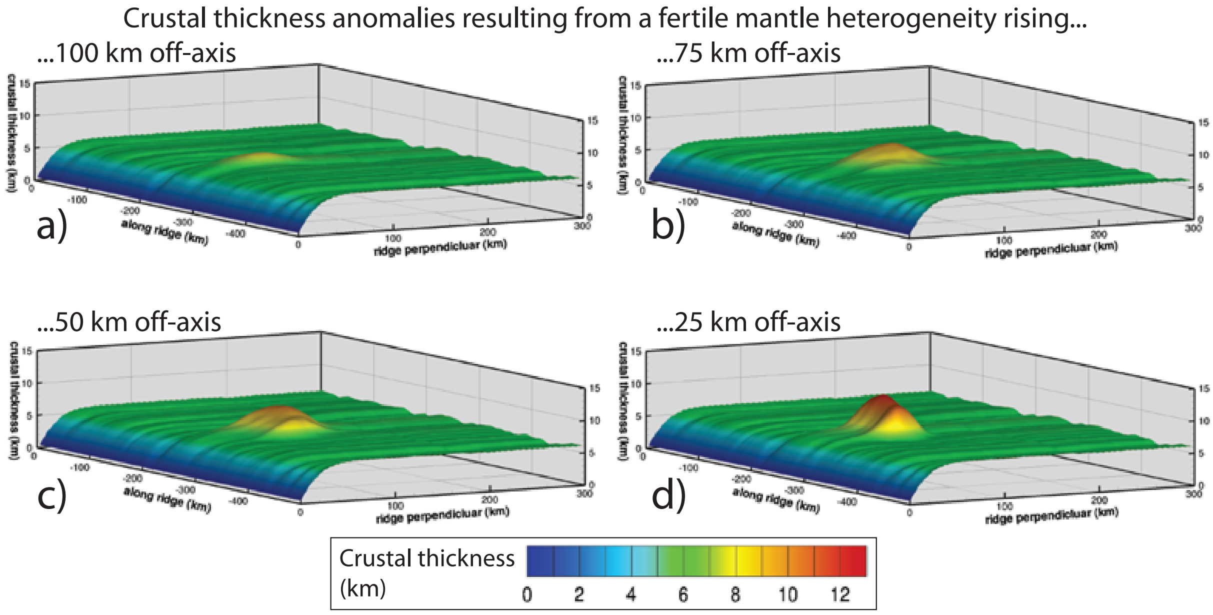


Figure 3.20: *Crustal thickness predicted by four numerical experiments, in which a fertile mantle heterogeneity (40 km diameter, 80 km length, oriented parallel to the ridge axis) is situated in the mantle at different distances to the ridge axis. (a) Only ∼900 m of additional oceanic crust is predicted, if the heterogeneity is located 100 km off-axis. At this distance, the decompression range is limited by the mantle flow trajectories. (b) 75 km off-axis, ∼2 km additional crust. (c) 50 km off-axis, ∼3 km additional crust. (d) 25 km off-axis, 7 km of additional crust are produced. The size of Grattan seamount is within the range of the experiments shown in (b) and (c).*

small ridge axis displacements on the melt production (Fig. 3.18).

Two scenarios, a weak mantle plume and a mantle heterogeneity, that potentially can cause melting anomalies have been compared. The position of the weak plume (Fig. 3.19) is chosen such that an asymmetric crustal thickness anomaly could evolve. A plume rising closer to the ridge would be pulled into the central melting region and the enhanced melt production should lead to crustal thickness anomalies on both sides of the ridge axis. This is not observed as a counterpart to Grattan seamount on the western side of the MAR is missing. If the plume were to rise at greater distance, its influence on the melt production at the ridge would vanish quickly — unless its flow were to somehow migrate towards the ridge melting region. A stronger plume could buoyantly migrate towards the ridge axis by following the base of a thermal lithosphere in the up-slope direction. However, a strong plume would also increase the melt production above its tail, which is likely to result in a more pronounced crustal thickness anomaly at the ”hot spot” than is seen at the location of Circe seamount. The formation of an island chain or a hot spot swell, both typical for hot spot volcanism, should also emerge, as opposed to the observations

A more fertile mantle heterogeneity (Fig. 3.20) is probably the best explanation for this particular melting anomaly at the MAR. The experiments show the large variations in crustal thickness that may result from a body of more fertile peridotite. The distance between the mantle anomaly and the ridge axis has a significant impact on whether or not a crustal thickness variation results. This is a consequence of the mantle flow at the ridge axis, which has its strongest vertical component central underneath the ridge axis. Since the mantle heterogeneity has a similar density as surrounding rocks (i.e. no thermal buoyancy), it will be advected passively with the mantle flow and cannot actively rise towards the ridge axis. The surface expression of melting a mantle heterogeneity is directly linked to the size of the heterogeneity itself. In the examples presented here, the diameter of the heterogeneity and the resulting seamount/ocean island is very similar.

Geochemical data (Almeev et al., 2007) indicate a change in basalt's water content as the crustal thickness increases towards Grattan seamount. This observation fits well with the fertile mantle heterogeneity, which is likely to have seen fewer previous melting events than the ambient depleted mantle peridotite. If the heterogeneity is large enough, water contents in its center may not have equilibrated with ambient, more depleted rocks, which should lead to ”wetter” melts.

An alternative to the fertile peridotite anomaly could be a depleted peridotite with a larger fraction of enriched pyroxenite veins. In this case, however, the water fractions in veins and matrix may have equilibrated over time so that the pooled melts would not show a strong wet signature. In addition, no clear pyroxenite signature is seen in geochemical data of the basalts dredged along the MAR in this region.

Chapter 4

2D and 3D numerical models on compositionally buoyant diapirs in the mantle wedge[1]

4.1 Introduction

Subduction of oceanic lithosphere is associated with melt generation in the mantle wedge between the descending slab and the overriding plate, leading to the formation of volcanic arcs. Two mechanisms are potentially responsible for the melt generation: a decrease of the mantle solidus temperature due to the presence of aqueous fluids rising from the dehydrating slab (a process commonly referred to as flux-melting) and adiabatic decompression melting of mantle rocks, which requires an upward velocity component in the solid-state mantle flow. The review by Pearce and Peate (1995) summarizes that melting underneath a volcanic arc results from a combination of volatile-addition and mantle decompression. There is a clear relationship between the amount of water added and the degree of volatile-induced melting (Stolper and Newman, 1994) on the one hand, and an inverse correlation between lithospheric thickness (acting as the upper barrier to mantle upwelling) and the amount of decompression melting on the other hand (Plank and Langmuir, 1988; Pearce and Peate, 1995). The importance of decompression melting for subduction zone volcanism is emphasized by the similarities between melts generated at subduction zones and mid-ocean ridges. These melts overlap in the major element composition and have similar extents of partial melting (Plank and Langmuir, 1988). Mantle peridotite has been inferred to be their common source.

[1] The following chapter has been prepared for publication and the personal pronoun "we" is used throughout this chapter. The authors are: Jörg Hasenclever, Jason Phipps Morgan, Lars H. Rüpke, and Matthias Hort. The numerical model has been developed by myself as described in chapter 2 of this thesis. I have conducted all model calculations, as well as summarized and visualized the results. The interpretation was done by all authors.

Large degrees of decompression melting (10-20%) require a significant upward motion of the mantle, at least below regions of volcanic activity. Early numerical models (e.g. Davies and Stevenson, 1992) could not explain this unless they allow for regions with positive buoyancy in the mantle wedge. Without these density anomalies the predicted flow field is similar to the analytical solution for isoviscous corner flow (Batchelor, 1967), which has been frequently used by studies focusing on the thermal evolution of subduction zones (e.g. Peacock, 1991; Peacock et al., 1994) or the transport of water (e.g. Iwamori, 1998). A diagonal upward flow towards the tip of the mantle wedge is predicted by several 2D numerical models (Eberle et al., 2002; Kelemen et al., 2003; van Keken, 2003) that include a viscously deforming overriding plate. More complex 2D numerical models including different rheological units such as oceanic sediments, basaltic crust and dry/wet mantle rocks, phase transitions and partial melting (e.g. Gerya and Yuen, 2003; Gorczyk et al., 2006) show rotating flow fields and plume-like wet diapirs that are difficult to interpret in the context of a three-dimensional subduction zone.

Since it has been first suggested that mantle diapirism could underlie most arc volcanoes (Marsh and Carmichael, 1974) more studies have provided evidence that three-dimensional features are present inside the mantle wedge. Along-trench variations in seismic attenuation (Nakajima and Hasegawa, 2003) and seismic velocities (Zhao et al., 2009) at the Honshu subduction zone have been found to correlate with clustering of volcanoes (Tamura et al., 2002). Recently, detailed 3D numerical models for subduction zones have become possible as parallel computers have increased in memory and speed. Honda and coworkers (e.g. Honda and Yoshida, 2005; Honda et al., 2007) suggest small-scale convection is the cause for these patterns and present 2D and 3D numerical models that have in common that they include a fixed low viscosity region within the wedge. They observe thermal instabilities resulting from conductive cooling form the top that form roll-like instabilities (Richter rolls) within the low-viscosity mantle wedge, with axes parallel to the shallow mantle flow towards the trench. A 3D model by Zhu et al. (2009), which is similar to the 2D version used by Gorczyk et al. (2006), predicts diapiric upwellings, but is very complex as it includes several mechanisms that strongly feedback into each other (e.g. different rheological units, a continuously changing subduction geometry as the trench retreats, water migration). A simplified model (Honda et al., 2010) predicts small-scale convection patterns if there is a small amount of chemical buoyancy, dispersed as tracer-particles, in the mantle wedge and more 2D-like flow patterns if there is a lot of chemical buoyancy.

In this study we present simple 2D and 3D numerical models of solid-state mantle flow at subduction zones. The subduction zone geometry, defined by slab and overriding plate, is simplified and fixed during all runs. The slab is kinematically prescribed to have a constant angle of subduction. The upper domain boundary — the base of the overlying lithosphere — is taken to be flat. All model calculations are isoviscous and we assume that dehydration of the subducting slab yields the formation of a hydrated and buoyant layer on top of the slab that has the potential to become buoyantly unstable and generate Rayleigh-Taylor-like instabilities.

We systematically explore different flow regimes in more than one hundred 2D simulations. From these experiments, we derive phase diagrams that show the behavior of the system as a function of four important parameters: subduction angle, subduction rate, water diffusivity, and mantle viscosity. For selected parameter combinations we conduct numerical simulations using a 3D extension of the 2D model and compare to which extent and under which conditions 2D models can be used to estimate the 3D behavior of this specific subduction zone setting.

4.2 Governing equations and numerical model

4.2.1 Governing equations and their numerical solution

In order to examine solid-state mantle flow and the advection-diffusion of volatiles (in our case water) in the mantle wedge we formulate numerical models in two- and three-dimensional Cartesian coordinates. We describe the mantle as an incompressible, isoviscous fluid with infinite Prandtl number and apply the Boussinesq approximation, that density differences only affect the buoyancy force term. Using the index notation and Einstein summation convention, the governing equations can be written as

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{4.1}$$

$$\frac{\partial p}{\partial x_i} = \frac{\partial \tau_{ij}}{\partial x_j} - \rho g\, e_z \tag{4.2}$$

$$\tau_{ij} = \eta \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{4.3}$$

with (4.1) satisfying conservation of mass by imposing incompressibility, (4.2) describing the force balance to ensure conservation of momentum, and (4.3) being the constitutive law. τ_{ij} denotes stress tensor, u velocity, x physical coordinate, p pressure, g gravitational acceleration, ρ density and e_z the unit vector in the vertical direction. For a complete list of variables, their meaning, units, and values the reader is referred to Tab. 4.1. In our model buoyantly driven flow is solely caused by density variations arising from compositional changes (i.e. water content). The bulk density of mantle rock containing a volume fraction ξ of volatiles with density ρ_C is therefore given by

$$\rho(C) = (1-\xi)\rho_M + \xi\rho_C \tag{4.4}$$

with ρ_M being the density of dry mantle rocks. We choose to not solve for the thermal evolution within the mantle wedge because this allows for a better identification of the effects of compositional buoyancy on mantle flow.

We decided to model the migration of water relative to the mantle as a diffusion process in which we can vary a single easy-to-interpret diffusivity parameter. Temporal changes in the water concentration field C are calculated using

$$\frac{\partial C}{\partial t} = \gamma \left(\frac{\partial^2 C}{\partial x_i^2} \right) - u_i \frac{\partial C}{\partial x_j} \tag{4.5}$$

with u being the flow field of the mantle and γ the effective migration diffusivity assumed for water. Diffusivity in this context can be viewed as the mobility of volatiles in the mantle, allowing the fluid to migrate into all spatial directions without a preferred orientation.

Table 4.1: *List of variables used in this study.*

variable	meaning	value, units
u	velocity	$\mathrm{km\,Myr^{-1}}$
p	pressure	Pa
τ	stress tensor	Pa
η	dynamic viscosity	Pa·s
g	gravitational acceleration	$\mathrm{m\,s^{-2}}$
e_z	unit vector in vertical direction	1
ρ	density	$\mathrm{kg\,m^3}$
ρ_M	mantle density	$3300\,\mathrm{kg\,m^3}$
ρ_C	water density	$1000\,\mathrm{kg\,m^3}$
C	water field	1
ξ	volume fraction of water in mantle	1
γ	water diffusivity	$\mathrm{m^2\,s^{-1}}$

For this study we favor the diffusion formulation over a vertical Darcy flow formulation because the latter strongly depends on mantle rock permeability (i.e. grain size) which is a poorly constrained parameter. We found that a purely vertical water migration forces the instabilities to solely develop above the slab dehydration region where a wet region would emerge. Furthermore, the diffusion formulation has the great advantage of conserving the amount of water during the migration as the equation is similar to the energy conservation equation in thermal diffusion problems. It also allows us to inject water into the domain at a defined rate using a flux boundary condition.

