University of Oxford

D.Phil. Thesis

Evolution of volcanic rifted margins: Origin of seaward dipping reflectors with insights from flexure and 'process–oriented' gravity modelling

Author:

Rebecca L. Morgan

Supervisors:

Prof. A. B. Watts and

Prof. J. A. Cartwright



A thesis submitted for the degree of Doctor of Philosophy

Department of Earth Sciences

St. John's College

Trinity Term 2018

Declaration of Authorship

I, Rebecca L. Morgan, declare that this project, titled 'Evolution of volcanic rifted margins: Origin of seaward dipping reflectors with insights from flexure and 'process–oriented' gravity modelling' and the work presented in it are my own. I hereby certify that the work presented in this project is all my own unaided work and that all material obtained from published or unpublished sources of any type including the internet have been fully acknowledged at the proper point in the text and included in the list of references.

Signed:

Date:

Abstract

Seaward dipping reflectors (SDRs) are ubiquitous features of the offshore regions of volcanic rifted continental margins where they comprise wedge–shaped packages of mainly extrusive lava flows. SDRs are typically observed on seismic reflection profiles; a compilation of which is studied in this thesis. A seismic reflection profile from offshore Greenland is re–processed in order to quantify the impact of semblance analysis on SDR geometry. With constraints on geometry known, this thesis presents a simple model in which SDRs are formed by a succession of dykes which intrude and load the crust. These loads cause a surface flexure, which is subsequently infilled and loaded by volcanic material. A flexural model explains the arcuate shape, limited offlap geometries and down—dip thickening of SDRs as observed in seismic reflection profile data. By comparing observed and calculated dips we are able to constrain the elastic plate model type and the effective elastic thickness of extended lithosphere, T_e. Results suggest a broken plate or significantly weakened continuous plate model is required to produce the characteristic arcuate shape. Decreasing the T_e for successive loads as rifting progresses produces offlap of sub–packages, while increasing the T_e produces onlap. We have verified our results using process–oriented gravity modelling, in which the gravity effect of surface volcanic infill loads is calculated and combined with the gravity effect of buried dyke loads and the gravity effect of their isostatic compensation. Results from a case study interpreting a set of seismic reflection surveys from the Orange Basin, offshore Namibia show good general agreement between observed Airy isostatic anomalies and calculated gravity anomalies with T_e in the range 1–3 km. The steep gradient that is often observed in the Airy isostatic gravity anomaly at rifted margins is therefore a useful proxy for the seaward edge of the dykes that intrude the crust prior to seafloor spreading.

Extended Abstract

This thesis explores the evolution of volcanic rifted margins, particularly focussing on the origin of seaward dipping reflectors (SDRs). First, a compilation of published SDR observations provides important measurements to differentiate between proposed formation mechanisms. As seismic reflection data is the primary data source, the robustness of SDR imaging both between different survey acquisitions, and different processing flows on the same field data, is investigated. Secondly, I develop a dyke–driven flexural model for SDR formation, constrained by both geometric observations from seismic reflection data and an extension of the process–oriented gravity modelling (POGM) methodology.

SDRs are ubiquitous features of the offshore regions of volcanic rifted margins (VRMs), where they comprise wedge–shaped packages of mainly extrusive lava flows. Their origin remains disputed, with some workers suggesting they form by progressive subsidence of extended crust, while others propose they are accommodated within the crust by one or more continent–dipping normal faults. At the present day, ongoing rifting in East Africa is associated with magmatism which may represent the formation of proto–SDRs.

SDRs are often associated with magmatic underplating and onshore eruption of Large Igneous Provinces (LIPs). Whereas the mechanical behaviour of the Earth's crust during non–volcanic rifting is well understood and quantified, the timing and role of magmatism during extension is less well known. For example, the position of the continent–ocean boundary, and the nature of the crust beneath SDRs, remains uncertain on many VRMs.

The characteristic features of SDRs include the arcuate shape and down–dip thickening; another setting in which these features are observed is the 'moats' of volcaniclastic sediments surrounding ocean islands. Further, the progressive offlap of dipping reflectors is also observed in geochemically defined stratigraphy of the

Deccan Traps. In both of these geological settings, these stratigraphic features have been modelled using lithospheric flexure driven by a surface load. Although surface loads are not required in the case of SDRs, previous models for their formation have suggested the existence of sub–vertical feeder dykes beneath the SDR package, which would act as a buried load. In this thesis, a dyke–driven flexural model for the formation of SDRs is used to investigate the strength of crust during rifting.

Constraining such a model requires observations of SDRs and the crust in which underlies them, ideally from multiple data sources. Primarily, seismic reflection data provides a 2D, and in some cases, 3D image of SDR geometry. In some regions, this data is complimented by IODP/DSDP drilling results, which when available confirm the lithology of SDRs as typically basaltic flows interbedded with subaerial sediments. Finally, as both of these data sources are isolated in extent and expensive to acquire, compilations of satellite and ship–track derived potential field measurements provide an independent, global data source.

As the availability of seismic reflection data has rapidly increased in recent decades, so has the number of margins on which SDRs have been identified. A representative global dataset of SDRs is compiled from seismic reflection data and take several key geometric measurements, including dip, vertical thickness and width (perpendicular to margin strike). Measured dips are typically in the range 5–15°, with few measurements greater than 30°. The majority of SDR packages are between 2–6 km thick, with a mean of 3 km, assuming a P wave velocity range of 3–5 $\mathrm{km\,s^{-1}}$. SDR packages on across–strike margin profiles are typically ∼50 km wide, with the narrowest ∼15 km (e.g. Rockall, *Barton and White* (1997a)) and the widest ∼200 km (e.g. Namibia, *Koopmann et al.* (2014)). This is somewhat complicated by the presence on some margins of an 'outer high', which separates SDRs into distinct inner and outer packages. Knowledge of these properties provides important constraints when differentiating between potential formation mechanism for SDRs.

Dip is a defining feature of SDRs, and a key constraint on flexural models. However, when measured directly from seismic reflection data, dip may be dependent on both the acquisition and processing histories. One of the key steps common to all processing workflows is the calculation of stacking velocities, typically using semblance analysis. In order to explore the sensitivity of SDR stratigraphy to variations in these velocities, two distinct stacks of the same seismic reflection line are produced by completing a minimal pre–stack processing flow and using two sets of stacking velocities. These high and low velocity values are chosen by enveloping the broad maxima in coherence over the SDRs during semblance analysis. The distribution of dips and thicknesses of SDRs on both final stacks produced here, and the originally published fully processed profile, were found to be statistically indistinct at or above the 70% confidence level using a Kolmogorov–Smirnov test.

A dataset consisting of 2D seismic reflection surveys from the Orange Basin, Namibia, is interpreted in order to investigate the detailed stratigraphy of SDRs and their crustal setting. Picking the most landward and seaward SDRs on each reflection profile demonstrates that there is little variation in width along strike, averaging 60–65 km. Despite a relatively small line spacing, typically 5–10 km, it is rarely possible to correlate a single reflector across multiple profiles. This suggests that the individual flows of which SDRs are comprised are less than 10 km in lateral extent. A transitional unit which lies between the termination of the SDR package and 'true' normal thickness oceanic crust is mapped and described, with discussion of potential origins for this unusual package. Two profiles spaced only a few kilometres apart, yet differing in acquisition and processing parameters, present an excellent opportunity to investigate the differences in SDR imaging. The overall size of the package and dip of prominent reflectors is unchanged, however, the transitional unit between SDRs and oceanic crust and the Moho are resolved quite differently on the two surveys.

Given the known geometric constraints, a finite–difference flexural model is developed in which SDRs are formed by a succession of dykes which intrude and load

the crust. These loads cause a surface flexure, which is subsequently infilled and loaded by volcanic material. This model, which does not require a bounding normal fault, can explain the arcuate shape, limited offlap geometries and down–dip thickening of SDRs as observed in seismic reflection data. By comparing observed and calculated dips, the model is used to constrain the elastic plate model type and the effective elastic thickness of extended lithosphere, T_e. Results suggest a broken plate or significantly weakened continuous plate model is required to produce the characteristic arcuate shape. Decreasing the T_e for successive loads as rifting progresses produces offlap of sub–packages, while increasing the T_e produces onlap.

These seismic reflection derived geometric results are verified using process–oriented gravity modelling. This methodology has been previously applied to rifted margins with sediment loads and their compensation. Here, POGM is extended to incorporate the gravity effect of surface volcanic infill loads, buried dyke loads and the gravity effect of their isostatic compensation. Results from the case study in the Orange Basin, offshore Namibia, using the results of the seismic interpretation show good general agreement between observed Airy isostatic anomalies and calculated gravity anomalies with T_e in the range 1–3 km.

The steep gradient that is often observed in the Airy isostatic gravity anomaly at rifted margins is therefore a useful proxy for the seaward edge of the dykes that intrude the crust prior to seafloor spreading, rather than a change in basement elevation at the boundary between oceanic and continental crust, as had been proposed by previous workers.

Acknowledgements

Firstly, my thanks to Tony Watts for his guidance, knowledge, enthusiasm and support throughout my D.Phil. and Masters project. I have thoroughly enjoyed my five years in the Marine Lab and the many scientific digressions discussed over coffee. My thanks also to Joe Cartwright for his support both in the field in Namibia and in the lab interpreting seismic data.

Many other researchers have provided particular advice and suggestions related to this work: thanks to Roger Swart, for his in–depth knowledge of Namibian geology; Mike Daly, for wide–ranging discussions of continental margins; Martino Foschi, Michele Paulatto and Claudia Bertoni for their help with seismic interpretation and processing. Particular thanks to Roger Buck and Xiaochuan Tian for many discussions on SDRs and for hosting me during a visit to Lamont–Doherty Earth Observatory. I also thank all others — too many to name — whose discussions at conferences, coffee breaks and field trips have informed this work.

My thesis examiners, Conall Mac Niocaill and Jan Inge Faleide, are thanked for their time and effort in reading my thesis, and for an interesting and insightful discussion in my viva voce.

The data presented in Chapter 4 is courtesy of Spectrum Geo. My thanks for providing this crucial dataset, which formed a key part of the publication *Morgan and Watts* (2018). My thanks to Jenny Collier and two anonymous reviewers for their comments, which helped improve this manuscript.

During my time in Oxford, I have been lucky to share the Marine Lab with fellow D.Phil. students whose discussions have been much appreciated: thanks to Dan, Johnny, Brook T, Lily, Dirk, Brook K, Maria, Harriet and Marta. I would also like to thank the many members of the technical and administrative staff of the Department of Earth Sciences (past and present) for their support over my 8 years in Oxford. In particular, thanks to Elizabeth for helping me find even the most

hard to track down paper, Elaine for her academic support and Steve for keeping the Marine Lab running.

I would like to thank my teachers at Fakenham High School for instilling in me a passion for learning and scientific curiosity. Without their time and energy, I would never have believed I could study at Oxford.

Finally, my eternal gratitude to my family for their love and support over the last few years and beyond, and to Rob, for being by my side throughout. I couldn't have done it without you all.

This report was written using the LaTeX document preparation system. Many of the figures in this thesis were prepared using Generic Mapping Tools (GMT) (*Wessel and Luis*, 2017).

Contents

Declaration of Authorship ii

Abstract iii

Extended Abstract iv

Acknowledgements viii

Contents x

List of Figures xvii

List of Tables xxi

Notation xxiii

1 Introduction 1
- 1.1 Continental Margins . . . 1
  - 1.1.1 Volcanic Rifted Margins . . . 2
  - 1.1.2 'Classic' VRM Architecture . . . 4
  - 1.1.3 Modelling Approaches . . . 5
  - 1.1.4 Thesis Aims . . . 6
- 1.2 Defining and Modelling SDRs . . . 7
  - 1.2.1 Comparing Existing Models for SDR Formation . . . 10
    - 1.2.1.1 Loading and Subsidence (*Mutter et al.*, 1982) . . . 10
    - 1.2.1.2 Faulting (*Gibson and Love*, 1989) . . . 11
    - 1.2.1.3 Comparison to Our Model . . . 13
  - 1.2.2 Subaerial vs Submarine SDRs . . . 13
- 1.3 Present Day Rifting — East African Rift . . . 14
  - 1.3.1 Onshore Analogues . . . 15

2 SDR Observations 17
- 2.1 Quantifying SDR Observations . . . 17
  - 2.1.1 Motivation . . . 17

2.1.2 Available Datasets 18
2.1.2.1 Seismic Reflection 18
2.1.2.2 Seismic Refraction 19
2.1.2.3 Drilling 19
2.1.2.4 Potential Field 19
2.2 Measurements from Seismic Reflection Profile Data 20
2.2.1 Methods 21
2.2.2 History of Acquisition 22
2.2.3 Results 23
2.2.3.1 Dip 23
2.2.3.2 Width 28
2.2.3.3 Thickness 28
2.2.3.4 Along–strike Length 30
2.3 Seismic Uncertainty 31
2.4 Observations from Drilling 31
2.5 Chapter Summary 33

3 Seismic Data Processing 35
3.1 Introduction 35
3.2 Re–processing 2D Seismic Reflection Data 37
3.2.1 The SIGMA Experiment 37
3.2.2 Motivation for Re–Processing EW9607 Transect III 39
3.2.3 Processing Approach 41
3.2.4 Pre–Stack Processing 42
3.2.5 Semblance Analysis 44
3.2.6 Investigating Migration 48
3.2.7 Alternative Processing Methods Investigated 49
3.3 Investigating the Impact of Stacking Velocities on SDR Geometries 50
3.3.1 Results from Transect III 50
3.3.1.1 Comparing High/Low Velocity Geometries 51
3.3.1.2 Statistical Significance of Results 54
3.3.2 Conclusions 55

4 Seismic Interpretation — Namibian Margin 57
4.1 Introduction to the Namibian margin 57
4.2 Comparison of Imaging across Different Surveys 59
4.2.1 Description of Surveys 60
4.2.1.1 ECL-89 60
4.2.1.2 ECL-89-Repro 62
4.2.1.3 SCOB-12 62
4.2.1.4 NAMZA-14 62
4.2.2 Comparison of Reflector Imaging on Adjacent Lines from Different Surveys 63
4.2.2.1 Results 67

4.2.3 Discussion . . . 68
4.3 Identifying SDRs on a Dense 2D Grid . . . 69
4.3.1 Overall SDR Package Geometry . . . 71
4.3.1.1 Discussion . . . 73
4.3.2 Single SDR Continuity . . . 75
4.3.2.1 Discussion . . . 76
4.4 Transition Package . . . 77
4.4.1 Detailed Description . . . 79
4.4.2 Comparison with South Atlantic Margins . . . 80
4.4.2.1 Comparison with Offshore Damara Belt, Northern Namibian Margin . . . 83
4.4.2.2 Comparison with Conjugate Margin — Argentina 84
4.4.3 Discussion . . . 85
4.5 Comparison with Potential Field Data . . . 86
4.6 Chapter Summary . . . 89

5 Flexural Modelling 91
5.1 Theoretical Introduction to Flexure . . . 91
5.1.1 Elastic Thickness in the Oceans and Continents . . . 94
5.1.2 Potential SDR Analogues . . . 96
5.1.2.1 Oceanic Islands . . . 96
5.1.2.2 Large Igneous Povinces . . . 98
5.1.2.3 Discussion . . . 98
5.2 Role of Surface Loading . . . 98
5.3 Dyke–Driven Flexural Model . . . 100
5.3.1 Summary of Existing Models . . . 100
5.3.2 Introduction to Model . . . 101
5.3.2.1 Finite–Difference Method . . . 101
5.3.2.2 Modelling Approach . . . 103
5.3.3 Results . . . 106
5.3.3.1 Impact of Elastic Thickness . . . 106
5.3.3.2 Broken vs. Continuous Plate . . . 107
5.3.3.3 Rotation of Dykes . . . 109
5.3.3.4 'Jumps' and 'Breaks' . . . 110
5.4 Chapter Summary . . . 111

6 Gravity Anomalies and Modelling 113
6.1 Sources of Data . . . 113
6.2 Airy Isostatic Anomaly . . . 114
6.2.1 Method of Calculation . . . 115
6.2.2 Global Map . . . 116
6.3 Introduction to Process–Oriented Gravity Modelling . . . 120
6.3.1 Methods of Calculating Gravity Anomalies . . . 121
6.4 Comparison to Object–Oriented Gravity Modelling . . . 122

