

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the author's prior consent.

The dynamics of dense water cascades: from laboratory scales to the Arctic Ocean.



Fred Wobus

School of Marine Science and Engineering

University of Plymouth

A thesis submitted to the University of Plymouth in partial
fulfilment of the requirements for the degree of:

Doctor of Philosophy

August 2013

Abstract

The dynamics of dense water cascades: from laboratory scales to the Arctic Ocean.

Fred Wobus

The sinking of dense shelf waters down the continental slope (or “cascading”) contributes to oceanic water mass formation and carbon cycling. Cascading is therefore of significant importance for the global overturning circulation and thus climate. The occurrence of cascades is highly intermittent in space and time and observations of the process itself (rather than its outcomes) are scarce. Global climate models do not typically resolve cascading owing to numerical challenges concerning turbulence, mixing and faithful representation of bottom boundary layer dynamics. This work was motivated by the need to improve the representation of cascading in numerical ocean circulation models. Typical 3-D hydrostatic ocean circulation models are employed in a series of numerical experiments to investigate the process of dense water cascading in both idealised and realistic model setups.

Cascading on steep bottom topography is modelled using POLCOMS, a 3-D ocean circulation model using a terrain-following s -coordinate system. The model setup is based on a laboratory experiment of a continuous dense water flow from a central source on a conical slope in a rotating tank. The descent of the dense flow as characterised by the length of the plume as a function of time is studied for a range of parameters, such as density difference, speed of rotation, flow rate and (in the model) diffusivity and viscosity. Very good agreement between the model and the laboratory results is shown in dimensional and non-dimensional variables. It is confirmed that a hydrostatic model is capable of reproducing the essential physics of cascading on a very steep slope if the model correctly resolves velocity veering in the bottom boundary layer. Experiments changing the height of the bottom Ekman layer (by changing viscosity) and modifying the plume from a 2-layer system to a stratified regime (by enhancing diapycnal diffusion) confirm previous theories, demonstrate their limitations and offer new insights into the dynamics of cascading outside of the controlled laboratory conditions.

In further numerical experiments, the idealised geometry of the conical slope is re-

tained but up-scaled to oceanic dimensions. The NEMO-SHELF model is used to study the fate of a dense water plume of similar properties to the overflow of brine-enriched shelf waters from the Storfjorden in Svalbard. The overflow plume, resulting from sea ice formation in the Storfjorden polynya, cascades into the ambient stratification resembling the predominant water masses of Fram Strait. At intermediate depths between 200-500 m the plume encounters a layer of warm, saline Atlantic Water. In some years the plume ‘pierces’ the Atlantic Layer and sinks into the deep Fram Strait while in other years it remains ‘arrested’ at Atlantic Layer depths. It has been unclear what parameters control whether the plume pierces the Atlantic Layer or not. In a series of experiments we vary the salinity ‘ S ’ and the flow rate ‘ Q ’ of the simulated Storfjorden overflow to investigate both strong and weak cascading conditions. Results show that the cascading regime (piercing, arrested or ‘shaving’ - an intermediate case) can be predicted from the initial values of S and Q . In those model experiments where the initial density of the overflow water is considerably greater than of the deepest ambient water mass we find that a cascade with high initial S does not necessarily reach the bottom if Q is low. Conversely, cascades with an initial density just slightly higher than the deepest ambient layer may flow to the bottom if the flow rate Q is high. A functional relationship between S/Q and the final depth level of plume waters is explained by the flux of potential energy (arising from the introduction of dense water at shallow depth) which, in our idealised setting, represents the only energy source for downslope descent and mixing.

Lastly, the influence of tides on the propagation of a dense water plume is investigated using a regional NEMO-SHELF model with realistic bathymetry, atmospheric forcing, open boundary conditions and tides. The model has 3 km horizontal resolution and 50 vertical levels in the s_h -coordinate system which is specially designed to resolve bottom boundary layer processes. Tidal effects are isolated by comparing results from model runs with and without tides. A hotspot of tidally-induced horizontal diffusion leading to the lateral dispersion of the plume is identified at the southernmost headland of Spitsbergen which is in close proximity to the plume path. As a result the lighter fractions in the diluted upper layer of the plume are drawn into the shallow coastal current that carries Storfjorden water onto the Western Svalbard Shelf, while the dense bottom layer continues to sink down the slope. This bifurcation of the plume into a diluted shelf branch and a dense downslope branch is enhanced by tidally-induced shear dispersion at the headland. Tidal effects at the headland are shown to cause a net reduction in the downslope flux of Storfjorden water into deep Fram Strait. This finding contrasts

previous results from observations of a dense plume on a different shelf without abrupt topography. The dispersive mechanism which is induced by the tides is identified as a mechanism by which tides may cause a relative reduction in downslope transport, thus adding to existing understanding of tidal effects on dense water overflows.

Contents

Abstract	v
Acknowledgements	xix
Author's declaration	xxi
1 Introduction	1
1.1 Background	1
1.2 Cascading in the Arctic	3
1.3 Modelling cascading	4
1.4 The Storfjorden overflow	5
1.5 Thesis framework	7
2 Methods	13
2.1 The numerical models	13
2.2 The POLCOMS model	15
2.3 The NEMO model	18
3 Numerical simulations of dense water cascading on a steep slope	29
3.1 Introduction	29
3.2 Background	29
3.3 POLCOMS model setup	34
3.4 Experiment design	36
3.5 Results	39
3.6 Discussion	53
3.7 Conclusions	57
4 The piercing of the Atlantic Layer by an Arctic shelf water cascade in an idealised study inspired by the Storfjorden overflow	59
4.1 Introduction	59

4.2	NEMO model setup	60
4.3	Results and Discussion	64
4.4	Conclusions	83
5	Tidally-induced lateral dispersion of the Storfjorden overflow plume	85
5.1	Introduction	85
5.2	NEMO model setup	86
5.3	Results	95
5.4	Discussion	106
5.5	Conclusions	111
6	Summary and further work	113
6.1	Summary of conclusions	113
6.2	Recommendations for further work	115
A	POLCOMS no-slip bottom boundary condition	119
B	NEMO no-slip bottom boundary condition	121
C	NEMO lateral diffusion operator rotation	127
C.1	Test runs without injection	127
C.2	Test runs with injection	128
D	Guide to compiling NEMO and NetCDF on 32- and 64-bit Windows	131
D.1	Software versions used	132
D.2	The Visual Studio solution and projects	134
D.3	Configure the projects (Compiler options etc.)	147
D.4	Why do NEMO and NetCDF not compile ‘out of the box’?	164
D.5	Compilation for 64-bit platforms	167
D.6	NetCDF file handling on Windows	171
E	Guide to setting up a regional NEMO model on the HPC cluster	175
E.1	Windows software installation	176
E.2	Installation on the cluster	181
E.3	Pre-processing	191

E.4	NEMO configuration	203
E.5	Compile-time configuration	203
E.6	Run-time configuration	206
E.7	How to run NEMO	210
F	Sequence of runs in NEMO	219
F.1	Mesh_mask run	222
F.2	Geostrophic adjustment run	222
F.3	Full forcing non-tidal spin-up run	223
F.4	Full forcing tidal spin-up run	224
	List of references.	227

List of Figures

1.1	Schematic of cascading in the Arctic Ocean	3
1.2	Map of the Storfjorden in the Svalbard archipelago	6
1.3	Comparison of exploratory vs. simulation modelling approaches	8
2.1	Vertical discretisation in the s_h -coordinate system	20
2.2	Cross-section of buoyancy frequency N^2	27
3.1	Schematic of the laboratory setup for the SZ97 experiments	30
3.2	Diagram of the vertical s -coordinate system	35
3.3	Schematics of the modelled flow and the measured variables	38
3.4	Results of a series of model runs	41
3.5	Vertical profiles of velocity and salinity	43
3.6	Downslope progression of the plume as a function of time	44
3.7	Series of model runs varying the Ekman depth	46
3.8	Series of model runs varying the vertical diffusivity	49
3.9	Velocity and salinity profiles for runs with increased κ	50
3.10	Plume cross-sections for constant κ and varying v	52
4.1	Model domain with conical slope at its centre	61
4.2	Cross-section plot and profiles from a model run with strong cascading .	64
4.3	Cross-section of tracer concentration showing 3 regimes	66
4.4	Downslope progression of the plume edge over time	69
4.5	Correlation of alongslope vs. downslope velocity	71
4.6	Downslope evolution of θ - S properties at the bottom	73
4.7	Temperature maximum at the bottom plotted against S and Q	75
4.8	θ - S schematic for a dense plume, from Rudels and Quadfasel (1991) . .	77
4.9	Percentage of tracer within 2 depth ranges plotted against S and Q . . .	78
4.10	Potential energy (PE) relative to the PE_{start} over time	80
4.11	Percentage of passive tracer within 2 depth ranges plotted against S and ΔPE	81

4.12	Max. penetration depth Z_{PTRC} plotted against ΔPE	82
5.1	Bathymetry of the model domain, plume path and major ocean currents	87
5.2	Tidal elevation from the TPXO7.2 tidal model	92
5.3	Comparison of cross section C in Fer et al. (2004) with model output	96
5.4	Comparison of cross section E in Fer et al. (2003) with model output	98
5.5	Map of bottom concentration of passive tracer at end of model run	100
5.6	Cross-section of overflow tracer in Storfjordrenna	101
5.7	Location of <i>shelf</i> and <i>slope</i> cross-sections used for flux calculation	103
5.8	Horizontal diffusivity coefficient κ_{hor} at the bottom	104
5.9	Time series of parameters averaged around the Sørkapp headland	105
5.10	Schematic of tidal dispersion of the Storfjorden overflow plume	110
B.1	Imposing a zero velocity condition at the sea bed	123
C.1	Rotation of the lateral diffusion operator without injection	127
C.2	Rotation of the lateral diffusion operator with injection	128
C.3	Rotation of the lateral diffusion operator with injection, replotted with a flat slope	129
D.1	Initial folder structure	136
D.2	Create a new Visual Studio Project	137
D.3	Drag and drop source files to populate the project.	138
D.4	Create a new Fortran project.	138
D.5	New Fortran project dialog.	139
D.6	Edit the solution manually	139
D.7	The header files are excluded from the compilation.	141
D.8	Add a new Win32 project.	142
D.9	Win32 application settings.	143
D.10	Manually modified sln-file	143
D.11	Completed project setup with all source and header files.	147
D.12	General settings in the netcdf_lib project.	148
D.13	C/C++ → General settings in the netcdf_lib project.	149
D.14	C/C++ → Preprocessor settings in the netcdf_lib project.	150

D.15 C/C++ → Code Generation settings in the netcdf_lib project.	151
D.16 General settings in the libnetcdf_f90 project.	152
D.17 Fortran → Preprocessor settings in the libnetcdf_f90 project.	153
D.18 Fortran → Data settings in the libnetcdf_f90 project.	153
D.19 Fortran → Floating Point settings in the libnetcdf_f90 project.	154
D.20 Fortran → External Procedures settings in the libnetcdf_f90 project. . .	154
D.21 Fortran → Libraries settings in the libnetcdf_f90 project.	155
D.22 General settings in the NEMO_v3_2 project.	156
D.23 Fortran → Preprocessor settings in the NEMO_v3_2 project.	157
D.24 Fortran → Data settings in the NEMO_v3_2 project.	157
D.25 Fortran → Floating Point settings in the NEMO_v3_2 project.	158
D.26 Fortran → External Procedures settings in the NEMO_v3_2 project. . .	158
D.27 Fortran → Libraries settings in the NEMO_v3_2 project.	159
D.28 Linker → General settings in the NEMO_v3_2 project.	160
D.29 Linker → Input settings in the NEMO_v3_2 project.	160
D.30 Linker → Manifest File settings in the NEMO_v3_2 project.	161
D.31 Linker → Debug settings in the NEMO_v3_2 project.	161
D.32 Linker → System settings in the NEMO_v3_2 project.	162
D.33 Compiling the NEMO_v3_2 project.	162
D.34 Files after compiling the NEMO_v3_2 project.	163
D.35 Screenshot of Dependency Walker	163
D.36 By default only the 32-bit platform is available for compilation.	167
D.37 Creating a new platform configuration.	168
D.38 Dialog for the creation of a new solution platform.	169
D.39 Properties of a Windows disk showing its file system (circled).	172
E.1 The X Window server has started	178
E.2 Desktop shortcuts after successful installation	179
E.3 Login dialog of the Bitvise SSH Client.	180
E.4 Recommended SFTP settings in the Bitvise SSH Client.	181
E.5 Split screen SFTP window in the Bitvise SSH Client.	181
E.6 Visual Studio 2008 solution for compilation of nocs_weights	190

E.7	Final model bathymetry derived from IBCAO	194
E.8	Location map of the regions of positive and negative river flux	199
E.9	Flow rate profile for the dense water injection.	201
F.1	Schematic flow diagram of sequence of model runs	221

List of Tables

2.1	Parameters for the $k - \varepsilon$ scheme in the GLS turbulence closure model . .	23
3.1	Summary of model parameters	36
4.1	Characteristics of the plume in the ‘arrested’ and ‘piercing’ regime . . .	68
D.1	Storage size of fundamental types in MS Visual Studio	171
E.1	Types of data fields in the Drakkar Forcing Set	195

Acknowledgements

This work was part of a wider study, to which many other people contributed. In the following I would like to acknowledge the inspiration, assistance and encouragement that I have benefitted from over the years.

My supervisor Georgy I. Shapiro (University of Plymouth) merits the biggest thank of all. This PhD would not have been possible without his encouragement, advice, endless hours of discussions and strategically timed words of motivation. I also thank my co-supervisors John M. Huthnance and Miguel Angel Morales Maqueda (National Oceanography Centre, Liverpool) for their assistance and constructive guidance along the way.

Many colleagues, collaborators and friends have shared their ideas, data and program codes. Andrei G. Zatsepin (Shirshov Institute of Oceanology, Moscow, Russia) kindly provided video footage of the laboratory experiments (Fig. 3.4e). The National Centre for Ocean Forecasting (NCOF) provided the NEMO-SHELF code (section 4.2.1). Dave Storkey (MetOffice, Exeter) patiently introduced me and my colleagues to the NEMO code. Hedong Liu and Jason Holt (National Oceanography Centre, Liverpool) contributed the code for the vertical PPM advection and the Pressure Jacobian horizontal pressure gradient schemes (section 4.2.1). Hedong Liu also assisted with the coding of the no-slip bottom boundary condition in NEMO. Vladimir V. Ivanov (AARI, St. Petersburg, Russia) freely shared his ideas and source code that led to the development of the s_h vertical coordinate system (section 2.3.1). Chapter 5 would not have been possible in its presented form without Yevgeny Aksenov (National Oceanography Centre, Southampton) who provided the global model data used for initial and open boundary conditions (section 5.2.4). With help from Maria Luneva and Clare O'Neill (National Oceanography Centre, Liverpool) I was able to run NEMO on a super-computing cluster. Maria Luneva also contributed the source code for the Smagorinsky lateral diffusion scheme (section 5.2.3) and kindly assisted with setting up the meteorological forcing data (section 5.2.5). James Harle (National Oceanography Centre, Liverpool) kindly shared his scripts to set up the tidal forcing data (section 5.2.5). Anna Akimova (Thünne-Institute of Sea Fisheries, Hamburg, Germany) provided the CTD station locations for Fig. 5.5b. Constructive suggestions by Ragnheid Skogseth (UNIS, Longyearbyen, Norway) helped shape some of the ideas in section 5.4.2. The \LaTeX template for

this document was kindly provided by Martin Coath (University of Plymouth). This work also benefitted greatly from the comments by anonymous reviewers who helped to improve the manuscripts that were submitted as part of this project.

I would also like to thank the following people (in no particular order) for great discussions and inspiration: Ilker Fer (Geophysical Institute, University of Bergen, Norway), Ursula Schauer (Alfred Wegener Institute, Germany), Achim Wirth (LEGI, Grenoble, France), Sonya Legg (GFDL, Princeton, USA), Anna Wåhlin (University of Gothenburg, Sweden). Special mention goes to Hugh VENABLES (British Antarctic Survey, Cambridge) who took me under his wing on the voyage of a lifetime to the Southern Ocean where I had the chance to experience the life of an oceanographer at sea.

This work was partly funded by NERC's Core Research Programme Oceans 2025, the EU FP7 MyOcean/MyOcean2 project and a University of Plymouth PhD studentship.

Dedication Finally, I am deeply indebted to my parents who never stopped believing in me. This PhD would not have been possible without your support. Anna & Ulrich – Thank you and Vielen Dank. This is for you!

Fred Wobus

Plymouth, Friday 23rd August, 2013

Post scriptum In April 2007 I came across a National Geographic article which laid bare humanity's destruction of the world's oceans in no uncertain terms. Brian Skerry's disturbing photographs opened my eyes to a global crisis that had unfolded during my adult lifetime. They formed part of my motivation to study the oceans and pursue this PhD. The numerical modelling of dense water cascades will not in itself save a single fish, clean up oil spills or restore anoxic dead zones. But I sincerely hope to contribute to the preservation of the oceans by furthering our understanding of the seas.

Author's declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award.

The scientific work was frequently presented at relevant scientific conferences and seminars. Two papers have been published in refereed journals. At the time of completion of this thesis, a third paper has been published as a discussion paper, but it has not cleared the peer review stage yet.

Author contribution

The scientific material presented in this thesis was developed as part of multi-author publications and parts of it have been a collaborative effort (see previous *Acknowledgements* section).

Paper 1 (JMR) The design of the experiments was partly based upon previous laboratory studies, while those aspects going beyond the laboratory setup were initially suggested by the supervisors and further developed by the author. The author conducted the model experiments, produced the results and figures. Several modifications to the model code were made under guidance from supervisors and staff at NOCL. The paper was written by the author with contributions from supervisors (some of the Introduction, as well as small parts of the Methods and Discussion).

Paper 2 (OM) The author set up the numerical model and wrote the code for the s_h -coordinate system and the modified rotation of the lateral intrusion operator. The investigation of cascading energy was stimulated by the supervisors and further developed by the author, who designed the S-Q parameter space and designed the series of experiments accordingly. The paper was written by the author with corrections and comments offered by the supervisors. The author also produced the figures, unless otherwise stated.

Paper 3 (OSD) The author set up the numerical model and designed the numerical experiments. The idea to investigate tides was developed in collaboration with the supervisors. The paper was written by the author with some comments offered by the supervisors and co-authors. The author produced all of the figures.

Signed: _____

Date: _____

Word count for the main body of this thesis: ~28,800

Total word count: ~64,500

Publications :

- (1) Wobus, F., Shapiro, G. I., Maqueda, M. A. M. and Huthnance, J. M. *Numerical simulations of dense water cascading on a steep slope*. Journal of Marine Research. **69(2-3), 391–415, 2011**. doi:10.1357/002224011798765268
- (2) Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. *The piercing of the Atlantic Layer by an Arctic shelf water cascade in an idealised study inspired by the Storfjorden overflow in Svalbard*. Ocean Modelling. **2013, in press**. doi:10.1016/j.ocemod.2013.03.003
- (3) Wobus, F., Shapiro, G. I., Huthnance, J. M., Maqueda, M. A. M. and Aksenov Y. *Tidally-induced lateral dispersion of the Storfjorden overflow plume*. Ocean Science Discussions. **10, 691-726, 2013**. doi:10.5194/osd-10-691-2013

Conference presentations, posters and seminars :

2013:

European Geosciences Union General Assembly, 07 – 12 April 2013. *Vienna, Austria*. Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M., Maqueda, M. A. M. and Aksenov Y. **Tidally-induced dispersion of the Storfjorden overflow plume onto the West Svalbard Shelf**. Geophysical Research Abstracts Vol. 15, EGU2013-4912, 2013

2012:

Bangor Polar Symposium, 8th December 2012. *School of Ocean Sciences, Bangor University*. Poster presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M., Maqueda, M. A. M. and Aksenov Y. **The influence of tides on the Storfjorden overflow**.

APECS Svalbard Workshop, 08th October 2012. *Online webinar*. Invited talk: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **The Storfjorden overflow – Dense water cascading in the Arctic Ocean**.

15th Biennial Challenger Conference for Marine Science, 3 – 6 September 2012. *University of East Anglia, Norwich*. Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **An Idealised Modelling Study of an Arctic Dense Water Cascade Piercing the Atlantic Layer**.

NEMO Users' Meeting, 23rd May 2012. *MetOffice, Exeter*. Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **An idealised modelling study of dense water cascading penetrating into ambient stratification**.

NOC seminar series, 16th May 2012. *National Oceanography Centre, Liverpool*. Invited talk: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **Dense water cascading in the Arctic Ocean – overview and idealised modelling results**.

European Geosciences Union General Assembly, 22 – 27 April 2012. *Vienna, Austria*. Poster presentation: Wobus, F., Shapiro, G. I., Maqueda, M. A. M. and

Huthnance, J. M. **Numerical simulations of dense water cascading on a steep slope.** Geophysical Research Abstracts. Vol. 14, EGU2012-4555, 2012

European Geosciences Union General Assembly, 22 – 27 April 2012. *Vienna, Austria.* Poster presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **The piercing of the Atlantic Layer by a dense water cascade in an idealised modelling study inspired by the Storfjorden overflow.** Geophysical Research Abstracts. Vol. 14, EGU2012-2816, 2012

2011:

Blue Horizons – 4th Annual Plymouth Marine Science Education Fund Conference, 14th December 2011. *Plymouth Marine Laboratory, Plymouth.* Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **Dense water cascading in the Arctic Ocean.**

CCOSE seminar series, 5th October 2011. *University of Plymouth.* Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **Dense water cascading off the continental shelf – overview and recent modelling results.**

UK Arctic Science Conference, 14 – 16 September 2011. *University of Leeds.* Oral presentation: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **Modelling density-stratified cascades on a steep slope.**

BAS seminar series, 3rd August 2011. *British Antarctic Survey, Cambridge.* Invited talk: Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M. **Dense water cascading off the continental shelf – overview and recent modelling results.**

European Geosciences Union General Assembly, 03 – 08 April 2011. *Vienna, Austria.* Oral presentation: Wobus, F., Shapiro, G. I., Maqueda, M. A. M. and Huthnance, J. M. **Modelling of shelf water cascades on the continental slope.** Geophysical Research Abstracts. Vol. 13, EGU2011-3027, 2011

UK Young Coastal Scientists and Engineers Conference, 30 – 31 March 2011.
National Oceanography Centre, Liverpool. Poster presentation: Wobus, F., Shapiro,
G. I., Maqueda, M. A. M. and Huthnance, J. M. **Modelling density-stratified
cascades on a steep slope.**

2010:

Spirit of Discovery – 3rd Annual Student-led Conference, 20th December 2010.
University of Plymouth. Oral presentation: Wobus, F., Shapiro, G. I., Maqueda,
M. A. M. and Huthnance, J. M. **Numerical simulations of dense water cascading
on a steep slope.**

14th Biennial Challenger Conference for Marine Science, 20th December 2010.
National Oceanography Centre, Southampton Oral presentation: Wobus, F., Shapiro,
G. I., Maqueda, M. A. M. and Huthnance, J. M. **Simulating laboratory experiments
of dense water cascades: a sensitivity study.**

Chapter 1

Introduction

1.1 Background

Spatial variations in temperature and salinity resulting from cooling, evaporation or surface-layer freezing over the shallow regions of continental shelves result in the formation of dense shelf waters which subsequently sink or cascade down the continental slope to a greater depth (Ivanov et al., 2004). During its descent, the plume of dense water is modified by mixing and entrainment, and detaches off the slope when reaching its neutral buoyancy level (Lane-Serff, 2009).

This process, termed ‘cascading’ by Cooper and Vaux (1949), transports shelf waters to the deep ocean. It thus contributes to ocean ventilation and water mass formation, and hence ocean circulation (Killworth, 1983), notably in the Antarctic (e.g. Baines and Condie, 1998; Bergamasco et al., 2004; Gordon et al., 2004; Darelius et al., 2009; Orsi and Wiederwohl, 2009) and in the Arctic (e.g. Aagaard et al., 1981; Rudels and Quadfasel, 1991; Schauer, 1995; Ivanov and Golovin, 2007). In addition to cascading, the high latitudes are key regions for ocean circulation through open-ocean convection (Marshall and Schott, 1999) as well as the upward mixing of abyssal waters towards the surface which partly closes the overturning budget (Heywood et al., 2002).

Dense water flows (or ‘cascades’) may also contribute to the export of carbon from shelf seas as a component of the ‘carbon pump’ (e.g. Holt et al., 2009). This is illustrated by the appearance of chlorophyll at 500 m down the slope as the clearest evidence of the Malin shelf cascade (Hill et al., 1998), the export of particulate matter down the

slope of northern Biscay (Wollast and Chou, 2001) and preferentially down canyons in the western Mediterranean (e.g. Sánchez-Vidal et al., 2008; Canals et al., 2009).

When frictional forces are relatively small, a body of dense water is in geostrophic balance and flows horizontally along density fronts that separate dense shelf water from lighter off shelf waters (Gill, 1982). A geostrophically adjusted dense water plume on the inclined topography of the continental slope will follow the contours of constant depth and move along the slope (Nof, 1983). Slopes and geostrophy thus combine to inhibit shelf-ocean exchange (Huthnance, 1995) and there is considerable interest in processes that break the geostrophic constraint to facilitate cross-slope flow.

Cascading is one such process on account of the importance of (turbulent) friction (Shapiro et al., 2003). There has been significant progress in describing its physical properties (Griffiths, 1986). A classification of cascades is provided by Shapiro et al. (2003). Shapiro and Hill (1997) developed a $1\frac{1}{2}$ -layer model with bottom friction, interfacial Ekman veering and entrainment (see Section 3.2.2); it lies between ‘stream-tube’ models (Smith, 1975) and full 3-D models.

In the Arctic Ocean there is growing understanding of the influence of cascading on the formation and maintenance of the halocline and on the global overturning circulation (Aagaard et al., 1981; Melling and Lewis, 1982; Rudels et al., 1996; Steele and Boyd, 1998; Rudels et al., 1999; Carmack, 2000; Furevik et al., 2007; Rudels, 2009; Turner, 2010). Despite some estimates (e.g. Ivanov et al., 2004; Rudels, 2012, and references therein), the role of cascading in the Arctic Ocean remains largely unquantified. Regarding the modelling of cascading, Furevik et al. (2007) state that “*Shelf processes and down slope sinking of waters is generally poorly described in climate models, and there is therefore little knowledge on how this will change in a future climate.*” The aim to improve the representation of cascading in ocean models has partly motivated this thesis.

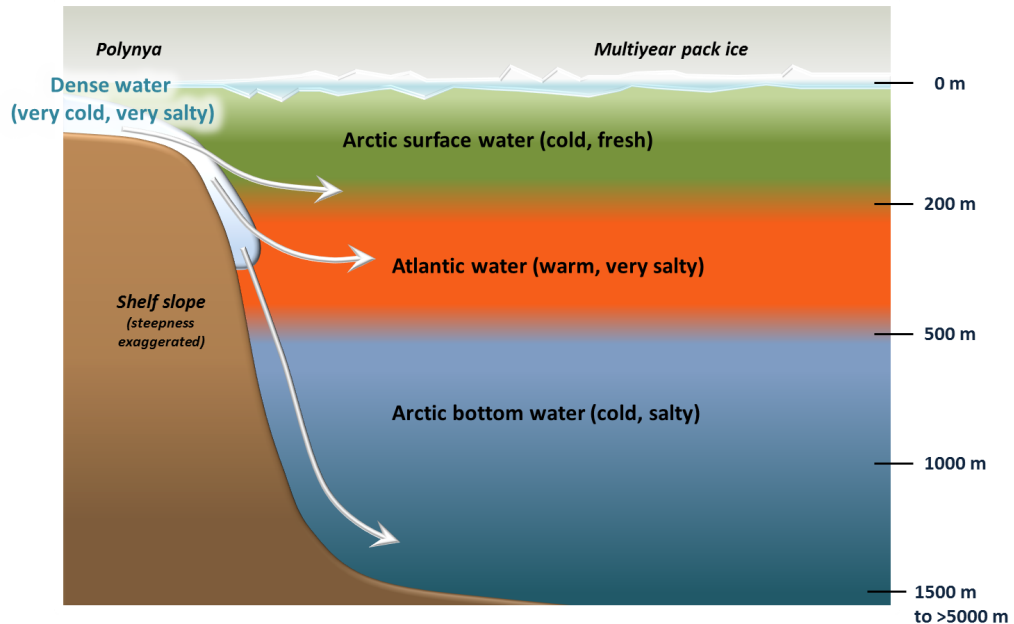


Figure 1.1: Schematic of cascading in the Arctic Ocean. Dense waters formed in shelf polynyas flow off the continental shelf and intrude into the ambient stratification shown here as idealised layers of the main Arctic water masses (after Aagaard et al., 1985; Aagaard and Carmack, 1989). Inspired by an illustration by Jayne Doucette (WHOI)¹.

1.2 Cascading in the Arctic

Winter cooling and sea ice formation form large amounts of brine-enriched shelf water over the vast shelves in the Arctic Ocean (Aagaard et al., 1985; Cavalieri and Martin, 1994; Rudels, 1995). Plumes of dense shelf water eventually spill over the continental shelf edge and flow down the slopes as dense water cascades (see e.g. Ivanov et al., 2004, for an overview of known cascading locations in the Arctic and other oceans). During their descent the cascading plumes entrain the ambient water, lose their initial density gradient and eventually disperse laterally into the ambient stratification (e.g. Aagaard et al., 1985; Jungclaus et al., 1995; Shapiro et al., 2003).

Dense water formation is particularly intense in coastal polynyas where constant re-

¹Article 'Is Global Warming Changing the Arctic?' on <http://www.whoi.edu/oceanus/viewArticle.do?id=9206>

moval of frazil ice results in high rates of ice formation and subsequent brine rejection leading to localised density excess (Maqueda et al., 2004). Arctic polynyas are estimated to produce a total of 0.7-1.2 Sv ($1 \text{ Sv} \equiv 10^6 \text{ m}^3 \text{ s}^{-1}$) of dense water over the entire Arctic Ocean (Cavaliere and Martin, 1994), making this process of deep water formation comparable to open ocean convection in the Greenland Sea (Smethie Jr. et al., 1986). The dense waters formed on the shelves thus significantly influence the heat and salt balance of the entire Arctic Ocean (Aagaard et al., 1985). Cascading also contributes to the maintenance of the cold halocline layer (Aagaard et al., 1981) and the replenishment of intermediate and deep Arctic waters (Rudels and Quadfasel, 1991; Rudels et al., 1994). A schematic representation of cascading in the Arctic Ocean is shown in Fig. 1.1.

1.3 Modelling cascading

Despite its demonstrated effect on water mass formation and carbon cycling, cascading is poorly represented in modern climate models. The localised occurrence and temporal intermittency of cascading generally means that field campaigns record the outcomes of cascading (e.g. Ivanov et al., 2004) while observations of the process itself remain largely elusive. For this reason, validation of models of cascading against measurements has proved difficult.

Instead, laboratory experiments have been a valuable tool to test models of cascading. The influence of the density difference, flow rate and rotation rate on the dense water flow has been investigated using rotating tanks with either conical or straight bottom slopes (e.g. Shapiro and Zatsepin, 1997; Etling et al., 2000; Cenedese et al., 2004; Sutherland et al., 2004). In addition to a simple laminar flow regime, several more complex regimes have been observed: roll-waves (Shapiro and Zatsepin, 1997) as well as vortices and eddies (Lane-Serff and Baines, 1998, 2000; Etling et al., 2000; Cenedese et al., 2004).

Numerical models have built successfully on these experiments and progress has been made in the modelling of relatively persistent dense water flows, notably the Faroe Bank Channel (see Legg et al., 2009, and references therein). Cascading occurs in the bottom boundary layer where questions of parameterising turbulent mixing and bottom friction continue to pose unresolved challenges for 3-D models (Lane-Serff, 2009), especially due to the fine resolution required to represent a small-scale process over large shelf areas.

1.4 The Storfjorden overflow

A well-known site of dense water formation and subsequent cascading is the Storfjorden, located between $76^{\circ}30' - 78^{\circ}30'$ N and $17^{\circ} - 22^{\circ}$ W in the south of the Svalbard archipelago (Fig. 1.2). Storfjorden is a sill-fjord where intense sea ice formation and brine rejection lead to the formation of brine-enriched shelf water (BSW) that subsequently spills over the sill located at approx. 77° N and 19° E at a depth of 115 m. During the winter freezing period, typically from late November to mid-May, a recurring latent-heat polynya in Storfjorden produces 0.06 to 0.07 Sv ($1 \text{ Sv} \equiv 10^6 \text{ m}^3 \text{ s}^{-1}$) of BSW (Schauer, 1995; Haarpaintner et al., 2001; Skogseth et al., 2004). Near the sill the overflow plume encounters the relatively fresh and cold East Spitsbergen Water (ESW) which mainly reduces its salinity (Fer et al., 2003). The plume subsequently spreads on the shelf where it is channelled through the Storfjordrenna on a westwards path, before turning northwards to sink down the continental slope of western Spitsbergen as a dense water plume (Quadfasel et al., 1988; Schauer, 1995; Fer and Ådlandsvik, 2008; Akimova et al., 2011, see Fig. 1.2). The Storfjorden overflow has been estimated to account for 5 to 10% of shelf waters delivered to the deep Arctic ocean (Quadfasel et al., 1988).

Downstream of the sill the plume initially assumes a two-layer structure – a dense homogenised bottom layer and an overlying diffuse layer (Fer et al., 2003). The lighter

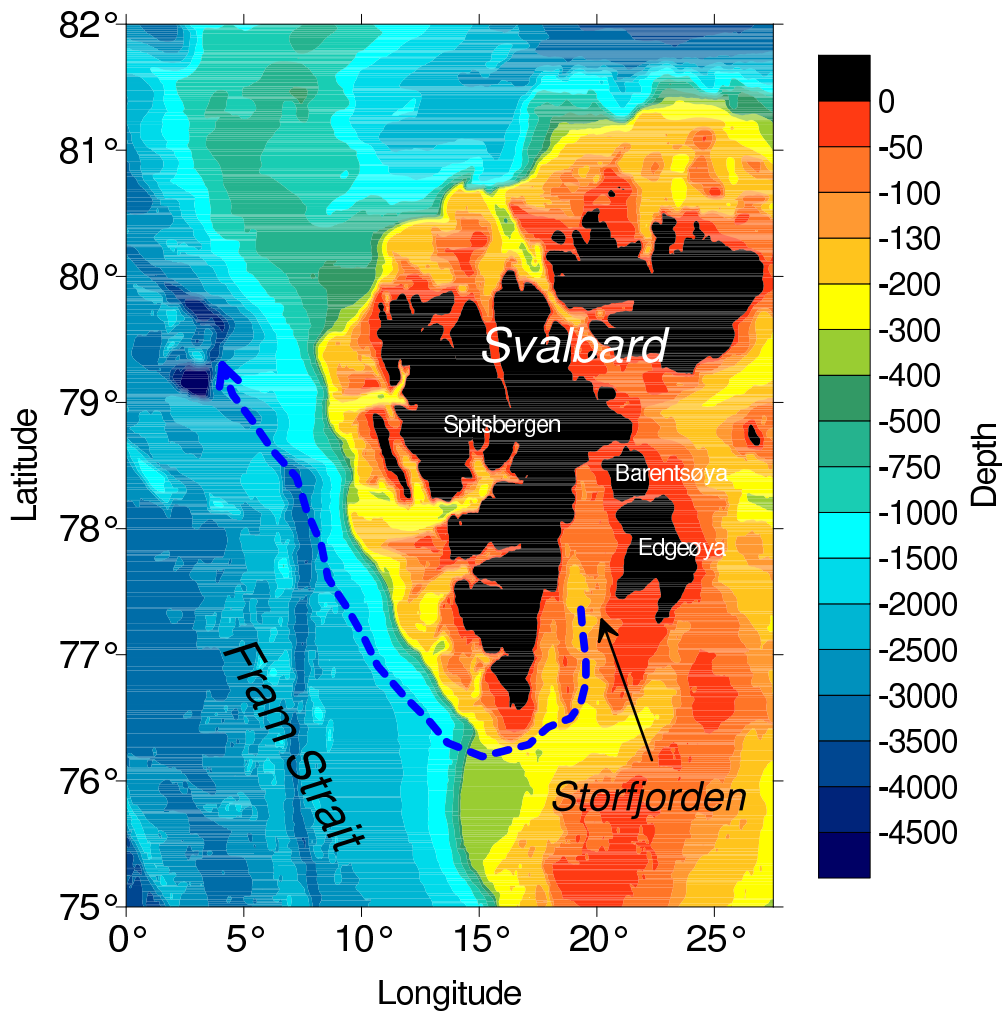


Figure 1.2: Map of the Storfjorden in the Svalbard archipelago. The pathway of the overflow plume (blue arrow) is approximated from observations (Quadfasel et al., 1988) and modelling (Fer and Ådlandsvik, 2008; Akimova et al., 2011). Bathymetry from IBCAO 2.23 (Jakobsson et al., 2008).

fractions of the overflow water in the upper layer mix with shelf waters and spread laterally (Fer et al., 2003). This part of the plume remains within the depth range of the Atlantic Water (approx. 200-500 m) and contributes to the northward freshening and cooling of the West-Spitsbergen Current (Schauer, 1995; Saloranta and Haugan, 2004). The densest fractions pass through the warm Atlantic Layer where they gain heat but lose only little salt as the salinity of the Atlantic Water is close to that of the plume at this stage (35.0 compared to 35.1, see Quadfasel et al., 1988). Shelf waters of

Storfjorden origin have been observed in the deep Fram Strait (at >2000 m) on several occasions, in 1986 (Quadfasel et al., 1988), 1988 (Akimova et al., 2011) and 2002 (Schauer et al., 2003). In observations at other times the cascade was arrested within the depth range of the Atlantic Layer, e.g. in 1994 (Schauer and Fahrbach, 1999) when it was observed no deeper than 700 m.

Observations of the polynya dynamics and dense water formation inside the fjord have been reported by Schauer (1995); Haarpaintner et al. (2001); Skogseth et al. (2004, 2005a, 2008). The variability of the overflow in the sill region was studied by Geyer et al. (2009, 2010), while observations of the plume on the Western Svalbard Shelf slope were reported by Quadfasel et al. (1988); Schauer (1995); Schauer and Fahrbach (1999); Akimova et al. (2011). Previous modelling studies investigating the Storfjorden overflow plume include a 2-layer reduced gravity model (Jungclaus et al., 1995), a 3-D regional model with idealised ambient and forcing conditions (Fer and Ådlandsvik, 2008), and a streamtube-model with parametrised entrainment (Akimova et al., 2011).

1.5 Thesis framework

1.5.1 Open questions

The scientific framework for this thesis was driven by several open questions that have been posed by the author and other oceanographers, both observationalists and ocean modellers.

A widely-cited paper by Marshall et al. (1997) questioned whether relatively small-scale phenomena like dense water cascades can be adequately modelled using a hydrostatic model. Investigating the suitability of such a model to accurately represent cascading was therefore a fitting starting point to the thesis. Any modifications to existing codes that may improve the model results could be tested on a simple test case and later applied to more realistic regional models.

With regards to the cascading process in general, a question that motivated parts

1.5. THESIS FRAMEWORK

of this thesis is of the kind: ‘If the ambient water column structure and the source water characteristics are known, can the fate of the cascade be predicted?’. It was posed by Takamasa Tsubouchi at the UK Arctic Science Conference 2011 in Leeds (see Tsubouchi et al., 2012) and also arises from studies by e.g. Rudels and Quadfasel (1991) and Schauer and Fahrbach (1999). If the plume fate (e.g. final depth) can be indeed predicted, a further question arises: ‘Which parameters control the descent of the cascading plume?’.

Dense water cascades originate in shelf regions from where they propagate towards the shelf edge and downwards into the deep ocean. Due to the shallow water depth on the shelf, the oscillatory motion of barotropic tides is particularly noticeable in these regions that cascading plumes traverse before reaching the continental slope. But ‘How do tides influence a dense water overflow plume?’ is a question that has hitherto not received enough attention (Sonya Legg, pers. comm., 2013). The final study in this thesis therefore addresses tidal effects on dense shelf water plumes.

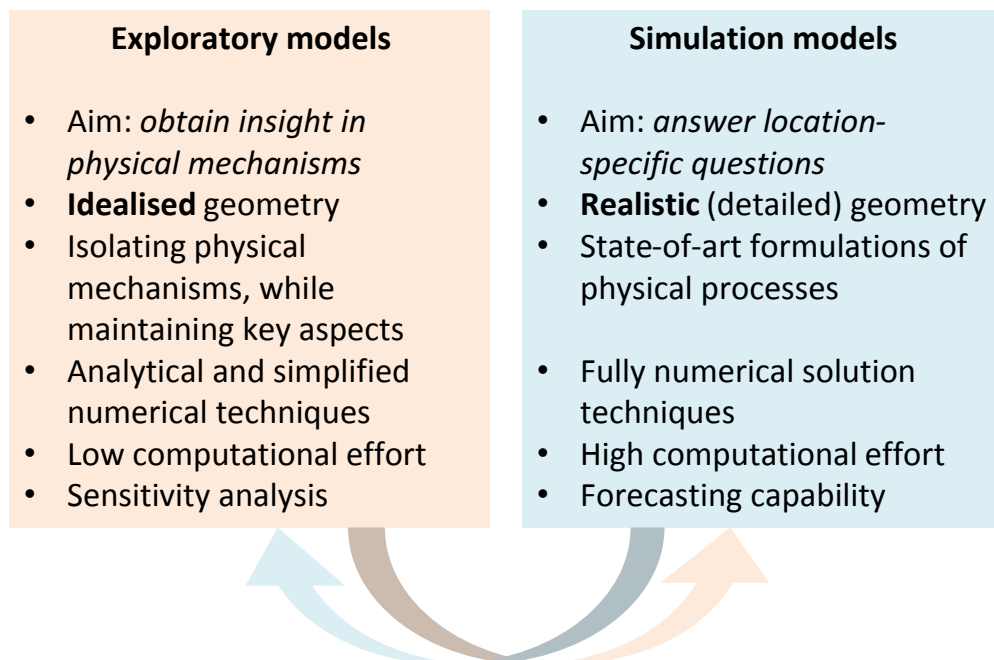


Figure 1.3: Comparison of exploratory vs. simulation modelling approaches. Adapted from Murray (2003) and Roos (2013).

1.5.2 Aims & objectives

To address these questions, this thesis uses a combination of approaches to modelling in order to advance the numerical modelling of dense water cascading. In general, numerical models used in geomorphology and geophysics were arranged by Murray (2003) along a continuum ranging from simplified exploratory models (using an idealised approach) to complex simulation models (using a more realistic approach) (Fig. 1.3).

The studies devised for this thesis take a route from idealised, small-scale models (exploratory approach) to realistic, oceanic-scale models (simulation approach) to achieve these aims:

- to improve the representation of bottom boundary layer physics in a 3-D ocean circulation model such that cascading can be successfully modelled and for the model to be validated against laboratory experiments and observations.
- to establish the relationship between a cascade's source water parameters and its ability to penetrate ambient stratification.
- to investigate effects of tides on the downslope fluxes of an Arctic dense water cascade.

The following objectives will be addressed in this work:

- Previous laboratory experiments are simulated in an ocean model.
- Improvements to the model code are developed, tested and validated.
- Turbulent and diffusive conditions that were not tested in the laboratory are investigated in a 3-D model.
- Model domain is scaled up from laboratory to oceanic dimensions to investigate cascading in an idealised Arctic Ocean.

- A regional model is devised to investigate tidal effects on cascading in the Svalbard region of the Arctic Ocean.

1.5.3 Thesis outline

Chapter 2 introduces the model codes used in this work.

Chapter 3 builds upon a previous laboratory experiment (Shapiro and Zatsepin, 1997) which is in itself an idealised model. The laboratory experiment is simulated to great precision and the results are non-dimensionalised for comparison with the laboratory data. Non-dimensionalisation is commonly used in geophysical fluid dynamics (GFD) as it helps bridge the difference in scale between laboratory/model experiments and the real ocean (Pedlosky, 1987; Dyke, 2007). The successful validation of the numerical model against laboratory experiments gives confidence in its results. The numerical experiments in this chapter also go beyond the simple laminar regime of the laboratory experiments and explore the dynamics of cascading in presence of simulated turbulence and mixing.

Chapter 4 follows on from the simulated laboratory experiments, but the ambient water encountered by the cascade is stratified in a scenario that is still idealised, but more realistic. Ambient stratification can be recreated in the laboratory only with great difficulty (e.g. Oster, 1965; Economidou and Hunt, 2009), hence this task is accomplished here using a numerical model. The scale of the model used for this task makes it impossible to validate model results against laboratory experiments or field observations. However, the lessons learnt in Chapter 3 are implemented to have trust in the validity of the model's results. The resulting model retains the idealised conical geometry of the laboratory experiment while the ambient water column structure was designed to represent the typical water masses of the Arctic Ocean. The up-scaling of the experiment to 'real' oceanic dimensions removes the need to non-dimensionalise the results, and increases their relevance to a wider audience outside of the GFD community.

Chapter 5 takes the simulation model approach to answer questions specific to the environment of Svalbard and the Storfjorden overflow. A regional model of Svalbard is set up to incorporate actual bathymetry, realistic initial conditions, open boundary conditions with realistic forcing, full atmospheric forcing and tides to simulate the overflow as realistically as possible. Model runs performed with and without tides are compared to isolate the influence of tides on the cascading plume. Studies investigating tidal effects on dense water plumes were scarce until the ‘AnSlope’ experiment in the Antarctic (Gordon et al., 2004) which prompted Ou et al. (2009) and Guan et al. (2009) to develop a hypothesis of the influence of tides on overflows. Chapter 5 assesses whether their hypothesis is applicable to the Storfjorden overflow in the Arctic Ocean.

Chapter 6 concludes this thesis with a summary of the findings and recommendations for further research.

Chapter 2

Methods

2.1 The numerical models

This thesis uses numerical ocean models to study dense water cascades at a range of spatial and temporal scales. While physical models are well suited to investigate complex local flow and transport processes, a numerical model is easily scaled up from the dimensions too impractical to build in a laboratory to spatial scales that resemble the real ocean (Dyke, 2007). The numerical experiments presented here first replicate, then expand on previous studies conducted using physical models in the laboratory. Finally the model attempts to simulate as realistically as possible a real-world scenario, because a model can be manipulated in ways that the real world cannot.

The cascading experiments use two numerical ocean models: POLCOMS (Proudman Oceanographic Laboratory Coastal Ocean Modelling System) and NEMO (Nucleus for European Modelling of the Ocean). The validation and inter-comparison of the two models is beyond the scope of this thesis. The reader is referred to the study by O'Neill et al. (2012), which showed that a coarse-resolution 7 km NEMO model was able to predict the temperature and salinity fields in Liverpool Bay and the Irish Sea with the same degree of confidence as a finer-resolving 1.8 km POLCOMS model.

POLCOMS was designed as a regional ocean model for the study of coastal and shelf processes and has been extensively used and validated for the European continental shelf and the North-East Atlantic, e.g. by Holt and James (2006) and Wakelin et al. (2009), and other regions such as the Black Sea (Enriquez et al., 2005) and the Mediter-

anean Sea (Bolaños et al., 2007). The version of POLCOMS used in this study was modified to be used under the Windows operating system (Enriquez et al., 2005).

NEMO is a state-of-the-art ocean modelling framework for oceanographic research, operational oceanography, regional forecasting and climate studies (Madec, 2008). Typical applications of NEMO include the study of long-term variability of global ocean circulation and water mass properties, and their effects on climate through the transport of heat (Drakkar Group, 2007), often with a focus on large-scale ocean current systems such as the Atlantic meridional overturning circulation (AMOC) (Blaker et al., 2012). In recent years, a significant driver for NEMO development has been its application to operational ocean forecasting (e.g. Storkey et al., 2010). The addition of a terrain-following s -coordinate system, a non-linear free surface formulation, sophisticated turbulence closure schemes and other refinements in NEMO-SHELF (see O’Dea et al. (2012) and Section 2.3 for further details) has made NEMO attractive for regional and shelf seas studies, including this thesis. NEMO-SHELF has been successfully validated for the North West European Continental Shelf (O’Dea et al., 2012; Edwards et al., 2012) and for the Irish Sea (O’Neill et al., 2012). The NEMO-SHELF code used here was adapted to run under the Windows operating system for the study in Chapter 4 (see Appendix D for details). The experiments in Chapter 5 were conducted on a parallel High Performance Computing cluster (see Appendix E).

In the study of cascading it is desirable to identify the dense water plume against the ‘background’ of the ambient water. In the POLCOMS model of a laboratory tank initially filled with homogeneous ambient water (Chapter 3) any density differences arise from salinity variations alone as the temperatures of plume and ambient water are identical. The modelled salinity field thus acts as a tracer for the plume as shown in Figs. 3.4a, b and f. When complex ambient stratification is present, the identification of the plume against a wide range of ambient densities is more challenging, and the

availability in NEMO of the passive tracer module TOP presents a significant advantage and one of the reasons why the experiments in Chapters 4 and 5 were carried out using NEMO. Additionally, development of POLCOMS recently ceased and resources were shifted towards development of NEMO, making it a more attractive long-term option with a growing community of users and developers.

2.2 The POLCOMS model

The numerical model used in Chapter 3 is POLCOMS (Holt and James, 2001), a finite difference ocean model that uses the incompressible, hydrostatic and Boussinesq approximations. The following overview is a brief summary based on Holt and James (2001) and the POLCOMS documentation¹.

The model equations are formulated in spherical polar sigma coordinates: χ (eastwards), ϕ (northwards) and σ (vertical), with $\sigma = (z - \zeta)/(h + \zeta)$, where z is the Cartesian vertical coordinate, h is the water depth relative to the reference sea level ($z = 0$) and ζ is the elevation above this; the total water depth is $H = h + \zeta$. The vertical coordinate σ is discretised onto terrain-following levels whose resolution is a function of depth (s -coordinate), so that the number of vertical levels can be enhanced near the top and bottom boundaries of the domain.

$$\sigma_k = \begin{cases} S_k + \frac{h_{i,j} - h_c}{h_{i,j}(C(S_k) - S_k)} & , h_{i,j} > h_c \\ S_k & , h_{i,j} \leq h_c \end{cases} \quad (2.1)$$

where S_k are n evenly spaced levels (n is an integer) between $\sigma = -(1 + 0.5(n - 2))^{-1}$ and $\sigma = 0.5(n - 2)^{-1}$. The deviation from the usual σ -levels is given by

$$C(S_k) = (1 - B) \frac{\sinh(\theta S_k)}{\sinh(\theta)} + B \frac{\tanh(\theta(S_k + 0.5)) - \tanh(0.5\theta)}{2 \tanh(0.5\theta)} \quad (2.2)$$

¹available online at http://coobs.pol.ac.uk/modl/metfcst/POLCOMS_DOCUMENTATION/polcoms.html

2.2. THE POLCOMS MODEL

The stretching function follows the s -coordinate transformation of Song and Haidvogel (1994). For the study in Chapter 3 the following parameters are used: $h_c = 10^{-4}$ (critical depth), $\theta = 7.4$ (surface control parameter), $B = 1$ (bottom control parameter).

In the horizontal, POLCOMS uses the B-grid discretisation, which ensures an accurate representation of the Coriolis force. The semi-implicit scheme is used for the Coriolis term as it conserves kinetic energy (see Cushman-Roisin and Beckers, 2011, chapter 2).

The density $\rho(T, S, p) = \rho(T, S, 0) + \rho'(T, S, p)$ is defined by an approximation to the full UNESCO equation of state following Mellor (1991).

POLCOMS solves the incompressible, hydrostatic, Boussinesq equations of motion. For computational efficiency, the equations are split into a fast barotropic component and a slow baroclinic component, so the eastward velocity is $u = \bar{u}(\chi, \phi, t) + u_r(\chi, \phi, \sigma, t)$ and the northward velocity is $v = \bar{v} + v_r$. The equations for the depth independent barotropic component are:

$$\frac{\partial \bar{u}}{\partial t} = f\bar{v} - (R \cos \phi)^{-1} \left(g \frac{\partial \zeta}{\partial \chi} + \rho_0^{-1} \frac{\partial P_a}{\partial \chi} \right) + H^{-1} (F_S - F_B) + NLB_\chi \quad (2.3)$$

and

$$\frac{\partial \bar{v}}{\partial t} = f\bar{u} - R^{-1} \left(g \frac{\partial \zeta}{\partial \phi} + \rho_0^{-1} \frac{\partial P_a}{\partial \phi} \right) + H^{-1} (G_S - G_B) + NLB_\phi \quad (2.4)$$

For the bottom stresses F_B and G_B a no-slip boundary condition is introduced by modification of the original POLCOMS code (see Appendix A). At vertical boundaries, a slip boundary conditions is used. The calculation of the surface stresses F_S and G_S from wind velocities is deactivated here for the simulation of an idealised laboratory configuration that requires no atmospheric forcing.

The equations for the depth varying baroclinic component are:

$$\frac{\partial u_r}{\partial t} = -L(u) + f v_r + \frac{u v \tan \phi}{R} - \Pi_\chi + D(u) - H^{-1}(F_S - F_B) - NLB_\chi \quad (2.5)$$

and

$$\frac{\partial v_r}{\partial t} = -L(v) - f u_r + \frac{u^2 \tan \phi}{R} - \Pi_\phi + D(v) - H^{-1}(G_S - G_B) - NLB_\phi \quad (2.6)$$

where R is the radius of the Earth and the buoyancy terms are:

$$\Pi_\chi = (R \cos \phi)^{-1} \left. \frac{\partial \psi}{\partial \chi} \right|_z \quad \Pi_\phi = R^{-1} \left. \frac{\partial \psi}{\partial \phi} \right|_z \quad (2.7)$$

The calculation of the buoyancy contribution to the horizontal pressure gradient term $-\nabla P$ (where ∇ is the horizontal gradient operator) is optimised to avoid errors with steep topography. Therefore, Eq. 2.7 has not been transformed onto σ -coordinates because the term is calculated by interpolating the buoyancy field onto horizontal planes through velocity points then integrating to calculate the hydrostatic pressure.

The depth means of the nonlinear and buoyancy terms are:

$$NLB_\chi = \int_{-1}^0 \left[-L(u) + \frac{u v \tan \phi}{R} - \Pi_\chi \right] d\sigma \quad (2.8)$$

$$NLB_\phi = \int_{-1}^0 \left[-L(v) + \frac{u^2 \tan \phi}{R} - \Pi_\phi \right] d\sigma \quad (2.9)$$

The advection terms are given by

$$L(a) = \frac{u}{R \cos \phi} \frac{\partial a}{\partial \chi} + \frac{v}{R} \frac{\partial a}{\partial \phi} + \Omega \frac{\partial a}{\partial \sigma} \quad (2.10)$$

with

$$\Omega = \frac{\sigma}{H} \frac{\partial \zeta}{\partial t} - (HR \cos \phi)^{-1} \left[\frac{\partial}{\partial \chi} \left(H \int_0^\sigma u d\sigma \right) + \frac{\partial}{\partial \phi} \left(H \cos \phi \int_0^\sigma v d\sigma \right) \right] \quad (2.11)$$

The advection equations are evaluated using the Piecewise Parabolic Method (e.g. Colella and Woodward, 1984), which creates significantly less numerical diffusion and dispersion than ordinary advection schemes and helps, thus, to preserve sharp property gradients and boundaries, as is required in the cascading experiments in Chapter 3.

POLCOMS, like other models of this type, replaces the vertical gradients of the stresses by a diffusion term:

$$D(a) = H^{-2} \frac{\partial}{\partial \sigma} \left(K_z \frac{\partial a}{\partial \sigma} \right) \quad (2.12)$$

where K_z is the eddy viscosity. For the experiments in Chapter 3 K_z is kept constant.

The model includes a fully non-linear free surface defined as

$$\frac{\partial \zeta}{\partial t} = (R \cos \phi)^{-1} \left[\frac{\partial}{\partial \chi} (H \bar{u}) + \frac{\partial}{\partial \phi} (H \cos \phi \bar{v}) \right] \quad (2.13)$$

which guarantees exact conservation of fluid volume. The model's ability to accommodate changes in volume is crucial for the reproduction of the laboratory experiments described in Chapter 3.

2.3 The NEMO model

For the studies presented in Chapters 4 and 5 we use the NEMO-SHELF model (O'Dea et al., 2012). The modifications to the standard NEMO code (Madec, 2008) for use in shelf seas and regional studies are described in detail by O'Dea et al. (2012). Therefore, only a brief summary of the differences is given in the following. The model's configuration specific to the studies in Chapter 4 and 5 is described as well as several

modifications to the NEMO-SHELF code which are specific to this thesis.

2.3.1 The ' s_h ' vertical coordinate system

A key departure of the NEMO shelf code from the open ocean is the use of a terrain-following s -coordinate discretisation in the vertical, instead of z -coordinates. The s -coordinate system is well suited to the modelling of density currents (see Chapter 3), but the horizontal boundaries between ambient layers (Fig. 4.1b) would suffer numerical diffusion over areas of sloping topography where s -levels intersect the isopycnals at an angle. The vertical coordinate system was therefore modified because neither the traditional s -coordinate nor z -coordinate systems suit a scenario where strong gradients are orientated vertically (in the ambient water) and also normal to the slope (at the upper plume boundary). The approach of blending s - and z -coordinates in this study can be traced back to Enriquez et al. (2005) who used a traditional s -coordinate stretching function (Song and Haidvogel, 1994) but achieved horizontal s -levels over the interior of a basin by capping its bathymetry. Ivanov (2011) changed the traditional s -coordinate formulation by introducing virtual seabeds at certain depth levels to maintain horizontal s -levels closer to the slope. The levels designated as virtual seabeds (here called s_h -levels²) follow the terrain only at shallower depths, while maintaining a prescribed depth over deep bathymetry.

The modified s_h -coordinate system, that is introduced here, refines the Ivanov (2011) approach by smoothing the transition between horizontal and terrain-following s -levels (Fig. 2.1). The smoothing reduces errors in the calculation of the second derivative of the s -level slope.

The algorithm calculating the s -level depths at a given location with bathymetric depth D starts by adding levels in the bottom boundary layer (BBL) equidistantly over a constant thickness H_{bbl} . This is to avoid any loss in vertical resolution with increasing

²subscript 'h' alludes to these levels being horizontal.

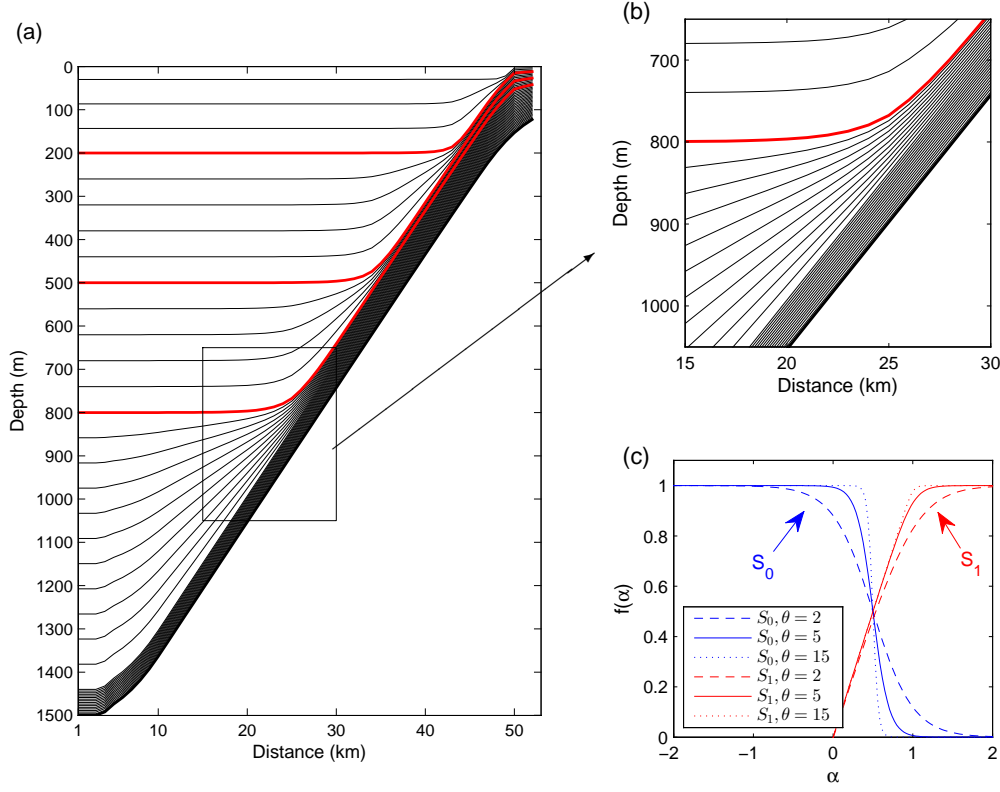


Figure 2.1: (a) The s_h -coordinate system shown as a cross-section through the centre of the model domain used in chapter 4. The box is magnified in (b) which shows that out of a total of 42 levels, at least 16 are reserved for a bottom boundary layer of constant thickness. The s_h -levels (i.e. virtual seabeds, in red) are placed at certain depth levels to flatten s -levels in the interior and coincide with isopycnals in the ambient water. Panel (c) shows the smoothing functions S_0 and S_1 (Eqns. (2.15) and (2.16) respectively) with different values for the smoothing parameter θ (see 2.3.1).

depth (as is the case with the traditional s -coordinate stretching function). The depths Z_h of the s_h -levels (the virtual seabeds) are then calculated based on their prescribed depths Z_l according to the following scheme.

Let $D_{lim}(D) = D - H_{bbl} - k\Delta z_{min}$ be the deepest depth that the s_h -level can be placed at, where H_{bbl} is the thickness reserved for the bottom boundary layer, k is the number of levels between the s_h -level and the top of the bottom boundary layer, and Δz_{min} is the minimum allowable level spacing. This leads to a simple function

$$Z_h = \begin{cases} Z_l & , D_{lim} > Z_l \\ D_{lim} & , D_{lim} \leq Z_l \end{cases} \quad (2.14)$$

where the s_h -level is either horizontal ($Z_h = Z_l$) or terrain-following ($Z_h = D_{lim}$). As a consequence its first derivative is discontinuous in one point, which leads to errors in horizontal pressure gradient calculations where its second derivative is undefined.