4.2.2 Numerical formulation

The above equations are solved numerically using newly developed 2D and 3D codes written in MATLAB. The pressure-velocity equations (4.1)–(4.3) are discretized using the finite element method with triangular (in 2D) and tetrahedral (in 3D) P2P1 Taylor-Hood elements with quadratic velocity and linear pressure interpolation functions. The equations for velocity and pressure are decoupled using a Schur complement formulation (Maday and Patera, 1989) resulting in an outer loop calculating the pressure solution within which the velocity solution is updated. Both, pressure and velocity part are solved using conjugate gradient algorithms. The pressure part is preconditioned by Jacobi iterations using the inverse-viscosity-scaled mass matrix, while the velocity part is preconditioned by a geometrical multigrid algorithm (single V-cycle) with a direct solver (Cholesky factorization) on the coarsest level. The large number of unknowns in the 3D simulations requires a parallelization of the numerical model which is done using MATLAB's *Parallel Computing Toolbox*. We perform a non-overlapping domain decomposition without creating so-called halos or ghost nodes around the subdomains. This approach has the advantage that no iterations between subdomain solutions are necessary to derive a global solution for the entire domain. Instead intermediate results during the pressure-velocity iterations are communicated and summed at nodes shared by subdomains, which leads directly to the global solution.

The advection-diffusion equation (4.5) is solved using operator splitting: The diffusion part is discretized using a finite element formulation based on the same Taylor-Hood elements that are used in the viscous flow problem. A Crank-Nicholson time approximation scheme is used for its improved stability and accuracy. The resulting matrix equation is solved by a conjugate gradient algorithm, which allows us to apply the same parallelization method as described above for the viscous flow part. Diagonal scaling proved to be sufficient for preconditioning the diffusion problem. Volatile advection is done by a Semi-Lagrange advection scheme with second-order accurate Predictor-Corrector back tracking in combination with a cubic smooth interpolation on unstructured 2D and 3D meshes (Shi and Phipps Morgan, 2010) that significantly reduces artificial interpolation-related numerical diffusion in the upwind direction.

4.2.3 Boundary conditions and initialization

The 2D and 3D models presented in this study are computed for the region between subducting slab and base of the over-riding plate (see Fig. 4.1). The slab is implemented as a kinematic boundary condition with the slab velocity prescribed along the inclined base of the computational region. A no-slip boundary condition is applied along the top of the domain corresponding to the idealized flat base of the overriding lithosphere. Velocities on the right-side boundary, where material flows both into and out of the computational region, are calculated in the first time step where all mantle material is water-free and no buoyancy forces are present. To do so, we use a boundary condition that allows horizontal flow through the right wall in the first time step. From then on, the horizontal velocities are fixed to these calculated values (the boundary velocities are shown in Fig. 4.1c). The reason for not allowing flow-through during the entire simulation is that buoyant upwellings in the domain would tend to ”escape” through the right boundary because viscous stresses vanish on flow-through boundaries.

Box dimensions depend on the subduction angle under investigation but reach down to about 300 km in all runs. For subduction angles of 20°, 30°, and 40°, the lateral extension in the direction of plate motion is 700 km, 500 km, and 350 km, resp. The along-trench extension (y-direction) in the 3D models is 300 km. We use periodic boundary conditions on the walls perpendicular to the trench. This is achieved by modifying the element connectivity matrix so that each corresponding pair of nodes at $y = 0$ and $y = y_{max}$ share the same node number in the connectivity matrix. Thus, nodes on the wall at $y = 0$ become connected to nodes inside the domain near the opposing wall, and vice versa, resulting in a periodic behavior of the system. A requirement for this method is that all nodes on the $y = 0$ plane have a counterpart with same x- and z-coordinates on the $y = y_{max}$ plane so that these nodes can be paired. A simple way to generate a mesh having this property is to mirror a FE mesh in the direction of periodicity, renumber the global nodes to get a unique node list and reorder the local node numbering for all elements in the mirrored part. The last step is necessary because the mirrored elements will have a different numbering scheme for their local nodes, which usually causes trouble in several code parts (for instance the element assembly part).

We use periodic boundary conditions because we think they work somewhat better to avoid the problem of the domain-size strongly affecting the spacing between diapirs. We found that the more frequently used free-slip boundary conditions on front and back walls affect the temporal evolution and along-trench spacing of the diapirs, especially if only a few diapirs with a large spacing evolve. At least 800 km trench length had to be modeled to make sure that instabilities in the central part of the domain were relatively unaffected by the symmetry planes. This is believed to be controlled by the preferred (i.e. fastest growing) wavelength, which is characteristic for Rayleigh-Taylor instabilities. Symmetry planes bounding a chain of instabilities must either intersect halfway between two plumes or through the center of a plume (otherwise there would be no symmetry), which forces the diapirs next to them to shift accordingly. If this shift is large compared to the characteristic spacing between the plumes, their number and ascent time can change. Runs with periodic boundary conditions also suffer from the problem of forcing the sum of all plume spacings to match the domain length but do not influence the positions of the first and last plume.

In addition to a sufficiently large domain the numerical resolution is key for high-quality results. We use unstructured meshes with maximum resolution (i.e. node spacing) of about 1.2 km within the region of 20–30 km above the slab to properly resolve the short-wavelength instabilities that evolve in the water-rich boundary layer. The node spacing is nowhere larger than 2.5 km with the only exception being the upper right region in the 3D models, which is tested to have no influence on the evolving instabilities. Of course one can do better than 1.2 km resolution in 2D models, especially if they run on a parallel cluster with many tens of GB of distributed memory, but in order to accurately compare 2D and 3D experiments the same resolution was chosen for both sets of experiments. We checked that the resolution used in all models is sufficient to resolve the evolving structures. Misleading results of under-resolved 3D experiments will be discussed below (see Fig. 4.10). The number of velocity unknowns in the 2D models is about 90,000 (45,000 nodes, each 2 degrees of freedom) and 12,000,000 in the 3D models (4,000,000 nodes, each 3 degrees of freedom). In addition, each node on an element's vertex is associated with a pressure unknown. Each 2D calculation was accomplished in few hours, while the 3D calculations ran for 1–2 weeks on 32 CPUs.

Slab dehydration is implemented as a water influx boundary condition at the top of the slab (i.e. at the domain bottom boundary) in all numerical runs. We use an estimate for the average water content in oceanic lithosphere below each square meter seafloor of $4.6 \cdot 10^5\,\mathrm{kg/m^2}$ (Rüpke et al., 2004), and assume that 80% of this water is released between 80–180 km depth. In doing so, we fix the amount of water per unit length of the slab so that the actual water influx per unit area into the domain depends on both subduction rate and subduction angle. For the same subduction angle, faster slabs release more water per time compared to a slow subduction. For the same subduction rate, slabs descending at steeper angle have a stronger water release per unit length because the slab section between 80–180 km depth is actually shorter than for a shallower subduction angle (in other words: the dehydration rate scales with the vertical velocity component). Prescribing the amount of water release per unit slab descent is more realistic than using the same influx per unit area for all angles and subduction rates. There is no along-arc variation in water influx in the 3D experiments. In all experiments the influx is scaled using a Gaussian weighting function having a peak value at 130 km depth and fading towards 80 and 180 km depth. We decided to have this smoothly varying water-release function so that diapirs would be less controlled by the upper edge of the water-release region. In several simulations without the smoothing function but with a uniform influx between 80 and 180 km depth we observed gravitational instabilities at the incoming 'edge' of the water-release region at 80 km depth.

In the 3D calculations the water content is set to zero for all material advected deeper than 300 km. This intervention helps to avoid diapir formation near the bottom right edge of the computational domain, where it cannot naturally evolve. This allows us to run the simulations for a longer time, and it has been checked to have no influence on diapirs at shallower depth. It is also consistent with the fact that at higher pressures mantle viscosity increases and the formation of compositional diapirs becomes less likely.

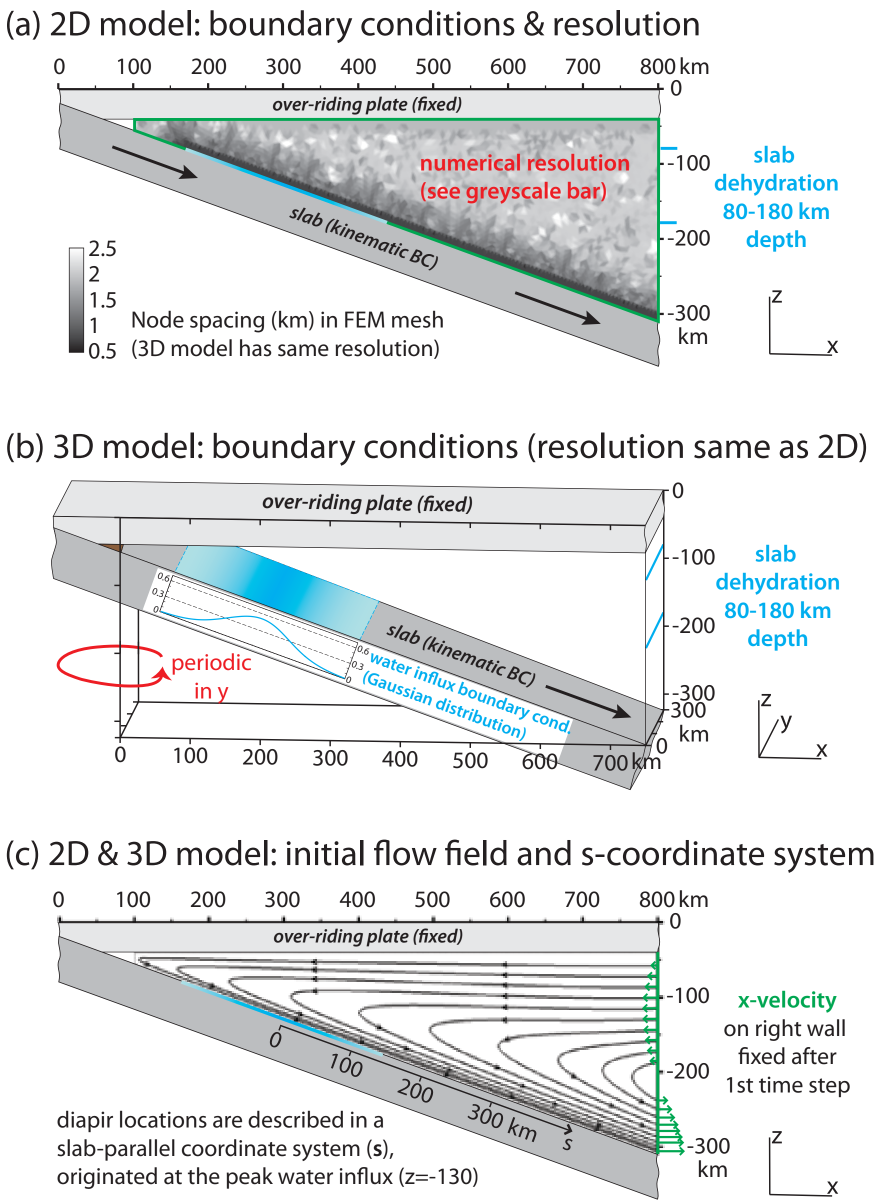

(a) 2D model: boundary conditions & resolution
0
100
200
300
400
500
600
700
800 km
over-riding plate (fixed)
numerical resolution
(see greyscale bar)
slab (kinematic BC)
2.5
2
1.5
1
0.5
Node spacing (km) in FEM mesh
(3D model has same resolution)
0
-100
-200
-300
km
slab
dehydration
80-180 km
depth
z
x
(b) 3D model: boundary conditions (resolution same as 2D)
over-riding plate (fixed)
slab (kinematic BC)
water influx boundary cond.
(Gaussian distribution)
periodic
in y
0.6
0.3
0
-100
-200
-300
300
km
700 km
z
y
x
(c) 2D & 3D model: initial flow field and s-coordinate system
over-riding plate (fixed)
x-velocity
on right wall
fixed after
1st time step
0
100
200
300 km
s
diapir locations are described in a
slab-parallel coordinate system (s),
originated at the peak water influx (z=-130)

Figure 4.1: *Numerical domain and boundary conditions (BCs) for the 2D and 3D models with 20° subduction angle; 30° and 40° models have similar BCs. (a) 2D models: over-riding plate and subducting slab are not part of the model; computational domain is enclosed by the green line. BCs: no-slip along top, subduction speed along bottom, free slip along the short left wall. Horizontal velocities at right boundary are calculated in 1st time step and fixed to these values from then on (see text and (c)). FE node spacing is shown in greyscale — it is similar in all 2D and 3D models. Slab dehydration through water influx BC between 80 and 180 km depth (blue line), smoothed using a Gaussian distribution. (b) 3D models: BCs on x- and z-normal planes equivalent to those in 2D models, periodic boundary conditions are used in the y-direction. Water influx same as in 2D models (blue region on top of slab, smoothed using the Gaussian function indicated by the blue graph). For details see text. (c) Flow lines calculated in the first time step. Green arrows indicate the horizontal velocities that are fixed from the 2nd time step on.*

4.3 Results

4.3.1 2D numerical experiments

All numerical experiments start with a water-free mantle wedge and all material having the reference density of 3300 kg/m^3. After subduction initiation water enters the domain through the slab surface between 80 and 180 km depth using a flux boundary condition. The temporal evolution of a 2D run is shown in Fig. 4.2. During the first few million years a water-rich boundary layer develops on top of the slab that is dragged downwards by the slab. The layer's thickness grows with time, at a rate mainly controlled by the water migration speed (diffusivity), until a critical thickness is reached and a Rayleigh-Taylor-like instability develops (see Fig. 4.2, 5 Myr, x=380 km). Since the model domain is two-dimensional the emerging upwelling has to completely intersect the corner flow in order to ascend towards the upper plate, which is indicated by the vanishing horizontal velocities above the instability. Within about 2 Myr the instability becomes a sheet-like diapir rising 130 km upwards to the base of the lithosphere, where the mushroom-shaped plume spreads out laterally. Upwelling speed is highest (about 150 km/Myr) by the time the diapir's head is about half-way up and reduces to 100 km/Myr at around 100 km depth after the head impinged on the lithosphere. The diapir formation has two major consequences: First, the mantle wedge corner flow is interrupted leading to an isolated convection cell between diapir and tip of the mantle wedge. Second, the removal of buoyant material from the boundary layer introduces a bulge-shaped disturbance in the layer (Fig. 4.2, 6.7 Myr, at x=420–450 km) triggering the formation of a second instability at x=460 km (Fig. 4.2, 7.9 Myr). The second instability rises as a separate sheet at about 100 km distance to the established one, which remains unaffected as it is continuously fed by new water-rich material dragged towards its root. Again the second diapir disturbs the boundary layer in the downstream direction and triggers a third instability at a distance similar to the spacing between the two existing ones (Fig. 4.2, 10.2 Myr).