6.4.1 Evaluating POGM and OOGM . . . 123
6.5 Ensemble Averaging of Gravity and Magnetics along Margins . . . 123
6.5.1 Method of Calculation . . . 124
6.5.2 Results . . . 125
6.6 Applying POGM to Dykes and SDRs . . . 128
6.6.1 Calculating Gravity in the SDR Model . . . 130
6.6.2 Fitting Modelled Gravity to the Airy Isostatic Anomaly . . . 131
6.7 Gravity for Alternative SDR Formation Models . . . 131
6.8 Chapter Summary . . . 132

7 Namibia Case Study **137**
7.1 Application of Flexural Model . . . 137
7.2 Application of POGM . . . 139
7.3 Discussion . . . 143
7.4 Implications of Model . . . 143
7.4.1 Isostatic Gravity Anomalies . . . 144
7.4.2 Flexure and the Role of Faulting . . . 144
7.4.3 Long–term Strength of Extended Continental Lithosphere . . . 145
7.5 Synthetic Seismic Reflection Data . . . 146
7.6 Rheological Implications of Model Results . . . 147
7.7 Chapter Summary . . . 149

8 Conclusions and Future Work **151**
8.1 Summary of Results . . . 151
8.1.1 SDR Observations . . . 151
8.1.2 Seismic Processing . . . 152
8.1.3 Seismic Interpretation . . . 152
8.1.4 Flexural Model . . . 153
8.1.5 Gravity Anomalies . . . 153
8.1.6 Namibia Case Study . . . 154
8.2 Future Work . . . 155
8.2.1 Applying the Modelling Approach to Other VRMs . . . 155
8.2.2 Expanding PySDR Model Code . . . 156
8.2.3 Acquiring Additional Data . . . 157
8.2.4 What Makes a Margin Magmatic? . . . 158

A Namibia Seismic Data Headers **159**
A.1 ECL-89 . . . 159
A.2 ECL-89-Repro . . . 160
A.3 SCOB-12 . . . 160
A.4 NAMZA-14 . . . 161

B Python Code **163**

B.1 PySDR Module Setup 163
B.2 Flexural and Gravity Modelling 163
B.3 Dip Measurement and Analysis 204
B.4 SEGY Plotting (using ObsPy) 213
B.5 Synthetic Seismic Reflection 217

C Matlab Code **223**
C.1 Isostatic Anomaly 223
C.2 Ensemble Averaging 224
C.3 Functions 232

D Shell Scripts **235**
D.1 Ensemble Averaging 235

E SDR Observations Table **239**

Bibliography **245**

List of Figures

1.1 Global distribution of active and passive margins, distinguishing magma–rich, magma–poor and transform passive 3
1.2 Schematic of typical volcanic rifted margin architecture 4
1.3 Global distribution of SDRs, LIPs and oceanic flood basalts 8
1.4 Schematic drawing of SDR package from *Mutter et al.* (1982) of the Outer Vøring Plateau, demonstrating identification of key SDR features and magnetic profile. 10
1.5 SDR end–member formation model schematic: a) dyke–fed loading and subsidence; b) fault–bound 12
1.6 Schematic interpretation of the volcanic system activated in the Manda–Harraro rift during September 2005, from *Ayele et al.* (2009) 15
1.7 Schematic of SDR package on mid–Norwegian margin, from *Abdelmalak et al.* (2015) 16

2.1 Histogram of acquisition year of seismic reflection data used in this compilation, full details given in Appendix Table E.1 20
2.2 Schematic demonstrating method of dip calculation using a moving tangential average 22
2.3 SDR dip vs. year of acquisition (from Table E.1) 23
2.4 SDR thickness vs. year of acquisition (from Table E.1) 24
2.5 Global histogram of SDR dip measurements, stacked by region . . . 25
2.6 Global SDR dip–depth joint density plot 26
2.7 Histogram of measured SDR package widths from data compilation 28
2.8 Global compilation of SDR thickness measurements 29
2.9 Histogram of a) Overlying sediment thickness and b) Water depth above SDRs measured in compilation 30
2.10 Kudu 9A-2 and 9A-3 stratigraphic correlation and lithology summary 32

3.1 EW9607 cruise track with location of Transect III highlighted . . . 38
3.2 EW9607 Transect III seismic reflection profile with corresponding magnetic anomaly 40
3.3 Radon and FK Filter processing panels 43
3.4 Velocity workflow using semblance analysis in Claritas 45
3.5 EW9607 Transect III re–processed seismic reflection profile using high stacking velocity values (also shown) 46

3.6 EW9607 Transect III re–processed seismic reflection profile using low stacking velocity values (also shown) . . . 47
3.7 EW9607 Transect III difference in high/low envelope of velocities derived from semblance analysis. . . . 48
3.8 Inner section of EW9607 Transect III seismic reflection profile, comparing results of high and low velocities. . . . 51
3.9 Outer section of EW9607 Transect III seismic reflection profile, comparing results of high and low velocities . . . 52
3.10 Outer SDR package dip distribution for high and low stacking velocities . . . 53
3.11 Inner SDR package dip distribution for high and low stacking velocities . . . 53
3.12 Graphical comparison of dip distribution statistical metrics for EW9707 Transect III velocities . . . 55

4.1 South Atlantic bathymetric and isostatic anomaly reconstruction at 120.4 Ma . . . 58
4.2 South Atlantic bathymetry/topography — perspective view . . . 59
4.3 Location of seismic surveys, Orange Basin, Namibia . . . 61
4.4 SCOB-12-ADJ2 seismic reflection profile . . . 64
4.5 NAMZA-14-ADJ1 seismic reflection profile . . . 65
4.6 Annotated comparison of SCOB-12-ADJ2 and NAMZA-14-ADJ1 seismic reflection profiles . . . 66
4.7 Distance between seismic lines NAMZA-14-ADJ1 and SCOB-12-ADJ2 . . . 67
4.8 Typical Orange Basin seismic reflection profile — survey SCOB-12 70
4.9 Outer SDR package line drawing — seismic reflection profile from survey SCOB-12 . . . 71
4.10 Seaward and landward SDRs definition, seismic profile from SCOB-12 survey . . . 73
4.11 Seaward and landward SDRs definition with ‘sag’ basin, seismic profile from SCOB-12 survey . . . 74
4.12 Strike line showing top SDR surface, seismic profile from SCOB-12 survey, Orange Basin, Namibia. . . . 75
4.13 Seaward and landward SDR extent map — Orange Basin, Namibia 76
4.14 Orange Basin, Namibia, SDR package width histogram . . . 77
4.15 Orange Basin, Namibia picked continuous SDRs map . . . 78
4.16 Transition package seismic image . . . 79
4.17 Orange Basin, Namibia transition package thickness map with EMAG2v3 magnetic anomaly . . . 80
4.18 Inner and outer SDR comparison, Orange Basin, Namibia . . . 82
4.19 Orange Basin, Namibia composite seismic reflection profile and isostatic gravity anomaly . . . 88

5.1 Schematic of the Earth’s response to deformation on different timescales 92
5.2 Diagnostic Waveband of Flexure . . . 93

5.3 Global T_e against age of oceanic crust at the time of loading 95
5.4 Annotated Seismic Reflection Profile over Seamount Moat, Oahu, Hawaii 97
5.5 Cone Surface Load Migration and Associated Flexure for Different T_e Values 99
5.6 Full schematic of progression of dyke–driven SDR model 104
5.7 Schematic of dyke–driven flexure model, defining parameters 105
5.8 Pre–rift stratigraphy comparing the end–member cases of constant infill across the flexural model domain 105
5.9 Flexure SDR Model — Examples of Varying Elastic Thickness . . . 107
5.10 Varying Width of Weak Zone in Flexural SDR Model 108
5.11 Varying T_e Gradient in Flexural SDR Model 109
5.12 Jumps and Breaks (Reduced Infill) in Dyke Driven Model 110

6.1 Comparison of the observed, free–air, Bouger and Airy isostatic anomaly over a rifted margin 117
6.2 Global gravity spherical harmonics, degree and order 20 118
6.3 Global Airy Isostatic Anomaly Map 119
6.4 Schematic of POGM Sediment Anomaly Calculation 121
6.5 Orthogonal profiles from the Greenland margin: Isostatic gravity anomaly, bathymetry and magnetic anomaly with calculated ensemble averages. 126
6.6 Map view of gridded ensemble average and residual isostatic gravity anomaly and magnetic anomaly data, Greenland Margin 127
6.7 Orthogonal profiles from the Namibian margin: Isostatic gravity anomaly, bathymetry and magnetic anomaly with calculated ensemble averages. 128
6.8 Map view of gridded ensemble average and residual isostatic gravity anomaly and magnetic anomaly data, Namibian Margin 129
6.9 Process–oriented gravity anomalies for an SDR package, demonstrating impact of reduced infill periods on the sum anomaly (stratigraphy also shown). 134
6.10 Process–oriented gravity anomalies for an SDR package, demonstrating impact of rifting and subsidence on the sum anomaly. . . . 135
6.11 Process–oriented gravity anomalies for an SDR package, comparing shape of sum anomaly with the SDR component of the anomaly alone. 136

7.1 Airy isostatic gravity anomaly and bathymetry of the volcanic rifted margin offshore Namibia 138
7.2 Namibia model fit summary 140
7.3 Process–oriented gravity anomalies for an SDR package, demonstrating impact of varying T_e on the sum anomaly. 141
7.4 Namibia model best–fit parameters 142
7.5 a). Increasing T_e and b). Decreasing T_e with successive loads resulting in onlap and offlap respectively 146

7.6 Velocity model and corresponding synthetic seismogram for final Namibia flexural model 147
7.7 Schematic of Yield Strength Envelope Demonstrating Effective Elastic Thickness 149

List of Tables

2.1 Condensed compilation of mean values concerning SDR geometry by region 27

3.1 Statistical summary of dip distribution comparisons for EW9707 Transect III 55

5.1 Variables for flexure and gravitational derivations and calculations . 92
5.2 Table with standard parameters used in flexural SDR model section 5.3.3, unless otherwise stated. 108

6.1 Summary of the main parameters used in the flexural and gravity modelling in this Chapter. The values used in this table apply to all figures except where explicitly stated, ... 130

7.1 Summary of the main parameters used in the flexural and gravity model applied to the Orange Basin, Namibia. 139

E.1 Compilation of geometric SDR observations from representative global dataset. 244

Notation

G	Gravitational constant
GEBCO	General Bathymetric Chart of the Oceans
POGM	Process–Oriented Gravity Modelling
OOGM	Object–Oriented Gravity Modelling
INSAR	Interferometric Synthetic Aperture Radar
FFT	Fast Fourier Transform
LIP	Large Igneous Province
SDR	Seaward Dipping Reflectors
CDP/CMP	Common Depth/Mid Point
MCS	Multi–Channel Seismic
ODP	Ocean Drilling Program
IODP	International Ocean Discovery Program
DSDP	Deep Sea Drilling Project
TWTT	Two–Way Travel Time
FDM	Finite Difference Method
RMS	Root–Mean Squared
T_e	Elastic Thickness
1D	One–Dimensional
2D	Two–Dimensional
3D	Three–Dimensional

Chapter 1

Introduction

1.1 Continental Margins

The Earth's crust is fundamentally different between the continents and the oceans, in terms of age, average composition and thickness. Oceanic crust is denser and thinner, formed at submarine spreading ridges. In contrast, continental crust is much more variable both in composition and age. We know from plate tectonics that horizontal movements of the Earth's plates creates new oceanic crust during rifting and spreading, and destroys older existing crust at subduction zones on geologic timescales. The Wilson cycle describes this cyclical creation and destruction of oceanic basins as the plates move. A fundamental aspect of this cycle is continental rifting — the breakup of existing continental crust, eventually developing into an oceanic basin with continental margins. Understanding how rifting occurs is fundamental to our knowledge of the Earth's surface; the record of this event is preserved at the transition between the continent and the ocean; the continental margin.

Continental margins, broadly subdivided into active (subduction) and passive (Figure 1.1) margins, are relevant in many different fields of Earth sciences. Sediments accumulated during rifting on passive margins provide important records of climatic change, biodiversity and often host voluminous hydrocarbons. Within passive margins, further subdivision distinguishes volcanic, non–volcanic and transform margins.

1.1.1 Volcanic Rifted Margins

Volcanic rifted margins, which likely comprise a majority of the world's rifted margins, are distinguished by their extrusive volcanism and intrusive magmatism at the time of continental break up (*Hinz*, 1981; *Mutter et al.*, 1982). The principal evidence for volcanism has come from seismic reflection and refraction profile data. Reflection data have revealed Seaward Dipping Reflectors (SDRs) offshore Norway, Greenland, Rockall Plateau, Antarctica, Southwest Africa, Northwest Australia, East Coast USA and India (*Hinz*, 1981). SDRs are characterised by their arcuate shape, down–dip thickening, and offlap patterns and have generally been attributed to sub–aerial volcanism associated with Large Igneous Province (LIP) formation prior to the creation of oceanic crust (*Planke and Eldholm*, 1994). Such an origin has been confirmed by the Ocean Drilling Program (ODP) which has revealed that SDRs comprise numerous lava flows intercalated with volcaniclastic sediments (e.g. Leg 104 (*Scientific Party*, 1987) and Leg 152 (*Teagle and Alt*, 1999). Refraction data have revealed that SDRs are often associated with high P–wave velocity lower crustal bodies, which have been interpreted as either a magmatic underplate or sill–like crustal intrusions (*White et al.*, 2008). SDRs have been described onshore in Iceland (*Bodvarsson and Walker*, 1964) and in Greenland (*Geoffroy et al.*, 1998, 2001) where they are associated with coastal dyke swarms.

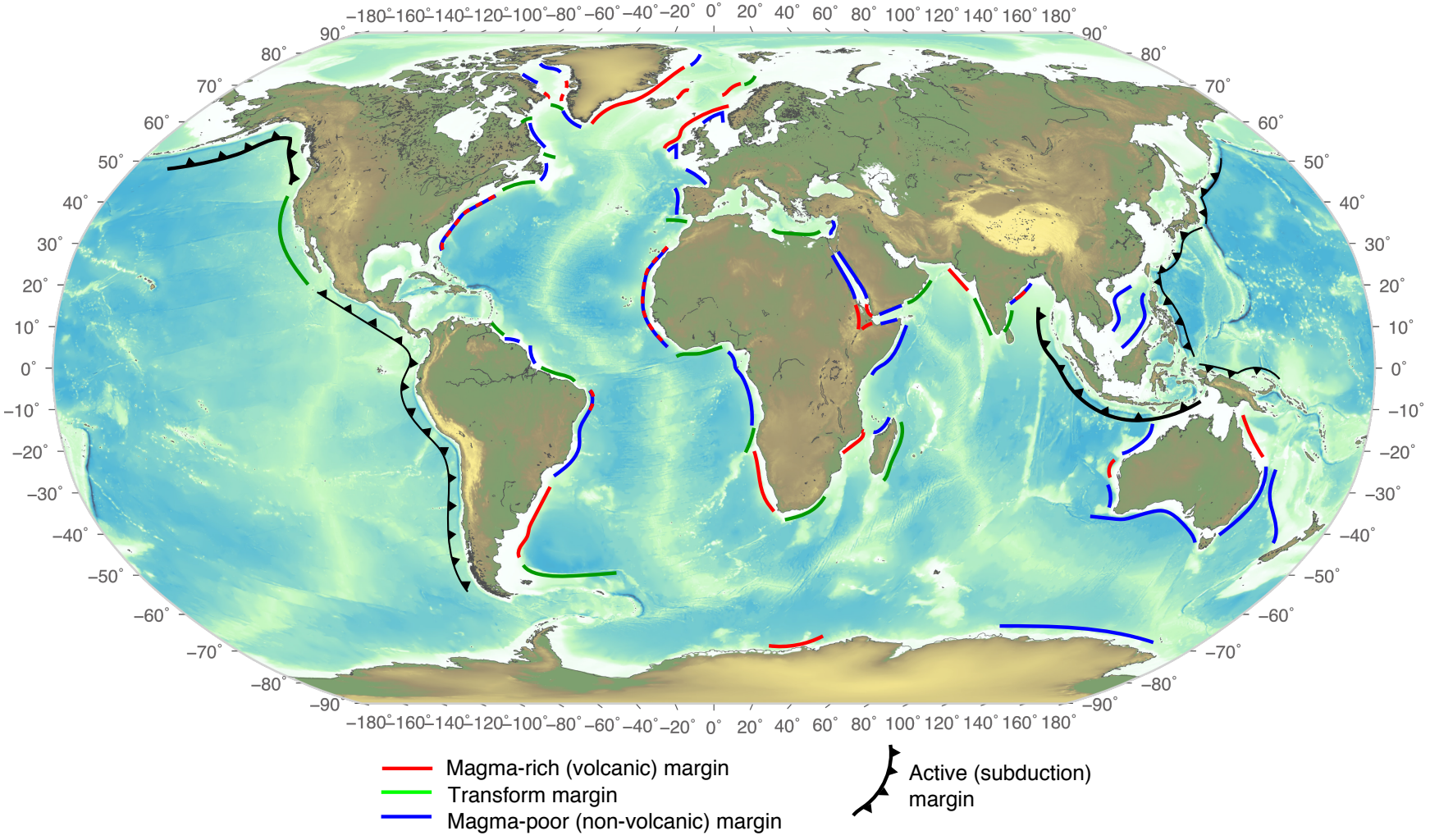


Figure 1.1: Global distribution of active and passive margins, distinguishing magma–rich, magma–poor and transform passive, after *Menzies et al.* (2002); *Haupert et al.* (2016).