In order to smoothly blend between these two cases, we start with a function S_0 that transitions smoothly between 1 to 0 whilst satisfying that $S_0(0.5) = 0.5$ (see blue curves in Fig. 2.1c):

$$S_0(x) = 0.5 \tanh(0.5 \theta - x \theta) + 0.5 \quad (2.15)$$

where θ is a non-dimensional smoothing parameter. For values of approximately $2 \leq \theta \leq 20$ the transition is smooth, but as $\theta \rightarrow \infty$ the function becomes a step function (with a step at $x = 0.5$). Integrating Eq. (2.15) gives Eq. (2.16):

$$S_1(\alpha) = 0.5 \alpha - \frac{0.5}{\theta} \log(\cosh(\theta - \alpha \theta)) + 0.5 - \frac{\log(2)}{2\theta} \quad (2.16)$$

where $\alpha = Z_l/D_{lim}$ is a scale factor for the prescribed s_h -level depth Z_l . Eq. (2.16) approximately satisfies $S_1(\alpha) \approx \alpha$ for $0 \leq \alpha \leq 1$ and $S_1(\alpha) \approx 1$ for $\alpha > 1$ (see red curves in Fig. 2.1c) so it could be used to blend smoothly from $Z_h = Z_l$ at depth (using the range $\alpha \geq 1$) into $Z_h = D_{lim}$ in the shallows (using the range $0 \leq \alpha < 1$).

While Eq. (2.16) closely matches the identity function $f(x) = x$ in the approximate range $0 \leq x \leq 0.5$ it does not exactly do so, especially for small values of θ (see dashed red curve in Fig. 2.1c for $\theta = 2$). The s_h -level could miss its target depth Z_l in the interior of the basin by a small margin, and a second smoothing function

$$S_2(\alpha) = \begin{cases} \alpha & , \alpha \leq 0.5 \\ 0.5 + 0.5 \tanh(2\alpha - 1) & , \alpha > 0.5 \end{cases} \quad (2.17)$$

is introduced to blend the identify function into Eq. (2.16). The final s_h -level depth Z_h is then derived as:

$$Z_h = D_{lim} \left((1 - S_2) \alpha + S_2 S_1 \right) \quad (2.18)$$

2.3.2 Turbulence closure

NEMO-SHELF includes the Generic Length Scale (GLS) turbulence model (Umlauf and Burchard, 2003) which we use in its k - ϵ configuration. The parameters, based on Warner et al. (2005) and Holt and Umlauf (2008), are listed in Table 2.1. The scheme's realistic vertical diffusivity and viscosity coefficients give confidence to the accurate representation of the frictional Ekman layer within the plume.

2.3. THE NEMO MODEL

Table 2.1: Parameters for the $k - \varepsilon$ scheme in the Generic Length Scale (GLS) turbulence closure model ^a

	value ^{a,b}	variable name in NEMO
ε_{min}	$10^{-12} \text{ m}^2 \text{ s}^{-3}$	rn_epsmin
k_{min}	$7.6 \times 10^{-6} \text{ m}^2 \text{ s}^{-2}$	rn_emin
m	1.5	zmm
n	-1.0	znn
p	3.0	zpp
c_1	1.44	rn_psi1
c_2	1.92	rn_psi2
c_3^+	1.0	rn_psi3p
c_3^-	-0.629	rn_psi3m
σ_k	1.0	rn_sc_tke
σ_ψ	1.3	rn_sc_psi
σ_{ψ_0}	2.01206499534943	rn_sc_psi0
c_μ^0	0.5268	rn_c0 ^c
c_{lim}	0.267	clim_galp ^d

^a Umlauf and Burchard (2003, 2005)

^b Warner et al. (2005)

^c parameter for the stability function 'A' in Canuto et al. (2001), value from Umlauf (2001)

^d the Galperin et al. (1988) limit, value from Holt and Umlauf (2008)

2.3.3 Numerical schemes

A further difference in NEMO-SHELF is the use of a non-linear free surface formulation with variable volume (Levier et al., 2007) which is advantageous for this study as it allows to account for volume changes as a result of the injection of dense water using the model's river scheme.

The advection scheme in the vertical is the Piecewise Parabolic Method (vPPM, by Liu and Holt, 2010) while the TVD (Total Variation Diminishing) scheme is used for horizontal advection. The high precision Pressure Jacobian scheme with Cubic polynomial fits which is particularly suited to the s -coordinate system is used as the horizontal pressure gradient algorithm (kindly made available by H. Liu and J. Holt, NOCL).

A no-slip boundary condition is implemented at the bottom (rather than the quadratic drag law, which is often used as standard bottom friction parametrisation in ocean models). For details on the code modifications, see Appendix B. Fine vertical resolution near the bottom (relative to the Ekman layer height) is prescribed to explicitly resolve the velocity profiles of the 'modified' Ekman spiral in the frictional bottom boundary layer (see section 3.2.2). Resolving bottom friction, rather than parameterising it, is demonstrated in Chapter 3 to increase the accuracy of modelling gravity currents in a rotating framework.

2.3.4 Rotation of the lateral diffusion operator

For the parametrisation of the subgrid-scale horizontal diffusion of tracers and momentum we use the Laplacian (harmonic) operator with constant diffusivity coefficients ($A_{h_t} = A_{h_m} = 3.0 \text{ m}^2\text{s}^{-1}$ for tracers and momentum respectively). Care is taken to separate the large lateral diffusion from the tiny diffusion in the diapycnal direction (see Griffies, 2004, for a discussion) by activating the rotated Laplacian operator scheme. For the study in Chapter 4 the calculation of the slope of rotation is modified to blend the slope of isopycnal surfaces with the slope of surfaces of constant geopotential de-

pending on the intensity of the background stratification. This approach was especially devised for the idealised ambient conditions in Chapter 4 where the calculation of isopycnal surfaces within a well-mixed ambient layer may lead to unphysical slope angles that cause lateral diffusion to ‘leak’ into the sensitive vertical diffusion.

Lateral diffusion processes occur predominantly along neutral surfaces (Griffies, 2004), which may not be easily characterised (in a well-mixed layer for example) and may be computationally expensive to derive, and are thus often approximated (see McDougall and Jackett, 2005, and references therein). Two such approximations for the slope m of operator rotation are considered: (i) calculation of the slope of isopycnal surfaces $m_{iso} = \frac{d\rho}{dx} / \frac{d\rho}{dz}$, and (ii) calculation of the slope m_{hor} of near-horizontal surfaces of constant geopotential derived from the time-evolving elevation of the sea surface.

The rotation of the diffusion operator according to m_{iso} is generally preferred in shelf seas models (H. Liu, pers. comm., 2012) where density gradients are generally well defined by prevalent stratification. However, in mixed layers of insignificant density gradients the calculation of m_{iso} can lead to unphysical fluctuations in the slope. The rotation of the diffusion operator is therefore limited to a maximum slope angle $m_{max} = 0.028$ which reflects the 1.8° inclination of the model topography³. Even with this safeguard in place the analytical description of our ambient density profile can lead to numerically spurious slopes within a well-mixed layer and the use of the m_{hor} slopes would be preferable in that case.

The model configuration for Chapter 4 therefore adopts a blended scheme where the Laplacian diffusion operator is rotated according to m_{iso} in stratified regions and according to m_{hor} in well-mixed regions. We assess here the degree of stratification

³The slope limit m_{max} can be approximated from the typical length scale L and depth scale H of the diffusion process: $m_{max} = \frac{H}{L}$. NEMO typically uses a value of $m_{max} = 0.01$ which is not suitable for steep topographical gradients in our scenario. This original value was derived for large-scale ocean models with a typical mixed layer depth of $H = 200\text{m}$. The length scale of lateral diffusion $L_{A_h} = 20\text{km}$ is in turn derived from a typical horizontal diffusion coefficient $A_h = 2000\text{m}^2\text{s}^{-1}$ while assuming a typical horizontal velocity of 10cm s^{-1}).

via the buoyancy frequency N^2 which is a NEMO model variable. Two additional parameters N^2_{hor} and N^2_{iso} are introduced in our configuration to define the lower limit of the buoyancy frequency below which we use m_{hor} and above which we use m_{iso} , while intermediate values are linearly interpolated. The final slope m for the rotation of the Laplacian diffusion operator is calculated as:

$$\alpha = \min\left(1, \frac{\max(0, (N^2 - N^2_{hor}))}{N^2_{iso} - N^2_{hor}}\right) \quad (2.19)$$

$$m = (1 - \alpha) \cdot m_{hor} + \alpha \cdot m_{iso}$$

While it may be possible to calculate suitable limits without prior knowledge, we derived $N^2_{hor} = 5 \times 10^{-6} \text{ s}^{-2}$ and $N^2_{iso} = 5 \times 10^{-5} \text{ s}^{-2}$ by visually inspecting cross-section plots of N^2 (see Fig. 2.2 for an example). In keeping with the standard NEMO code, a 2D Shapiro-filter is applied to the final values of m as well as an additional reduction by 50% near coastal boundaries. Furthermore, the code that specially adapts lateral diffusion in model levels within and just below the surface mixed layer was removed.

The effects of various operator rotation schemes and the advantages of using a blended approach are shown in Appendix C.

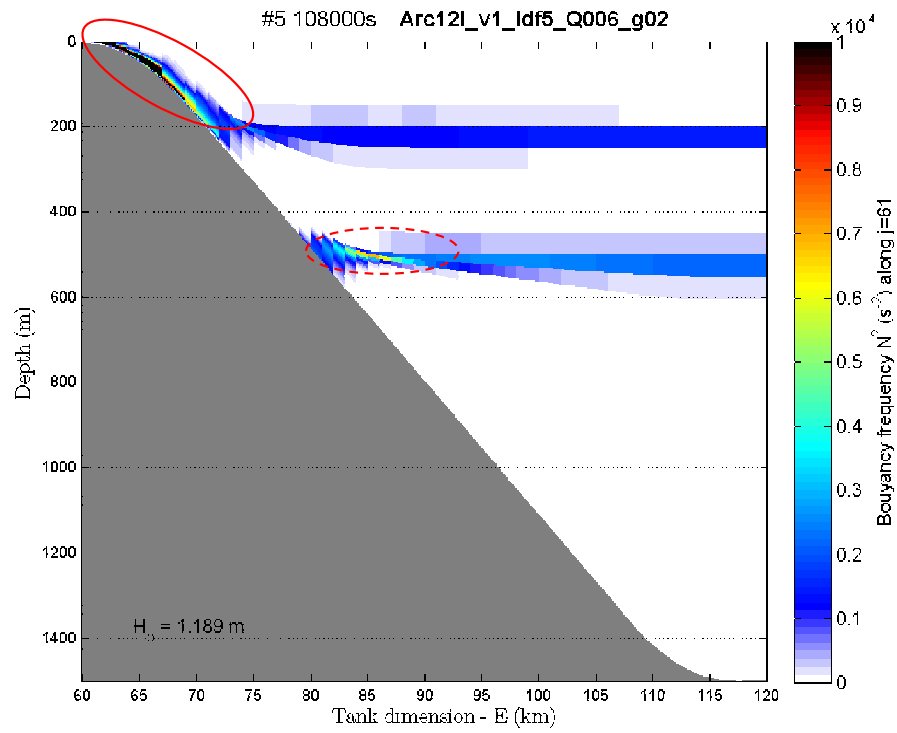


Figure 2.2: Cross-section of the buoyancy frequency N^2 (s⁻²) close to the start of the model run (after 30 hours) when the plume is just starting to descend the slope (solid circle). At this stage the two interfaces between the 3 layers are starting to diffuse into a stable stratification which also shows up as areas of high buoyancy frequency (dashed red circle).

Chapter 3

Numerical simulations of dense water cascading on a steep slope

3.1 Introduction

Laboratory experiments by Shapiro and Zatsepin (1997) are revisited by applying a 3-D ocean model to the study of cascading on a steeply-sloping rotating cone. First, the laboratory results are reproduced to validate the 3-D model. Areas of applicability of the previously developed simplified theories of cascading are identified and Ekman dynamics in the frictional boundary layer are explored. Numerical experiments then go beyond the scope of the laboratory experiments by investigating parameters and regimes which are difficult to create in the laboratory and are challenging to analyse with simplified theories.

This chapter is structured as follows: Section 3.2 gives some background on the laboratory experiments that formed the basis for the numerical experiments. Section 3.3 describes the model setup and the experimental design. Results are presented in Section 3.5 and discussed in Section 3.6. The chapter closes with conclusions in Section 3.7.

3.2 Background

3.2.1 Laboratory experiments

The laboratory experiments carried out by Shapiro and Zatsepin (1997), hereafter referred to as SZ97, consisted of a solid cone placed in a tank mounted on a rotating

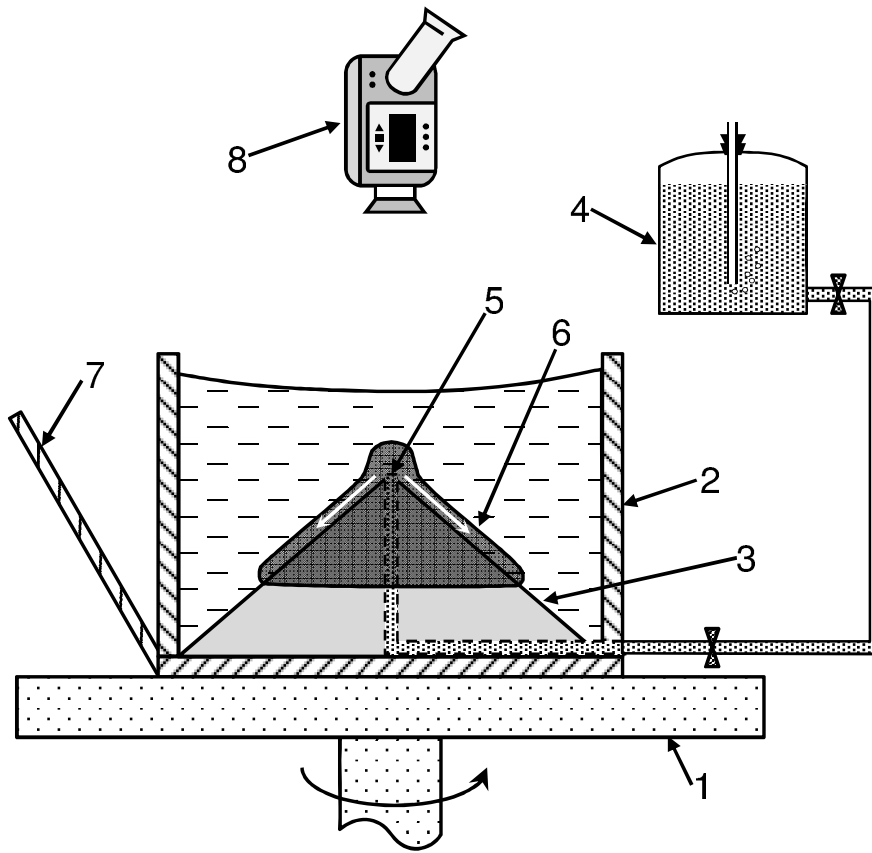


Figure 3.1: Schematic of the laboratory setup used by Shapiro and Zatsepin (1997): 1, rotating turntable; 2, water tank; 3, cone; 4, Mariotte's bottle; 5, dense water inflow; 6, dense water plume; 7, side-view mirror; 8, video camera (Adapted from fig. 1 in Shapiro and Zatsepin, 1997).

turntable (Fig. 3.1). The tank ($50 \times 50 \times 46$ cm) was filled with a homogeneous water solution while dense water was injected at the tip of the cone, which then propagated downwards along the wall of the cone. The dense water was coloured so the spread of the plume could be observed using a video camera (Fig. 3.1).

The geometry of a cone has a number of advantages over a straight slope. A cone simulates a virtually endless slope along which the downslope velocity can be studied for long periods of time without the plume reaching any lateral boundary. This avoids the requirement for a large domain and possible complications with lateral boundary conditions in numerical experiments.

3.2. BACKGROUND

The main experimental parameters were the reduced gravity g' , the flow rate Q and the Coriolis parameter f . The density difference represented by g' was created by mixing in varying amounts of salt into the inflowing water. The Coriolis parameter f was varied by different rotation speeds of the turntable. The flow rate Q was regulated by a valve connected to a Mariotte's bottle. SZ97 observed limited mixing only in the region around the cone tip where a bulbous dome forms over the inflow. Diapycnal mixing between the ambient and injected fluids was not observed outside of this region near the inflow. The propagation of the plume was described using a simplified theory of cascading, which is introduced in the following section.

3.2.2 The SZ97 'reduced physics' model

It is known that if the effects of friction are negligible, a steady-state gravity current in a rotating framework would simply flow along contours of constant depth (e.g. Griffiths, 1986). The reference alongslope velocity referred to hereafter as the Nof speed (Nof, 1983) is given by Eq. (3.1):

$$V_{Nof} = \frac{g'}{f} \tan \theta \quad (3.1)$$

where f is the Coriolis parameter and θ is the slope angle. The reduced gravity g' is defined as $g' = g \frac{\Delta \rho}{\rho_0}$, where g is the acceleration due to gravity, $\Delta \rho$ is the density difference between the dense water and ambient water, and ρ_0 is the ambient density.

Downslope motion of dense water is facilitated by forces, such as friction with the bottom or at the interface between the plume and the ambient water, that break the constraints of potential vorticity conservation. Friction brings the flow velocity to zero at the bottom, the viscosity of the fluid propagates its effects into the flow, while at some distance from the bottom the Coriolis force dominates in the interior of the flow over the diminishing frictional force. This creates a thin boundary layer of the order of the Ekman depth, which arises as a key height scale for the bottom boundary layer.

3.2. BACKGROUND

Accounting for the slope angle θ the Ekman depth H_e is defined as:

$$H_e = \sqrt{\frac{2\nu}{f \cos \theta}} \quad (3.2)$$

where ν is the vertical viscosity.

Shapiro and Hill (1997), hereafter referred to as SH97, studied gravity currents in a rotating framework using a $1\frac{1}{2}$ -layer model, which assumed a homogeneous layer of dense water overlaid by a homogeneous upper layer of ambient water. In contrast to ‘stream-tube’ or ‘slab’ models (Smith, 1975; Killworth, 1977; Price et al., 1993) which assumed a vertically uniform velocity distribution within the plume, the velocity structure in SH97 is fully three-dimensional. Compared to ‘full physics’ 3-D numerical models the SH97 model assumes a simplified density structure and hence belong to a class of ‘reduced physics’ models.

SH97 showed that in the special case of steady state cascading over a plane slope with no entrainment and no upper layer flow the plume height cannot exceed $h_f = 1.78 \times H_e$. In this case, the downslope velocity can be expressed as a fixed fraction ($u_F = 0.2 V_{Nof}$) of the alongslope velocity, meaning that the lower edge of the plume crosses the isobaths at a constant angle. A different approach was taken by Killworth (2001), who parametrised the rate of descent based on an energy equilibrium solution as $\frac{dD}{ds} = \frac{1}{400}$ where D is a downslope bathymetric variable and s is the along-stream variable.

The solution of the SH97 model for the horizontal velocity profiles describes a veering of velocity initially reminiscent of the ‘classic’ Ekman spiral (see Ekman (1905) and fig 8-3 in Cushman-Roisin and Beckers (2011), p. 224). In the two-layer fluid of the SH97 model, however, the velocity profiles follow a ‘modified’ Ekman spiral (see Fig. 3.5). The ‘modified’ Ekman spiral includes the interfacial Ekman layer which enhances downslope transport and incorporates additional velocity veering at the bound-

ary between dense and ambient water.

The equations for the ‘modified’ Ekman spiral – i.e. horizontal velocities in a 2-layer flow – are given here in non-dimensional variables for the lower layer (V_l) of the plume and the upper layer (V_u) of the ambient liquid (for a detailed derivation, see Shapiro and Hill, 2003, appendix A). Velocities V_l and V_u are scaled by V_{Nof} ; the plume height h_F and the height above the bottom z are scaled by H_e .

$$V_l = -\left(\frac{i}{2}\right) \left\{ \exp\left(-(1+i)(h_F - z)\right) + \left[2 - \exp\left(-(1+i)h_F\right) \right] * \exp\left(-(1+i)z\right) - 2 \right\} \quad (3.3)$$

$$V_u = \left(\frac{i}{2}\right) \exp\left(-(1+i)z\right) * \left[\exp\left(\frac{(1+i)h_F}{2}\right) - \exp\left(-\frac{(1+i)h_F}{2}\right) \right]^2 \quad (3.4)$$

Plots of equations (3.3) and (3.4) are shown in Fig. 3.5 for comparison with velocity profiles from a ‘full physics’ three-dimensional numerical model.

SZ97 adapted the SH97 model to a gravity current flowing down a conical slope. This version of the model was formulated in a curvilinear rotating orthogonal coordinate system, set the entrainment velocity to zero and used non-dimensional variables. The non-dimensionalisation used here follows precisely the scheme introduced by SZ97 (see therein for details): L_0 as a horizontal length scale of the dense water plume, T_0 as the time scale (Eq. (3.5)), the Ekman depth H_e (Eq. (3.2)) as the vertical height

scale and V_{Nof} (Eq. (3.1)) as the scale for velocities in the downslope and alongslope direction.

$$L_0 = \frac{Q}{2\pi \cos \theta V_{Nof} \sqrt{2} H_e G_m}, \quad T_0 = \frac{\sqrt{2} L_0}{V_{Nof}} \quad (3.5)$$

where $G_m \cong 1.12$ is a numerical constant.

The length scale L_0 is therefore proportional to $Q/(V_{Nof} H_e)$, i.e. proportional to the length of the plume from tip to front and suitably scales with the flow rate Q while H_e and V_{Nof} are the ‘natural’ scales for the plume height and velocities respectively. In Section 3.5 this non-dimensionalisation scheme is used to compare results from the ‘full physics’ model with the ‘reduced physics’ model and the laboratory experiments.

3.3 POLCOMS model setup

The bathymetry of the POLCOMS model setup is based on the tank used in the Shapiro and Zatsepin (1997) laboratory experiments (Fig. 3.1). The central cone has the original dimensions (radius $r = 25$ cm, inclination angle $\theta = 39^\circ$). The base of the ‘tank’ was extended by 10 cm to avoid boundary complications when the plume reaches the bottom of the cone. The height of the tank was slightly reduced to 32 cm, leaving a depth of 12 cm at the cone tip.

The horizontal grid resolution was chosen to be $\Delta x = 5$ mm for a 120×120 grid to sufficiently resolve lateral details of the descending plume. This small horizontal step addresses potential issues with evaluating bottom pressure on a steep slope when the bottom in the adjacent cell (laterally) is displaced by more than one cell in the vertical (Haney, 1991). Time steps for the barotropic and baroclinic components were set to $\Delta t = 0.75$ ms and 15 ms respectively to satisfy the Courant–Friedrichs–Lewy condition.

The vertical resolution was configured specifically to resolve the physics of Ekman veering in the bottom and interfacial boundary layers. In the vertical 45 s -levels (see

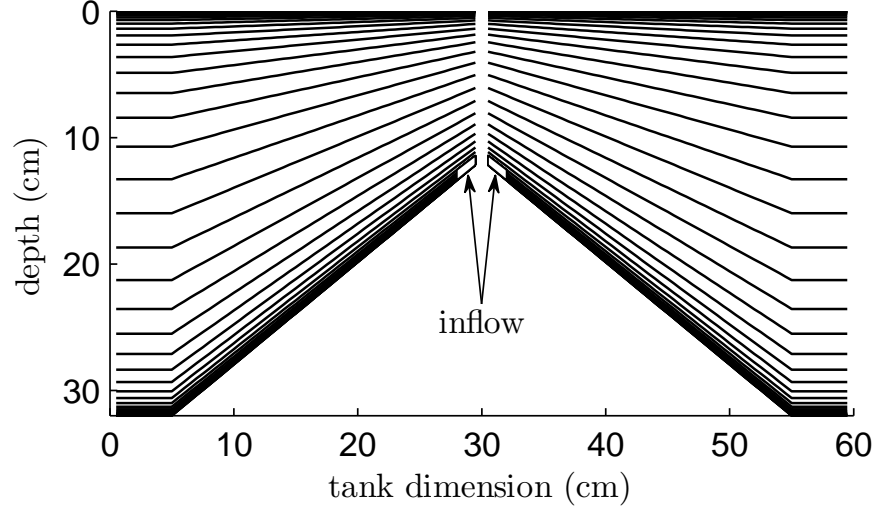


Figure 3.2: Diagram showing the vertical resolution of the model setup with 45 s -levels. Arrows indicate the location of the dense water inflow near the top of the cone which extends vertically over 15 s -levels.

Fig. 3.2) are used. The stretching function for the vertical coordinates is adjusted in order to have 10 computational levels within the frictional layer near the injection point. This gives the finest vertical resolution of $\Delta z = 0.04 \text{ mm}$ at the bottom near the cone tip and the largest s -level spacing $\Delta z = 27.1 \text{ mm}$ in the interior at the tank edges.

The standard bottom boundary condition in POLCOMS is the quadratic drag law using an empirical drag coefficient C_D . Following results by Shapiro and Hill (2003) and Wirth (2009) that proper resolution of the velocity profiles in the frictional layer significantly improves modelling accuracy this layer is explicitly resolved here. Ekman theory requires that friction against the bottom brings the interior velocity (inside the plume) to zero at the bottom boundary (Ekman, 1905). The model code was therefore changed to a no-slip bottom boundary condition (see Appendix A), which, given the fine vertical resolution near the bottom, leads to the development of an Ekman spiral.

The model code related to river inputs was adapted to simulate the injection of salty dense water at the cone tip into a circular region around the mid-point of the model domain (highlighted in Fig. 3.2). The temperature is kept at 20°C for inflowing and

ambient water, while the density difference characterised by g' is simulated by varying the salinity of the inflowing water.

3.4 Experiment design

Table 3.1: Summary of model parameters used in validation (Section 3.5.1) and comparison runs (Section 3.5.3), in experiments that vary the Ekman depth (Section 3.5.4) and experiments with a stratified plume interface (Section 3.5.5).

Parameter	Validation	Comparison	Ekman depth	Stratified plume
f (s^{-1})	1.6	1.0 – 4.1	1.6	1.6
Q ($cm^3 s^{-1}$)	0.3 – 5.0	0.3 – 7.4	1.7 – 3.7	1.6 – 2.1
g' ($cm s^{-2}$)	0.4 – 4.2	0.3 – 9.0	1.2 – 1.7	1.5 – 1.7
κ ($m^2 s^{-1}$)	1.3×10^{-9}	1.3×10^{-9}	1.3×10^{-9}	$10^{-9} - 10^{-5}$
ν ($m^2 s^{-1}$)	2×10^{-6}	2×10^{-6}	$10^{-6} - 10^{-4}$	2×10^{-6} ($10^{-6} - 10^{-5}$)
H_e (cm)	0.179	0.112 – 0.227	0.127 – 1.27	0.179 (0.127 – 0.4)
Pr_ν	1538	1538	769 – 76923	2000 – 0.2 (1 – 0.1)

The following modelling experiments can be grouped into two sets. In the first set of runs the laboratory conditions are simulated by using molecular values of diffusivity and viscosity to validate the model and compare its results to the original laboratory experiments. In a second set of runs the values of either viscosity and diffusivity, or both, are increased to observe the response of the flow to changes in the Ekman depth and investigate the behaviour of a sinking plume with a diffuse (i.e. density stratified) interface, as those latter runs simulate the effects of enhanced mixing. All experiments use a constant value of viscosity ν and diffusivity κ throughout the model domain.

The first set of numerical experiments simulates the non-turbulent (laminar) nature of the flow as observed in the laboratory. The viscosity was not measured in the original laboratory experiments and SZ97 assumed a reference value for molecular viscosity ($\nu \approx 10^{-6} m^2 s^{-1}$). A slightly higher value ($\nu = 2 \times 10^{-6} m^2 s^{-1}$) is used for those model runs comparing the 3-D model to the laboratory experiments (see Table 3.1) to account for possible impurities in the tap water that was originally used in the laboratory. At

oceanic scales there is no appreciable difference between the diffusivity of heat and salt due to turbulence, while at laboratory scales, when molecular processes are significant, they differ by approximately 2 orders of magnitude. In the first set of experiments, the values for the horizontal and vertical diffusivity are based on a reference value for the molecular diffusivity of salt in sea water ($\kappa = 1.3 \times 10^{-9} \text{ m}^2 \text{ s}^{-1}$) as the density difference in all experiments is created by salt alone. The resulting Prandtl Number in the horizontal and vertical is $Pr_h = Pr_v = \frac{\nu}{\kappa} = 1538$.

The second set of numerical experiments simulates conditions that were not created in the laboratory. The response of the plume to changes in the Ekman depth H_e is examined by modifying the vertical viscosity from $\nu = 10^{-6}$ to $10^{-4} \text{ m}^2 \text{ s}^{-1}$ (Section 3.5.4). In these runs the Prandtl Number in the horizontal remains unchanged ($Pr_h = 1538$), but the vertical Prandtl Number Pr_v ranges from 769 to 76900. The model runs that investigate the effects of a diffuse interface (see Section 3.5.5) create a smooth transition between plume and ambient water by modifying the vertical diffusivity in the model from $\kappa = 10^{-9}$ to $10^{-5} \text{ m}^2 \text{ s}^{-1}$. In these runs the vertical Prandtl Number varies from $Pr_v = 2000$ to 0.2. Further experiments are conducted for a plume with a stratified interface at $\kappa = 10^{-6} \text{ m}^2 \text{ s}^{-1}$ by varying the viscosity from $\nu = 10^{-6}$ to $10^{-5} \text{ m}^2 \text{ s}^{-1}$, resulting in a Prandtl Number of $Pr_v = 1$ to 0.1. The latter experiments which investigate a plume with a stratified interface (Section 3.5.5) where $Pr_v = O(1)$ are thought to be the most representative of oceanic conditions where diffusion of momentum (viscosity), heat and salt are all of the same order due to turbulence. Table 3.1 summarises the model parameters for the experiments.

In keeping similar conventions to the laboratory observations by SZ97 the modelled flow is divided into 3 zones (Fig. 3.3a). In the injection zone near the inflow fluid is trapped and a bulbous dome develops (see also Figs. 3.4e and 3.4f). The second zone is the viscous flow forming the main part of the cascade where the dominating balance

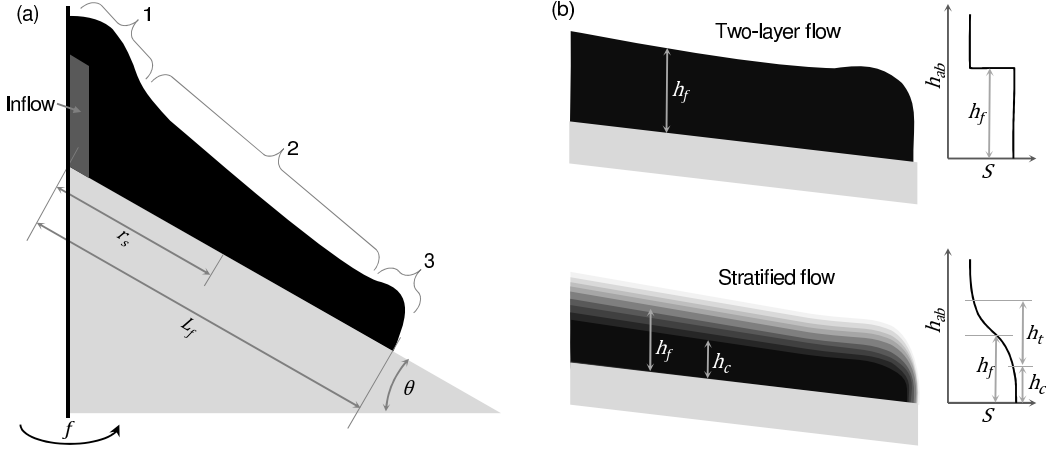


Figure 3.3: Schematics of the modelled flow and the measured variables. (a) Cross-section of an idealised flow on a slope with angle θ . The downslope length of the plume is measured as L_f , while the plume height, salinity and velocity profiles are sampled at a downslope distance r_s . The flow is divided into these zones: 1, injection zone forming a bulbous dome over the inflow at the cone tip; 2, main viscous flow; 3, plume head. (b) The two types of flows with an idealised profile of salinity (S) against height above bottom (h_{ab}) showing how the flow height h_f , the height of the plume core h_c and the height of the transitional layer h_t are measured.

of forces is between friction, buoyancy and the Coriolis force. The profiles of salinity and velocity structure are sampled in this zone at a downslope distance of r_s . A bulging plume head forms the third zone. The exact limits of these zones are not discussed objectively here, as POLCOMS covers all of them. Instead, it is merely ensured that r_s always falls within the main zone (i.e. zone 2 in Fig. 3.3a) of the flow.

For quantitative analysis of the cascade downslope velocity the length of the plume L_f (from the cone tip to the head of the plume) is measured as a function of time t (Fig. 3.3a). L_f is derived from the model output in 1 s intervals as an average downslope radius of the axisymmetric area covered by dense water in the bottom s -level.

Two main flow regimes are distinguished: a 2-layer flow and a stratified flow (Fig. 3.3b). Low values of diffusivity reduce diapycnal mixing and maintain a sharp interface between the dense flow and the ambient water in a 2-layer flow. High values of diffusivity,

on the other hand, cause the plume interface to become blurred and a density-stratified transitional layer is observed. The height of the flow h_f is measured in the vertical as the height above the bottom where the salinity is the mean between the salinity of the cascading and the ambient water. The two regimes are defined by comparing the height of the flow h_f , the height of the transitional layer h_t , the height of the plume core h_c and the Ekman depth H_e . A 2-layer regime is a flow with a sharp interface where the transitional layer is thinner than the plume core ($\frac{h_t}{h_c} \leq 1$) and thinner than the Ekman depth. A stratified flow with a ‘blurred’ interface has a transitional layer that is thicker than the plume core ($\frac{h_t}{h_c} > 1$). Section 3.5.5 will show that both plume and transitional layer may be considerably thicker than the Ekman depth in a stratified flow.

3.5 Results

3.5.1 Validation of the model against laboratory experiments

The POLCOMS model (see Section 3.3) was run for a range of values of the governing parameters f , Q and g' similar to those in the laboratory experiments (see Table 3.1) and two different versions of the bottom boundary condition: a ‘slip’ bottom boundary condition with the quadratic drag law ($C_D = 0.005$) and a no-slip bottom boundary condition similar to that used in the derivation of the equations for the bottom Ekman spiral. Additionally the original video footage of the laboratory experiments (courtesy of Andrei G. Zatsepin, see Fig. 3.4e for a screen shot) is inspected to validate POLCOMS qualitatively and quantitatively.

The effects of the different bottom boundary conditions are shown in Figures 3.4a to 3.4c for run ‘V’ ($f = 1.6 \text{ s}^{-1}$, $Q = 2.6 \text{ cm}^3\text{s}^{-1}$, $g' = 0.8 \text{ cm s}^{-2}$). In the run using a ‘slip’ bottom boundary condition with the quadratic drag law the plume eventually disintegrated into wobbling swirls (Fig. 3.4a); producing results very different from the laboratory experiments. Only the model run using a no-slip bottom boundary condition (Fig. 3.4b) reproduced the roughly circular plume that was observed in the laboratory

(compare with Fig. 3.4e).

Figure 3.4c shows fair quantitative agreement between the laboratory experiment and its matching model run using the no-slip bottom boundary condition (*solid line*), while the quadratic drag law (*dashed line*) does not initiate the downslope transport necessary to match the descent speed observed in the laboratory. Hence the runs with the ‘slip’ bottom boundary condition were discarded and are not discussed any further in this paper. The quadratic drag law is discussed further by Ross et al. (2002) and Cenedese and Adduce (2010).

Figures 3.4e and 3.4f show the same experiment ‘V’ as a snapshot (also after 145 s) from video footage of the original laboratory experiment and a 3-D rendering of the model output of the run using the same governing parameters. The model setup with a no-slip bottom boundary condition and near-molecular values for ν and κ (see Table 3.1) was validated against the laboratory experiments in 4 more numerical runs. These runs are presented in Figure 3.4d as plots of the downslope plume speed (plume length L_f as a function of time t). All validation runs A–D, V show good agreement between laboratory and model results. The results from those runs demonstrate how accurately POLCOMS is able to reproduce the laboratory experiments given identical cone geometry and the appropriate boundary conditions.

The mirrored side-view of the lab experiment (Fig. 3.4e) shows the formation of the bulbous dome where injected water accumulates before downslope transport is initiated. This feature is reproduced by the model and can be seen in a 3-D rendering of the model output shown in Figure 3.4f. The simulated injection of dense fluid into the model domain does not only initiate a plume, but also slightly raises the free surface elevation at the injection site near the cone tip (highlighted in Fig. 3.2 and Fig. 3.3a). The initial fluid displacement at the start of injection is detectable as concentric surface waves dispersing outwards at the wave speed ($c = \sqrt{gH}$). The upwards displacement of the

3.5. RESULTS

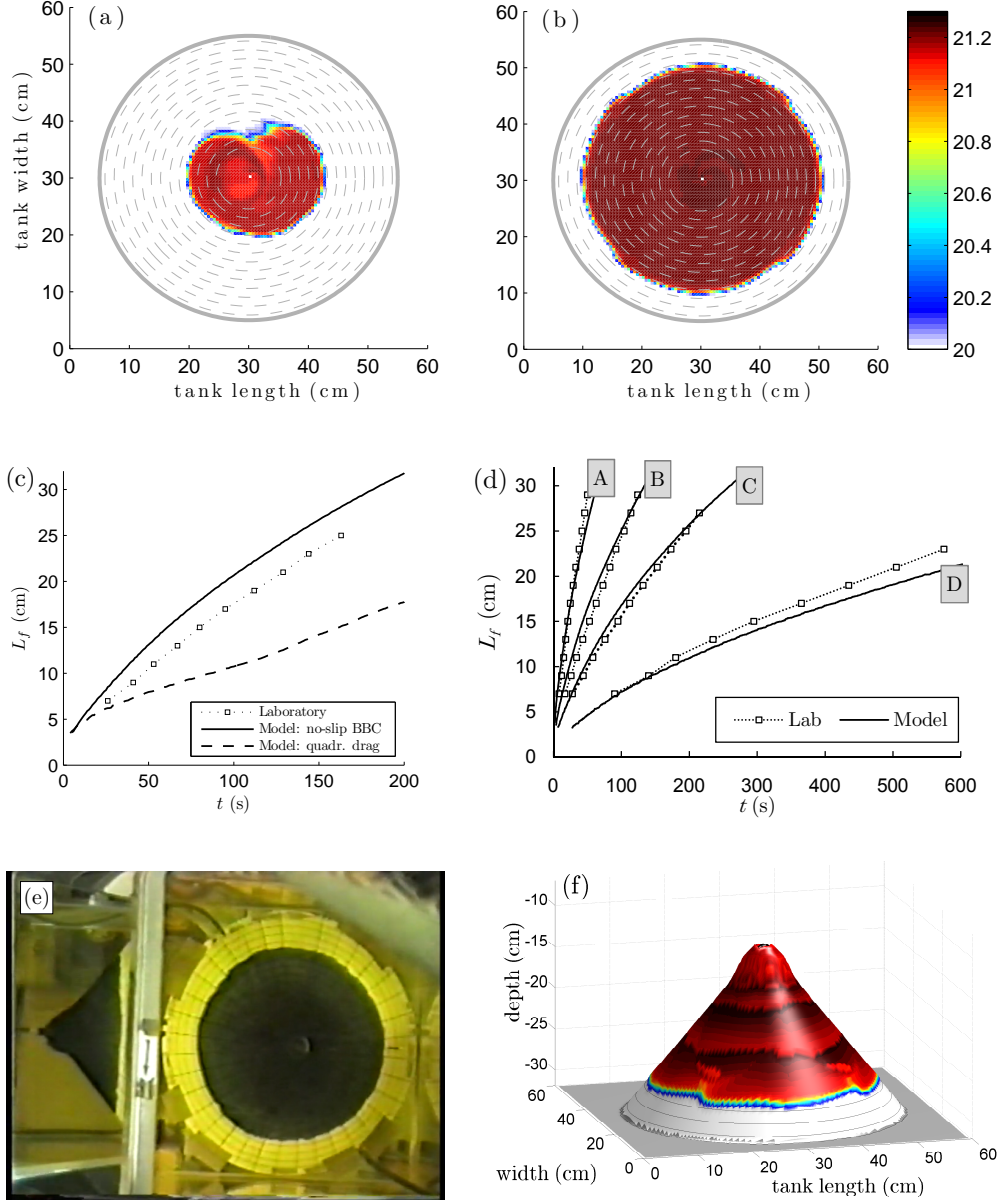


Figure 3.4: (a) – (c) Model output using different bottom boundary conditions for run ‘V’ ($f = 1.6\text{ s}^{-1}$, $Q = 2.6\text{ cm}^3\text{ s}^{-1}$, $g' = 0.8\text{ cm s}^{-2}$). Plume salinity (shaded) after 145 s - (a) quadratic drag law and (b) no-slip bottom boundary condition - and (c) downslope plume speed (Length of the plume L_f as a function of time t) in both runs compared to the matching laboratory experiment. (d) – (f) Validation of model to laboratory experiments using runs with a no-slip bottom boundary condition - (d) downslope plume speed for additional model validation runs using a range of governing parameters f (s^{-1}), Q ($\text{cm}^3\text{ s}^{-1}$), g' (cm s^{-2}): [A] 1.6, 5.0, 4.2; [B] 1.6, 2.0, 3.4; [C] 1.6, 2.0, 0.4; [D] 1.6, 0.3, 0.5. (e) video snapshot of the lab footage and (f) 3-D rendering of plume salinity after 145 s for run ‘V’ (colour scheme as in (b)).

ambient water through continued injection also forms a bulge at the surface maintained by geostrophic adjustment. This bulge is responsible for a velocity field with anti-cyclonic vorticity high above the plume. The contribution of this velocity component will be considered in the following section.

3.5.2 Comparison with reduced physics model - velocity profiles

The result from a validation run in POLCOMS, a ‘full physics’ 3-D ocean circulation model, is compared here with the simplified SZ97 model - the ‘reduced physics’ model. First, the profiles of horizontal velocity and plume salinity shown in Figure 3.5 for model run ‘V’ (see Section 3.5.1) are examined. The theoretical velocity profiles V_D and V_A are compared to the modelled velocities v_d and v_a in the downslope and alongslope direction respectively. The velocity profiles V_D and V_A are calculated using equation (A7) in Shapiro and Hill (2003) where the ambient velocity \mathbf{u}_0 was taken to be equal to the computed velocity at the free surface caused by the bulging surface (this velocity extends throughout the top layer) and the height of the interface ξ was taken to be equal to be the plume height h_f (grey circles in Fig. 3.5). The resulting equations are given in Eqs. (3.3) and (3.4).

The two panels in Fig. 3.5 are shown for different times in the model run. After 50 s (Fig. 3.5a), the salinity profile gives $h_f = 1.82 \times H_e$. After 145 s (Fig. 3.5b), the plume head is approaching the bottom of the cone at which point the plume has spread and doubled the circumference of the front from 82–165 cm leading to a decrease in plume height to $h_f = 1.55 \times H_e$. Despite different geometry and a non-steady state mode of propagation of the plume, its thickness is not significantly different from the simple estimate obtained by SH97 for a steady-state cascade on a plane slope. The decrease in plume height is a result of the increasing circumference of the plume edge during its downslope descent and is in agreement with the prediction of the SZ97 model which was derived specifically for the conical geometry.

3.5. RESULTS

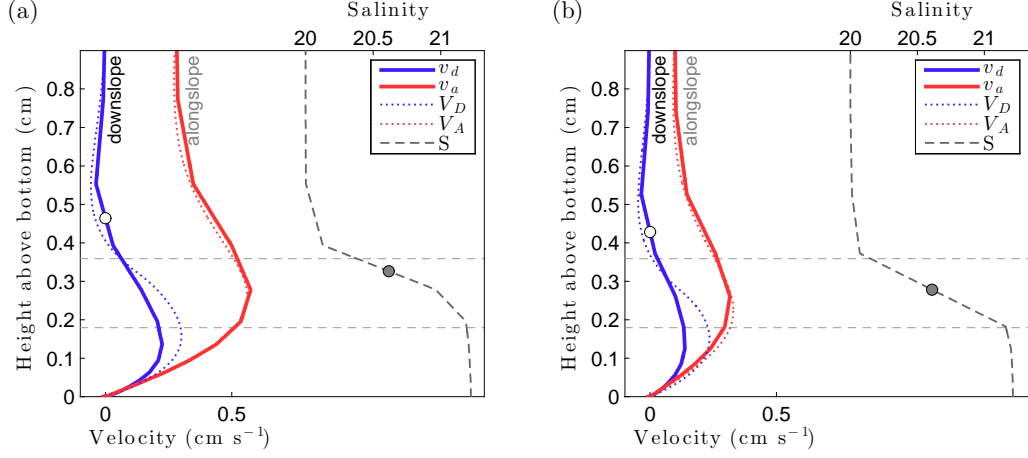


Figure 3.5: Vertical profiles of velocity (v_d - downslope, v_a - alongslope) and salinity for the model run shown in Figures 3.4f and 3.4b after (a) 50 s and (b) 145 s. White circles show layer height of positive downslope velocity. V_D and V_A show the theoretical profiles for a 2-layer flow (using Eqs. (3.3) and (3.4)). Profiles were sampled at a downslope radius r_s when the plume had reached length L_f and height h_f (grey circle) (relative to the Ekman depth $H_e = \sqrt{\frac{2\nu}{f\cos\theta}}$): (a) $L_f = 13.1$ cm, $r_s = 9.8$ cm, $h_f = 1.82 \times H_e$; (b) $L_f = 26.2$ cm, $r_s = 19.6$ cm, $h_f = 1.55 \times H_e$. The dashed horizontal lines mark the height of H_e and $2 \times H_e$.

In both snapshots, the modelled alongslope velocity v_a compares very well with the SH97 theory, while the downslope velocity v_d shows a slight deviation: the return flow is weaker and displaced upwards. This is attributed to the departure of the plume interface from the assumption in SH97 of two sharply separated layers. The salinity profiles from the model show a transition between plume and ambient water to suggest some diapycnal mixing and diffusion across the interface, which is not included in the SH97 model. This will be studied in detail in Section 3.5.5.

The maximum downslope velocities in the numerical runs are slightly smaller than predicted by the analytical theory - evident by $v_d < V_D$ (see Fig. 3.5). This is not surprising as the theoretical profile V_D was obtained for a plane slope assuming sufficient fluid supply at all times. On a conical slope, however, the circumference of the front expands over time thus reducing the available fluid supply to the leading edge which

accounts for the reduction in downslope transport over time.

3.5.3 Comparison with reduced physics model - plume speed

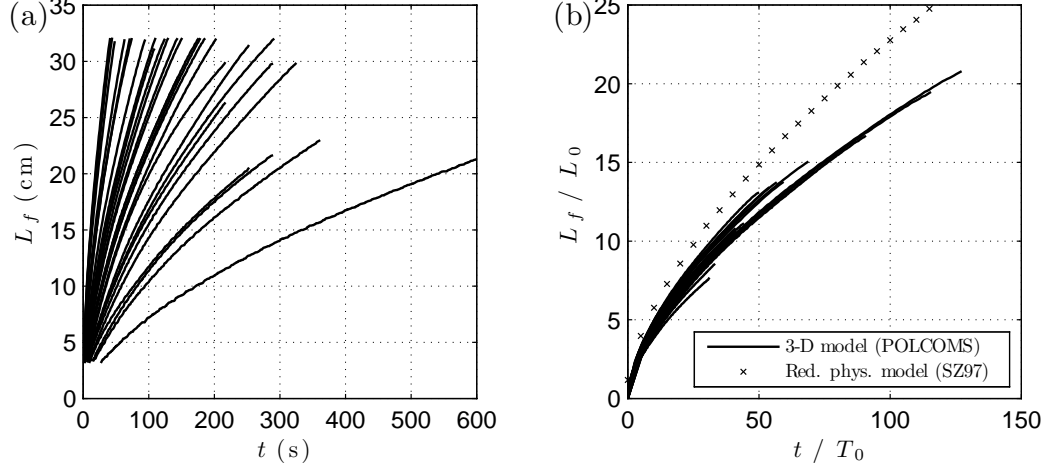


Figure 3.6: Downslope progression of the plume as a function of time in (a) dimensional and (b) non-dimensional variables (scaling given in Eq. (3.5)) for 29 model runs. The solution of the reduced physics model of Shapiro and Zatspein (1997) is also shown (as \times) in (b).

Results from 29 numerical runs (including those used for validation in Section 3.5.1) are presented in Figure 3.6 as plots of the downslope propagation of the plume (plume length L_f as a function of time t) in dimensional and in non-dimensional variables. Analysing the results in dimensional values (Fig. 3.6a) reveals that the downslope propagation speed is quite scattered between experiments where the governing parameters f , Q and g' vary (see Table 3.1). The reduced physics model, however, predicts that in non-dimensional variables using the scales in Eq. (3.5) all experiments (irrespective of the variations in f , Q and g') should collapse onto the same line.

Figure 3.6b shows the non-dimensionalised downslope propagation of the plume obtained from the SZ97 reduced physics model (\times symbols) and from the POLCOMS model runs (solid lines). Despite some scatter the model results collapse onto a curve which demonstrates that the non-dimensionalisation given in Eq. (3.5), which was de-

rived from the reduced physics model, describes the self-similar nature of the processes reproduced by the full physics model. The SZ97 reduced physics model also captures the slowing of the plume speed over time (as a result of the expansion of the leading edge of the dense front as the plume descends downslope), as already revealed by analysis of the downslope velocities in Section 3.5.2. The two curves from POLCOMS and the reduced physics model in Figure 3.6b, however, show that the reduced physics model also slightly overestimates the downslope propagation speed. A possible reason for this is that in the 3-D model the density interface is slightly ‘less sharp’ than it is assumed in a 2-layer model. The validated model will be studied in Sections 3.5.4 and 3.5.5 to investigate new parameter regimes which were not created in the laboratory or are not included in the reduced physics model.

At this point it is appropriate to keep in mind that POLCOMS uses the hydrostatic approximation. As this implies that vertical accelerations are smaller than acceleration due to gravity (Kamenkovich, 1977; Pedlosky, 1987) it is useful to compare the magnitude of these accelerations. First, the downslope plume speed is derived as $u_F = \frac{dL_f}{dt}$, where L_f is the plume length shown in Figure 3.6a. Then, the typical vertical velocity component is estimated as $W = u_F \sin \theta$ ($\theta = 39^\circ$) and its acceleration is given by $\frac{dW}{dt}$, which is compared to the reduced gravity $g' = \frac{\Delta\rho}{\rho_0}g$ (derived from the salinity within the plume). For the runs shown in Figure 3.6 the ratio of the vertical acceleration to the reduced gravity acceleration is between 0.001 and 0.01. Therefore, non-hydrostaticity is not required to capture cascading.

3.5.4 Comparison with reduced physics model - Ekman depth

This section examines the response of the cascading plume to changes in the bottom Ekman layer brought about by different values of viscosity. The vertical viscosity affects the height of the frictional boundary layer and thus the Ekman depth H_e (see Eq. (3.2)). According to the reduced physics model, in experiments varying the viscosity ν and

3.5. RESULTS

thus H_e , (i) the plume height h_f should scale with H_e (i.e. $h_f/H_e = \text{const}$), and (ii) the plume speed in non-dimensional variables should remain unchanged because the scaling in Eq. (3.5) includes H_e .

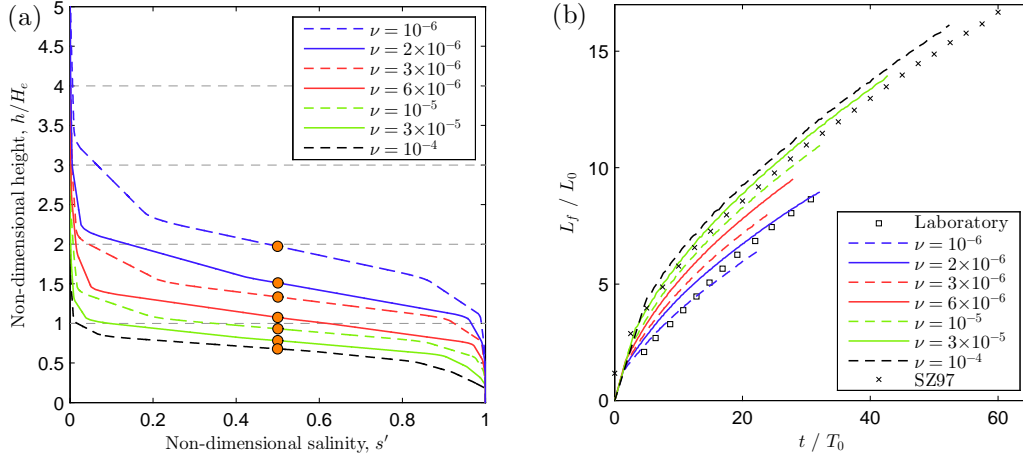


Figure 3.7: Series of model runs varying the Ekman depth $H_e = \sqrt{\frac{2\nu}{f \cos \theta}}$ by modifying vertical viscosity ν (shown in m^2s^{-1}): (a) Profiles of non-dimensional salinity $s' = (S - S_{amb})/(S_{inj} - S_{amb})$; height above bottom is normalised by the Ekman depth H_e ; orange circles mark the plume height h_f where the salinity crosses $S = \frac{1}{2}(S_{inj} + S_{amb})$; where S , S_{inj} , S_{amb} are the actual salinity and the salinity in injected and ambient fluids respectively. (b) Plume length as a function of time in non-dimensional variables according to the scaling in Eq. (3.5).

Figure 3.7 shows salinity profiles and downslope plume progression plots for a number of runs that vary the vertical viscosity by 2 orders of magnitude from $\nu = 10^{-6}$ to $10^{-4} \text{m}^2\text{s}^{-1}$ (keeping κ constant at $1.3 \times 10^{-9} \text{m}^2\text{s}^{-1}$). The shown salinity profiles (Fig. 3.7a) are averaged over a number of profiles sampled along a circle centred at the cone tip with a sampling radius $r_s = 0.5L_f$ from model output when the downslope descent of the plume is well developed ($15 \text{cm} \leq L_f \leq 20 \text{cm}$). The salinity profiles are averaged for this comparison to remove any high-frequency fluctuations from the measurements of the plume heights (e.g. ripples along the plume interface).

For the given range in viscosity, the Ekman depth increases 10-fold from $H_e = 0.127$

to 1.27 cm, and all curves in Figures 3.7a and 3.7b should theoretically collapse onto the same line. However, the absolute plume thickness increases from 0.25 to 0.86 cm, but its thickness relative to the Ekman depth (h_f/H_e) reduces from $2.0 \times H_e$ to $0.7 \times H_e$ as successive experiments increase ν (see Fig. 3.7a). The downslope propagation of the plume is shown in Figure 3.7b as plots of the non-dimensional plume length L_f/L_0 as a function of non-dimensional time t/T_0 . In this non-dimensional framework, the plume advances (relatively) slower in runs with low ν and faster in runs with high ν . This is because lower viscosity reduces friction and enhances the constraints of geostrophy making it more difficult for the rotating fluid to cross the isobaths (Note, that under weak rotation, lower friction leads to a faster descent). On the other hand, increased viscosity also affects plume volume (increase in plume height) and momentum (reduced downslope speed).

The curves in Figure 3.7b collapse to some degree, but not as well as in Figure 3.6b indicating that the 3-D model captures additional dynamics concerning the influence of viscosity on the cascading dynamics which go beyond the SZ97 reduced physics model. However, given the large range of values of ν the plume propagation still retains properties of a self-similar process. Comparisons with the full physics numerical model show that despite the simplifications employed by the reduced physics model it produces reasonable estimates of the main parameters of cascading subject to the plume having a sharp density interface with the ambient water. While it does not fully capture the effects of viscosity, the reduced physics model performs better for a 2-layer flow than in the case of a plume with a blurred interface, which is investigated in the following section.

3.5.5 Density stratification at the plume interface

The reduced physics models, e.g. SH97 and SZ97 as well as the ‘streamtube’ and ‘slab’-models by Smith (1975); Killworth (1977); Price et al. (1993) are all based on a 2-layer

3.5. RESULTS

density structure, where a homogeneous plume is overlaid by a homogeneous upper layer. While the reduced physics model by SH97 considers a 3-D velocity field and provides a solution for interfacial Ekman veering it still assumes a sharp density interface between the two fluids. This assumption has proved to be reasonable for cases of weak diapycnal mixing, e.g. observed by Visbeck and Thurnherr (2009). However, in the real ocean there are cases when density stratification in the interfacial layer between the plume and the ambient water is significant (e.g. Girton and Sanford, 2003). This regime was not investigated in the SZ97 laboratory experiments and is not included in the reduced physics model. Unfortunately there is no simple analytical model for the stratified Ekman layer (McWilliams et al., 2009). Therefore, the numerical model POLCOMS is used to investigate the behaviour of the cascading plume in the absence of a sharp interface.

A stratified interface could be simulated in a number of ways. This study employs the method of increasing the value of vertical diffusivity κ to simulate the effects of enhanced diapycnal mixing which smoothes the sharp interface and maintains a stratified flow. In a series of model runs the vertical diffusivity is modified from $\kappa = 10^{-9}$ to $10^{-5} \text{ m}^2\text{s}^{-1}$ to simulate different degrees of stratification. The degree of stratification is assessed by measuring the thickness h_t of the transition zone (Fig. 3.3b) where salinity ranges from 20% to 80% of the total salinity difference between the plume and the ambient water (marked by squares in Fig. 3.8a), while the height of the plume core h_c is defined by a plume salinity $\geq 80\%$ of the inflow salinity. The regime is considered a stratified flow with a ‘blurred’ interface when $\frac{h_t}{h_c} > 1$.

Salinity profiles for the runs with different diffusivities κ are presented in Figure 3.8a. Model runs where $\kappa = 10^{-9}$; 10^{-8} and $10^{-7} \text{ m}^2\text{s}^{-1}$ produce a plume with a sharp interface ($\frac{h_t}{h_c} = 0.56$; 0.57 and 0.65, and $Pr_v = \frac{\nu}{\kappa} = 2000$; 200 and 20 respectively), while $\kappa = 10^{-6}$; 5×10^{-6} and $10^{-5} \text{ m}^2\text{s}^{-1}$ show a blurred interface with significant stratification

3.5. RESULTS

($\frac{h_i}{h_c} = 2.26; 3.98$ and 4.00 , and $Pr_v = 2; 0.4$ and 0.2 respectively). In conclusion, the transition between the flow regimes occurs when the Prandtl number in the vertical is approximately unity, i.e. a 2-layer flow with a ‘sharp’ interface is observed at $Pr_v \gg 1$ and a stratified flow with a ‘blurred’ interface is observed at $Pr_v \lesssim 1$.

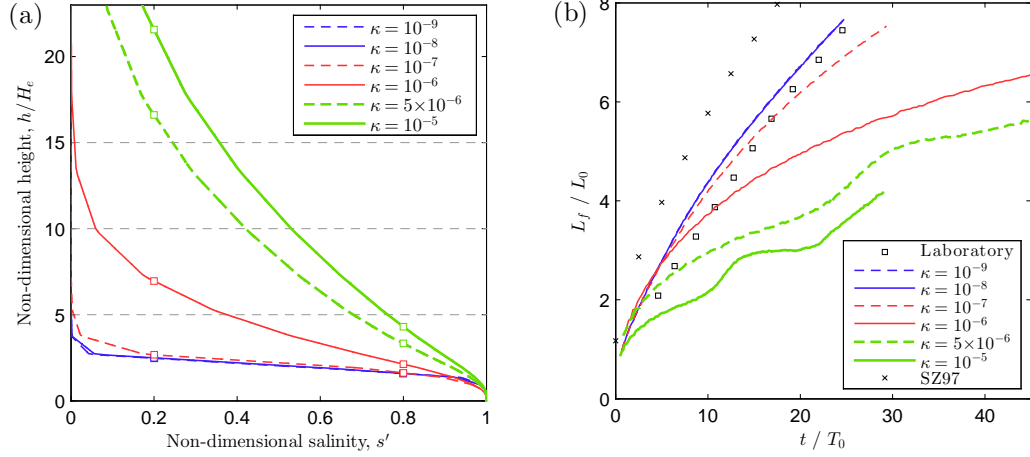


Figure 3.8: Model runs with sharply separated ($\kappa = 10^{-9}$ to $10^{-7} \text{ m}^2\text{s}^{-1}$) and stratified plumes ($\kappa = 10^{-6}$ to $10^{-5} \text{ m}^2\text{s}^{-1}$). A stratified plume interface corresponds to higher values of vertical diffusivity κ (shown in m^2s^{-1}). Non-dimensionalised (a) salinity profiles (squares indicate height where salinity is 20% and 80% of the salinity difference) and (b) plume propagation drawn as in Figure 3.7.

The downslope progression of the plume is shown in Figure 3.8b by the non-dimensional plume length L_f/L_0 versus non-dimensional time t/T_0 . In runs with a sharp interface ($\kappa = 10^{-9}$ to $10^{-7} \text{ m}^2\text{s}^{-1}$) the plume propagation speed follows a similar curve as the laboratory experiment and shows some agreement with the SZ97 reduced physics model, while the flow with a stratified interface ($\kappa = 10^{-6}$ to $10^{-5} \text{ m}^2\text{s}^{-1}$) deviates significantly from the reduced physics theory. In model runs with higher diffusivities but the same viscosities the downslope speed of the plume is reduced. At very high diffusivity ($\kappa = 10^{-5} \text{ m}^2\text{s}^{-1}$, green solid line in Fig. 3.8b) the plume propagation is very irregular. In this case the downslope flow nearly stops as most of the inflowing dense water is mixed upwards into the water column and the plume breaks up into swirls and eddies.

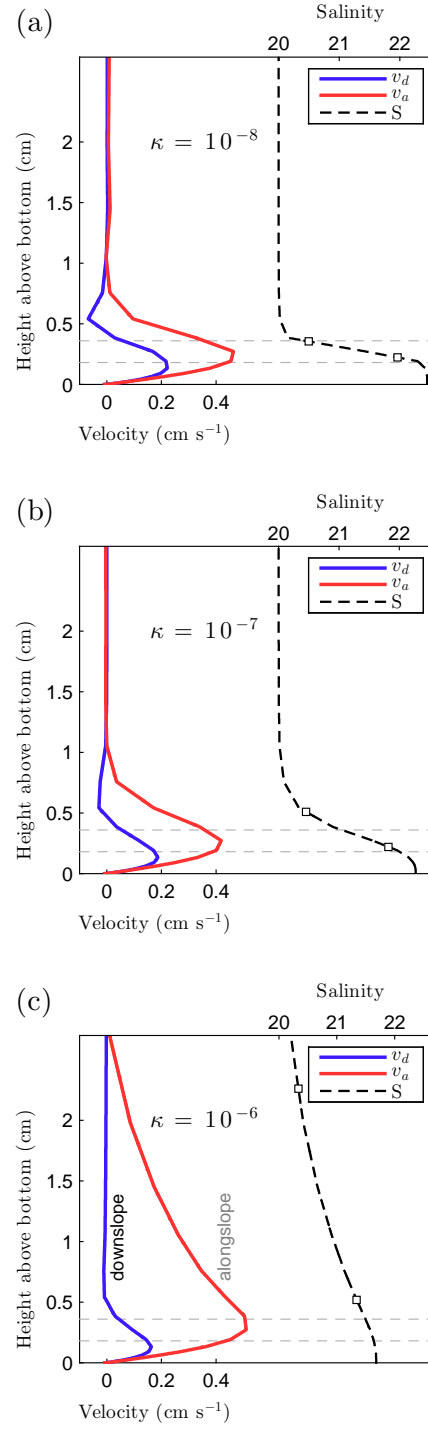


Figure 3.9: Downslope and alongslope velocity as well as salinity for runs with increased diffusivity κ (shown in m^2s^{-1}). Profiles were sampled in the middle of a fully developed plume ($L_f = 18\text{cm}$, $r_s = 9\text{cm}$) after (a) 73 s, (b) 81 s and (c) 216 s. The dashed horizontal lines mark the height of H_e and $2H_e$.