A second experiment with a steeper subduction angle and a slower subduction rate shows a slightly different behavior (Fig. 4.3). Again a water-rich layer forms on top of the slab, becomes unstable after a few million years and leads to the formation of a stable sheet-like diapir at x=240 km. The secondary instabilities triggered by this upwelling, however, appear much closer to the diapir itself (see the small bulge next to the root of the upwelling in Fig. 4.3, 6.7 Myr) so that they feed into the existing diapir channel rather than forming a separate instability. The result is a single, pulsating diapir. This pattern repeats for several million years with a recurrence period of about 2.2 Myr after the first pulse. Note that there is a larger time span between the formation of the diapir at about 5 Myr and the 1st pulse at about 8 Myr.

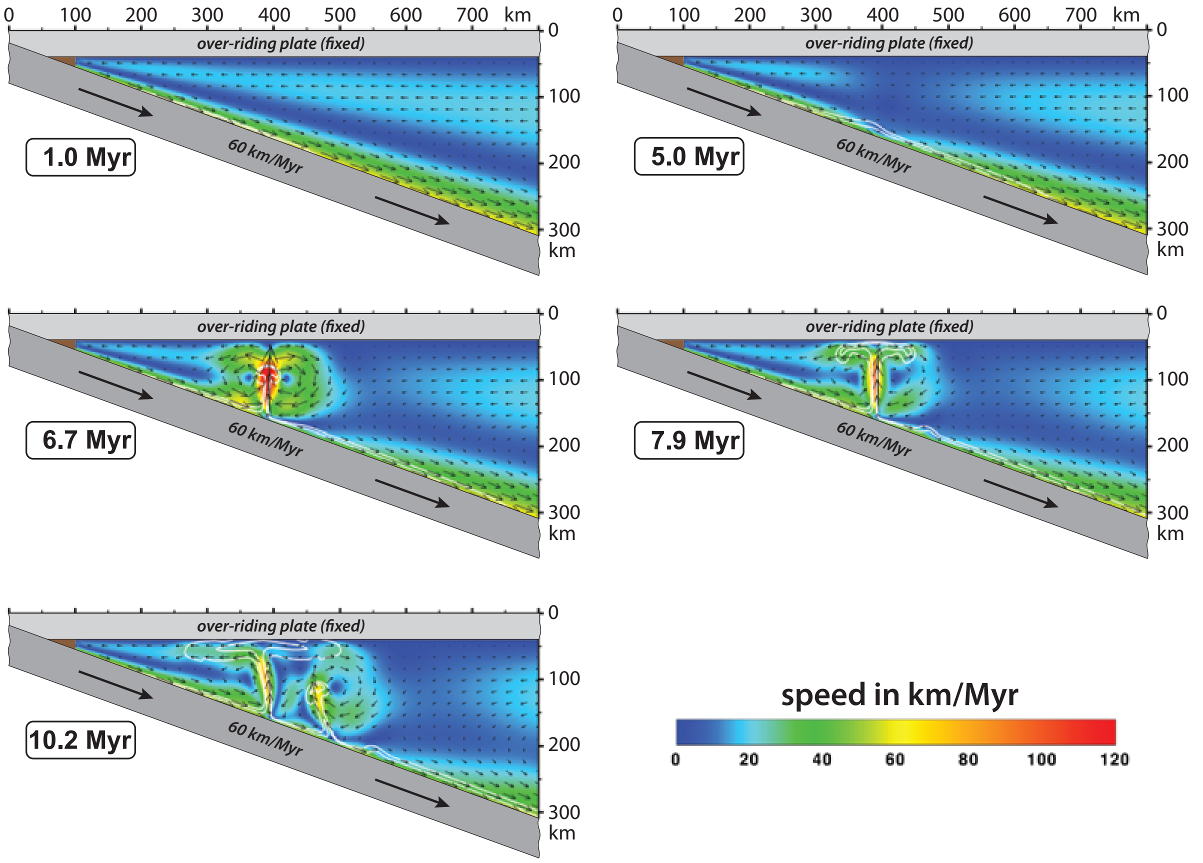


Figure 4.2: *Temporal evolution of a 2D numerical experiment (angle=20°, speed=60 km/Myr, viscosity=*10^{19} Pa·s*, water diffusivity=*10^{-7} *m*^{2}*/sec). Colors show magnitude of velocity, white isolines show density anomalies of 0.1% and 1%. This run is representative for the "multiple instabilities" (MI) regime (see text and Fig. 4.4–4.5).*

The general pattern observed in many 2D experiments includes all or some of the following stages: 1) formation and growth of a water-rich boundary layer on top of the slab; 2) formation of a single Rayleigh-Taylor-like instability after a critical boundary layer thickness has been reached; 3) evolution and rise of a (sheet-like) diapir; 4) more rapid formation of a secondary instability caused by the disturbances introduced by the previous instability.

We conducted a total of 135 2D runs to systematically explore the parameter range spanned by subduction rate ($30 - 120\,km/Myr$), subduction angle (20°, 30°, 40°), mantle viscosity (10^{19}, $3 \cdot 10^{19}$, 10^{20} Pa·s), and water diffusivity that parameterizes the speed of water migration with respect to the moving mantle (10^{-6}, 10^{-7}, $10^{-8}\,\mathrm{m^2s^{-1}}$). Depending on their behavior we classify the 2D runs into five different regimes allowing us to plot them in phase diagrams as shown in Fig 4.5. Snapshots illustrating each phase are given in Fig. 4.4. The different regimes are:

- SI (shallow instability, Fig. 4.4a): For steep subduction angles, low subduction rates

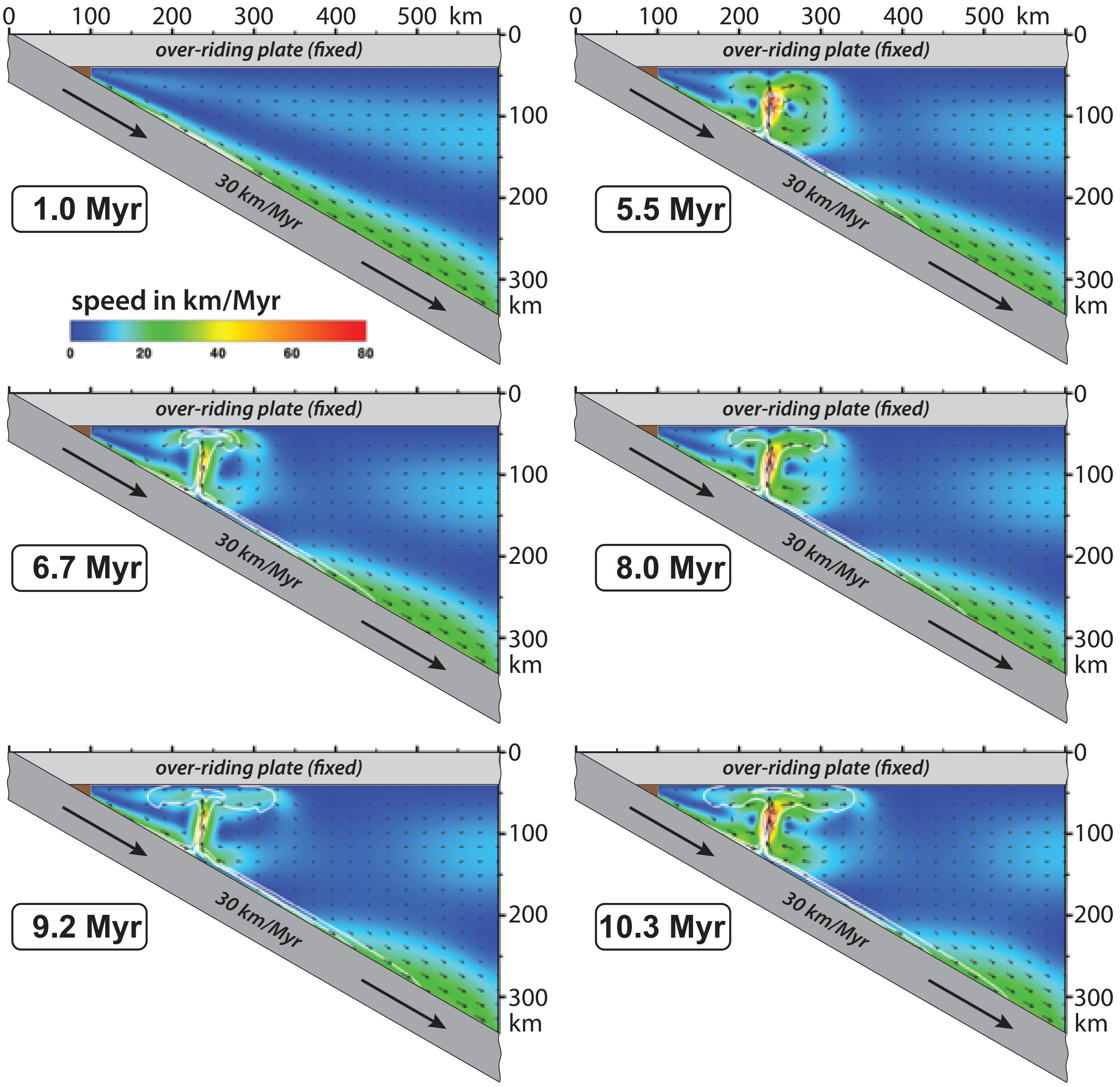


Figure 4.3: *Temporal evolution of a 2D numerical experiment (angle=30°, speed=30 km/Myr, viscosity=*10^{19} Pa·s*, water diffusivity=*10^{-7} *m*^{2}*/sec). Colors show magnitude of velocity, white isolines show density anomalies of 0.1% and 1%. The run is representative for the "pulsating instability" (PI) regime (see text and Fig. 4.4-4.5).*

and high water mobility we find single shallow instabilities with constantly high buoyancy flux. These diapirs are located within the region of dehydration (around 130 km depth where the water influx is highest) and transport a significant portion of the water-rich material upwards leaving insufficient buoyant material for secondary instabilities at greater depth.

- PI (pulsating instability, Fig. 4.4b): The formation of the first diapir causes secondary instabilities so close to its root (within about <30 km) that they feed into the existing channel. The pulses consist of rapidly upwelling, water-rich material

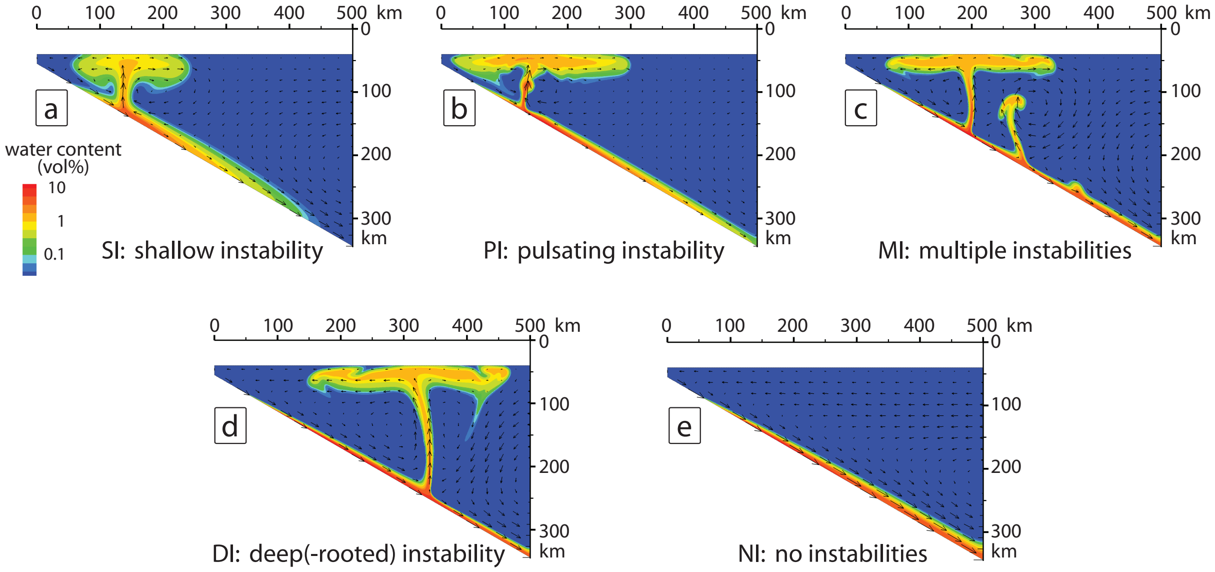


Figure 4.4: *Representative snapshot for each of the five identified flow regimes in the 2D numerical experiments. For a better comparison only runs with a 30° subduction angle are shown. All regimes have been also identified in the 20° and 40° experiments (see Fig. 4.5). Colors are water concentration; black arrows show the flow field.*

and show frequencies between 1.5-4.5 Myr. Steeper subduction angles support the formation of pulsing diapirs as the instabilities begin to migrate up-slope towards the existing diapir while they grow.

- MI (multiple instabilities, Fig. 4.4c): The first diapir emerging from the buoyant layer triggers secondary instabilities downstream leading to a series of upwellings with similar spacing. The distance between the first diapir and secondary instabilities varies between few tens and few hundreds of kilometers, depending on water diffusivity, subduction rate, and mantle viscosity. The MI regime is bounded by the single pulsating diapir regime (proximal end-member) and the single deep-seated diapir regime (distal end-member).

- DI (deep-rooted instability, Fig. 4.4d): The slow growth of the boundary layer leads to a single instability at greater depth (beyond the deep dehydration limit). In general these single diapirs are very stable as they are continuously fed by the water-rich material dragged downwards by the slab. No subsequent instabilities are triggered, either because the single diapir consumes most of the water leaving an insufficient buoyancy source for subsequent instabilities or because the formation of the second diapir would be outside of the numerical domain.