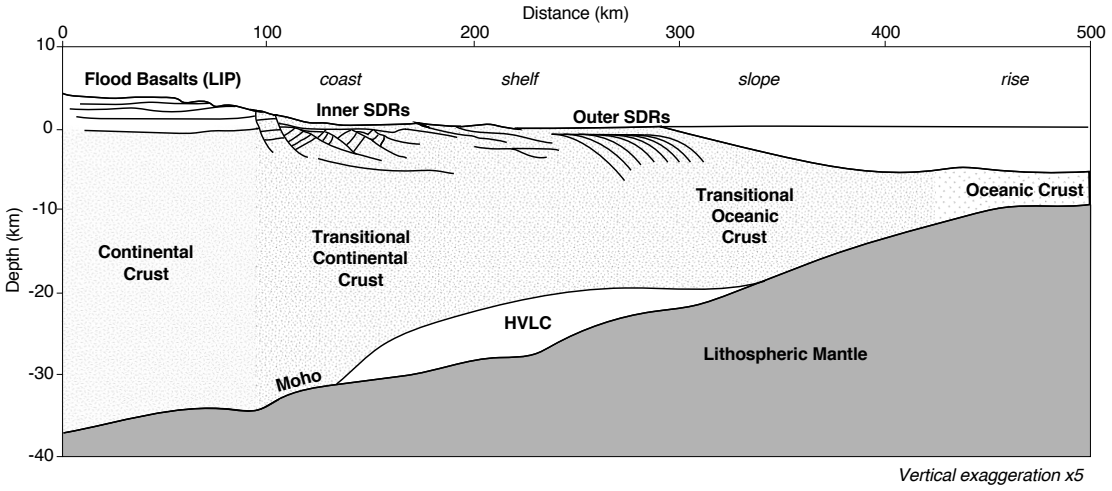


Figure 1.2: Schematic of typical volcanic rifted margin architecture, modified from *Menzies et al.* (2002); *Haupert et al.* (2016). HVLC = High Velocity Lower Crust, often associated with seaward dipping reflectors (SDRs). Horizontal and vertical scales are approximate and shown with 5x vertical exaggeration. Note the distinction between Inner and Outer SDRs and uncertainty of the nature of transitional crust beneath these packages.

Since the discovery of SDRs by *Hinz* (1981), increasing numbers of the Earth's passive margins have been recognised as volcanic, as increased acquisition and imaging quality of seismic reflection data reveals the key distinguishing features of SDRs and often associated intrusions. Why some margins are magmatic while others are not remains uncertain — typically, VRMs are associated in time and space with a proposed mantle plume, although the exact relationship between these two geologic events is heavily debated.

1.1.2 'Classic' VRM Architecture

SDRs are an important part of understanding this picture; they are incredibly voluminous and common to all VRMs (Figure 1.2). They typically lie over crust of uncertain origin, where there is significant thinning and a transition between continental and oceanic crust. Understanding the state of the lithosphere during their formation would provide insight into its strength during this transition.

Arguably, the best–studied volcanic rifted margins are offshore Norway and Greenland in the North Atlantic Ocean, where volcanism has been linked to excess magmatism associated with the Iceland plume (*Parkin and White*, 2008). However, it is not clear whether the flows and intercalated sediments that comprise SDRs are accommodated within the crust by landward dipping border faults (*Gibson and Love*, 1989; *Eldholm et al.*, 1995; *Geoffroy et al.*, 2015) or pass vertically into dykes (*Mutter et al.*, 1982; *Planke and Eldholm*, 1994; *Paton et al.*, 2017) or sills (*White et al.*, 2008). As a result, it is not known whether SDRs reflect extensional features, analogous to the fault–bounded sediment filled syn–rift basins that are accommodated within the crust during rifting, or constructional features formed by dyke and sill loading and flexure prior to, during, or following rifting. Part of the difficulty in constraining SDR formation is seismic imaging, which generally does not have the resolution to resolve steeply dipping structures such as high–angle faults or feeder dykes. Indeed, some reflectors that have been proposed as SDRs in the absence of drilling may not be volcanic, but purely sedimentary in origin.

1.1.3 Modelling Approaches

It has recently been proposed (*Buck*, 2017) that offshore SDRs form by lithospheric flexure in response to dyke and infill loading during the early stages of rifting. In this model, dykes act as a sub–surface or buried load, which creates a surface flexure into which sub–aerial basaltic flows and volcaniclastic sediments accumulate. Since continuous extension allows dykes to be intruded into the crust progressively, the material infilling the surface flexure forms distinct wedge–shape geometries with down–dip thickening, resembling the geometries and internal stratigraphy of flexural moats that flank oceanic island and seamount loads in the interiors of oceanic plates.

While *Buck* (2017) presented examples of seismic reflection profiles of SDRs from rifted margins in the Atlantic and Indian Oceans, he did not quantitatively compare predictions of his flexural model to either seismic or gravity anomaly data.

Seismic reflection profile data, when depth–migrated, provide constraints on the surfaces of flexure and, hence, the long–term strength of extended lithosphere. For example, the widths, thicknesses and internal dips of individual SDR packages can be used to constrain the elastic thickness in a flexural model, and how it varies spatially and temporally. The free–air gravity anomaly is sensitive to both the magnitude of the loads and the deformation that they cause. A dyke that intrudes the crust constitutes a mass excess as it cools and solidifies following its emplacement and would be expected to be associated with a positive gravity anomaly; the dyke–driven surface flexure results in a mass deficiency and the displacement of less dense crustal material into denser mantle, and would be expected to be associated with a negative gravity anomaly. The sum gravity anomaly would comprise a central 'high' over the dyke loads which is flanked by 'lows' over the flexure and therefore the SDRs.

1.1.4 Thesis Aims

The purpose of this thesis is to use seismic reflection profiles, together with gravity anomaly data, to constrain the formation of SDR packages and their mechanism of emplacement. First, a database of seismic reflection data is compiled from which typical SDR properties may be derived. We then investigate how sensitive these measurements are to a standard seismic reflection processing workflow, and in particular semblance analysis. A detailed interpretation of a 2D seismic reflection survey from the Orange Basin, Namibia provides more detailed insights into SDR package geometry, and is used as the test region for the flexural and process–oriented gravity model developed later in the thesis.

We focus here on the outer SDR packages that abut oceanic crust, where dyke emplacement appears to be localised. We use a new finite difference model in which dyke loads cause surface flexures that are infilled by volcaniclastic sediment and basaltic lava flows, which is found to be is generally in accord with seismic and gravity observations. Furthermore, the model provides constraints on the thermal

and mechanical properties, state of stress and long–term strength of extended lithosphere during continental rifting.

1.2 Defining and Modelling SDRs

SDRs were first identified by *Hinz* (1981) on multiple volcanic margins. Since the publication of *Hinz* (1981) SDRs were further identified on a number of global margins (Figure 1.3). Although SDRs have now been recognised for more than 30 years, their method of formation is still heavily debated within the literature. It is likely that the term 'SDRs' encompasses features with quite distinct genetic identities; for example, SDRs are often subdivided into 'inner' and 'outer' packages, where the 'inner' package lies closest to the landward side of the margin. Both are seaward dipping reflectors in the literal sense, but there is no evidence they share a common mechanism of formation. Many features identified on seismic reflection profiles are seaward dipping — taken literally, this would include many sediments found in fault–bound extensional basins. Distinguishing SDRs as they are commonly understood — extrusive arcuate volcanic flows, basaltic in composition with interbedded sediments, is an important step in understanding their role in margin development.

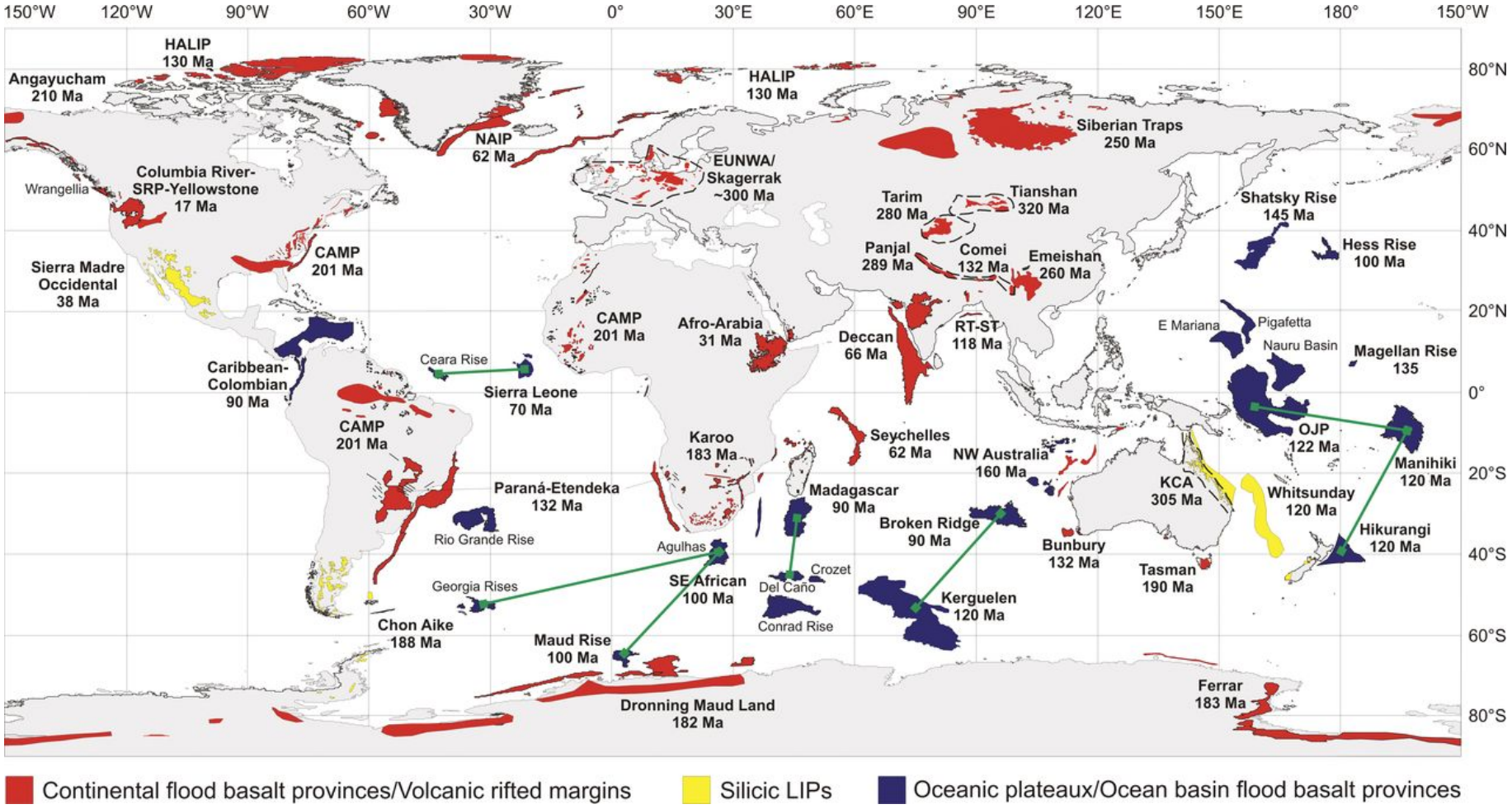


Figure 1.3: Global distribution of SDRs, LIPs and oceanic flood basalts (blue), with approximate ages and common name labelled, from *Self et al.* (2014). The Paranà–Etendeka, recently dated at $134\,\mathrm{Ma}$ (*Dodd et al.*, 2015), and North Atlantic Igneous Province (NAIP, $62\,\mathrm{Ma}$) are discussed in this thesis. Although SDRs are often shown continuously along entire margins (e.g. CAMP and Deccan), the degree of continuity is likely not that well known.

One of the key characteristics of SDRs is their arcuate, convex–up, shape, with dip increasing with depth. This is coupled with down–dip thickening, although this feature is also seen in sedimentary deposits in syn–depositional fault–bound basins. This leads to one of the end–member models proposed for their formation: a listric, continentward–dipping, normal fault e.g. *Gibson and Love* (1989). Mechanically this is challenging — particularly in margins with multiple 'trains' of SDRs. This would require multiple faults, likely shallowing into a single detachment at depth. At the other extreme, with no fault involvement, is the proposal that SDRs form due to loading of sequential flows during extrusion and spreading at an incipient spreading centre e.g. *Mutter et al.* (1982). In this model, each subsequent flow acts as a load on all those below it, causing subsidence which is maximised closest to the spreading axis where loading is thickest, leading to the arcuate shape. It is likely that both of these mechanisms contribute to the production of seaward dipping geometries in different circumstances — distinguishing the subsets of SDRs and quantifying the strength of the crust in each case is important in understanding the temporal and spatial development of continental rifting.

This thesis develops a flexural model for the formation of SDRs. Crucially, in order to produce the amplitude of subsidence necessary with reasonable inputs, the SDRs are not just loaded by subsequent flows, but also by the dykes beneath them. These feeder dykes act as a buried load, denser than the surrounding crust, effectively 'pulling down' the crust and creating the accommodation space for further volcanic flows. Flexure accounts for both the characteristic arcuate shape and down–dip thickening, which no other model is able to satisfactorily do. This mechanism requires active extension and extrusion of volcanic flows, and is therefore only applicable to SDRs found at the seaward edge of margins, commonly termed the 'outer' package. This is also where the arcuate shape is more commonly observed; 'inner' packages often have more linear geometries that are more readily explained by a fault–bound model.

When using a flexural model, it is possible to constrain the elastic thickness (T_e) of the crust at the time of loading. Other input parameters include the size and density of the load and surrounding crust; these can be well constrained using drilling

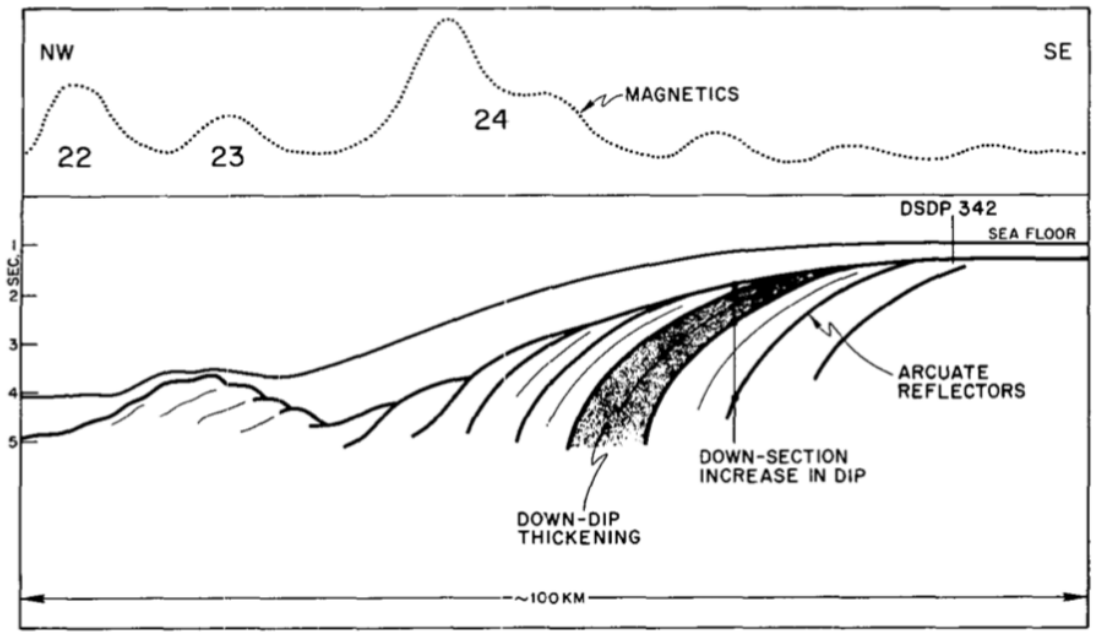


Figure 1.4: Schematic drawing of SDR package from *Mutter et al.* (1982) of the Outer Vøring Plateau, demonstrating identification of key SDR features and magnetic profile. The arcuate shape, down–dip thickening and down–section increase in dip have been recognised since the earliest publications on the topic. Note the thinning of the package seaward.

and seismic observations. The T_e provides a measure of integrated strength, although it is not directly comparable to any physical boundary in the crust. In the oceans, it typically follows the 450 °C isotherm (*Watts*, 1978). Whether in this transitional phase from continent to oceanic crust this remains true is an open question. Similarly, changes in T_e through time during rifting would be reflected in changes in the geometry of SDRs.