3.5. RESULTS

The velocity structure is also different between runs with sharp and diffuse plume interfaces. Figure 3.9 shows the velocity profiles for 3 runs ($\kappa = 10^{-8}$ to $10^{-6} \text{ m}^2 \text{ s}^{-1}$) that characterise this transition (top to bottom) from a 2-layer system to a stratified interface. With increasing diffusivity, it takes longer for the plume to develop and initiate the downslope descent. The profiles are therefore sampled from the model output at an equal downslope distance of $r_s = 9 \text{ cm}$ at the time when the plume has reached a downslope length of $L_f = 18 \text{ cm}$.

The alongslope velocity v_a reaches its maximum within the first 2 Ekman depths and gradually recedes to zero at approximately the height above the bottom where the salinity reaches its ambient value. In case of a stratified interface ($\kappa = 10^{-6} \text{ m}^2 \text{ s}^{-1}$) v_a shows a smooth transition from its maximum value back to zero high above the core of the plume. The maximum alongslope velocity increases in the stratified case.

The maximum downslope velocity v_d reduces as diffusivity increases (while viscosity remains the same) for the runs shown in Figure 3.9. The downslope flow does not follow the salinity profile, but remains confined to the bottom boundary layer with a height of about $2 \times H_e$. The most notable change in the downslope velocity is the reduction of the return flow (where $v_d < 0$) indicating a gradual disappearance of the interfacial Ekman layer as the stratification of the plume interface intensifies. While the velocity profiles in a sharply separated flow match a ‘modified Ekman spiral’ (see section 3.2.2, plot shown as V_D and V_A in Fig. 3.5) the profiles for a stratified interface resemble the ‘classic’ Ekman spiral. In the strongly stratified case the downslope transport within the plume behaves as it would within a 1-layer flow; hence density becomes a nearly-passive tracer. This demonstrates that intense diapycnal diffusion can slow or even arrest cascading in the specialised case where vertical diffusivity is increased while vertical viscosity remains constant.

A further set of runs investigates the behaviour of a diffuse plume (high κ) when the

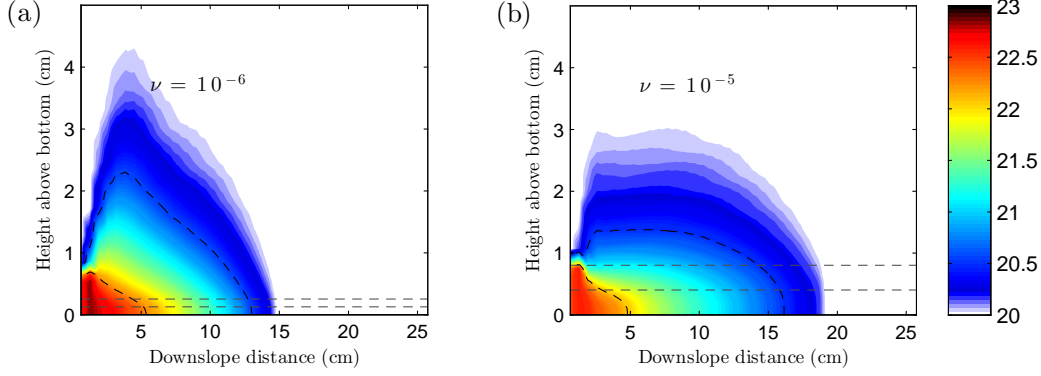


Figure 3.10: Downslope salinity cross-section of the plume at 144 s for constant vertical diffusivity $\kappa = 10^{-6} \text{ m}^2 \text{ s}^{-1}$ and varying vertical viscosity ν (shown in $\text{m}^2 \text{ s}^{-1}$). The section shows the plume thickness relative to the bottom, hence the slope is not shown. The dotted contours indicate where salinity is 20% and 80% of the salinity difference. The dashed horizontal lines mark the height of H_e and $2H_e$.

viscosity ν is also increased. Figure 3.10 shows cross-sections of the plume in runs that vary the viscosity ν from 10^{-6} to $10^{-5} \text{ m}^2 \text{ s}^{-1}$ while the diffusivity is kept constant at $\kappa = 10^{-6} \text{ m}^2 \text{ s}^{-1}$. In these runs the Ekman depth H_e ranges from 0.127 to 0.4 cm (highlighted by horizontal lines in Figure 3.10). The Prandtl Number in the vertical is $Pr_v = O(1)$ in these runs, which makes them directly relevant to observations in a turbulent ocean. The series of plots shows how the plume reacts to a thicker bottom Ekman layer: more of the water that is diffused upwards is captured by the cascading flow and transported downslope. After 144 s the plume has advanced 14.4 and 18.6 cm respectively. As a consequence of increasing Ekman depth, the plume becomes more elongated and moves faster. This series of experiments shows that the slowing effect of increased diapycnal mixing can be offset by increased viscosity because of the increased downslope transport capacity of the cascade. This is interpreted as an indication that increased mixing between the plume and the ambient water would generally increase the downslope propagation of a cascade.

3.6 Discussion

There is growing understanding of the importance of processes (including cascading) in the bottom boundary layer (BBL) in the shaping of exchanges between the shelves and the deep ocean (e.g. Huthnance et al., 2009). However, modern large-scale ocean and climate models do not resolve small-scale processes occurring in the BBL, in particular on the shelf edges and continental slopes. It is felt here that the importance of accurately representing BBL processes has so far been somewhat overlooked. A number of attempts have been made to represent the BBL by so-called ‘slab’ parametrisations (e.g. Beckmann and Döscher, 1997). However, Killworth (2003), reviewing these approaches, highlights the importance of actually resolving the physics within the BBL by saying that “*no BBL ‘slab’ parametrisation can hope to compete with a model that resolves the BBL properly, assuming adequate physics within the layer*”. The physics in the BBL is characterised by Ekman veering, which was originally derived using a no-slip bottom boundary condition. Therefore, this boundary condition is commonly used in the study of gravity currents (e.g. Özgökmen and Chassignet, 2002; Wirth, 2009). The present study employs the same bottom boundary condition and sufficient vertical resolution to accurately resolve the correct physics in the BBL, thus avoiding the parametrisation based on the quadratic drag law. The BBL is resolved here by ~ 10 computational levels.

The present POLCOMS model set-up is successfully validated against a series of laboratory experiments (Shapiro and Zatsepin, 1997), both quantitatively and qualitatively. The velocity structure and the downslope speed of the cascade also compare very well with the reduced physics model of Shapiro and Hill (1997) in those regimes where the flow has a 2-layer density structure. The results confirm that non-hydrostatic effects which are not included in POLCOMS and many other ocean models do not affect these models’ ability to represent cascading as the magnitude of vertical veloc-

ity accelerations is very small compared to the reduced gravity acceleration, even on a slope as steep as 39° . This is in contrast to common belief that non-hydrostaticity should be included in the modelling of processes with a scale of less than about 10 km (Marshall et al., 1997), but it is consistent with basic principles of geophysical fluid dynamics (Kamenkovich, 1977; Pedlosky, 1987) and was confirmed e.g. by the comparison of hydrostatic with non-hydrostatic modelling of a river plume on a scale of 1 to 20 km (McEwan, 2010). Studying the Mediterranean overflow Sannino et al. (2013) further showed that non-hydrostaticity is not required to reproduce hydraulic control at the Straits of Gibraltar. However, in the study of subgrid-scale entrainment processes by individual turbulent eddies that are not fully resolved in current models the non-hydrostatic effect could be important.

Another consideration is the potential impact of the centrifugal force on the laboratory experiments. Both POLCOMS and the reduced physics models are geophysical fluid dynamics formulations in which, as is customary, the centrifugal force of rotation is absorbed into an effective gravity. In these models, only the Coriolis force is explicitly retained. Can the centrifugal force be neglected in the laboratory experiments or should it be included as an extra term in the numerical model? Since the gravitational and centrifugal forces are the only driving forces in the problem, it seems natural to gauge the import of the centrifugal force against that of gravity by calculating the ratio of their projections along the bottom slopes, which is $f^2 r / 4g \cot \theta$, where r is the distance to the axis of rotation. This ratio is smaller than 5% in the majority of laboratory experiments and model runs, except 2 experiments where the Coriolis parameter is above 2.5 s^{-1} and even those were not outliers in the intercomparison of experimental and model results. It is argued here that it is appropriate to neglect the centrifugal force in this simulation of cascading.

Shapiro and Hill (1997) derived the cascading downslope velocity on a plane slope

3.6. DISCUSSION

as $u_F = 0.2 V_{Nof}$ and the parametrisation of the descent rate by Killworth (2001) can be written for the given slope angle ($\theta = 39^\circ$) as $u_F = \frac{1}{400} \frac{V_{Nof}}{\sin \theta} = 0.004 V_{Nof}$. On a conical slope the cascade slows over time (see Fig. 3.6a), so neither of the two theories is strictly applicable. To compare model results to these descent rate estimates, the alongslope and downslope velocities are derived as follows. For a fully developed plume, where $L_f > 6$ cm, the downslope velocity is calculated as $u_F = \frac{dL_f}{dt}$, and the alongslope velocity V_{Nof} is derived using Eq. (3.1) from the reduced gravity g' measured in the model output at a sampling radius $r_s = 0.75 L_f$ (see Fig. 3.3). An overall average ratio of all downslope and alongslope velocities from the 29 runs shown in Figure 3.6a is calculated using linear regression as $\frac{u_F}{V_{Nof}} = 0.19$ ($R^2 = 0.749$), which is surprisingly close to the ratio of $\frac{u_F}{V_{Nof}} = 0.2$ in the Shapiro and Hill (1997) formula confirming it as a useful tool for providing estimates of cascading parameters from observations.

In a 2-layer regime both the downslope and alongslope flows are confined to a thin layer at the bottom with a thickness of about 2 Ekman depths. This agrees with analytical theories applicable for this regime (Shapiro and Hill, 1997; Wåhlin and Walin, 2001) and is consistent with ocean observations (e.g. Baringer and Price, 1997; Visbeck and Thurnherr, 2009) and laboratory studies (Cenedese et al., 2004). The flow remains a 2-layer structure even for radical changes of viscosity by 2 orders of magnitude. While the plume does not reach the full height of the Ekman layer for higher viscosities, an increase in viscosity causes a thicker plume in real terms. This appears in contrast to Legg et al. (2008) who found that as viscosity increased plume thickness was reduced. Their different numerical configuration which did not fully resolve the Ekman layer could be seen as the reason for a significant influence of uncontrolled numerical viscosity and diffusivity on their results.

In cases of strong diapycnal mixing, the plume interface blurs significantly and the 2-layer reduced physics model no longer applies. The interface can be defined as blurred

when the transitional layer between plume and ambient density is much thicker than the core of the plume itself. This definition is practical to separate different regimes with very different properties.

In the case of a smooth interface the plume diffuses upwards and becomes much thicker than the bottom Ekman layer, and much of the dense water moves out of reach of the boundary layer to which the downslope transport remains confined, but is involved in alongslope transport. A comparable situation has been observed in overflow plumes where transport increases downstream due to entrainment (Girton and Sanford, 2003; Matt and Johns, 2007).

The velocity profiles for a 2-layer density structure match the ‘modified’ Ekman spiral (Shapiro and Hill, 1997, 2003) and show the presence of an interfacial Ekman layer evident by a return flow just above the plume interface. In a flow with a strongly blurred interface the alongslope velocity reacts quickly to the smooth density transition, while the downslope velocity remains confined to the bottom Ekman layer. It cannot be confirmed whether the downslope velocity would eventually (i.e. after a longer experiment time) adapt its profile to the density structure in the same way as the alongslope velocity. It would be interesting to investigate this difference in adaptation time for dense flows over corrugated bathymetry, such as canyons, where the absolute direction of the downslope and alongslope components of the flow changes frequently. The lack of a return flow in case of a diffuse plume is an indication for the dissolution of the interfacial Ekman layer under the influence of strong diapycnal mixing.

Previous studies have considered a sharp interface between the flow or reservoir of dense water and the ambient water for reservoirs or flows that are much thicker than the Ekman depth (e.g. Wåhlin and Walin, 2001; Shapiro and Hill, 2003; Wirth, 2009). They found that a thin layer (of height $h \approx H_e$) of dense water forms near the bottom and starts to move downslope while the main body of dense water is confined to alongslope

motion according to Nof (1983). It is shown that this is not true for cases where a thick plume (of height $h \gg H_e$) is formed by the upward diffusion of dense plume water and the interface is strongly blurred. It is further found that the downslope flow does not form a thin layer of the order of the Ekman layer height H_e and the present findings only agree to the extent that downward propagation is slowed or even arrested.

A thick, but diffused cascade is shown to be strongly affected by the Ekman depth as the slowing effect on downslope motion of high diffusivity κ can be compensated for by increased viscosity ν . The results presented in Section 3.5.5 show that the speed of downslope propagation increases in a regime where diffusivity and viscosity are both increased which simulates the effects of increased turbulence. The diffuse plume moves downslope faster because the height of the Ekman layer encompasses most of the plume such that $h \sim O(H_e)$. This is consistent with figure 4b in Shapiro and Hill (1997) which shows that the entrainment process speeds up downslope propagation of the plume in a reduced physics model. The acceleration of a cascade due to mixing has practical implications as oceanic turbulence tends to affect diffusivity and viscosity to a comparable degree (as ν increases, so does κ and vice versa) and therefore suggests that increased diapycnal mixing in a highly turbulent regime increases downslope transport. This finding helps explain observations in areas of tidally generated turbulence in the Ross Sea (Padman et al., 2009). Chapter 5 will return to the subject of tidal mixing and its influence on dense water plumes.

3.7 Conclusions

The results in this chapter show that the correct resolution of bottom boundary layer physics is critical to successfully model cascading, while non-hydrostaticity is not required to capture the descending plume. The traditional square drag law fails to capture the Ekman veering at the bottom boundary and is shown to insufficiently represent bottom friction, while the 3-D numerical model with a no-slip bottom boundary condition

3.7. CONCLUSIONS

and increased vertical resolution near the bottom was successfully validated against laboratory experiments.

The simulations using POLCOMS highlight the areas of applicability of the previous reduced physics theory, which is only applicable to a 2-layer flow with a sharp interface between the cascading plume and the ambient water. A dense flow with a stratified interface was investigated with the full physics 3-D numerical model, POLCOMS. The results show that downslope transport is reduced when the plume interface is strongly diffused, but enhanced in a regime that simulates the effects of increased turbulence where diffusivity and viscosity are both increased.

Chapter 4

The piercing of the Atlantic Layer by an Arctic shelf water cascade in an idealised study inspired by the Storfjorden overflow

4.1 Introduction

The Storfjorden overflow, a plume of dense brine-enriched water, results from sea ice production in the Storfjorden polynya in Svalbard (Schauer, 1995; Skogseth et al., 2005a). The cold, dense water cascades into Fram Strait and encounters a layer of warm, saline Atlantic Water at depths between approximately 200 and 500 m. In some years the plume ‘pierces’ the Atlantic layer and continues to sink into the deep Fram Strait while in other years it remains ‘arrested’ at Atlantic Layer depths (Quadfasel et al., 1988; Schauer, 1995; Schauer and Fahrbach, 1999; Schauer et al., 2003). The eventual depth of the cascaded waters has a proven effect on the maintenance of the Arctic halocline (when the plume is arrested) and (when piercing occurs) the ventilation of the deep Arctic basins (Rudels et al., 2005).

It has been unclear what parameters control the fate of the plume. This study therefore addresses the following questions:

- Can it be predicted when the cascade will be arrested and when it will pierce the Atlantic Layer from the knowledge of the ambient conditions and the source water parameters alone?

- How does the cascading regime respond to changes in the flow rate and/or the salinity of the overflow waters?

This chapter is structured as follows: Section 4.2 describes the configuration of the model and the experimental design. Results are presented and discussed in Section 4.3. The conclusions to address the above questions are given in Section 4.4.

4.2 NEMO model setup

An idealised ocean basin with a conical slope is modelled using a 3-D ocean circulation model based on NEMO-SHELF (O’Dea et al., 2012). The adaptations for shelf seas studies by O’Dea et al. (2012) to the standard NEMO code by Madec (2008) and custom adaptations specifically introduced for this study are described in Section 2.3.

4.2.1 Model configuration

The model (see Section 2.3) is set up at 1 km horizontal resolution on a 109×109 grid. In the vertical, the s_h -coordinate system (Section 2.3.1) is configured with 42 levels. Over a bottom layer of constant thickness (60 m) 16 levels are evenly spaced to give a near-bottom vertical resolution of at least 3.75 m. The s_h -levels to coincide with the interfaces between the ambient water masses are placed at 200 and 500 m and a third s_h -level is inserted at 800 m to form a virtual sea bed for the levels below the deepest interface at 500 m. Vertical resolution in the interior ranges from 30 to 60 m (Figs. 2.1a and 2.1b). The s_h -levels in this study are smoothed with values of θ equal to 4, 6 and 8 at the depths of 200, 500 and 800 m respectively. The baroclinic time step is 40 s with time splitting for the barotropic component every 20 steps.

The model’s river inflow scheme is used to simulate an overflow of dense water at the top of the slope. The ‘river’ injection grid cells are arranged over a 50 m-thick layer above the bottom at 115 m depth in a 3 km-wide ring around a central ‘island’ of land grid cells (Fig. 4.1a). The island’s vertical walls avoid a singularity effect at

4.2. NEMO MODEL SETUP

the centre of rotation and prevent inflowing water from sloshing over the cone tip. A constant flow rate Q (in m^3s^{-1}) of water at a given salinity S is evenly distributed over all injection grid cells. The inflowing water is marked with a passive tracer ‘PTRC’ (using the MYTRC/TOP module) by continually resetting the PTRC concentration to 1.0 at the injection grid cells.

4.2.2 Model geometry and water masses

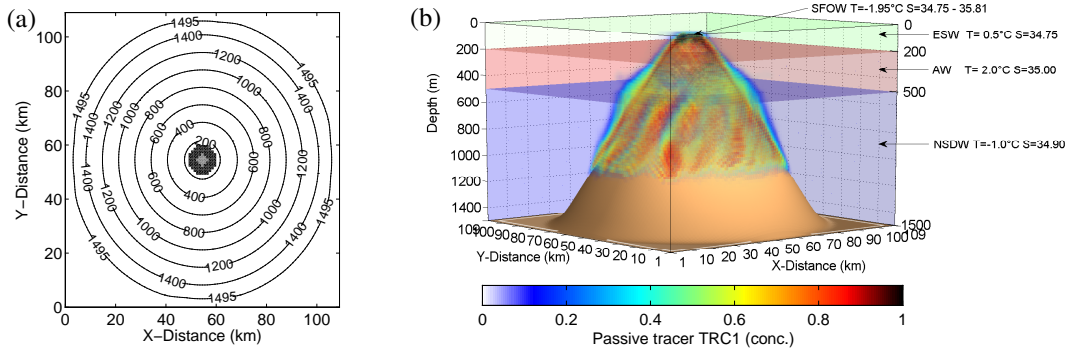


Figure 4.1: (a) Depth contours of the model bathymetry with a conical slope at its centre. The grid cells of the dense water inflow (solid black) are arranged around a central ‘island’ (grey). (b) 3-D schematic of the model domain with the ambient water masses in their initial state: East Spitsbergen Water (ESW), Atlantic Water (AW), Norwegian Sea Deep Water (NSDW) and Storfjorden Overflow Water (SFW). The plume of SFW during one of the numerical experiments is shown as a volume rendering of passive tracer concentration (colour scheme below plot).

4.2.3 Model validation

Prior to the model experiments described here the customised NEMO-SHELF code (Section 2.3) was applied to the model experiments of Wobus et al. (2011, see Chapter 3). Those model runs successfully validated the results against the laboratory experiments by Shapiro and Zatsepin (1997) and NEMO was able to match the laboratory results with the same degree of confidence as the POLCOMS model in Chapter 3. In an injection-less control run any spurious velocities were found to remain well below 1 cm s^{-1} indicating the accuracy of the horizontal pressure gradient scheme. Numerical diffusion at horizontal isopycnals was also effectively controlled.

At this point it seems appropriate to add a brief note on the condition of ‘hydrostatic inconsistency’ which was brought to the attention of the ocean modelling community by Haney (1991) and others (see Sikirić et al., 2009, for a review). Written for a constant slope angle θ and bathymetric depth D , Haney (1991) states that if $R = \left| \frac{\sigma}{D} \frac{\Delta x \tan \theta}{\delta \sigma} \right|$, the model should satisfy $R \leq 1$ for the finite difference scheme to be hydrostatically consistent and convergent. Mellor et al. (1994), however, showed that this condition strongly depends on the exact nature of the numerical scheme, and convergent results can be obtained even for values $R \gg 1$. In fact, in the POLCOMS model used in Chapter 3 the worst-case was $R = 101$, yet a close agreement was achieved between model and laboratory experiments. In the present study we get $R \leq 8$, which gives confidence in the results.

The conical slope has an angle of 1.8° to capture the bathymetry of Svalbard’s western continental slope. The conical geometry acts like a near-infinite slope wrapped around a central axis (Fig. 4.1). An advantage of a conical slope is that rotating flows can be studied for long periods of time without the plume reaching any lateral boundary, thus avoiding possible complications with boundary conditions in a numerical model. The depth ranges from 115 m at the flattened tip of the cone to 1500 m at its foot. The maximum model depth of 1500 m is shallower than Fram Strait, but deep enough to observe whether the modelled plume has descended past the depth range of the Atlantic Layer.

4.2.4 Ambient and initial conditions

The ambient conditions in the model ocean are based on the three main water masses that the descending plume encounters successively (cf. Fer and Ådlandsvik, 2008). The surface layer of East Spitsbergen Water (ESW) is typical of winter conditions, the middle layer of Atlantic Water (AW) is typical of early spring and the deep layer of Norwegian Sea Deep Water (NSDW) is based on late spring climatology (World Ocean Atlas

2001, Conkright et al., 2002). Ambient waters (Fig. 4.1b) are stagnant at the start of each run and no momentum forcing is applied.

A fourth water mass, which is called here Storfjorden overflow water (SFOW), is introduced as a continuous flow at the shallowest part of the slope in 115 m (Fig. 4.1a), which is the sill depth of the Storfjorden. As SFOW is the result of sea ice formation and brine rejection its temperature is always set to approximate freezing point, $T = -1.95^{\circ}\text{C}$. The injected flow is further characterised by a prescribed salinity S and flow rate Q which vary between model runs, which aim to represent previously observed conditions.

4.2.5 Experiment design

Using observations of the densest waters found within the fjord during 1981 to 2002 (see table 3 in Skogseth et al., 2005b) the inflow salinity S is varied from 34.75 to 35.81. The flow rate Q is varied from 0.01 to 0.08 Sv, based on observations at the sill of a mean volume transport of 0.05 to 0.08 Sv (Schauer and Fahrbach, 1999; Skogseth et al., 2005a; Geyer et al., 2009). No attempt is made in the present study to model the dense water formation process itself. The flow rate Q and the salinity S of the simulated overflow waters are intended to capture the parameters of the SFOW behind and at the sill.

A total of 45 model runs are performed using the NEMO model setup described in Section 4.2. The dense water parameters are varied while the initial conditions are identical in all runs. All runs are integrated over a duration of 90 days. Appendix D describes how the model code was compiled and run under the Windows operating system.

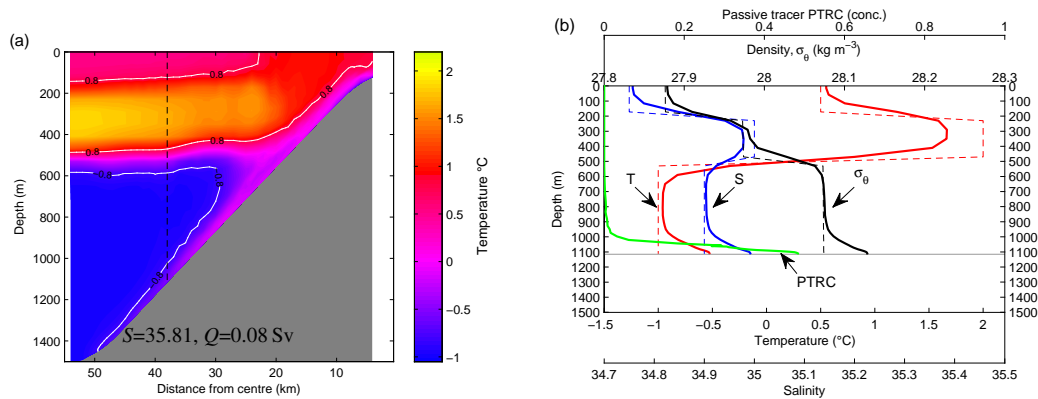


Figure 4.2: (a) Temperature section (after 24 days) in a model run with strong cascading. The isotherms drawn at -0.8 and 0.8°C (white lines) are an approximate boundary between the cascade and ambient water where their slope is parallel to the bottom. The vertical dashed line marks the sampling of the vertical profiles in (b): temperature (red), salinity (blue), density (black) and PTRC concentration (green). Initial conditions are shown as dashed lines.

4.3 Results and Discussion

4.3.1 Plume evolution

With the start of each experiment the injected dense water forms a plume of approximately circular shape which spreads downslope. At the leading edge of the plume wave-like baroclinic instabilities gradually develop into meanders and eddies reaching a width of 8 – 12 km. At depth, where the Rossby radius of deformation is approx. $R_o = 4\text{ km}$, the size of these features thus conforms to the expected horizontal length scale of $2 \times R_o$ to $3 \times R_o$ (Griffiths and Linden, 1982).

On its descent the plume successively encounters East Spitsbergen Water (ESW) near the sill, then Atlantic Water (AW) at intermediate depths and finally Norwegian Sea Deep Water (NSDW). Fig. 4.2a shows a temperature cross-section where the plume has penetrated all three ambient layers and reached the bottom of the slope. A thin warm layer above the bottom is emphasised by the -0.8°C isotherm parallel to the slope between 700 and 1400 m. This is a sign of the plume warming as it passes through

warm AW during its descent yet retaining a sufficient density contrast to continue to greater depths. This signature of a near-bottom temperature and salinity maximum was observed in Fram Strait by Quadfasel et al. (1988).

The cascade in Fig. 4.2a also drives warm water from the Atlantic Layer to the surface. The upwelling effect of a cascade is not caused by continuity alone (ambient water moving upwards to replace descending colder water) as it would not be induced if the same amount of dense water were injected in the deepest layer. Upwelling is also a result of velocity veering in the bottom and interfacial Ekman layers as shown by Shapiro and Hill (1997) in a $1\frac{1}{2}$ -layer model and by Kämpf (2005) in laboratory experiments.

The ambient waters in Fig. 4.2a are also modified as a result of the dense water flow. The surface layer of ESW has been displaced from the inflow area and the Atlantic Layer shows signs of cooling near the slope. The 0.8°C isotherms which may serve as both shallow and deep boundaries of the Atlantic Layer have been displaced upwards indicating an upwelling of warm water towards the surface. This is in contrast to the control run without any dense water injection where all isotherms remain horizontal.

The vertical profiles at a location in just over 1100 m depth (Fig. 4.2b) show the plume as a density maximum above the bottom. A similar gradient is evident in the temperature and salinity profiles. The PTRC concentration is used to determine the plume height h_F in the following section.

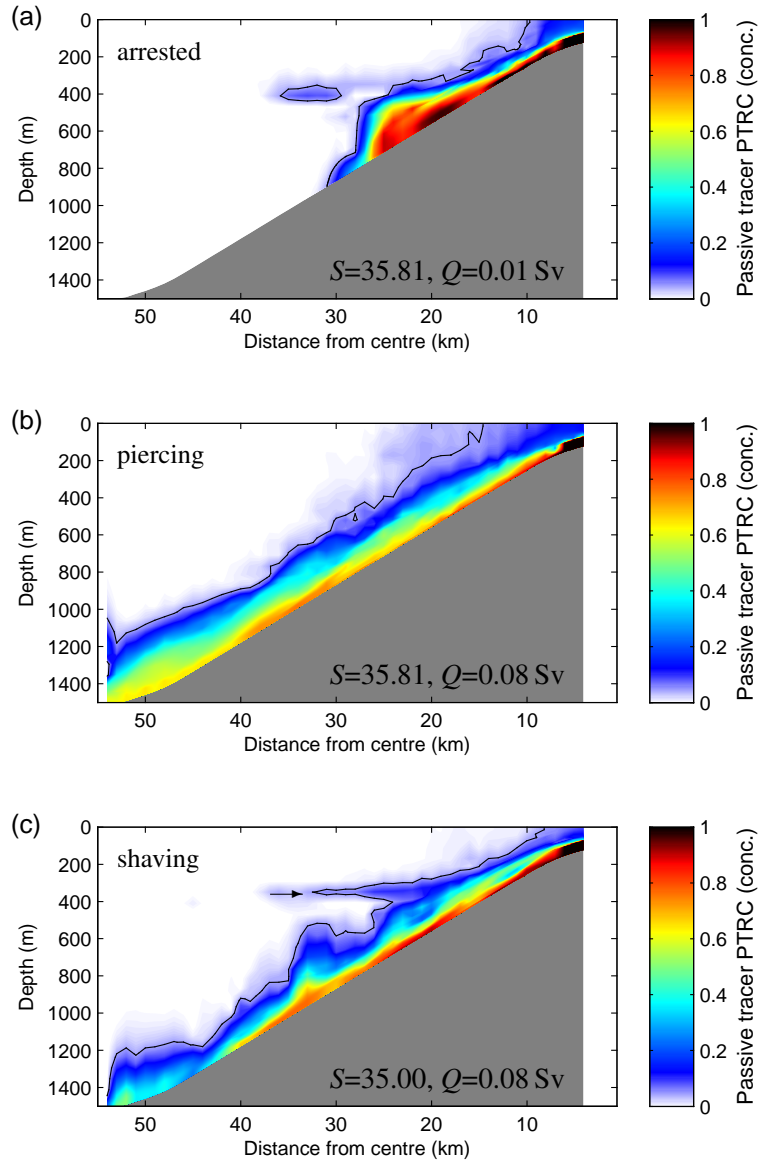


Figure 4.3: Cross-section of tracer concentration after 90 days from experiments with three different combinations of SFOW inflow salinity S and flow rate Q . In all cases the initial SFOW density is higher than the density of NSDW in the bottom layer. The concentration $\text{PTRC} = 0.05$ is shown as a solid contour. The arrow in (c) points out an example of a lateral intrusion described as ‘shaving’ by Aagaard et al. (1985).

4.3.2 Cascading regimes

The numerical experiments reveal three regimes of cascading: (i) ‘arrested’ - the plume remains within or just below the Atlantic Layer (Fig. 4.3a), (ii) ‘piercing’ - the plume pierces the Atlantic Layer and continues to the bottom of the slope (Fig. 4.3b) and an intermediate regime (iii) ‘shaving’ - where a portion of the plume detaches off the bottom, intrudes into the Atlantic Layer while the remainder continues its downslope propagation (Fig. 4.3c). The latter regime was so named by Aagaard et al. (1985) who inferred it from observations. The arrested regime was observed in 1994 (Schauer and Fahrbach, 1999), while the piercing regime was observed in 1986 (Quadfasel et al., 1988), in 1988 (see Akimova et al., 2011) and in 2002 (Schauer et al., 2003).

For the ‘arrested’ and ‘piercing’ regimes we examine the thickness of the plume h_F which is derived from vertical profiles of PTRC as the height above the bottom where the concentration drops below 50% of the value reached at the seabed. Values are averaged in space along the plume edge and up to 10 km behind the plume front and in time over the 20 days before the flow reaches 1400 m depth.

The plume thickness in the model varies between 30 and 228 m, which is generally greater than observations in Fram Strait of a 10-100 m thick layer of Storfjorden water at depth (Quadfasel et al., 1988). The disparity appears smaller for this model than in modelling studies by Jungclauss et al. (1995) and Fer and Ådlandsvik (2008) who reported $h_F \approx 200\text{--}400$ m. However, it should be noted that the plume thickness is very sensitive to the chosen tracer threshold value, and the plume thickness for this model could fall into the same range as Fer and Ådlandsvik (2008) if a different threshold was used. We therefore do not overemphasise the detailed comparison of the modelled plume height with actual observations of the Storfjorden plume as many aspects of this model setup are idealised and not designed to replicate observed conditions.

The absolute plume thickness h_F is normalised by the Ekman depth H_e defined here

4.3. RESULTS AND DISCUSSION

as $H_e = \sqrt{2\nu/f \cos \theta}$ for a given slope angle θ and the vertical viscosity ν (calculated here by the GLS turbulence closure scheme) which is averaged over the core of the plume. The vertical diffusivity κ is also shown to assess the vertical Prandtl number $Pr_v = \nu/\kappa$ which is $\sim \mathcal{O}(1)$.

The Entrainment ratio is calculated as $E = w_e/u_F$, where w_e is the entrainment velocity dh_F/dt (Turner, 1986) and $u_F = dL/dt$ is the downslope speed (L is the distance of the plume edge from the inflow) of the flow. E is calculated over the time taken by the flow until it has reached 1400 m depth (or until the end of the experiment if this depth is not reached). The results for both subsets of experiments are summarised in Table 4.1.

Table 4.1: Characteristics of the plume in the ‘arrested’ and ‘piercing’ regime: plume height h_F , vertical viscosity ν , vertical diffusivity κ , Ekman depth H_e , normalised plume height $\frac{h_F}{H_e}$ and entrainment ratio E . One standard deviation is given in brackets.

	arrested (10 runs)	piercing (16 runs)	
h_F	166 (43)	44 (11)	m
ν	9.2 (2.9)	5.7 (0.4)	$\times 10^{-3} \text{ m}^2 \text{ s}^{-1}$
κ	9.6 (4.2)	6.3 (0.4)	$\times 10^{-3} \text{ m}^2 \text{ s}^{-1}$
H_e	11 (1.7)	9 (0.3)	m
$\frac{h_F}{H_e}$	14.9 (4.2)	4.8 (1.0)	
E	5.4×10^{-3} (2.6×10^{-3})	0.33×10^{-3} (0.29×10^{-3})	

Values for vertical viscosity ν and Ekman depth H_e are typical for oceanic scales (e.g. Cushman-Roisin and Beckers, 2011) and they are similar in both regimes. However, the plume height h_F differs considerably between both sets of experiments. A piercing plume is on average 44 m thick towards the bottom end of the flow compared to 166 m in experiments where the plume is arrested. An explanation is found in the entrainment ratio E which changes with the depth level of the plume head and thus varies through time. The value of E is larger while the plume head is at the depth level of a density interface in the ambient waters (which is a considerable portion of the total ex-

periment time in arrested runs). Its value is smaller during the plume's descent through a homogeneous layer of ambient water (as it does for the majority of the experiment time in piercing runs).

Based on buoyancy considerations alone one could expect that the incoming plume with a density greater than the density of the bottom layer (in this case for $S > 34.85$) should always penetrate into that layer. However, the results show that this is not the case because of mixing processes that result in density changes of the plume as it progresses downslope over time.

4.3.3 Rate of descent

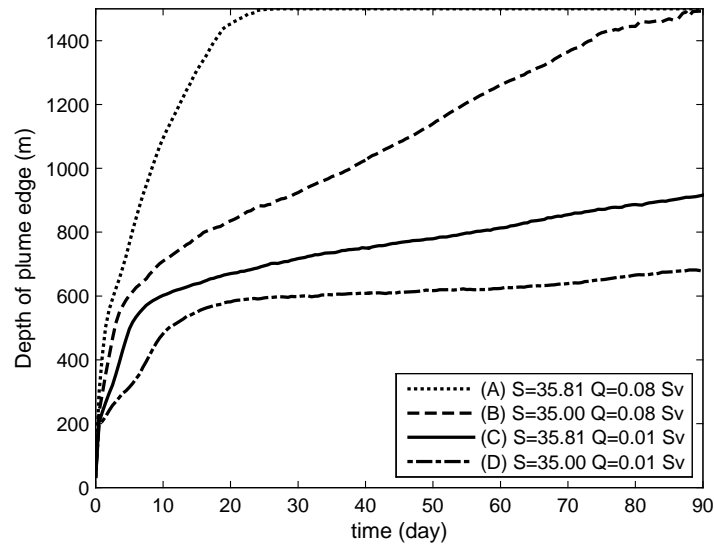


Figure 4.4: Downslope progression of the plume edge for four example runs with varying S and Q .

In this section, we examine the downslope propagation of the plume. Fig. 4.4 shows the depth of the plume edge over time calculated from the deepest appearance of a concentration $\text{PTRC} \geq 0.05$ in the bottom model level. The plume speed slows over time, which is due to (i) the increase in diameter of the leading edge as the plume progresses further down the cone which causes a thinning of the plume that in turn increases the effect of drag on the plume and (ii) the mixing of the plume with ambient

4.3. RESULTS AND DISCUSSION

waters resulting in a gradual decrease in density contrast, especially upon encountering the transition between ambient water masses at 200 and 500 m. The plume in run D ($S=35.00$, $Q=0.01$ Sv, Fig. 4.4) slows noticeably at the 200 m interface (between ESW-AW), while the other runs are less affected at this depth level. In all runs the plume is slowed upon encountering the 500 m depth level of the AW-NSDW interface, but the plume in run A which has the strongest inflow ($S=35.81$, $Q=0.08$ Sv) is least affected and reaches the bottom of the slope after only 20 days. Fig. 4.4 demonstrates that plumes with different initial parameters spend varying lengths of time flowing through and mixing with the different layers of ambient water which affect the final fate of the plume (see sections 4.3.4 and 4.3.5).

At this point it is appropriate to include a note on the relationship between the downslope speed of the plume front and its alongslope speed. For each model run the downslope speed u_F is calculated for the latter part of the experiment when the descent rate is roughly constant - from 20 days (or when the plume edge has passed 800 m depth, if earlier) until the end of the model run or when the plume edge has reached 1400 m (cf. Fig. 4.4). For the same time period we also derive the reduced gravity $g' = g \frac{\Delta\rho}{\rho_0}$ based on the density gradient across the plume front. Experiments where the plume is arrested and g' is close to 0 or even negative (due to the overshoot at the front) are excluded.

Fig. 4.5 compares the downslope velocity component u_F to the alongslope component $V_{Nof} = \frac{g'}{f} \tan \theta$ (Nof, 1983), where $f = 1.415 \times 10^{-4} \text{ s}^{-1}$ is the Coriolis parameter and $\theta = 1.8^\circ$ is the slope angle. An overall average ratio of all downslope and alongslope velocities from all 45 runs is calculated using linear regression as $\frac{u_F}{V_{Nof}} = 0.19$ ($R^2 = 0.977$) which is surprisingly close to the ratio of $\frac{u_F}{V_{Nof}} = 0.2$ given by Shapiro and Hill (1997) as a simplified formula for the quick estimation of cascading parameters from observations. The Killworth (2001) formula for the rate of descent of a gravity current can be written for the given slope angle ($\theta = 1.8^\circ$) as $u_F = \frac{1}{400} \frac{V_{Nof}}{\sin \theta} = 0.08 V_{Nof}$ making the

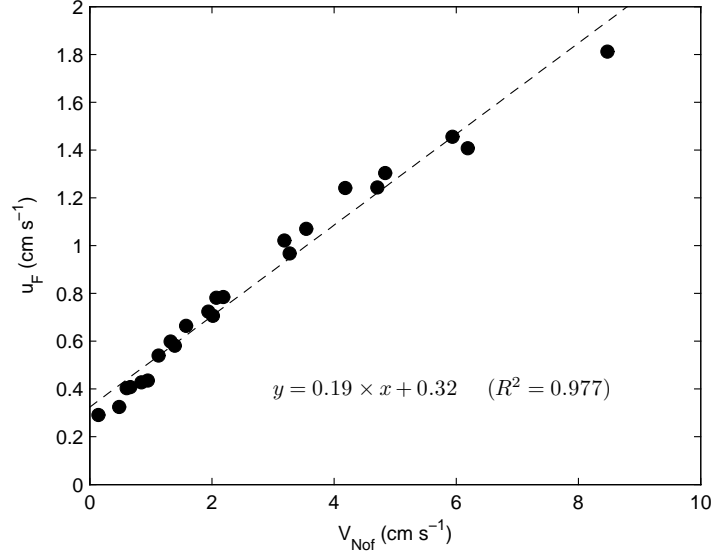


Figure 4.5: Correlation between the alongslope geostrophic velocity scale ($V_{Nof} = \frac{g'}{f} \tan \theta$) and the downslope velocity of the plume front (u_F). Data are plotted for runs with a positive density gradient at the plume front.

modelled downslope velocities approximately $2.4\times$ greater than Killworth's prediction.

Shapiro and Hill (1997) developed their formula for a $1\frac{1}{2}$ -layer model of cascading on a plane slope and assuming a sharp separation between ambient water and a plume with a normalised thickness of $\frac{h_F}{H_e} \approx 1.78$. A ratio of $\frac{u_F}{V_{Nof}} = 0.19$ was computed for those runs with a positive density gradient at the plume front, which naturally puts them in the 'piercing' category. The normalised plume height averaged over those runs is $\frac{h_F}{H_e} = 4.7$, which indicates a more diluted plume than assumed for the Shapiro and Hill (1997) model.

Chapter 3 investigated the flow of dense water down a conical slope in absence of density gradients in the ambient water. It was found that prescribing enhanced vertical diffusion slows the downslope progression of the plume, while prescribing enhanced vertical viscosity increases downslope transport (given sufficient supply of dense water). The agreement with the descent rate prediction of Shapiro and Hill (1997) was shown in Chapter 3 not to be limited to cascades with a sharp interface and a thin plume with $h_F \sim$

$\mathcal{O}(H_e)$, but also applicable to thick and diffuse plumes as long as the vertical diffusivity κ and viscosity ν are of approximately the same magnitude (i.e. a vertical Prandtl number of $Pr_v \sim \mathcal{O}(1)$). This study confirms the Shapiro and Hill (1997) descent rate formula in a model using the GLS turbulence closure scheme (rather than prescribed turbulence). The agreement in Fig. 4.5 is explained by plumes of the ‘piercing’ regime meeting the aforementioned Prandtl number criterion (see Table 4.1).

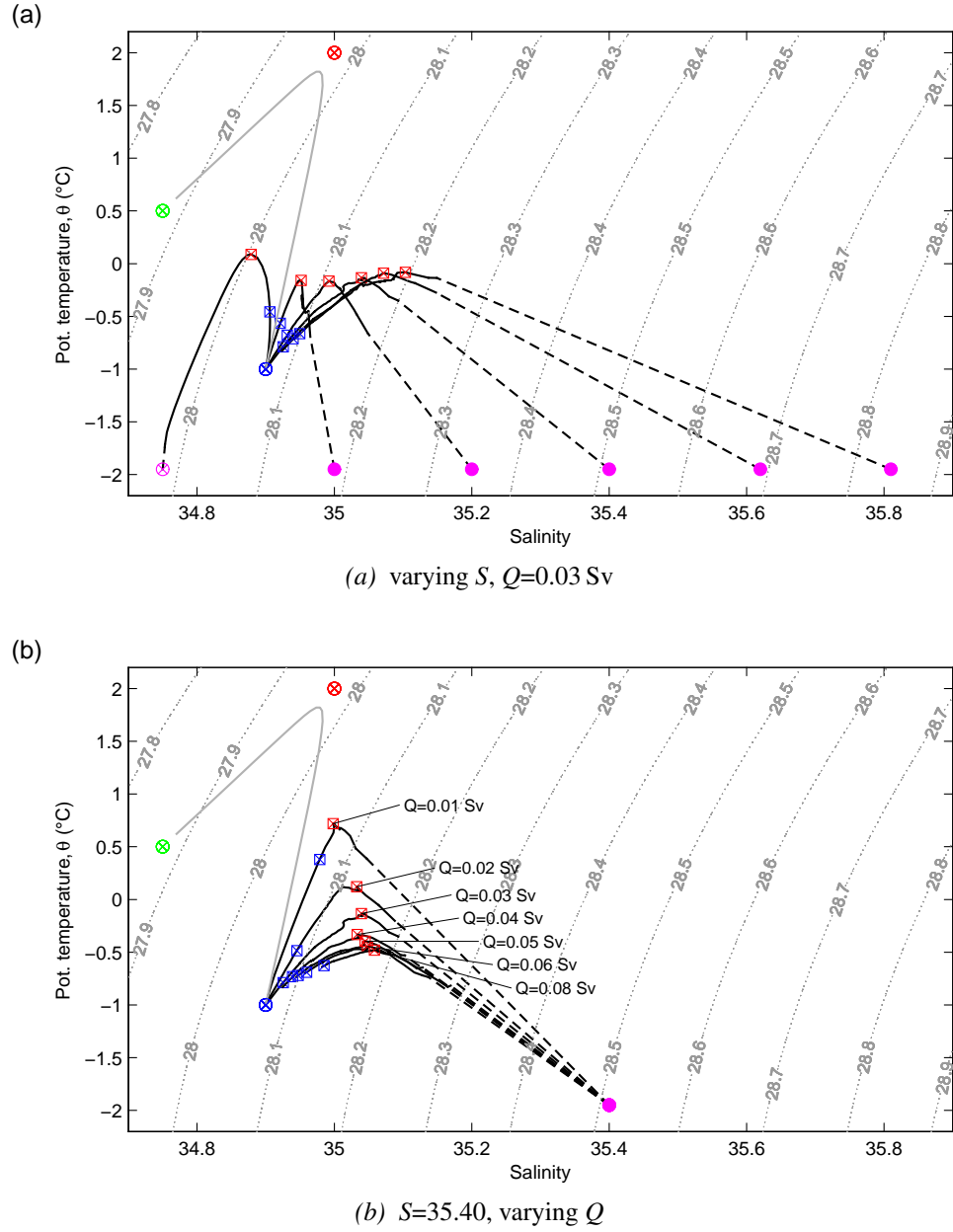


Figure 4.6: Downslope evolution of θ - S properties in the bottom model level on the slope. Curves are plotted for two series of model runs after 90 days: (a) varying inflow salinity S and (b) varying flow rate Q . The four different water masses in the model's initial conditions are indicated by crossed circles: green, ESW; red, AW; blue, NSDW; magenta, SFOW. Filled magenta dots indicate SFOW that is denser than any ambient waters. The temperature maximum on the slope is marked by a crossed red square, while the deepest penetration of passive tracers with concentration $PTRC > 0.05$ is marked by a blue square. The mixing within the injection grid cells is shown by the dashed black line. The faint gray curve is from a run without any injection ($Q=0$) for comparison.

4.3.4 Mixing characteristics

On its downslope descent the plume (SFOW) mixes with and entrains three ambient water masses (ESW, AW and NSDW). Entrainment implying a volume increase is based on a potentially arbitrary distinction between plume water and ambient water which could result in imprecise heat and salt budgets. In the following we therefore concentrate on the mixing process where these budgets remain well defined. Fig. 4.6 shows θ - S diagrams that trace the water properties down the slope at the end of each experiment (after 90 days). The θ - S values are plotted for the bottom model level at increasing depths from inflow region down to 1500 m. We show the θ - S properties for two experiments series: Q is constant and S varies (Fig. 4.6a), and Q varies and S is constant (Fig. 4.6b).

The dashed portion of the mixing curves in Fig. 4.6 shows that a considerable amount of mixing takes place within the injection grid cells. Any water introduced into the model is immediately diluted by ambient water. These processes take place over a very small region of the model and are not considered any further. Instead we focus on the common feature of all curves in Fig. 4.6: the temperature rises to a temperature maximum (marked by red squares) due to the plume's mixing with warm Atlantic Water. A very similar mixing characteristic was described by Fer and Ådlandsvik (2008) for a single overflow scenario ($S = 35.3$, $T = -1.9^\circ\text{C}$, $Q_{avg} = 0.07\text{ Sv}$) in a 3-D model study using ambient conditions similar to ours.

Amongst the series with constant $Q=0.03\text{ Sv}$ (Fig. 4.6a) only the weakest cascade (inflow salinity $S=34.75$) retains traces of ESW in the bottom layer after 90 days. In the experiments with more saline inflow ($S \geq 35.00$), the θ - S curve in Fig. 4.6a only spans three water masses - SFOW, AW and NSDW - while ESW is no longer present near the seabed. The salinity at the temperature maximum is nearly identical (red squares in Fig. 4.6a) for runs with the same flow rate Q .

4.3. RESULTS AND DISCUSSION

The experiments with a constant inflow salinity S (Fig. 4.6b) reveal that as Q increases the temperature maximum drops. At high flow rates the plume water is warmed to a lesser degree by the warm ambient water due to a larger volume of cold water entering the system.

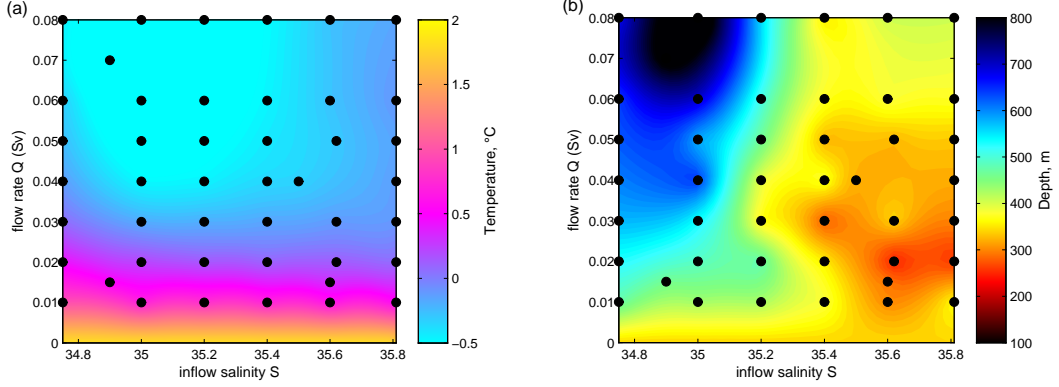


Figure 4.7: Characteristics of the temperature maximum in the bottom model level after 90 days is plotted against forcing parameters S and Q for all 45 experiments. (a) shows the temperature of the temperature maximum (in $^{\circ}\text{C}$) and (b) shows the depth (in m) at which it occurs. Results from the control run (without any dense water injection) were used to set the values corresponding to $Q = 0$.

We will now analyse the combined effect of varying both S and Q , and also consider the depth at which the temperature maximum occurs. The plume's mixing with warmer ambient waters (especially the Atlantic Water) warms the initially cold flow of dense water and also changes the depth distribution of temperature.

For all model runs we determine the temperature maximum and depth of the temperature maximum found in the bottom model level at the end of each experiment. The results are plotted against S and Q to investigate the full range of forcing parameters for all model runs. In Fig. 4.7 each experiment is marked by a black dot at a modelled combination of S and Q and the temperature maximum (in Fig. 4.7a) and its depth (in Fig. 4.7b) are shaded as coloured contours that span the S - Q space.

Fig. 4.7a shows that the magnitude of the temperature maximum (in $^{\circ}\text{C}$) is primar-

4.3. RESULTS AND DISCUSSION

ily dependent on Q and almost independent of S , which confirms the interpretation of Fig. 4.6 for a wider range of forcing parameters. Cascades with low flow rates ($Q \leq 0.02 \text{ Sv}$) are warmed by the ambient water to 0.2°C and above, while at higher flow rates ($Q \geq 0.03 \text{ Sv}$) the cold cascade lowers the temperature maximum below 0°C .

The flow rate dependence of the maximum bottom temperature in Fig. 4.7a can be explained by the different thermal capacity of the volume of plume water as Q changes, compared to the unchanged thermal capacity of the Atlantic Water. The salinity dependence of the depth of the temperature maximum in Fig. 4.7b is related to the salinity being the main driver of density at low temperatures. Plumes of lower salinity are thus less dense, causing them to advance downslope at slower speeds. A slowly descending plume remains in the Atlantic Layer for longer and more AW is mixed into the plume. Hence more warm Atlantic water gets advected downslope, causing the temperature maximum to occur at deeper depths in experiments with low S .

The mixing between the cold cascade and the warm ambient waters does not only lower the bottom-level temperature maximum, it also alters its depth which initially occurs within between 200 and 500 m at the start of each experiment. Fig. 4.7b shows that the depth of the temperature maximum has been displaced upslope (shallower than 400 m, shaded yellow) or downslope (deeper than 600 m, shaded blue) by the end of each experiment. In experiments where $S \leq 35.20$ the temperature maximum occurs at depths of 600 to 800 m while it remains at shallower depths of 200 to 400 m in experiments with $S > 35.20$. We conclude that the final depth of the temperature maximum is thus primarily dependent on the inflow salinity S .

By prescribing a varying salinity at the overflow we are able to recreate (in Fig. 4.6a) the schematic of Arctic cascading developed by Rudels and Quadfasel (1991), which is reproduced here in Fig. 4.8. Owing to the similarity in the ambient conditions and comparable parameters at the simulated overflow, the shape of the θ - S curve and the

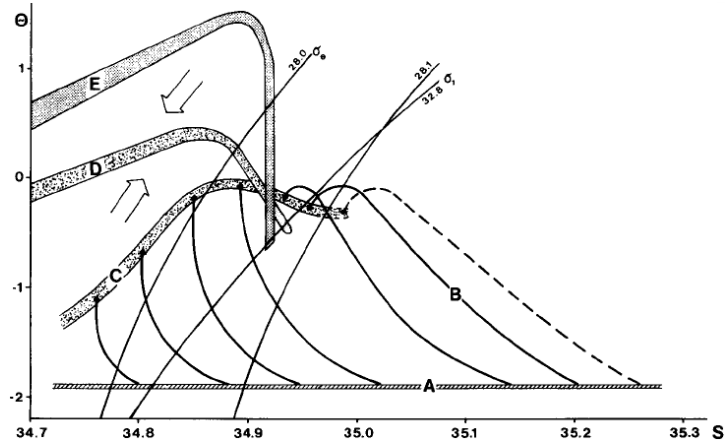


Figure 4.8: Schematic of the downslope evolution of θ - S properties of a dense water plume (from Rudels and Quadfasel, 1991). The mixing curves for source waters of different salinities (A) evolving (B) towards a temperature maximum due to the entrainment of Atlantic Water compare well with model results in Fig. 4.6a.

magnitude of the temperature maximum are in good agreement with this generalisation.

The results in this section expand on the Rudels and Quadfasel (1991) schematic and describe the response in the mixing to variations in volume transport at the sill (see Fig. 4.6b). The maximum bottom temperature along the plume path is mainly a function of the flow rate (see Fig. 4.7a). The depth at which the temperature maximum occurs, on the other hand, is mainly a function of the inflow salinity.

In summary, the following processes and factors affect the temperature maximum on the slope: (i) downslope advection of AW by the plume, (ii) the plume's momentum arising from its density gradient, (iii) mixing of the plume with Atlantic Water, (iv) the smallness of the thermal expansion coefficient at low temperatures, and (v) the total thermal capacity of the plume water.

4.3.5 Depth penetration of the plume

In the following, we investigate how the salinity S and flow rate Q of the dense water inflow affect the plume's final depth level. We quantify the downslope penetration of SFOW by calculating how much passive tracer (PTRC) is resident within a given depth

4.3. RESULTS AND DISCUSSION

range by the end of the model run. The total mass of PTRC, M_{PTRC} , is integrated over a given volume V using the modelled tracer concentrations C_{PTRC} as:

$$M_{\text{PTRC}} = \int_V C_{\text{PTRC}} dV \quad (4.1)$$

The penetration of the cascade into a given depth range is calculated as a percentage of M_{PTRC} within the given range compared to the total M_{PTRC} over the entire domain. A model run and its dense water supply can then be characterised according to the depth range containing more than 50% of PTRC that has been injected over 90 days.

In Fig. 4.9 we plot the results against S and Q for each of the 45 model runs. The final tracer percentage present within the given depth range is shaded in a contour plot where the S - Q combination of each experiment is marked by a black dot.

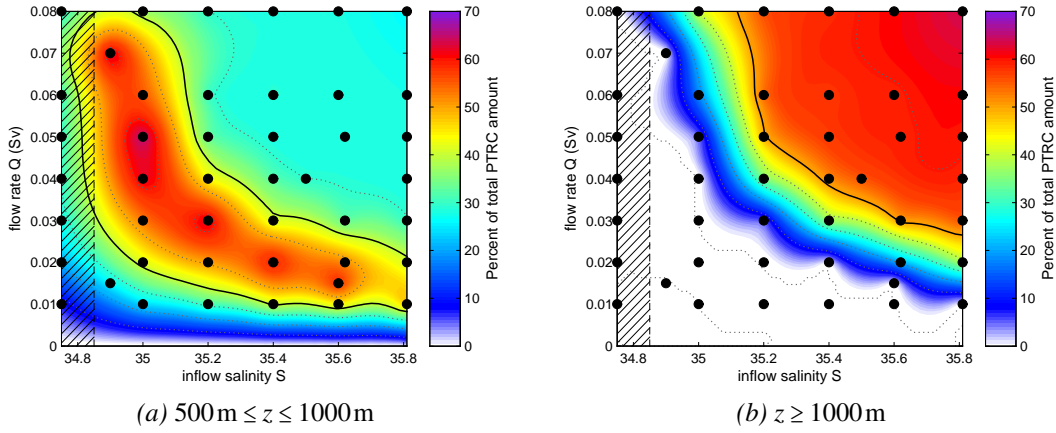


Figure 4.9: Presence of passive tracer (PTRC) (a) between 500 to 1000 m and (b) below 1000 m. Within the given depth range the percentage of tracer out of the total amount injected over 90 days is plotted against S and Q of all 45 model runs (black dots). The 50% contour is emphasised. The salinity range outside of the hatched area results in an initial plume density greater than the deepest ambient layer. Any value for $Q = 0$ (i.e. a control run without any tracer injection) is naturally zero.

In those model runs where the majority of PTRC is present between 500 and 1000 m at the end of the experiment, the plume has intruded into the Atlantic Layer and into

the AW-NSDW interface, but has not retained a strong enough density contrast to flow deeper. The combinations of S and Q producing this result are emphasised in Fig. 4.9a as the dots within the red shading indicating a tracer penetration greater than 50%. In the S - Q parameter space these runs are arranged in a curved band from low- S /high- Q via medium- S /medium- Q towards high- S /low- Q . In runs with lower S /lower Q (towards the lower left corner of the graph) the majority of the plume waters is trapped at shallower depths. In experiments with higher S /higher Q (towards the upper right corner of the graph) the plume reaches deeper as shown in Fig. 4.9b which is plotted for the presence of PTRC below 1000 m.

Fig. 4.9 provides a useful tool in classifying the prevailing regime in each experiment as ‘arrested’ (10 runs, Fig. 4.9a) or ‘piercing’ (16 runs, Fig. 4.9b) regarding the plume’s capacity to intrude into the Atlantic Layer or pass through it respectively. In the remaining experiments the plume either remains largely above the Atlantic Layer or the piercing ability is not clearly defined (which includes the ‘shaving’ regime).

The combinations of S/Q resulting in each of the regimes in Fig. 4.9 show that the initial density of the plume is not the only controlling parameter for the final depth of the cascade. At low flow rates, a plume which is initially denser than any of the ambient waters might not reach the bottom, while at high flow rates a lower initial density is sufficient for the plume to reach that depth. In the following section we explain the physics behind this result by considering the availability and sources of energy that drive the plume’s descent.

4.3.6 Energy considerations

The final depth level of the plume depends on kinetic energy available for the downslope descent and the plume’s mixing with ambient waters which dissipates energy. Even a closed system without any external forcing could contain available potential energy (APE, see Winters et al., 1995), but the APE in the model’s initial conditions is negligi-

4.3. RESULTS AND DISCUSSION

ble (as calculated using the algorithm described in Ilıcak et al., 2012) and remains near-constant during an injection-less control run. The only energy supply in the present model setup (a closed system except for the dense water injection) thus derives from the potential energy of the injected dense water, which is released on top of lighter water. Therefore, any kinetic energy used for descent and mixing must have been converted from this initial supply of potential energy.

From the model output the average potential energy (in J m^{-3}) is derived by integrating over the entire model domain:

$$PE = \frac{1}{V_{tot}} g \int_V \rho z dV \quad (4.2)$$

where g is the acceleration due to gravity (9.81 m s^{-2}), V is the grid cell volume and $V_{tot} = \int dV$ is the total volume of the model domain.

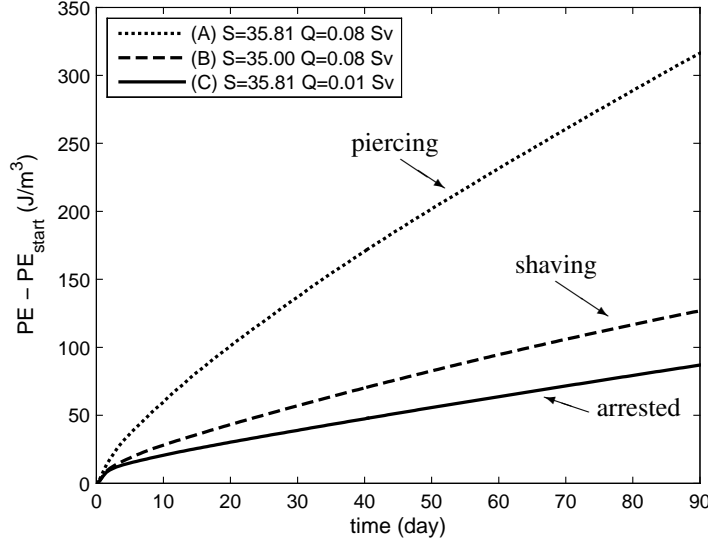


Figure 4.10: Increase over time in potential energy (PE) relative to the PE_{start} at the beginning of the experiment for three example runs varying S and Q . The labels point out the cascading regime (see Fig. 4.3).

The system's increase in potential energy over time is plotted in Fig. 4.10 for runs A, B and C (see also Fig. 4.4). In all runs PE is shown to be increasing as dense water

4.3. RESULTS AND DISCUSSION

is continually injected. One of the runs (run A, high S /high Q) was shown in Fig. 4.9b to fall into the piercing regime, while run B (low S /high Q) corresponds to the shaving regime and the plume in run C (high S /low Q) is arrested. The piercing run achieves a notably higher total PE at the end of the experiment than in the other cases. We now consider only the final value of potential energy increase after 90 days (ΔPE) from the values derived at the start and end of each experiment:

$$\Delta PE = PE_{end} - PE_{start} \quad (4.3)$$

In Fig. 4.11 we plot the final percentage of tracer mass found at the depth ranges 500-1000 m and 1000-1500 m against S and ΔPE . In contrast to Fig. 4.9 the contours of equal tracer percentage per depth range are now horizontal. This reveals that the cascading regime is a function of the potential energy gain ΔPE and independent of the inflow salinity and confirms that the initial density is not the only (or even the most significant) controlling parameter affecting the fate of the plume.