- NI (no instabilities, Fig. 4.4e): For high viscosities, fast subduction rates and/or low water diffusivities the buoyant boundary layer grows at a rate too slow to reach a critical thickness within the numerical domain. A steady state situation is reached

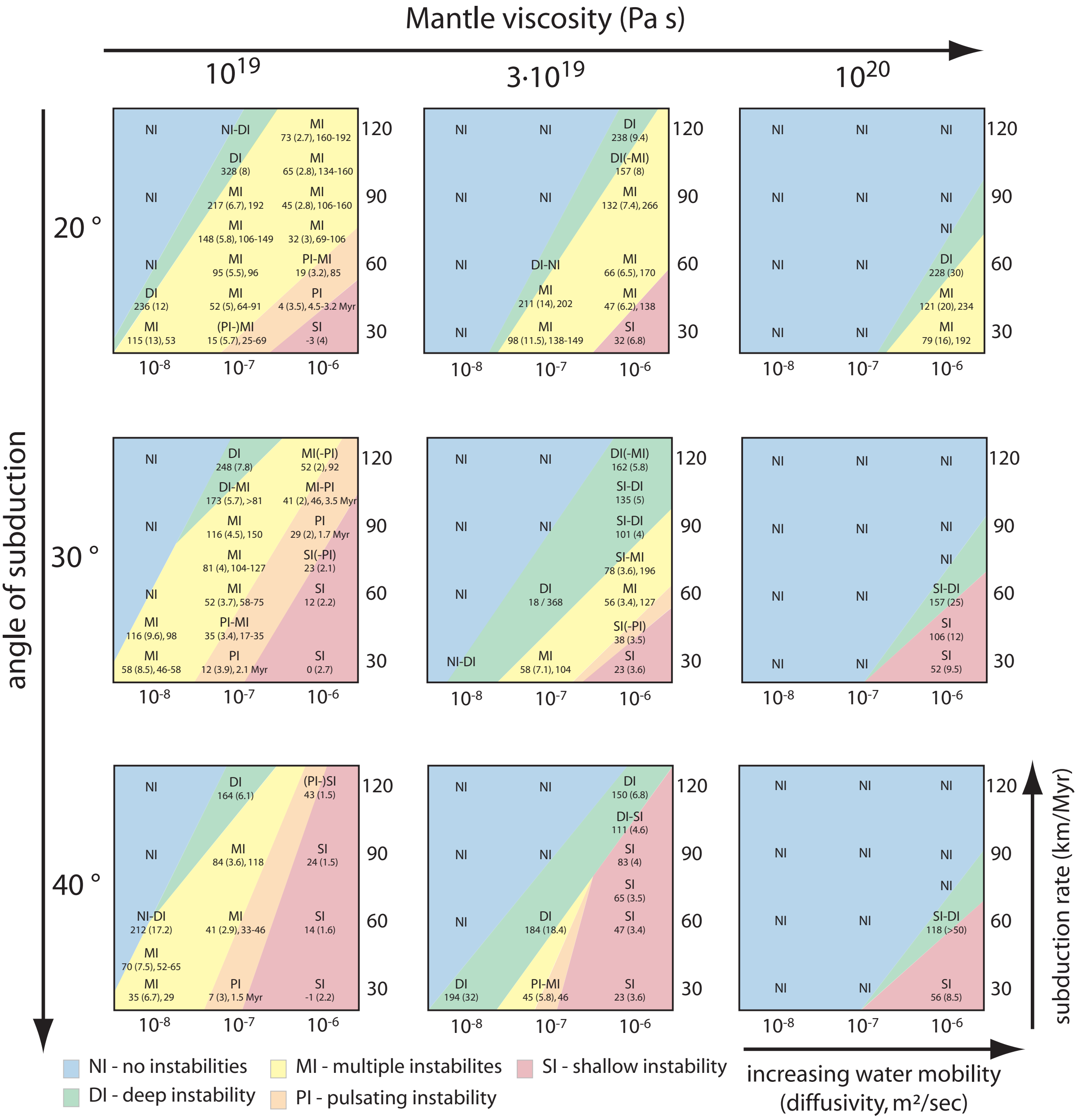


Figure 4.5: *All 2D simulations have been classified into five flow regimes (see also Fig 4.4 and text) and are shown in parameter space in this figure. A diagram is shown for nine combinations of mantle viscosity (columns) and subduction angle (rows). Each phase diagram shows the regimes for parameter combinations of water diffusivity (on x-axis) and subduction rate (on y-axis). A systematic shift towards "no instabilities" (NI) is observed for increasing mantle viscosity. "Shallow instabilities" (SI) are favored in geometries with a steeper subduction angle, whereas "multiple instabilities" (MI) cover a wider parameter range for shallow dip angles. The numbers indicate where plumes start (in the along slab coordinate system s) and when (time t in Myr since simulation start). The syntax reads as follows: for DI and SI: $s(t)$; for MI: $s(t),\Delta s$; for PI: $s(t)$,pulse interval (Myr).*

with a stable water-rich boundary layer on top of the slab that is advected into the deeper mantle. No instabilities emerge at any time.

The only regime leading to a steady state situation is the no diapir case (NI), as in all other experiments water is captured in the mantle wedge due to its net buoyancy. This leads to a continuously increasing total water content in the wedge with time. A steady state could potentially be reached in all simulations if water would be removed at a certain rate, for instance by incorporation of water into melt that leaves the mantle during eruptions. Since we do not include any mechanisms to remove the water, the mantle wedge becomes more water-rich with time, which affects the density contrast between wet diapirs and their surroundings, hence the forces driving the diapirs. An experiment that started as a pulsating diapir can change to a multiple instability regime if the density contrasts become smaller as the background water content increases. In this case the instability growth rate reduces (smaller buoyancy forces) and the diapir spacing consequently increases. These transient simulations are used to mark transitions between regimes in the phase diagrams (e.g. MI-PI in Fig. 4.5).

General trends in the 2D experiments are seen in the phase diagrams (Fig. 4.5). The faster the slab the less time there is for the boundary layer to grow and to become unstable before it gets dragged outside the numerical domain. The same is true for higher mantle viscosities, where the instabilities need a longer time to grow. A higher water mobility (diffusivity) allows a faster growth of the layer and favors the formation of instabilities at shallower depths, whereas a low mobility often leads to no instabilities. For steep subduction scenarios the buoyancy forces driving the instabilities and the viscous forces dragging the buoyant layer downwards become more opposing in direction. Thus, the boundary layer on top of a steep slab moves slower and accumulates faster into a layer of critical thickness. The phase diagrams indicate this: Instabilities on top of steep slabs need less time to form compared to their shallow subduction counterparts. As a consequence, the MI regime narrows in favor of the SI regime. In the limit, SI and DI merge so that no connecting MI regime is observed for any combination of subduction rate and water diffusivity (lower right panel in Fig. 4.5, 40°, 10^{20} Pa·s).

4.3.2 3D numerical experiments

With the 2D results in mind we want to compare the findings to 3D calculations of the same scenarios. The important questions are: Can the same regimes be identified in 3D models? If so, do they appear for the same subduction parameters? More generally: How much intuition can be drawn from 2D models approximating the behavior of 3D geodynamic problems? In order to make this comparison as consistent as possible we conduct the 3D experiments using a numerical model with identical solution strategy and algorithm and, most important, the same numerical grid resolution. The same time step-limiting criterion (Courant criterion) is used for the 3D runs, which along with the similar node spacing leads to equivalent time steps. The 2D and 3D codes are identical except that the 3D code runs in parallel on a 32 CPU (128 Gb memory) cluster due to its significantly higher computational workload.

Analogous to the 2D experiments described above the 3D runs start with a water-free mantle wedge at reference density into which water is released through the slab surface between 80 and 180 km depth. The evolution of an example run, referred to in the following as "3D04", is shown in Fig. 4.6 (subduction angle: 20°, subduction rate: 30 km/Myr, mantle viscosity: $3 \cdot 10^{19}$ Pa·s, water diffusivity: 10^{-7} m^{2}s^{-1}). The corresponding 2D run is

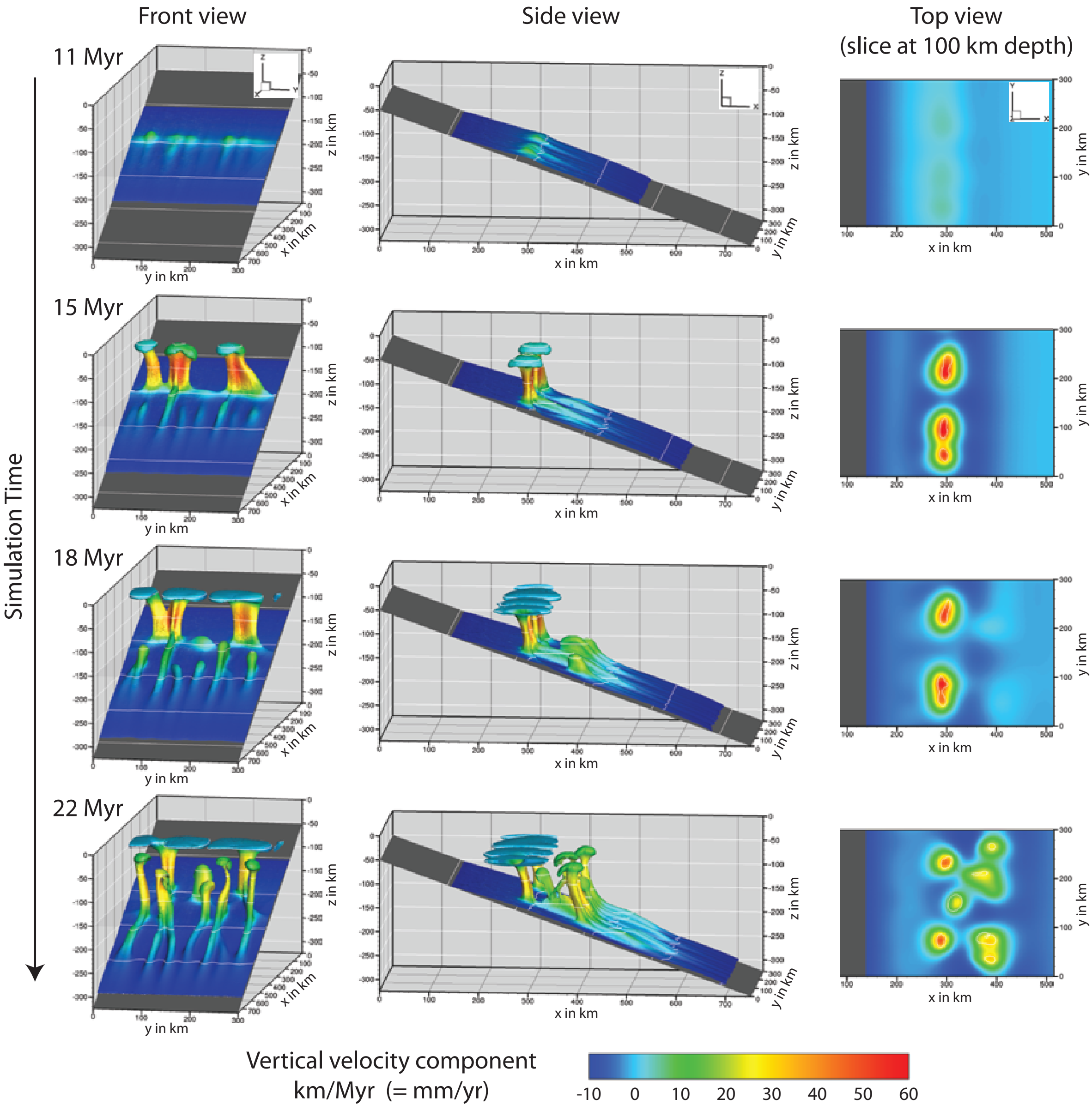


Figure 4.6: *Temporal evolution of numerical experiment* **3D04** *(angle=20°, speed=30 km/Myr, viscosity=$3 \cdot 10^{19}$ Pa·s, water diffusivity=10^{-7} m^2/sec). Left and mid column: dark grey plane shows top of the slab, colored surface represents the isosurface for 1% water content, colors show the vertical velocity component. Right column: horizontal slice at 100 km depth. Colors are vertical velocity component, white lines show where the 1% water content isosurface penetrates the slice.*

located in the MI (multiple instabilities) regime (see Fig. 4.5). The growth of the boundary layer is very similar to the 2D run until after about 6 Myr instabilities become visible as ripples that are aligned parallel to the direction of slab motion. The ripples are spaced from 40 to 80 km in the trench-parallel direction and grow slowly in amplitude for several million years. Their morphology changes eventually to less elongated cone-like structures (upper row in Fig. 4.6). After about 11 Myr enough buoyant material has accumulated to rapidly form diapirs that rise through the wedge within 3–4 Myr. Some diapirs merge during ascent so that three plumes exist at about x=300 km, two of which merge within the next 4 Myr. The disturbances resulting from the rapid removal of buoyant material cause the ripples to bend and disconnect downstream of the plume positions causing the formation of subsequent plumes at about x=400 km. Another diapir at x=300 km closes the gap between the two existing plumes. It is difficult to define a plume spacing that is characteristic for this run but the three plumes around x=300 km that exist after 22 Myr could represent such a spacing. Note that the five diapirs at greater depth begin to merge and increase their spacing to a distance similar to that in the 1st row of plumes. Although the flow dynamics obviously have a strong three-dimensional character it is interesting to note that the corresponding 2D run shows first instabilities at very similar time (11.5 Myr) and x-coordinate (x=298 km). As soon as diapirs have developed, however, the flow dynamics differ between 2D and 3D calculations.

Fig. 4.7 shows a time series of another 3D experiment (”3D02” in the following) that includes a lower viscosity mantle (10^{19} Pa·s) and a faster plate motion (60 km/Myr) compared to 3D04. The 2D run with this parameter combination predicts a behavior of 3D02 that is similar to 3D04, except that the time until instabilities form is about half as long (5.5 Myr instead of 11 Myr). Indeed, first ripple-shaped instabilities turn into cones after about 5 Myr (Fig. 4.7, first row) and a first series of plumes at x=300 km has crossed the mantle wedge after 7 Myr. Downstream the ripples buckle and rapidly turn into new instabilities that grow fast and rise close to the existing plumes. A complex diapir distribution evolves with several connected plumes forming wet ”curtains” in the large-scale mantle flow field rather than pipes. Close to the slab these curtains stretch over more than 100 km, whereas at shallower depth they are attracted to the first row of plumes and become narrower. Compared to 3D04 the initial ripple spacing is closer, their morphology is more pronounced and the disturbance of the ripples in the downstream direction is more effective. The lower viscosity of the mantle can explain both effects Ð it allows a faster deformation, growth and ascent of the diapirs. The different behavior of 3D04 and 3D02 is also seen in the horizontal slices at 100 km depth that show vertical velocity and water concentration (right columns of Fig. 4.6 and 4.7). The more isolated diapirs in 3D04 lead to separate circular regions of mantle upwelling, whereas the stretched and connected plumes in 3D02 form elongated regions of mantle decompression.