1.2.1 Comparing Existing Models for SDR Formation

1.2.1.1 Loading and Subsidence (*Mutter et al.*, 1982)

Following *Hinz* (1981), *Mutter et al.* (1982) also identified SDRs offshore the Norwegian margin. These reflectors are defined as lying within the upper oceanic

crust, rather than lying within a deformed (or otherwise) continental crust. *Mutter et al.* (1982) identify three key features in their schematic (shown in Figure 1.4): arcuate reflectors, down–section increase in dip and down–dip thickening.

Their proposed 2D model (shown in Figure 1.5a) begins with a sub–aerial spreading axis, extruding flows over long distances in both directions. As this process continues, the oldest flows and feeder dykes are depressed, or loaded, by those above, leading to the arcuate shape. This process is essentially equivalent to sub-aerial creation of oceanic crust; however, unlike in the submarine environment, lavas are able to travel great distances. After several million years, *Mutter et al.* (1982) envisage the spreading centre subsiding to submarine depths, which results in the shortening of flow lengths as the area transitions into 'normal' seafloor spreading. This is coupled with a decrease in extrusion rate. The authors note that although the reflectors bear resemblance to deltaic sequences, the down–dip thickening is typically not observed in this setting unless accompanied by significant syn–depositional faulting.

The concept of stacking flows which overlap and depress others is not unique to this setting: *Bodvarsson and Walker* (1964) identified arcuate flows in Iceland to which they attributed a similar process. In that case, they identified dyke injection as the main driver of crustal extension, with normal faulting account for only 10% of that due to dykes in the same section. The lava flow length in Iceland is proposed by *Bodvarsson and Walker* (1964) to be of the order of 10 km; this would be of a similar magnitude to individual reflectors typically observed in SDRs.

1.2.1.2 Faulting (*Gibson and Love*, 1989)

Alternatively, it has been proposed that the rotation of horizons characteristic of SDRs is produced by faulting (see Figure 1.5b). *Gibson and Love* (1989) model the development of SDRs from the Vøring plateau, in which a listric normal faults rotates pre–existing lavas, forming a synclinal lava pile. These lavas are fed by dykes which themselves are intruded along one of the faults during extensional

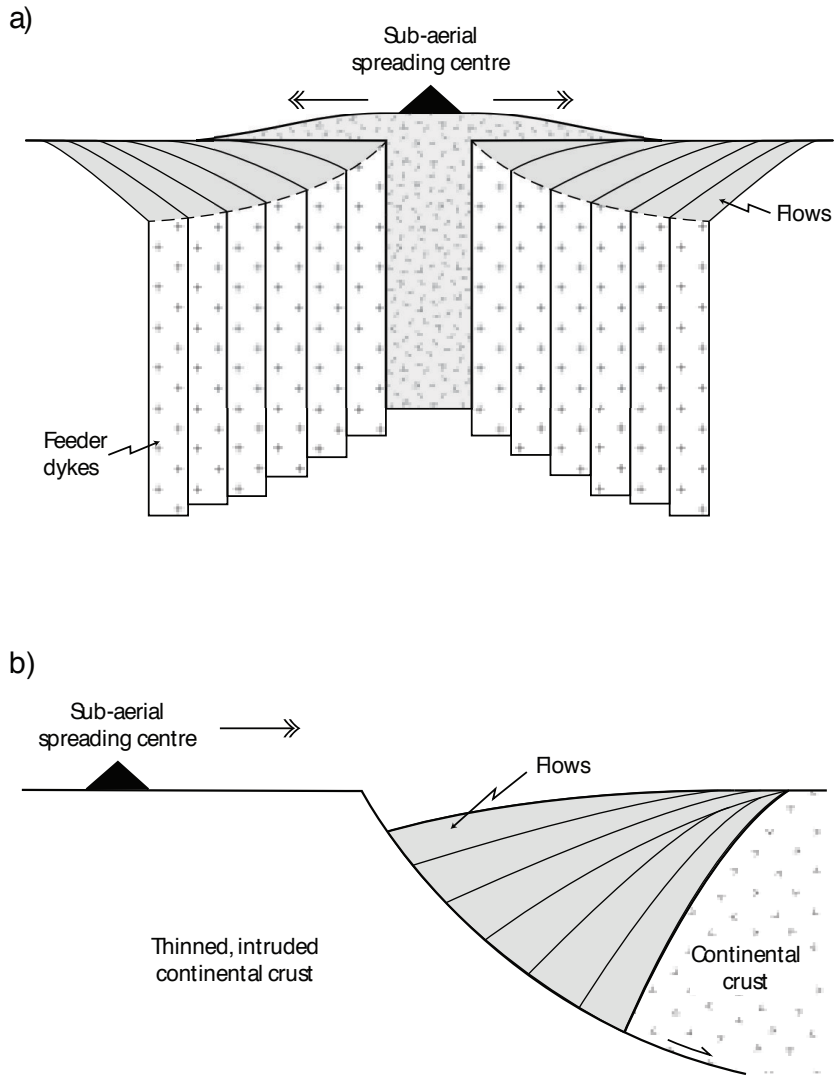


Figure 1.5: SDR end–member formation model schematic: a) dyke–fed loading and subsidence; b) fault–bound. Note the implied difference in crustal origin in the two models — in the loading and subsidence model, SDRs lie above magmatic crust, while in the fault–bound model SDRs lie above existing (perhaps thinned and intruded) continental crust.

deformation. The listric fault is near vertical at the surface, but shallows into a listric detachment at depth, and is assumed to be active during eruption.

A study on the Rockall margin by *Barton and White* (1997a) suggested a fault–bound, landward SDR package emplaced above stretched and intruded continental crust. Seaward of this package, an outer high separates the inner and outer SDR packages, where the outer set lie above transitional crust. It is therefore likely

that both of these end–member models apply in different geological contexts.

1.2.1.3 Comparison to Our Model

The model proposed in this thesis is most similar to the model of *Mutter et al.* (1982). The system is driven by the load of the feeder dykes, which create accommodation space (defined as the depression below a zero height horizontal surface) for lavas to flow into. This differs slightly from *Mutter et al.* (1982), who envisage flows building an edifice before subsiding. Subsequent flows acting as a load on earlier flows and dykes is fully captured in the flexural model developed. By using a numerical flexural model, we are able to quantify the strength of the crust during loading, and better understand how the formation of SDRs is related to the overall rifting process.

1.2.2 Subaerial vs Submarine SDRs

SDRs have been frequently drilled by DSDP/ODP/IODP legs, which have confirmed their subaerial, basaltic origin. In Greenland, the SIGMA project transect III, which is processed and discussed in this thesis, the SDRs drilled by Leg 152 are picrites followed by a uniform sequence of depleted Icelandic tholeiites (*Saunders et al.*, 1998). Five legs drilled mafic basement, and found oxidised flow tops and paleosols within the basalts, supporting their subaerial origin. Further, seismic reflection observations on the East Greenland margin show that the bedding within the volcanic flow units is truncated, suggesting they were eroded and hence possibly emplaced in an elevated setting, significantly above sea level.

However, SDRs may not be formed exclusively subaerially. Further seaward on the SIGMA III transect, a second package lies beyond a hummocky outer high. *Hopper et al.* (2003) propose that this sequence marks the transition from subaerial, through shallow, to deep submarine extension. With a large enough supply of magma to overflow seafloor topography, significant flow lengths of basalt can form in deep marine settings — on the order of 1 km (*Gregg and Fornari*, 1998). This

places constraints on the rate of subsidence relative to spreading rates, if a clear transition to submarine SDR formation can be identified. This raises the question of how subsidence progresses with extension on the many margins in which an outer high is absent — does magma supply wane, such that SDR formation ceases before submarine extension begins?

Although an outer high is often absent, many margins have a thin, chaotic package at the tail–end of the SDRs, before the transition to full seafloor spreading and oceanic crust production (see Chapter 4). Perhaps this package marks a similar transition, from subaerial to submarine extension, without a large enough magma supply to support formation of large outflow lengths in a submarine setting. This could relate to the timing of plume impingement on the margin, as many VRMs are associated with upwelling.

1.3 Present Day Rifting — East African Rift

Studying spreading centres is challenging, as there are few active subaerial regions. Iceland and the East African Rift are the most commonly studied, although observations from Iceland are difficult due to interactions with glacial landforms (*Ayele et al.*, 2009). The Afar depression is an area of active extension and volcanism that lies at the triple junction between the Red Sea, the Gulf of Aden and the East African Rift. A number of studies have been carried out in this region using both satellite and seismicity observations.

Ayele et al. (2009) investigated the 2005 mega–dyke emplacement event in the Manda–Harraro nascent oceanic rift. They found that in a single event, an 8 m wide, ∼60 km long and ∼10 km deep dyke was intruded along a segment of this rift. INSAR–derived subsidence patterns and the location of seismicity suggested that at one end of the rift, the dykes were fed by two shallow magma chambers, around 3–5 km in depth. However, the larger lateral extent of dyking under the Ado'Ale Volcanic Complex did not have any associated subsidence, and the authors speculate the source was at least 10 km deep (Figure 1.6).

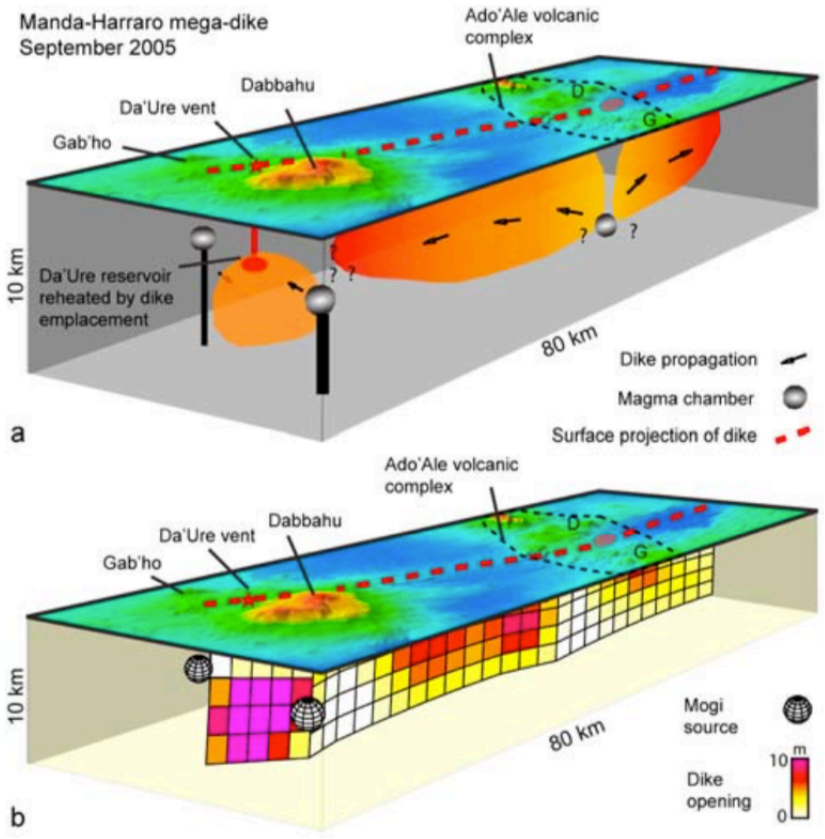


Figure 1.6: Schematic interpretation of the volcanic system activated in the Manda–Harraro rift during September 2005, from *Ayele et al.* (2009). Note the scale of the 'mega–dyke', extending $80\,\mathrm{km}$ in length and with a magma source positioned approximately $10\,\mathrm{km}$ in depth, linking the volcanic complexes.

1.3.1 Onshore Analogues

Understanding the nature of the transitional crust beneath SDRs is one of the most important challenges, and as yet impossible to sample directly. *Abdelmalak et al.* (2015) identify an onshore analogue in the Seve Nappe Complex of the Scandinavian Caledonides in Sarek, Sweden. Multiple generations of mafic dykes are observed with cross–cutting relationships (see schematic in Figure 1.7). This demonstrates that the first generation of dykes are rotated relative to later generations; this observation would be common to both end–member SDR formation models.

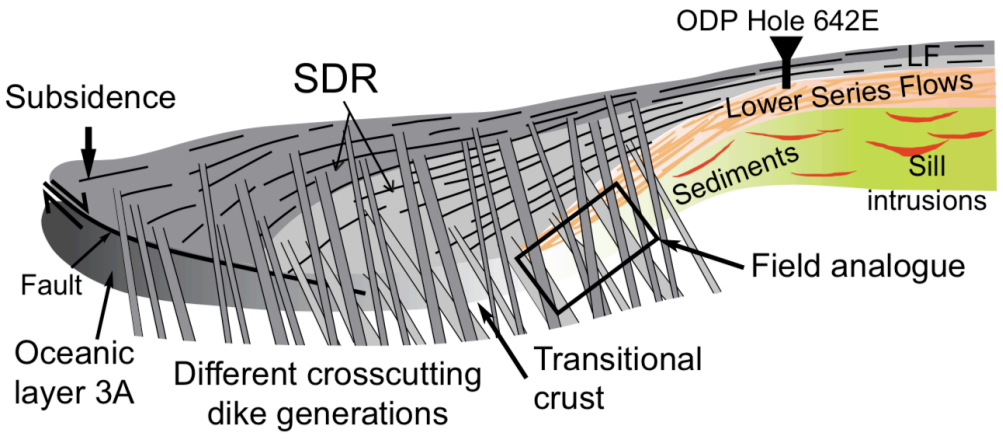


Figure 1.7: Schematic of SDR package on mid–Norwegian margin, from *Abdelmalak et al.* (2015). This transitional crust is compared to a subaerial analogue in the Seve Nappe Complex of the Scandinavian Caledonides. Note the rotation of SDR feeder dykes, demonstrated by cross–cutting relationships of different generations. Here, the seaward termination of the SDRs is fault bound and lies above oceanic layer 3A.

Chapter 2

SDR Observations

2.1 Quantifying SDR Observations

In order to quantitatively evaluate any proposed model for SDR formation, we compile a dataset of observations which capture the stratigraphic and geometric variation observed in SDR packages globally. This dataset, and statistical properties derived from it, will be used to constrain modelling efforts (Chapter 5) along side potential field data (Chapter 6).

2.1.1 Motivation

Understanding how consistent the geometry (e.g. dip, width and thickness) of SDR packages is across and along margins could inform the choice of parameters in formation models. Over the past 30 years, a large amount of seismic reflection and refraction data has been collected, processed and published; in this Chapter, we seek to build a representative compilation of this available data, including examples from the major known volcanic rifted margins.

As SDRs are increasingly recognised on more rifted margins, and generally assumed to share a common formation mechanism, understanding the variation (or lack thereof) in their structure will improve our understanding of the role of volcanism in margin formation. Further, comparing standardised measurements across

conjugate margins could provide insights into the degree of symmetry during continental rifting. For example, understanding whether SDR dips have a very restricted or wide range, and how that range varies across different margins, could aid in discriminating between formation models.

2.1.2 Available Datasets

2.1.2.1 Seismic Reflection

Seismic reflection data provide a unique opportunity to understand the 2D and 3D geometry of SDRs. As the availability of 2D data is much greater in the published record, this dataset will contain key metrics derived from 2D seismic reflection profiles only.

The distinguishing feature of an SDR is the dip — to measure this, we use a moving average approach along the length of the reflector in order to capture the changes in dip with depth. For a 'classic' arcuate SDR, we expect dip to increase with depth. Other important measurements for comparing SDR packages both along the same margin and across conjugate margins is their overall thickness (in depth), width (perpendicular to margin strike) and length (parallel to margin strike).