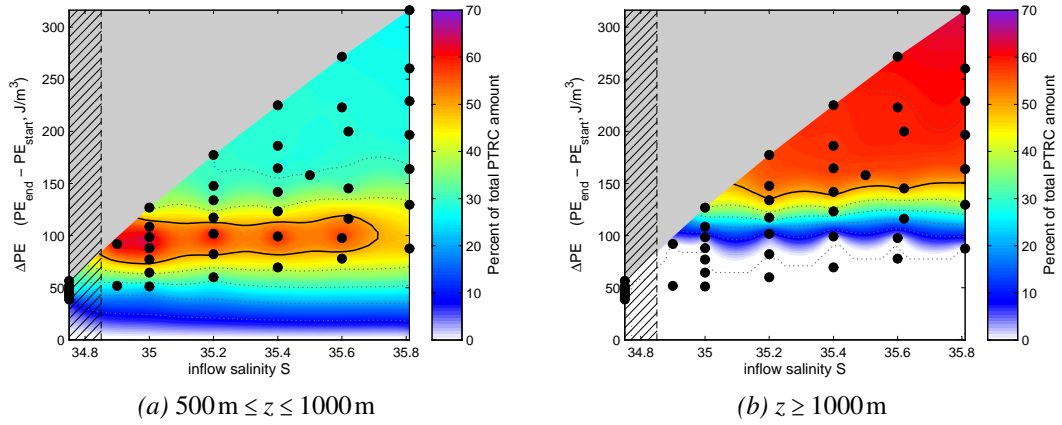


Figure 4.11: Similar to Fig. 4.9, but the percentage of tracer at a given depth range is plotted against S and ΔPE . Areas of untested S - ΔPE combinations are blanked. The control run ($Q = 0$, i.e. no tracer injection) is represented by $\Delta PE = 0$.

The analysis is extended to more depth ranges and we compute M_{PTRC} (Eqn. (4.1))

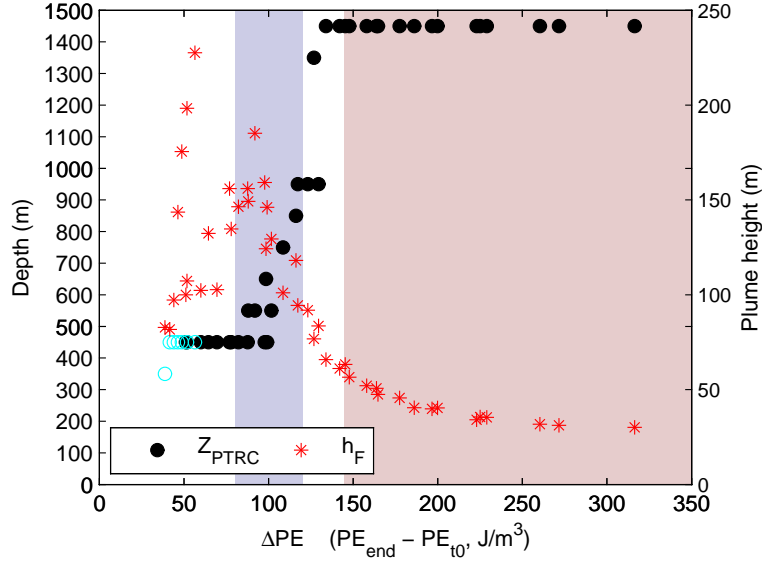


Figure 4.12: The depth level Z_{PTRC} at which the maximum amount of PTRC is found at the end of each run plotted against the gain in potential energy ΔPE (black bullets). Experiments with $S=34.75$ where the initial density is insufficient to penetrate the bottom layer are marked in cyan. Red stars show the average plume height h_F (in m) measured from tracer profiles. The approximate ΔPE ranges corresponding with arrested runs (light blue, cf. Fig. 4.11a) and piercing runs (light red, cf. Fig. 4.11b) are shaded.

in 100 m bins. The depth of the bin with the highest tracer mass gives Z_{PTRC} which is plotted against ΔPE in Fig. 4.12. The correlation between ΔPE and Z_{PTRC} (black bullets) shows very little scatter and indicates a functional relationship between the potential energy gain and the depth of penetration. With increasing potential energy in the system the plume is capable of first breaching the 200 m then the 500 m density interface in the ambient water. The abrupt transition from arrested ($Z_{PTRC} \approx 500$ m) to piercing ($Z_{PTRC} \approx 1500$ m) can be explained by the lack of stratification in the bottom layer. In most experiments where the plume breaches the AW-NSDW interface it also continues to the bottom of the slope after flowing through a homogeneous layer of NSDW.

Using the buoyancy flux of a density current, a concept similar to the flux of potential energy, Wells and Nadarajah (2009) reported a functional dependence between the

4.4. CONCLUSIONS

intrusion depth Z of a density current and the geostrophic buoyancy flux $B_{geo} = g'V_{Nof}h$ (where h is the initial height of the flow from a line source), the entrainment ratio E and the ambient buoyancy frequency N as $Z \sim E^{-\frac{1}{3}} B_{geo}^{\frac{1}{3}}/N$. However, their results are not readily applicable to this model where a non-linear ambient stratification with sharp density interfaces causes N to vary during the plume's descent. Neither is E constant during the model experiments.

Fig. 4.12 also plots the plume height h_F (red stars) against the potential energy gain ΔPE . It shows high h_F in runs with low ΔPE (those runs where the plume is arrested in the Atlantic Layer), and a low h_F in high- ΔPE runs when the plume spends little time transiting the AW and flows straight through to the NSDW layer.

The slow but steady rise in PE in Fig. 4.10 may suggest that any addition, however slow, of dense water (and thus potential energy) could eventually lead to the piercing regime if the initial SFLOW density is greater than the density of the bottom layer (which is the case in our setup for $S > 34.85$). Under this assumption the ΔPE -axis in Fig. 4.12 can be taken as a proxy for time. As time progresses (and ΔPE increases) the entrainment ratio E reduces (i.e. h_F shrinks) as the plume moves from the Atlantic Layer into the deep NSDW layer. When a certain threshold is passed, the plume has modified the ambient water sufficiently such that subsequent overflow waters pass through the AW relatively unimpeded (with less dilution) and penetrate into the deep waters. There is a caveat though, which works against the plume's piercing ability. The flow also needs to 'act quickly' (as is achieved by a high flow rate) to counteract mixing processes that cause the plume to dilute in the ambient waters.

4.4 Conclusions

We perform a series of model experiments using idealised conical geometry and simplified ambient conditions to study the penetration of a dense water cascade into ambient stratification. The model setup was inspired by conditions previously observed at

4.4. CONCLUSIONS

Svalbard in the Arctic Ocean. We investigate how variations in the parameters of the overflow - its initial salinity S and the flow rate Q - affect the fate of the plume.

We reproduce the main regimes where the plume is either (i) arrested at intermediate depths, (ii) pierces the intermediate layer and descends to the bottom of the continental slope or (iii) partially detaches off the bottom, intrudes into the intermediate layer while the remainder continues downslope. Our results show that for our given model setup the regime is predictable from the initial source water properties - its density (typically given by the salinity S as the temperature is practically constant at near-freezing) and volume transport Q .

The results show that even a cascade with high initial salinity S may not pierce the Atlantic Layer if its flow rate Q is low. The initial density of the plume is therefore not the only parameter controlling the depth penetration of the plume. The combined effect of S and Q on the cascade's regime is explained by the system's gain in potential energy (ΔPE) arising from the introduction of dense water at shallow depth and a functional relationship exists between ΔPE and the penetration depth and thus the prevailing regime.

Chapter 5

Tidally-induced lateral dispersion of the Storfjorden overflow plume

5.1 Introduction

In this chapter a high-resolution regional model (NEMO-SHELF, see Section 2.3) is used to investigate the effect of tides on the Storfjorden overflow plume in Svalbard, Arctic Ocean. Detailed studies into tidal effects on dense water flows were scarce until the AnSlope project (Gordon et al., 2004) in the Ross Sea, Antarctica revealed a dense bottom layer of 100 to 300 m thickness which is many times the Ekman depth (Padman et al., 2009). An analytical model (Ou et al., 2009) and subsequent numerical experiments (Guan et al., 2009) identified tidally-induced shear dispersion as a key mechanism for augmenting the spread and descent of dense water on the shelf and shelf-slope. In the Antarctic case of a flat wide shelf without any abrupt topography, tidal mixing was shown to increase the off-shelf transport of dense shelf waters into the deep basin.

With regards to modelling Arctic ocean circulation and ocean-ice interactions there is growing understanding that tidal effects should not be neglected (e.g. Holloway and Proshutinsky, 2007; Postlethwaite et al., 2011). In the shallow polynya area of the Storfjorden, where the dense waters are formed, the circulation has been shown to be sensitive to mixing due to wind and tides (Skogseth et al., 2007), but considering Svalbard's complex topography it remains an open question whether the plume's downstream response to tidal forcing is comparable to the findings of Padman et al. (2009) and Ou

et al. (2009) in Antarctica. A series of numerical experiments focus on the effects of the tides on the overflow plume to address the following questions:

- How do tides affect the Storfjorden overflow plume?
- Which physical processes explain tidally-induced modifications in the plume's behaviour?
- Do tides cause an increase in the downslope transport of Storfjorden water into the deep Fram Strait?

This chapter is structured as follows: Section 5.2 describes the configuration of the model, its data sources and the experimental design. Results are presented in Section 5.3 and discussed in Section 5.4. Section 5.5 concludes the chapter with answers to the above questions.

5.2 NEMO model setup

A 3-D ocean circulation model, based on NEMO-Shelf (O'Dea et al., 2012), is used in the configuration described in Section 2.3 with the addition of atmospheric forcing, open boundary conditions and tides. The 480 by 540 km domain with a uniform horizontal resolution of 3 km has its south-western corner at 75.0 °N, 6.0 °E and encompasses most of the Svalbard archipelago as well as the Storfjordrenna to its south, the Spitsbergen continental slope to its west and parts of the eastern Fram Strait (Fig. 5.1). A technical description of the configuration and setup of the model runs on a High Performance Computing system is given in Appendix E.

5.2.1 Bathymetry preparation

The model bathymetry from IBCAOv3 (Jakobsson et al., 2012) was first interpolated onto the model grid and then slightly adjusted as follows. First, the bathymetry was smoothed using 2-D convolution with a 3×3 Gaussian kernel in order to reduce near-bottom pressure gradient errors. Second, the adjustment of Martinho and Batteen

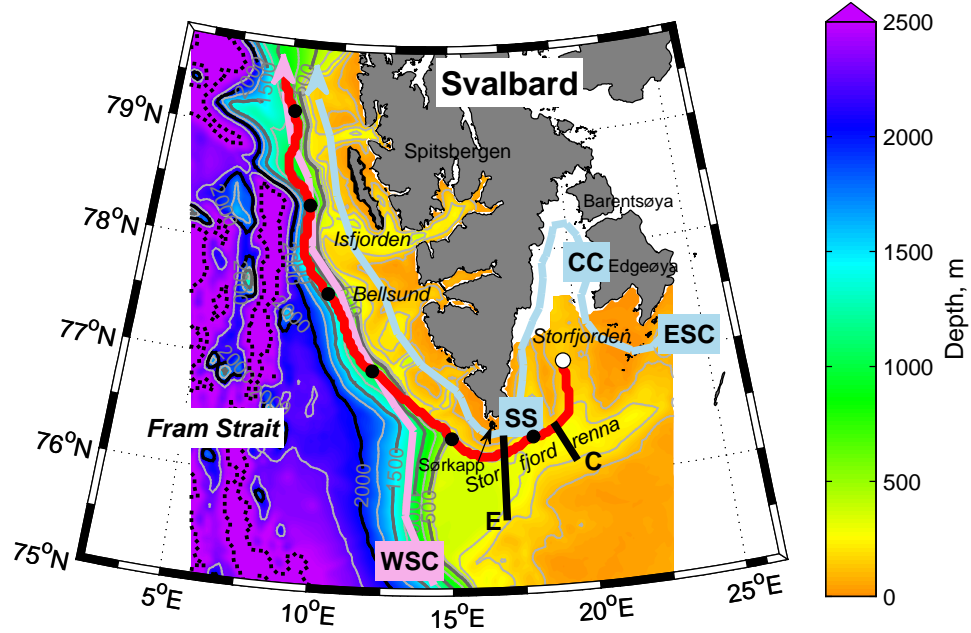


Figure 5.1: Bathymetry of the model domain (shaded). Faint contours are drawn every 100 m for the first 400 m, then every 250 m with stronger lines at 500, 1000, 1500 and 2000 m (labelled). The capped bathymetry at 2500 m is shown as a black dotted line. A generalised plume pathway (red line) is marked every 100 km (filled circles) downstream of the sill (open circle). Schematic locations of main currents (taken from Saloranta, 2001; Skogseth et al., 2005b) are abbreviated as WSC=West Spitsbergen Current, ESC=East Spitsbergen Current, CC=Cyclonic Coastal Current, SS=Sørkappstrømmen. Labelled cross-sections are: C (Aug 2002) in Stor fjordrenna (from Fer et al., 2004), E (May/Jun 2001) south of Sørkapp (from Fer et al., 2003).

(2006) was applied with a critical slope parameter value of 0.3 which limits slope angles to no more than $\sim 4^\circ$ on the continental slope. This type of bathymetry adjustment, common to σ -level models, preserves numerical stability while remaining faithful to the topographical characteristics of the terrain (see Sikirić et al., 2009, and references therein).

5.2.2 Vertical co-ordinate system

In the vertical, the model has 50 computational levels using the s_h -coordinate system (see section 2.3.1, and Wobus et al., 2013) which was especially designed for the modelling of near-bottom density currents. The bottom-most 16 levels follow the terrain such that vertical resolution near the bottom is never coarser than 7.5 m. Equidistant level spacing within the bottom boundary layer avoids any loss in vertical resolution with increasing depth (as is the case with the traditional s -coordinate stretching function). Above the bottom layer three s_h -levels are inserted at 200, 500 and 1000 m to keep levels in the interior mostly horizontal.

5.2.3 Subgrid-scale parametrisation

The horizontal small-scale physics in the conservation equation for tracers (i.e. diffusion of heat, salt & passive tracers) are written in NEMO as follows (Madec, 2008):

$$D_{hor} = \nabla \cdot (\kappa_{hor} \mathfrak{R} \nabla T) \quad (5.1)$$

where \mathfrak{R} is a rotation matrix containing the slopes r between the surface along which the diffusive operator acts (the present model used the case where $r = \frac{d\rho}{dx} / \frac{d\rho}{dz}$ for isoneutral surfaces) and the model s -level. The horizontal diffusion is thus represented by two factors – (i) the diffusivity coefficient κ_{hor} , which depends on the velocity field and (ii) the local tracer gradient ∇T .

Horizontal viscosity coefficients are constant at $20 \text{ m}^2 \text{ s}^{-1}$ while the horizontal diffusivity coefficients evolve with the flow field according to a Smagorinsky (1963) scheme implemented in NEMO by Luneva and Holt (2010) and further refined by Shapiro et al. (2013).

5.2.4 Boundary and initial conditions

The model is initialised with fields taken from a global NEMO simulation using the tripolar ORCA 1/12° grid (resolution ca. 3.5 km in the Arctic Ocean, ca. 4 km in the vicinity of Svalbard and ca. 9km globally) and 75 vertical z -levels (the global NEMO, in a similar configuration, is described in detail, e.g., by Blaker et al., 2012). The same model data, provided every 5 days, is also applied at ocean grid points of the domain edge using NEMO’s BDY module for unstructured open boundaries. With regards to the model’s forcing at the open boundaries, the given configuration can be described as a ‘one-way nested’ solution.

The fields of temperature, salinity and sea ice provided by the global model are applied to the corresponding model fields using the Flow Relaxation Scheme (FRS, see Davies, 1976; Engedahl, 1995). The model variables are relaxed to externally-specified values over relaxation zone next to the model boundaries as described by Madec (2011):

Given a model prognostic variable Φ (e.g. the model’s temperature field):

$$\Phi(d) = \alpha(d) \Phi_e(d) + (1 - \alpha(d)) \Phi_m(d) \quad d = 1, N \quad (5.2)$$

where Φ_m is the model solution, Φ_e is the external value, d gives a discrete distance from the model boundary and α is the blending parameter that varies from 1 at $d = 1$ to a small value at $d = N$, where $N = 7$ in this study. This scheme is equivalent to adding a relaxation term to the prognostic equation for Φ of the form:

$$-\frac{1}{\tau}(\Phi - \Phi_e) \quad (5.3)$$

where the relaxation time scale τ is given by a function of α and the model time step Δt :

$$\tau = -\frac{1-\alpha}{\alpha}\Delta t \quad (5.4)$$

where the function α is a smooth *tanh* function:

$$\alpha(d) = 1 - \tanh\left(\frac{d-1}{2}\right) \quad d = 1, N \quad (5.5)$$

In practical terms, the model solution Φ_m is completely prescribed by the external value Φ_e at the outer edge of the model domain, while the weighting α with which Φ_e is blended into the model field Φ_m then decreases gradually towards the inner edge of the relaxation zone. A relaxation zone helps smooth out spurious reflections of outgoing signals from the model boundary. Further numerical stabilisation is achieved by applying 4× factor to horizontal diffusivity and viscosity coefficients to the iso-neutral laplacian diffusion operator within the relaxation zone.

Engedahl (1995) reported that the FRS can be successfully applied to boundary values of sea level elevation and baroclinic velocities as well. FRS performs well if the transports at the boundaries are well known (as would be the case of nesting within a global model), but cited the caveat that the boundaries should be situated away from areas of strong inflow and outflow. The ocean circulation within the Svalbard model domain (Fig. 5.1), however, is strongly characterised by the northward flow of the West Spitsbergen Current into the domain's southern boundary and out of its northern boundary. It was therefore decided not to relax the model's baroclinic velocities to the external global model velocities using the FRS. Furthermore, the simultaneous imposition of both baroclinic velocities from the global model and barotropic transports with sea level elevations from the tidal model (see next Section 5.2.5) is likely to result in an ill-posed problem. These issues are discussed further by Engquist and Majda (1977), Marchesiello et al. (2001) and references therein.

Thus, the Flather condition (Flather, 1976) is chosen to impose, at the model boundaries, the sea surface elevation and barotropic velocities from the global model. This particular mix of relaxation and Flather schemes avoids over-specification of external values (Marchesiello et al., 2001) and eases the incorporation of tidal elevations (see following Section 5.2.5). The Flather condition is a radiation condition on the depth-mean transport normal to the open boundary. In NEMO it is written as (from Madec, 2011):

$$U = U_e + \frac{c}{h}(\eta - \eta_e) \quad (5.6)$$

where U is the depth-mean velocity normal to the boundary and η is the sea surface height. Again, the subscript e denotes the external value taken from the global model. The speed of external gravity waves is given by $c = \sqrt{gh}$ where h is the water depth. NEMO sets the depth-mean normal velocity along the edge of the model domain equal to the external depth-mean normal velocity, and applies a correction term that allows gravity waves generated internally to exit the model boundary.

A further correction is applied to the normal velocities around the boundary to ensure that the integrated volume flow through the boundary is zero and thus prevent a possible drift in total ocean volume over time.

5.2.5 Atmospheric and tidal forcing

Fields from the Drakkar Forcing Set (DFS4.1¹) are applied using the bulk CORE formulation (Large and Yeager, 2004) to calculate atmospheric fluxes at the ocean surface. DFS4.1 was compiled from NCEP and ECMWF reanalysis products by the Drakkar group (Brodeau et al., 2010) and provides global coverage at 320×161 resolution for air temperature and specific humidity at 2 m, wind velocity at 10 m and 192×94 for

¹obtained from the National Oceanography Centre's ORCA1 project website at <ftp://ftp.noc.soton.ac.uk/omffftp/DFS4.1>

short and longwave radiation, precipitation and snow.

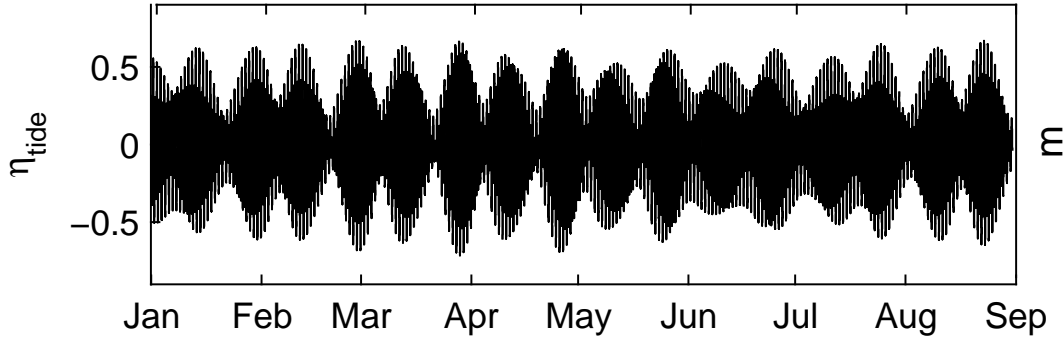


Figure 5.2: Tidal elevation η_{tide} from the TPXO7.2 tidal model (Egbert and Erofeeva, 2002) for 75.03 °N, 16.32 °E on the southern domain boundary.

The 1/12 ° global model within which the Svalbard domain is nested does not include tides. Tidal forcing is additionally prescribed at the domain boundaries by adding the barotropic transport U_{tide} and sea level elevation η_{tide} from a tidal model to the terms U_e and η_e in Eq. (5.6) (see previous Section 5.2.4). The tidal model that provides the barotropic transport and sea level elevation values is the TPXO7.2 model (TOPEX/Poseidon crossover solution) developed by Egbert and Erofeeva (2002) at Oregon State University. TPXO7.2 is a medium-resolution 0.25°×0.25° fully global assimilation model. The tidal model assimilates T/P altimetry between 66° N/S latitude, ERS (European Remote sensing Satellite) data and coastal and benthic tide gauge data from the Arctic and Antarctic, making it particularly suitable for this study. An example time series of combined tidal elevation η_{tide} from the Q1, O1, P1, K1, N2, M2, S2, K2 and M4 constituents is shown in Fig. 5.2. A more complete description of tides in the study region can be found in Gjevik et al. (1994), Skogseth et al. (2007) and Postlethwaite et al. (2011).

5.2.6 Overflow parametrisation and passive tracers

For the purposes of this study the model is set up to accurately represent, as closely as possible, the downstream evolution of the dense water plume once it has left the fjord. The model does not attempt to represent the polynya dynamics, sea ice formation, brine rejection and vertical convection that result in the formation and accumulation of dense water in the fjord's interior. A very simple sea ice model is used to block sea surface fluxes where SST drops below freezing temperature to prevent further cooling, but brine rejection is not explicitly represented. Instead the outflow is parametrised based on previous observations in the fjord (Anderson et al., 1988; Schauer, 1995; Haarpaintner, 1999; Haarpaintner et al., 2001; Fer et al., 2003; Skogseth et al., 2005b, 2008) and the results of dedicated modelling studies (Haarpaintner et al., 2001; Skogseth et al., 2004, 2005a, 2009). The overflow parametrisation in the model is designed to capture the end result of the aforementioned processes inside the fjord.

The inner fjord is excluded from the model by land-masking the basin north of the line between the Crollbreen glacier (77.2 °N, 17.4 °E) in the east and Kvalpynten² (77.45 °N, 20.9 °E) in the west (unshaded areas west of Barentsøya and Edgeøya in Fig. 5.1). Of the fjord's basin remains a 35 km-wide (between 18.9 and 20.3 °E) artificial bay behind the sill (located at 77.35 °N, 19.5 °E) where the injection of water at $T = -1.92$ °C with an elevated salinity simulates the end result of the dense water formation processes inside the fjord.

The freezing period and thus dense water accumulation within the fjord starts around November. Beginning between January and March the dense water starts to spill over the sill in a series of pulses (Schauer, 1995) producing an average of 0.06 to 0.07 Sv of dense water over an overflow period lasting 5 to 6 months (see Skogseth et al., 2004, 2005a, 2008, for more details). An idealised flow rate profile (similar to Fer and Åd-

²<http://stadnamn.npolar.no/> is a useful resource on Svalbard place names.

landsvik, 2008) is used to capture the observed variability during an overflow season. This profile is identical in all runs.

The model is initially run for 140 days from the end of August to mid-January to allow dense water remaining from the previous overflow period (contained in the initial conditions) to drain from the fjord. The prescribed overflow begins in mid-January with a 2 week ramp-up reaching 0.086 Sv by beginning of February, continues at full strength for 60 days until the end of March and ramps back down to zero over 75 days until mid-June. During this period an average of 0.06 Sv exits the injection bay. Each model run continues thereafter for 2.5 months until the end of August giving a total model period of 1 year per run.

To study the plume's downstream evolution the dense water is marked with on-line fully-diffusive passive tracers. Passive tracers enable a truly Lagrangian view of waters originating in the fjord, something that is near impossible in the ocean where the plume is typically identified by its T-S signature only. NEMO's TOP/MYTRC module is used with the additional implementation of the lateral boundary condition $dC/dx = 0$ to prevent tracer with concentration C from reflecting back into the model domain. Three passive tracers TRC1, TRC2 and TRC3 (with values 0 to 1.0) are used. They correspond with the ramp-up, constant flow and ramp-down periods of the overflow cycle to separate the effects of dense water leaving the fjord at different times. A combined picture of the entire plume emerges by simply adding up the three tracer fields (the passive tracer is a linear operand, so that the sum of gradients is equal exactly to the gradient of the sum).

5.2.7 Experimental setup

The response of the dense water plume to overflow strength (i.e. salinity if dense water injected behind the sill), tides (by turning tidal forcing on and off) and wind strength (by activating the DFS4.1 wind fields or setting them to zero) is independently tested in

a total of 16 model runs. Four different overflow scenarios are tested - LOW, MEDIUM, HIGH & EXTREME - with a dense water salinity at the sill of $S_{\text{sill}} = 35.2, 35.5, 35.8$ and 36.1 respectively. While salinities of up to 35.8 have been observed in Storfjorden (Rudels et al., 2005), the most saline scenario ($S_{\text{sill}} = 36.1$) has not been observed and thus represents an idealised extreme end of the parameter space.

Each overflow scenario is performed with and without tides, but with atmospheric forcing including winds. Each of these experiments is then repeated without wind. The following results mainly focus on the HIGH overflow scenario ($S_{\text{sill}} = 35.8$) by comparing the tidal run to its non-tidal control. Experiments with zero wind strength are only referred to when it is appropriate to isolate the tidal mixing from wind-driven mixing. For details on the spin-up sequence leading up to each run, see Appendix F.

5.3 Results

5.3.1 Comparison with observations

The HIGH scenario in the model aims to be representative of deep cascading events that took place in 1986, 1988 and 2002 when a dense water plume of Storfjorden origin was detected deeper than 2000 m in eastern Fram Strait (Quadfasel et al., 1988; Schauer et al., 2003; Akimova et al., 2011). Large-scale observations in the Storfjorden region were conducted in August 2002 by Fer et al. (2004). Their section C in Storfjordrenna (see location in Fig. 5.1) is reproduced in Fig. 5.3 for comparison with the model results. The observations were made in August, approximately 4 months after the overflow peaked around April 2002 (Rudels et al., 2005), while the model's idealised injection profile achieves peak overflow conditions in March. Fig. 5.3 therefore shows the model results for July in order to compare plumes during the same stage of the flow. Fig. 5.4 reproduces observations of section E by Fer et al. (2003) in May/June 2001 at a location further downstream (see location in Fig. 5.1). These 2001 cross-sections are compared with the modelled MEDIUM scenario as overflow conditions in

5.3. RESULTS

that season were weaker than in 2002 (see Skogseth et al., 2005b).

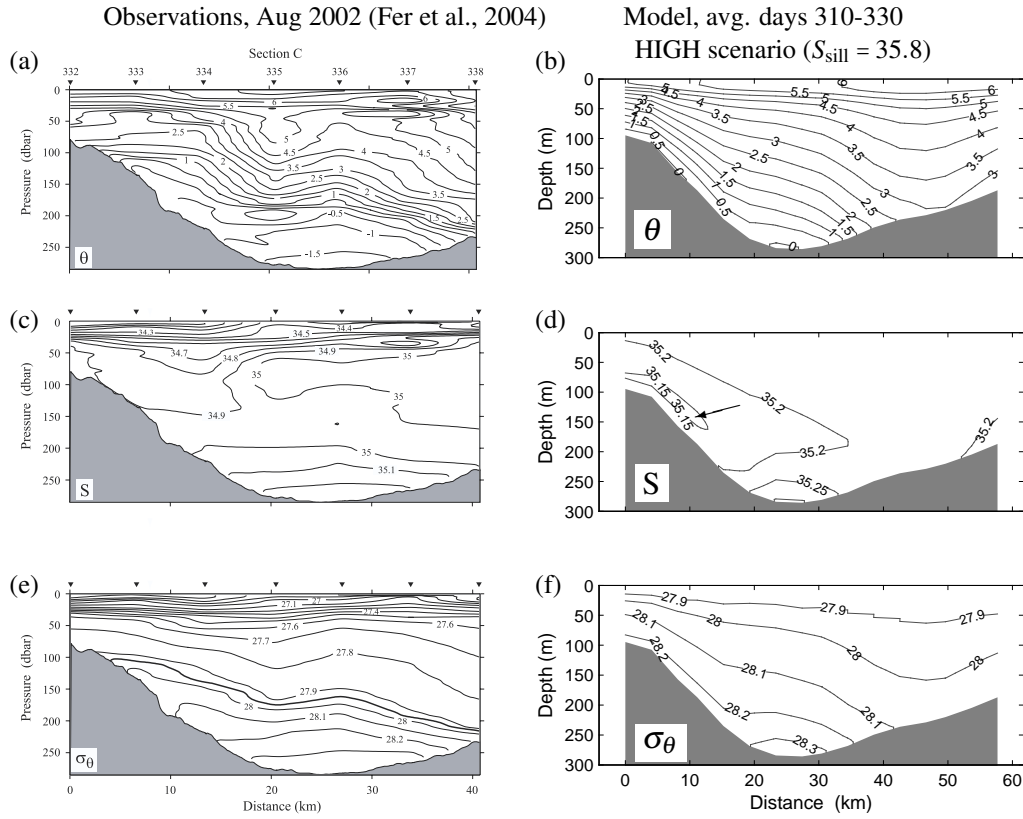


Figure 5.3: Comparison of cross section C in Fer et al. (2004) (left) with model of the HIGH overflow scenario ($S_{sill} = 35.8$, with tides) (right). Observations and averaged model cross-sections show the plume approx. 4 months after the peak overflow period. The arrow in (d) points out a patch of low salinity.

Considering the overflow parametrisation and initial/forcing conditions which were not based on these particular years the model shows a fair agreement in several aspects. The model represents well the surface warming to $\sim 6^{\circ}\text{C}$ in summer, yet loses to some degree the cold bottom layer temperature still evident in the observations (Figs. 5.3a,b and 5.4a,b). A warmer bottom layer temperature in the model is attributed to differences in the ambient conditions of the plume. The salinity sections (Figs. 5.3c,d and 5.4c,d) show the saline bottom layer of the plume and an overlying fresher layer. A comparable patch of low salinity was noted by Piechura (1996) and further investigated by Fer et al. (2003) who suggested as a possible reason the lateral exchange of fresher shelf water into a layer between the dense plume core and the overlying Atlantic Water. Near the surface such an exact separation of water masses is not well represented by the model which shows much reduced salinity stratification. A possible reason is the lack of a sophisticated sea ice model which means that summer ice melt and surface freshening are not well resolved. This is deemed acceptable for the present study, as it focusses mainly on the plume in the near-bottom layer. The comparison of bottom layer density (Figs. 5.3e,f and 5.4e,f) shows good agreement in the slope of isopycnals which ‘lean’ against the Spitsbergen side of the Storfjordrenna (left-hand side of the plots).

The results of simulations with and without tides showing the spread of Storfjorden Overflow Water (SFOW) from the fjord onto the slope are presented in Fig. 5.5. The spread of SFOW is visualised by the bottom-level concentration of passive tracer, which is injected together with the dense water behind the sill. At the end of each model run – 7.5 months after the start of the overflow – large amounts of dense water have spread out of the fjord and on to the continental slope west of Svalbard, while some overflow tracer still remains ‘trapped’ behind the sill. South of the sill the flow forms eddies and fills the depressions in the Storfjordrenna. This deep channel between the fjord’s mouth and the Sørkapp steers the plume (assisted by geostrophic adjustment, Fer et al., 2003)

5.3. RESULTS

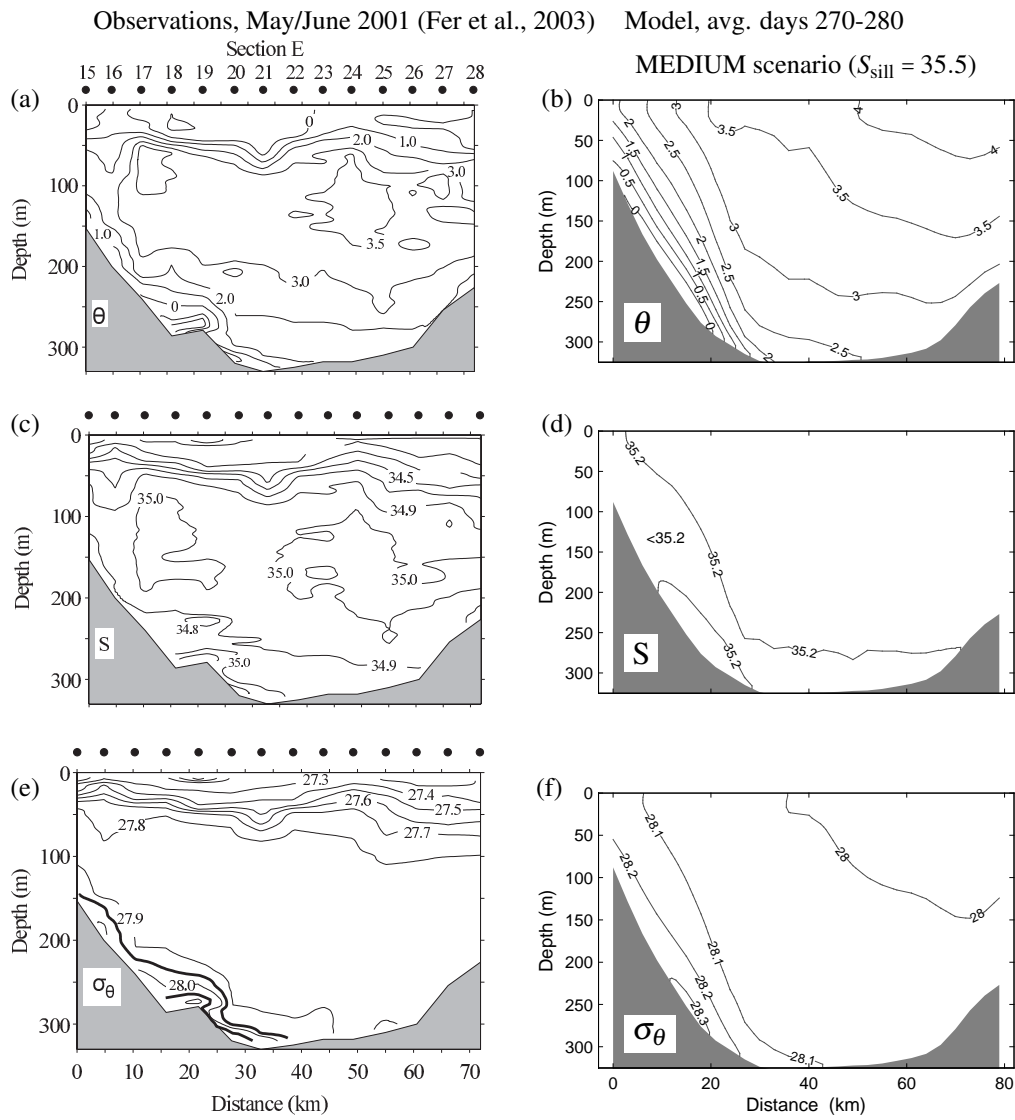


Figure 5.4: Comparison of cross section E in Fer et al. (2003) (left) with model of the MEDIUM overflow scenario ($S_{\text{sill}} = 35.5$, with tides) (right). The model data is an 11-day average for the same time of year as the observations.

west towards the shelf break.

In the model the SFOW first reaches depths greater than 2000 m during early summer and significant amounts of plume water remain at this depth in August (Fig. 5.5). At this time the plume extent for the HIGH scenario compares well with the locations where in 1986, 1988 and 2002 dense shelf water was observed near the seabed (see Fig. 5.5b for station locations from Akimova et al., 2011). The model is thus seen to represent adequately the Ekman advection that facilitates downslope transport of tracer in the bottom boundary layer.

Field observations typically report a bottom layer thickness of 50-60 m, but the plume can be as thin as 10-20 m in deep Fram Strait (Quadfasel et al., 1988; Schauer, 1995; Schauer and Fahrbach, 1999). In the absence of tracer analysis this is likely an underestimate as the density structure does not show low source water concentrations in the interfacial layer between the plume core and the ambient water. In this study (as in Fer and Ådlandsvik, 2008) the plume thickness is evaluated by passive tracers which is considered to result in an overestimate. A plume thickness on the order of 50 to 100 m on the shelf (see Fig. 5.6) is thus seen as a good agreement with observations. However, in deep Fram Strait at depths >1000 m the model typically shows a diffuse plume with a thickness in excess of 200 to 300 m. Previous models by Jungclaus et al. (1995) and Fer and Ådlandsvik (2008) similarly overestimated the thickness of the deep plume. Until better agreement with observations can be achieved in this depth range it appears wise to treat model results in deep Fram Strait with some caution. This study therefore focusses on the intermediate part of the flow - from the plume's spread into the Storfjordrenna to the start of its descent down the Western Svalbard slope.

5.3.2 Tidal effects on off-shelf transport

At the end of the modelled overflow period the tracer concentration behind the sill is lower in tidal runs (an example run is shown in Fig. 5.5b) compared to the non-tidal con-

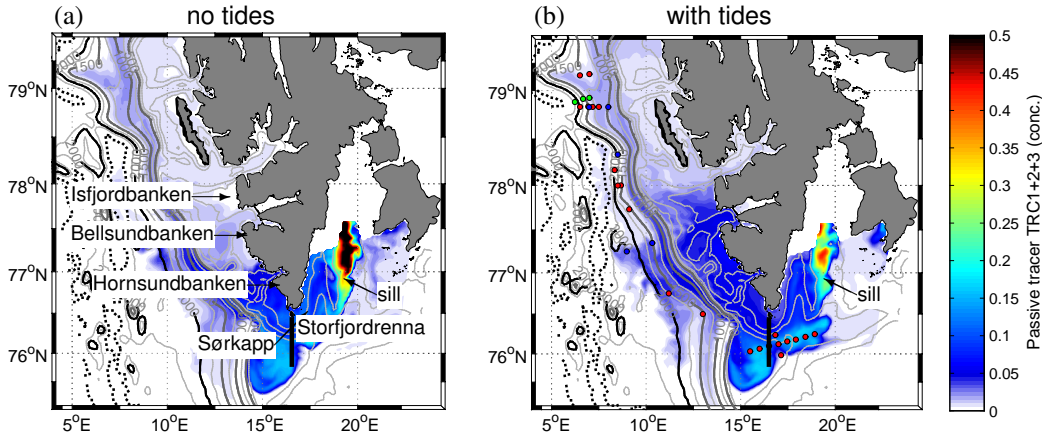


Figure 5.5: Snapshot of passive tracer concentration in the bottom model level at the end of the model run (31st August). The bold line due south from the Sørkapp marks the location of the cross-section in Fig. 5.6. Coloured dots in (b) mark CTD stations where the plume was observed in 1986 (blue dots), 1988 (green dots) and 2002 (red dots) (from Akimova et al., 2011).

trol run (Fig. 5.5a). The tides are thus seen to cause a more efficient flushing of dense water from the fjord region towards the shelf edge. Figure 5.6 compares cross-sections of the average tracer concentration during the period of maximum overflow (April – May) at the southern tip of Spitsbergen where the plume leaves the Storfjordrenna. The tidal run (Fig. 5.6b) reveals a thicker plume of comparable concentration. This, again, suggests increased transport of tracer towards the shelf edge in tidal runs. Increased plume thickness in tracer concentration is closely matched by a greater cross-sectional area between sea bed and the isopycnals (overlaid in Fig. 5.6) to indicate increased mass transport. The difference in mass flux by tidal effects is not evenly distributed. At a depth of 300 to 370 m the bottom layer bounded by the 28.2 isopycnal is thicker while in the shallows, between 0 to 100 m depth, the layer between 28.1 and 28.2 kg m^{-3} is thicker. The tidally affected plume is thus denser in the deep part of the trough, but lighter and more diffuse in the shallows.

In the following the fluxes of the model's passive tracer are compared between non-tidal and tidal runs. The tracer flux Q_{TRC} is integrated over the cross-sectional area

5.3. RESULTS

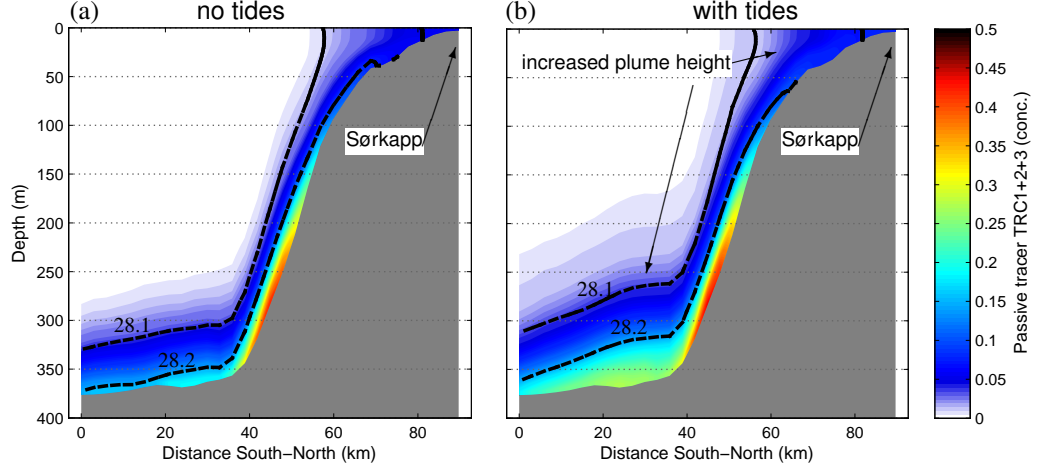


Figure 5.6: Cross-section of overflow tracer (average concentration during April – May) for a South–North transect from 75.78 °N, 16.32 °E in Storfjorden northwards to the Sørkapp headland at the southern tip of Spitsbergen (location shown on map in Fig. 5.5). The 28.1 and 28.2 kg m⁻³ isopycnals are superimposed.

A from the tracer concentration $C_{\text{TRC}} = \text{TRC}_1 + \text{TRC}_2 + \text{TRC}_3$ where $C_{\text{TRC}} \geq 0.01$ as follows:

$$Q_{\text{TRC}} = \int \int_A C_{\text{TRC}} U \, dx \, dz \quad (5.7)$$

where U is the velocity normal to the section in the downstream direction (see arrows in Fig. 5.7a), and x and z are the horizontal and vertical coordinates respectively. This calculation is carried out for a cross-section starting in deep Fram Strait and reaching onto the Western Svalbard Shelf (Fig. 5.7a). The chosen location is positioned downstream of the cross-section in Fig. 5.6 in order to test whether greater off-shelf transport translates into increased transport into the deep basin. Two parts of the cross-section are analysed separately - the *shelf* part in depths of 0 to 300 m and the *slope* part in depths of 300 to 2000 m. The proportion of the flux through a partial section is calculated and then compared with the total flux passing through the combined section. This method normalises the result and corrects for varying absolute amounts of tracer released in

different overflow scenarios.

The relative tracer fluxes through the slope part of the section for all 16 model runs are shown in Fig. 5.7b. The majority of the SFOW, 83 to 92% of the total, passes through the slope section (at > 300m depth). It is therefore reasonable to refer to the plume on the slope as the ‘main’ cascade. However, the remainder of 8% to 17% of plume waters represents a significant amount of SFOW that does not sink but remains on the shallow shelf.

Figure 5.7b demonstrates the tides’ role in controlling the proportion of overflow waters being diverted onto the shelf and thus away from the main flow. In each of the tidal runs a lower proportion of the total tracer flux (a 2% difference when averaged over all runs) is measured on the slope compared to the shelf. Qualitatively, this result is independent of whether wind was included in the model run or not. Wind-driven effects appear to contribute to SFOW being diverted onto the shelf as the percentage of slope flux in Fig. 5.7b is generally lower in model runs with wind compared to those without wind. The following results, however, focus on tidal effects.

Why is the proportion of SFOW flowing down the deep slope decreased by the tides? The answer is found by backtracking along the plume path and analysing tidal effects on the flow before it splits into shelf and slope branches. The following section will discuss tidal modifications of the plume near the Sørkapp headland, which lies in close proximity to the plume’s path during its spreading phase on the shelf.

5.3.3 Lateral dispersion of the plume

The overflow of tracer in non-tidal and tidal runs starts diverging near the Sørkapp headland. Past this point a smaller proportion of the plume waters sinks down the deeper slope under tidal conditions (Fig. 5.7b) and tides also cause more overflow water to propagate northwards on the West Svalbard Shelf (Fig. 5.5). The simulated lateral diffusion in the vicinity of the Sørkapp is diagnosed by following Eq. 5.1 and analysing

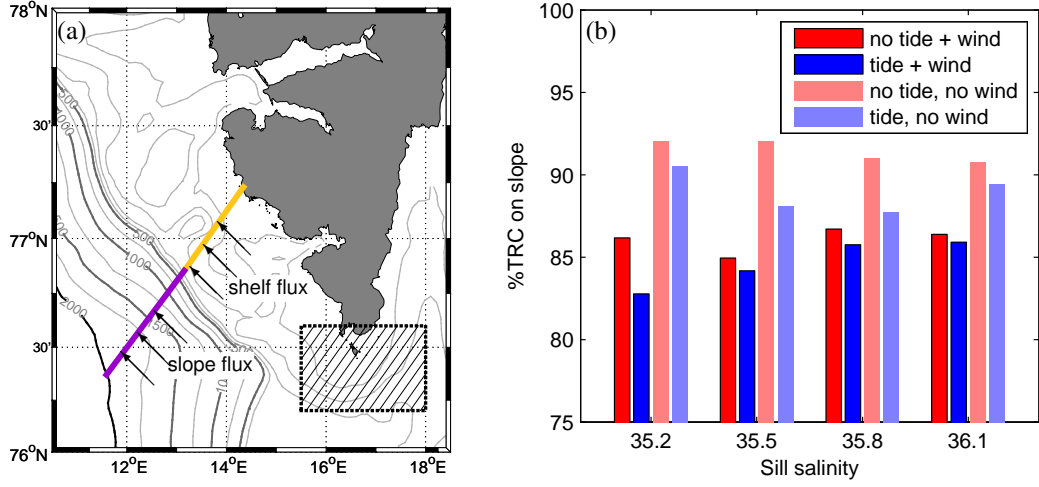


Figure 5.7: (a) Location of a cross-section divided into *shelf* and *slope* parts for calculation of Q_{TRC} (Eq. (5.7)). The hatched box is the averaging area for Fig. 5.9. (b) Percentage of tracer flux passing through the *slope* section out of the total flux measured at the combined slope+self section. Blue bars are tidal runs, red bars are non-tidal runs. Runs with zero wind strength are shown in paler colours for comparison.

variations of the horizontal diffusivity coefficient κ_{hor} and the horizontal tracer gradient ∇T .

The model derives the horizontal diffusivity coefficient κ_{hor} from the velocity field using the Smagorinsky (1963) scheme. The coefficient κ_{hor} is vertically averaged over a 50 m thick bottom layer to capture the horizontal diffusivity governing the densest part of the plume. Maps of $\langle \kappa_{\text{hor}} \rangle$ are then averaged over the whole overflow period from mid-January to the end of August to give an impression of its general spatial distribution, which is shown in Fig. 5.8 for two runs of the HIGH overflow scenario ($S_{\text{sill}} = 35.8$). The non-tidal vs. tidal comparison reveals that a tidally-induced increase in κ_{hor} (up to 3-fold compared to non-tidal runs) is greatest in the shallow regions around the Sørkapp and in other shallow areas where the depth is < 100 m (Fig. 5.8b). In fact the tides cause an increase in κ_{hor} in most areas, with the exception of small patches on the slope showing insignificant change (Fig. 5.8c).

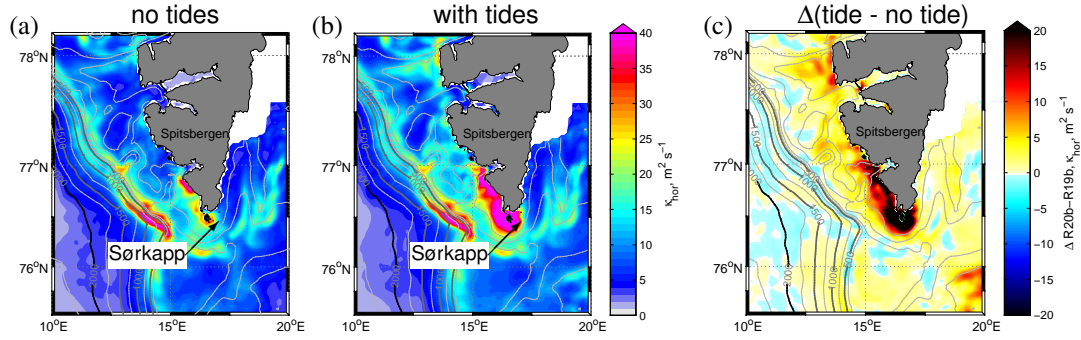


Figure 5.8: Horizontal diffusivity coefficient κ_{hor} in the bottom layer (0-50 m above seabed). Values in (a) and (b) are averaged over the modelled overflow period from mid January to end of August. The difference is shown in Panel (c) – warm colours ranging to black represent a tidally-induced increase in κ_{hor} .

Figure 5.8c clearly identifies a hotspot of tidally increased diffusion coefficients near the Sørkapp. Within a 65×45 km box centred on 76.4°N , 16.8°E (highlighted in Fig. 5.7a) spatial averages for a 50 m thick bottom layer are calculated of the horizontal diffusivity and vertical viscosity coefficients, as well as several horizontal tracer gradients. The resulting time series plots (Fig. 5.9) show both maximum and average values for a tidal run (blue) and a non-tidal run (red) of the HIGH overflow scenario.

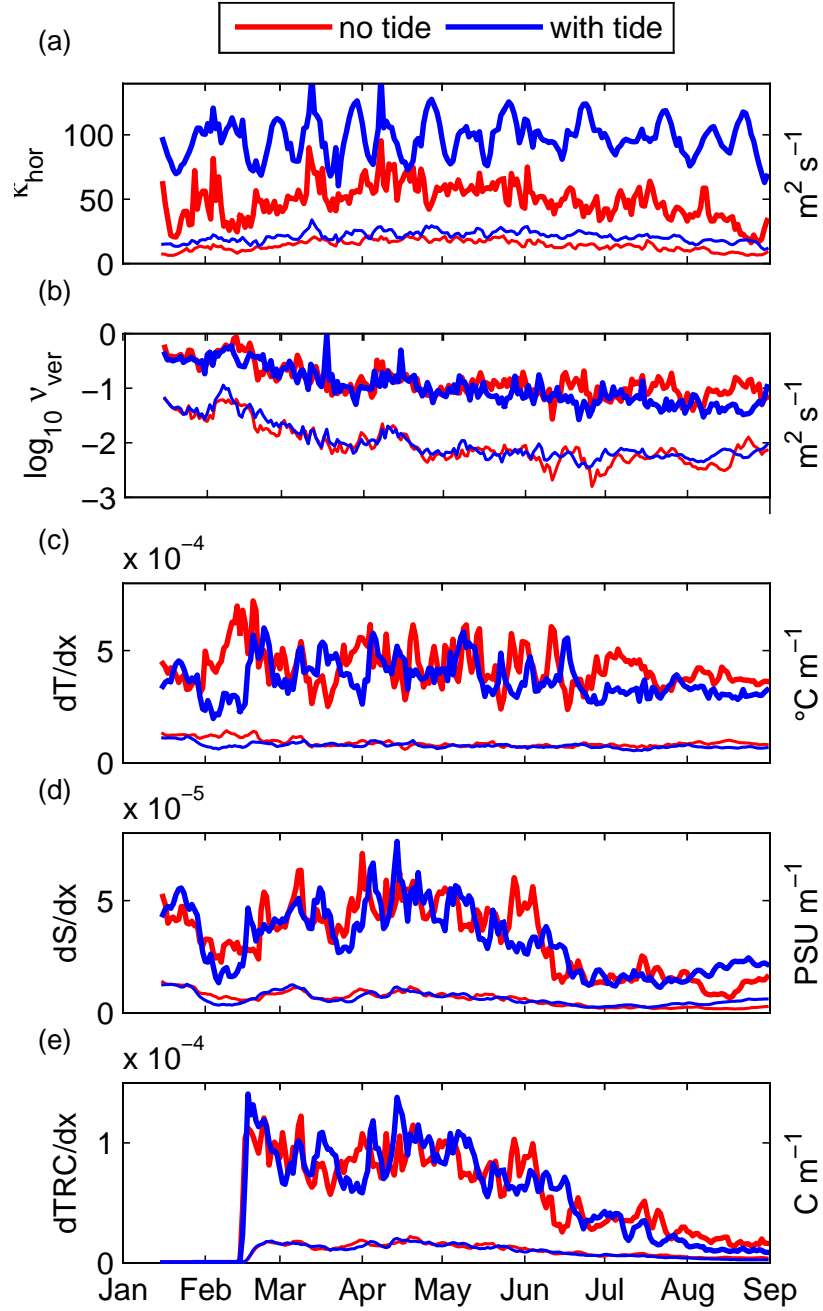


Figure 5.9: Time series of averages calculated within a 50 m thick bottom layer over an area around the Sørkapp headland (hatched box in Fig. 5.7a) for the non-tidal (red) and tidal run (blue) of the HIGH scenario ($S_{\text{sill}} = 35.8$). (a) horiz. diffusivity coefficient, (b) vert. viscosity coefficient, (c) horiz. temperature gradient, (d) horiz. salinity gradient, (e) horiz. gradient of passive tracer concentration. Bold lines are maximum values, thin lines are cross-sectional averages.

The time series of κ_{hor} in Fig. 5.9a emphasises the tidal effect on the horizontal diffusivity coefficient in the Sørkapp region. The maximum as well as average values of κ_{hor} around the Sørkapp are consistently higher in the tidal run. The same is true for all other pairs of non-tidal/tidal runs (not shown). The maximum values of the tidal κ_{hor} also reveal a low-frequency signal of 2-weekly periodicity, which corresponds with the spring-neap cycles in Fig. 5.2. Fig. 5.9b also reveals that tides do not significantly affect the vertical viscosity ν_{ver} .

Similarly, spatial averages of ∇T are also calculated in the box around the Sørkapp to investigate the tidal influence on the second factor in Eq. (5.1). The time series of horizontal gradients in temperature, salinity and passive tracer concentration are shown in Figs. 5.9c,d,e respectively. Those time series mostly show no or very little tidal effect on the tracer gradients. Only during July and August there are small differences in the maximum values of tracer gradients at a time when the flow of SFOW near the Sørkapp is already diminishing (Fig. 5.9e).

5.4 Discussion

In the ocean, the effects of tides cannot be observed in isolation from other processes. A regional model of Svalbard is set up to study the effect of tides on the evolution of a dense water plume from its formation in the Storfjorden to its descent into Eastern Fram Strait. By turning on and off the model's tidal forcing it is possible to isolate the tides' contribution to the descent of the Storfjorden overflow plume. The model uses realistic bathymetry, atmospheric forcing, initial and open boundary conditions, but the details of sea ice formation and polynya dynamics are parametrised rather than represented explicitly in the model. The parametrisation of the dense water outflow behind the sill is designed to approximate the effects of these processes on the dense water inside the fjord (as described by Haarpaintner et al., 2001; Skogseth et al., 2004, 2005a, and others). The analysis of the model results thus places less emphasis on the

upstream plume near the sill and rather focusses on its downstream evolution.

5.4.1 The 'AnSlope' hypothesis

Downstream of the sill (behind which the model parametrises the dense water overflow), the plume is topographically steered into Storfjordrenna, a channel between the Storfjorden and the continental slope of Western Svalbard. During this early stage of the flow the tidal runs show a thicker plume and faster drainage of dense water out of the fjord and towards the shelf edge. This effect is attributed to the mechanism suggested by Padman et al. (2009) and supported by analytical and numerical models (Ou et al., 2009; Guan et al., 2009, respectively). Their conclusions, based on results of the AnSlope experiment (Gordon et al., 2004), are referred to in the following as the 'Anslope hypothesis'.

Ou et al. (2009) proposed that tidal turbulence increases the plume thickness which lifts the density interface above the Ekman layer where it is shielded from further dilution by frictional processes. Ekman veering within a thicker bottom boundary layer, coupled with tide-induced shear dispersion thus enhances the spread of dense water on the shallow shelf towards the shelf slope (see also Wobus et al., 2011). Ou et al. (2009) further conclude that tides ultimately augment the downslope transport of dense waters from the shelf into the deep basins, which can be confirmed in the model for the upstream part of the flow where inclusion of tides leaves a lower concentration of tracer in the fjord area at the end of the run compared to non-tidal runs.

The tidally increased plume thickness on the shelf also agrees well with the 'AnSlope' hypothesis. The thickness of a dense plume in steady-state scales with the Ekman depth $D_{Ek} = \sqrt{2\nu_{ver}/f \cos \theta}$ (where f is the Coriolis parameter and θ is the slope angle, see Shapiro and Hill, 1997; Wåhlin and Walin, 2001). Yet, the greater plume thickness in tidal runs cannot be explained by an increased Ekman depth as the model shows that the vertical viscosity ν_{ver} is largely unaffected by the tides (Fig. 5.9b). Instead, horizontal

diffusion acting on a vertically sheared flow (see Geyer and Signell, 1992) is seen as a crucial mechanism resulting from the oscillatory nature of a tidal flow (Ou et al., 2009).

In general, shear dispersion arises from current shear and vertical diffusion (Taylor, 1953). A fluid parcel is first strained by shear in the direction of flow, then the upper and lower boundaries of the slanted column of fluid are blurred by vertical diffusion (Zimmerman, 1986; Geyer and Signell, 1992). The tracer is thus spread laterally by vertical mixing acting upon a horizontally sheared flow.

The modelled lateral dispersion of the plume is the result of a hotspot of tidally enhanced horizontal diffusion. The model's diffusion term derives from two factors, the diffusivity coefficients κ_{hor} based on the velocity field and the horizontal tracer gradient (Eq. (5.1)). Tidally enhanced shear alone is the cause for increased κ_{hor} values near the headland where velocities in the shallows reach 1 m s^{-1} during peak tidal flow and thus lead to greater bottom shear. The net result is increased diffusion in Eq. (5.1), even though the lateral tracer gradient ∇T , remains unaffected by the tides (Fig. 5.9).

While small transient eddies are not explicitly resolved in the model, the Smagorinsky (1963) scheme also captures the diffusive effect of such eddies which form around headlands in tidal flows (see Zimmerman, 1981). The boundary layer vorticity generated in the shallows along the headland separates from the coast and wraps up to form transient eddies that drive a strong recirculating flow near the headland (Signell, 1989; Geyer and Signell, 1992). But this eddy-driven form drag (McCabe et al., 2006) alone does not explain dispersion. Realistic bottom friction, as explicitly modelled in this study, gives rise to vertical shear as a result of tidal flow over the seabed. This generates the turbulence which itself is strained by vertical shear to enhance horizontal diffusion (Holt and Proctor, 2001).

The tidal-induced increase of horizontal diffusion is most pronounced during the intermediate stage of the cascade when the plume rounds the Sørkapp headland at the

southern tip of Spitsbergen. The consequences of the resulting widening of the plume are, however, different to those observed in Antarctica where a wide and thick plume was attributed to tidal effects. In Svalbard the result of a wider plume is that lighter fractions of overflow water are drawn into the Svalbard Coastal Current (here called the Sørkappstrømmen (SS) see Fig. 5.1) which transports tracer northwards on the shelf and into the mouths of the Bellsund and Isfjorden. This shallow surface current is present in non-tidal as well as tidal runs, but in non-tidal runs the overflow plume interacts to a lesser degree with this current by (i) being narrower as a result of smaller horizontal diffusion, and (ii) being thinner which allows the plume to ‘dive under’ the current which is strongest at the surface.

5.4.2 Two-layer structure and bifurcation

The model reveals that, beginning near the Sørkapp headland, the plume effectively forks into two branches separated by the shelf edge. The deep branch that sinks down the slope takes a path that compares well with observations (e.g. Quadfasel et al., 1988; Fer et al., 2003; Akimova et al., 2011). This part of the plume is naturally the denser water from the bottom layer of the plume that incorporates saline remainders from the previous winter (Fer et al., 2003). The shelf branch, being the lighter fractions overlying the dense core, is shown to be most affected by tidal dispersion and shown to widen under the influence of the tides. The widening towards the shelf, shown here to be caused by horizontal diffusion and enhanced by tidal effects, not only transports plume waters onto the shelf (Fig. 5.5b), but must in exchange also import shelf waters into the diluted upper plume layer. This leads to patches of low salinity above the plume core, which were first noted by Piechura (1996) and further investigated by Fer et al. (2003) in sections taken in Storfjordrenna. The latter study showed those fresher waters to have a salinity lower than both the plume core and Atlantic Water, which suggests intrusions of fresher shelf waters.

5.4. DISCUSSION

The present model results add to the picture of a two-layer plume by Fer et al. (2003). They showed that dense the bottom layer spreads at a rate consistent with Ekman veering while the diluted upper layer widens at twice the rate as the dense bottom layer. The model demonstrates that the upper layer is dispersed laterally by tidal effects at the headland. The subsequent widening of the plume's upper layer towards the shelf leads to lateral exchanges with the Coastal Current, which were already hinted at by Fer et al. (2003). It can now be demonstrated that the vertical structure of the plume translates into horizontal bifurcation beginning at the southern headland of Spitsbergen.