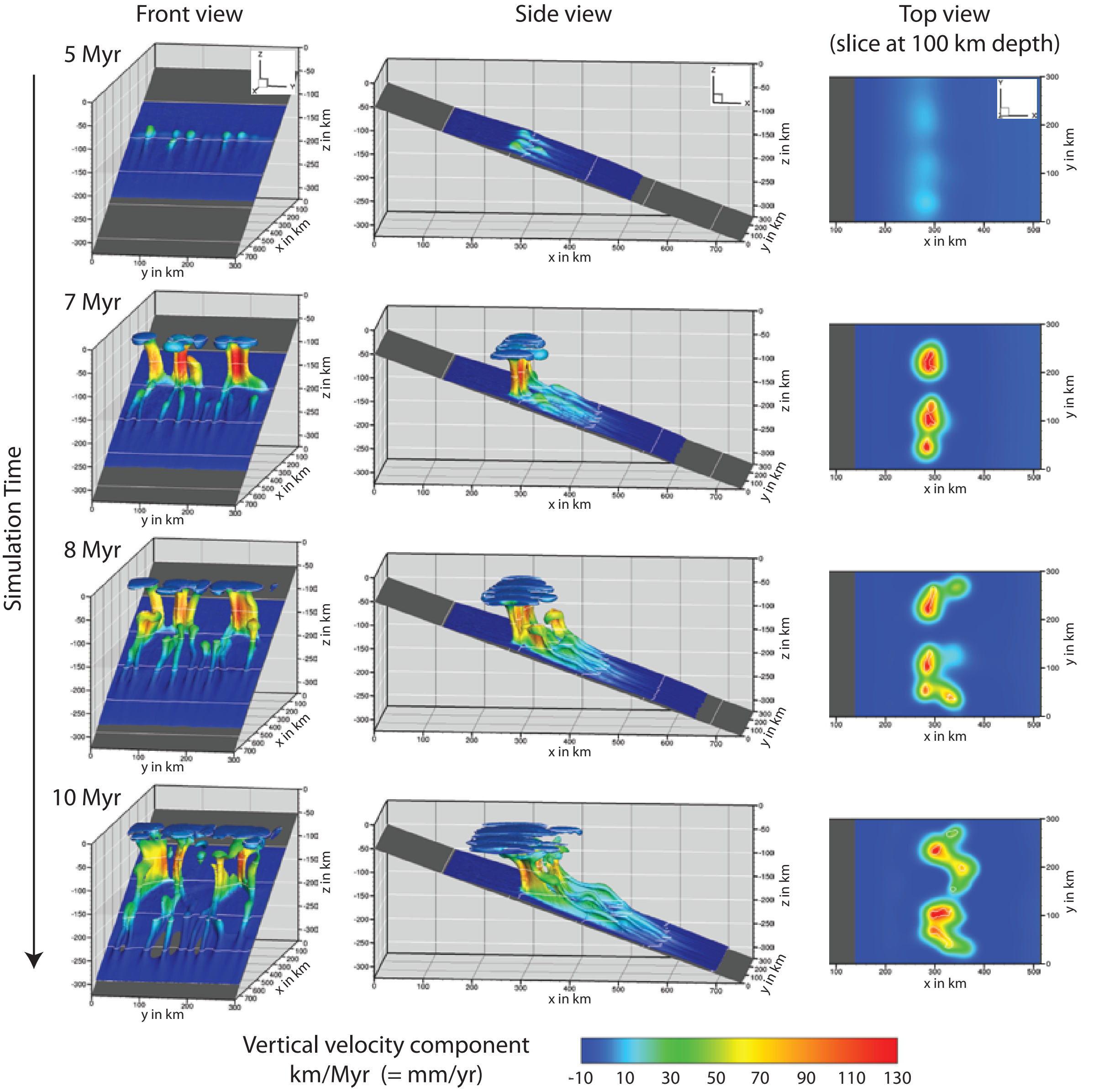


Figure 4.7: *Temporal evolution of numerical experiment* **3D02** *(angle=20°, speed=60 km/Myr, viscosity=*10^{19} Pa·s*, water diffusivity=*10^{-7} m^2*/sec). Left and mid column: dark grey plane shows top of the slab, colored surface represents the isosurface for 1% water content, colors show the vertical velocity component. Right column: horizontal slice at 100 km depth. Colors are vertical velocity component, white lines show where the 1% water content isosurface penetrates the slice.*

Fig. 4.8 shows the unusual case of a 2D-like upwelling that we observed only in one 3D calculation ("3D13"). The initial ripple structures are closely spaced in the y-direction and turn into cones almost simultaneously (6 Myr at x=125 km). Although the subsequent upwelling is strongest in two regions (between y=35–129 km and y=184–272 km at 10 Myr), the variations in water concentration in the along-trench direction are small and a uniform sheet-like upwelling develops shortly after (15 Myr). This stable region of mantle upwelling is repeatedly disturbed by single diapirs that are generated downstream of the

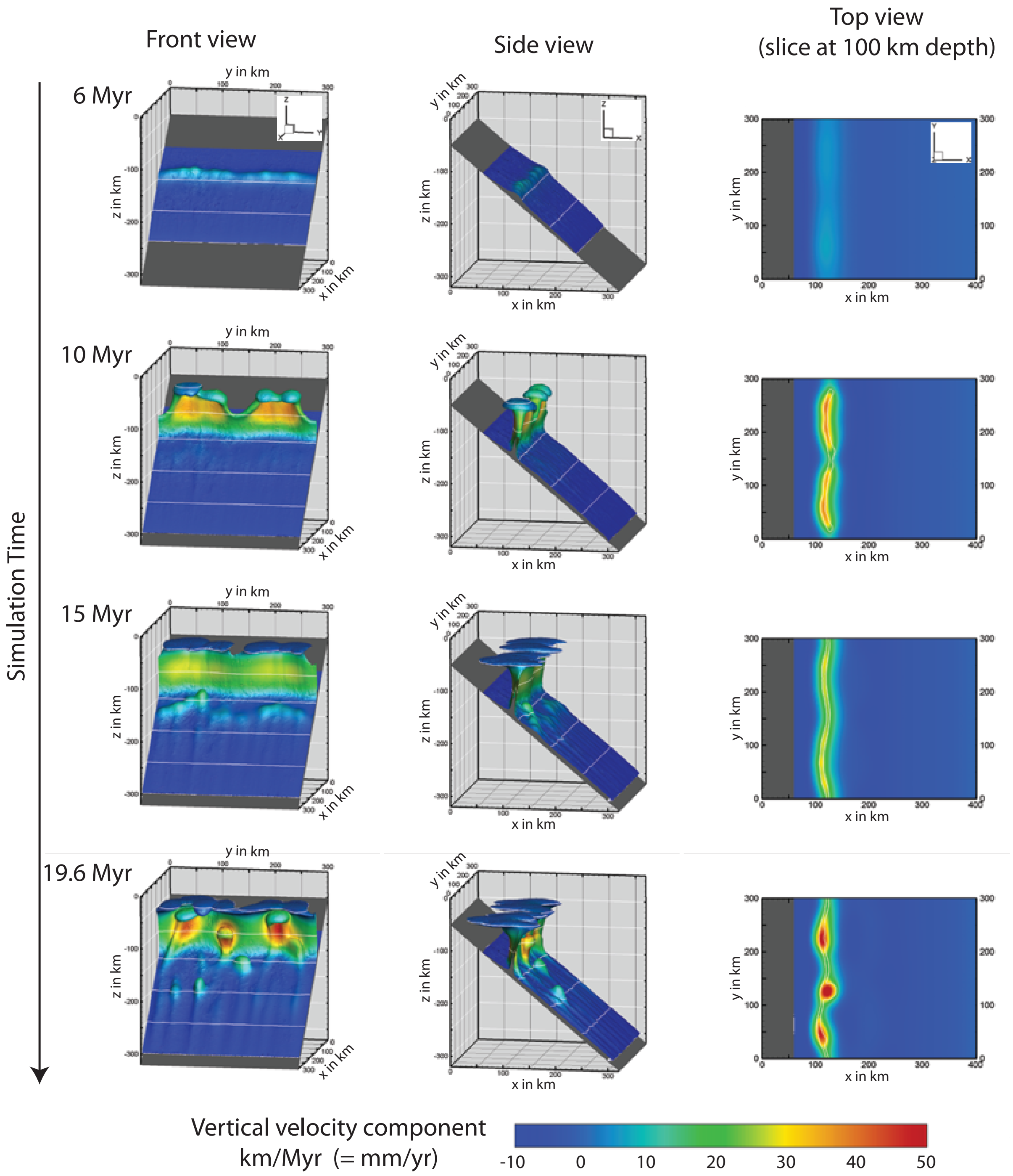


Figure 4.8: *Temporal evolution of numerical experiment* **3D13** *(angle=40°, speed=30 km/Myr, viscosity=*$3 \cdot 10^{19}$ Pa·s*, water diffusivity=*10^{-7} *m*^{2}*/sec). Left and mid column: dark grey plane shows top of the slab, colored surface represents the isosurface for 1% water content, colors show the vertical velocity component. Right column: horizontal slice at 100 km depth. Colors are vertical velocity component, white lines show where the 1% water content isosurface penetrates the slice.*

sheet, migrate upwards along the slab surface and detach at the location of the sheet (for example at 19.6 Myr, where three diapirs with a higher vertical velocity simultaneously rise through the sheet). The corresponding 2D run (PI-MI regime transition) predicts

Table 4.2: *Parameter settings of all 3D runs in this study, as well as locations and times of diapirs (if present). All 3D runs have a water diffusivity of* $10^{-7} m^2 s^{-1}$*. Locations of diapirs are given in the along slab coordinate s (see Fig.4.1c). Time t is taken when the instability grew to twice the layer thickness.*

case	Setup			3D			2D	
	angle °	viscosity *Pa s*	speed *mm/yr*	# diapirs (rip.)	s (t) km (Myr)	Δy diapirs (rip.) km	regime	s (t) km (Myr)
3D01	20	10^{19}	30	8 → 5 (11)	27 (5)	30-50 (20-30)	(PI-)MI	15 (5.7)
3D02[a]	20	10^{19}	60	5 → 3 (10)	75 (4.5)	60-120 (20-40)	MI	95 (5.5)
3D03	20	10^{19}	90	4 → 3 (9)	138 (4)	50-100 (20-50)	MI	217 (6.7)
3D04[b]	20	$3 \cdot 10^{19}$	30	5 → 3 → 2 (7)	83 (10.5)	60-110 (30-50)	MI	98 (11.5)
				5 → 4 (6)	181 (17.5)	40-85 (33-55)		
3D05	20	$3 \cdot 10^{19}$	60	3 → 2 (4)	309 (12)	114 (23, 57, 70-80)	DI-NI	-
					475 (19)			
3D06	20	$3 \cdot 10^{19}$	90	0 (5)	-	- (33-77)	NI	-
3D07	20	10^{20}	30	0 (3-4)	-	60-100	NI	-
3D08	20	10^{20}	60	0 (0)	-	-	NI	-
3D09	20	10^{20}	90	0 (0)	-	-	NI	-
3D10[c]	20	$3 \cdot 10^{19}$	30	3 (4-6)	165 (15)	40-100 (40-70)	(MI-)DI	225 (23)
3D11[c]	20	$3 \cdot 10^{19}$	60	0 (0-2)	-	- (30-72)	NI	-
3D12[c]	20	$3 \cdot 10^{19}$	90	0 (0)	-	-	NI	-
3D13[d]	40	$3 \cdot 10^{19}$	30	4 →sheet (8-10)	39 (5.5)	57 (22-37)	PI-MI	41 (5.6)
				$2-6$ (8-10)	65, 131 (>13)	52-97 (22-37)		
3D14	40	$3 \cdot 10^{19}$	60	2 (5)	143 (7.5)	127-173 (42-88)	DI	184 (18.4)
3D15	40	$3 \cdot 10^{19}$	90	0 (1)	-	-	NI	-
3D16[e]	40	10^{20}	30	2 (6)	158 (21)	137-163 (42-52)	NI	-
3D16-2.5[f]	40	10^{20}	30	2 (?)	95, 120 (15)	143 (?)	NI	-
3D16-5[g]	40	10^{20}	30	2 (?)	47 (9)	151 (?)	NI	-

[a] Run shown in Fig.4.6

[b] Run shown in Fig.4.7

[c] Run has 50% less water influx, i.e. 40% of the slabs water content is released instead of 80%.

[d] Run shown in Fig.4.8

[e] Run shown in Fig.4.10 (right column)

[f] Low resolution (2.3 km) run, shown in Fig.4.10 (mid column)

[g] Low resolution (5 km) run, shown in Fig.4.10 (left column)

the exact same location and onset time for the first sheet-like instability. Discrepancies between the 2D and 3D run exist for the dynamics of the subsequent instabilities that form downstream of the first upwelling. In the 3D calculation, they develop into separate instabilities that ascend faster than the instabilities in the 2D model.

We conducted a set of 16 high-resolution 3D runs for different parameter combinations and summarize characteristic times and length scales in Tab. 4.2. Snapshots of six of these runs are shown in Fig. 4.9. With increasing slab speed the plumes are located deeper and eventually disappear from the model domain. The diapir spacing in the along-trench direction increases with subduction rate, especially for the steep subduction case. Here (as an end-member result) a sheet-like upwelling at shallow depth is observed for the slowest slab speed.