The along–strike length is the most difficult attribute to accurately measure; survey spacing in 2D datasets is often too wide along–strike to determine whether a package is truly continuous along strike. Similarly, the spacing of the survey typically limits the degree to which any single reflector can be mapped laterally to several kilometres maximum (Chapter 4). Whether this is a true reflection of the extent of a single horizon within the SDR package, or a limit of the dataset, is difficult to ascertain without access to multiple 3D surveys over regions known to have SDRs.

2.1.2.2 Seismic Refraction

Less common than seismic reflection, seismic refraction experiments provide additional information about the velocity structure above and within the SDR package (*Barton and White*, 1997b). In this Chapter, we use the results of refraction over SDR packages to constrain the range of velocities with which we approximate the conversion from time to depth of non–depth migrated reflection profiles. This constitutes the majority of data included in the compilation (Appendix E).

In addition to converting data from the time to depth domain, velocity information also provides information about the possible lithological composition of SDR packages.

2.1.2.3 Drilling

Seafloor drilling experiments rarely drill deep enough to penetrate SDR packages; typically SDRs lie in significant water depth and below thick post–rift sediments (Chapter 4). However, there are several key examples of drilling experiments which have penetrated the 'feather–edge' of the SDR package at the landward end, notably including the Kudu 9A series in Namibia (*McLachlan*, 1990).

Identification of the lithology in these drillholes confirms the basaltic nature of SDRs and also provides information about the depositional setting, with underlying and overlying sediments. In the case of landward SDRs, this is typically subaerial (*McLachlan*, 1990). In this Chapter, drilling results provide another proxy for velocities used in approximating depth conversion.

2.1.2.4 Potential Field

In the absence of targeted seismic reflection/refraction data or direct drilling, potential field data can be a useful proxy for submarine structure. Global datasets compiled from ship–track and satellite data allows comparison of VRMs even without access to seismic reflection data. However, interpretation of potential field data

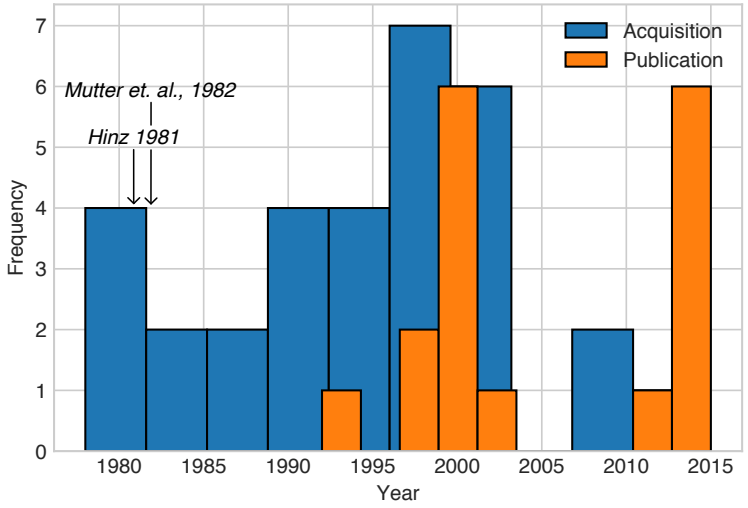


Figure 2.1: Histogram of acquisition year of seismic reflection data used in this ocmpilation, full details given in Appendix Table E.1. Key papers in the initial identification of SDRs annotated (*Hinz*, 1981; *Mutter et al.*, 1982). Where acquisition year is not known, earliest publication year is provided as a proxy.

is inherently non–unique. We do not use gravity or magnetic data in this compilation, but gravity anomalies will be used extensively in Chapter 7 to constrain results of the flexural model.

2.2 Measurements from Seismic Reflection Profile Data

We have compiled a global dataset of multichannel seismic reflection profile images and used it to determine the dips of prominent reflectors in offshore SDR packages at volcanic rifted margins, as well as the approximate widths and thicknesses (Appendix Table E.1; summarised in Table 2.1). The age of acquisition of the seismic images (where known) ranges from 1978 to 2015 (Fig. 2.1).

2.2.1 Methods

We use the Two–Way Travel Time (TWTT) to convert individual reflectors to depth using a suite of 1D profiles with assumed velocities between 3–5 $\mathrm{km\,s^{-1}}$. Such a range includes the velocities derived from observations such as Vertical Seismic Profiles (VSPs) at ODP boreholes in the feather edge of SDR packages at Site 642 (*Planke and Eldholm*, 1994), sonobuoy data from the South Atlantic (*Rabinowitz*, 1976), and wide–angle seismic refraction data from the Edoras Bank (*Barton and White*, 1997a).

We use a windowed moving average of tangential dip measurements along the length of an individual reflector to quantify the range of dips of each reflector, as the typical arcuate shape is not fully captured by a single measurement (Figure 2.2). Firstly, the reflector is picked and the x, y values exported. These values are interpolated onto a regularly spaced grid before windowing. If the reflector is long enough in the y dimension, in time of depth, the window is defined as 0.16 s or 0.25 km respectively. If the reflector is too short for this window, the window is defined as one–fifth of the reflector thickness. For the majority of data which is *not* depth migrated, the interpolated coordinates are then converted to depth using a suite of 1D velocity profiles, resulting in a range of potential geometry measurements for each reflector. Finally, now all data is in depth and the window is known, the moving average dip values are calculated.

The full range of potential dips and thicknesses is captured by calculating the depths for all possible velocities within this range. Although an increase in velocity with depth *within* the SDR package is plausible, this would not affect the range of results presented here. It is possible that the distribution of observed dips may be different if a 2D velocity profile had been used. However, these packages lie under varying thicknesses of sediments and water depths, and the packages themselves are of various thicknesses. We therefore conclude that approximate nature of this conversion, in the absence of pre–stack field data, supports the use of a 1D velocity profile.

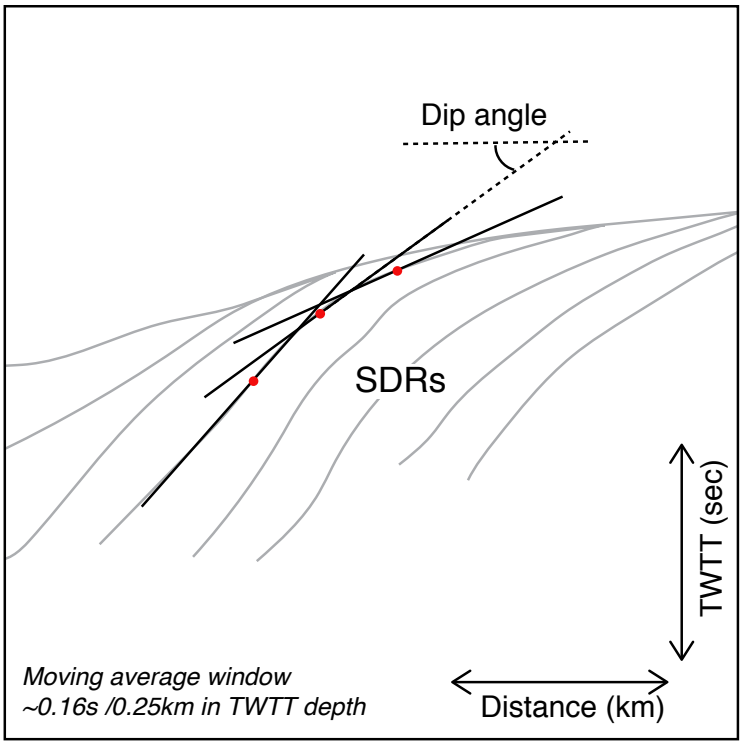


Figure 2.2: Schematic demonstrating method of dip calculation using a moving tangential average. Black solid lines show tangents to the SDR at the corresponding red circle contact points. The window used to calculate the moving average is approximately $0.16\,\mathrm{s}$ in TWTT or $0.25\,\mathrm{km}$ in depth, if the data has been depth migrated.

2.2.2 History of Acquisition

We recognise that acquisition systems have improved significantly during this time. Early systems were based on relatively short analogue streamers and untuned air-gun arrays while more recent systems were based on relatively long ($>5\,\mathrm{km}$) digital streamers and tuned arrays. The depth–migrated seismic images acquired by commercial companies such as ION Geophysical (*Paton et al.*, 2017) have significantly enhanced resolution of Moho, as well as SDRs. The large majority of seismic images in our global database were acquired by academic institutions and have not been depth–migrated.

Importantly, we find no correlation between the year of acquisition and measured dip or thickness (Figures 2.3 and 2.4), despite significant improvements that have occurred in acquisition as well as processing techniques since the late 1970's. We

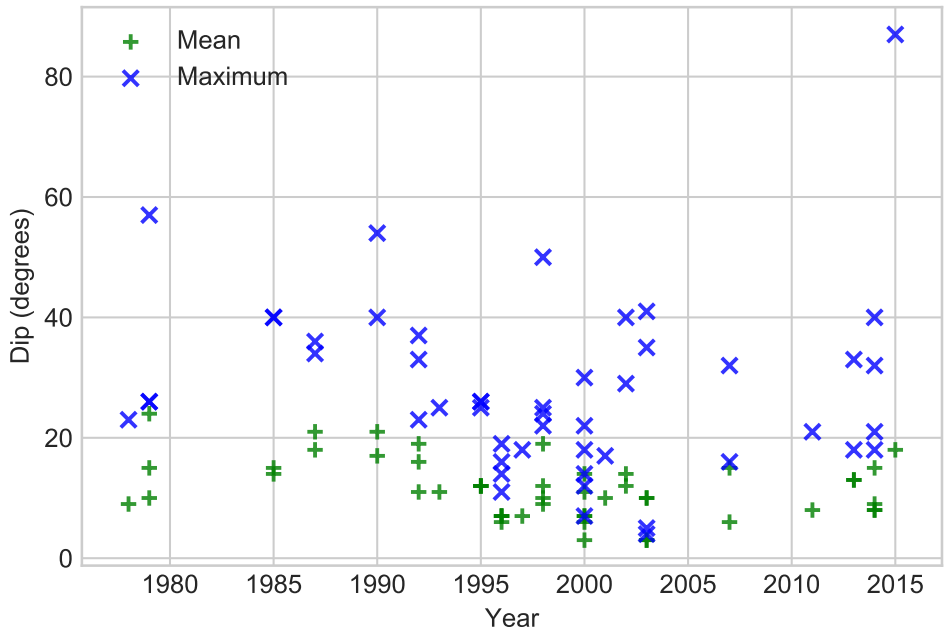


Figure 2.3: Mean (green +) and maximum (blue x) dip against year of acquisition (where available; if unavailable, earliest publication) from Appendix Table E.1.

investigated this relationship as one may expect to find increased dip and thickness values resolved at the base of the SDR package with longer streamers and a better controlled source. We therefore do not weight or modify our data according to acquisition year (Figure 2.1), as the lack of correlation rules out any potential sampling bias.

2.2.3 Results

2.2.3.1 Dip

A histogram of all measured dips, grouped by region, is shown in Figure 2.5, while Figure 2.6 aggregates this data and shows the relationship between dip and relative depth within the SDR package. These figures demonstrate that measured dips are typically in the range 5–15°, with few measurements greater than 30°.

The mean dip of SDRs — remarkably consistent across all margins — is between 5–10°. In comparison to the 'moats' of basalts and volcaniclastic sediments which

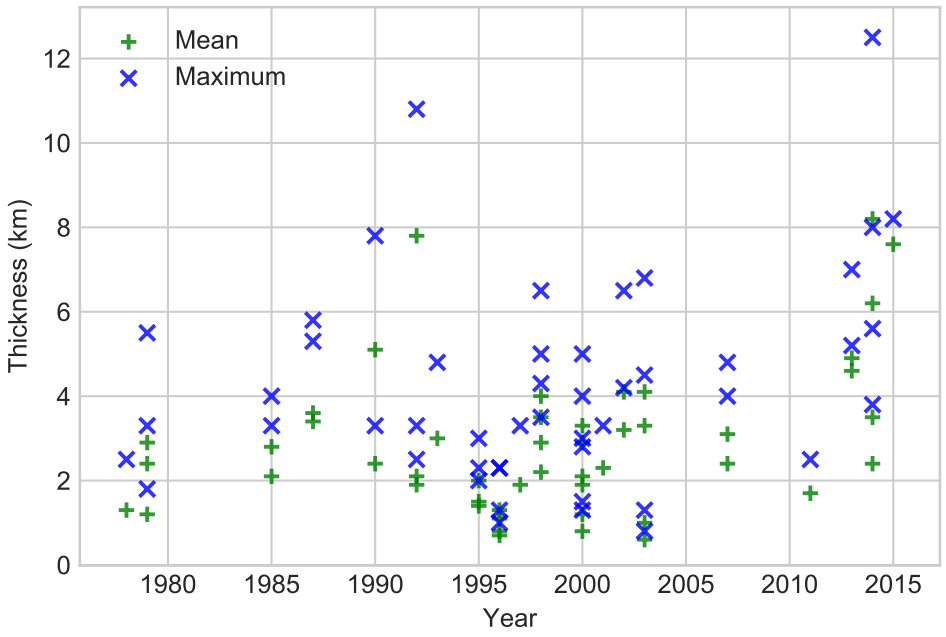


Figure 2.4: Mean (green +) and maximum (blue x) thickness against year of acquisition (where available; if unavailable, earliest publication) from Appendix Table E.1.

flank volcanic islands, this number is slightly larger (Hawaiian moat approx. 2° (*ten Brink and Watts*, 1985)). It is important to note that these figures may at first appear small, given the significant vertical exaggeration commonly used for displaying seismic reflection profiles.

Chapter 5 will explore a flexural model for the formation of SDRs, which can be readily compared to seamount loading models: although many factors contribute to the magnitude of dip (assuming it is flexural in origin) it is likely that the older oceanic crust on which the Hawaiian islands are emplaced is much stronger than the transitional crust onto which SDRs are deposited, reducing the amount of flexure and therefore rotation.

The maximum dip is much more variable; this is likely due to two factors — true geological variability and increased difficulty in imaging the base of the SDR sequence at steeper dips. The quality of seismic imaging at the base of SDR packages through basaltic flows and intercalated sediments is poor, and at greater

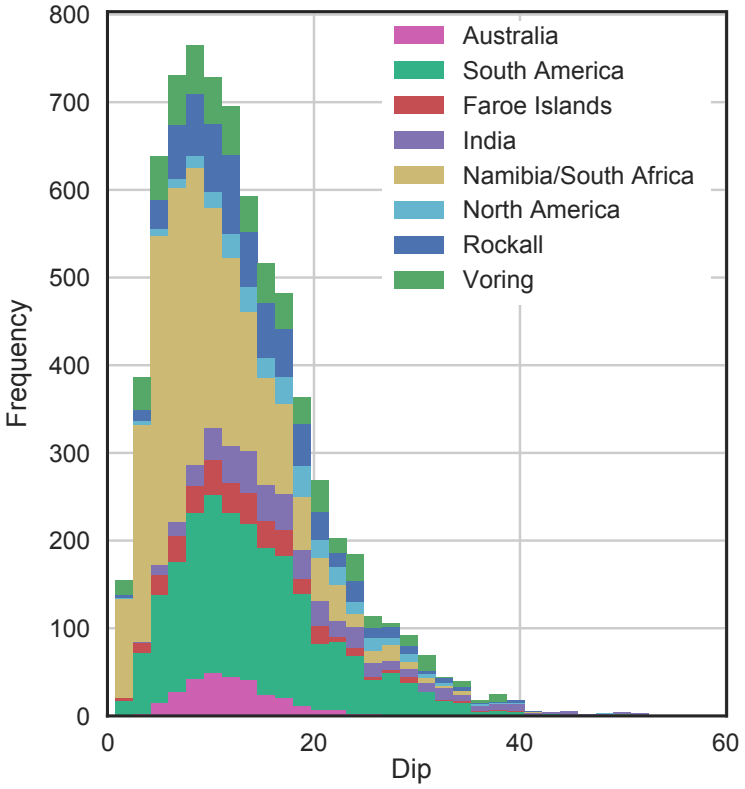


Figure 2.5: Global histogram of SDR dip measurements, with stacked bars representing different regions of VRMs. A single profile with dip values $>60°$ omitted for clarity (see Appendix Table E.1, from *Horn* (2015)).

angles is much poorer than at the top of the package. The steeper dips at greater relative depths are therefore more difficult to accurately constrain.

Another difficulty is profile orientation, which may result in measurement of apparent rather than true dip. However, this is difficult to correct for in the absence of 3D seismic data.