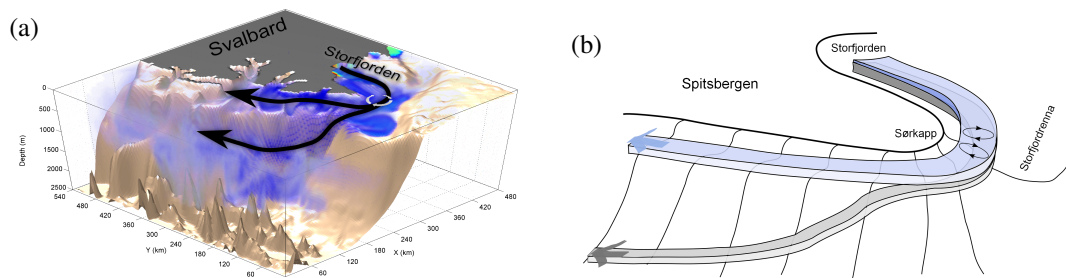


Figure 5.10: (a) Three-dimensional volume rendering of tracer concentration for the same day of the tidal run as in Fig. 5.5b (same colour scheme). (b) Schematic of tidal dispersion of the Storfjorden overflow plume. Tidally-induced shear dispersion near the Sørkapp diverts the lighter fractions of plume water from its upper layer onto the shelf.

Fig. 5.10 develops a schematic of this concept. Panel 5.10a shows an annotated three-dimensional view of the model output at the end of the same model run as in Fig. 5.5b (HIGH scenario, with tides). The overlaid arrows highlight the two branches of the plume separated by the edge of the Western Svalbard Shelf. A simplified schematic which highlights the two plume layers taking different pathways is shown in Fig. 5.10b. It should be noted that the depicted vertical separation into two layers is based merely on the qualitative definitions given by Fer et al. (2003) – a dense lower layer with a homogenised density structure and an diffuse upper layer with larger density gradients. Based on their observations Fer et al. (2003) additionally defined both layers by fixed

density thresholds. Any chosen threshold will depend strongly on the source water characteristics in a given year, and will make it inevitable that the bottom layer will grow thinner through mixing with the overlying layer. The layers depicted in the schematic in Fig. 5.10b are therefore merely conceptual, and no general quantitative definition can be given.

The diversion of overflow waters away from the main flow at the headland results in an overall reduction of downslope fluxes into deep Fram Strait for tidal runs compared to non-tidal simulations. The difference may appear small, but it is nevertheless the opposite of what would be expected based on previous findings by Padman et al. (2009), Ou et al. (2009) and Guan et al. (2009), who found that tidal effects increase the downslope transport of dense plumes. Their analysis could assume a wide and flat shelf as the closest headland (Cape Adare) is 65 km from the plume's path. The present study shows a discrepancy with their hypothesis and reveals as its cause the interactions of tides with the complex topography of Svalbard. While it is clear that Storfjorden Overflow Water does occasionally reach great depths in Fram Strait (Quadfasel et al., 1988; Schauer et al., 2003), it is proposed that intense overflow events occur together with increased transport of fjord water onto the Western Svalbard shelf.

5.5 Conclusions

The Storfjorden overflow plume is investigated using a regional model of Svalbard. Further understanding is gained with regards to (i) the AnSlope hypothesis (Padman et al., 2009; Ou et al., 2009; Guan et al., 2009) which elucidates tidal effects on the propagation of dense water plumes, and (ii) the vertical structure of the Storfjorden plume (Fer et al., 2003) and their response to tidal flow. In the upstream section of the flow the model runs including tides confirm a thicker plume and enhanced off-shelf transport of dense water out of the fjord as observed during the AnSlope project. Yet, in contrast to AnSlope results the relative downslope transport of Storfjorden water into

5.5. CONCLUSIONS

the deep Fram Strait is reduced by the tides. The following reasons for the discrepancy are identified.

The plume's path from its formation region in the fjord to the shelf break is intercepted by a headland where tidally-enhanced horizontal diffusion results in lateral dispersion of plume waters. The two vertical layers of the plume – a dense bottom layer and a diluted upper layer – are affected by tidal dispersion to different degrees. The widening of the plume and its dilution is enhanced in the upper layer which occupies to a greater degree the shallow coastal side where tidal effects are strongest. The division of the plume into two vertical layers can thus be translated into a bifurcation of the plume path into a deep branch and a shallow branch respectively.

The plume forks into two branches at the southern headland of Spitsbergen, where tidally-enhanced lateral exchanges merge the diluted upper plume layer into the shallow Coastal Current that transports lighter fractions of the plume northwards onto the shelf. Passing the headland at a greater depth, the dense bottom layer is less affected by tidal dispersion and thus able to maintain its core density and subsequently sink down the slope.

In conclusion, it is proposed that the mechanism of tidally-induced lateral dispersion of a dense water plume could potentially be significant in other cascading regions where similar topographic features intersect the plume's path between its source and the shelf edge.

Chapter 6

Summary and further work

6.1 Summary of conclusions

The proper representation of bottom friction and Ekman veering in the bottom boundary layer is found to be crucial to accurately capture cascading. The traditional quadratic drag law parametrisation for bottom friction is shown to be insufficient to accurately represent the cross-slope transport of the cascade. A new friction parametrisation which incorporates the Coriolis force is not currently available, hence bottom friction is explicitly resolved rather than parametrised. This is achieved by using fine resolution near the bottom together with no-slip bottom boundary condition. The model, once configured accordingly, is successfully validated against laboratory experiments of cascading, which also shows that non-hydrostaticity is not required to model cascading.

Simplified models have been able to describe well a 2-layer system consisting of a homogeneous plume overlaid by a homogeneous ambient water mass of reduced density with a sharp interface between the two layers. The 3-D ocean models used here offer a advantage over those ‘reduced physics’ models. The ‘full physics’ model was used to represent diffuse plumes with a blurred interface between plume and ambient water. It was shown that the thickness of the interfacial layer compared to the thickness of the plume can be used as a simple guide to the applicability of ‘reduced physics’ models which are only applicable if the interfacial layer is thinner than the dense bottom layer of the plume. In a fully turbulent regime of increased diffusivity and viscosity the cascade’s downslope transport is increased due to a thicker bottom Ekman layer within which more plume water is frictionally veered in the downslope direction com-

pared to cases with thinner Ekman layers. An increase of diffusivity alone, which blurs the plume interface without changes to Ekman layer thickness, has the opposite effect and reduces downslope transport due to the upwards dilution of plume waters into the ambient fluid.

When modelling cascading into ambient stratification, the subgrid-scale parametrisation of lateral diffusion and the influence of spurious numerical mixing are crucial to the accuracy of the model. We effectively controlled spurious numerical mixing arising from the inclination of model levels to the horizontal by using a specially adapted vertical discretisation which maintains horizontal s -levels in the ocean's interior while near-bottom s -levels follow the slope of the bathymetry. The rotation angle of the Laplacian operator used in the horizontal diffusion parametrisation is also modified to minimise spurious diffusion effects.

An idealised model set up to resemble conditions in the Arctic Ocean investigates the descent of cold, dense shelf waters into ambient stratification. The model reproduces three regimes that were previously described for Arctic cascades, where dense shelf waters encounter a layer of warm, saline Atlantic Water upon their descent. The regimes are identified as (i) 'arrested', when the plume remains within or just below the Atlantic Layer, (ii) 'piercing', when the plume passes through the Atlantic Layer and continues to the bottom of the slope and an intermediate regime (iii) 'shaving', when a portion of the plume detaches off the bottom, intrudes into the Atlantic Layer while the remainder continues its downslope propagation.

The source water properties, its density defined by the salinity S at freezing temperature and the volume transport Q , define the 'strength' of a cascade by controlling the flux of potential energy at the dense water source. This potential energy is added to the system by introducing dense water at a shallower depth than where it would be neutrally buoyant within the ambient water column. In practical terms the cascade should

thus not be characterised just by its initial density difference alone, but additionally by its flow rate too.

It is demonstrated that the regime of the cascade is predictable, for an idealised case, given the source water properties S and Q . The predictive capability arises from a functional relationship between the potential energy flux (given by S and Q) and the penetration depth, and thus the regime of the cascade.

The influence of the tides on the Storfjorden overflow in Svalbard is investigated using a regional ocean model which includes actual bathymetry, realistic initial conditions, open boundary forcing, atmospheric forcing and tides. Comparisons between tidal and non-tidal simulations isolate the effect of the tides on the propagation of a dense water plume out of Storfjorden. Previous results from a wide shelf region in Antarctica had indicated that tidal mixing enhances the transport of dense shelf water towards the shelf edge. The Svalbard model supports these findings, but also identifies a hotspot of tidally-induced lateral dispersion at a headland located in the plume's path. At this headland, the plume branches and a proportion of the plume waters are dispersed onto the Western Svalbard Shelf, a process that is enhanced by the influence of the tides. In presence of such a topographic obstacle in the plume's path we find that tidal mixing during the shallow spreading phase on the shelf is shown to promote the diversion of plume waters onto the shelf instead of allowing them to sink into the deep ocean. The results thus identify a mechanism by which tides may cause a relative reduction in downslope transport, thus adding to existing understanding of tidal effects on dense water overflows.

6.2 Recommendations for further work

6.2.1 An improved bottom friction parametrisation

Ocean circulation models typically employ the quadratic drag law to parametrise frictional stresses at the sea bed (Gill, 1982). The bottom stress τ is thus *added* to the

model. A different approach, taken in this thesis, is to *resolve* τ by using a no-slip bottom boundary condition in conjunction with fine vertical resolution in the bottom boundary layer (BBL). Either way, a bottom stress τ implies a transverse Ekman transport $\tau/\rho f$ relative to the flow above. The accuracy in the calculation of the bottom stress, whether parametrised or resolved, is crucial for this transport, termed ‘Ekman drainage’ (e.g. by Shapiro and Hill, 1997). Chapter 3 showed that an ocean model’s ‘default setting’ for the bottom stress (i.e. the quadratic drag law) may lead to inferior results in the detailed study of bottom boundary layer processes.

When resolving the Ekman spiral in the velocity profiles the Coriolis effect is implicitly incorporated into the bottom stress term, leading to a more accurate representation of Ekman drainage in the boundary layer. This is seen as a reason for the improvement brought about by resolving the frictional stress. Due to the extra computational demands the fine resolution in the BBL required to resolve bottom stresses is not feasible in global climate models which would typically reserve only a single model level for the entire thickness of the BBL.

A new bottom friction parametrisation which includes the Coriolis force is thus called for. It would be able to represent Ekman veering in the bottom frictional layer and thus enable Ekman drainage (and hence cascading) to be presented in large-scale models without significant increases in computational cost.

6.2.2 A semi-analytical model of cascading into ambient stratification

The idealised scenario presented in Chapter 4 lends itself well to the development of a simplified (semi-)analytical model which gives the final penetration depth of the majority of the overflow water. This new model could be a special case of the Shapiro and Hill (1997) $1\frac{1}{2}$ -layer model. Its parameters would include the properties of the three ambient layers as well as the source water. Using various entrainment parametrisations the model could be tuned to fit with observations of the Storfjorden overflow. Such a

model could build on work by Wells and Nadarajah (2009), who relied on linear ambient stratification and constant entrainment coefficient. The work presented here (e.g. in Fig. 4.12) shows that entrainment varies between depths where the plume pierces an interface between two ambient layers and when the plume flows through a largely homogeneous layer.

6.2.3 Flexible vertical discretisation

Matching vertical and horizontal tracer gradients to observations has long been a challenge in ocean models. The ocean modelling community adopted the technique of using terrain-following vertical coordinates (e.g. Song and Haidvogel, 1994) from atmospheric circulation models (where it was introduced by Phillips, 1957) in order to overcome computational constraints of traditional horizontal vertical coordinates (z -coordinate system) in the vicinity of steep slopes (e.g. Blumberg and Mellor, 1983). Lateral mixing in the ocean interior is largely horizontal while cascading dynamics occurs along the steep slopes of continental shelves (Griffies, 2004). The novel s_h -coordinate system (Section 2.3.1) for vertical discretisation is an attempt at improving the representation of processes operating at varying angles to the horizontal in the bottom boundary layer, while the development of a new stretching function by Siddorn and Furner (2013) addresses the issue of varying vertical resolution in the surface boundary layer. The recent development of a hybrid between z and s -coordinates by Shapiro et al. (2013) continues this work. Several approaches towards time-evolving vertical grids have also been made, e.g. by Hiester et al. (2011) using Fluidity-ICOM (Piggott et al., 2008) or most recently by Gräwe et al. (2013) using a more traditional model (GETM, see Burchard and Bolding, 2002). Vertical discretisations, whether static or adaptive, are clearly an active area of research. It is hoped that future models will incorporate flexible approaches that can be tuned by the model's user to suit the particular problem without having to switch between separate codes.

6.2.4 Improvements in capturing deep plume characteristics

The models employed for Chapter 4 and 5 showed a deficiency in representing the cascade at depths greater than the Atlantic Layer as a thin dense layer at the bottom. While the modelled bottom plume is indeed a sharply separated dense bottom layer for some strong cascading conditions (e.g. in Fig. 4.2a), its thickness at depths greater than 1000 m often greatly exceeds that measured in observations. This issue of exaggerated plume thickness was also encountered by Fer and Ådlandsvik (2008) using ROMS with a standard s -coordinate formulation and a Mellor-Yamada turbulence scheme. It is possible that the turbulence schemes such as GLS (used in this thesis) and MY2.5 (used by Fer and Ådlandsvik, 2008) are not up to the task of faithfully representing turbulent mixing within the plume in presence of sharp ambient density gradients. Compared with observations (such as Quadfasel et al., 1988) the resulting plume appears too diffuse, suggesting excess vertical mixing as prescribed by the aforementioned turbulence models. The current crop of 3-D models (and their turbulence schemes) thus appear not to represent well the ‘deep mixing’ alluded to by Akimova et al. (2011). Future studies are hoped to further explore strategies of turbulence modelling with the aim of improving the plume representation at greater depths.

Finally, the author hopes that the research into modelling of cascading continues to eventually result in the inclusion of the process into global climate models for a much more refined representation of the global ocean circulation than is currently possible.

Appendix A

POLCOMS no-slip bottom boundary condition

A no-slip bottom boundary condition (NSBBC) is introduced in POLCOMS to optimise the model for cascading. In POLCOMS, the array `csq(i, j)` normally holds the value of the bottom friction term

$$csq = C_d \sqrt{(u^2 + v^2)}$$

in the original code using a quadratic drag law bottom boundary condition.

The no-slip bottom boundary condition is implemented by simply changing the calculation of `csq` to:

$$csq(i, j) = (2.0 * aa(1, i, j)) / (dsuv(1, i, j) * h(i, j) + tiny)$$

where `aa(1, i, j)` is the (below-)bottom value of eddy viscosity in U-columns and `dsuv(1, i, j)` is `sigov(2, i, j) - sigov(1, i, j)`, which is the non-dimensional height of the bottom grid cell. This makes `dsuv(1, i, j) * h(i, j)` the dimensional height of the bottom grid cell.

This modification achieves the following. The opposite of the above-bottom velocity is applied to the below-bottom grid cell. Thus the velocity at exactly the sea bed is imposed to be zero (see Fig. B.1 for a sketch).

This new code is applied in `b3dinit.for`

```
...
do j=1, jesub
  do i=1, iesub
    if ( ipexu(i,j).ne.0 ) then
#ifdef NO_SLIP_BOTTOM_BOUND_COND
      ! no-slip bottom boundary cond.
      csq(i,j) = (2.0*aa(1,i,j)) / (dsuv(1,i,j)*h(i,j)+tiny)
#else
      csq(i,j) = cbf*sqrt(u(2,i,j)**2+v(2,i,j)**2)
#endif
      fb(i,j) = u(2,i,j)*csq(i,j)
      gb(i,j) = v(2,i,j)*csq(i,j)
    endif
  enddo
enddo
...
```

and again in `barot.for`:

```
...
#ifdef NO_SLIP_BOTTOM_BOUND_COND
  ! no-slip bottom boundary cond.
  csq(i,j) = (2.0*aa(1,i,j)) / (dsuv(1,i,j)*h(i,j)+tiny)
#else
  csq(i,j) = cbf * sqrt(u(2,i,j)**2 + v(2,i,j)**2)
#endif
  fb(i,j) = u(2,i,j) * csq(i,j)
  gb(i,j) = v(2,i,j) * csq(i,j)
...
```

The code thus modified was used for the study in Chapter 3.

Appendix B

NEMO no-slip bottom boundary condition

A new value for the namelist parameter `nn_bfr` (type of bottom friction) is introduced in NEMO. If set to 3, a ‘no-slip’ bottom boundary condition (NSBBC) (see e.g. Day, 1990) is used. This is similar to the ‘free-slip’ condition (setting `nn_bfr=0`), but adds the additional constraint of $U = 0$ at $z = z_{bottom}$. The initialisation of the friction variables is therefore identical to the free-slip condition. The following code is added in `zdfbfr.f90`:

```
SUBROUTINE zdf_bfr_init
...
SELECT CASE (nn_bfr)
...
CASE ( 3 )
    IF (lwp) WRITE (numout, *) 'no-slip bottom boundary condition'
    ! The friction terms are not needed
    bfrua(:, :) = 0.e0
    bfrva(:, :) = 0.e0
...
```

Nothing further needs to be done in `zdf_bfr()` which updates the bottom velocity trends in case of quadratic friction. This is because `zdf_bfr()` essentially contains the parametrisation of friction, but the NSBBC does not attempt to parametrise friction.

If a zero bottom velocity is imposed, this has an effect on the code dealing with the vertical diffusion of momentum (vertical viscosity), because the horizontal velocity has to gently approach zero near the bottom (without sudden jumps as are common in slab-model parametrisations).

The vertical viscosity is computed in `dyn_zdf_imp`, which updates the after-velocity (u_a, v_a) with the results of the vertical diffusive mixing trend given by:

$$dz(avmu\ dz(u)) = \frac{1}{e3u}(dk+1) \left(\frac{avmu}{e3uw} dk(ua) \right)$$

Where `avmu` is the vertical viscosity in a given grid box.

The resulting system of equations is a tridiagonal system which is computed using the standard matrix methods. Incidentally, NEMO does not employ the `tridag()` function from the Numerical Recipes toolkit (as e.g. POLCOMS does, see Appendix A), but implements its own code based on recursive doubling which lends itself better to parallelisation, as described in Hockney and Jesshope (1988, chapter 5.4.2 ‘Recursive doubling’).

In NEMO the U-point where the last velocity at $z = jpk - 1$ is calculated actually above the seabed by half a grid cell, while the last velocity at $z = jpk$ lies half a grid cell below the seabed. The point at which the bottom boundary condition $U = 0$ should be applied therefore lies halfway between the last two grid cells (see Fig. B.1).

The way to impose the bottom boundary condition follows the same general approach as in POLCOMS (see Appendix A) - i.e. the tangential velocity between the bottom two grid cells will be zero. This is achieved when the two velocities above and below the bottom cancel out one another:

$$U_{jpk-1} = -U_{jpk}$$

This assumes that the last two grid boxes are exactly the same height (which may not be strictly true). By cancelling out the velocities either side of the seabed, the bottom velocity becomes zero, which is what the NSBBC intends (Fig. B.1).

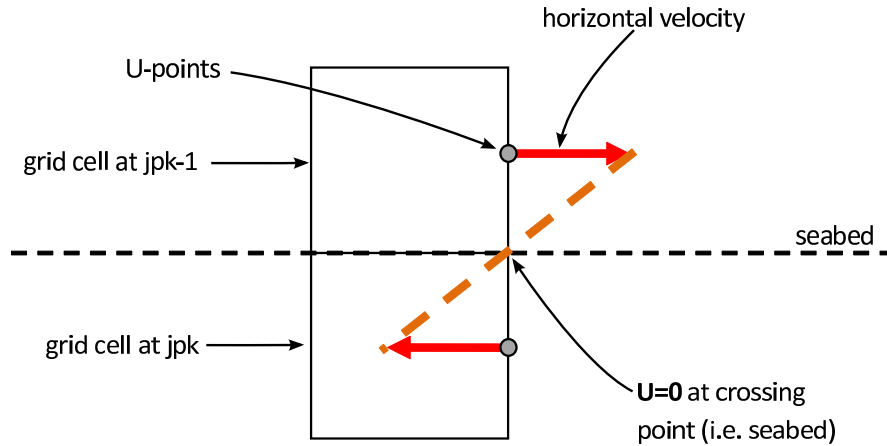


Figure B.1: Imposing a zero velocity at the sea bed by setting horizontal velocities above and below the sea bed to opposite signs. Cell indices follow the NEMO nomenclature.

The original NEMO code did not anticipate the introduction of a NSBBC and thus limits the computation of the matrix of velocity equations to levels 1 to $jpk-1$, which omits calculation of the reversed velocity in the below-bottom grid cell (below the seabed). NEMO assumes that the adjustment to the velocity based on the friction term has already been calculated (in `zdf_bfr`) and any treatment of below-bottom velocities would therefore be unnecessary. The tridag code therefore needs to be changed to perform the matrix calculations on levels 1 to jpk . This brings about several changes to `dyn_zdf_imp()`.

In addition to the code setting the surface end of the matrix, the bottom needs to be initialised too. Below the loop marked “Surface boundary conditions”, the following is added:

```
#ifdef key_nsbbc
  DO jj = 2, jpjm1                ! Bottom boundary conditions
    DO ji = fs_2, fs_jpim1        ! vector opt.
      zwd(ji,jj,jpk) = zwi(ji,jj,jpk) - 1._wp
      zws(ji,jj,jpk) = 0._wp
    END DO
  END DO
#endif
```

The first recurrence loop is changed from `DO jk = 2, jpkm1` to `DO jk = 2, jpk`.

The second recurrence (which applies the viscosity) is left unchanged, but an extra calculation is added for the bottom, which applies the condition sketched out in Fig. B.1:

```
#ifdef key_nsbbc                                !=====
    IF (nn_bfr == 3) THEN                        !== no-slip bottom boundary condition ==
        DO jj = 2, jpbm1                        !=====
            DO ji = fs_2, fs_jpbm1              ! vector opt.
                ! impose zero tangential velocity at the seabed,
                ! which is halfway between last 2 U-points
                ! (jk = ikbu & ikbu+1)
                ikbu = mbku(ji, jj)              ! deepest ocean u-level
                ua(ji, jj, ikbu+1) = - 0.5_wp * (ub(ji, jj, ikbu) + ↵
                    ↵ un(ji, jj, ikbu))
            END DO
        END DO
    END IF
#endif
```

The mean of the now-velocity (`un`) and before-velocity (`ub`) is used for numerical stability. The loop that finishes off the second recurrence is again changed from `DO jk = 2, jpkm1` to `DO jk = 2, jpk`. The loop for the third recurrence is also changed to wrap up the bottom of the matrix, which is now at `jpk` rather than `jpkm1`.

```
        DO jj = 2, jpbm1                        !== third recurrence :
                                                !== SOLk = ( Lk - Uk * Ek+1 ) / Dk ==
            DO ji = fs_2, fs_jpbm1              ! vector opt.
#ifdef key_nsbbc
                ua(ji, jj, jpk) = ua(ji, jj, jpk) / zwd(ji, jj, jpk)
#else
                ua(ji, jj, jpkm1) = ua(ji, jj, jpkm1) / zwd(ji, jj, jpkm1)
#endif
            END DO
        END DO
```

The final loop which applies the matrix coefficients to `ua` in an upwards-looking direction is also extended by 1 level. It is changed from `DO jk = jpk-2, 1, -1` to `DO jk = jpk-1, 1, -1`. The loop marked “Normalization to obtain the general momentum trend `ua`” is also extended by one level from `DO jk = 1, jpkm1` to `DO jk = 1, jpk`. The same logic is then applied to the calculation of the V-velocities below.

In order for these modifications to work, one more change is necessary to the velocity masks. The code which first initialises the tridag matrix multiplies the u- and v-velocities with their respective masks `umask` and `vmask`. In the standard NEMO code, these masks are 0 for the last grid cell which is below the seabed.

This is changed in `dommsk.f90:dom_msk()` below the loop marked “2. Ocean/-land mask at u-, v-, and z-points (computed from `tmask`)” which calculates `umask/vmask`. Another loop is added to extend the mask values down by one level:

```
#ifdef key_nsbbc
    DO jj = 1, jpjml
        DO ji = 1, fs_jpiml      ! vector loop
            ikbu = mbku(ji, jj)      ! deepest ocean u- & v-levels
            ikbv = mbkv(ji, jj)
            umask(ji, jj, ikbu+1) = umask(ji, jj, ikbu)
            vmask(ji, jj, ikbu+1) = vmask(ji, jj, ikbv)
        END DO
    END DO
#endif
```

With these changes the velocity profiles show an Ekman spiral near the bottom (compare Fig. 3.5) which shows that the no-slip bottom boundary condition works as intended.

The numerical experiments in Chapters 4 and 5 were both configured to use this new bottom boundary condition. It should be noted that no comprehensive testing was undertaken to make sure the two settings `key_nsbbc` and `nn_bfr==3` would work

with any other configurations. However, the code modifications presented here are fairly simple and should not impact on the other boundary conditions if they are used with code compiled with the custom preprocessor key `key_nsbbc`.

Appendix C

NEMO lateral diffusion operator rotation

A new scheme for the rotation of the lateral diffusion operator is described in Section 2.3.4. This appendix describes the results from several test runs that were conducted to test the performance of the new scheme. The aim of the tests was to establish how well the model maintains the horizontal isopycnals in the ambient stratification. Note, that the runs described in this appendix were performed before the geometry of the experiments in Chapter 4 was finalised.

C.1 Test runs without injection

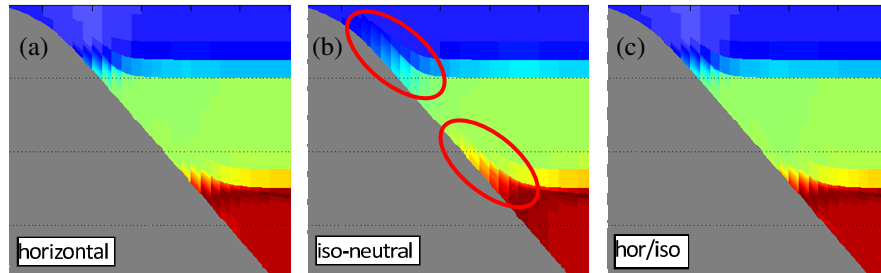


Figure C.1: Cross-sections of potential density σ_θ after 11.25 days. Three forms of rotation of the lateral diffusion operator are compared in runs without injection: (a) horizontal slopes for operator rotation (run Arc11Nldf_tLmL_3), (b) isopycnal operator rotation (run Arc11Nldf_iso_3), (c) new blended scheme (run Arc11Nldf_isoC_3). The up-slope creep of the ambient isopycnals is highlighted by red circles in (b).

The first set of test runs was performed without any injection. As the ambient water column structure is initialised with horizontal isopycnals, the lateral diffusion operator rotated in the horizontal (geopotential) direction performs well, because the isopycnals align with the plane in which lateral diffusion operates. The initially sharp isopycnals are nevertheless diffused over time, but this is expected in any model of this type. More

importantly the interfaces remain at the same depth after a period of time (Fig. C.1a).

Fig. C.1b shows that the default implementation of the iso-neutral operator rotation introduces an unphysical phenomenon - the diffusion is more pronounced and the interface creeps in the up-slope direction where the isopycnals intersect the slope.

Using the new blended rotation scheme (Fig. C.1c), the diffusion maintains the isopycnals at their initial depth yielding results nearly identical to the horizontal operator rotation. This is not surprising as the buoyancy frequency remains low throughout the domain of these control runs, and the blended rotation scheme reverts to the horizontal operator slopes most of the time.

C.2 Test runs with injection

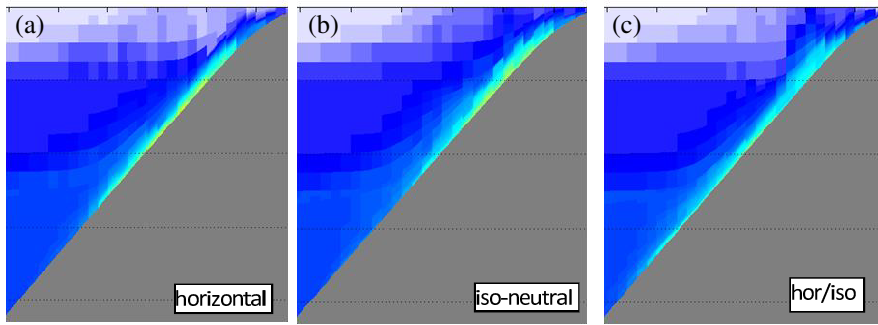


Figure C.2: Cross-sections of potential density σ_θ after 11.25 days. Three forms of rotation of the lateral diffusion operator are compared in runs with injection: (a) Arc11Nldf_tLmL_3, (b) Arc11Nldf_iso_3, (c) Arc11Nldf_isoC_3. The shown section is located 90° in a clock-wise direction (southern transect) from the off-centre point of injection (on the eastern side of the cone).

Another set of runs was performed with injection of dense water to study the impact of the different lateral diffusion schemes on the progression of the plume, which can be seen in Figs. C.2 and C.3. While the difference between the horizontal and iso-neutral operator rotation is hard to make out, the blended hor/iso rotation scheme has a noticeable impact on the plume height Fig. C.2c and Figure C.3c). While the plume exhibits an unexpected plume height much greater than the Ekman height (for vertical viscosity

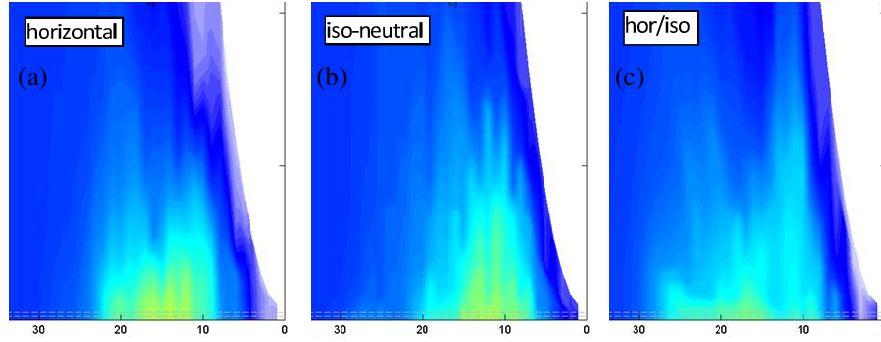


Figure C.3: Same as Fig. C.2, but drawn with the slope flattened out. The flow direction is right to left (0 - 30 km downslope). The vertical extent of the plot is 100 m.

of $1 \text{ cm}^2 \text{ s}^{-1}$, H_e is $\sim 1.2 \text{ m}$), the hor/iso operator does allow for a more ‘natural’ plume shape - a thick upslope bulge and a thinner tongue flowing down-slope. The unexpectedly thick plume (20 to 50 m) is thought to be caused by the injection of the source water over a very small area, which creates unphysical mixing effects. Furthermore, the injection point is located so close to the first interface (between Arctic surface water and Atlantic Water) that the injected flow has no time to adjust itself into a thin tongue before interacting with the interface between the 2 upper layers.

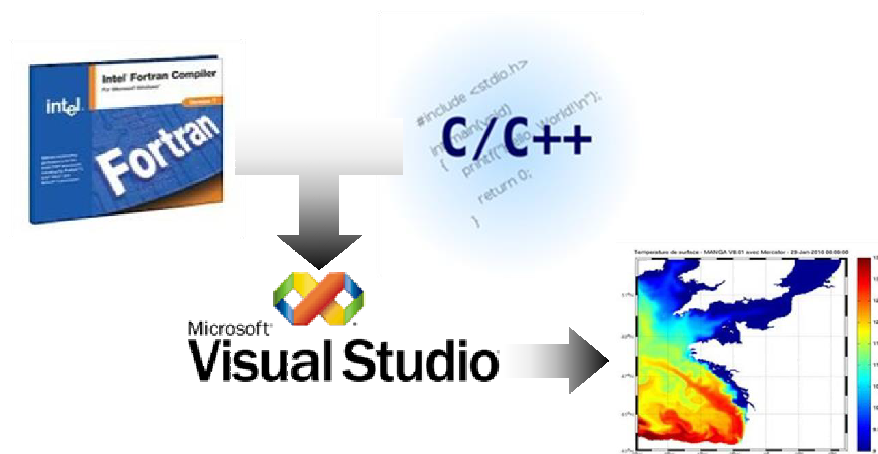
These results suggest that the new operator rotation scheme which blends iso-neutral (isopycnal) and horizontal (geopotential) slopes has the desired effect in maintaining the ambient isopycnal interfaces and favours the downslope propagation of the plume.

Despite the method of neutral surface slope calculation appearing rather crude and possibly not well suited to the vertical s -coordinate system, it is felt that the expected small improvement brought about by a more sophisticated scheme (such as 3-D contouring via kriging) would not be justified.

Note, that Griffies’s iso-neutral diffusion (inspired by Griffies et al., 1998) is provided in NEMO only from version 3.3 and was therefore not available in the v3.2 code used for the experiments in Chapter 4.

Appendix D

Guide to compiling NEMO and NetCDF on 32- and 64-bit Windows



D.1 Software versions used

The following versions of software were used to prepare this guide.

Microsoft Visual Studio 2008 Professional Edition

- with Service Pack 1 installed (installer file VS90sp1-KB945140-ENU.exe downloadable from <http://www.microsoft.com/download/en/details.aspx?id=10986>, will require internet connection to download the update).
- After installation, the Automatic Windows Update will find and download more updates to this service pack. Note: It is not known whether the additional installation of the service pack makes a difference to the ability to compile NEMO. However, service packs often fix important issues and problems with updates are rare.

Intel Visual Fortran Compiler Professional Edition v11.1.067 Update 7 for Windows

- installer file `w_cprof_p_11.1.067_novsshell.exe`

NEMO v3.2 source code derived from original v3.2 release with additional modifications by NOC-L, UKMO and UoP.

- The original (stock) code available from the NEMO SVN repository. First, one needs to register on the NEMO home page at <http://www.nemo-ocean.eu>. The same username/password is then used in SVN to download the source from following SVN URL: http://forge.ipsl.jussieu.fr/nemo/svn/tags/nemo_v3_2/NEMO
- For instruction on how to download the official code base and which SVN repository to use, see <http://www.nemo-ocean.eu/Using-NEMO/User-Guides/Basics/NEMO-Quick-Start-Guide>

NetCDF v3.6.3

- see <http://www.unidata.ucar.edu/software/netcdf/release-notes-3.6.3.html>
- download `netcdf-3.6.3.tar.gz` from <ftp://ftp.unidata.ucar.edu/pub/netcdf/>
- HINT: download any 3.6.x source code distribution which includes the `win32/NET` folder
- These releases are likely to ship with an older `cfortran.h` v4.3, which needs to be updated in order for the following instruction to work.

cfortran.h v4.4-14

- documentation by the author Burkhard Burow can be found at <http://www-zeus.desy.de/~burow/cfortran/> The latest version downloadable from its official home page is the older `cfortran.h` v4.3, which comes with netcdf v3.6.3.
- It is recommended to use a newer version v4.4-14 which contains a few useful modifications for compilation under Windows using Visual Studio.
- The updated versions can be found at http://cfortran.sourceforge.com/documentation/4.4-14/cfortran_8h-source.html (base URL is <http://cfortran.sourceforge.com/>) This download site provides linux-style packaged repositories for download, but it is not obvious how to download individual files. The `cfortran.h` file is listed as Doxygen-output within a web page, but its contents are easily copy-pasted from the web browser into a file to replace the one supplied in the netcdf 3.6.3 distribution.

- Note, that no attempt was made to review the changes that have been made since the original release by Burkhard Burow. Some of the later updates appear to have been made with Visual Studio in mind, so it made sense to use the updated version instead.

inttypes.h/stdint.h

- The netcdf library code requires the system header files `inttypes.h` and `stdint.h`. This header file defines the sized integers and is part of the C-99 standard, which is not fully supported by Microsoft Visual C++.
- These files are not available within Visual Studio 2008 by default, but can easily be downloaded and added to the project.
- Download the archive containing both files from <http://code.google.com/p/msinttypes/downloads/list> and copy the two header files (*.h) to `netcdf-3.6.3\win32\NET\libsrc`.

D.2 The Visual Studio solution and projects

The compilation is split into three projects grouped together within a single Visual Studio solution:

- **netcdf_lib** (*static C library*)
- **libnetcdf_f90** (*static F90 library*)
- **NEMO_v3_2** (F90 executable)

D.2.1 Why use static libraries?

The method presented here makes use of static libraries, which are compiled into the main code. One advantage is that the final executable file contains all code, and no

DLL-file has to be distributed with the .exe file. This simplifies installation. NEMO requires recompilation for every new model domain anyway, so it is no big deal to re-link the static lib every time. One disadvantage of a static library is that the linker is not run when the library is compiled. The programmer will therefore not be warned of any undefined symbols until the main program (i.e. NEMO) is linked. This can be overcome by compiling a dynamic link library (DLL), which is essentially a full program, but without a `main()` function (in C terms). The linker is invoked to construct the final DLL file, which will flag up any unresolved symbols at this stage. Another advantage of a DLL is the possible reuse of the same file without the requirement of recompilation. But, as already mentioned, this is only a very slim advantage as NEMO requires frequent recompilation anyway (e.g. when changing a preprocessor key or defining a new domain size, more on this in Section E.5).

To list the functions that are present in a static library, use the `DUMPBIN` tool provided with Visual Studio ¹. It can be run from the Visual Studio Command Prompt (not from the normal CMD prompt). The command

```
dumpbin.exe /symbols <libfile>
```

lists all the symbols used, the external ones have the 'External' prefix.

D.2.2 Set up the folder structure

1. Copy the NEMO code into the compilation folder.
2. Alongside the folder for the NEMO code, copy the unpacked (i.e. untarred/unzipped) `netcdf-3.6.3` folder and add the additional header files `inttypes.h` and `stdint.h` to the folder `netcdf-3.6.3\win32\NET\libsrc`.
3. Create an empty folder `sln` in the main directory alongside the NEMO folder and the NetCDF folder.

¹For more info, see [http://msdn.microsoft.com/en-us/library/c1h23y6c\(v=VS.90\).aspx](http://msdn.microsoft.com/en-us/library/c1h23y6c(v=VS.90).aspx)

4. Inside the `sln` folder create the folder `VS2008_comp_netcdf363` to hold the solution and project files. The folders are now arranged as shown in Fig. D.1.

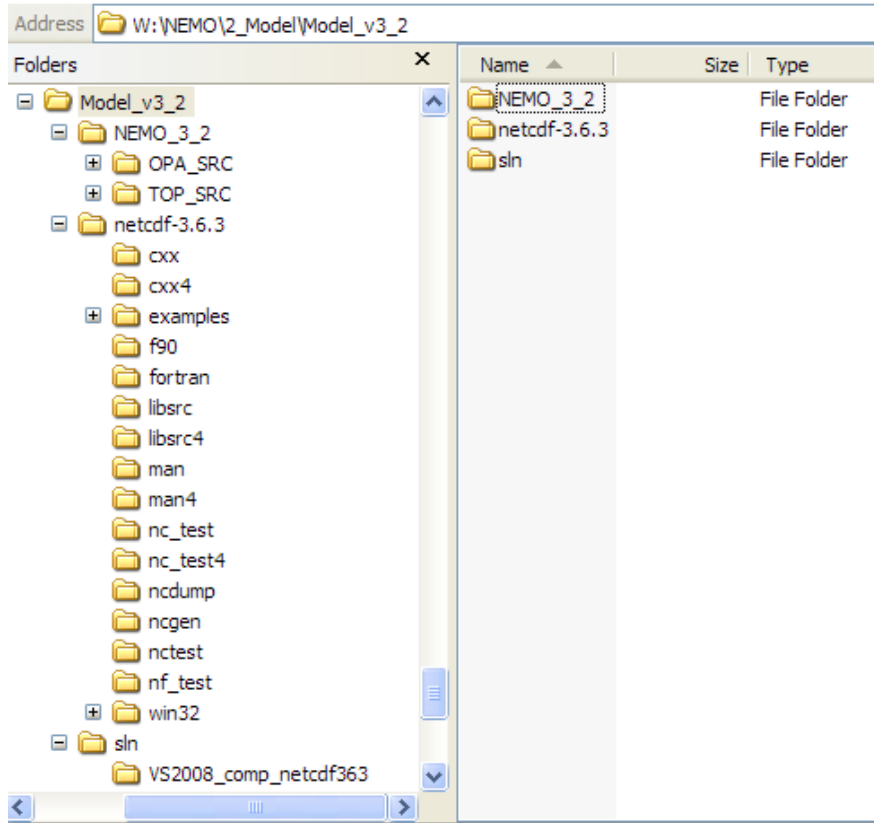


Figure D.1: Initial folder structure.

D.2.3 Create the NEMO_v_3_2 project

5. Open Visual Studio 2008 and create a new project (Intel(R) Visual Fortran → Console Application → Empty Project) for the model code. Enter `NEMO_v3_2` as the Name and select the `VS2008_comp_netcdf363` as the 'Location' (Fig. D.2). Click OK.

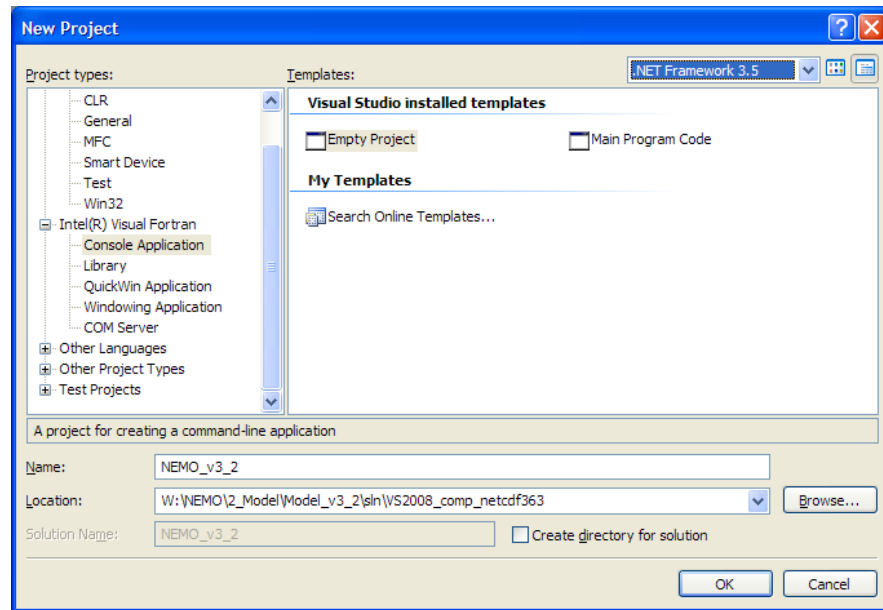


Figure D.2: Create a new Visual Studio Project.

6. Visual studio has the unwelcome habit of creating unnecessary subfolders for new projects. This can be rectified by hand:

Quit Visual Studio, save all projects. Go to the VS2008_comp_netcdf363 folder. Visual studio has created a subfolder NEMO_v3_2 containing a solution file NEMO_v3_2.sln and a Fortran project file NEMO_v3_2.vfproj. Copy both these files to the folder above, and delete the subfolder.

7. Load the solution and open the 'Solution explorer' window.
8. In the 'Solution explorer', delete the folder 'Resource files'.
9. Add all *.h90 files to the folder 'Header files' and all *.f90 files to the folder 'Source files'. It is advisable to create subfolders within 'Source folders' to mirror the folder structure of the source code. Do not forget to add a folder for the code in the OPA_SRC folder. The easiest way to add the files is by drag & drop (Fig. D.3)

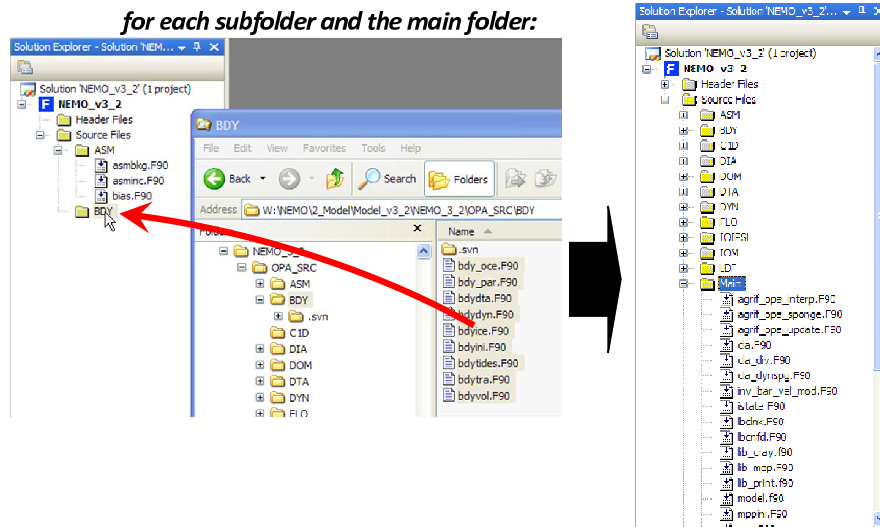


Figure D.3: Drag and drop source files to populate the project.

D.2.4 Create the libnetcdf_f90 project

10. Add a Fortran project called **libnetcdf_f90** which contains the NetCDF f90 bindings (Fig. D.4).

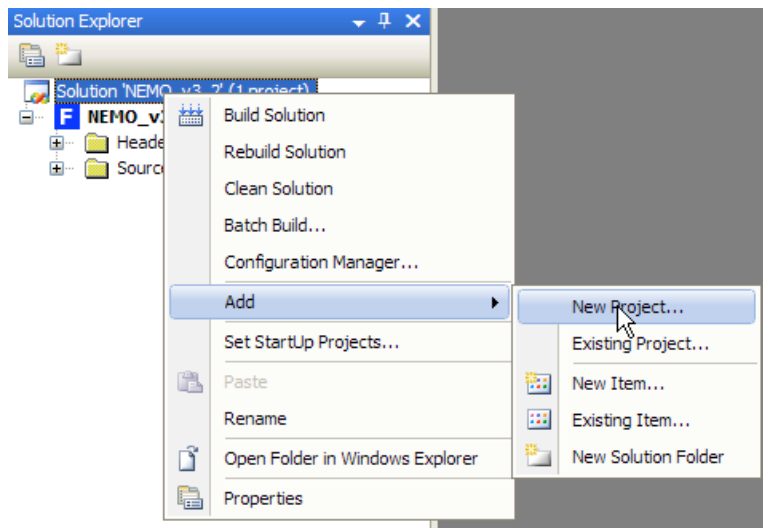


Figure D.4: Create a new Fortran project.

11. From the list of available Visual Fortran project types, choose 'Static library', enter `libnetcdf_f90` as the project's 'Name', use the same 'Location' and choose 'Add to Solution' (Fig. D.5).

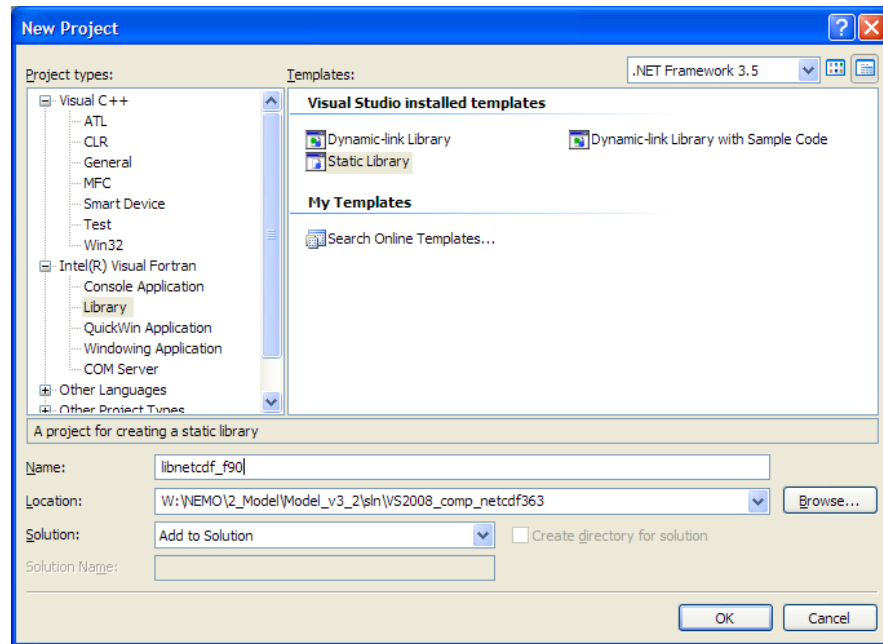


Figure D.5: New Fortran project dialog.

12. Visual Studio has done the same annoying thing by creating a subfolder for this project. Quit Visual Studio. From the `libnetcdf_f90` subfolder (which was created inside `VS2008_comp_netcdf363` folder), move the `libnetcdf_f90.vfproj` file to the solution folder (up one folder). Open the Solution file in a text editor, and modify the path from `libnetcdf_f90/libnetcdf_f90.vfproj` to `libnetcdf_f90.vfproj` to reflect the changed path of the project file (Fig. D.6).

```
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "NEMO_v3_2", "NEMO_v3_2.vfproj", "{2807028C-368C-4234-B0F9-AB5077E27BEE}"
EndProject
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "libnetcdf_f90", "libnetcdf_f90.vfproj", "{C6F869D0-29B4-48CB-8566-3B3434112616}"
EndProject
```

↓

```
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "NEMO_v3_2", "NEMO_v3_2.vfproj", "{2807028C-368C-4234-B0F9-AB5077E27BEE}"
EndProject
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "libnetcdf_f90", "libnetcdf_f90.vfproj", "{C6F869D0-29B4-48CB-8566-3B3434112616}"
EndProject
```

Figure D.6: Edit the solution manually to reflect a change in directory structure.

13. The source and header files for the `libnetcdf_f90` project are located in the `netcdf-3.6.3\f90` folder. Only two of the `*.f90` files are actual sources files, which will be compiled. The file `netcdf.f90` includes other `*.f90` files as if they

were header files.

14. Add `netcdf.f90` and `typeSizes.f90` to 'Source files'.

15. Add the source files for the NetCDF3 interfaces 'Header files'. These are all *.f90 files that do **NOT** start with `netcdf4...`

- `netcdf_attributes.f90`
- `netcdf_constants.f90`
- `netcdf_dims.f90`
- `netcdf_expanded.f90`
- `netcdf externals.f90`
- `netcdf_file.f90`
- `netcdf_overloads.f90`
- `netcdf_text_variables.f90`
- `netcdf_variables.f90`
- `netcdf_visibility.f90`

16. Select all header files, open the properties page and set

Exclude File from Build → Yes

and repeat for all other Configurations (Debug and Release), see Fig. D.7. Note that the icon for each file has changed. The icon no longer contains a down-arrow (the down-arrow means 'this file is to be compiled').

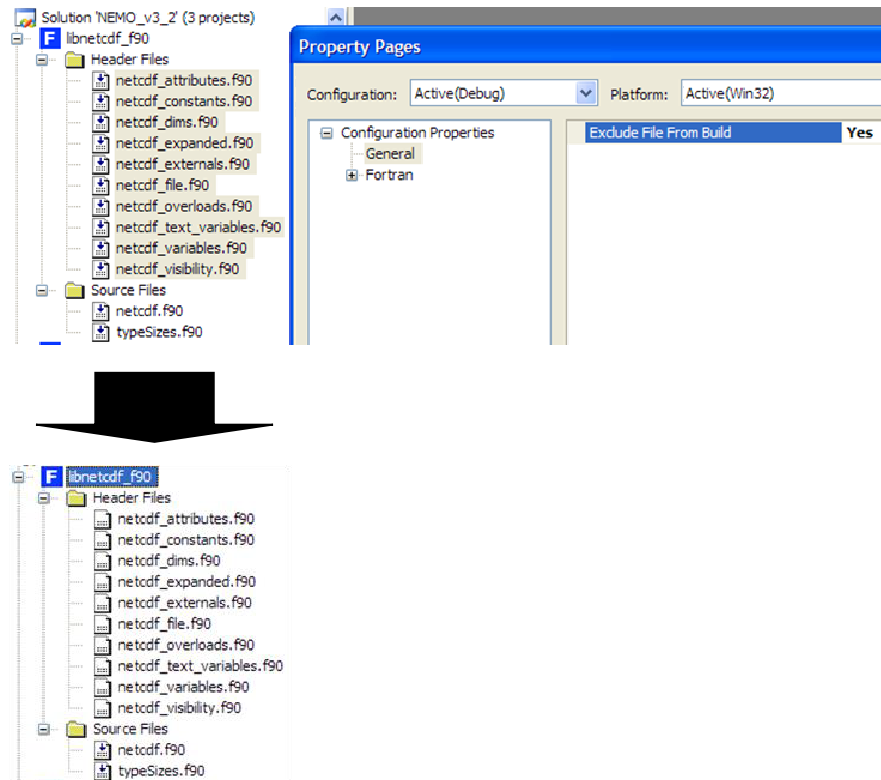


Figure D.7: The header files are excluded from the compilation.

D.2.5 Create the netcdf_lib project

17. Add a new Visual C++ project to the solution (Fig. D.8). This will be a project for the static library **netcdf_lib** to contain the NetCDF library code, which is written in C.

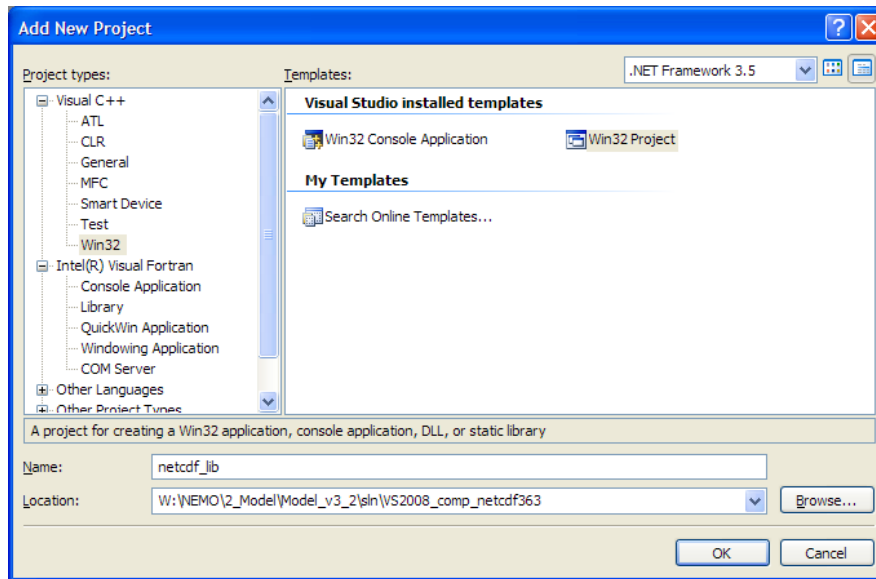


Figure D.8: Add a new Win32 project.

18. In the next screen, configure the 'Application settings' (Fig. D.9). Set the 'Application type → Static library' and uncheck 'Precompiled headers' and make sure that 'MFC' is also unchecked.

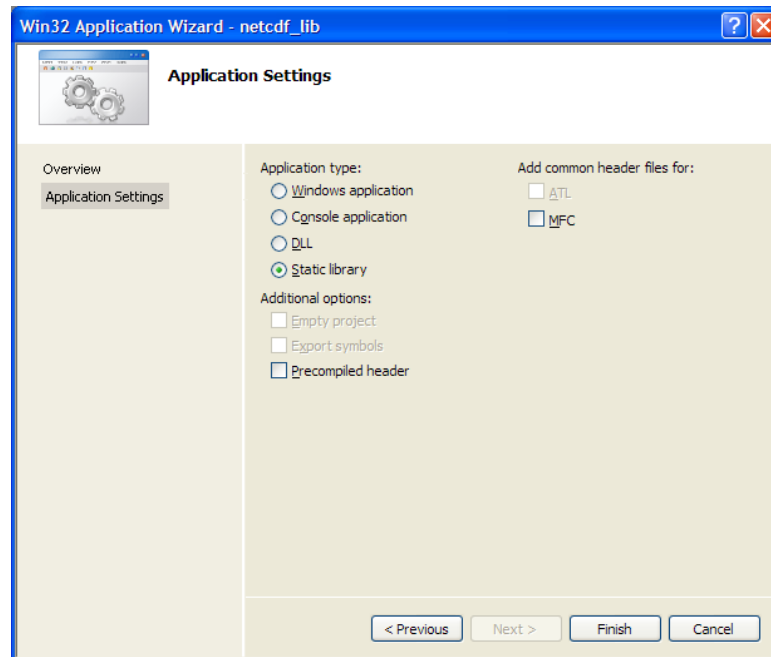


Figure D.9: Win32 application settings.

19. To tidy up the subfolder mess, quit Visual Studio (click ‘yes’ if it asks whether to save any changes). Move the `netcdf_lib.vcproj` file from its subfolder `netcdf_lib` to the solution directory. Delete the `netcdf_lib` subfolder (ignore the `ReadMe.txt` file - that can be deleted too). Modify the `NEMO_v3_2.sln` file in a text editor to reflect that move (as above for the **NEMO_v3_2** project), see Fig. D.10 .

```
Microsoft Visual Studio Solution File, Format Version 10.00
# Visual Studio 2008
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "NEMO_v3_2", "NEMO_v3_2.vfproj", "{28E...}
EndProject
Project("{6989167D-11E4-40FE-8C1A-2192A86A7E90}") = "libnetcdf_f90", "libnetcdf_f90.vfproj", "{...}
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "netcdf_lib", "netcdf_lib.vcproj", "{1...}
EndProject
```

Figure D.10: Beginning of the manually modified sln-file to reflect the desired directory structure.

20. Reopen the solution. Expand the **netcdf_lib** project in the Solution Explorer window. Delete the `ReadMe.txt` reference and the `Resource files` folder.

21. Add the source and header files from the `netcdf-3.6.3\libsrc` folder to this project. **DO NOT** add `ffio.c` (as it will not compile in Visual Studio and it is not required) and **DO NOT** add `fort-nc4.c` (the NetCDF4 interfaces are not required) or `t_nc.c` (which is actually for a test program).
22. Add the source and header files from the `netcdf-3.6.3\fortran` folder to this project.
23. There are additional files required for the Visual Studio compilation. From `netcdf-3.6.3\win32\NET`, add `config.h` to Header files and `getopt.c` to source files. From `netcdf-3.6.3\win32\NET\libsrc`, add `inttypes.h`, `stdint.h`, `nfconfig.inc` and `stdafx.h` to Header files and `stdafx.cpp` and `netcdf.cpp` to Source files. **NOTE**, that `inttypes.h` and `stdint.h` need to be downloaded separately (see above).
24. The source files included in the project (paths relative to the `netcdf-3.6.3` folder) are:

- `libsrc\attr.c`
- `libsrc\dim.c`
- `libsrc\error.c`
- `fortran\fort-attio.c`
- `fortran\fort-control.c`
- `fortran\fort-dim.c`
- `fortran\fort-genatt.c`
- `fortran\fort-gening.c`
- `fortran\fort-genvar.c`
- `fortran\fort-lib.c`

- `fortran\fort-misc.c`
- `fortran\fort-v2compat.c`
- `fortran\fort-varlio.c`
- `fortran\fort-varaio.c`
- `fortran\fort-vario.c`
- `fortran\fort-varmio.c`
- `fortran\fort-varsio.c`
- `win32\NET\getopt.c`
- `libsrc\libvers.c`
- `libsrc\nc.c`
- `libsrc\ncx.c`
- `libsrc\posixio.c`
- `libsrc\putget.c`
- `libsrc\string.c`
- `libsrc\utf8proc.c`
- `libsrc\v1hpg.c`
- `libsrc\v2i.c`
- `libsrc\var.c`

25. The header files included in the project are

- `fortran\cfortran.h`
- `win32\NET\config.h`
- `libsrc\fbits.h`

- `fortran\fort-lib.h`
- `win32\NET\libsrc\inttypes.h`
- `libsrc\nc.h`
- `libsrc\nc3convert.h`
- `libsrc\nc3local.h`
- `fortran\ncfortran.h`
- `libsrc\ncio.h`
- `libsrc\ncx.h`
- `libsrc\netcdf.h`
- `libsrc\netcdf3.h`
- `libsrc\netcdf3l.h`
- `libsrc\onstack.h`
- `libsrc\rnd.h`
- `win32\NET\libsrc\stdint.h`
- `libsrc\utf8proc.h`
- `libsrc\utf8proc_data.h`

26. The finished project is shown in Fig. D.11. The solution now contains all code necessary for compilation. Note, that files in the ‘Header files’ subfolder of the projects are not actually compiled it is useful to include them in the project for reference, so they get picked up by the Visual Studio search function and can easily be opened (by double click) from within the solution. The solution is now ready to be configured. Each project has been created with a 2 project configurations ‘Debug’ and ‘Release’. Many of the following settings will apply to both and need to be duplicated (unless specified).

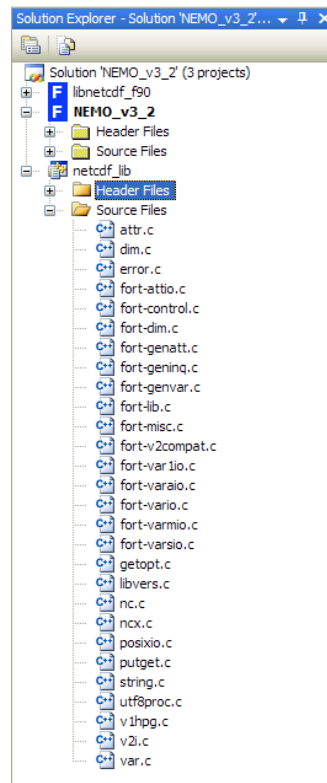


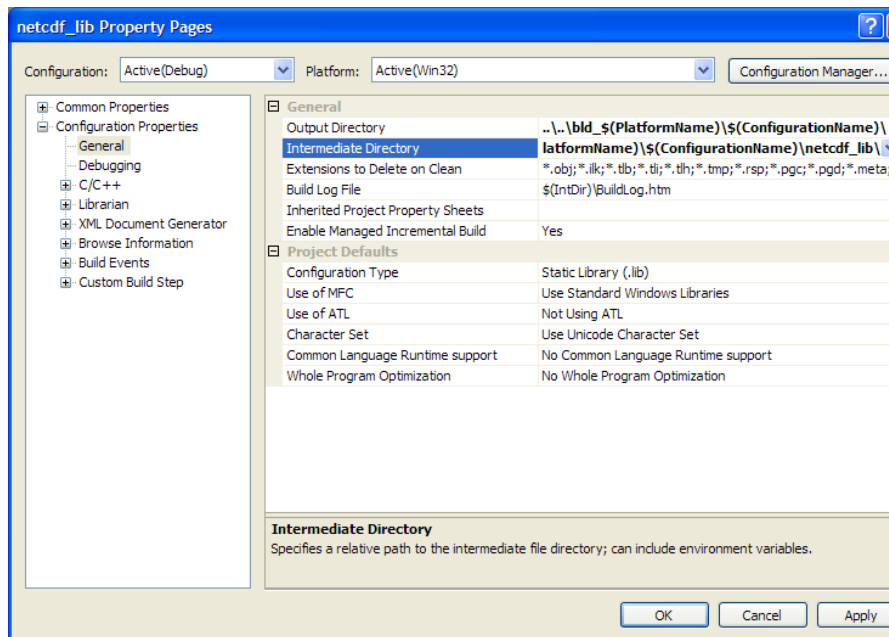
Figure D.11: Completed project setup with all source and header files.