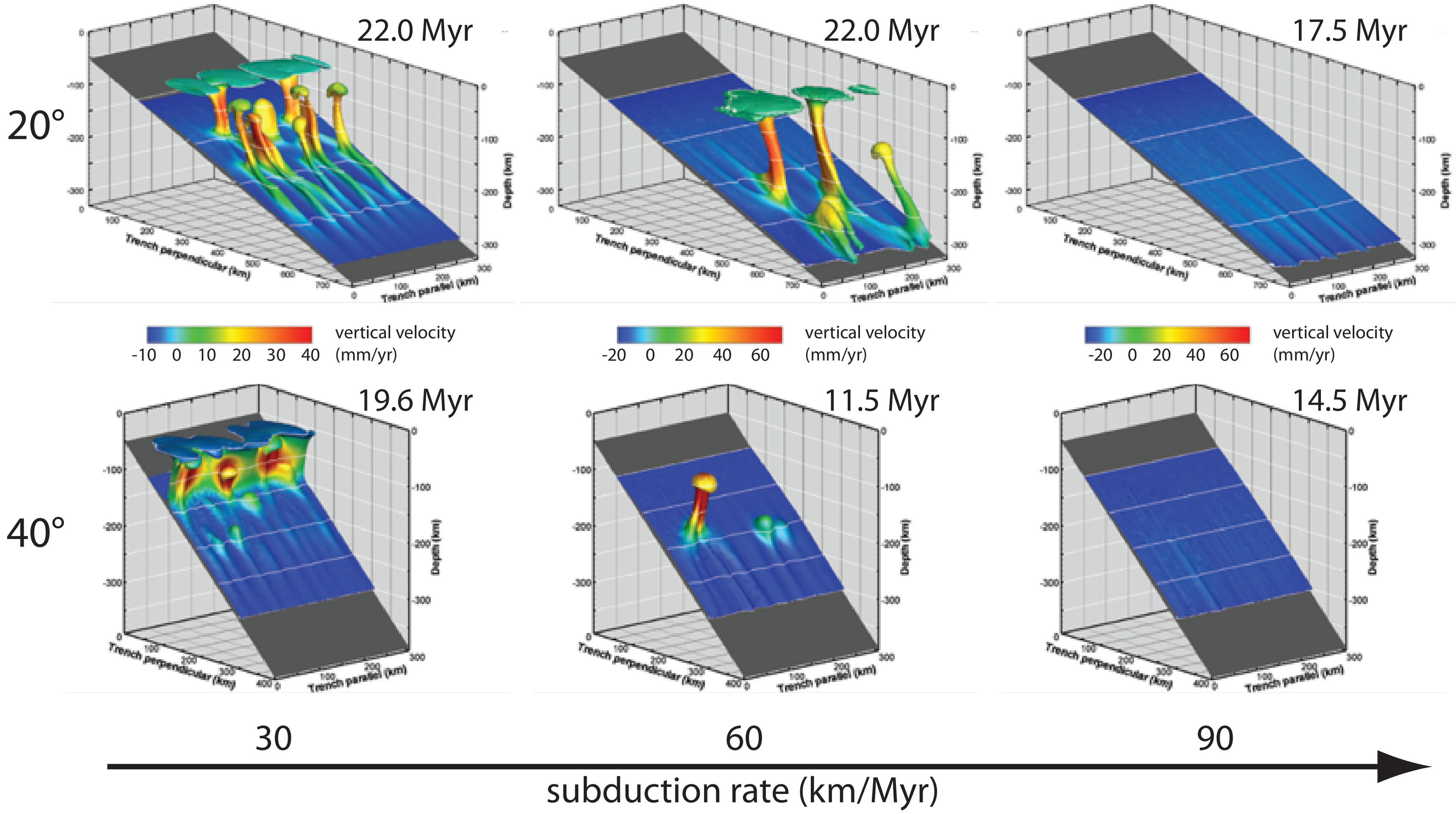


Figure 4.9: *Snapshots of 3D runs with a 20° (top row) and 40° subduction geometry. Slab speed increases from left to right, other parameters are the same (viscosity=*$3 \cdot 10^{19}$ Pa·s*, water diffusivity=*10^{-7} *m*^{2}*/sec).*

4.4 Discussion

The subduction process clearly involves large temperature contrasts as the descending plate is about zero degrees Celsius at its top prior to subduction, and continuously heats up towards ambient mantle temperatures as it undergoes subduction. The material in the mantle wedge mainly provides the required thermal energy. It has been generally supposed that a continuous influx of hotter mantle prevents the wedge from freezing out (e.g. van Keken, 2003). It is also known that the viscosity of mantle rocks is strongly temperature dependent, thus, flow at subduction zones involves and is strongly affected by viscosity changes. Nevertheless, for the present study we decided to solve for iso-viscous flow with pseudo-diffusive water migration. This approach has the advantage of a greatly simplified model geometry that leads to an easier inter-comparison between different model calculations.

We have conducted 2D and 3D numerical experiments to investigate Rayleigh-Taylor-like instabilities caused by compositional buoyancy resulting from slab dehydration. The first indication that the buoyant boundary layer on top of the slab is becoming unstable is the formation of ripples that are aligned parallel to the slab motion. We see these wave-like instability patterns in all 3D calculations, but they become more pronounced for shallow subduction geometries and low mantle viscosities (e.g. Runs 3D01-3D06). Lower viscosity would lead to faster viscous deformation if all other forces (e.g. buoyancy forces) remained the same. Compared to a steeply descending slab, the shallow subduction case is closer to the classic Rayleigh-Taylor instability (RTI) because the buoyancy forces in the underlying layer are acting almost parallel to the density gradient. However, several differences exist between the classic RTI and the geodynamic problem studied here. The buoyancy of the underlying layer increases with time because water is added to it as the slab dehydrates. The water also migrates, which increases the thickness of the buoyant layer. Depending on how fast the water migration is compared to the fastest growing wavelength of the associated RTI, the morphology of the instability can be similar to or quite different from the classic RTI. Unfortunately, the water migration speed is not well constrained since the migration mechanism itself is poorly known. Another difference with RTI are shear forces, resulting from viscous drag by the slab, that act parallel to the boundary layer. This additional force is probably why we see wave-like initial structures that are elongated in the direction of motion rather than cone-like instabilities. Cone-like structures resulting from diapiric upwelling arise only after the ripples grow to a thickness that allows focused flow within the buoyant layer. In case of a steep subduction angle the buoyancy forces driving the instabilities have a larger component opposing the viscous drag of the slab. Thus, the subduction rate of the buoyant layer may be considerably reduced, depending on the viscosity structure within the mantle wedge. This can lead to a faster and shallower instability, and can also overprint the initial wave-like instabilities. We think this effect is seen in run 3D13 (Fig. 4.8), where the upwelling is almost sheet-like and ripples are hardly seen beforehand.

A comparison of 2D and 3D calculations with the same parameter settings reveals that the location and onset time for the first instability are often similar (e.g. 3D04, 3D07) or even identical (3D13). Several parameter combinations that lead to no instabilities in the 2D models do not show diapirs in 3D either (3D06, 3D08, 3D09, 3D11, 3D12, 3D15). Thus, less time-consuming 2D calculations are a good way to scan the parameter space beforehand, to find parameter combinations that are worthwhile for computationally expensive 3D calculations. The exact evolution of a three-dimensional geodynamical problem in which buoyancy forces are important, however, cannot be studied in two-dimensional models. Rise times of diapirs or the formation time of subsequent diapirs, for instance, strongly differ between our 2D and 3D calculations.

In general we find, that diapirs develop easier and faster in the 3D models. The few 3D runs with diapirism, whose 2D counterparts show no diapirs, indicate this. We think this has the following reasons:

1. In 3D, the diapirs in the mantle wedge do not necessarily interrupt the corner flow, whereas the 2D upwellings do.

2. The preferred and most efficient shape of the upwellings is a pipe-like upward flow, which cannot exist in 2D.

3. The flow in the source layer that feeds the diapirs is more efficient in 3D — it transports material from both horizontal directions towards the diapir's root, whereas in 2D flow occurs along one horizontal axis.

The diapirs provide a potential mechanism for decompression melting in the mantle wedge. Clustering of quaternary volcanic centers along the Honshu arc (for Catalogue of Quaternary Volcanoes in Japan, 1999) and its correlation with low seismic velocity anomalies in the underlying mantle wedge has led to the idea of ”hot fingers” (Tamura et al., 2002) underlying the subduction zone volcanoes. Honda and Saito (2003) suggest that small-scale convection in the wedge generates trench-perpendicular elongated low velocity anomalies. They focused on this mechanism in several subsequent publications, because the elongated shapes do not fit well into the common image that plume-like diapirism would lead to circular upwelling regions. However, the diapiric structures that we see in our calculations are often elongated as they emerge from the wave-like instabilities forming on top of the slab (see Fig. 4.7). In the case of more cylindrical plume-like upwellings, the regions of mantle decompression can become elongated if diapirs are cascaded in the

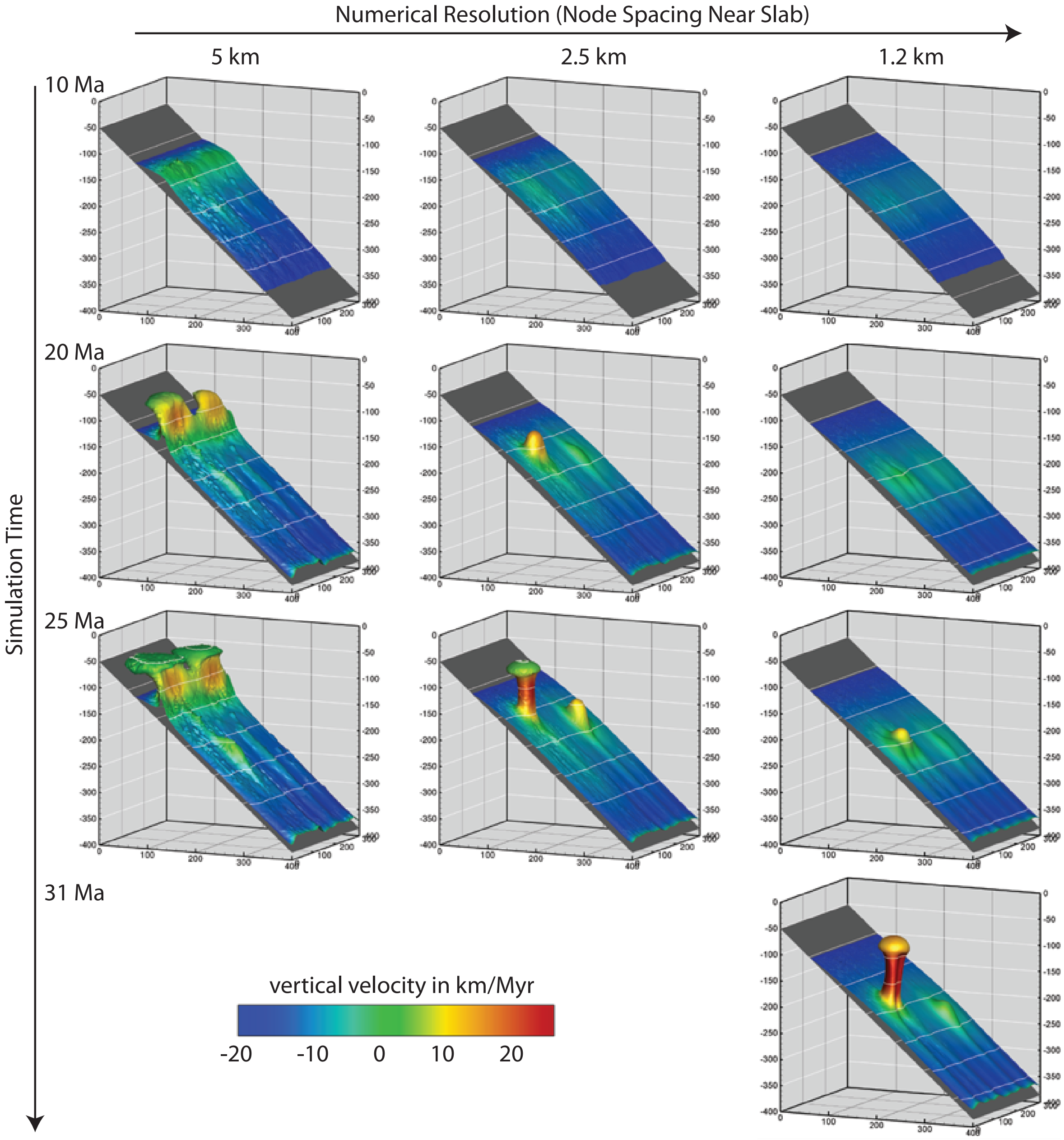

Numerical Resolution (Node Spacing Near Slab)
5 km
2.5 km
1.2 km
10 Ma
20 Ma
25 Ma
31 Ma
Simulation Time
vertical velocity in km/Myr
-20
-10
0
10
20

Figure 4.10: *Temporal evolution of three numerical runs with identical parameter settings: subduction angle 40°, subduction rate 30 km/Myr, mantle viscosity* 10^{20} Pa·s, *water diffusivity* 10^{-7} m^2/sec. *The only difference is the resolution of the numerical grid: 5 km node spacing (run* **3D16-5**, *left column), 2.5 km (***3D16-2.3***, middle column), and 1.2 km (***3D16***, right column). The coarse grid cannot properly resolve the growing boundary layer and overestimates its thickness and growth rate. This leads to the wrong result of a shallow sheet-like instability. The intermediate run captures the basic dynamics of the problem (two diapirs instead of a sheet) but is still wrong in the predicted time and along-slab position of the plumes. We believe that only the highest resolution tested predicts the correct behavior in both, time and space. All 2D and 3D runs presented in this study have been conducted using the 1.2 km grid resolution resolution of the run on the right hand side.*

trench-perpendicular direction as shown in Fig. 4.6. Cascading is likely to occur as the diapirs arise from the wave-like instabilities on top of the slab, which themselves are elongated parallel to the slab's motion. Most obvious in our 2D calculations but also in the 3D models, a starting diapir disturbs its source layer and can easily give rise to new subsequent instabilities in the downstream direction. Even in the calculation that shows a strong 2D character of sheet-like initial diapirism (Fig. 4.8), subsequent diapirs form downstream of the established upwelling region and lead to elongated trench-perpendicular regions of preferred diapirism. These patterns arise when the water migration mechanism conserves chemical buoyancy. In other calculations where the water is assumed to rapidly ascent in only the vertical direction, sheet-like upwelling structures were found (e.g. Honda et al., 2010).

Water/fluid transport in the mantle wedge is a poorly known process; different numerical formulations have been used to approximate it (e.g. Cagnioncle et al., 2007; Gerya and Yuen, 2003). We decided to treat water migration as a diffusion-like process (Eq. 4.5) in all calculations presented here, for the following reasons: 1) It is numerically stable and depends on the single easy-to-control model parameter — diffusivity. 2) The volume of water is conserved during its migration. This is of great importance since the amount of water available at the top of the slab is critical for the boundary layer growth and the evolution of instabilities. Numerical formulations that are based on particle-tracking (also called tracers or markers based methods) typically have problems in conserving the mass of the compositional field which they represent once their properties are mapped to the numerical grid where they affect density and/or viscosity. 3) A diffusion equation allows us to exactly define a water influx boundary condition. Using the marker technique it is difficult to create a smooth and uniform influx because each particle represents a certain volume or mass of material, thus, only discrete volume fractions can be added with time. To overcome this problem a large number of tracers would be required, which strongly increases the computational workload, especially in high-resolution 3D calculations. Using too few particles on the other hand can be dangerous because it introduces disturbances to the propagating hydrous front. This is likely to affect the boundary layer dynamics and the time evolution of Rayleigh-Taylor instabilities, as their spacing and growth rate is partly controlled by initial disturbances (e.g. Schmeling, 1987).