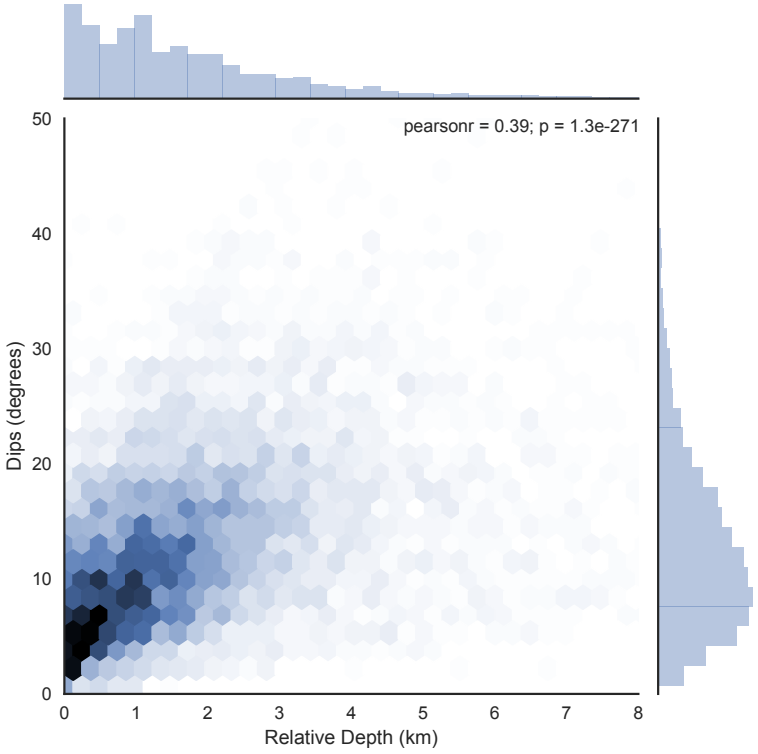


Figure 2.6: Global SDR dip–depth joint density plot, with corresponding histograms shown on each axis. Intensity of colour in joint plot indicates number of measurements in this hex. Plot range limited for clarity excluding some extreme values, see Appendix Table E.1.

Table 2.1: Condensed compilation of mean values concerning SDR geometry by region. See supplementary information Table E.1 for the full dataset with individual records per citation. **N.B.** One profile with four reflectors from the East Coast US was included in the global compilation, but not in this table, due to the small sample size. Approximate water depth and overlying sediment thickness are included as a proxy for amount of stretching. When SDRs are separated into inner and outer packages, the width is measured for each of those packages individually.

Region	**# Reflectors**	**Package Width** (km)	**Approx. Water Depth** (km)	**Approx. Overlying Sed. TWTT** (s)	**Dip** (°)	**Std. Dev. Dip**	**Thickness** (km)	**Std. Dev. Thickness**
SW Africa	37	74	2.1	2.0	8	3.7	3.0	0.8
Australia	12	25	3.9	0.6	12	4.5	1.7	0.4
India	14	21	2.1	1.0	16	6.5	2.1	0.7
NE Atlantic	37	34	2.2	0.8	12	5.2	2.7	0.7
S. America	51	60	2.8	2.5	14	5.7	3.6	0.9
Global	155	53	2.4	1.7	12	5.1	3.0	0.8

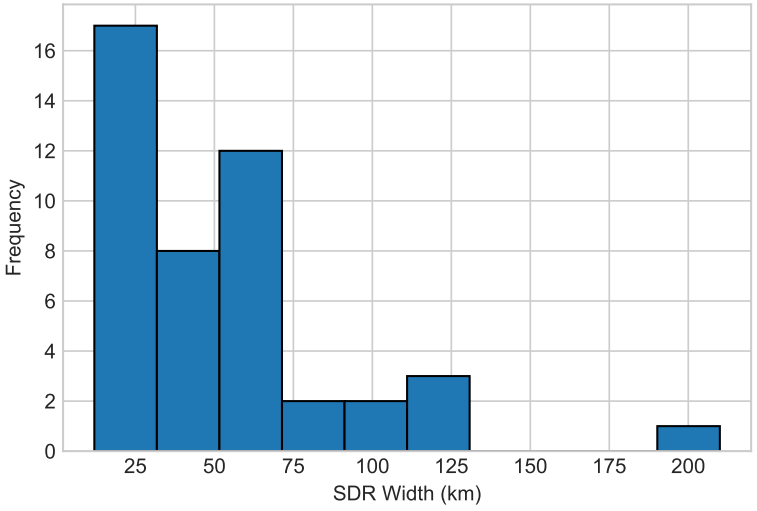


Figure 2.7: Histogram of measured SDR package widths from data compilation. Note extreme value of 200 km from *Koopmann et al.* (2014) (see Appendix Table E.1).

2.2.3.2 Width

We measure the width as the total distance between the landward and seaward limits of the SDR, perpendicular to strike, on each seismic reflection profile. A histogram compiling the global results is shown in Figure 2.7, with a majority of measurements <75 km. SDR packages on across–strike margin profiles are typically ∼50 km wide, with the narrowest ∼15 km (e.g. Rockall, *Barton and White* (1997a)) and the widest ∼200 km (e.g. Namibia, *Koopmann et al.* (2014).

On some margins, offshore SDR packages are divided into an 'inner' and an 'outer' package, separated by an outer high (e.g. Norway — *Planke et al.* (2000); Namibia — *Bauer et al.* (2000); Argentina — *Franke et al.* (2010)). In these cases, the total width may be significantly greater.

2.2.3.3 Thickness

The majority of SDR packages are 1–6 km thick, with a mean of 3 km, assuming a P wave velocity range of 3–5 $km\,s^{-1}$, as shown in Figure 2.8. Only a handful

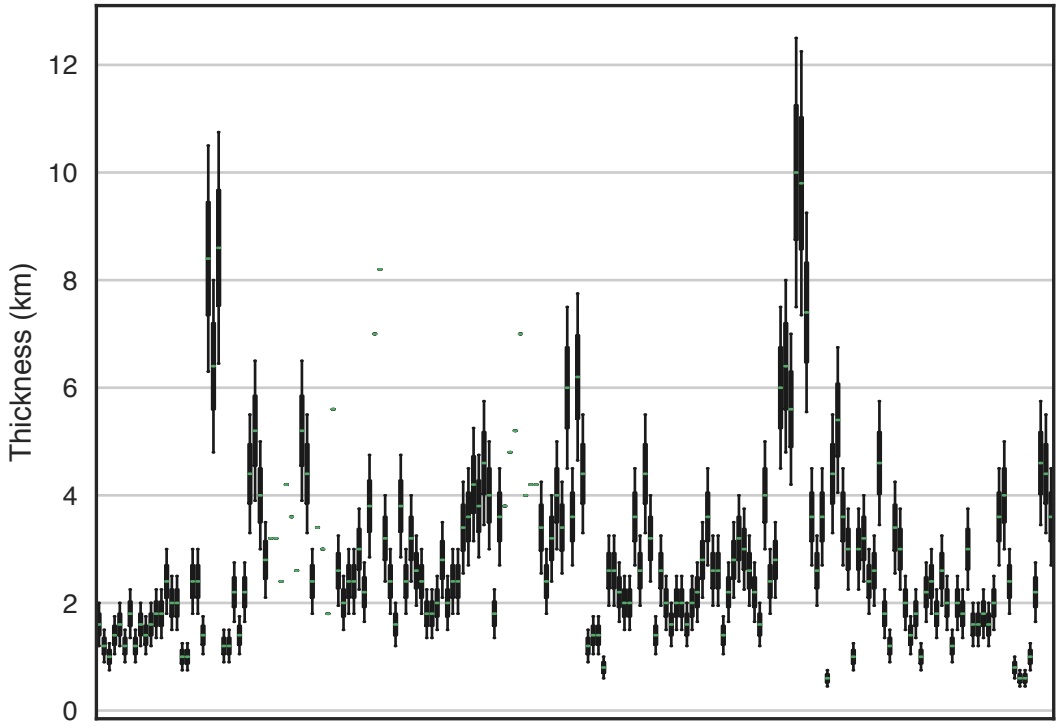


Figure 2.8: Global compilation of SDR thickness measurements, shown using box plots, where green bar indicates the median value. Where only a green bar is shown, this indicates a measurement obtained from depth migrated data, removing the uncertainty in measurement. Note the greater range of thickness estimates for thicker SDRs due to the propagation of uncertainty in the range of velocities used.

of measurements exceed 6 km, and in cases where data has been approximately converted to depth the range is very large.

We also collated information on the overlying sediment thickness (in TWTT) and water depth (Figure 2.9), as a proxy for the stretching factor (β) at this position on the margin. Water depths follow a broadly normal distribution, with an average value in the range 2–3 km. Overlying (post–rift) sediment thickness is more variable, but typically less than 2.5 s. This combination of several kilometres of water depth and thick post–rift sediments suggests SDRs occur at a position on the margin which has undergone significant stretching.

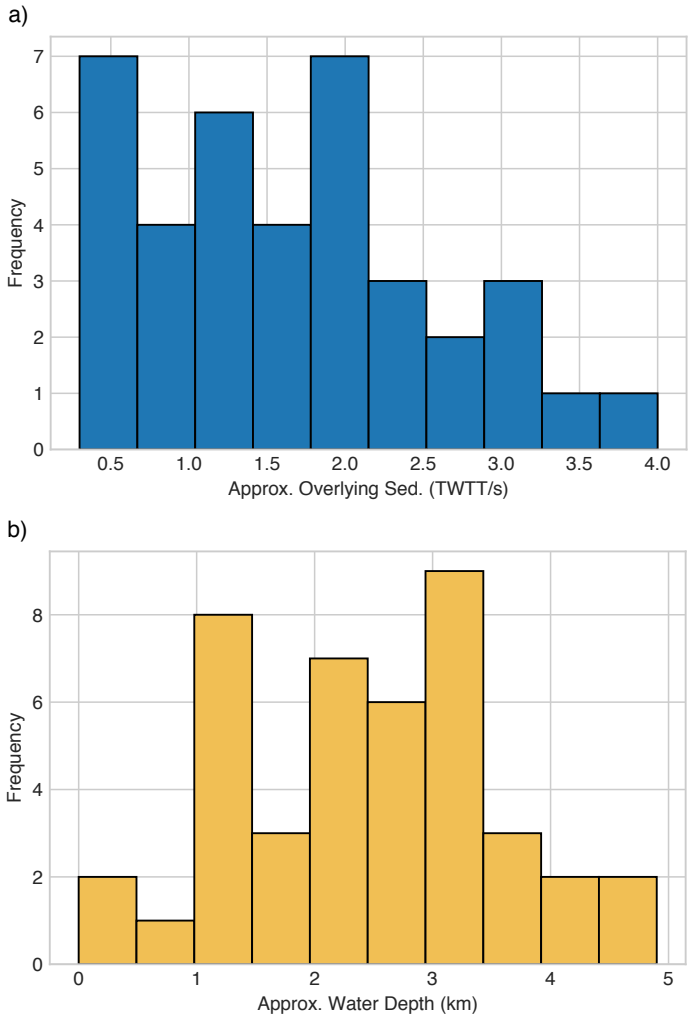


Figure 2.9: Histogram of a) Overlying sediment thickness and b) Water depth above SDRs measured in compilation. Together, these values provide a proxy for the stretching factor (β) at this position on the margin.

2.2.3.4 Along–strike Length

There are comparatively few along–strike margin profiles of SDRs and so it is difficult to determine their length, but they have been mapped along >2000 km of the East Greenland, and East Coast US rifted margins (*Coffin and Eldholm*, 1994; *Planke et al.*, 2000).

2.3 Seismic Uncertainty

As SDRs are by definition observed on seismic reflection data, it is important to understand the inherent limits of this type of dataset in resolving this particular type of geological feature. In some surveys, refraction data is also acquired, providing additional information on velocities and reducing uncertainty in the processing of reflection data. However, this is not always the case. As the defining feature of SDRs is their dip, we must consider the influence of the processing steps applied to the reflection data on the resulting observed dip. In Chapter 3, the impact of semblance analysis is investigated in order to quantify the effects of this process on the resulting SDR geometry. The range of dips observed from using different velocities during stacking is likely to be smaller than that observed here using 1D profiles on already stacked data; in this compilation, we are not using the raw field data as it is typically not available in the public domain.

2.4 Observations from Drilling

Seafloor drilling provides the only opportunity to directly sample the lithology of SDRs in situ. In Namibia, a sequence of wells (Kudu 9A1-3, (*Wickens and McLachlan*, 1990)) sampled the landward edge of the inner SDR package (detailed further in Chapter 4). At the base of the well, a non–marine unit is encountered in which basalts and volcaniclastic sandstones are interbedded with aeolian sandstone, confirming the subaerial origin of this SDR package (Figure 2.10). These subaerial basalts in Kudu 9A-2 may be a tuff, with basalts clasts and a chloritic ash matrix, while the aeolian sandstone is well sorted and rounded (*Wickens and McLachlan*, 1990).

Volcaniclastic units are also found interbedded with shallow marine sandstones higher in the stratigraphic column, although these units are much thinner <20 m. The subaerial/submarine nature of SDRs on the Namibian margin is explored further in Chapter 4 using a dense grid of 2D seismic reflection surveys.

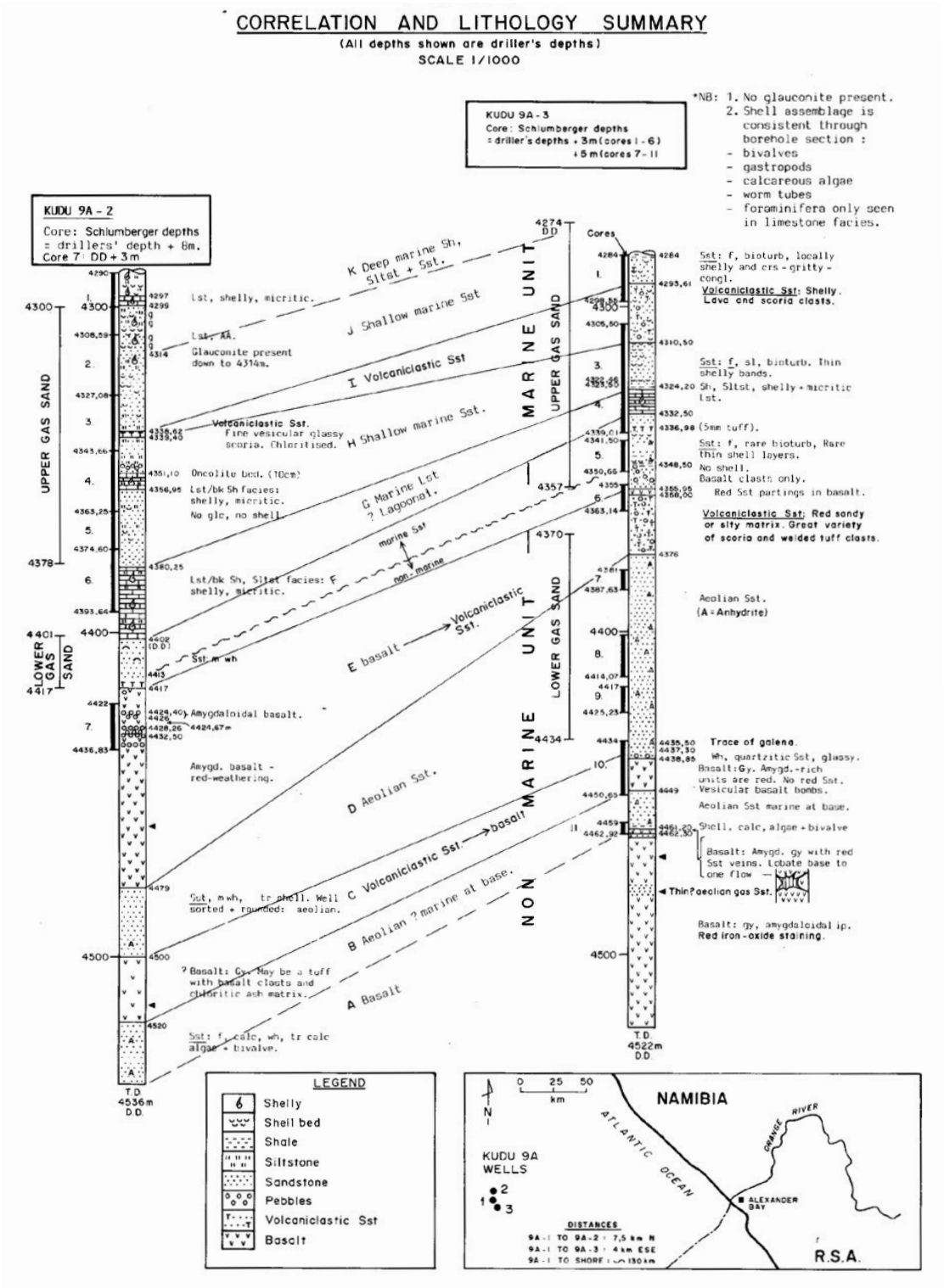


Figure 2.10: Kudu 9A-2 and 9A-3 stratigraphic correlation and lithology summary, from *Wickens and McLachlan* (1990).