27. Before proceeding, configure the project build order. Open the properties of the solution. Set **NEMO_v3_2** to be the 'Single startup project'. Right-click the solution and select 'Project dependencies' and configure: Project **libnetcdf_f90** depends on **netcdf_lib**. Project **NEMO_v3_2** depends on **libnetcdf_f90**.

D.3 Configure the projects (Compiler options etc.)

D.3.1 Configure the netcdf_lib project

28. Open the project properties of **netcdf_lib**. Turn to 'Configuration properties'.
29. In section General (see Fig. D.12), set (for Debug and Release):
Output directory → `..\..\bld_$(PlatformName)\`
↳ `$(ConfigurationName)\`
Intermediate Directory → `..\..\bld_$(PlatformName)\`



```
.
..\..\netcdf-3.6.3\libsrc
..\..\netcdf-3.6.3\win32\NET
..\..\netcdf-3.6.3\win32\NET\libsrc
..\..\netcdf-3.6.3\fortran
```

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

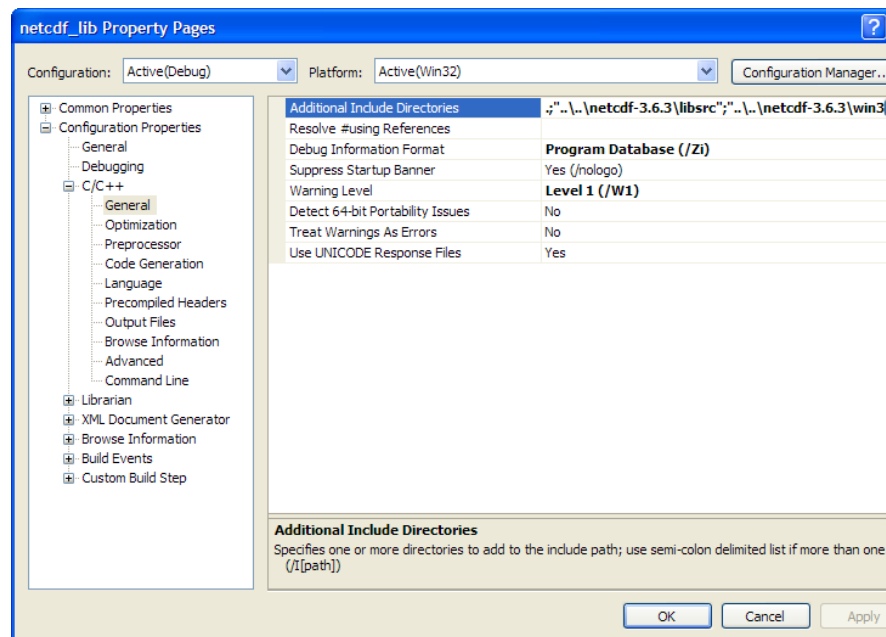


Figure D.13: C/C++ → General settings in the **netcdf_lib** project.

32. In the section C/C++ → Preprocessor (Fig. D.14) set the ‘Preprocessor Definitions’ (default should be e.g. WIN32 ; NDEBUG ; _LIB in Release) to read:

```
WIN32
NDEBUG          ( _DEBUG in Debug build )
_LIB
_WINDOWS
PowerStationFortran
VERSION=3.6.3
VISUAL_CPLUSPLUS
FCALLSC_QUALIFIER=__cdecl
_FILE_OFFSET_BITS=64
HAVE_STRERROR
```

The definition of `PowerStationFortran` might appear peculiar, but `PowerStation` was bought out by Intel who then rebranded the product as the Intel

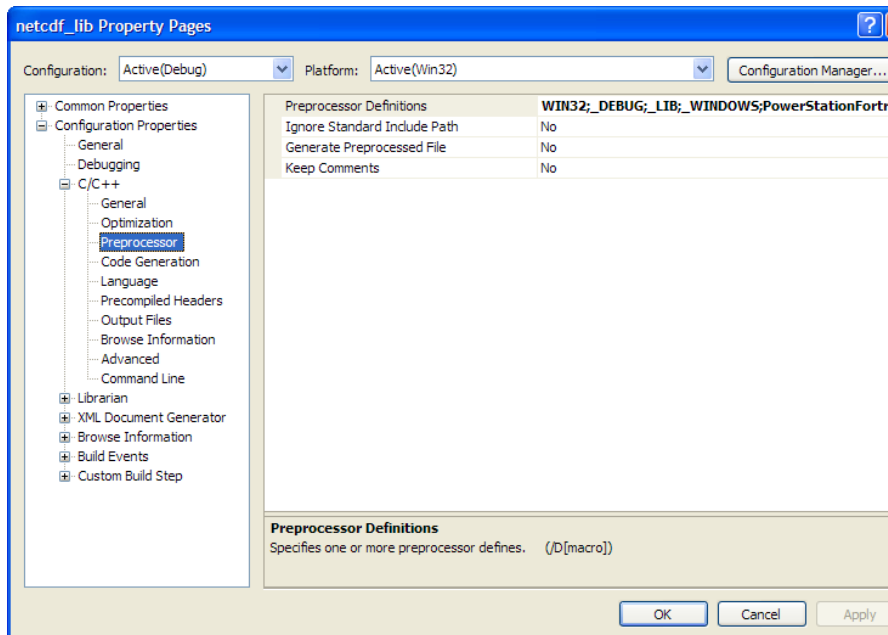


Figure D.14: C/C++ → Preprocessor settings in the **netcdf_lib** project.

Visual Fortran Compiler. This historic compiler key suggests that the development of `cfortran.h` has not quite kept up. The definition of the `FCALLSC_QUALIFIER` changes the calling convention (see also section 4.1) and must match the compiler switches for the other Fortran projects (Fortran → External Procedures → Calling convention → C, `REFERENCE (/iface:cref)`).

33. In the section C/C++ → Code Generation (Fig. D.15), set

Runtime Library → Multi-threaded Debug (/MTd)

- or -

Runtime Library → Multi-threaded (/MT)

for Debug and Release respectively. Also check that the 'Floating Point Model' is `Precise (/fp:precise)`. This should be the default.

34. Open the header file `config.h`. Change the following statements which defines the assumption on the size of the `off_t` type, which is just a long (4-byte

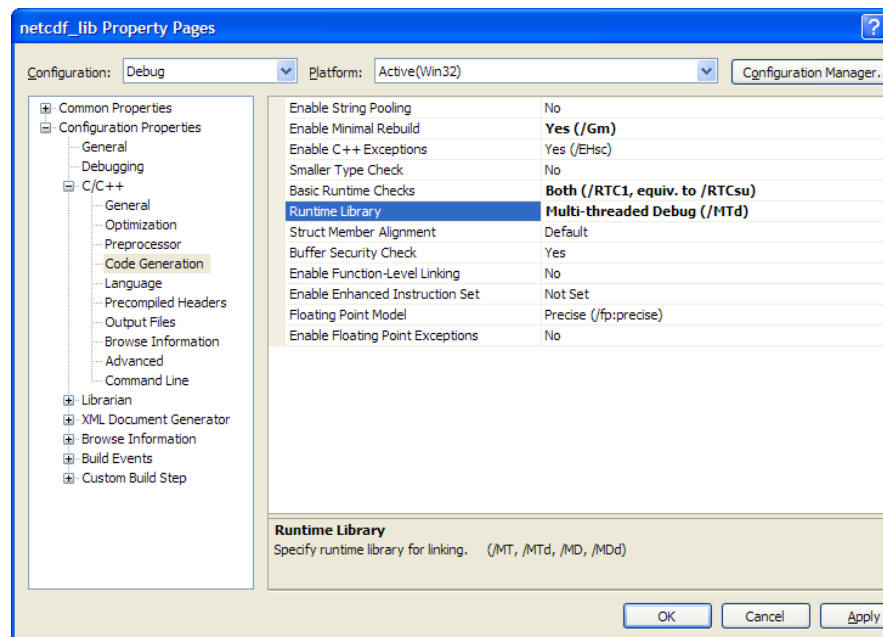


Figure D.15: C/C++ → Code Generation settings in the **netcdf_lib** project.

type) on 32-bit and 64-bit windows (see ² and the notes on 64-bit compilation in Section D.5):

```
/* The number of bytes in a off_t */
#define SIZEOF_OFF_T 4
```

35. The static library project **netcdf_lib** can now be compiled. There should be no warnings or errors.

D.3.2 Configure the libnetcdf_f90 project

36. Open the property pages for **libnetcdf_f90**. Note, that these properties now configure the Fortran compiler and will therefore look quite different from the properties available for a C/C++ project (Fig. D.16).

37. In the section General, set (for Debug and Release)

²For more on off_t see [http://msdn.microsoft.com/en-us/library/323b6b3k\(v=VS.90\).aspx](http://msdn.microsoft.com/en-us/library/323b6b3k(v=VS.90).aspx) and the example in <http://www.mail-archive.com/bug-coreutils@gnu.org/msg08412.html>

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

Output Directory → `..\..\bld_${PlatformName} \`

`↳$(ConfigurationName) \`

Intermediate Directory → `..\..\bld_${PlatformName} \`

`↳$(ConfigurationName) \libnetcdf_f90\`

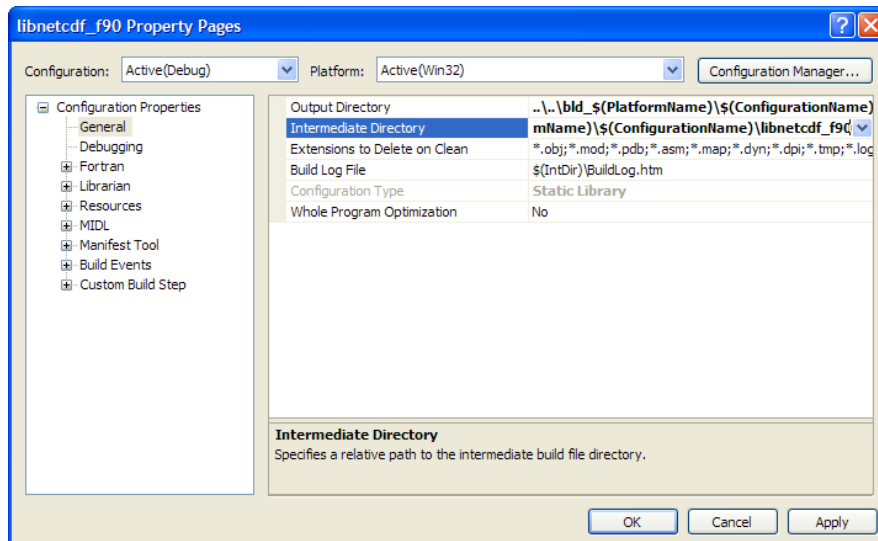


Figure D.16: General settings in the **libnetcdf_f90** project.

38. In the section Fortran → Preprocessor, set (as in Fig. D.17)

Preprocess Source File → Yes (/fpp)

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

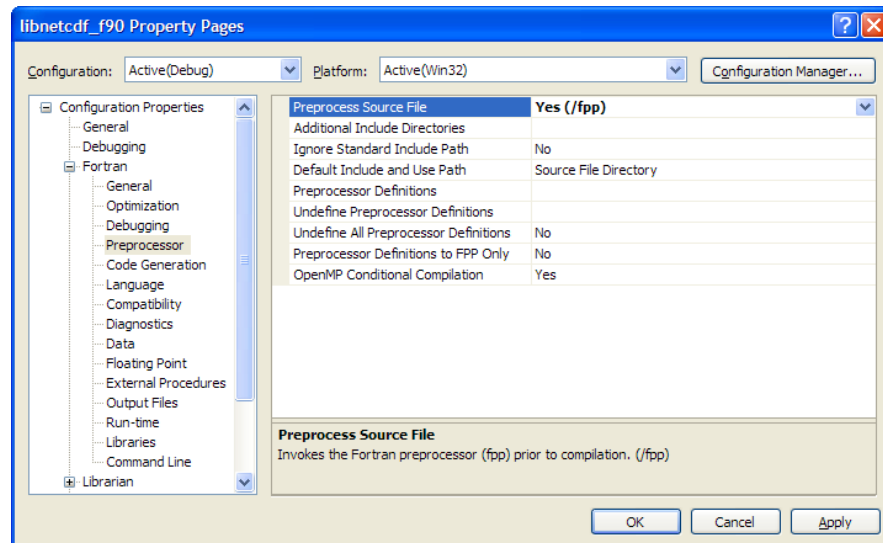


Figure D.17: Fortran → Preprocessor settings in the **libnetcdf_f90** project.

39. In the section Fortran → Data, set (as in Fig. D.18)

Default Real KIND → 8 (/real_size:64)

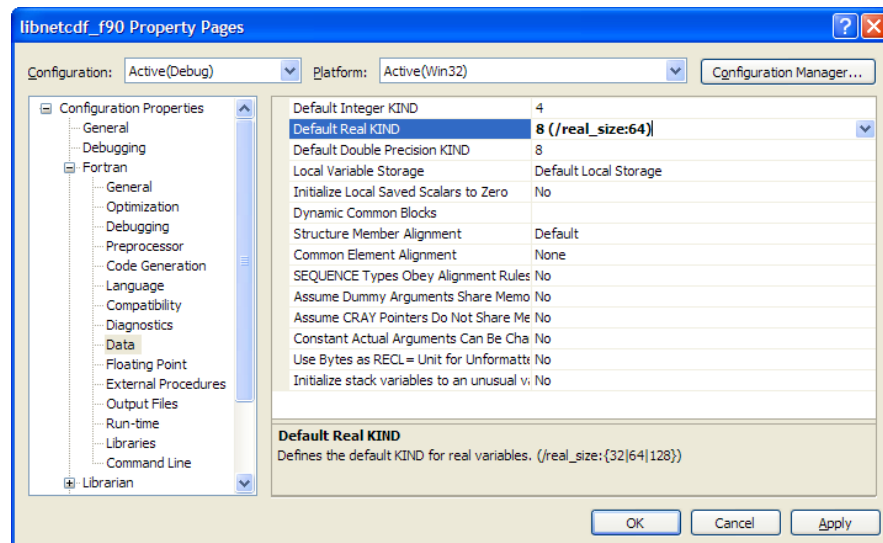


Figure D.18: Fortran → Data settings in the **libnetcdf_f90** project.

40. In the section Fortran → Floating Point, set (as in Fig. D.19)

Floating Point Model → Source (/fp:source)

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

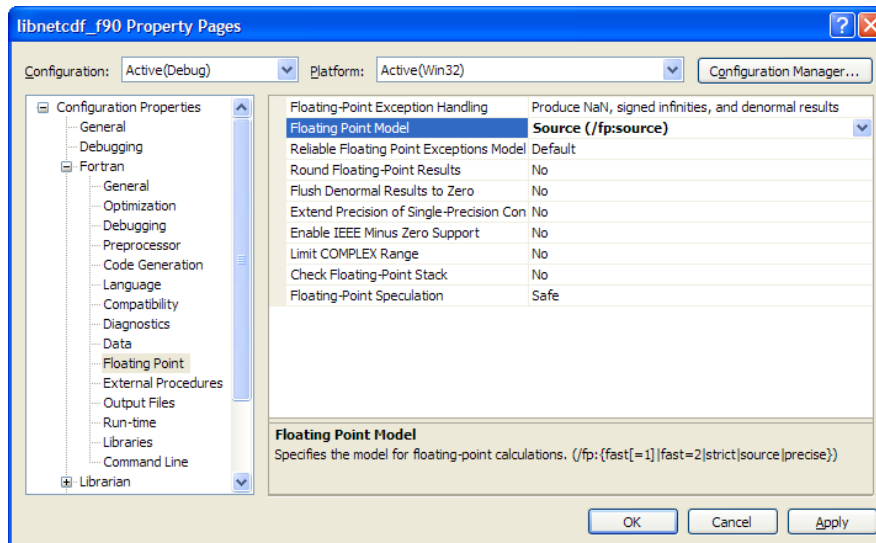


Figure D.19: Fortran → Floating Point settings in the **libnetcdf_f90** project.

41. In the section Fortran → External Procedures, set (as in Fig. D.20)

Calling convention → C, REFERENCE (/iface:cref)

Name Case Interpretation → Upper Case (/names:uppercase)

String Length Argument Passing → After Individual String Argument
(/iface:mixed_str_len_arg)

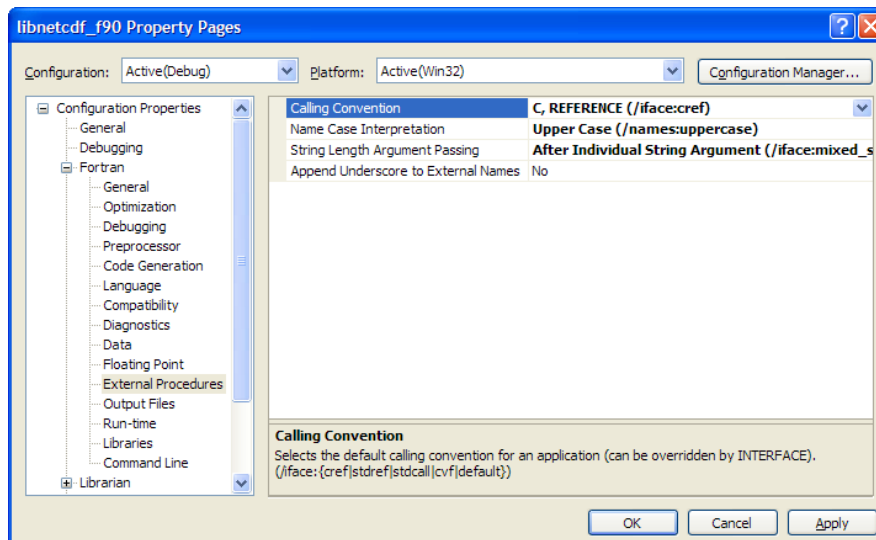


Figure D.20: Fortran → External Procedures settings in the **libnetcdf_f90** project.

42. In the section Fortran → Libraries, set (as in Fig. D.21)

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

Runtime Library → Debug Multithreaded (/libs:static /threads /dbglibs)

- or -

Runtime Library → Multithreaded

for Debug and Release builds respectively.

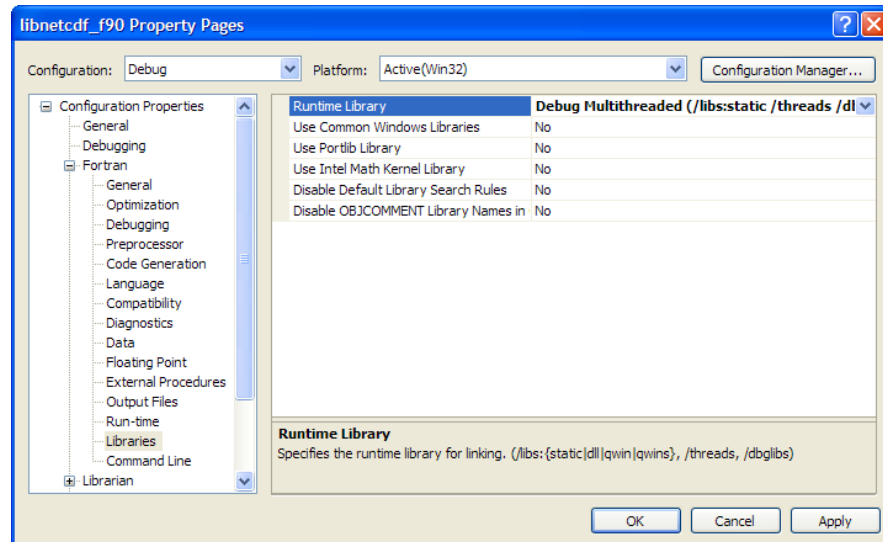


Figure D.21: Fortran → Libraries settings in the **libnetcdf_f90** project.

43. The static library project **libnetcdf_f90** can now be compiled.

D.3.3 Configure the NEMO_v_2 project

44. Open the project properties for the **NEMO_v3_2** project (Fig. D.22).

45. In the section General, set (for Debug and Release)

Output Directory → `..\..\bld_${PlatformName}\`

`↳$(ConfigurationName)\`

Intermediate Directory → `..\..\bld_${PlatformName}\`

`↳$(ConfigurationName)\nemo\`

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

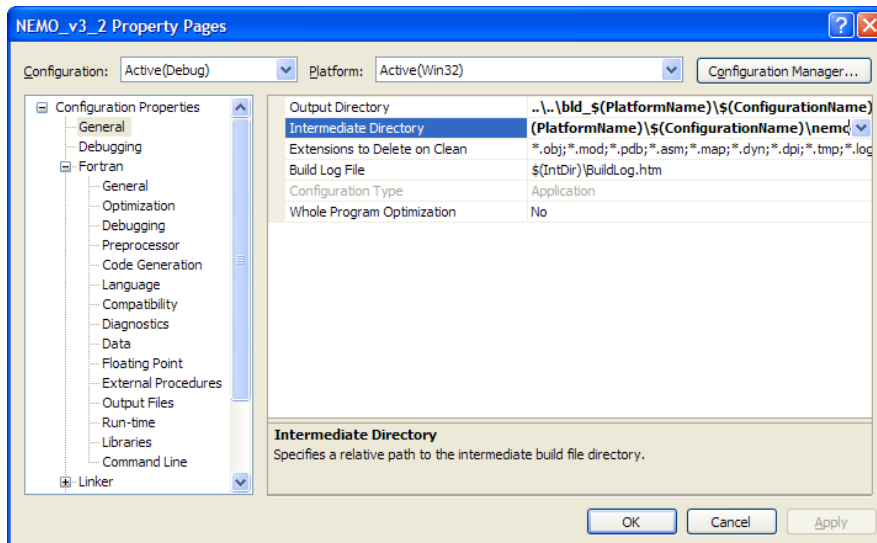


Figure D.22: General settings in the **NEMO_v3_2** project.

46. In the section Fortran → Preprocessor, set ‘Additional Include Directories’ to

```
..\..\bld_$(PlatformName)\$(ConfigurationName)\libnetcdf_f90
..\..\NEMO_3_2\OPA_SRC
..\..\NEMO_3_2\OPA_SRC\ZDF
..\..\NEMO_3_2\OPA_SRC\LDF
..\..\NEMO_3_2\OPA_SRC\DOM
..\..\NEMO_3_2\OPA_SRC\OBS
..\..\NEMO_3_2\OPA_SRC\TRA
```

47. In the section Fortran → Preprocessor, set (see Fig. D.23)

Preprocess Source file → Yes (/fpp)

48. Then add the required compiler keys `key_...` to the Preprocessor → Definitions according to the needs of the compilation (Fig. D.23).

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

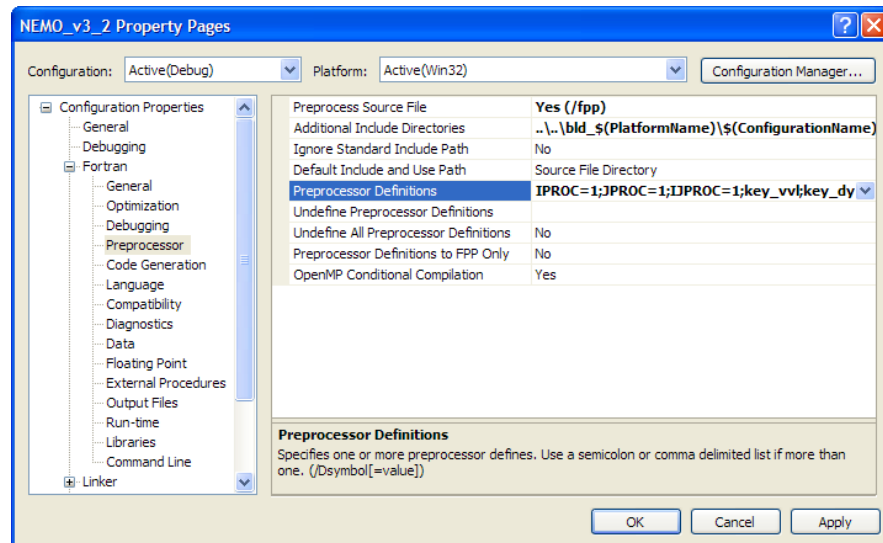


Figure D.23: Fortran → Preprocessor settings in the **NEMO_v3_2** project.

49. In the section Fortran → Data, set (see Fig. D.24)

Default Real KIND → 8 `(/real_size:64)`

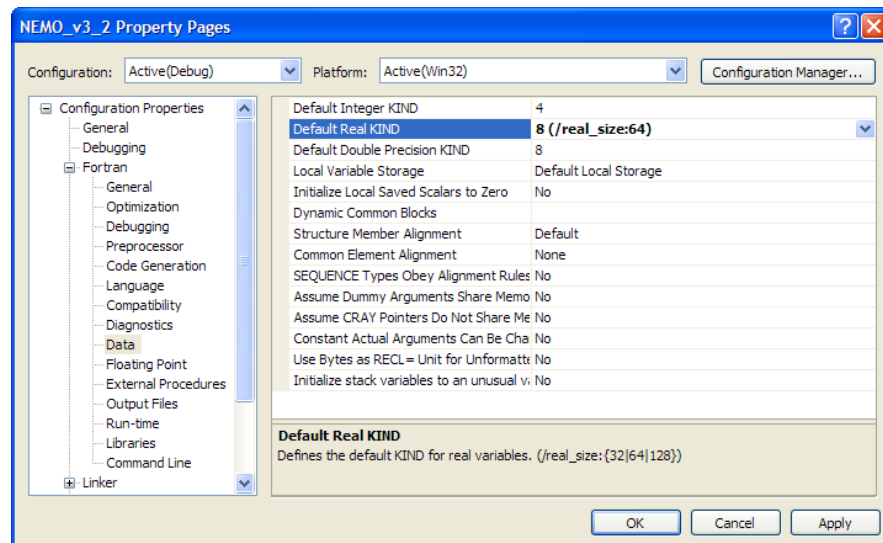


Figure D.24: Fortran → Data settings in the **NEMO_v3_2** project.

50. In the section Fortran → Floating Point, set (see Fig. D.25)

Floating Point Model → Source `(/fp:source)`

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

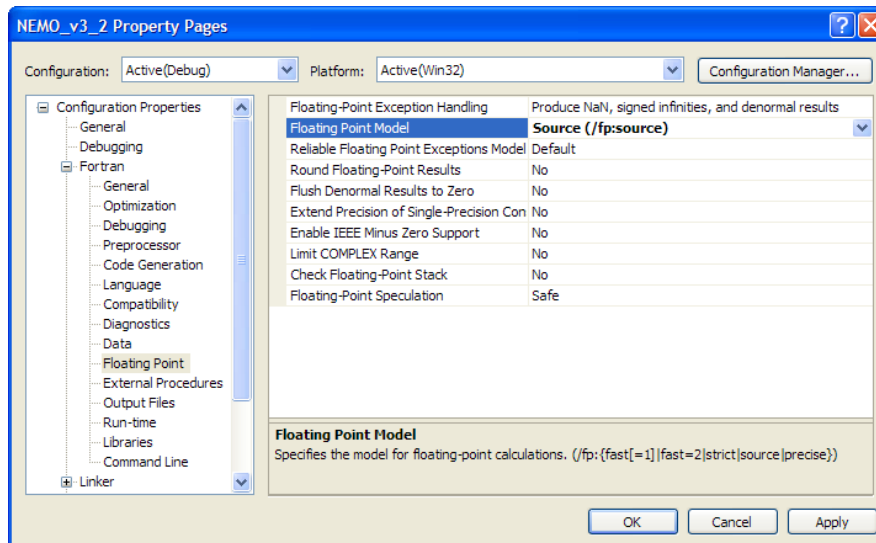


Figure D.25: Fortran → Floating Point settings in the **NEMO_v3_2** project.

51. In the section Fortran → External Procedures, set (see Fig. D.26)

Calling convention → C, REFERENCE (/iface:cref)

Name Case Interpretation → Upper Case (/names:uppercase)

String Length Argument Passing → After Individual String Argument
(/iface:mixed_str_len_arg)

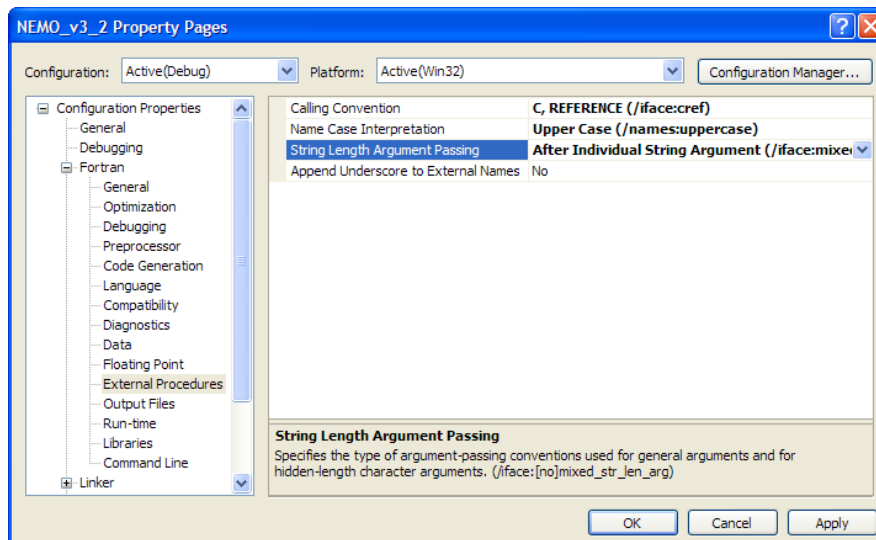


Figure D.26: Fortran → External Procedures settings in the **NEMO_v3_2** project.

52. In the section Fortran → Libraries, set (see Fig. D.27)

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

Runtime Library → Debug Multithreaded (/libs:static /threads /dbglibs)

- or -

Runtime Library → Multithreaded

for Debug and Release builds respectively.

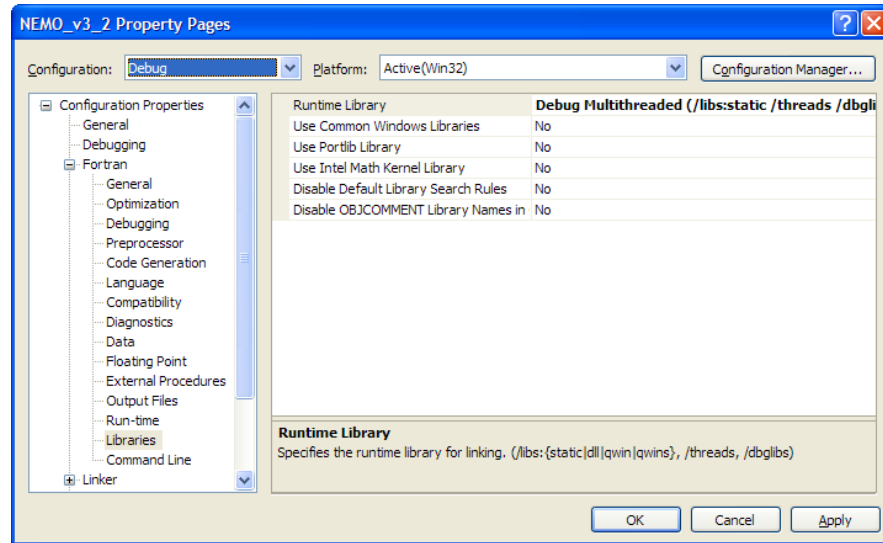


Figure D.27: Fortran → Libraries settings in the NEMO_v3_2 project.

53. Now configure the linking. Set (for Debug and Release) (Fig. D.28)

Output file → \$(OutDir) \model.exe

Additional Library Directories → .. \.. \bld_\$(PlatformName) \

↳ \$(ConfigurationName) \

Note that the 'Output file' setting is optional. If the default is used, the final executable will be called NEMO_v3_2.exe (as the default file name is derived from the name of the project). The file name for the final executable is ultimately down to personal preference.

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

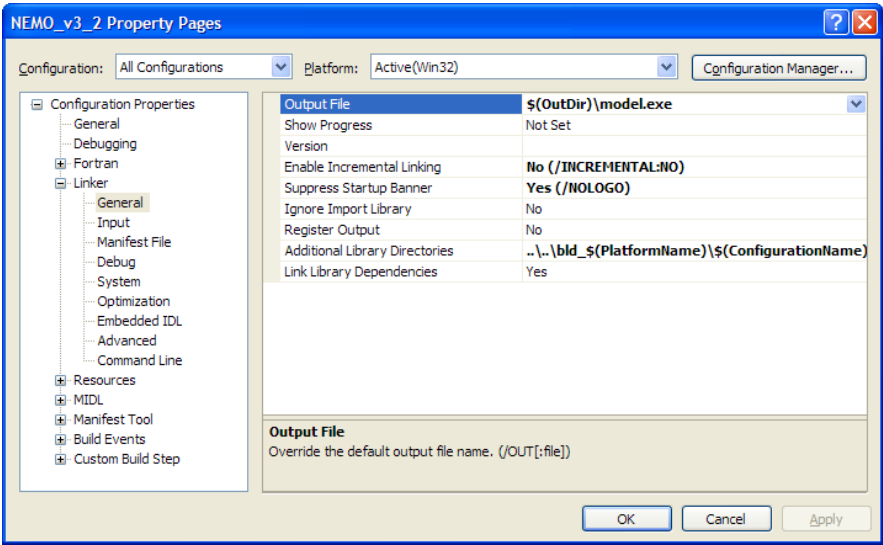


Figure D.28: Linker → General settings in the NEMO_v3_2 project.

54. In the section Linker → Input, set (see Fig. D.29)

Additional Dependencies → netcdf_lib.lib libnetcdf_f90.lib

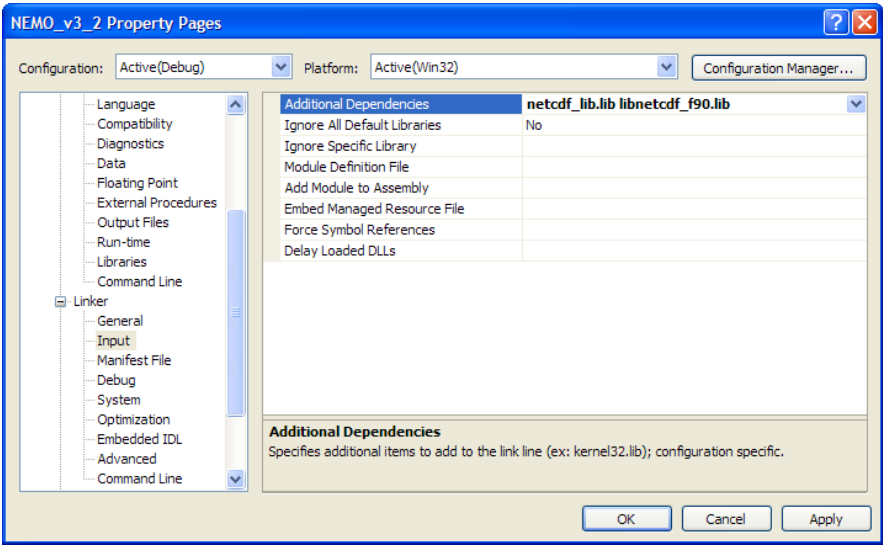


Figure D.29: Linker → Input settings in the NEMO_v3_2 project.

55. In the section Linker → Manifest File, set (for Debug and Release) (see Fig. D.30)

Manifest File → \$(IntDir)/\$(TargetName).intermediate.manifest

This is an optional setting which will prevent the main output directory from being cluttered with files that are specific to the Visual Studio compilation and

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

are no longer required for successful compilation.

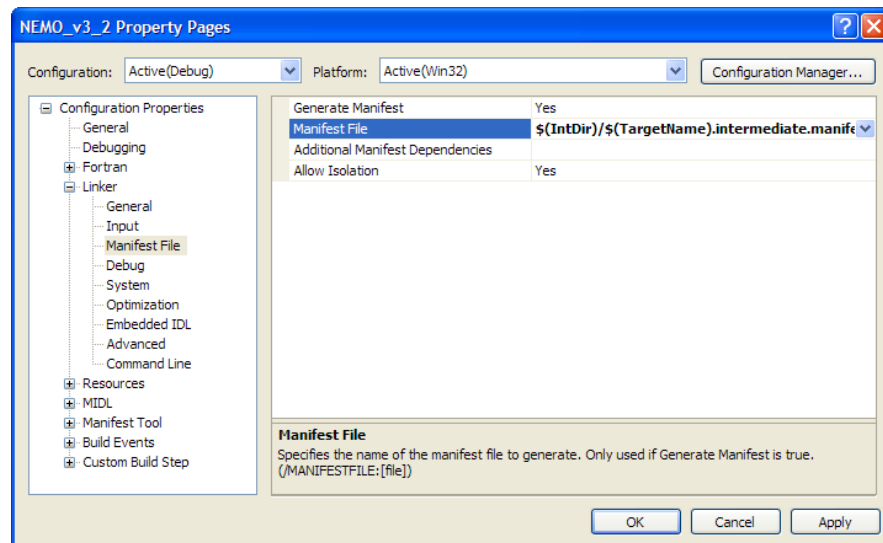


Figure D.30: Linker → Manifest File settings in the **NEMO_v3_2** project.

56. In the section Linker → Debug, set (for Debug only) (see Fig. D.31)

Generate Program Database File → `$(IntDir)\$(TargetName).pdb`

This is an optional setting which will prevent the main output directory from being cluttered with files that are specific to the Visual Studio compilation.

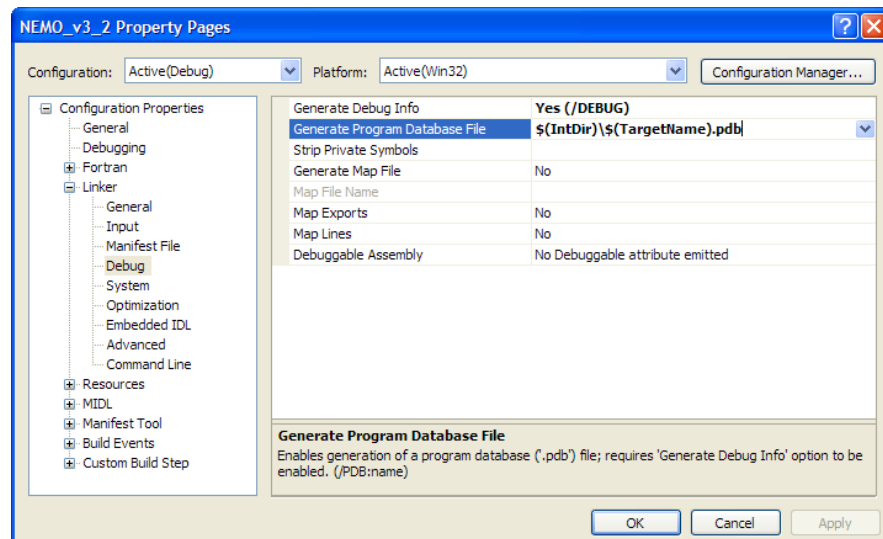


Figure D.31: Linker → Debug settings in the **NEMO_v3_2** project.

D.3. CONFIGURE THE PROJECTS (COMPILER OPTIONS ETC.)

57. In the section Linker → System, set (for Debug and Release) (see Fig. D.32)

Stack Reserve Size → 8000000

As NEMO v3.2 allocates all of its arrays on the stack, the required stack size will depend on the size of the model domain. The given value has been tested to work with a domain size of 648000 grid cells ($j_{pi} \times j_{pj} \times j_{pk}$).

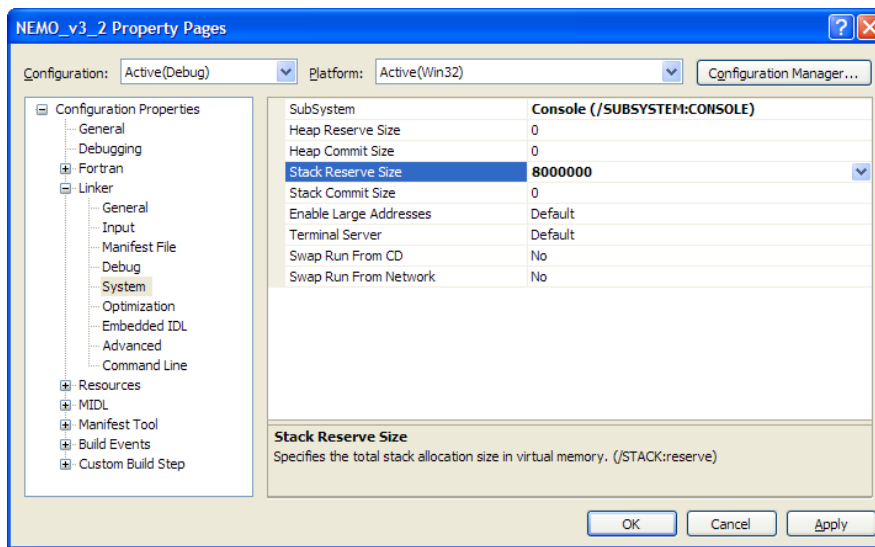


Figure D.32: Linker → System settings in the **NEMO_v3_2** project.

58. The model project **NEMO_v3_2** can now be compiled (Fig. D.33).

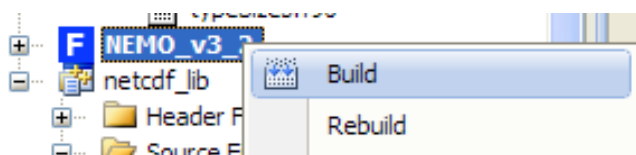
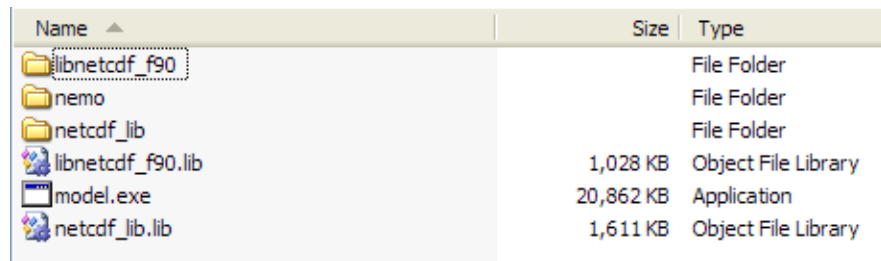


Figure D.33: Compiling the **NEMO_v3_2** project.

59. The compilation folder (e.g. 'Debug') will now contain the files and folders shown in Fig. D.34.



Name	Size	Type
libnetcdf_f90		File Folder
nemo		File Folder
netcdf_lib		File Folder
libnetcdf_f90.lib	1,028 KB	Object File Library
model.exe	20,862 KB	Application
netcdf_lib.lib	1,611 KB	Object File Library

Figure D.34: Files after compiling the **NEMO_v3_2** project.

60. Copy the file `model.exe` to the folder where the model is run (together with the namelist and the input netcdf files). The `*.lib` files are NOT required to run the model and should not be copied. All code that is required to run the model is linked into the `model.exe` file.

D.3.4 Distributing the precompiled executable

If the NEMO executable is compiled according to the above instructions, the final `model.exe` file should run on most Windows machines (unless processor-specific optimizations are used). It is useful to confirm that the executable will not require any non-standard DLLs which may or may not be installed on the target machine.

Open `model.exe` in Dependency Walker, which can be downloaded from <http://www.stevemiller.net/apps/> for both 32-bit and 64-bit Windows systems. The screenshot in Fig. D.35 confirms that `model.exe` only references system DLLs, which will be present on any Windows system.

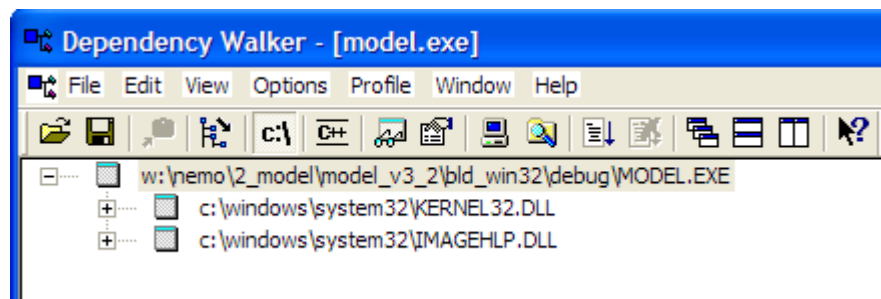


Figure D.35: Screenshot of Dependency Walker displaying details on the internals of the compiled `model.exe` executable.

D.4 Why do NEMO and NetCDF not compile ‘out of the box’?

The instructions above seem like a never-ending list of changes to the default compiler settings. In addition to that, one has to download suitable versions of various header files, which are either too old (e.g. `cfortran.h`) or not even available on a Windows system (e.g. `inttypes.h`).

While the mismatching include files can be excused by recent Visual Studio versions being incompatible with the rather old NetCDF code, the requirement to tune the various compiler and linker settings is more puzzling.

There appears to be a fundamental mismatch in how the default compiler and linker settings in Visual C (used for NetCDF library, i.e. the *called* code) and Intel Visual Fortran (used for NEMO, i.e. the *calling* code) handle the calling conventions in a mixed-language environment (i.e. mixing C/C++ with F77/F90 code).

The compiler settings listed above have the aim of creating object code from C-code (i.e. the NetCDF library) which is laid out in a way that it can be called from within Fortran code (i.e. NEMO plus its F90 bindings) and do 2 things right:

1. **link correctly.** The Fortran linker will fail if the C/C++ linker exports a function under the name `_NF_OPEN()` and the Fortran code attempts to call a function named `_nf_open@16()`. The naming of functions and how the internal function names in the object code are generated (so-called ‘name mangling’) is related to the calling conventions. The ‘Fortran → External Procedures’ setting ‘Name Case Interpretation → Upper Case (/names:uppercase)’ is partly responsible (together with the setting of the ‘Calling convention’) to make sure that the C++-linker will create the same function names that the Fortran linker is expecting to call (for more info on calling conventions, see point 2b below and Section D.4.1).

2. **run correctly.** The executable will fail at run time, if

- (a) there is no agreement between called and calling code on who is responsible for the stack handling. When a function is called (inside the library), the arguments and return values are pushed on the stack, which has to be cleaned up when the function call is finished. Different calling conventions will handle the stack unrolling differently, and the code will fail if e.g. the library expects the main code to do the stack handling, while the main code expects the library to do it. The execution will access improper memory and fail with an exception. For more info on calling conventions, see below.
- (b) there is no agreement on whether arguments are passed by value or by reference. The `cfortran.h` macros appear to create code which expects arguments to be passed by value, hence the requirement for the ‘Fortran → External Procedures’ setting ‘Calling convention → C, REFERENCE (/iface:ceref)’.
- (c) there is a mismatch in string argument passing. C and Fortran have very different concepts of how strings are handled. The code created by the `cfortran.h` macros will handle the conversion from Fortran-strings to C-strings and vice-versa, i.e. add the terminating ‘\0’ character for C-code and pad out with spaces for Fortran. However, when a function is called in Fortran, a string is always pushed onto the stack as a character pointer together with an integer denoting its length. This extra integer means that the function in the object code actually gains an extra argument for every string in its argument list that is visible in the code. So the C-code created by the `cfortran.h` macros when parsed and compiled by the Visual C/C++ compiler appears to create function calls where each string argument is immediately followed by the integer length value. This explains the re-

quirement for the ‘Fortran → External Procedures’ setting ‘String Length Argument Passing → After Individual String Argument (/iface:mixed_str_len_arg)’.

D.4.1 Background on calling conventions

There are two main classes of calling conventions `__cdecl` and `__stdcall`³.

- `__cdecl` is the ‘natural’ C calling convention; it supports the definition of vararg functions (like `printf`).
 - Arguments are passed by reference
 - Object code function names are uppercase
 - (on Win32) names are prefixed with an underscore (e.g. `_NF_OPEN`)
- `__stdcall` is the default calling convention for DLL functions, so it does not need to be specified, if those functions are only going to be called through their DLL API.
 - Arguments are passed by value
 - Object code function names are uppercase
 - (on Win32) names are prefixed with an underscore (e.g. `_NF_OPEN`)
 - (on Win32) names have suffix `@n`, where `n` is the number of bytes to be removed from the stack after the function call (e.g. `_NF_OPEN@16`)

The distinguishing feature of `__cdecl`, its support for vararg, is not required for compiling the NetCDF library interface, so either calling convention would work, although only `__cdecl` has been thoroughly tested for the compilation from static libraries as described.

³Based on info from <http://stackoverflow.com/questions/6334283/declspec-and-stdcall-vs-declspec-only>

There is more information on the `iface` options of the Intel Visual Fortran compiler on http://software.intel.com/sites/products/documentation/hpc/compilerpro/en-us/fortran/lin/compiler_f/copts/fortran_options/option_iface.htm

Side note: The alternative calling convention `__stdcall` would be useful if the NetCDF library was compiled as a DLL, which appears to be the default for compilation of the `cfortran.h` header under Visual Studio. This default in `cfortran.h` is overruled by the addition Preprocessor key `FCALLSC_QUALIFIER=__cdecl`. When a DLL is compiled, the exporting of the library function is also of importance and the setting `__declspec(dllexport)` needs to be turned on. This can be achieved by defining the preprocessor keys `DLL_NETCDF` and `NC_DLL_EXPORT`.

D.5 Compilation for 64-bit platforms

Notice how the Visual Studio toolbar lists not only the project configuration (e.g. Debug/Release), but also the type of target platform (e.g. Win32, Fig. D.36).

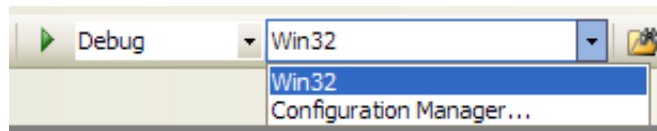


Figure D.36: By default only the 32-bit platform is available for compilation.

The platform for 64-bit Windows compilation is called `x64`, but this setting is not usually available as standard and needs to be added to the solution manually. This MSDN article [http://msdn.microsoft.com/en-us/library/9yb4317s\(v=VS.90\).aspx](http://msdn.microsoft.com/en-us/library/9yb4317s(v=VS.90).aspx) explains the procedure, which is repeated here with comments specific to NEMO.

Note : The 64-bit compilation is offered by the Professional edition of Visual Studio. The Visual Studio 2008 Express edition does not support the 64-bit platform as standard. Read the above-mentioned MSDN article and see these web pages for further

information:

- <http://jenshuebel.wordpress.com/2009/02/12/visual-c-2008-express-edition-and-64-bit-targets/>
- <http://blogs.msdn.com/b/windowssdk/archive/2009/06/15/installing-win-7-sdk-rc-and-vs2008-rtm-can-disable-vc-configuration-platform-choices.aspx>

D.5.1 Configure the projects for 64-bit compilation

1. Open the 'Configuration Manager', open the 'Active solution platform' drop down menu and select 'New...' (Fig. D.37)

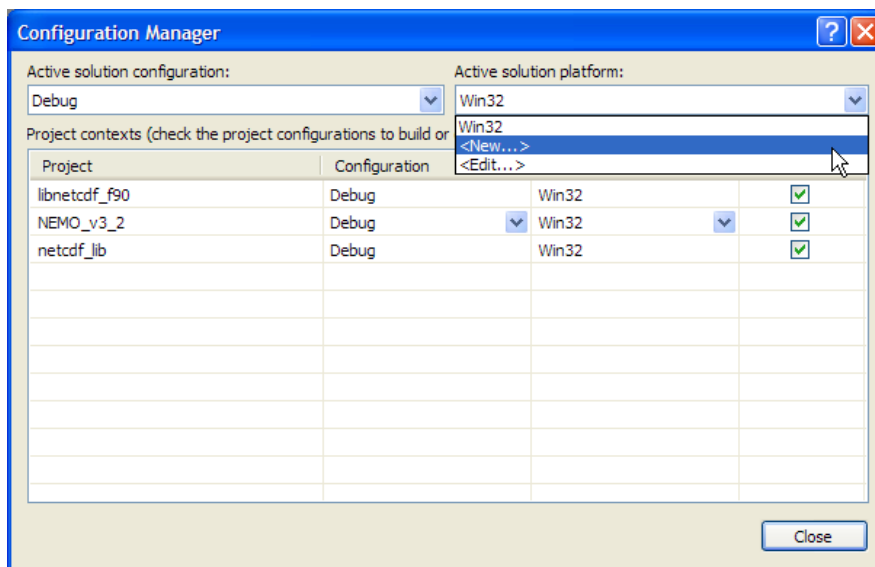


Figure D.37: Creating a new platform configuration.

2. A small dialog opens offering a choice of Platforms. Open the drop down menu and choose x64. Select 'Copy settings from' Win32 and check the 'Create new project platforms' box (Fig. D.38). Click OK.

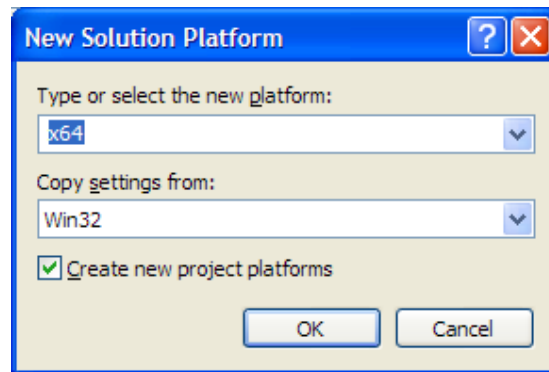


Figure D.38: Dialog for the creation of a new solution platform.

3. The solution has now gained 6 more project configurations (two new 64-bit Debug and Release configurations for each of the 3 projects).

When ‘copying’ the settings from Win32 to each x64 configuration Visual Studio has the nasty habit of resetting the Output Directory and Intermediate Directory of every project configuration. This needs to be rectified manually and their values in each Debug/Release configuration for the x64 platform can be copied from the values in the respective Win32 configuration.

4. Open the properties of each of the three projects, switch the platform to x64 and in the General section, set (for Debug and Release):

Output directory → `..\..\bld_$(PlatformName)\$(ConfigurationName)\`

and depending on the project, set

(for **netcdf_lib**) Intermediate Directory → `..\..\bld_$(PlatformName)\$(ConfigurationName)\netcdf_lib\`

(for **libnetcdf_f90**) Intermediate Directory → `..\..\bld_$(PlatformName)\$(ConfigurationName)\libnetcdf_f90\`

(for **NEMO_v3_2**) Intermediate Directory → `..\..\bld_$(PlatformName)\$(ConfigurationName)\nemo\`

5. The fundamental type sizes do not vary between 32-bit and 64-bit compilations (see Section D.5.2, and the derived type `off_t` is defined as a long on both 32-bit and 64-bit Windows platforms. Make sure the size assumption is correct in the `config.h` file, and if not already done make the following modification:

```
/* The number of bytes in a off_t */  
#define SIZEOF_OFF_T 4
```

6. The Preprocessor Definitions for the **netcdf_lib** project do not have to be adjusted despite the presence of the WIN32 definition. This definition is one of the defaults when the project is first created and can be left unchanged, even for building on a 64-bit platform.
7. The code can now be compiled using the new x64 platform type. The resulting executable can be run on a 64-bit Windows system where it is executed in native mode.

D.5.2 Background on sizes of fundamental types

How does Visual Studio handle the sizes of fundamental types when switching from 32-bit to 64-bit compilation? The only thing the C and C++ standards define as far as type sizes is concerned is ⁴ :

$$\text{sizeof char} \leq \text{sizeof short} \leq \text{sizeof int} \leq \text{sizeof long}$$

Win64 defines `int` and `long` to be 32-bit. Presumably for compatibility reasons as much as anything else. Be aware that `size_t` is 64-bit though.

Table D.1 lists the amount of storage required for fundamental types in Microsoft C++ as implemented in Visual Studio 2008 ⁵. This information shows that the type

⁴See <http://www.velocityreviews.com/forums/t492580-sizeof-int-on-x64-system.html>

⁵See [http://msdn.microsoft.com/en-us/library/cc953fe1\(v=VS.90\).aspx](http://msdn.microsoft.com/en-us/library/cc953fe1(v=VS.90).aspx)

sizes are not dependant on a 32-bit or 64-bit compilation, and no further modifications need to be made to the code to adapt it to a 64-bit environment.

Table D.1: Storage size (in bytes) of the fundamental data types in MS Visual Studio 2008

Type	Size
bool	1 byte
char, unsigned char, signed char	1 byte
short, unsigned short	2 bytes
int, unsigned int	4 bytes
long, unsigned long	4 bytes
float	4 bytes
double	8 bytes

D.5.3 Note on platform-specific preprocessor switches

WIN32 is a name that you could use in your own code and so might clash with Microsoft's usage. `_WIN32` is a name that is reserved for the implementor (in this case MS) because it begins with an underscore and an uppercase letter - you are not allowed to use such names in your own code, so there can be no clash ⁶. This means that a preprocessor switch `WIN64` can be defined for any code which is only to be executed in a 64-bit environment.

D.6 NetCDF file handling on Windows

D.6.1 Operating system support for large files

Model output files may become very large, and may easily reach many gigabytes in size. The support for large files (> 2GB) is dependant on the operating system. The file system which supports large files under Windows is called NTFS and it is the default when a disk is formatted. These days, only some memory cards for digital cameras or MP3 players will still use the older FAT32 file system where file sizes are limited to a maximum of 2GB.

⁶See <http://stackoverflow.com/questions/662084/whats-the-difference-between-the-win32-and-win32-defines-in-c>

The working folder where the model is run should be located on a disk formatted with NTFS. One can quickly check that this is the case: Right click on the drive letter in Windows Explorer and bring up the properties of the drive. The property page will show the type of file system used (see screenshot in Fig. D.39).

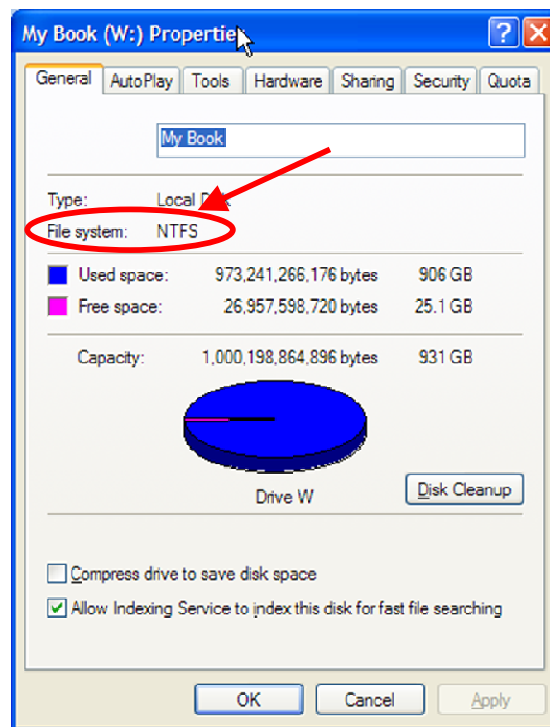


Figure D.39: Properties of a Windows disk showing its file system (circled).

D.6.2 NetCDF file formats

There are several different NetCDF files formats (see <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf.html#File-Format>). The NetCDF v3.6.3 library offers a choice of these two formats:

- Classic file format
- 64-bit Offset Format for large files

The newer 64-bit Offset Format supports large files and large records (e.g. for arrays) with the file. This file format is not much different from the classic format - only a few

bytes in the file header are different. It makes obvious sense to use always use the 64-bit Offset Format in case a record grows too large for the classic format to handle and NEMO does in fact create all files in this format by default. However, there are some programs which cannot read the 64-bit Offset Format, and only handle the classic format. One such format-limited program is `ncBrowse`; see next section.

D.6.3 File format issues with `ncBrowse`

While the excellent `Ncview` is available on UNIX-like systems⁷, the only software to quickly inspect and visualise the contents on NetCDF files on Windows is `ncBrowse` (downloadable from <http://www.epic.noaa.gov/java/ncBrowse/>).

`ncBrowse` does not (yet?) support the 64-bit Offset Format and will not be able to open such files. It therefore makes sense to switch NEMO back to the classic format during the initial phase of a modelling project where needs to frequently inspect the NetCDF files to check the model is working.

There is an easy way to prevent NEMO from creating files in the 64-bit Offset Format:

1. Open the NEMO solution.
2. In the project `libnetcdf_f90`, open the file `netcdf_constants.f90`.
3. Search the file for `nf90_64bit_offset`. The editor should jump to line 49.
4. Change the line reading

```
nf90_64bit_offset = 512,      &  
- to -  
nf90_64bit_offset = 0,      &
```

⁷The `Ncview` home page is at http://meteora.ucsd.edu/~pierce/ncview_home_page.html

5. This has the simple effect that every time the NEMO ‘thinks’ is asking for a file to be created in the 64-bit Offset Format (with format code 512) it actually creates it using the Classic Format (format code 0).
6. Recompile the model and run.

The output files created by the model compiled in this way can now be opened and visualised using ncBrowse. After testing is finished and short model runs have been confirmed to work fine, the actual production runs are likely to be much longer (more time steps) and are expected to create massive amounts of data in large records. At that stage of the project, **it is recommended to reverse this modification!** Post-processing of the files must then be done with different software, which is capable of reading the 64-bit Offset Format, such as MATLAB.

Appendix E

Guide to setting up a regional NEMO model on the HPC cluster



background image shows the University of Plymouth High Performance Computing cluster (from www.plymouth.ac.uk).

This appendix is a rough guide on how to set up a regional NEMO model on the University of Plymouth High Performance Computing (HPC) cluster. The instructions cover aspects of configuring the model used for the Svalbard experiments in Chapter 5. The documentation of setup on the HPC cluster should be transferable in large parts to any other multi-processor cluster running a variant of Linux/UNIX. Assistance by staff at NOC Liverpool is kindly acknowledged.

The following tasks were performed on a Windows computer:

- access the cluster via remote login shell
- transfer files to and from the cluster
- pre-processing, generating of model input files
- post-processing and data analysis of model output files

This leaves the following tasks to be performed on the cluster itself:

- compilation of the model code
- running the model
- collating of individual output files (i.e. part-domain file per processor) into combined (i.e. whole-domain) output files

E.1 Windows software installation

This section describes one of many possible ways to configure software to access the cluster (command-line shell) and to transfer files between the Windows computer and the cluster. Many different types of software for this task exist and only the author's personal choice is given here.

E.1.1 Cygwin - XTerm

Cygwin provides a Unix-like environment on Windows computers. It comes with a standard command-line interface (shell) and also provides the X Window System (X11). This section describes how to install and run `xterm` on a Windows desktop, so it can be used to access the HPC cluster. The X11 tool `xterm` is the standard terminal emulator in X11. It provides an easy-to-use command-line shell environment with superior copy-paste functionality compared to other shells.