One of our findings is the importance of the numerical resolution that is used for the model calculations. Insufficient numerical resolution will lead to an overestimate of the growth rate of the initial instabilities. Artificially high growth rates favor the formation of sheet-like, less elongated mantle upwellings. This is highlighted by the three calculations shown in Fig. 4.10. The calculations are identical in all parameters except that the numerical node spacing decreases by a factor of 2 from the left to the middle column (5 km vs. 2.5 km near the slab), and by another factor of 2 from the middle to the right column (2.5 km vs. 1.2 km). In terms of computational work these refinements correspond to 220k, 1,6M, and 12M velocity unknowns, resp., which is a factor of about 7.4 increase in problem size with each refinement. This odd factor results from the unstructured nature of the mesh that, as opposed to a structured FD grid, has no definite number of nodes into the x-, y-, or z-dimension.

The seriously under-resolved run (Fig. 4.10, left column, 5 km minimum node spacing) predicts an almost two-dimensional flow field, in which the buoyant mantle rises as a sheet at shallow depth. Refining the node spacing by a factor of two and conducting the exact same calculation we observe two diapirs that develop at a later time and at greater depth (middle column of Fig. 4.10, 2.3 km minimum node spacing). Here the flow field is clearly three-dimensional. Another mesh refinement with half the node spacing and the same model parameters leads to two diapirs somewhat similar to the ones observed in the intermediate resolution run, but they form at a later time and at greater depth (Fig. 4.10, right column, 1.2 km minimum node spacing). We therefore conclude that run 3D16-5 fails to predict the correct flow field and the correct temporal evolution of the problem at hand, run 3D16-2.3 captures the fundamental dynamics but still fails to predict the correct temporal and spatial evolution. Only run 3D16 is suitable to study the problem at hand. Assuming that 3D16 shows the physically correct solution the temporal errors of 3D16-5 and 3D16-2.3 are 57% and 29%, and their spatial (x-location) errors are 39% and 22%, resp. For all 3D calculations presented in this study we have used highest resolution, namely 1.2 km node spacing near the slab's top. In general, a grid (or node spacing) of 1 km and less should be used in numerical calculations investigating dynamics at subduction zones in order to ensure semi-quantitatively correct results. Qualitative conclusions may be drawn from less well-resolved calculations with 2–3 km grid or node spacing. A coarser numerical discretization holds the danger of misleading fluid dynamical results and geodynamic conclusions. The numerical resolution is even more important for variable viscosity calculations where flow in narrow weak boundary layers will have to be properly resolved (Phipps Morgan et al., 2007). In spite of these potential resolution issues, the 3D experiments show geologically intriguing structures that motivate an alternative slab-dehydration-linked origin of so-called ”hot fingers” patterns in the distribution of arc volcanic centers.

Bibliography

Allegre, C. and D. Turcotte (1986). Implications of a two-component marble-cake mantle. *Nature 323*, 123–127.

Almeev, R., F. Holtz, J. Koepke, K. Haase, and C. Devey (2007). Depths of partial crystallization of H2O-bearing MORB: Phase equilibria simulations of basalts at the MAR near Ascension Island (7 ° 11 ° S). *J. Petrol. 49*(1), 25–45. doi: 10.1093/petrology/egm068.

Amante, C. and B. W. Eakins (2009). ETOPO1 1 arc-minute global relief model: Procedures, data sources and analysis. *NOAA Technical Memorandum NESDIS NGDC-24*, 19pp.

Amestoy, P. R., I. S. Duff, S. Pralet, and C. Vömel (2003). Adapting a parallel sparse direct solver to architectures with clusters of SMPs. *Parallel Computing 29*, 1645–1668. doi: 10.1016/j.parco.2003.05.010.

Arrow, K. J. and L. Hurwicz (1958). Gradient method for concave programming I: local results, in: *Studies in Linear and Nonlinear Programming* (K. J. Arrow, L. Hurwicz, and H. Uzawa (eds.)), Stanford University Press, Stanford, CA.

Asimow, P., J. Dixon, and C. Langmuir (2004). A hydrous melting and fractionation model for mid-ocean ridge basalts: Application to the Mid-Atlantic Ridge near the Azores. *Geochem. Geophy. Geosys. 5*(1), Q01E16.

Bass, J. D. (1995). Elasticity of Minerals, Glasses, and Melts, in: *A Handbook of Physical Constants* (T. J. Ahrens (ed.)), American Geophysical Union.

Batchelor, G. K. (1967). *An Introduction to Fluid Dynamics.* Cambridge University Press.

Bathe, K.-J. (1996). *Finite Element Procedures.* Prentice-Hall, Inc.

Behn, M., M. Boettcher, and G. Hirth (2007). Thermal structure of oceanic transform faults. *Geology 35*(4), 307–310.

Benzi, M., G. Golub, and J. Liesen (2005). Numerical solution of saddle point problems. *Acta Numerica 14*, 1–137.

Bercovici, D. (2007). Mantle Dynamics Past, Present, and Future: An Introduction and Overview, in: *Mantle Dynamics* (G. Schubert and D. Bercovici (eds.)), *Treatise on Geophysics, 7*, Elsevier.

Bonatti, E. (1990). Not so hot "hot spots" in the oceanic mantles. *Science 250*, 107–111.

Bourdon, E. and C. Hemond (2001). Looking for the "missing endmember" in South Atlantic ocean mantle around Ascension Island. *Miner. Petrol. 71*(1), 127–138.

Braun, M. and R. Sohn (2003). Melt migration in plume-ridge systems. *Earth Planet. Sc. Lett. 213*, 417–430.

Briggs, W. L., V. E. Henson, and S. F. McCormick (2000). *A Multigrid Tutorial.* SIAM.

Bruguier, N. J., T. A. Minshull, and J. M. Brozena (2003). Morphology and tectonics of the Mid-Atlantic Ridge, 7 °–12 °S. *J. Geophys. Res. 108*(B2), 2093. doi: 10.1029/2001JB001172.

Cagnioncle, A., E. M. Parmentier, and E. LT (2007). Effect of solid flow above a subducting slab on water distribution and melting at convergent plate boundaries. *J. Geophys. Res. 112*, B09402. doi: 10.1029/2007JB004934.

Chen, Y., T. A. Davis, W. W. Hager, and S. Rajamanickam (2006). Algorithm 8xx: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. Technical Report TR-2006-005, CISE Dept, Univ. of Florida, Gainesville, FL.

Connolly, J. A. D. and K. Petrini (2002). An automated strategy for calculation of phase diagram sections and retrieval of rock properties as a function of physical conditions. *J. Metamorph. Geol. 20*, 697–708.

Connolly, J. A. D., M. W. Schmidt, G. Solferino, and N. Bagdassarov (2009). Permeability of asthenospheric mantle and melt extraction rates at mid-ocean ridges. *Nature 462*(7270), 209–212. doi: 10.1038/nature08517.

Cordery, M. J. and J. Phipps Morgan (1993). Convection and melting at mid-ocean ridges. *J. Geophys. Res. 98*(B11), 19,477–19,503.

Dabrowski, M., M. Krotkiewski, and D. Schmid (2008). MILAMIN: MATLAB-based finite element method solver for large problems. *Geochem. Geophy. Geosys. 9*(4), Q04030. doi: 10.1029/2007GC001719.

Davies, J. and D. Stevenson (1992). Physical model of source region of subduction zone volcanics. *J. Geophys. Res. 97*(B2), 2037–2070.

Davies, T. A. (2006). *Direct Methods for Sparse Linear Systems.* SIAM Book Series on the Fundamentals of Algorithms. Philadelphia: SIAM.

Deubelbeiss, Y. and B. Kaus (2008). Comparison of Eulerian and Lagrangian numerical techniques for the Stokes equations in the presence of strongly varying viscosity. *Phys. Earth Planet. Int. 171*(1-4), 92–111.

Donea, J. and A. Huerta (2003). *Finite Element Methods for Flow Problems.* Wiley.

Eberle, M., O. Grasset, and C. Sotin (2002). A numerical study of the interaction between the mantle wedge, subducting slab, and overriding plate. *Phys. Earth Planet. Int. 134*, 191–202.

Elliott, T. (2003). Tracers of the slab, in: *Inside the subduction factory* (J. M. Eiler (ed.)), *Geophysical Monograph, 138*, American Geophysical Union.

Elman, H., V. Howle, J. Shadid, and R. Shuttleworth (2006). Block preconditioners based on approximate commutators. *SIAM J. Sci. Comput. 27*(5), 1651–1668.

Engelman, M. S., R. L. Sani, P. M. Gresho, and M. Bercovier (1982). Consistent vs. reduced integration penalty methods for incompressible media using several old and new elements. *Int. J. Numer. Meth. Fl. 2*, 25–42.

for Catalogue of Quaternary Volcanoes in Japan, C. (1999). Catalogue of quaternary volcanoes in Japan. Technical report.

Forsyth, D. W. and B. Wilson (1984). Three-dimensional temperature structure of a ridge-transform-ridge system. *Earth Planet. Sc. Lett. 70*, 355–362.

Fortin, M. and A. Fortin (1985). Newer and Newer Elements for Incompressible Flow, in: *Finite elements in Fluids* (R. H. Gallagher, G. F. Carey, J. T. Oden, and O. C. Zienkiewicz (eds.)), Volume 6, Wiley.

Gerya, T. and D. Yuen (2003). Rayleigh–Taylor instabilities from hydration and melting propel 'cold plumes' at subduction zones. *Earth Planet. Sc. Lett. 212*, 47–62.

Ghiorso, M. S. and R. O. Sack (1995). Chemical mass transfer in magmatic processes IV. A revised and internally consistent thermodynamic model for the interpolation and extrapolation of liquid-solid equilibria in magmatic systems at elevated temperatures and pressures. *Contrib. Mineral. Petr. 119*, 197–212.

Gorczyk, W., T. V. Gerya, J. A. D. Connolly, D. A. Yuen, and M. Rudolph (2006). Large-scale rigid-body rotation in the mantle wedge and its implications for seismic tomography. *Geochem. Geophy. Geosys. 7*(5), Q05018. doi: 10.1029/2005GC001075.

Hansen, U. and A. Ebel (1988). Time-dependent thermal convection – a possible explanation for a multiscale flow in the earth's mantle. *Geophys. J. Int. 94*, 181–191.

Helffrich, G. and B. Wood (2001). The earth's mantle. *Nature 412*, 501–507.

Hestenes, M. R. (1969). Multiplier and gradient methods. *J. Optimiz. Theory App. 4*, 303–320.

Hestenes, M. R. and E. Stiefel (1952). Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand. 49*(6), 409–436.

Hirschmann, M. (2000). Mantle solidus: experimental constraints and the effects of peridotite composition. *Geochem. Geophy. Geosys. 1*, 2000GC000070.

Hirschmann, M., M. Ghiorso, and L. Wasylenki (1998). Calculation of peridotite partial melting from thermodynamic models of minerals and melts. I. Review of methods and comparison with experiments. *J. Petrol. 39*(6), 1091–1115.

Hirth, G. and D. Kohlstedt (1996). Water in the oceanic upper mantle: Implications for rheology, melt extraction and the evolution of the lithosphere. *Earth Planet. Sc. Lett. 144*(1-2), 93–108.

Hirth, G. and D. Kohlstedt (2003). Rheology of the Upper Mantle and the Mantle Wedge: A View from the Experimentalists, in: *Inside the subduction factory* (J. M. Eiler (ed.)), *Geophysical Monograph, 138*, American Geophysical Union.

Honda, S., T. Gerya, and G. Zhu (2010). A simple three-dimensional model of thermochemical convection in the mantle wedge. *Earth Planet. Sc. Lett. 290*, 311–318. doi: doi:10.1016/j.epsl.2009.12.027.

Honda, S. and M. Saito (2003). Small-scale convection under the back-arc occurring in the low viscosity wedge. *Earth Planet. Sc. Lett. 216*(4), 703–715. doi: 10.1016/S0012-821X(03)00537-5.

Honda, S. and T. Yoshida (2005). Effects of oblique subduction on the 3-D pattern of small-scale convection within the mantle wedge. *Geophys. Res. Lett. 32*, 1–4. doi: 10.1029/2005GL023106,.

Honda, S., T. Yoshida, and K. Aoike (2007). Spatial and temporal evolution of arc volcanism in the northeast Honshu and Izu-Bonin arcs: Evidence of small-scale convection under the island arc? *Island Arc 16*, 214–223. doi: 10.1111/j.1440-1738.2007.00567.x.

Hughes, T. J. R. (2000). *The Finite Element Method.* Dover Publications.

Hughes, T. J. R., W. K. Liu, and A. Brooks (1979). Finite element analysis of incompressible viscous flows by the penalty function formulation. *J. Comput. Phys. 30*, 1–60.

Ito, G. and J. Mahoney (2005). Flow and melting of a heterogeneous mantle: 1. Method and importance to the geochemistry of ocean island and mid-ocean ridge basalts. *Earth Planet. Sc. Lett. 230*(1-2), 29–46.

Iwamori, H. (1998). Transportation of H2O and melting in subduction zones. *Earth Planet. Sc. Lett. 160*, 65–80.

J Lin, J., G. Purdy, H. Schouten, J. Sempere, and C. Zervas (1990). Evidence from gravity data for focused magmatic accretion along the Mid-Atlantic Ridge. *Nature 344*, 627–632.

Jha, K., E. M. Parmentier, and J. Phipps-Morgan (1994). The role of mantle-depletion and melt-retention buoyancy in spreading-center segmentation. *Earth Planet. Sc. Lett. 125*, 221–234.

Johnson, H. P. and R. L. Carlson (1992). Variation of sea floor depth with age: a test of models based on drilling results. *Geophys. Res. Lett. 19*, 1971–1974.

Karato, S. and P. Wu (1993). Rheology of the upper mantle: a synthesis. *Science 260*, 771–778.

Katz, R., M. Spiegelman, and C. Langmuir (2003). A new parameterization of hydrous mantle melting. *Geochem. Geophy. Geosys. 4*(9), 1073. doi: 10.1029/2002GC000433.

Kelemen, P., G. Hirth, N. Shimizu, and M. Spiegelman (1997). A review of melt migration processes in the adiabatically upwelling mantle beneath oceanic spreading ridges. *Philos. T. R. Soc. A 355*, 283–318.

Kelemen, P. B., J. L. Rilling, E. M. Parmentier, L. Mehl, and B. R. Hacker (2003). Thermal structure due to solid-state flow in the mantle wedge beneath arcs, in: *Inside the subduction factory* (J. M. Eiler (ed.)), *Geophysical Monograph, 138*, American Geophysical Union.

Kohlstedt, D. and B. Holtzman (2009). Shearing melt out of the earth: An experimentalist's perspective on the influence of deformation on melt extraction. *Ann. Rev. Earth Planet. Sci. 37*, 561–593. doi: 10.1146/annurev.earth.031208.100104.