2.5 Chapter Summary

- A dataset of geometric measurements obtained from 2D seismic reflection profiles is collected, including examples from the major volcanic rifted margins.
- The dip of SDRs is quantified using a windowed moving average and a suite of 1D velocity profiles in order to capture the range of plausible values over the arcuate shape.
- The dip and thickness of SDRs is not correlated with the year of acquisition.
- Measured dips are typically in the range 5–15°, with few values greater than 30°.
- SDR widths (perpendicular to margin strike) are typically <75 km, and thicknesses (in depth/time) typically 1–6 km.
- SDRs typically lie in several kilometres of water depth, beneath significant thicknesses of post–rift sediment.
- When SDRs are directly sampled by drilling, basalts and volcaniclastic sediments are found on the Namibian margin.

Chapter 3

Seismic Data Processing

3.1 Introduction

Seaward dipping reflectors (SDRs) were first described by *Hinz* (1981) who identified 'extensive wedges of oceanward dipping sub–acoustic basement reflectors' on seismic data from eight examples of passive continental margins. Multichannel seismic reflection profiles were used in each case to confirm the presence of these wedges. DSDP Leg 38 (*Talwani et al.*, 1976) had previously drilled the uppermost part of the wedge offshore the Vøring Plateau, Norway and in Southwestern Africa, confirming their basaltic nature. In the intervening years, SDRs have been identified on many rifted margins globally (Figure 1.1, e.g. Namibia/South Africa, Uruguay/Argentina, Greenland/Norway, Australia), and are one of the defining features of a volcanic, or magma rich, margin, often also associated with high P–wave velocity lower crustal bodies.

Although onshore analogues have been identified in proto–oceanic rifts (*Corti et al.*, 2015), the only place geologists can observe 'true' SDRs which have undergone the complete history of geologic processes is beneath the seafloor. We therefore have three primary sources of observational data: seismic reflection and refraction, ocean floor drilling and potential field measurements (both shipborne and satellite–based). The most useful and unambiguous of these techniques when

determining 2D or 3D structure, and the data source from which SDRs are named, is seismic reflection.

First used in the 1920's (*Kearey et al.*, 2002), seismic surveying is now used extensively by both academia and industry to improve our understanding of the sub–surface. Whereas ocean drilling is limited in depth by engineering capability and lithology, and potential field data are inherently non–unique in interpretation, seismic reflection data offer a unique insight into the 2D and 3D structure kilometres beneath the Earth's surface.

Although the seismic sections produced by stacking common–midpoint (CMP) gathers look like a geological cross–section, there are many pitfalls in directly interpreting them in this way. First and foremost, the data must be converted from the time domain to the depth domain by choosing stacking velocities. This is a fundamental step in processing the data; however, there are many other processing steps which are required to obtain the best possible representation of the true geological structure. In each case, there is a trade–off between improving the quality of the imaging and potentially adding artefacts which may obscure true geological structure. For dipping reflectors like SDRs, these considerations are particularly important, as the underlying assumption in many processing steps is that of flat reflectors.

This chapter explores the impact of stacking velocities on the resulting geometries of SDRs by re–processing a seismic reflection survey acquired on the Greenland volcanic rifted margin in 1996. Later in this thesis, we use the geometries of SDRs to constrain potential models of their formation (Chapter 7); good knowledge of the uncertainties in the geometry from both acquisition and processing will therefore allow more informed interpretation of these results.

3.2 Re–processing 2D Seismic Reflection Data

3.2.1 The SIGMA Experiment

The SIGMA (Seismic Investigations of the Greenland Margin) experiment collected four multichannel seismic (MCS) reflection lines offshore the eastern Greenland margin on the R/V Maurice Ewing in 1996. The aims of the project were primarily to study magmatism and the deep structure of the VRM, aided by deployments of ocean bottom seismometers (OBS), ocean bottom hydrophones (OBH) and portable onshore seismometers collecting wide–angle data (*Hopper et al.*, 2003).

Transect III (location shown in Figure 3.1) lies in the proposed distal zone of the Iceland plume (*Holbrook et al.*, 2001), ∼600 km south of the proposed hotspot track (*Hopper et al.*, 2003). Transect III was chosen for analysis because it crosses a wide area of thickened oceanic crust with two distinct SDR packages, separated by an outer high. Drilling has confirmed that SDRs along this transect are composed of picrites with overlying depleted Icelandic tholeiites (*Saunders et al.*, 1998). Continental contamination is only found near the base of the SDR sequence (*Saunders et al.*, 1998), suggesting that the crust underlying the SDRs is igneous in nature; new crust formed at the incipient spreading centre, not thinned pre–existing continental crust. This is relevant to the model proposed in Chapter 5 of this thesis, in which subsidence and rotation of SDRs is driven by underlying feeder dykes formed at the spreading centre.

Notably, the igneous crust observed on Transect III is significantly wider on the Greenland margin than on its conjugate, Hatton Bank, Rockall Plateau. Accounting for this asymmetry requires a mechanism which can produce significantly different amounts of extension on each side of the ridge axis, without reversing the orientation of SDRs observed on each side. If this asymmetry were due to a ridge jump, we would expect to see a reversal of SDR dip as the spreading began at an offset location; this is not observed here.

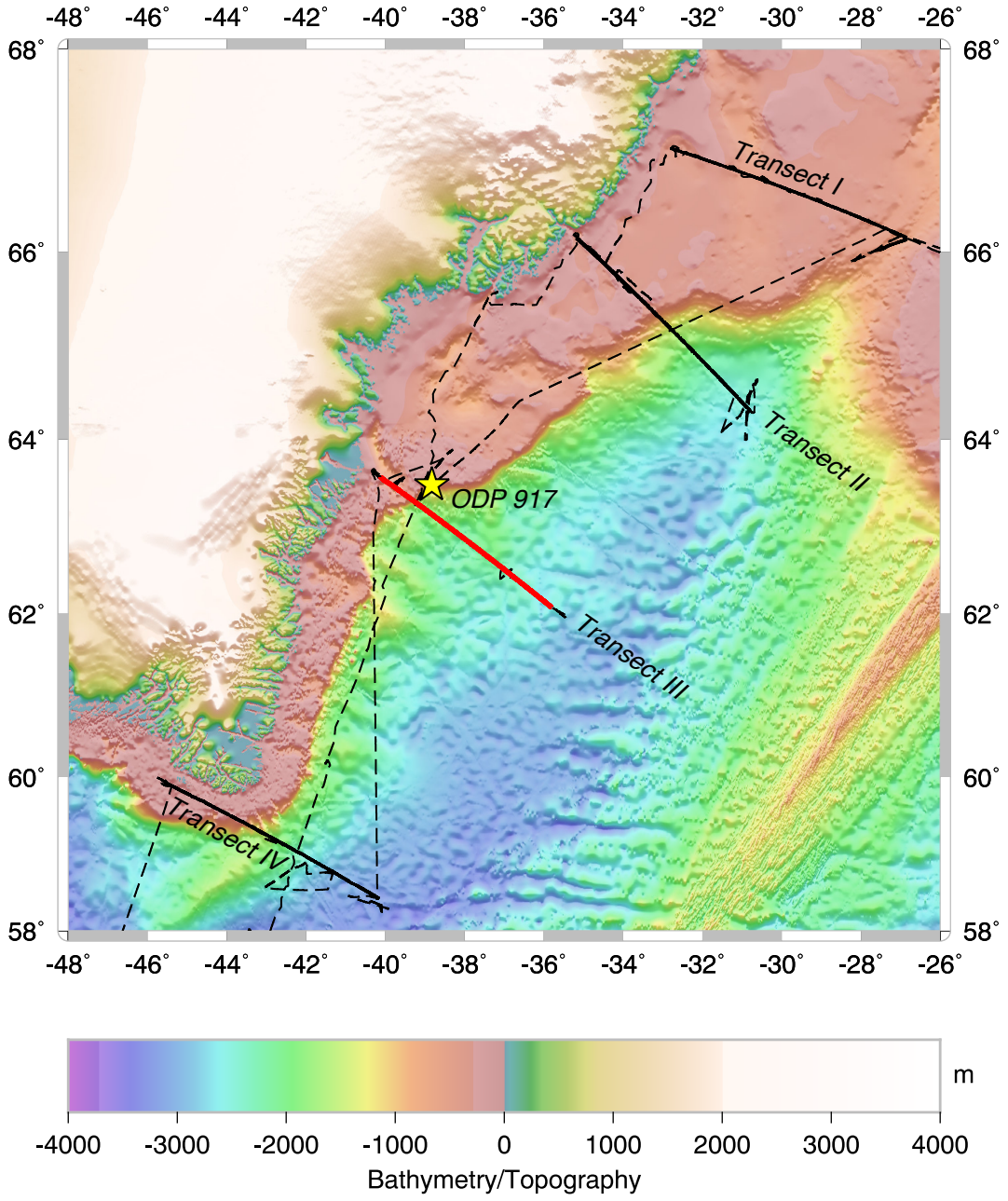


Figure 3.1: EW9607 cruise track with location of Transect III highlighted (thick red line). Dashed black line indicates the cruise track, with solid black lines indicating the locations of other seismic reflection profiles collected on this cruise. Yellow star indicates the location of ODP site 917. Bathymetry from GEBCO 2008.

At ODP site 917, flows at the base of the upper series have been dated to 55.8 Ma (*Tegner and Duncan*, 1999). The termination of the SDRs occurs within magnetic anomaly C24n (Figure 3.2), approximately 53 Ma, suggesting that this innermost package of SDRs was erupted in a maximum of 3 Myr, before passing into chaotic

reflections of the outer high. *Hopper et al.* (2003) identify this high as the transition from subaerial to subaqueous spreading, characterised by a rough basement and short, shallow marine extrusions. Seaward of this outer high lies a second package of SDRs identified by *Hopper et al.* (2003) as deep marine extrusives; when compared to the subaerial SDRs, these deep extrusives are much longer, linear sheet flows, with a corresponding decrease in half spreading rate from opening at $3.3\,\mathrm{cm\,yr^{-1}}$ to $1.7\,\mathrm{cm\,yr^{-1}}$. This outer package of SDRs has not been drilled to determine their age or composition and now lies in 2–3 km of water depth. At the termination of this outer package of SDRs, there is a relatively abrupt ($\sim$5 km) transition into oceanic crust.

A full velocity model for Transect III (*Hopper et al.*, 2003) shows a steeply dipping Moho between 150 km and 160 km along the profile, identified by *Hopper et al.* (2003) as the continent–ocean boundary (COB) based on the division of basement seaward of this point into two distinct layers, with velocities characteristic of oceanic crust layers 2 and 3. This further supports the spreading centre origin of the SDRs.

Acquisition on the Ewing used a 4.2 km, 160 channel streamer with a receiver group spacing of 25 m. The source was a 130 l, 20–gun array shot at a time interval of 21±1s at approximately 4.7 knots, resulting in an approximate shot interval of 50 m (*Dahl-Jensen and Holbrook*, 1996), resulting in a fold of 40. We obtained the raw shot data in SEGY format from the Marine Geoscience Data System (MGDS) platform and re–processed it using GLOBE Claritas.

3.2.2 Motivation for Re–Processing EW9607 Transect III

The aim in re–processing this dataset is to investigate the sensitivity of the final SDR geometry, in particular dip, to the choice of stacking velocities, which are often picked using a measure of semblance. Prior to this, a number of pre–stack processing steps were taken to improve the data quality — in particular, multiple noise was a significant problem. No post–stack processing was undertaken, in

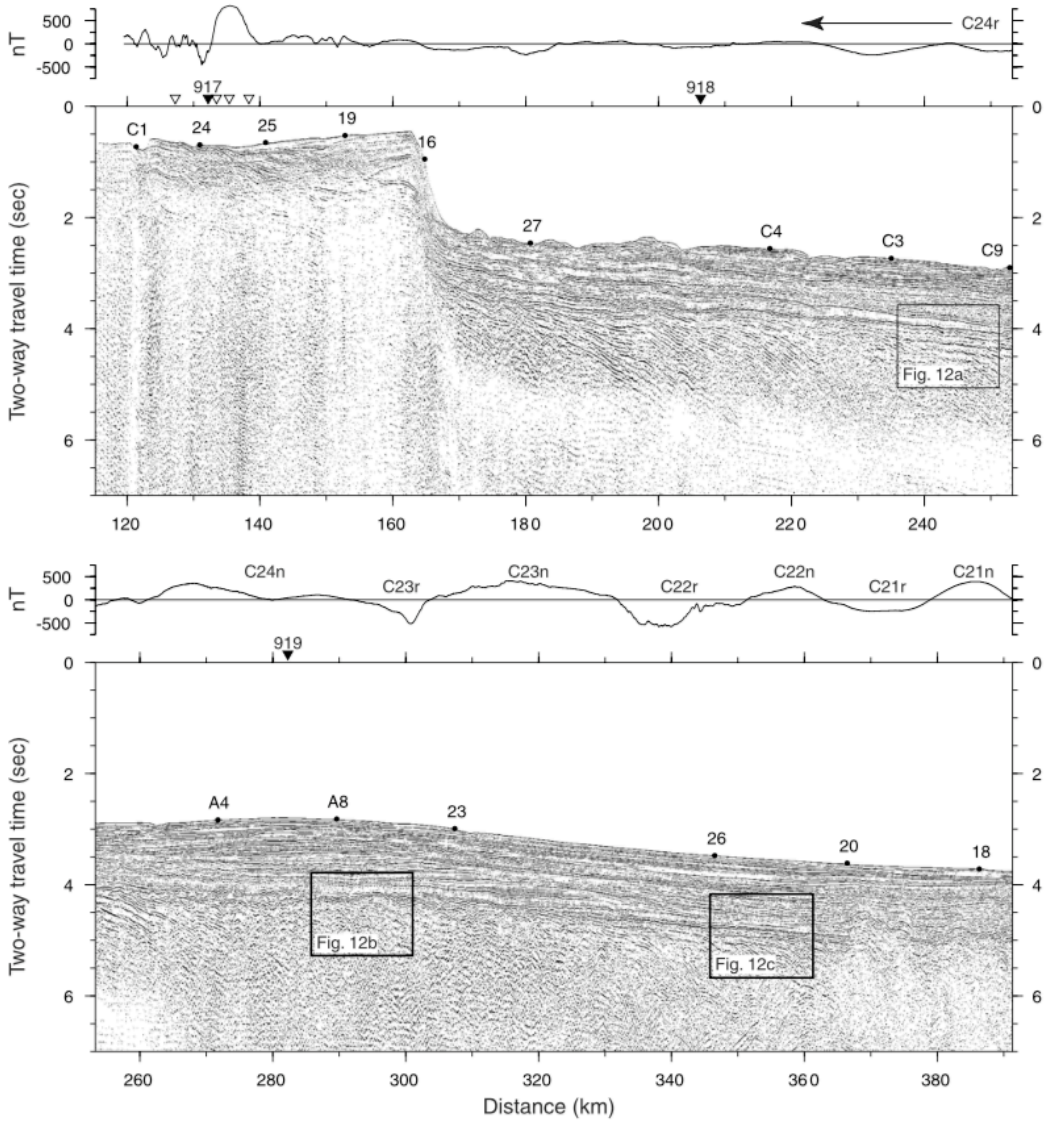


Figure 3.2: EW9607 Transect III with Magnetic Anomaly after (*Hopper et al.*, 2003). Three zones of SDRs are visible; on the inner part of the margin around 170–200 km and 240–270 km and on the outer part of the margin around 350–360 km.

order to restrict the number of steps which could be potential sources of variation in geometry to focus on the impact of velocity.

Every stage in the processing sequence of any seismic reflection profile may affect the geometry, and particularly the dip, of the imaged reflectors. However, the one step which must be common to all profiles is stacking, and this requires velocity information. Although in some reflection profiles, such as those imaging well–stratified sediments, semblance will have a clear maxima at all depths, this is

often not the case in SDRs. Due to their dip and internal variations in lithology, the semblance often lacks a clear maxima, so a range of values may be plausibly chosen at any given depth.