1. Install Cygwin-X according to the instructions on <http://x.cygwin.com/docs/ug/setup-cygwin-x-installing.html>
2. Select the following packages for installation (as listed by the web page, plus some additional suggestions)
 - xorg-server
 - xinit
 - xorg-docs
 - X-start-menu-icons
 - openssh
 - xterm
3. The installation will create a desktop shortcut called 'Cygwin Terminal'. This is the standard shell, which can be used to access the cluster, but `xterm` is preferred.
4. Create a desktop shortcut named 'XWin Server' with the following target:

```
C:\cygwin\bin\run.exe /usr/bin/bash.exe -l -c /usr/bin/startxwin.exe
```

5. Double-click the shortcut to start the X server. Every time the Windows computer is restarted, this has to be done once again before `xterm` can be used. The X11 startup scripts will also start a `xterm` shell window, which is logged on the local Windows machine (Fig. E.1). This local shell is not very useful for the purpose of this guide and can be closed. It can be reopened at any by right-clicking the Cygwin/X system tray icon (Fig. E.1) and selecting `Applications → xterm`.

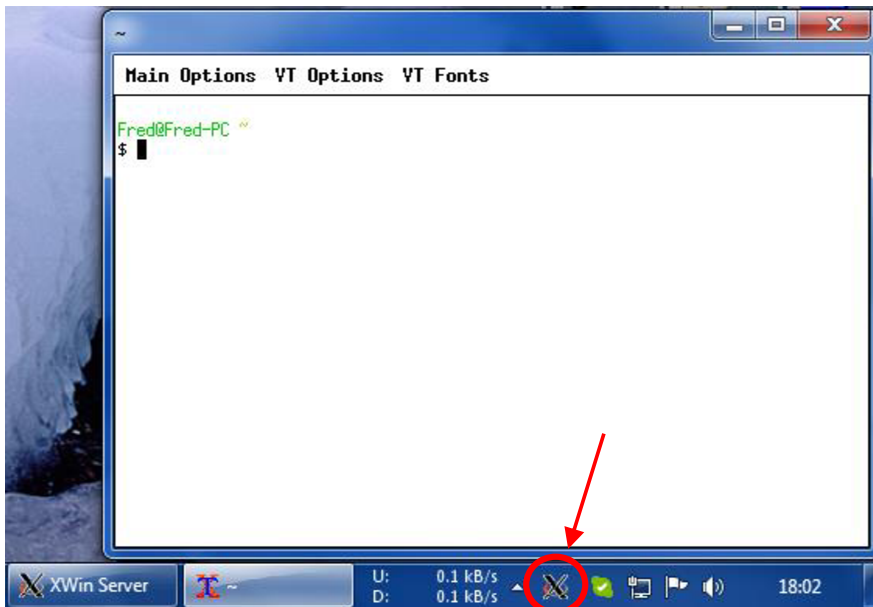


Figure E.1: The X Window server has started and opened a local `xterm` window. The system tray icon of the Cygwin/X server is pointed out in red.

6. On the Windows desktop, create another shortcut called 'FOTcluster2 xterm' (Fig. E.2) with the following target:

```
C:\cygwin\bin\run.exe -p /usr/X11R6/bin xterm -display  
127.0.0.1:0.0 +ls -sb +tb -sl 500 -geometry 133x74+0+0  
-e /usr/bin/ssh -Y username@hostname
```
7. Replace `username` with the name of your user account, and replace `hostname` with the network name or IP-address of the HPC cluster. This shortcut will open an `xterm` shell logged in to the cluster (after password entry). The `-Y` option is

useful as it sets up the shell to tunnel X Window connections from X11 programs (run e.g. `xclock` to test this).



Figure E.2: Desktop shortcuts after successful completion of installation steps in Section E.1.1.

8. The tutorial at http://www.linuxproblem.org/art_9.html explains how to configure a ‘mutual trust’ between the Windows and cluster machine to enable login without entering a password. This is optional. Further details are given on the web page.

E.1.2 Bitwise SSH Client

The software ‘Bitwise SSH Client’ (formerly known as Tunnelier) is a very advanced graphical SFTP client (Fig. E.3) used to transfer files over secure SSH connections. The software also includes a shell terminal, but the **xterm** (described in the previous section) is considered to be much more user-friendly (especially when it comes to copy-paste commands).

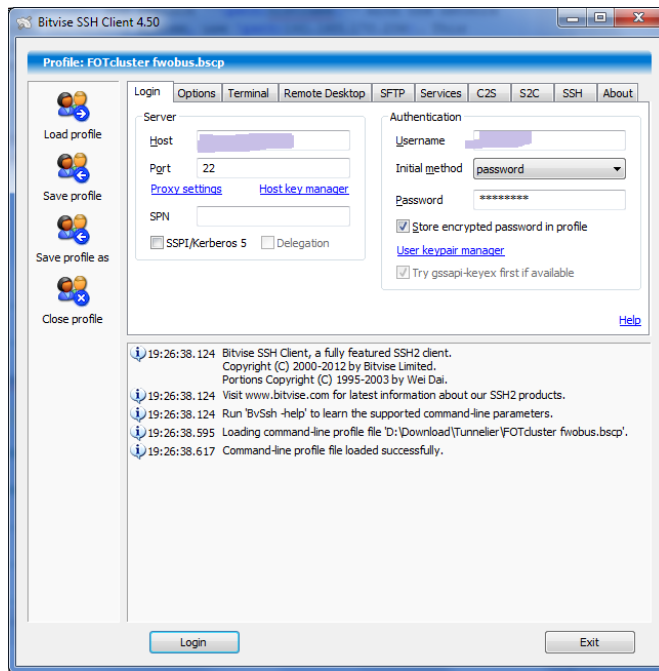


Figure E.3: Login dialog of the Bitvise SSH Client.

1. The Bitvise SSH Client can be downloaded from <http://www.bitvise.com/tunnelier>. It is free for personal use.
2. Enter the following settings (adjust IP address if needed):
 - Login → Server → **Host** <the IP-address>
 - Login → Server → **Port** 20
 - Login → Authentication → **Username** <your username>
 - Login → Authentication → **Initial method** password
 - Login → Authentication → **Password** <your password>
 - Options → **On Login** check Open SFTP
3. Configure the SFTP panel as shown in Fig. E.4.
4. Click 'Login'. A split screen window for SFTP transfer will open (Fig. E.5). Use this window to transfer files from the local Windows machine (left hand-side) to

E.2. INSTALLATION ON THE CLUSTER

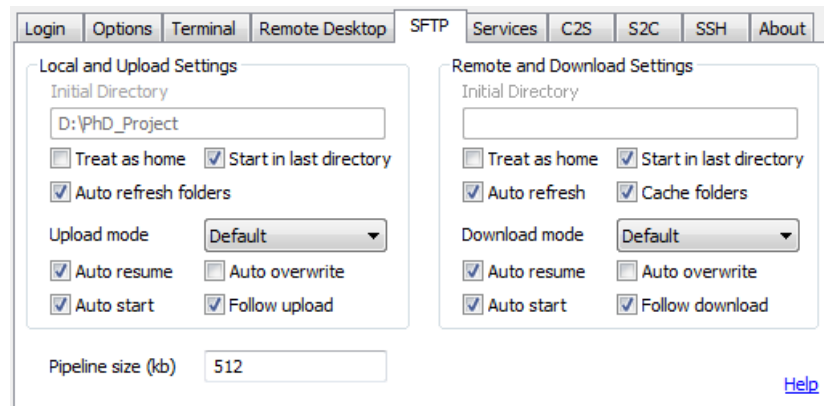


Figure E.4: Recommended SFTP settings in the Bitvise SSH Client.

the cluster (right hand-side). Entire folder structures can be copied.

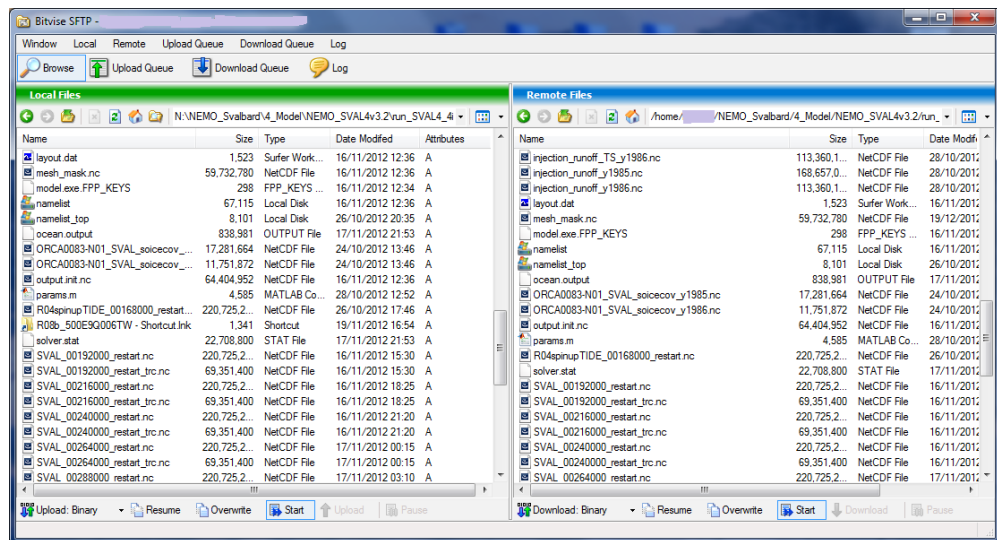


Figure E.5: Split screen SFTP window in the Bitvise SSH Client.

E.2 Installation on the cluster

E.2.1 System prerequisites

The UNIX system should have these components already installed:

- **Perl** The system used by the author came with v5.8.8, but any similar version should be fine.
- **ifort** The Intel Fortran compiler. Version 11.1 was used here.

- **MPI** Should come pre-installed as part of the compiler. Otherwise OpenMPI can be downloaded free of charge.
- **netcdf** The NetCDF libraries are required by NEMO to link. Version 4.0.1 was used here, but any version higher than 4 should work. If not already present on the system, the netcdf libraries can be downloaded from <http://www.unidata.ucar.edu/downloads/netcdf/index.jsp>.

NOTE: from version 4.2 onwards, the installation requires two parts:

1. netCDF C library
2. netCDF Fortran 77 and Fortran 90 libraries

Both of these packages need to be installed according to the instructions contained in the download packages. This process is fairly straightforward on a Linux/UNIX system and is therefore not further explained here.

E.2.2 Install fcm

The build system used here for compiling NEMO is **fcm**, developed by the MetOffice. It can be downloaded from their website at <http://www.metoffice.gov.uk/research/collaboration/fcm>. **Fcm** provides many advanced feature for complex Fortran software packages, but only a minimal set of functions – the build system – is used for compiling NEMO. Version 2.2 of **fcm** was used here, but later versions 2.3 and upwards might also work.

To install **fcm**, simply unpack the downloaded *.tgz file to a directory in the user's home directory (e.g. ~/fcm-2-2/). Open ~/.bashrc in a text editor and add the following line:

```
export PATH=$PATH:~/fcm-2-2/bin/
```

E.2.3 Additional Perl modules

Fcm is based on Perl and requires several additional Perl modules to be installed on the system. If the user has administrative right (root access) to the system, these extra modules can easily be installed system-wide. If not, they have to be installed in the user's home directory. This means that this custom installation of the modules has to be repeated for every new NEMO user, hence system-wide installation is preferred.

Additional Perl modules for **fcm v2.2** are:

- `HTTP::Date`
- `XML::Parser`

The instruction on <http://linuxgazette.net/139/okopnik.html> were the basis for the following guide.

1. Perl needs to be told where the extra modules are going to be located, as they will not be in the standard location at `/usr/`. In a text editor, open the file `~/.bashrc` and add the following lines (at the end, if unsure where):

```
if [ -z "$PERL5LIB" ]
then
    # If PERL5LIB wasn't previously defined, set it...
    PERL5LIB=~/.myperl/lib
else
    # ...otherwise, extend it.
    PERL5LIB=$PERL5LIB:~/.myperl/lib
fi

MANPATH=$MANPATH:~/.myperl/man

export PERL5LIB MANPATH
```

2. Now, create the necessary directories:

```
mkdir -p ~/myperl/lib
mkdir -p ~/myperl/man/man{1,3}
```

3. Log out and log back in again. Perl will now treat that location as a part of @INC (the list of directories to search for libraries and modules). The following command confirms this:

```
perl -wle 'print for grep /myperl/, @INC'
```

4. Prepare the Perl installation by configuring the CPAN module with the shell command:

```
perl -MCPAN -we 'shell'
```

Some people have referred to it as a ‘pecking chicken’ install (that is, if you set a chicken over the ‘Enter’ key and it then keeps pecking at the key, it’ll get through the installation successfully). Two small **modifications to the standard procedure** are needed: when the script asks you for any extra arguments for `Makefile.PL`, you need to supply it with the following list (assuming that you chose `~/myperl` as your private lib directory):

```
LIB=~/myperl/lib INSTALLSITEMAN1DIR=~/myperl/man/man1
INSTALLSITEMAN3DIR=~/myperl/man/man3
```

You also need to make sure that the `UNINST` parameter is turned off; you can do this by setting

```
UNINST=0
```

when that question comes up during the installation. (This is the default behavior unless you set it, but you might as well make sure.)

If the CPAN shell had already been configured at some point in the past, it is sufficient to simply modify the existing configuration. Again, start the shell using `perl -MCPAN -we 'shell'` and issue the following commands at the `cpan>` prompt:

E.2. INSTALLATION ON THE CLUSTER

```
o conf makepl_arg "LIB=~ /myperl /lib_␣↵
  ↳ INSTALLSITE MAN1DIR=~ /myperl /man /man1_␣↵
  ↳ INSTALLSITE MAN3DIR=~ /myperl /man /man3 "
o conf make_install_arg UNINST=0
o conf commit
```

5. Now install the additional modules for **fc** by typing these two commands:

```
perl -MCPAN -we 'install "HTTP::Date"'
perl -MCPAN -we 'install "XML::Parser"'
```

6. The latter of these may take a little longer to execute. The installation of the XML parser involves lots of compilation and configuration, but it should all work automatically.

E.2.4 User's `bashrc` configuration

Further to the two additions to the `~/.bashrc` file given above (adding **fc** to the `PATH` and configuring the locations of Perl modules), add the following lines:

```
export MPI_HOME=/opt/ofed/1.5.2/mpi/intel/openmpi-1.4.2/

# allow core files
ulimit -c unlimited

# stack size unlimited
ulimit -s unlimited

# User specific environment and startup programs
export PATH=$PATH:$HOME/bin
```

The definition of `MPI_HOME` is system dependent and may already be provided by the system-wide shell configuration (check with `echo $MPI_HOME`). The same goes for the last line which adds the user's own `~/bin/` directory to the path. This setting may already exist (check with `echo $PATH`).

The NEMO code can now be compiled, see Section E.5.1.

E.2.5 Compiling **nocscombine**

On a multi-processor system NEMO creates individual output files per processor. Each output file only contains the output fields for the domain tile ‘seen’ by that particular processor. At the end of each run the tiled output needs to be collated into combined fields that cover the entire model domain.

A shell script `combine_outputs` was developed for this task (Section E.7.2). The script requires the program **nocscombine** which does the heavy lifting of netCDF data processing. For ease of use, this program should be installed in the user’s path (copied to the `~/bin/` directory).

The source code for **nocscombine** can be downloaded from <http://www.mmnt.net/db/0/0/ftp.soc.soton.ac.uk/omfftp/NEMO>. The compilation will have to be adapted to the given operating system and compiler. This requires knowledge of compiling code, e.g. how to write a Makefile. The following guide is just a quick summary of the steps that were taken to compile **nocscombine**.

1. Unpack the tar archive. The source files are in fixed format ¹, which doesn’t agree well with modern compilers.
2. The tool **freeform.pl** (a perl script from <http://marine.rutgers.edu/po/tools/perl/freeform>) was used to convert the source files to F90 ‘free format’. The file endings of all source files were changed to `*.f90`.
3. The following **Makefile** was used for compiling **nocscombine** with Intel’s **ifort**. The location of the netCDF libraries (and possibly some compiler flags) may need to be adjusted. Note, that if netCDF version 4.2 or greater is used, the linker

¹See http://www.amath.unc.edu/sysadmin/DOC4.0/fortran/user_guide/C_f90.doc.html for more information on Fortran’s code formats.

E.2. INSTALLATION ON THE CLUSTER

will require 2 libraries (containing the C and Fortran code in separate libraries: ‘-lnetcdf -lnetcdf’).

```
NCHOME = /usr/local/netcdf4.0.1
FC      = ifort
FFLAGS= -fpp -O3 -traceback -DLARGE_FILE -I$(NCHOME)/include
LFLAGS= -L$(NCHOME)/lib
LIBS=    -lnetcdf

.SUFFIXES:
.SUFFIXES: .f90 .o

OBJFILES = nocscombine.o ncfixcoord.o rtime.o make_global_file.o ↵
           ↵ ncread_and_collate.o handle_err.o

nocscombine : $(OBJFILES)
               $(FC) $(LFLAGS) -o $@ $(OBJFILES) $(LIBS)

.f90.o:
           $(FC) $(FFLAGS) -c $<

clean :
           rm $(OBJFILES)
```

4. The command `make nocscombine` compiles the program.
5. The resulting executable is then copied to `~/bin/`

E.2.6 nocs_weights (nocsSCRIP)

The **nocs_weights** software is used to generate and manipulate interpolation weights for use with the Interpolation On the Fly (IOF) option in NEMO v3.x. These utilities rely heavily on the Spherical Coordinate Remapping and Interpolation Package (SCRIP²). **nocs_weights** was written by the NEMO team at NOCS and can be downloaded from `ftp://ftp.soc.soton.ac.uk/msmftp/NEMO/`. The version used here

²SCRIP available from `http://climate.lanl.gov/Software/SCRIP/`

was adapted by M. Luneva (NOCL) and then further modified to be compiled under Windows for the use of this study.

Compilation of the the individual components of **nocs_weights** creates three executables **scrip**, **scripgrid** and **scripshape** (and the optional **scripinterp**). The Visual Studio 2008 solution defines one project file per executable. The solution also includes the nextcdf-3.6.3 C-library **netcdf_lib**, **libnetcdf_f90** Fortran library and the **libscrip** library, which is part of the **nocs_weights** package. The directory structure is set up in the following way.

```
NOCS_WEIGHTS
+---bin
+---data
|   +---SVAL2_160x180
+---libscrip
+---utils
\---win32
    +---bld_Win32
    |   \---Release
    |       +---libnetcdf_f90
    |       +---libscrip
    |       +---netcdf_lib
    |       +---scrip
    |       +---scripgrid
    |       \---scripshape
+---netcdf-3.6.3
|   +---cxx
|   +---cxx4
|   +---examples
|   |   +---C
|   |   +---CDL
|   |   +---CXX
|   |   +---CXX4
|   |   +---F77
|   |   \---F90
|   +---f90
|   +---fortran
```

E.2. INSTALLATION ON THE CLUSTER

```
|  +---libsrc
|  +---libsrc4
|  +---man
|  +---man4
|  +---ncdump
|  +---ncgen
|  +---nc_test
|  +---nc_test4
|  +---nf_test
|  \---win32
|      \---NET
|          +---examples
|          +---libsrc
|          +---ncdump
|          +---ncgen
|          +---nc_test
|          \---nc_test
\---sln
```

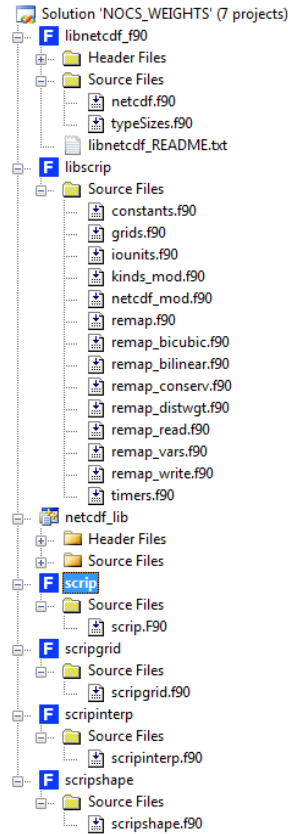


Figure E.6: Visual Studio 2008 solution for compilation of the **nocs_weights** components.

Detailed instructions on how to compile a program against the netcdf-3.6.3 libraries under Visual Studio are given in Appendix D. While those instructions describe how to set up the **NEMO_V3_2** project and compile the executable `model.exe` from the NEMO sources, the same setup is used for the three projects that compile the executables `scrip.exe`, `scripgrid.exe` and `scripshape.exe`. A screenshot of the fully configured Visual Studio solution is given in Fig. E.6.

After compilation, the three executables are copied to `NOCS_WEIGHTS\bin`. The usage of these programs to create the weights files is described in Section E.3.4.

E.3 Pre-processing

The following section describes the setup of the regional model of Svalbard for the experiments in Chapter 5.

E.3.1 Coordinates (horizontal grid)

Creating the horizontal grid is often the first step in defining the model domain. The `coordinate.nc` file defines (amongst other things like horizontal scaling factors) the latitude/longitude position of every single grid point in the model.

For Chapter 5 the grid was defined to have a horizontal step of exactly 3 km in both directions (rather than an even fraction of degrees). The script `R0_Create_Domain.m` calls the following function:

```
[ glamt,gphit, glamu,gphiu, glamv,gphiv, glamf,gphif ] = ↵  
    ↵ calc_coordinates( SWlon,SWlat, jpi,jpj, dimxm,dimym );
```

to calculate the coordinate arrays for T-points (`glamt,gphit`), U-points (`glamu,gphiu`), V-points (`glamv,gphiv`) and F-points (`glamf,gphif`). The calculation is based on the coordinates of the South-Western corner of the domain (`SWlon,SWlat` - in degrees), the number of grid points in the longitudinal/latitudinal direction (`jpi,jpj`), and the horizontal step size in those directions (`dimxm,dimym` - in metres).

E.3.2 Bathymetry

This section expands on Section 5.2.1 which only briefly describes the preparation of the Svalbard bathymetry. The processing starts with the file `IBCAO_V3_30arcsec_RR.asc` downloaded from http://www.ngdc.noaa.gov/mgg/bathymetry/arctic/grids/version3_0/. This file contains, in ASCII format, the bathymetry data for the entire Arctic Ocean. The script `IBCAOV3_region_to_grd.m` crops a partial area and writes a Surfer GRD-file. Limiting the area to $0 \rightarrow 30^\circ\text{W}$, $72 \rightarrow 82^\circ\text{N}$ is primarily concerned with the reduction of the amount of data that the subsequent step

has to deal with. The resulting file is `IBCAO_V3_30arcsec_RR_SVALBARD.grd`.

When the script `R0_Create_Domain.m` is run, it calls the following function

```
raw_bathy = load_and_interpolate_bathy( bathy_grd_file, glamt,gphit, ↵  
    ↵ rn_sbot_max );
```

where `bathy_grd_file` is set to `IBCAO_V3_30arcsec_RR_SVALBARD.grd` and `glamt,gphit` are the T-point coordinates created by the `calc_coordinates()` function. The last parameter `rn_sbot_max` is the maximum depth as defined in the `namelist` file.

The function `load_and_interpolate_bathy()` loads the data from the GRD file and interpolates it onto the model grid (`glamt,gphit`) using Matlab's built-in `interp2` function. The depths are then sign-inverted and capped to a maximum of `rn_sbot_max` (in this case 2500 m). The next step runs the function

```
[bathy, stats]=calc_slope_parameter(bathy, bathy_adjust_crit, 0, gphit);
```

to apply the slope parameter correction of Martinho and Batteen (2006) to adjust the bathymetry in such a way that the slope parameter *SP* does not exceed `bathy_adjust_crit = 0.3`. No volume conservation is applied during this adjustment. The modified bathymetry differs from the original on the Western Svalbard slope, where the slope angle is particularly steep ($> 4^\circ$ in places).

The bathymetry is then smoothed with a 2-D convolution filter using a 3×3 Gaussian kernel.

```
kernel = fspecial('gaussian',[3 3], 1);  
bathy_smooth = conv2(bathy, kernel, 'same');  
  
% apply multiplier mask, = 1 at depth>rn_sbot_max/20 tending towards ↵  
    ↵ 0 in the shallows  
frac=min(1.0,bathy ./ (rn_sbot_max/20));  
bathy = frac .* bathy_smooth + (1-frac) .* bathy;
```


The smoothed data is applied using a linear mask (`frac`) which fades out the smoothed bathymetry to the original bathymetry in a gradual way from 125 m ($2500\text{m}/20$) below the surface (full smoothing is applied below this depth) upwards to the surface (where no smoothing is applied). This gradation of the smoothing function preserves the shapes of islands and headlands.

The bathymetry is further adjusted by closing the Freemansundet (between Barentsøya and Edgeøya) and the northern parts of the Storfjorden where the bathymetry along a line between 77.205414°N , 17.429278°E and 20.92982°N , 77.43625°E is set to land.

Next, very shallow parts of the bathymetry are adjusted as follows: Every point shallower than 1.5 m is set to land, while any remaining point shallower than 3 m is set to 3 m. The minimum depth is imposed to prevent very thin vertical cells which tend to violate the vertical Courant-Friedrichs-Lewy (CFL, Courant et al., 1928) condition.

Finally any isolated grid locations (e.g. fjords that no longer have a connection to the open sea after interpolation and grid points north of the line which closes the Storfjorden) are identified according to the following scheme. The algorithm first creates a binary mask `bathy_img` which is 0 in wet points (where `bathy(i, j) > 0`) and 1 at dry points. Then, the location `bathy_max_i`, `bathy_max_j` of the deepest bathymetry is found (i.e. any *i/j* location of a point at 2500 m depth in our case of capped bathymetry). Next, the `imfill` function (available in Matlab's Image Processing Toolbox) performs a flood-fill starting at the deepest point to create a mask of only those points directly connected to that point. Thus, any grid points that cannot be reached from the central deep location (via water, i.e. wet points) is blanked (depth set to 0).

```
bathy_img = imfill(logical(bathy_img), [bathy_max_i bathy_max_j]);  
idx=find(bathy_img==0);  
bathy(idx)=0;
```

The final bathymetry is shown in Fig. E.7.

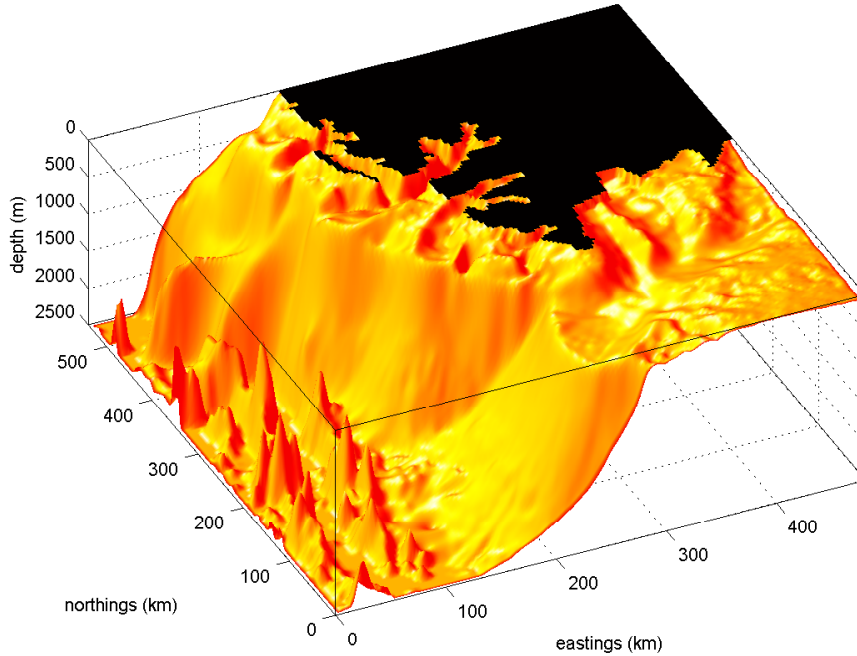


Figure E.7: Final model bathymetry after processing the raw IBCAO data. Note, that any isolated areas of water, which aren't connected to the main model domain west and south of Svalbard, are blanked and set to land (black areas).

E.3.3 Atmospheric forcing files

The atmospheric forcing for the Svalbard experiments in Chapter 5 was the Drakkar Forcing Set DFS4.1 (Brodeau et al., 2010). This data set was downloaded from the National Oceanography Centre's ORCA1 project website at `ftp://ftp.noc.soton.ac.uk/omfftp/DFS4.1`. The data set contains 8 different parameters which are provided on a global grid of 2 different resolutions (Table E.1).

Table E.1: Types of data fields in the Drakkar Forcing Set (DFS4.1).

Data type	Resolution	Description
lwdn	192×94	Downwelling long wave radiation
swdn	192×94	Downwelling short wave radiation
precip	192×94	Precipitation
snow	192×94	Snowfall
q2m	320×161	Specific humidity at 2 m
t2m	320×161	Air temperature at 2 m
u10	320×161	Zonal wind at 10 m
v10	320×161	Meridional wind at 10 m

NEMO allows several ways of splitting large data sets into multiple files - per year, per day or per month. In the following example, there is one file per variable, per year. The files (shown here for the year 1986) have been renamed from their original DFS file name to fit the NEMO naming convention:

- 1d_DFS4.1_lwdn_y1986.nc
- 1d_DFS4.1_swdn_y1986.nc
- 1m_DFS4.1_precip_y1986.nc
- 1m_DFS4.1_snow_y1986.nc
- 6hr_DFS4.1_q2m_y1986.nc
- 6hr_DFS4.1_t2m_y1986.nc
- 6hr_DFS4.1_u10_y1986.nc
- 6hr_DFS4.1_v10_y1986.nc

By using NEMO's Interpolation on the Fly (IOF) option, the files can be used in their original form and need not be processed to fit the Svalbard model grid. This run-time interpolation is configured by weights files, as described in the next section.

E.3.4 Weights files

The data in the Drakkar Forcing Set 4.1 is given on two global grids with resolutions of 192×94 (swdn, lwdn, precip and snow) and 320×161 (t2, q2, u10 and v10) as shown in Table E.1. The components **scrip**, **scripgrid** and **scripshape** (see Section E.2.6 for compilation instructions) of the **nocs_weights** package create the weights files for NEMO's Interpolation on the Fly (IOF) module. The weights files are used to interpolate the 2-D forcing data from the data files' source grid to the model domain's target grid.

The commands to prepare the Svalbard weights files were run in the directory `NOCS_WEIGHTS\data\SVAL2_160x180`. The latter part of the directory name refers to the target resolution of 160×180 grid points used in the Svalbard model.

Each of the two resolutions in the source data files requires a weights file. To create a weight files, the three executable are run in sequence: **scripgrid** → **scrip** → **scripshape**. This sequence is run twice, once for each resolution. The executables are controlled by a combination of namelist files and command line parameters. The command line parameters can be recorded in Windows BAT-files, which are given below.

The batch file to create the weights file `DFS_192x94_to_160x180_weights.nc` for the 192×94 resolution (swdn, lwdn, precip and snow):

```
..\..\bin\scripgrid namelist_swdn_lwdn_precip_snow
..\..\bin\scrip namelist_swdn_lwdn_precip_snow
..\..\bin\scripshape data_nemo_bilin_192x94.nc ^
    ↪ DFS_192x94_to_160x180_weights.nc
```

Below is the namelist for running **scripgrid** and **script** for the 192×94 resolution. The namelist parameter `input_file` refers to the file `1985_1d_DFS4.1_swdn.nc` from the DFS which contains the coordinates of the source grid.

```
!-----
&grid_inputs      ! parameters for "scripgrid"
!-----
    input_file='..\..\1985_1d_DFS4.1_swdn.nc'
    nemo_file = 'coordinates.nc'
    datagrid_file = 'remap_data_grid_192x94.nc'
    nemogrid_file = 'remap_nemo_grid_192x94.nc'
```

E.3. PRE-PROCESSING

```
method = 'regular'
input_lon = 'lon'
input_lat = 'lat'
nemo_lon = 'glamt'
nemo_lat = 'gphit'
nemo_mask = 'none'
nemo_mask_value = 10
input_mask = 'none'
input_mask_value = 10

/
!-----
&remap_inputs                                ! parameters for "scrip"
!-----
num_maps = 1
grid1_file = 'remap_data_grid_192x94.nc'
grid2_file = 'remap_nemo_grid_192x94.nc'
interp_file1 = 'data_nemo_bilin_192x94.nc'
interp_file2 = 'nemo_data_bilin_192x94.nc'
map1_name = 'data_to_nemo_bilinear_Mapping'
map2_name = 'nemo_to_data_bilinear_Mapping'
map_method = 'bilinear'
normalize_opt = 'frac'
output_opt = 'scrip'
restrict_type = 'latlon'
num_srch_bins = 90
luse_grid1_area = .false.
luse_grid2_area = .false.

/
```

Batch file to create the weights file `DFS_320x161_to_160x180_weights.nc` for the 320×161 resolution (t2, q2, u10 and v10):

```
..\..\bin\scripgrid namelist_t2_q2_u10_v10
..\..\bin\scrip namelist_t2_q2_u10_v10
..\..\bin\scripshape data_nemo_bilin_320x161.nc ↵
    ↵ DFS_320x161_to_160x180_weights.nc
```

The namelist for running **scripgrid** and **scrip** for the 320×161 resolution:

```
!-----
&grid_inputs                                ! parameters for "scripgrid"
!-----
input_file='..\..\1985\1985_6hr_DFS4.1_t2m.nc'
nemo_file = 'coordinates.nc'
datagrid_file = 'remap_data_grid_320x161.nc'
nemogrid_file = 'remap_nemo_grid_320x161.nc'
method = 'regular'
input_lon = 'lon'
input_lat = 'lat'
nemo_lon = 'glamt'
nemo_lat = 'gphit'
nemo_mask = 'none'
```

E.3. PRE-PROCESSING

```
nemo_mask_value = 10
input_mask = 'none'
input_mask_value = 10
/
!-----
&remap_inputs           ! parameters for "scrip"
!-----
num_maps = 1
grid1_file = 'remap_data_grid_320x161.nc'
grid2_file = 'remap_nemo_grid_320x161.nc'
interp_file1 = 'data_nemo_bilin_320x161.nc'
interp_file2 = 'nemo_data_bilin_320x161.nc'
map1_name = 'data_to_nemo_bilinear_Mapping'
map2_name = 'nemo_to_data_bilinear_Mapping'
map_method = 'bilinear'
normalize_opt = 'frac'
output_opt = 'scrip'
restrict_type = 'latlon'
num_srch_bins = 90
luse_grid1_area = .false.
luse_grid2_area = .false.
/
```

The result of the two batch scripts are two weights files:

- DFS_192x94_to_160x180_weights.nc
- DFS_320x161_to_160x180_weights.nc

These files will be referred to by the NEMO namelist to configure the CORE bulk formula forcing (see Section E.6.1).

E.3.5 Rivers (dense water injection)

Skogseth et al. (2004) modelled the ice formation during 4 winters (1998-2001) and found that between 893 and 1133×10^9 kg of salt was released to the water column. This produced 0.9 to 1.1×10^9 m³ or 0.06 to 0.07 Sv (freezing period average) of brine-enriched shelf water with salinities between 35.05 and 35.45 . The density varied between 28.23 and 28.55 kg m⁻³. In the Svalbard model, the brine flowing out of the fjord is prescribed just behind the fjord's sill. This is achieved by use of the model's river runoff scheme, which is conveniently configured using netCDF files containing grid cells providing a given flux of water at a given salinity and temperature.

In the Svalbard model, the ice formation and brine release inside the fjord is seen as an

effective addition of salt (and not water). Any brine-enriched water that is added by the river runoff scheme must therefore be balanced by the removal of an equal amount of water (i.e. volume) at a nearby location. In addition to the regions of positive river flux, an equal, but negative, flux is prescribed nearby to balance the inflowing volume. These regions of positive and negative flux are placed around the fjord's sill (Fig. E.8).

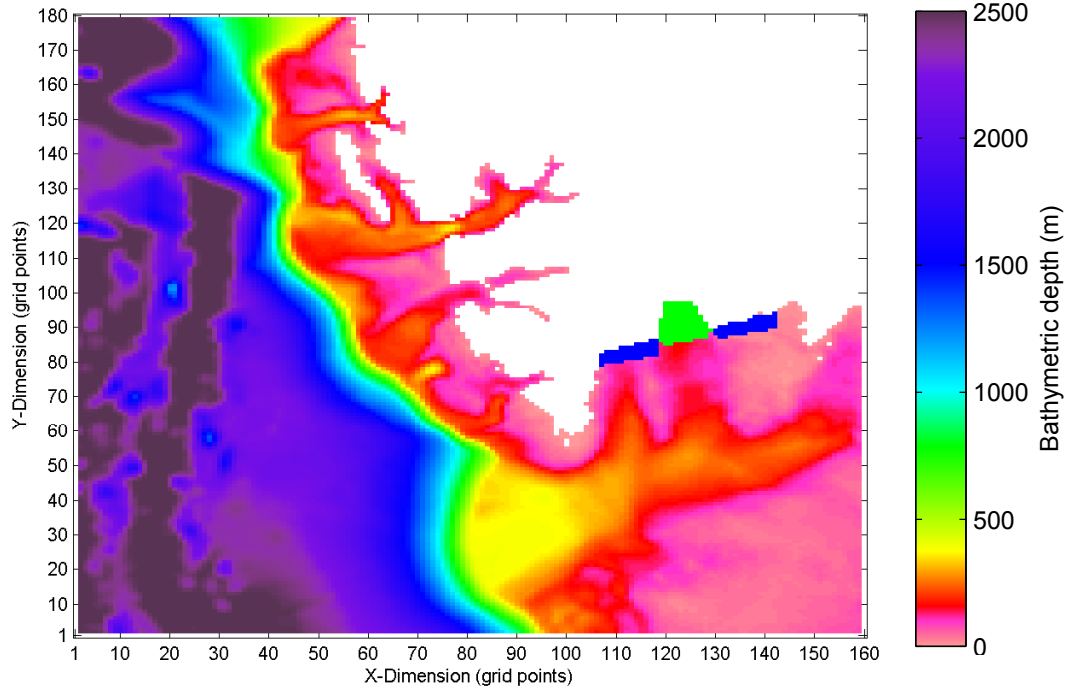


Figure E.8: Crude location map of the regions of positive (green) and negative (blue) river flux. The shading shows the bathymetry of the model domain.

The area of the positive flux region (green in Fig. E.8) is $936 \times 10^9 \text{ m}^2$ (volume = $3.144 \times 10^{10} \text{ m}^3$ covering 104 grid cells). The negative flux area (blue) is $963 \times 10^9 \text{ m}^2$ (volume = $3.542 \times 10^{10} \text{ m}^3$ covering 107 grid cells).

The Matlab script `R0_Create_Rivers_SVAL4.m` defines these areas and produces the river runoff files. The positive flux is applied to layer above the bottom of up to 100 m within the surface (`posflux_dep_from_surface = 100`). The negative flux is applied to a layer from the surface down to 50 m depth (`negflux_dep_from_surface = 50`). The overall volume flux applied to each type of area (posflux and negflux) is the same, but of opposite signs. In all Svalbard runs, the volume flux (time-averaged over the injection period) is 0.06 Sv (`Avg_`

`volume_flux = 0.06 * 1E6`). The total mass of salt that is prescribed to be injected ranges from 250×10^9 kg (`Total_salt_mass = 250E9`) to 1000×10^9 kg (for the scenarios resulting in a sill salinity of 35.2 and 36.1 respectively).

The custom compiler key `key_fluid_inj_bot` turns on several code modifications that allow the river runoff to be applied to a number of grid cells from the surface downwards or a number of grid cells from the bottom upwards. The code is modified in `divcur.F90:div_cur()`, `sshwzv.F90:ssh_wzv()`, `sbc_rnf.F90:sbc_rnf_init()` and `trasbc.F90:tra_sbc()`.

Initially, the model is run without any dense water injection from (beginning of day on) 28th Aug 1985 to (the end of day on) 15th Jan 1986 (140 days, 168000 time steps). This is called the spin-up period where the high-resolution regional model adjusts itself to the external forcing from the global model. For more details on the sequence of runs, see Appendix F.

During the model runs with injections, the overflow is gradually ramped up from 0 to the maximum flux from (beginning of day on) 16th Jan 1986 to (the end of day on) 31st Jan 1986. The steady-flow period on maximum flux lasts from (beginning of day) 1st Feb 1986 until (the end of day on) 31st March 1986, after which the flow is ramped back down to zero between (the beginning of day on) 1st April and (the end of day on) 15th June 1986. In terms of model time steps, the injection phases are defined as follows:

Ramp-up:	from	169201	(start of day on 16-Jan-1986)
	to	188400	(end of day on 31-Jan-1986)
Steady-flow:	from	188401	(start of day on 01-Feb-1986)
	to	259200	(end of day on 31-Mar-1986)
Ramp-down:	from	259201	(start of day on 01-Apr-1986)
	to	350400	(end of day on 15-Jun-1986)

The three periods have a length (in seconds) of ΔT_u , ΔT_s and ΔT_d for Ramp-up, Steady-flow and Ramp-down periods respectively. For the average volume flux during the entire injection period to match the desired $F_{V_{avg}} = 0.06\text{Sv}$ the maximum volume flux $F_{V_{max}}$ is calculated as follows.

$$F_{V_{max}} = F_{V_{avg}} \frac{\Delta T_u + \Delta T_s + \Delta T_d}{0.5(\Delta T_u + \Delta T_d) + \Delta T_s}$$

giving $F_{V_{max}} = 0.0861 \text{ Sv} = 8.61 \times 10^4 \text{ m}^3 \text{ s}^{-1}$. The profile giving the volume flux F_V on a given day during the injection period is shown in Fig. E.9.

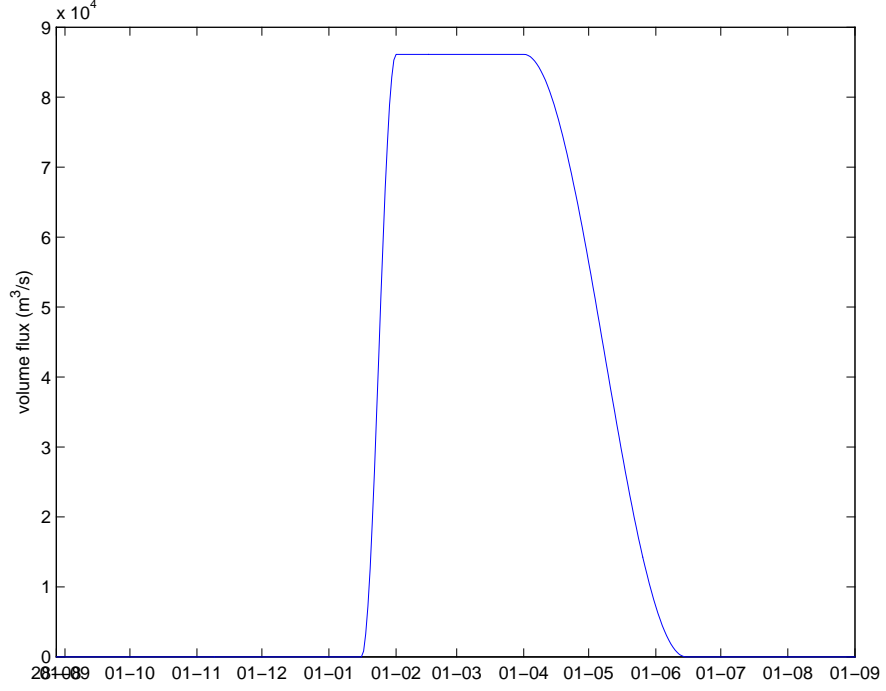


Figure E.9: Flow rate profile for the dense water injection.

This volume flux F_V (in $\text{m}^3 \text{ s}^{-1}$) is then converted to the volume flux F_{RR} (in $\text{m}^2 \text{ s}^{-1}$) which is written to the model's river runoff file.

$$F_{RR} = \frac{F_V * \rho_0}{A};$$

where ρ_0 is NEMO's reference density (1020 kg m^{-3}) and A is the total area over which the flux is applied. As the river inflow area covers 104 grid cells (each a square of $3 \times 3 \text{ km}$), this area A is 936000000 m^2 .

A given mass of salt M_S (ranging from 250 to $1000 \times 10^9 \text{ kg}$) is applied to the inflow region over the overflow period. From this value, the salinity of the inflowing water is calculated as

follows. The total mass of salt M_S (in kg_S , kilogrammes of salt) is converted into an average salt flux $F_{S_{avg}}$ (in kg_S/s):

$$F_{S_{avg}} = \frac{M_S}{\Delta T_u + \Delta T_s + \Delta T_d}$$

where $\Delta T_u, \Delta T_s, \Delta T_d$ are the times (in s) of the three injection periods (see above). As with the volume flux above, this average value is turned into a maximum value (during the steady-flow period):

$$F_{S_{max}} = F_{S_{avg}} \frac{\Delta T_u + \Delta T_s + \Delta T_d}{0.5(\Delta T_u + \Delta T_d) + \Delta T_s}$$

Using the same smooth ramp-up/ramp-down as for the volume flux, this maximum value gives the salt flux F_S (in kg_S/s) on a given day.

First, an assumed source salinity S_{src0} of 34.9 (salinity in the fjord before injection) is converted to a mass of salt per unit volume at the source M_{src} (in kg_S/m^3), which gives

$$M_{src} = \rho_0 * S_{src0} / 1000;$$

The salt flux F_S (in kg_S/s) on a given day gives the salinity in the mass of salt per unit volume M_{fjord} (in kg_S/m^3) inside the fjord:

$$M_{fjord} = \frac{F_S}{F_V} + M_{src}$$

where F_V is the volume flux (in m^3s^{-1}) on that same day. This gives the final salinity S of the injection water:

$$S = M_{fjord} \frac{1000}{\rho_0};$$

This value S is then written to the river runoff file. The temperature of the injected water is kept constant at -1.92°C . The script `R0_Create_Rivers_SVAL4.m` writes these files:

- `injection_runoff_y1985.nc`
- `injection_runoff_y1986.nc`

- `injection_runoff_TS_y1985.nc`
- `injection_runoff_TS_y1986.nc`
- `injection_runoff_dep.nc`

The file `*_runoff_*` contains the volume flux, while `*_runoff_TS_*` contains the T-S of the injected water and the depths at which the water is injected is contained in the `*_runoff_dep_*` file. There are two files each for the runoff and runoff_TS files to cover the entire model period spanning two years of 1985 and 1986.

E.4 NEMO configuration

A NEMO model run is configured for a specific experiment in two ways:

- the **compile-time configuration** (i.e. How the code is compiled)
- the **run-time configuration** (i.e. Which parameters are supplied in the namelist and input files)

The run will work unless the code is compiled according to the run-time parameters and vice versa. This is more of an art form than a science and takes time to develop an intimate understanding of the inner workings of NEMO. For the runs performed for this thesis, almost nothing worked right out of the box. Many parts of the code, namelist and the input files had to be carefully hand-crafted to achieve the desired results.

E.5 Compile-time configuration

NEMO v3.2 uses static memory allocation for its model arrays. The dimensions of the model arrays (such as the variable `sn` holding the current salinity field) are therefore defined at compile-time. These arrays are allocated on the stack at the size defined in the code ³.

The domain decomposition (splitting of global domain into local tiles per processor) is also defined at compile-time. Additionally, the code uses preprocessor keys (known as `#define`'s

³This was changed to dynamic allocation on the heap from v3.3.1, but the array dimensions are still defined at compile-time.

E.5. COMPILE-TIME CONFIGURATION

in the C/C++-world) to turn on/off sections of code depending on the requirements. The knowledge of the exact preprocessor keys is thus just as vital to re-creating a model run as the `namelist` file (see Section E.6).

In light of the aforementioned design of the NEMO code, the following situations require a re-compilation of the NEMO code:

- change of domain resolution (e.g. switch from $1/10^\circ$ to $1/20^\circ$)
- change of domain decomposition (processor tiling)
- change in preprocessor keys for new type of run
- any code changes/bug fixes
- move of the code to a new machine/architecture

The compiled NEMO code is thus highly immobile (between runs and platforms) and a compiled executable cannot be shared between different user groups. A NEMO executable file is always specifically compiled for a given type of run.

E.5.1 How to compile NEMO

NEMO is compiled using **fc**m (details on the installation are given above in Section E.2.2). The compilation is configured in the `bld.cfg` file, which is **fc**m's equivalent of a Makefile. In fact, **fc**m will create a Makefile from the instructions given in `bld.cfg`. The user, however, is only concerned with configuring the `bld.cfg` file. An example of this file is given below.

```
# -----
# File header
# -----

CFG::TYPE      bld # This is a build file
CFG::VERSION   1.0

# -----
# Destination
# -----
# Where is /src located?
DEST           $HERE/..

# -----
```

E.5. COMPILE-TIME CONFIGURATION

```
# Build declarations
# -----

# -----
# Preprocessor keys
# nb the ::NEMO here corresponds to the name of the subdirectory in /src. Case sensitive!
# -----

tool::fppkeys::NEMO key_sval_160_180 key_my_trc key_my_trc_bdy key_top key_bdy key_no_freezing key_traldf_smag ↗
    ↘ key_traldf_c3d key_zdfgls key_mxl_off key_mod_sigma key_nsbbc key_sbc_blk_core_tau key_fluid_inj_bot ↗
    ↘ key_hpg_prj key_vvl key_dynspg_ts key_ldfslp key_mpp_mpi IPROC=10 JPROC=10 IJPROC=83

# -----
# Name of executable. Needs to match name of main code file (model.f90 for NEMO)
# -----

target                model.exe

# -----
# Tell fcm which code files to include
# -----

search_src false

src::IOIPSL            IOIPSL/src
src::NEMO/OPA_SRC      NEMO/OPA_SRC/
src::NEMO/OPA_SRC/ASM  NEMO/OPA_SRC/ASM
src::NEMO/OPA_SRC/BDY  NEMO/OPA_SRC/BDY
src::NEMO/OPA_SRC/C1D  NEMO/OPA_SRC/C1D
src::NEMO/OPA_SRC/DIA  NEMO/OPA_SRC/DIA
src::NEMO/OPA_SRC/DOM  NEMO/OPA_SRC/DOM
src::NEMO/OPA_SRC/DTA  NEMO/OPA_SRC/DTA
src::NEMO/OPA_SRC/DYN  NEMO/OPA_SRC/DYN
src::NEMO/OPA_SRC/FLO  NEMO/OPA_SRC/FLO
src::NEMO/OPA_SRC/IOM  NEMO/OPA_SRC/IOM
src::NEMO/OPA_SRC/LDF  NEMO/OPA_SRC/LDF
src::NEMO/OPA_SRC/OBC  NEMO/OPA_SRC/OBC
src::NEMO/OPA_SRC/OBS  NEMO/OPA_SRC/OBS
src::NEMO/OPA_SRC/SBC  NEMO/OPA_SRC/SBC
src::NEMO/OPA_SRC/SOL  NEMO/OPA_SRC/SOL
src::NEMO/OPA_SRC/TRA  NEMO/OPA_SRC/TRA
src::NEMO/OPA_SRC/TRD  NEMO/OPA_SRC/TRD
src::NEMO/OPA_SRC/ZDF  NEMO/OPA_SRC/ZDF
src::NEMO/TOP_SRC      NEMO/TOP_SRC
src::NEMO/TOP_SRC/TRP  NEMO/TOP_SRC/TRP
src::NEMO/TOP_SRC/MY_TRC NEMO/TOP_SRC/MY_TRC

# -----
# Compiler/linker options
# -----

excl_dep              use::netcdf

exe_dep
pp::NEMO              1

tool::ar              ar
tool::cc              mpicc
```

E.6. RUN-TIME CONFIGURATION

```
tool::cflags      -I$MPI_HOME/include
tool::cpp         mpicc
tool::cppflags    -E -C -I$MPI_HOME/include
tool::fc         mpif90
tool::fflags      -O3 -r8 -traceback -I/usr/local/netcdf4.0.1/include/
tool::fpp         cpp
tool::fpflags     -E -P -traditional -I$MPI_HOME/include

tool::geninterface none
tool::ld         mpif90
tool::ld_libsearch -I$MPI_HOME/include -L$MPI_HOME/lib64
tool::ldflags     -L $MPI_HOME/lib64/ -L/usr/local/netcdf4.0.1/lib/ -lnetcdf -L$HERE/../lib -limf -lm
tool::make       gmake
```

Note, that if netCDF version 4.2 or greater is used, the linker (tool::ldflags) will require two netCDF libraries `-lnetcdf -lnetcdff` because C and Fortran codes are split into separate libraries from v4.2.

To compile NEMO, cd to the `cfg` directory containing the `bld.cfg` file and type the following command:

```
fcm build
```

The default config filename for **fcm** is `bld.cfg`. If the `cfg` file has a different name (e.g. to allow easy switching between build configurations), the following command is used:

```
fcm build <cfg_file_name>
```

To clean a previous compilation use:

```
fcm build <cfg_file_name> --clean
```

Again, the `<cfg_file_name>` argument is optional if the file is called `bld.cfg`.

E.6 Run-time configuration

The run-time configuration consists of the namelist and a number of input files. This part of the configuration (which has to match the compile-time configuration, see Section E.5) changes in situations that include:

- run for a different year (length of run, different forcing files etc.)
- change in vertical grid parameters
- change in forcing files (e.g. turn on river runoff)

- change in diffusion parameters

E.6.1 Run-time config: namelist

The namelist file contains the heart of the model run configuration. Its file name is hard-coded and cannot be changed. It is, however, common to maintain several namelist files under different names, while the run-script links the desired file into the working directory (where the executable runs) under the default name ‘namelist’. The namelist defines the names and locations of most input files (some names of input files are also hard-coded and cannot be changed). The namelist also contains all other run-time parameters.

The format of the namelist file is standardised, as the I/O code reading its contents is built-in to the Fortran language. A small excerpt from a namelist is given below:

```
!-----
&namrun      !  parameters of the run
!-----
nn_no       =      0  !  job number
cn_exp      =  "SVAL" !  experiment name
nn_it000    = 168001 !  first time step (from 16-Jan-1986)
nn_itend    = 441600 !  last time step (until end of day on 31-Aug-1986)
nn_date0    = 19850828 !  initial calendar date yymmdd (used if nn_rstctl=1)
nn_leapy    =      1 !  Leap year calendar (1) or not (0)
nn_istate   =      1 !  output the initial state (1) or not (0)
nn_stock    = 24000 !  frequency of creation of a restart file (modulo referenced to 1) (every 20 days)
nn_write    = 1200 !  frequency of write in the output file (modulo referenced to nit000) (every 24 hours)
ln_dimgnnn  = .false. !  DIMG file format: 1 file for all processors (F) or by processor (T)
ln_mskland  = .false. !  mask land points in NetCDF outputs (costly: + ~15%)
ln_clobber  = .true.  !  clobber (overwrite) an existing file
nn_chunksz  = 100000 !  chunksize (bytes) for NetCDF file (working only with iom_nf90 routines)
ln_rstart   = .true.  !  start from rest (F) or from a restart file (T)
ln_depwr    = .false. !  write depths file
nn_rstctl   =      2 !  restart control = 0 nit000 is not compared to the restart file value
                  !  = 1 use ndate0 in namelist (not the value in the restart file)
                  !  = 2 calendar parameters read in the restart file
cn_ocerst_in = "R03spinupNT_00168000_restart" !  suffix of ocean restart name (input)
cn_ocerst_out = "restart" !  suffix of ocean restart name (output)
tunitfmt    = "('seconds_since_',_,I4.4,'-',I2.2,'-',I2.2,'_',I2.2,':',I2.2,':',I2.2)"/
```

The namelist is structured into sections that start with the & character followed by a section name. The body of a namelist section contains the parameters value, most commonly in the `variable = value` format. The section ends with a \ character on a line on its own. Comments can be added after a ! character and run until the end of the line as in Fortran syntax.

E.6.2 Run-time config: input files

The following input data are provided to the model at run-time as netCDF input files. Most are optional and it depends on the individual run which input files are required. The bathymetry and coordinates files are compulsory.

- **bathymetry** defined in file `bathy_meter.nc`
- **coordinates** defined in file `coordinates.nc`
- **restart file** contains initial T-S, velocity, SSH fields from previous run. The file name is user-defined in namelist file, e.g.

– `R03spinupNT_00168000_restart.nc`

- **initial conditions** If no restart file is available, the initial T-S fields (but not velocity) can also be loaded from netCDF files. This requires compilation with the preprocessor keys `key_dtatem` and `key_dtasal`. These keys activate the subroutines `dta_tem()` and `dta_sal()` which read data from these two files (file names are hard-coded in NEMO v3.2, but configurable in the namelist in later versions):

– `data_1m_potential_temperature_nomask.nc`

– `data_1m_salinity_nomask.nc`

- **atmospheric forcing files** Depending on the type of forcing (e.g. CORE bulk formula) there is typically one file per forcing variable. Below is a namelist section configuring the atmospheric forcing using the Drakkar Forcing Set.

```
!-----
&namsbc_core ! namsbc_core CORE bulk formulae
!-----
!           ! file name ! frequ. (hrs) ! variable ! time interpol. ! clim ! 'yearly' / !
!           ! weights   ! rotation !
!           !           ! (if <0 mnths)! name    ! (logical) ! (T/F) ! 'monthly' !
!           ! filename  ! pairing  !
sn_wndi    = '6hr_DFS4.1_u10' , 6 , 'u10' , .true. , .false. , 'yearly' , .false. ,
!           ! 'DFS_320x161_to_160x180_weights' , 'U'
sn_wndj    = '6hr_DFS4.1_v10' , 6 , 'v10' , .true. , .false. , 'yearly' , .false. ,
!           ! 'DFS_320x161_to_160x180_weights' , 'V'
```


E.6. RUN-TIME CONFIGURATION

```
sn_qsr      = '1d_DFS4.1_swdn' , 24 , 'swdn' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_192x94_to_160x180_weights' , ''  
sn_qlw      = '1d_DFS4.1_lwdn' , 24 , 'lwdn' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_192x94_to_160x180_weights' , ''  
sn_tair     = '6hr_DFS4.1_t2m' , 6 , 't2' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_320x161_to_160x180_weights' , ''  
sn_humi     = '6hr_DFS4.1_q2m' , 6 , 'q2' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_320x161_to_160x180_weights' , ''  
sn_prec     = '1m_DFS4.1_precip' , -1 , 'precip' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_192x94_to_160x180_weights' , ''  
sn_snow     = '1m_DFS4.1_snow' , -1 , 'snow' , .true. , .false. , 'yearly' , .false. , ↵  
             ↵ 'DFS_192x94_to_160x180_weights' , ''  
ln_2m       = .true. ! air temperature and humidity referenced at 2m (T) instead 10m (F)  
ln_taudif   = .false. ! HF tau contribution: use "mean of stress module - module of the mean stress" data  
rn_pfac     = 1. ! multiplicative factor for precipitation (total & snow)  
nn_tau000   = 8400 ! gently increase the wndi/wndj stress over the first tau000 time-steps  
cn_dir      = './' ! root directory for the location of the bulk files (NOTE: run script makes soft-links!)
```

Details on the weights files `DFS_192x94_to_160x180_weights.nc` and `DFS_320x161_to_160x180_weights.nc` which are referred to in the above namelist snippet are given in the previous Section E.2.6.

- **boundary conditions** If open boundaries are used (`key_bdy`) the data for T-S, Velocity, SSH can be provided in a set of files for the barotropic (file name `*_bt_*`) and baroclinic variables:

```
- bdydata_bt_grid_T.nc  
- bdydata_bt_grid_U.nc  
- bdydata_bt_grid_V.nc  
- bdydata_grid_T.nc  
- bdydata_grid_U.nc  
- bdydata_grid_V.nc
```

- **tidal open boundary conditions** are activated by the namelist parameter `nambdy→ln_bdy_tides`. If set to `true`, the following namelist section will be parsed:

```
!-----  
&nambdy_tide ! tidal forcing at unstructured boundaries
```

```
!-----  
fildtide      = 'bdytide_'      ! file name root of tidal forcing files  
tide_cpt(1)   = 'Q1'           ! names of tidal components used  
tide_cpt(2)   = 'O1'           ! names of tidal components used  
tide_cpt(3)   = 'P1'           ! names of tidal components used  
tide_cpt(4)   = 'K1'           ! names of tidal components used  
tide_cpt(5)   = 'N2'           ! names of tidal components used  
tide_cpt(6)   = 'M2'           ! names of tidal components used  
tide_cpt(7)   = 'S2'           ! names of tidal components used  
tide_cpt(8)   = 'K2'           ! names of tidal components used  
tide_cpt(9)   = 'M4'           ! names of tidal components used  
ln_tide_date  = .true.         ! adjust tidal harmonics for start date of run  
ln_harm_ana   = .false.        ! Use Harmonic Analyzer  
ln_tide_czbar = .true.         ! apply Equilibrium Tide  
/
```

This example namelist section defines 9 constituents. NEMO expects 3 files per constituents resulting in a total of 27 files. The following example lists the 3 file names for the M2 constituent:

- bdytide_M2_grid_T.nc
- bdytide_M2_grid_U.nc
- bdytide_M2_grid_V.nc

The Matlab script `R1_Create_BDY_tide.m` was written for this project to create the `bdytide_*.nc` files (see also Section F.4).

E.7 How to run NEMO

The NEMO code is fully parallelised using the MPI (Message Passing Interface) library. MPI is a language-independent communications protocol used to facilitate the communication between the many individual processors (CPUs) in a high-performance computing cluster. The NEMO code is structured that each processor solves the model equations only for a subsection of the global model domain (a so-called ‘tile’). When each processor has finished the computations for a timestep, it exchanges some data with neighbouring processors about the contents of overlapping boundaries between tiles. When each processor has received the boundary data from its neighbours it continues on its own, and in parallel with all other processors, until the next interchange.

The process of decomposing the domain into tiles and exchanging the tiles' boundary data cannot be automated and has to be programmed in the model code using the MPI library calls, and NEMO provides this feature. For the MPI communication to work, an MPI-enabled program has to be started with the command `mpirun`. A typical command line to execute MPI code looks like this:

```
mpirun -np <number of processes> <program>
```

where `<number of processes>` tells the MPI subsystem how many of the computer's processors are to be used. In case of NEMO, the domain decomposition is specified at compile time via the `IPROC`, `JPROC` and `IJPROC` preprocessor keys (see Section E.5.1) and the `<number of processes>` should be equal to the value of `IJPROC`. The second argument `<program>` is the name of the executable including any command-line arguments.

This method of starting NEMO will work on a system where all the available processors are installed on the same computer, i.e. the same motherboard. If the available processors are spread out over more than one physical computer – as in a computing cluster – an additional tool is required to farm the tasks out to different processors on different machines. This is achieved by a resource manager⁴ with a built-in scheduler.