Kuo, B.-Y. and D. W. Forsyth (1988). Gravity anomalies of the ridge-transform system in the South Atlantic between 31 and 34.5 °S: Upwelling centers and variations in crustal thickness. *Mar. Geophy. Res. 10*, 205–232.

Lee, C.-T., Q. Yin, R. L. Rudnick, J. T. Chesley, and S. B. Jacobsen (2000). Osmium isotopic evidence for mesozoic removal of lithospheric mantle beneath the Sierra Nevada, California. *Science 289*, 1912–1916. doi: 10.1126/science.289.5486.1912.

Ligi, M., E. Bonatti, A. Cipriani, and L. Ottolini (2005). Water-rich basalts at mid-ocean-ridge cold spots. *Nature 434*, 66–69.

Maday, Y. and A. T. Patera (1989). Spectral element methods for the incompressible Navier-Stokes equations, in: *State-of-the-art surveys on computational mechanics* (A. K. Noor and J. T. Oden (eds.)), ASME.

Malkus, D. S. and T. J. R. Hughes (1978). Mixed finite element methods – reduced and selective integration techniques: A unification of concepts. *Comput. Methods Appl. Mech. Engrg. 15*(1), 63–81.

Manley, C., A. Glazner, and G. Farmer (2000). Timing of volcanism in the Sierra Nevada of California: Evidence for pliocene delamination of the batholithic root? *Geology 28*(9), 811–814.

Marsh, B. and I. Carmichael (1974). Benioff zone magmatism. *J. Geophys. Res. 79*, 1196–1206.

May, D. and L. Moresi (2008). Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics. *Phys. Earth Planet. Int. 171*, 33–47. doi: 10.1016/j.pepi.2008.07.036.

Minshull, T. A., N. J. Bruguier, and J. M. Brozena (1998). Ridge-plume interactions or mantle heterogeneity near Ascension Island? *Geology 26*(2), 115–118.

Morgan, W. J. (1971). Convection plumes in the lower mantle. *Nature 230*, 42–43.

Nakajima, J. and A. Hasegawa (2003). Estimation of thermal structure in the mantle wedge of northeastern Japan from seismic attenuation data. *Geophys. Res. Lett. 30*(14), 1760. doi: 10.1029/2003GL017185.

Oxburgh, E. R. and E. M. Parmentier (1977). Compositional and density stratification in oceanic lithosphere – causes and consequences. *J. Geol. Soc. Lond. 133*, 343–355.

Paige, C. and M. Saunders (1975). Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal. 12*(4), 617–629.

Parmentier, E. M. and J. Phipps Morgan (1990). Spreading rate dependence of three-dimensional structure in oceanic spreading centres. *Nature 348*, 325–328.

Peacock, S. (1991). Numerical simulation of subduction zone pressure-temperature-time paths: constraints on fluid production and arc magmatism. *Philos. T. R. Soc. A 335*(1638), 341–353.

Peacock, S. M., T. Rushmer, and A. B. Thompson (1994). Partial melting of subducting oceanic crust. *Earth Planet. Sc. Lett. 121*, 227–224.

Pearce, J. and D. Peate (1995). Tectonic implications of the composition of volcanic arc magmas. *Ann. Rev. Earth Planet. Sci. 23*, 251–285.

Pelletier, D., A. Fortin, and R. Camarero (1989). Are FEM solutions of incompressible flows really incompressible? (or how simple flows can cause headaches!). *Int. J. Numer. Meth. Fl. 9*, 99–112.

Philpotts, A. R. and J. J. Ague (2009). *Principles of Igneous and Metamorphic Petrology.* Cambridge University Press.

Phipps Morgan, J. (1997). The generation of a compositional lithosphere by mid-ocean ridge melting and its effect on subsequent off-axis hotspot upwelling and melting. *Earth Planet. Sc. Lett. 146*, 213–232.

Phipps Morgan, J. (2001). Thermodynamics of pressure release melting of a veined plum pudding mantles. *Geochem. Geophy. Geosys. 2*, 2000GC000049.

Phipps Morgan, J. and D. W. Forsyth (1988). Three-dimensional flow and temperature perturbations due to a transform offset: Effects on oceanic crustal and upper mantle structure. *J. Geophys. Res. 93*(B4), 2955–2966.

Phipps Morgan, J., J. Hasenclever, M. Hort, L. Rüpke, and E. M. Parmentier (2007). On subducting slab entrainment of buoyant asthenosphere. *Terra Nova 19*(3), 167–173.

Phipps Morgan, J. and W. J. Morgan (1999). Two-stage melting and the geochemical evolution of the mantle: a recipe for mantle plum-pudding. *Earth Planet. Sc. Lett. 170*, 215–239.

Plank, T. and C. H. Langmuir (1988). An evaluation of the global variations in the major element chemistry of arc basalts. *Earth Planet. Sc. Lett. 90*, 349–370.

Powell, M. J. D. (1969). A method for nonlinear constraints in minimization problems, in: *Optimization* (R. Fletcher (ed.)), Academic Press, London.

Press, W., S. Teukolsky, W. Vetterling, and B. Flannery (1992). Solution of Linear Algebraic Equations, in: *Numerical Recipes*, Cambridge University Press.

Rubin, K., I. V. der Zander, M. Smith, and E. Bergmanis (2005). Minimum speed limit for ocean ridge magmatism from 210Pb–226Ra–230Th disequilibria. *Nature 437*(22), 534–538. doi: 10.1038/nature03993.

Rüpke, L., J. M. J Phipps Morgan, M. Hort, and J. Connolly (2004). Serpentine and the subduction zone water cycle. *Earth Planet. Sc. Lett. 223*(1-2), 17–34.

Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems.* SIAM.

Saad, Y. and M. H. Schultz (1986). GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Comput. 7*(3), 856–869.

Sani, R. L., P. M. Gresho, R. L. Lee, and D. F. Griffiths (1981). The cause and cure (?) of the spurious pressures generated by certain FEM solutions of the incompressible Navier-Stokes equations: part 1. *Int. J. Numer. Meth. Fl. 1*, 17–43.

Sani, R. L., P. M. Gresho, R. L. Lee, D. F. Griffiths, and M. S. Engelman (1981). The cause and cure (!) of the spurious pressures generated by certain FEM solutions of the incompressible Navier-Stokes equations: part 2. *Int. J. Numer. Meth. Fl. 1*, 171–204.

Schilling, J., G. Thompson, R. Kingsley, and S. Humphris (1985). Hotspot-migrating ridge interaction in the South Atlantic. *Nature 313*, 187–191.

Schmalzl, J. and G. Houseman (1996). Mixing in vigorous, time-dependent three-dimensional convection and application to earth's mantle. *J. Geophys. Res. 101*(B10), 21847–218858.

Schmeling, H. (1987). On the relation between initial conditions and late stages of Rayleigh-Taylor instabilities. *Tectonophysics 133*(1-2), 65–80.

Schubert, G., D. L. Turcotte, and P. Olson (2001). *Mantle Convection in the Earth and Planets.* Cambridge University Press.

Shaw, D. M. (1970). Trace element fractionation during anatexis. *Geochim. Cosmochim. Ac. 34*, 237–243. doi: 10.1016/0016-7037(70)90009-8.

Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain. Technical Report CMU-CS-94-125, School of Computer Science, Carnegie Mellon University, Pittsburgh.

Shi, C. and J. Phipps Morgan (2010). Accurate cubic-spline-like interpolation on unstructured 2D and 3D meshes. *in prep.*

Small, C. (1998). Global Systematics of Mid-Ocean Ridge Morphology, in: *Faulting and Magmatism at Mid-Ocean Ridges* (W. R. Buck, P. T. Delaney, J. A. Karson, and Y. Lagabrielle (eds.)), *Geophysical Monograph, 106*, American Geophysical Union.

Smith, B. F., P. E. Bjorstad, and W. D. Gropp (2004). *Domain Decomposition.* Cambridge University Press.

Smolarkiewicz, P. K. (1984). A fully multidimensional positive definite advection transport algorithm with small implicit diffusion. *J. Comput. Phys. 54*, 325–362.

Sparks, D. W. and E. M. Parmentier (1993). The structure of three-dimensional convection beneath oceanic spreading centres. *Geophys. J. Int. 112*, 81–91.

Stemmer, K., H. Harder, and U. Hansen (2006). A new method to simulate convection with strongly temperature-and pressure-dependent viscosity in a spherical shell: Applications to the earth's mantle. *Phys. Earth Planet. Int. 157*, 223–249.

Stolper, E. and S. Newman (1994). The role of water in the petrogenesis of Marina trough magmas. *Earth Planet. Sc. Lett. 121*, 293–325.

Stracke, A., B. Bourdon, and D. McKenzie (2006). Melt extraction in the earth's mantle: Constraints from U-Th-Pa-Ra studies in oceanic basalts. *Earth Planet. Sc. Lett. 244*, 97–112.

Takei, Y. and B. Holtzman (2009a). Viscous constitutive relations of solid-liquid composites in terms of grain boundary contiguity: 1. grain boundary diffusion control model. *J. Geophys. Res. 114*(B6), B06205. doi: 10.1029/2008JB005850.

Takei, Y. and B. Holtzman (2009b). Viscous constitutive relations of solid-liquid composites in terms of grain boundary contiguity: 2. compositional model for small melt fractions. *J. Geophys. Res. 114*(B6), B06206. doi: 10.1029/2008JB005851.

Tamura, Y., Y. Tatsumi, D. Zhao, Y. Kido, and H. Shukuno (2002). Hot fingers in the mantle wedge: new insights into magma genesis in subduction zones. *Earth Planet. Sc. Lett. 197*, 105–116.

Turcotte, D. L. and G. Schubert (2002). *Geodynamics.* Cambridge University Press.

van der Vorst, H. A. (1992). Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Comput. 13*(2), 631–644.

van der Vorst, H. A. (2003). *Iterative Krylov Methods for Large Linear Systems.* Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press.

van Keken, P. (2003). The structure and dynamics of the mantle wedge. *Earth Planet. Sc. Lett. 215*, 323–338.

van Keken, P. and S. Zhong (1999). Mixing in a 3D spherical model of present-day mantle convection. *Earth Planet. Sc. Lett. 171*, 533–547.

Verführt, R. (1984). A combined conjugate gradient–multigrid algorithm for the numerical solution of the Stokes problem. *IMA J. Numer. Anal. 4*, 441–455.

Wilson, J. T. (1963). Hypothesis of earth's behavior. *Nature 198*, 925–929.

Yamamoto, M. and J. Phipps Morgan (2009). North arch volcanic fields near Hawaii are evidence favouring the restite-root hypothesis for the origin of hotspot swells. *Terra Nova 21*(6), 452–466. doi: 10.1111/j.1365-3121.2009.00902.x.

Zhao, D., Z. Wang, N. Umino, and A. Hasegawa (2009). Mapping the mantle wedge and interplate thrust zone of the northeast Japan arc. *Tectonophysics 467*(1-4), 89–106. doi: 10.1016/j.tecto.2008.12.017.

Zhu, G., T. V. Gerya, D. A. Yuen, S. Honda, T. Yoshida, and J. A. D. Connolly (2009). Three-dimensional dynamics of hydrous thermal-chemical plumes in oceanic subduction zones. *Geochem. Geophy. Geosys. 10*(11), Q11006. doi: 10.1029/2009GC002625.

Zienkiewicz, O. C. and R. L. Taylor (1989). *The Finite Element Method, Vol. 1, The Basis*. Butterworth-Heinemann.

Danksagung

Während der Anfertigung dieser Arbeit haben mich viele Freunde und Kollegen unterstützt, denen ich von Herzen danken möchte.

Ganz besonders möchte ich meinen beiden Betreuern Matthias Hort und Jason Phipps Morgan danken. Beide haben sehr viel Zeit und Energie darauf verwendet, um mit mir neue Ideen und Strategien zu diskutieren. Diese zahlreichen fruchtbaren Diskussionen haben wesentlichen Anteil an der Vollendung der Arbeit in der vorliegenden Form. Aber auch in zwischenmenschlicher Hinsicht habe ich euch beiden sehr viel zu verdanken. Ihr habt mich motiviert, getröstet, mir auf die Schulter geklopft — was auch immer nötig war. Vielen Dank an euch beide!

Unschätzbaren Anteil an der Vollendung dieser Arbeit hat Lea Scharff, die nicht müde wurde mir Energie zu schenken und Mut zuzusprechen, wenn mal wieder etwas nicht nach Plan lief. Auch an erfolgreichen Käferjagden war sie öfters beteiligt. Vielen, vielen Dank dafür und für die ca. 1000 Korrekturvorschläge — und ich hoffe ich kann mich bald revanchieren.

Grosser Dank gebührt ebenfalls meinen Eltern, Joachim und Ingeborg, die mich von Anfang an unterstützt haben, ganz gleich welche Wege ich beruflich eingeschlagen habe.

Freunde, die ich während meines Aufenthaltes an der Cornell Universität kennenlernt habe, haben die Zeit dort zu einem unvergesslichen Erlebnis gemacht. Ganz besonders möchte ich Greg Kirkpatrick und Chao Shi für die fachlichen Diskussionen, die Unterstützung bei Hard- und Softwareproblemen und für die entspannenden Stunden in Ale House und Chapter House bedanken.

Vielen Dank auch an Gabriela Depine, die mir bei der Wohnungssuche in Ithaca und bei anderen „Problemen“ sehr geholfen hat. Come back to science!!!!

Bei den Mitgliedern der Arbeitsgruppe *Vulkanologie* sowie dem gesamten Institut für Geophysik der Universität Hamburg möchte ich für fachlichen Input, spannende Kickerspiele und weitere Unterstützungen bedanken.

Lars Rüpke vom Leibniz Institut für Meereswissenschaften IFM-Geomar möchte ich für die Möglichkeit danken, in seiner Arbeitsgruppe eine interessante geodynamische Fragestellung bearbeiten zu können. Ihm und Karthik Iyer möchte ich für Anregungen und die Einführung in Subduktionszonen-Prozesse danken.

Boris Kaus ist in gewisser Weise verantwortlich dafür, dass ich die Arbeit an einem neuen numerischen Code begonnen habe. Auch wenn ich zwischenzeitlich unzählige Male geflucht habe..., danke für das interessante Gespräch auf dem Marktplatz in Erice und weitere Anregungen auf späteren Tagungen.

Und ohne meine Fahrräder hätte ich’s wohl auch nicht geschafft...