The resource manager in a computing cluster coordinates the processing resources. It keeps track of the jobs that have been submitted to the cluster. The scheduler allows the user to submit jobs, cancel jobs and enquire about the usage and current workload of processors and machines. The scheduler used for NEMO on the University of Plymouth HPC cluster is **qsub**. Its main task is to find a number of free processors in the cluster, and submit the user's job to those machines without the user having to know anything about the workload distribution within the cluster. If the cluster is used to capacity, the scheduler will queue the job and start it as soon as processing resources become available again.

⁴See documentation of TORQUE resource manager for an example. <http://www.clusterresources.com/torquedocs/index.shtml>

E.7.1 Qsub run script

The user submits a NEMO job to the cluster using the **qsub** command and a run script. This script consists of 3 parts:

- parameters for **qsub** (disguised as shell comments after **#PBS**)
- (optional) set up the environment/directory structure for the job to run. Typical steps include:
 1. create working directory
 2. copy model executable to working directory
 3. link or copy model input files (e.g. bathymetry, coordinates, restart files, forcing files etc.) to working directory
- call **mpirun**. The command is given a list of target cluster nodes, selected by the **qsub** scheduler, on which the code is started.

Excerpts from a NEMO run script used for the Svalbard model runs (Chapter 5) are given below. The extensive error handling on the original script has been removed for clarity and brevity.

```
#!/bin/bash
...
# PART (1) Arguments to qsub script

#PBS -S /bin/bash
#PBS -V

# Set the name of the job
#PBS -N R27_250E9Q006NTNW

# Request job to run on a given number of nodes (nodes=...)
#      and a given number of processors (ppn=...)
#PBS -l nodes=11:ppn=8
```

E.7. HOW TO RUN NEMO

```
# Set the queue that the job will run in
#PBS -q default

# Load the modules environment
. /etc/profile.d/modules.sh

# Load the necessary mpirun
# module load ofed/intel/openmpi/1.4.2
# module load intel/ict/3.2

cd ${PBS_O_WORKDIR}

# PART (2) SETUP WORKING DIRECTORY
...
export JOBNAME="${PBS_JOBNAME}"
export SCRIPT_ABS_PATH="$(cd "${PBS_O_WORKDIR}" && pwd -P)"
export SCRIPT_DIR=$(dirname "$SCRIPT_ABS_PATH") # to get the path ↗
export HOME_DIR=$(dirname "$SCRIPT_DIR")
export RUN_SET=$(basename "$SCRIPT_DIR")
export DATA_DIR="${HOME_DIR}/${RUN_SET}/${JOBNAME}"
export EXE_FILE="model.exe"
export WORK_DIR="${HOME_DIR}/RUNS/${JOBNAME}"

# CREATE WORKING DIRECTORY
mkdir -p "${WORK_DIR}"

cd "${WORK_DIR}"

# EXECUTABLE
cp --preserve=timestamps "${EXE_DIR}/${EXE_FILE}" "${WORK_DIR}"

# FPP KEYS used to build the EXEC
export MAKE_FILE="${HOME_DIR}/${RUN_SET}/Makefile"
export NPROCS=$(grep 'export FPPKEYS__NEMO' $MAKE_FILE | perl -pe '
    s/^.*IJPROC=(\d+).*$/\1/' )

# MODEL INPUT FILES
ln -s ${DATA_DIR}/coordinates.nc ${WORK_DIR}/coordinates.nc
```

E.7. HOW TO RUN NEMO

```
ln -s "${DATA_DIR}/bathy_meter.nc" "${WORK_DIR}/bathy_meter.nc"
ln -s "${DATA_DIR}/data_lm_potential_temperature_nomask.nc" ↵
    ↵ "${WORK_DIR}/data_lm_potential_temperature_nomask.nc"
ln -s "${DATA_DIR}/data_lm_salinity_nomask.nc" ↵
    ↵ "${WORK_DIR}/data_lm_salinity_nomask.nc"

# Restart file
for RESTART_FILE in `ls -l ${DATA_DIR}/*restart*.nc 2> /dev/null`
do
    RFILE=$(basename $RESTART_FILE)
    echo "run_NEMO: ln -s $RFILE ${WORK_DIR}/$RFILE"
    ln -s "${DATA_DIR}/$RFILE" "${WORK_DIR}/$RFILE"
done

# Forcing
...
# Open boundary BDY input files
...
# River runoff forcing files
...
# Open boundary BDY tidal constituent files
...

# PART (3) RUN MODEL
mpirun --report-bindings -v -machinefile $PBS_NODEFILE -np $NPROCS ↵
    ↵ $EXE_FILE

exit 0
```

The above excerpts show examples of the 3 parts of the run script. The `qsub` parameters are defined in Part (1). They are given as shell comments, but they will be recognised by `qsub`. The job in the given example will be run on a maximum of 11 nodes with 8 processors each, a total of 88 processors. This total number must be no smaller than the value of the `IJPROC` preprocessor key, but may be larger.

Part (2) of the script sets up a working directory and changes to it (using `cd`). The executable file and all the models input files are copied or linked to the working directory. When the executable is run in the working directory it will create all its output in the working directory. This

way the original run directory isn't populated with the multitude of model output files and the user may choose which ones to copy back to the run directory and archive for post-processing. This is done by the script **combine_outputs** described in the following Section E.7.2.

In the case illustrated above, the working directory `$WORK_DIR` is created under `~/RUNS/`. The subdirectory has the same name as the job name (`#PBS -N xxx`). Some example code is given above to link any netCDF file with `restart` in its file name from the run directory to the working directory. Similar code can be used to create links for forcing and other input files.

The last section of the script, Part (3), actually runs the job using the command `mpirun`. In case of a cluster with multiple nodes it requires a machinefile with a list of cluster nodes that are free to run the job. This list is created by `qsub` as a temporary file whose name is given in the variable `$PBS_NODEFILE`. It is entirely up to `qsub` to choose the list of cluster nodes according to the cluster's workload and job priorities.

With the run script in place, the NEMO job is started with the following command line

```
qsub <run script>
```

where `<run script>` is the script's file name. Once the job is started the user can inspect the progress of the job by monitoring the scheduling queue using the `qstat` command.

During a model run, the text file `time.step` in the working directory is continuously updated containing a single number for the time step index the model is currently working on. This file can be inspected to monitor the progress of the model run. Any textual output from the model is written to the file `ocean.output` which is also created in the working directory. Its contents give a good impression of the progress of the run. In case the run fails, the tag

E R R O R

in the `ocean.output` file marks error messages that give clues as to why the model aborted.

E.7.2 combine_outputs

The shell script **combine_outputs** is run after the model has finished to collate the individual 'per processor' output files (each containing data covering only the processor's domain tile) into single output files covering the entire model domain. The script (which is typically installed

in the user's `~/bin` directory) is started with the following command line, executed in the directory containing the run directories:

```
combine_outputs <name of run>
```

where `<name of run>` is the run's subdirectory. The main work of netCDF processing to combine the fields is done by the program **nocscombine** which was introduced in Section E.2.5.

The script **combine_outputs** drives **nocscombine** and performs several other tasks that finalise a model run. As explained in the previous Section E.7.1, the model is run in a temporary 'working directory', while the 'run directory' contains the model configuration (e.g. namelist) and any run-specific input files. The job of **combine_outputs** is to populate the run directory with all the results that were written to the working directory, such that the working directory can be deleted after **combine_outputs** is completed. **Combine_outputs** achieves the following tasks:

- Copy the following files from the working directory to the run directory (they form a useful reference that documents the model run):

- `solver.stat`
 - `time.step`
 - `date.file`
 - `ocean.output`
 - `layout.dat`

- Process the following sets of per-processor output files (the given names are followed by '`_xxxx`' where `xxxx` is the processor number):

- `mesh_mask`
 - `output.init`
 - `output.abort`

- <restart_file>
- <*_grid_T>
- <*_grid_U>
- <*_grid_V>
- <*_grid_W>
- <*_ptrc_T>

combine_outputs is written to tolerate the case when some of these files don't actually exist. The `output.abort` file, for example, is not usually written if the model completes successfully, but should be processed (and inspected) if it exists.

After completion of **combine_outputs**, the working directory can be deleted and the run directory now contains a complete record of the model's inputs and outputs.

Appendix F

Sequence of runs in NEMO

This appendix describes the sequence of actions that are performed to set up a full forcing run in NEMO, such as those performed for Chapter 5. This sequence is designed for the case where the initial conditions are taken from existing model output (that includes velocity and SSH fields), and will need to be modified for the case where the initial conditions are taken from climatology (which does not include velocity fields). The flow diagram in Fig. F.1 summarises the following description. Several aspects are specific to the Svalbard model runs of Chapter 5, but some techniques (e.g. geostrophic adjustment) can also be used for other model setups.

1. **A simple mesh_mask run.** This first run is set up with only bathymetry and coordinates files. The initial T-S conditions are arbitrary, constant values. The model is run for 1 timestep. This “run” generates a `mesh_mask.nc` file after which it may abort with errors (but this is OK).
2. **Geostrophic adjustment run.** The `mesh_mask.nc` can now be used to interpolate the initial T-S from the $1/12^\circ$ ORCA model onto the SVAL model grid. The initial conditions are fed into a geostrophic adjustment run (where T-S remains fixed, and only velocities evolve over time). This run has no atmospheric forcing, but open boundaries. The open boundary conditions do not evolve in time, but remain fixed (set to the initial conditions). The run is integrated for 12 months. While T-S remain fixed throughout the run, the velocities and SSH fields adjust themselves to the T-S field and the bathymetry. After about 9 months (or earlier) the velocity and SSH fields should settle down. The restart file from the end of this run is used as the initial restart conditions for the following runs. There are several options for the geostrophic adjustment:

- (a) Start from complete rest (zero velocities, and zero SSH)

-
- (b) Start from ORCA model state (existing, i.e. non-zero, velocities, existing SSH field with values of approx. -1.0 to -1.5 m)

For the Svalbard model runs the option (b) was chosen. Hence the geostrophic adjustment run was started with the velocity field already present in the initial conditions.

3. **Full forcing run.** A full run can now be set up using the restart file from the end of the geostrophic adjustment run. This run thus starts in a geostrophically adjusted state. A full run has atmospheric forcing, so as this full run starts up, the wind should not start suddenly. New code was developed (compiler key `key_sbc_blk_core_tau` with namelist parameter `namsbc_core.nn_tau000`) to set a 7 day ramp-up period for the wind strength. The full run also has a simple sea ice-if model and dense water forcing with passive tracers. Tides are optional.

Two sets of ‘spin-up’ runs are performed to advance the model time to the start of the overflow period – one without tides and one with tides. Based on the restart files of the spin-up runs, the full injection runs are performed. Each injection run is performed 4 times for each of the 4 injection scenarios (different dense water salinity) giving a total of 8 runs (Fig. F.1).

Note, that for Chapter 5 the 2 spin-up runs and 8 injections runs were repeated with zero wind-strength (compiler key `key_core_wfac` with namelist parameter `rn_wfac = 0`). These additional experiments aim isolate the effect of the tides independently of any wind-driven effects. This gives a total 16 injection runs (based on 4 spin-up runs), which is not reflected in Fig. F.1 for brevity.

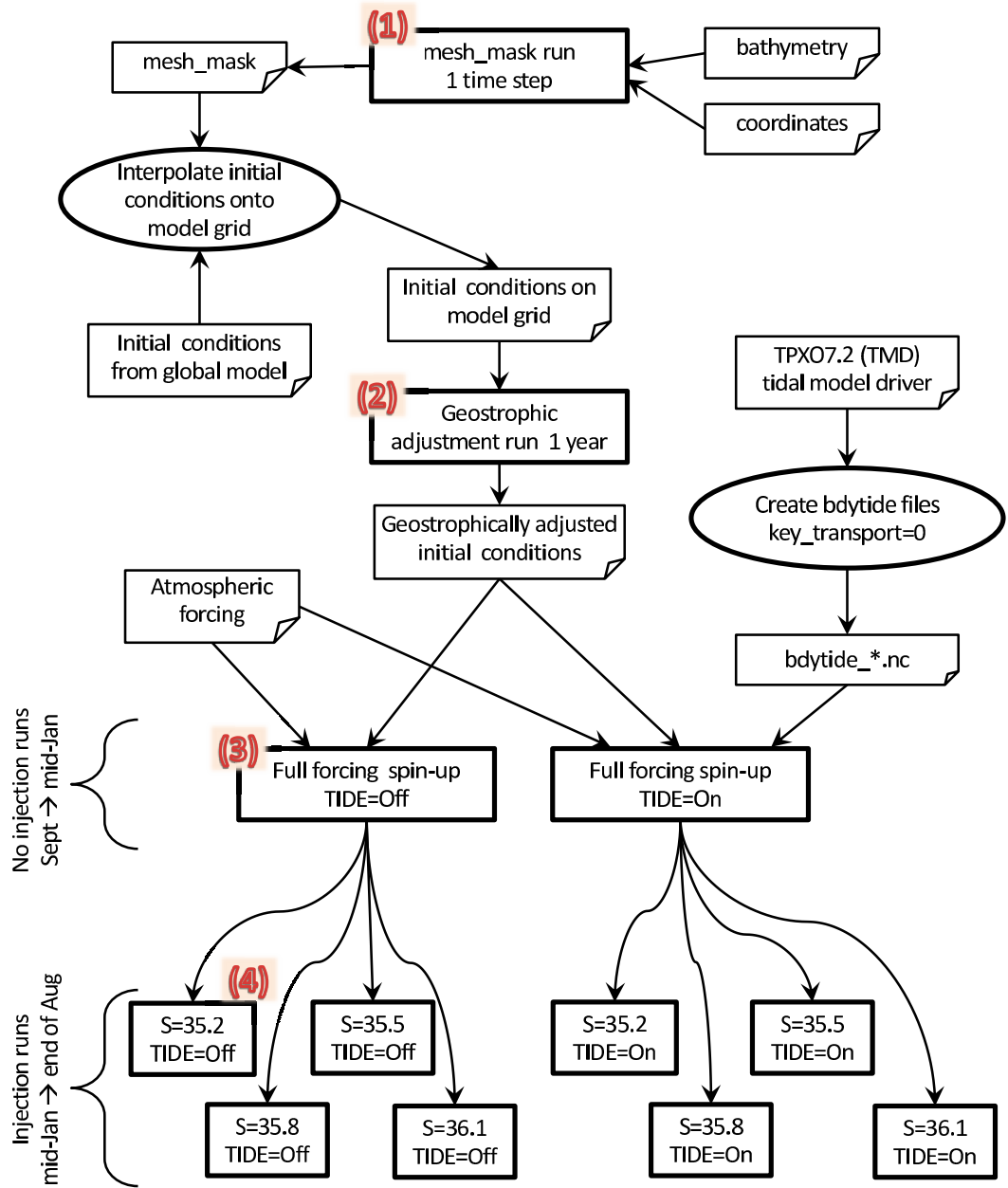


Figure F.1: Schematic flow diagram of sequence of runs leading up to the setup of 8 injection runs. The 4 stages are labelled with numbers in brackets. Model runs are boxes with bold lines. Preprocessing scripts (Matlab) are bold ovals. The arrows show flow control or data I/O.

F.1 Mesh_mask run

This simple run **R01** is compiled with these keys:

Compiler keys: `key_sval_160_180 key_no_freezing key_traldf_smag ↵
key_traldf_c3d key_zdfgls key_mxl_off key_mod_sigma ↵
key_nsbbc key_sbc_blk_core_tau key_fluid_inj_bot key_hpg_prj ↵
key_vvl key_dynspg_ts key_ldfslp key_mpp_mpi IPROC=6 ↵
JPROC=6 IJPROC=31`

1. run `R0_Create_Domain.m` Matlab-script to generate the `coordinates.nc` and `bathy_meter.nc` files.
2. run the model for 1 timestep (no forcing, no open boundaries, no restart file)
3. This run only generates the `mesh_mask.nc` file.

It does not matter whether all of the initialisation files are present or if the `namelist` has errors. The model may report errors if it was configured to load non-existing initial T-S files. However, as the creation of the `mesh_mask` is one of the first things the model does, such an error is of no consequence at this stage.

F.2 Geostrophic adjustment run

The run **R02geo** is compiled with these keys:

Compiler keys: `key_sval_160_180 key_geo_adj key_bdy key_no_freezing ↵
key_traldf_smag key_traldf_c3d key_zdfgls key_mxl_off ↵
key_mod_sigma key_nsbbc key_sbc_blk_core_tau key_fluid_inj_bot ↵
key_hpg_prj key_vvl key_dynspg_ts key_ldfslp key_mpp_mpi IPROC=
6 ↵
JPROC=6 IJPROC=31`

- `key_geo_adj`: geostrophic adjustment. T-S are never updated, only velocities evolve over time and adjust to the T-S field.

F.3. FULL FORCING NON-TIDAL SPIN-UP RUN

- `key_bdy`: open boundaries

This stage requires the `mesh_mask.nc` from run **R01**.

1. run Matlab-script `R0_Create_RestartFile.m` to create the restart file `ORCA0083-N01_19850828_restart.nc` that contains the initial fields of the ORCA 1/12° model.
2. run `R0_Create_OpenBDY.m` to create `bdydata_*` files.

R02geo is then run for 12 months (441600 time steps until 31 Aug 1985) to create a restart file for the following runs (from **R03xx** and above). The resulting file `SVAL_00441600_restart.nc` file is renamed to `R02geo_00441600_restart.nc`.

F.3 Full forcing non-tidal spin-up run

The run **R03spinupNT** is compiled with the following keys:

Compiler keys: `key_sval_160_180 key_bdy key_no_freezing`
↳ `key_traldf_smag key_traldf_c3d key_zdfgls key_mxl_off`
↳ `key_mod_sigma key_nsbbc key_sbc_blk_core_tau`
↳ `key_fluid_inj_bot key_hpg_prj key_vvl key_dynspg_ts key_ldfslp`
↳ `key_mpp_mpi IPROC=6 JPROC=6 IJPROC=31`

- `key_bdy`: open boundaries
- `key_no_freezing`: protection against freezing
- atmospheric forcing is turned on
- tides are turned off

The model is run from 28-Aug-1985 (time step 1) for 140 days until the end of day on 15-Jan-1986 (time step 168000). This is the time period during which there is no injection. This period is thus shared between all runs sharing the same wind and tidal forcing.

F.4. FULL FORCING TIDAL SPIN-UP RUN

It is the first in this series which runs with atmospheric forcing (from Drakkar Forcing Set) and sea-ice forcing (from ORCA 1/12° model). To avoid a shock due to the ‘sudden’ onset of wind (which could lead to unphysical inertial oscillations), the u/v wind stress is gradually ramped up to its normal value over the period of a week (via custom namelist parameter `namsbc_core.nn_tau000 = 8400`).

1. run `scripgrid`, `scrip`, `scripshape` in this order to create weights files `DFS_320x161_to_160x180_weights.nc` for the higher resolution grid of `t2/q2/u10/v10` variables and `DFS_192x94_to_160x180_weights.nc` for lower resolution grid of `swdn/lwdn/precip/snow` variables in the DFS (See E.3.4 for details).
2. run `R0_Create_IceForcing.m` to create annual sea ice forcing files `ORCA0083-N01_SVAL_soicecov_y1985.nc` and `ORCA0083-N01_SVAL_soicecov_y1986.nc`.
3. set up namelist with section `namsbc_iif` and `namsbc.nn_ice = 1`.
4. run **R03spinupNT** for 168000 time steps (open boundaries, atmospheric forcing + ice forcing, initial conditions from geostrophically adjusted restart file `R02geo_00441600_restart.nc`)

The result of this run is a restart file which forms the basis of all further *non-tidal* injection runs.

F.4 Full forcing tidal spin-up run

Compiler keys: `key_sval_160_180` `key_bdy` `key_no_freezing`✓
↳ `key_traldf_smag` `key_traldf_c3d` `key_zdfgls` `key_mxl_off`✓
↳ `key_mod_sigma` `key_nsbbc` `key_sbc_blk_core_tau`✓
↳ `key_fluid_inj_bot` `key_hpg_prj` `key_vvl` `key_dynspg_ts` `key_ldfslp`✓
↳ `key_mpp_mpi` `IPROC=6` `JPROC=6` `IJPROC=31`

This run **R04spinupTIDE** is the tidal equivalent of **R03spinupNT**. Compiler keys are identical, as the tides are controlled by the namelist. This run also uses identical atmospheric and

F.4. FULL FORCING TIDAL SPIN-UP RUN

sea ice forcing files as **R03spinupNT**.

The model is run from 28-Aug-1985 (time step 1) for 140 days until the end of day on 15-Jan-1986 (time step 168000).

1. run `R1_Create_BDY_tide.m` to create `bdytide_*.nc` files
run with `key_transport=0`, i.e. the tidal velocities are those calculated by the tidal model (in m s^{-1}) but the bathymetry may not match between tidal model and NEMO model.
2. run **R04spinupTIDE** for 168000 time steps (open boundaries, atmospheric forcing + ice forcing, tides, initial conditions from geostrophically adjusted restart file `R02geo_00441600_restart.nc`).

The result of this run is a restart file which forms the basis of all further *tidal* injection runs.

List of references.

- Aagaard, K. and Carmack, E. C.: 1989, The role of sea ice and other fresh water in the Arctic circulation, *Journal of Geophysical Research* **94**(C14), 14485–14498.
- Aagaard, K., Coachman, L. K. and Carmack, E. C.: 1981, On the halocline of the Arctic Ocean, *Deep Sea Research Part A. Oceanographic Research Papers* **28**(6), 529–545.
- Aagaard, K., Swift, J. H. and Carmack, E. C.: 1985, Thermohaline Circulation in the Arctic Mediterranean Seas, *Journal of Geophysical Research* **90**(C3), 4833–4846.
- Akimova, A., Schauer, U., Danilov, S. and Núñez-Riboni, I.: 2011, The role of the deep mixing in the Storfjorden shelf water plume, *Deep Sea Research Part I: Oceanographic Research Papers* **58**(4), 403–414.
- Anderson, L. G., Jones, E. P., Lindegren, R., Rudels, B. and Sehlstedt, P. I.: 1988, Nutrient regeneration in cold, high salinity bottom water of the Arctic shelves, *Continental Shelf Research* **8**(12), 1345–1355.
- Baines, P. G. and Condie, S. A.: 1998, Observations and modelling of Antarctic downslope flows: A review, in S. Jacobs and R. Weiss (eds), *Ocean, Ice and Atmosphere: Interactions at the Antarctic Continental Margin*, Vol. 75, American Geophysical Union, pp. 29–49.
- Baringer, M. O. N. and Price, J. F.: 1997, Mixing and spreading of the Mediterranean outflow, *Journal of Physical Oceanography* **27**(8), 1654–1677.
- Drakkar Group (Barnier, B., Brodeau, L., Le Sommer, J., Molines, J.-M., Penduff, T., Theetten, S., Treguier, A.-M., Madec, G., Biastoch, Arne, Böning, Claus W., Dengg, Joachim, Gulev, S., Bourdallé Badie, R., Chanut, J., Garric, G., Alderson, S., Coward, A., de Cuevas, B., New, A., Haines, K., Smith, G., Drijfhout, S., Hazeleger, W., Severijns, C. and Myers, P.): 2007, Eddy permitting ocean circulation hindcasts of past decades, *Clivar Exchanges* **12**(3), 8–10.
- Beckmann, A. and Döscher, R.: 1997, A method for improved representation of dense water spreading over topography in geopotential-coordinate models, *Journal of Physical Oceanography* **27**(4), 581–591.
- Bergamasco, A., Defendi, V., Budillon, G. and Spezie, G.: 2004, Downslope flow observations near Cape Adare shelf-break, *Antarctic Science* **16**(2), 199–204.

LIST OF REFERENCES.

- Blaker, A. T., Hirschi, J. J.-M., Sinha, B., de Cuevas, B., Alderson, S., Coward, A. and Madec, G.: 2012, Large near-inertial oscillations of the Atlantic meridional overturning circulation, *Ocean Modelling* **42**, 50–56.
- Blumberg, A. F. and Mellor, G. L.: 1983, Diagnostic and prognostic numerical circulation studies of the South Atlantic Bight, *Journal of Geophysical Research* **88**(C8), 4579–4592.
- Bolaños, R., Osuna, P., Wolf, J., Monbaliu, J. and Sanchez-Arcilla, A.: 2007, The POLCOMS-WAM wave-current interaction model: development and performance in the NW Mediterranean, in C. Guedes-Soares and P. Kolev (eds), *Maritime industry, ocean engineering and coastal resources: proceedings of the 12th International Congress of the International Maritime Association of the Mediterranean, IMAM, Varna, Bulgaria, 2-6 September 2007*, Taylor Francis, pp. 685–691.
- Brodeau, L., Barnier, B., Treguier, A.-M., Penduff, T. and Gulev, S.: 2010, An ERA40-based atmospheric forcing for global ocean circulation models, *Ocean Modelling* **31**(3-4), 88–104.
- Burchard, H. and Bolding, K.: 2002, GETM – A General Estuarine Transport Model. Scientific Documentation. Technical Report EUR 20253 EN, European Commission,
- Canals, M., Danovaro, R., Heussner, S., Lykousis, V., Puig, P., Trincardi, F., Calafat, A. M., Palanques, A. and Sánchez-Vidal, A.: 2009, Cascades in Mediterranean Submarine Grand Canyons, *Oceanography* **22**(1), 26–43.
- Canuto, V. M., Howard, A., Cheng, Y. and Dubovikov, M. S.: 2001, Ocean Turbulence. Part I: One-Point Closure Model–Momentum and Heat Vertical Diffusivities, *Journal of Physical Oceanography* **31**(6), 1413–1426.
- Carmack, E. C.: 2000, The Arctic Ocean’s freshwater budget: Sources, Storage and Export, in E. L. Lewis, E. P. Jones, P. Lemke, T. D. Prowse and P. Wadhams (eds), *The freshwater budget of the Arctic Ocean*, Kluwer Academic Publishers, The Netherlands, pp. 91–126.
- Cavalieri, D. J. and Martin, S.: 1994, The contribution of Alaskan, Siberian and Canadian coastal polynyas to the halocline layer of the Arctic Ocean, *Journal of Geophysical Research* **99**(C9), 18343–18362.
- Cenedese, C. and Adduce, C.: 2010, A new parameterization for entrainment in overflows, *Journal of Physical Oceanography* **40**(8), 1835–1850.
- Cenedese, C., Whitehead, J. A., Ascarelli, T. A. and Ohiwa, M.: 2004, A dense current flowing down a sloping bottom in a rotating fluid, *Journal of Physical Oceanography* **34**(1), 188–203.

LIST OF REFERENCES.

- Colella, P. and Woodward, P. R.: 1984, The piecewise-parabolic method (PPM) for gas-dynamical simulations, *Journal of Computational Physics* **54**(1), 174–201.
- Conkright, M. E., Locarnini, R. A., Garcia, H. E., O’Brien, T. D., Boyer, T. P., Stephens, C. and Antonov, J. I.: 2002, World Ocean Atlas 2001: Objective Analyses, Data Statistics, and Figures, CDROM Documentation, National Oceanographic Data Center, Silver Spring, MD,
- Cooper, L. H. N. and Vaux, D.: 1949, Cascading Over the Continental Slope of Water from the Celtic Sea, *Journal of the Marine Biological Association of the United Kingdom* **28**, 719–750.
- Courant, R., Friedrichs, K. and Lewy, H.: 1928, Über die partiellen Differenzengleichungen der mathematischen Physik, *Mathematische Annalen* **100**(1), 32–74.
- Cushman-Roisin, B. and Beckers, J.-M.: 2011, *Introduction to Geophysical Fluid Dynamics, 2nd Edition - Physical and Numerical Aspects*, Academic Press. URL: <http://engineering.dartmouth.edu/~cushman/books/GFD.html>
- Darelius, E., Smedsrud, L. H., Østerhus, S., Foldvik, A. and Gammelsrød, T.: 2009, Structure and variability of the Filchner overflow plume, *Tellus A* **61**(3), 446–464.
- Davies, H. C.: 1976, A lateral boundary formulation for multi-level prediction models, *Quarterly Journal of the Royal Meteorological Society* **102**(432), 405–418.
- Day, M. A.: 1990, The no-slip condition of fluid dynamics, *Erkenntnis* **33**(3), 285–296.
- Dyke, P. P. G.: 2007, *Modeling Coastal And Offshore Processes*, Imperial College Press.
- Economidou, M. and Hunt, G. R.: 2009, Density stratified environments: the double-tank method, *Experiments in Fluids* **46**(3), 453–466.
- Edwards, K. P., Barciela, R. and Butenschön, M.: 2012, Validation of the NEMO-ERSEM operational ecosystem model for the North West European Continental Shelf, *Ocean Science* **8**(6), 983–1000.
- Egbert, G. and Erofeeva, S.: 2002, Efficient inverse modeling of barotropic ocean tides, *Journal of Atmospheric and Oceanic Technology* **19**(2), 183–204.
- Ekman, V. W.: 1905, On the influence of the Earth’s rotation on ocean-currents, *Arkiv för Matematik, Astronomi och Fysik* **2**(11), 1–53. URL: <http://www.aos.princeton.edu/WWWPUBLIC/gkv/history/Ekman05.pdf>
- Engedahl, H.: 1995, Use of the flow relaxation scheme in a three-dimensional baroclinic ocean model with realistic topography, *Tellus A* **47**(3), 365–382.

LIST OF REFERENCES.

- Engquist, B. and Majda, A.: 1977, Absorbing boundary conditions for numerical simulation of waves, *Proceedings of the National Academy of Sciences* **74**(5), 1765–1766. URL: <http://www.pnas.org/content/74/5/1765.abstract>
- Enriquez, C. E., Shapiro, G. I., Souza, A. J. and Zatsepin, A. G.: 2005, Hydrodynamic modelling of mesoscale eddies in the Black Sea, *Ocean Dynamics* **55**(5), 476–489.
- Etling, D., Gelhardt, F., Schrader, U., Brennecke, F., Kühn, G., Chabert d’Hieres, G. and Didelle, H.: 2000, Experiments with density currents on a sloping bottom in a rotating fluid, *Dynamics of Atmospheres and Oceans* **31**(1-4), 139–164.
- Fer, I. and Ådlandsvik, B.: 2008, Descent and mixing of the overflow plume from Storfjord in Svalbard: an idealized numerical model study, *Ocean Science* **4**(2), 115–132.
- Fer, I., Skogseth, R. and Haugan, P. M.: 2004, Mixing of the Storfjorden overflow (Svalbard Archipelago) inferred from density overturns, *Journal of Geophysical Research* **109**(C1), C01005.
- Fer, I., Skogseth, R., Haugan, P. M. and Jaccard, P.: 2003, Observations of the Storfjorden overflow, *Deep Sea Research Part I: Oceanographic Research Papers* **50**(10-11), 1283–1303.
- Flather, R. A.: 1976, A tidal model of the northwest European continental shelf, *Memoires de la Societe Royale de Sciences de Liege* **6**, 141–164.
- Furevik, T., Mauritzen, C. and Ingvaldsen, R.: 2007, The flow of Atlantic water to the Nordic Seas and Arctic Ocean, in J. B. Ørbæk, R. Kallenborn, I. Tombre, E. N. Hegseth, S. Falk-Petersen and A. H. Hoel (eds), *Arctic Alpine Ecosystems and People in a Changing Environment*, Springer, pp. 123–146.
- Galperin, B., Kamntha, L. H., Hassid, S. and Rosati, A.: 1988, A quasi-equilibrium turbulent energy model for geophysical flows, *Journal of the Atmospheric Sciences* **45**(1), 55–62.
- Geyer, F., Fer, I. and Eldevik, T.: 2009, Dense overflow from an Arctic fjord: Mean seasonal cycle, variability and wind influence, *Continental Shelf Research* **29**(17), 2110–2121.
- Geyer, F., Fer, I. and Smedsrud, L. H.: 2010, Structure and forcing of the overflow at the Storfjorden sill and its connection to the Arctic coastal polynya in Storfjorden, *Ocean Science* **6**(1), 401–411.
- Geyer, W. R. and Signell, R. P.: 1992, A reassessment of the role of tidal dispersion in estuaries and bays, *Estuaries and Coasts* **15**(2), 97–108.
- Gill, A. E.: 1982, *Atmosphere–Ocean Dynamics*, Academic Press.

LIST OF REFERENCES.

- Girton, J. B. and Sanford, T. B.: 2003, Descent and modification of the overflow plume in the Denmark Strait, *Journal of Physical Oceanography* **33**(7), 1351–1364.
- Gjevik, B., Nøst, E. and Straume, T.: 1994, Model simulations of the tides in the Barents Sea, *Journal of Geophysical Research: Oceans* **99**(C2), 3337–3350.
- Gordon, A. L., Zambianchi, E., Orsi, A. H., Visbeck, M., Giulivi, C. F., Whitworth, Thomas, I. and Spezie, G.: 2004, Energetic plumes over the western Ross Sea continental slope, *Geophysical Research Letters* **31**(21), L21302.
- Gräwe, U., Holtermann, P., Klingbeil, K. and Burchard, H.: 2013, Modelling the vertical thermal stratification in the North Sea - advantages of using adaptive coordinates, *Geophysical Research Abstracts* **15**, EGU2013–3467. URL: <http://meetingorganizer.copernicus.org/EGU2013/EGU2013-3467.pdf>
- Griffies, S. M.: 2004, *Fundamentals of Ocean Climate Models*, Princeton University Press.
- Griffies, S. M., Gnanadesikan, A., Pacanowski, R. C., Larichev, V. D., Dukowicz, J. K. and Smith, R. D.: 1998, Isonutral diffusion in a z-coordinate ocean model, *Journal of Physical Oceanography* **28**(5), 805–830.
- Griffiths, R. W.: 1986, Gravity currents in rotating systems, *Annual Review of Fluid Mechanics* **18**(1), 59–89.
- Griffiths, R. W. and Linden, P. F.: 1982, Laboratory experiments on fronts. Part 1 Density-driven boundary currents, *Geophysical and Astrophysical Fluid Dynamics* **19**(3-4), 159–187.
- Guan, X., Ou, H.-W. and Chen, D.: 2009, Tidal effect on the dense water discharge, Part 2: A numerical study, *Deep Sea Research Part II: Topical Studies in Oceanography* **56**(13-14), 884–894.
- Haarpaintner, J.: 1999, The Storfjorden Polynya: ERS-2 SAR observations and overview, *Polar Research* **18**(2), 175–182.
- Haarpaintner, J., Gascard, J.-C. and Haugan, P. M.: 2001, Ice production and brine formation in Storfjorden, Svalbard, *Journal of Geophysical Research: Oceans* **106**(C7), 14001–14013.
- Haney, R. L.: 1991, On the pressure gradient force over steep topography in sigma coordinate ocean models, *Journal of Physical Oceanography* **21**(4), 610–619.
- Heywood, K. J., Naveira Garabato, A. C. and Stevens, D. P.: 2002, High mixing rates in the abyssal Southern Ocean, *Nature* **415**(6875), 1011–1014.

LIST OF REFERENCES.

- Hiester, H. R., Piggott, M. D. and Allison, P. A.: 2011, The impact of mesh adaptivity on the gravity current front speed in a two-dimensional lock-exchange, *Ocean Modelling* **38**(1-2), 1–21.
- Hill, A. E., Souza, A. J., Jones, K., Simpson, J. H., Shapiro, G. I., McCandliss, R., Wilson, H. and Leftley, J.: 1998, The Malin cascade in winter 1996, *Journal of Marine Research* **56**(1), 87–106. URL: <http://www.ingentaconnect.com/content/jmr/jmr/1998/00000056/00000001/art00004>
- Hockney, R. W. and Jesshope, C. R.: 1988, *Parallel Computers (2nd Edition)*, Adam Hilger.
- Holloway, G. and Proshutinsky, A.: 2007, Role of tides in Arctic ocean/ice climate, *Journal of Geophysical Research: Oceans* **112**(C4), C04S06.
- Holt, J. T. and James, I. D.: 2001, An s-coordinate density evolving model of the northwest European continental shelf, Part 1: Model description and density structure, *Journal of Geophysical Research* **106**(C7), 14015–14034.
- Holt, J. T. and James, I. D.: 2006, An assessment of the fine-scale eddies in a high-resolution model of the shelf seas west of Great Britain, *Ocean Modelling* **13**(3-4), 271–291.
- Holt, J. T. and Proctor, R.: 2001, Dispersion in Shallow Seas, in J. H. Steele, S. A. Thorpe and K. K. Turekian (eds), *Encyclopedia of Ocean Sciences*, Elsevier, pp. 742–747.
- Holt, J. T. and Umlauf, L.: 2008, Modelling the tidal mixing fronts and seasonal stratification of the Northwest European Continental shelf, *Continental Shelf Research* **28**(7), 887–903.
- Holt, J. T., Wakelin, S. and Huthnance, J. M.: 2009, Down-welling circulation of the northwest European continental shelf: A driving mechanism for the continental shelf carbon pump, *Geophysical Research Letters* **36**(14), L14602.
- Huthnance, J. M.: 1995, Circulation, exchange and water masses at the ocean margin: the role of physical processes at the shelf edge, *Progress In Oceanography* **35**(4), 353–431.
- Huthnance, J. M., Holt, J. T. and Wakelin, S. L.: 2009, Deep ocean exchange with west-European shelf seas, *Ocean Science* **5**(4), 621–634.
- Ilicak, M., Adcroft, A. J., Griffies, S. M. and Hallberg, R. W.: 2012, Spurious diapycnal mixing and the role of momentum closure, *Ocean Modelling* **45–46**, 37–58.
- Ivanov, V.: 2011, How summer ice depletion in the Arctic Ocean may affect the global THC?, *Geophysical Research Abstracts* **13**, EGU2011–4457. URL: <http://meetingorganizer.copernicus.org/EGU2011/EGU2011-4457.pdf>

LIST OF REFERENCES.

- Ivanov, V. V. and Golovin, P. N.: 2007, Observations and modeling of dense water cascading from the northwestern Laptev Sea shelf, *Journal of Geophysical Research* **112**(C9), C09003.
- Ivanov, V. V., Shapiro, G. I., Huthnance, J. M., Aleynik, D. L. and Golovin, P. N.: 2004, Cascades of dense water around the world ocean, *Progress In Oceanography* **60**(1), 47–98.
- Jakobsson, M., Macnab, R., Mayer, L., Anderson, R., Edwards, M., Hatzky, J., Schenke, H. W. and Johnson, P.: 2008, An improved bathymetric portrayal of the Arctic Ocean: Implications for ocean modeling and geological, geophysical and oceanographic analyses, *Geophysical Research Letters* **35**(7), L07602.
- Jakobsson, M., Mayer, L., Coakley, B., Dowdeswell, J. A., Forbes, S., Fridman, B., Hodnesdal, H., Noormets, R., Pedersen, R., Rebesco, M., Schenke, H. W., Zarayskaya, Y., Accettella, D., Armstrong, A., Anderson, R. M., Bienhoff, P., Camerlenghi, A., Church, I., Edwards, M., Gardner, J. V., Hall, J. K., Hell, B., Hestvik, O., Kristoffersen, Y., Marcussen, C., Mohammad, R., Mosher, D., Nghiem, S. V., Pedrosa, M. T., Travaglini, P. G. and Weatherall, P.: 2012, The International Bathymetric Chart of the Arctic Ocean (IBCAO) Version 3.0, *Geophysical Research Letters* **39**(12), L12609.
- Jungclauss, J. H., Backhaus, J. O. and Fohrmann, H.: 1995, Outflow of dense water from the Storfjord in Svalbard: A numerical model study, *Journal of Geophysical Research* **100**(C12), 24719–24728.
- Kamenkovich, V. M.: 1977, *Fundamentals of ocean dynamics*, Elsevier.
- Kämpf, J.: 2005, Cascading-driven upwelling in submarine canyons at high latitudes, *Journal of Geophysical Research* **110**(C2), C02007.
- Killworth, P. D.: 1977, Mixing of the Weddell Sea continental slope, *Deep Sea Research* **24**(5), 427–448.
- Killworth, P. D.: 1983, Deep convection in the World Ocean, *Reviews of Geophysics* **21**(1), 1–26.
- Killworth, P. D.: 2001, On the rate of descent of overflows, *Journal of Geophysical Research* **106**(C10), 22267–22275.
- Killworth, P. D.: 2003, Inclusion of the bottom boundary layer in ocean models, in P. Müller (ed.), *Proceedings of the 13th 'Aha Huliko'a Hawaiian Winter Workshop on Near Boundary Processes and their Parameterization, 2003*, pp. 177–185. URL: <http://www.soest.hawaii.edu/PubServices/2003pdfs/Killworth.pdf>

LIST OF REFERENCES.

- Lane-Serff, G. . and Baines, P. G.: 2000, Eddy formation by overflows in stratified water, *Journal of Physical Oceanography* **30**(2), 327–337.
- Lane-Serff, G. F.: 2009, Overflows and Cascades, in J. H. Steele, K. K. Turekian and S. A. Thorpe (eds), *Encyclopedia of Ocean Sciences*, Academic Press, Oxford, pp. 265–271.
- Lane-Serff, G. F. and Baines, P. G.: 1998, Eddy formation by dense flows on slopes in a rotating fluid, *Journal of Fluid Mechanics* **363**, 229–252.
- Large, W. G. and Yeager, S.: 2004, Diurnal to decadal global forcing for ocean and sea-ice models : the data sets and flux climatologies. NCAR Technical Note, NCAR/TN-460+STR, CGD Division of the National Center for Atmospheric Research,
- Legg, S., Chang, Y., Chassignet, E. P., Danabasoglu, G., Ezer, T., Gordon, A. L., Griffies, S. M., Hallberg, R. W., Jackson, L., Large, W. G., Özgökmen, T., Peters, H., Price, J. F., Riemschneider, U., Wu, W., Xu, X. and Yang, J.: 2009, Improving oceanic overflow representation in climate models: the Gravity Current Entrainment Climate Process Team, *Bulletin of the American Meteorological Society* **90**(5), 657–670.
- Legg, S., Jackson, L. and Hallberg, R. W.: 2008, Eddy-resolving modeling of overflows, *Geophysical Monograph Series* **117**, 63–81.
- Levier, B., Treguier, A.-M., Madec, G. and Garnier, V.: 2007, Free surface and variable volume in the nemo code, MERSEA IP report WP09-CNRSSTR-03-1A, Laboratoire de Physique des Oceans, Brest, URL: http://www.nemo-ocean.eu/content/download/258/1661/version/1/file/NEMO_vvl_report.pdf
- Liu, H. and Holt, J. T.: 2010, *Combination of the Vertical PPM Advection Scheme with the Existing Horizontal Advection Schemes in NEMO*, MyOcean Science Days, 1-3 December 2010, Météo-France International Conference Center Toulouse, France. URL: http://mercator-myoceanv2.netaktiv.com/MSD_2010/Abstract/Abstract_LIUhedong_MSD_2010.doc
- Luneva, M. V. and Holt, J. T.: 2010, *Physical shelf processes operating in the NOCL Arctic Ocean model*, Arctic Ocean Model Intercomparison Project, Workshop 14, 19-22 October 2010, Woods Hole Oceanographic Institution. URL: <http://www.whoi.edu/fileservlet.do?id=77125&pt=2&p=83808>
- Madec, G.: 2008, NEMO ocean engine. Note du Pôle de modélisation, Institut Pierre-Simon Laplace (IPSL), France, No. 27. ISSN: 1288-1619. URL: <http://www.nemoocean.eu/About-NEMO/Reference-manuals>

LIST OF REFERENCES.

- Madec, G.: 2011, NEMO ocean engine v3.3. Note du Pôle de modélisation, Institut Pierre-Simon Laplace (IPSL), France, No. 27. ISSN: 1288-1619. URL: <http://www.nemoocean.eu/About-NEMO/Reference-manuals>
- Maqueda, M. A. M., Willmott, A. J. and Biggs., N. R. T.: 2004, Polynya dynamics: a review of observations and modeling, *Reviews of Geophysics* **42**(1), RG1004.
- Marchesiello, P., McWilliams, J. C. and Shchepetkin, A.: 2001, Open boundary conditions for long-term integration of regional oceanic models, *Ocean Modelling* **3**(1-2), 1–20.
- Marshall, J., Hill, C., Perelman, L. and Adcroft, A.: 1997, Hydrostatic, quasi-hydrostatic, and nonhydrostatic ocean modeling, *Journal of Geophysical Research* **102**(C3), 5733–5752.
- Marshall, J. and Schott, F.: 1999, Open-ocean convection: Observations, theory, and models, *Reviews of Geophysics* **37**(1), 1–64.
- Martinho, A. S. and Batteen, M. L.: 2006, On reducing the slope parameter in terrain-following numerical ocean models, *Ocean Modelling* **13**(2), 166–175.
- Matt, S. and Johns, W. E.: 2007, Transport and entrainment in the Red Sea outflow plume, *Journal of Physical Oceanography* **37**(4), 819–836.
- McCabe, R. M., MacCready, P. and Pawlak, G.: 2006, Form Drag due to Flow Separation at a Headland, *Journal of Physical Oceanography* **36**(11), 2136–2152.
- McDougall, T. J. and Jackett, D. R.: 2005, The material derivative of neutral density, *Journal of Marine Research* **63**, 159–185.
- McEwan, R.: 2010, Non-hydrostatic effects in a propagating large-scale river plume, *14th Biennial Challenger Conference for Marine Science, Book of Abstracts*, p. 93. URL: <http://www.challenger2010.org.uk/programme>
- McWilliams, J. C., Huckle, E. and Shchepetkin, A. F.: 2009, Buoyancy effects in a stratified Ekman layer, *Journal of Physical Oceanography* **39**(10), 2581–2599.
- Melling, H. and Lewis, E.: 1982, Shelf drainage flows in the Beaufort Sea and their effect on the Arctic Ocean pycnocline, *Deep-Sea Research A* **29**(8), 967–985.
- Mellor, G. L.: 1991, An Equation of State for Numerical Models of Oceans and Estuaries, *Journal of Atmospheric and Oceanic Technology* **8**(4), 609–611.
- Mellor, G. L., Ezer, T. and Oey, L.-Y.: 1994, The pressure gradient conundrum of sigma coordinate ocean models, *Journal of Atmospheric and Oceanic Technology* **11**(4), 1126–1134.

LIST OF REFERENCES.

- Murray, A. B.: 2003, Contrasting the goals, strategies, and predictions associated with simplified numerical models and detailed simulations, in P. R. Wilcock and R. M. Iverson (eds), *Prediction in Geomorphology, Geophys. Monogr. Ser., vol. 135*, AGU, Washington, D. C., pp. 151–165.
- Nof, D.: 1983, The translation of isolated cold eddies on a sloping bottom, *Deep Sea Research Part A. Oceanographic Research Papers* **30**(2), 171–182.
- O’Dea, E. J., Arnold, A. K., Edwards, K. P., Furner, R., Hyder, P., Martin, M. J., Siddorn, J. R., Storkey, D., While, J., Holt, J. T. and Liu, H.: 2012, An operational ocean forecast system incorporating NEMO and SST data assimilation for the tidally driven European North-West shelf, *Journal of Operational Oceanography* **5**(1), 3–17. URL: <http://www.ingentaconnect.com/content/imarest/joo/2012/00000005/00000001/art00002>
- O’Neill, C. K., Polton, J. A., Holt, J. T. and O’Dea, E. J.: 2012, Modelling temperature and salinity in Liverpool Bay and the Irish Sea: sensitivity to model type and surface forcing, *Ocean Science* **8**(5), 903–913.
- Orsi, A. H. and Wiederwohl, C. L.: 2009, A recount of Ross Sea waters, *Deep Sea Research Part II: Topical Studies in Oceanography* **56**(13-14), 778–795.
- Oster, G.: 1965, Density Gradients, *Scientific American* **213**, 70–76.
- Ou, H.-W., Guan, X. and Chen, D.: 2009, Tidal effect on the dense water discharge, Part 1: Analytical model, *Deep Sea Research Part II: Topical Studies in Oceanography* **56**(13-14), 874–883. Southern Ocean Shelf Slope Exchange.
- Özgökmen, T. M. and Chassignet, E. P.: 2002, Dynamics of two-dimensional turbulent bottom gravity currents, *Journal of Physical Oceanography* **32**(5), 1460–1478.
- Padman, L., Howard, S. L., Orsi, A. H. and Muench, R. D.: 2009, Tides of the northwestern Ross Sea and their impact on dense outflows of Antarctic Bottom Water, *Deep Sea Research Part II: Topical Studies in Oceanography* **56**(13-14), 818–834.
- Pedlosky, J.: 1987, *Geophysical Fluid Dynamics*, 2nd edn, Springer.
- Phillips, N. A.: 1957, A coordinate system having some special advantages for numerical forecasting, *Journal of Meteorology* **14**(2), 184–185.
- Piechura, J.: 1996, Dense bottom waters in Storfjord and Storfjordrenna, *Oceanologia* **38**(2), 285–292.

LIST OF REFERENCES.

- Piggott, M. D., Gorman, G. J., Pain, C. C., Allison, P. A., Candy, A. S., Martin, B. T. and Wells, M. R.: 2008, A new computational framework for multi-scale ocean modelling based on adapting unstructured meshes, *International Journal for Numerical Methods in Fluids* **56**(8), 1003–1015.
- Postlethwaite, C. F., Maqueda, M. A. M., le Fouest, V., Tattersall, G. R., Holt, J. T. and Willmott, A. J.: 2011, The effect of tides on dense water formation in Arctic shelf seas, *Ocean Science* **7**(2), 203–217.
- Price, J. F., Baringer, M. O., Lueck, R. G., Johnson, G. C., Ambar, I., Parrilla, G., Cantos, A., Kennelly, M. A. and Sanford, T. B.: 1993, Mediterranean Outflow mixing and dynamics, *Science* **259**(5099), 1277–1282.
- Quadfasel, D., Rudels, B. and Kurz, K.: 1988, Outflow of dense water from a Svalbard fjord into the Fram Strait, *Deep Sea Research Part A. Oceanographic Research Papers* **35**(7), 1143–1150.
- Roos, P. C.: 2013, Idealized modelling of tide propagation in large-scale semi-enclosed basins, *Geophysical Research Abstracts* **15**, EGU2013–13823. URL: <http://meetingorganizer.copernicus.org/EGU2013/EGU2013-13823.pdf>
- Ross, A. N., Linden, P. F. and Dalziel, S. B.: 2002, A study of three-dimensional gravity currents on a uniform slope, *Journal of Fluid Mechanics* **453**, 239–261.
- Rudels, B.: 1995, The Thermohaline Circulation of the Arctic Ocean and the Greenland Sea, *Philosophical Transactions of the Royal Society of London. Series A: Physical and Engineering Sciences* **352**(1699), 287–299.
- Rudels, B.: 2009, Arctic Ocean Circulation, in J. H. Steele, K. K. Turekian and S. A. Thorpe (eds), *Encyclopedia of Ocean Sciences*, Academic Press, Oxford, pp. 211–225.
- Rudels, B.: 2012, Arctic Ocean circulation and variability – advection and external forcing encounter constraints and local processes, *Ocean Science* **8**(2), 261–286.
- Rudels, B., Anderson, L. G. and Jones, E. P.: 1996, Formation and evolution of the surface mixed layer and halocline of the Arctic Ocean, *Journal of Geophysical Research* **101**(C4), 8807–8821.
- Rudels, B., Björk, G., Nilsson, J., Lake, I. and Nohr, C.: 2005, The interactions between waters from the Arctic Ocean and the Nordic Seas north of Fram Strait and along the East Greenland Current: results from the Arctic Ocean-02 Oden Expedition, *Journal of Marine Systems* **55**(1–2), 1–30.

LIST OF REFERENCES.

- Rudels, B., J. Friedrich, H. and Quadfasel, D.: 1999, The Arctic Circumpolar Boundary Current, *Deep Sea Research Part II: Topical Studies in Oceanography* **46**(6-7), 1023–1062.
- Rudels, B., Jones, E. P., Anderson, L. G. and Kattner, G.: 1994, On the intermediate depth waters of the Arctic ocean, in O. M. Johannessen, R. D. Muench and J. E. Overland (eds), *The Polar Oceans and their role in shaping the global environment*, Vol. Geophysical Monograph 85, American Geophysical Union, Washington, D.C., pp. 33–46.
- Rudels, B. and Quadfasel, D.: 1991, Convection and deep water formation in the Arctic Ocean-Greenland Sea System, *Journal of Marine Systems* **2**(3-4), 435–450.
- Saloranta, T. M.: 2001, Hydrographic structure of the sea west of Svalbard along and across the continental slope. Reports in Meteorology and Oceanography 2, Geophysical Institute, University of Bergen,
- Saloranta, T. M. and Haugan, P. M.: 2004, Northward cooling and freshening of the warm core of the West Spitsbergen Current, *Polar Research* **23**(1), 79–88.
- Sánchez-Vidal, A., Pasqual, C., Kerhervé, P., Calafat, A., Heussner, S., Palanques, A., de Madron, X. D., Canals, M. and Puig, P.: 2008, Impact of dense shelf water cascading on the transfer of organic matter to the deep western Mediterranean basin, *Geophysical Research Letters* **35**(5), L05605.
- Sannino, G., Sanchez-Garrido, J. C., Liberti, L., Pratt, L., Bellafore, D. and Carniel, S.: 2013, On the relevance of non-hydrostatic modeling in the Gibraltar Strait region, *Geophysical Research Abstracts* **15**, EGU2013–14065. URL: <http://meetingorganizer.copernicus.org/EGU2013/EGU2013-14065.pdf>
- Schauer, U.: 1995, The release of brine-enriched shelf water from Storfjord into the Norwegian Sea, *Journal of Geophysical Research* **100**(C8), 16015–16028.
- Schauer, U. and Fahrbach, E.: 1999, A dense bottom water plume in the western Barents Sea: downstream modification and interannual variability, *Deep Sea Research Part I: Oceanographic Research Papers* **46**(12), 2095–2108.
- Schauer, U., Rudels, B., Fer, I., Haugan, P. M., Skogseth, R., Björk, G. and Winsor, P.: 2003, Return of deep shelf/slope convection in the Western Barents Sea?, *Seventh Conference on Polar Meteorology and Oceanography and Joint Symposium on High-Latitude Climate Variations*, The American Meteorological Society, Hyannis, MA.
- Shapiro, G. I. and Hill, A. E.: 1997, Dynamics of dense water cascades at the shelf edge, *Journal of Physical Oceanography* **27**(11), 2381–2394.

LIST OF REFERENCES.

- Shapiro, G. I. and Hill, A. E.: 2003, The alternative density structures of cold/saltwater pools on a sloping bottom: The role of friction, *Journal of Physical Oceanography* **33**(2), 390–406.
- Shapiro, G. I., Huthnance, J. M. and Ivanov, V. V.: 2003, Dense water cascading off the continental shelf, *Journal of Geophysical Research* **108**(C12), 3390–3409.
- Shapiro, G. I. and Zatsepin, A. G.: 1997, Gravity current down a steeply inclined slope in a rotating fluid, *Annales Geophysicae* **15**(3), 366–374.
- Shapiro, G., Luneva, M., Pickering, J. and Storkey, D.: 2013, The effect of various vertical discretization schemes and horizontal diffusion parameterization on the performance of a 3-D ocean model: the Black Sea case study, *Ocean Science* **9**(2), 377–390.
- Siddorn, J. R. and Furner, R.: 2013, An analytical stretching function that combines the best attributes of geopotential and terrain-following vertical coordinates, *Ocean Modelling* **66**, 1–13.
- Signell, R. P.: 1989, *Tidal dynamics and dispersion around coastal headlands*, Ph.d. thesis, Massachusetts Institute of Technology, MIT/WHOI-89-38, Cambridge, Massachusetts. 162 pp.
- Sikirić, M. D., Janeković, I. and Kuzmić, M.: 2009, A new approach to bathymetry smoothing in sigma-coordinate ocean models, *Ocean Modelling* **29**(2), 128–136.
- Skogseth, R., Fer, I. and Haugan, P. M.: 2005a, Dense-water production and overflow from an Arctic coastal polynya in Storfjorden, in H. Drange, T. Dokken, T. Furevik, R. Gerdes and W. Berger (eds), *The Nordic Seas: An Integrated Perspective. AGU Geophysical Monograph Series 158*, American Geophysical Union, pp. 73–88. URL: <http://www.agu.org/cgi-bin/agubooks?topic=. .GM&book=OSGM1584238&search=>
- Skogseth, R., Haugan, P. M. and Haarpaintner, J.: 2004, Ice and brine production in Storfjorden from four winters of satellite and in situ observations and modeling, *Journal of Geophysical Research* **109**(C10), C10008.
- Skogseth, R., Haugan, P. M. and Jakobsson, M.: 2005b, Watermass transformations in Storfjorden, *Continental Shelf Research* **25**(5-6), 667–695.
- Skogseth, R., Nilsen, F. and Smedsrud, L. H.: 2009, Supercooled water in an Arctic polynya: observations and modeling, *Journal of Glaciology* **55**(189), 43–52.
- Skogseth, R., Sandvik, A. D. and Asplin, L.: 2007, Wind and tidal forcing on the meso-scale circulation in Storfjorden, Svalbard, *Continental Shelf Research* **27**(2), 208–227.

LIST OF REFERENCES.

- Skogseth, R., Smedsrud, L. H., Nilsen, F. and Fer, I.: 2008, Observations of hydrography and downflow of brine-enriched shelf water in the Storfjorden polynya, Svalbard, *Journal of Geophysical Research* **113**(C8), C08049.
- Smagorinsky, J.: 1963, General circulation experiments with the primitive equations: I. The basic experiment, *Monthly Weather Review* **91**(3), 99–164.
- Smethie Jr., W. M., Ostlund, H. G. and Loosli, H. H.: 1986, Ventilation of the deep Greenland and Norwegian Seas: Evidence from Krypton-85, tritium, carbon-14, and argon-39, *Deep Sea Research Part A. Oceanographic Research Papers* **33**(5), 675–703.
- Smith, P. C.: 1975, A streamtube model for bottom boundary currents in the ocean, *Deep Sea Research and Oceanographic Abstracts* **22**(12), 853–873.
- Song, Y. and Haidvogel, D.: 1994, A semi-implicit ocean circulation model using a generalized topography-following coordinate system, *Journal of Computational Physics* **115**(1), 228–244.
- Steele, M. and Boyd, T.: 1998, Retreat of the cold halocline layer in the Arctic Ocean, *Journal of Geophysical Research* **103**(C5), 10419–10435.
- Storkey, D., Blockley, E. W., Furner, R., Guiavarc'h, C., Lea, D., Martin, M. J., Barciela, R. M., Hines, A., Hyder, P. and Siddorn, J. R.: 2010, Forecasting the ocean state using NEMO: The new FOAM system, *Journal of Operational Oceanography* **3**(1), 3–15. URL: <http://www.ingentaconnect.com/content/imarest/joo/2010/00000003/00000001/art00001>
- Sutherland, B. R., Nault, J., Yewchuk, K. and Swaters, G. E.: 2004, Rotating dense currents on a slope. Part 1. Stability, *Journal of Fluid Mechanics* **508**, 241–264.
- Taylor, G. I.: 1953, Dispersion of Soluble Matter in Solvent Flowing Slowly through a Tube, *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* **219**(1137), 186–203.
- Tsubouchi, T., Bacon, S., Naveira Garabato, A. C., Aksenov, Y., Laxon, S. W., Fahrbach, E., Beszczynska-Möller, A., Hansen, E., Lee, C. M. and Ingvaldsen, R. B.: 2012, The Arctic Ocean in summer: A quasi-synoptic inverse estimate of boundary fluxes and water mass transformation, *Journal of Geophysical Research: Oceans* **117**(C1), C01024.
- Turner, J. S.: 1986, Turbulent entrainment: the development of the entrainment assumption, and its application to geophysical flows, *Journal of Fluid Mechanics* **173**, 431–471.

LIST OF REFERENCES.

- Turner, J. S.: 2010, The Melting of Ice in the Arctic Ocean: The Influence of Double-Diffusive Transport of Heat from Below, *Journal of Physical Oceanography* **40**(1), 249–256.
- Umlauf, L.: 2001, *Turbulence Parameterisation in Hydrobiological Models for Natural Waters*, PhD thesis, Fachbereich Mechanik der Technischen Universität Darmstadt.
- Umlauf, L. and Burchard, H.: 2003, A generic length-scale equation for geophysical turbulence models, *Journal of Marine Research* **61**(2), 235–265.
- Umlauf, L. and Burchard, H.: 2005, Second-order turbulence closure models for geophysical boundary layers. A review of recent work, *Continental Shelf Research* **25**(7-8), 795–827.
- Visbeck, M. and Thurnherr, A. M.: 2009, High-resolution velocity and hydrographic observations of the Drygalski Trough gravity plume, *Deep Sea Research Part II: Topical Studies in Oceanography* **56**(13-14), 835–842.
- Wåhlin, A. K. and Walin, G.: 2001, Downward migration of dense bottom currents, *Environmental Fluid Mechanics* **1**(2), 257–279.
- Wakelin, S., Holt, J. T. and Proctor, R.: 2009, The influence of initial conditions and open boundary conditions on shelf circulation in a 3D ocean-shelf model of the North East Atlantic, *Ocean Dynamics* **59**(1), 67–81.
- Warner, J. C., Sherwood, C. R., Arango, H. G. and Signell, R. P.: 2005, Performance of four turbulence closure models implemented using a generic length scale method, *Ocean Modelling* **8**(1-2), 81–113.
- Wells, M. G. and Nadarajah, P.: 2009, The Intrusion Depth of Density Currents Flowing into Stratified Water Bodies, *Journal of Physical Oceanography* **39**(8), 1935–1947.
- Winters, K. B., Lombard, P. N., Riley, J. J. and D’Asaro, E. A.: 1995, Available potential energy and mixing in density-stratified fluids, *Journal of Fluid Mechanics* **289**, 115–128.
- Wirth, A.: 2009, On the basic structure of oceanic gravity currents, *Ocean Dynamics* **59**(4), 551–563.
- Wobus, F., Shapiro, G. I., Huthnance, J. M. and Maqueda, M. A. M.: 2013, The piercing of the Atlantic Layer by an Arctic shelf water cascade in an idealised study inspired by the Storfjorden overflow in Svalbard, *Ocean Modelling* **in press**.
- Wobus, F., Shapiro, G. I., Maqueda, M. A. M. and Huthnance, J. M.: 2011, Numerical simulations of dense water cascading on a steep slope, *Journal of Marine Research* **69**(2-3), 391–415.

LIST OF REFERENCES.

- Wollast, R. and Chou, L.: 2001, The carbon cycle at the ocean margin in the northern Gulf of Biscay, *Deep Sea Research Part II: Topical Studies in Oceanography* **48**(14-15), 3265–3293.
- Zimmerman, J. T. F.: 1981, Dynamics, diffusion and geomorphological significance of tidal residual eddies, *Nature* **290**(5807), 549–555.
- Zimmerman, J. T. F.: 1986, The tidal whirlpool: A review of horizontal dispersion by tidal and residual currents, *Netherlands Journal of Sea Research* **20**(2-3), 133–154.