In this section, after the pre–stack processing, we choose a low–bound and high–bound set of velocities through the SDRs using semblance analysis. Once stacked, SDRs are measured with the same approach used in Chapter 2 to determine the statistical significance of the changes in dip and thickness for the chosen range of velocities. This context will aid interpretation of both measured dips in a global context, and inform the use of the dip as a constraint for modelling.

3.2.3 Processing Approach

The goal of this re–processing is not to produce the best possible final image of the entire transect. We focus on standard pre–stack processing to best improve the imaging of SDRs, with no post–stack processing, as my focus is on the impact of variance in stacking velocities. Transect III has previously been processed and published by *Hopper et al.* (2003). As the steps taken in processing are outlined below, we note key similarities and differences between our approaches. Notably, *Hopper et al.* (2003) used post–stack migration, which will not be undertaken here.

The data was first read in to Claritas from the SEGY file, using RECORDNUM and FTRACE as the primary and secondary keys, and saved for processing in the `hdf5` data format. As the transect has a simple 2D linear geometry, the MGEOM tool was used in Claritas to add geometry information, supported by acquisition parameters from the SEGY headers and the cruise report (*Dahl-Jensen and Holbrook*, 1996) (160 channels, 138 m near offset, approximately 50 m shot spacing, 25 m group spacing, 12.5 m CDP spacing).

3.2.4 Pre–Stack Processing

The first step in processing was to resample the data from 2 ms recorded at acquisition to 4 ms. This step reduces the Nyquist frequency by half, effectively filtering out the frequencies above 125 Hz. Frequencies greater than this could introduce aliased noise, so removing them early in the processing steps reduces the chance of errors later. Resampling was achieved with the RESAMPLE module, assuming zero phase.

After resampling, a number of pre–stack filtering steps were combined within a single Claritas job. First, a front (also known as refraction) mute was applied to reduce the noise from direct and refracted arrivals. A Butterworth filter was applied with a lower band of 0–3 Hz and an upper band of 60–70 Hz. In each case, the extreme frequency (smaller for lower, higher for upper) is the stop value, and the other value represents the edge of the pass band.

True amplitude recovery (TAR) was carried out using two steps — a linear scale of $1\,\mathrm{dB\,s^{-1}}$ and a T^2 spherical divergence correction. Both steps were tested to ascertain the best possible value. Spherical divergence was calculated using a pre–existing velocity grid, picked on an earlier pass processing the same dataset. Finally, the AREAL module in Claritas was used to calculate peak and RMS energy values for data quality checking and future trace editing.

At this stage, a brute stack was produced and used to pick a new stretch mute. This removes data at far offsets which have been 'overstretched' by the NMO correction, distorting the reflection event. The unstacked data were then cleaned using TREDIT to remove channels or specific traces identified as anomalous using the AREAL output data.

The remaining processing steps are all used to reduce the impact of multiple reflections, which present a significant challenge in this dataset. Fortunately, the major seabed multiples do not occur at the same depths as the SDRs. The most successful technique proved to be the parabolic Radon transform, using the PRT_DEMULT module, combined with an F–K filter using the FK_FILT module (Figure 3.3).

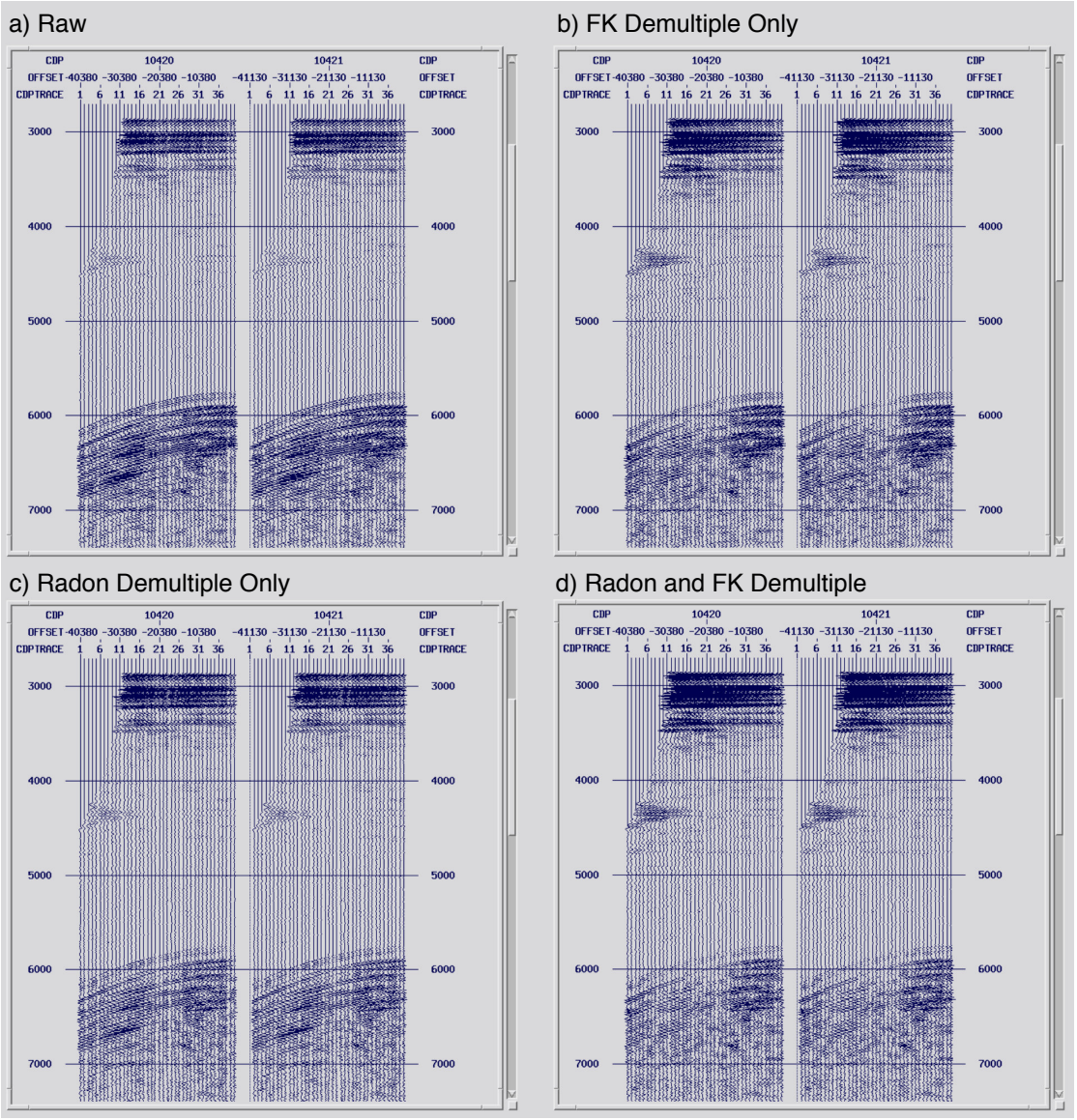


Figure 3.3: Radon and FK Filter processing panels, all shown for the same CDP gather: a) without Radon or DK filter, b) with FK de–multiple only, c) with Radon only and d) with both Radon and FK demultiple. Note the significant improvement in multiple suppression at depths around 6000 ms. Vertical scale shown in milliseconds.

Although this doesn't completely remove the multiple energy, it does result in significant multiple suppression.

Finally, the last processing operation prior to the final stack was dip moveout, or DMO. This is very important when processing reflection data over a stratigraphy which is known to contain reflectors with non–negligible dips. The DMO_FK module (*Hale*, 1984) in Claritas applies a dip move out correction in the F–K domain. In order to use the module, the pre–stack data must first be NMO

corrected and sorted into common offset sections. In this case, BINSORT1 is used module to bin the data with bin widths of 25 m, within the offset range of 138–4113 m. The ratio of shotpoint to CDP spacing determines the number of offsets processed at one time, in this case 4, such that traces across the neighbouring bins will contribute to others within the same mix. After the DMO is applied, the inverse NMO correction is applied, and the sorting is reversed using the BINSORT2 module, such that a pre–stack, DMO corrected datafile may be written to disc.

3.2.5 Semblance Analysis

After the DMO was applied, semblance analysis was used to pick two distinct velocity fields — one picking the lowest values within the maxima, and the other picking the highest. Throughout this process, the gathers were checked to ensure that reflectors were not excessively over or under–corrected. Velocities were picked every 100 CDPs, equivalent to 1.25 km along the profile. At many depths, including over the well defined horizontal sedimentary horizons, there is no difference in these two selections. However, over the SDRs, the maxima in semblance is often poorly defined, such that a wide 'envelope' of reasonable velocities may be picked. For an example of the semblance and velocities picked, see Figure 3.4 which shows a CDP gather over the inner SDR package. The resulting interval velocity grids are shown in Figures 3.5 and 3.6, with the difference between the high and low velocity values shown in Figure 3.7. Note this highlights the distinct packages of SDRs where the largest differences are found, with differences propagating at depth below the SDRs.

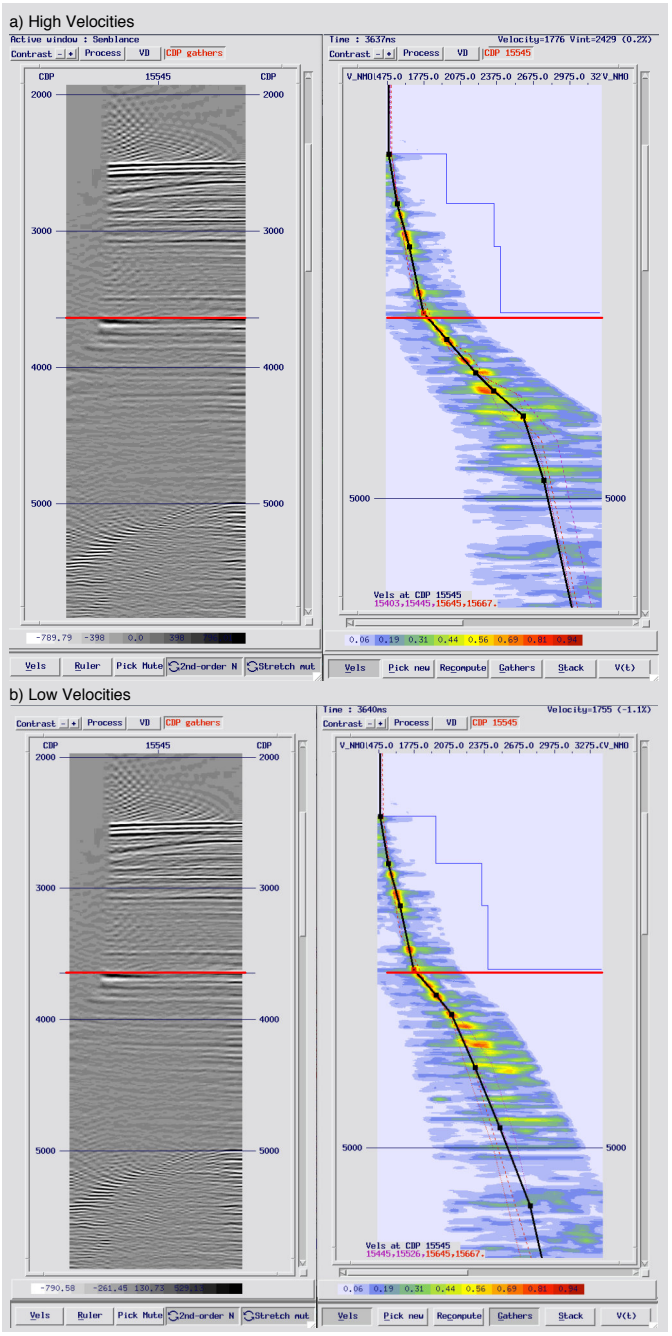


Figure 3.4: Velocity workflow using semblance analysis in Claritas for the same CDP gather in each panel: a) High velocity envelope and b) Low velocity envelope. Vertical dimension in time, with annotations in milliseconds, on all panels. Left hand side shows CDP gathers with the chosen velocities applied. Right hand side shows the semblance analysis panel, with the thick black line indicating the RMS velocity function and the thin blue line indicating the corresponding interval velocity function (*Dix*, 1955). The top of the SDR package lies at $\sim 3600\,\mathrm{ms}$, indicated by the red line on both profiles. Note the broadening peaks in semblance (shown by bright yellows—reds) with depth, particularly over the SDR package.

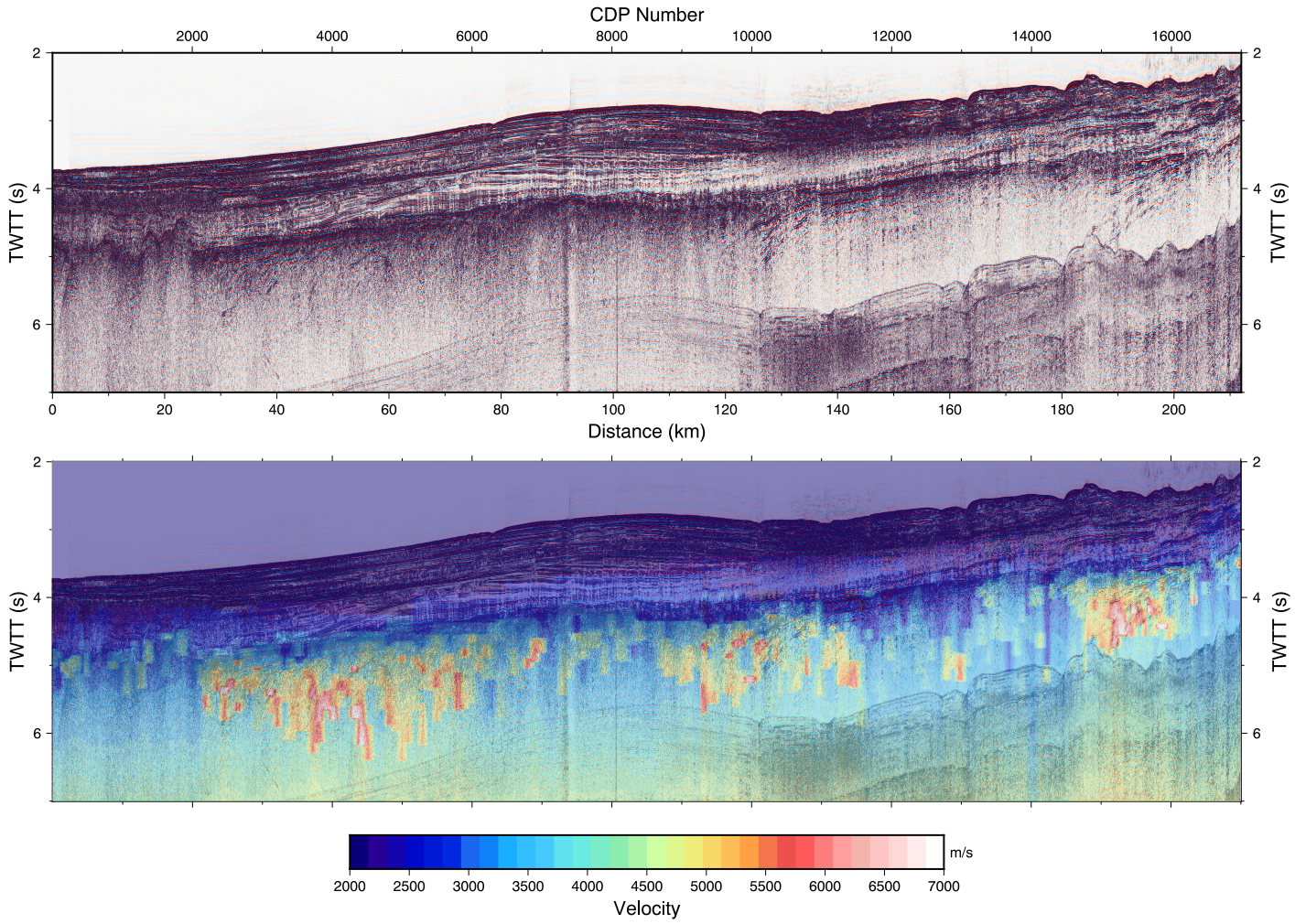


Figure 3.5: EW9607 Transect III re–processed seismic reflection profile using high stacking velocity values (also shown). Both the resulting seismic reflection profile, and the same profile with corresponding interval velocities (*Dix*, 1955) overlain are shown for clarity. Note the peaks in velocity correspond with the locations of the SDR packages, as expected for their basaltic composition.

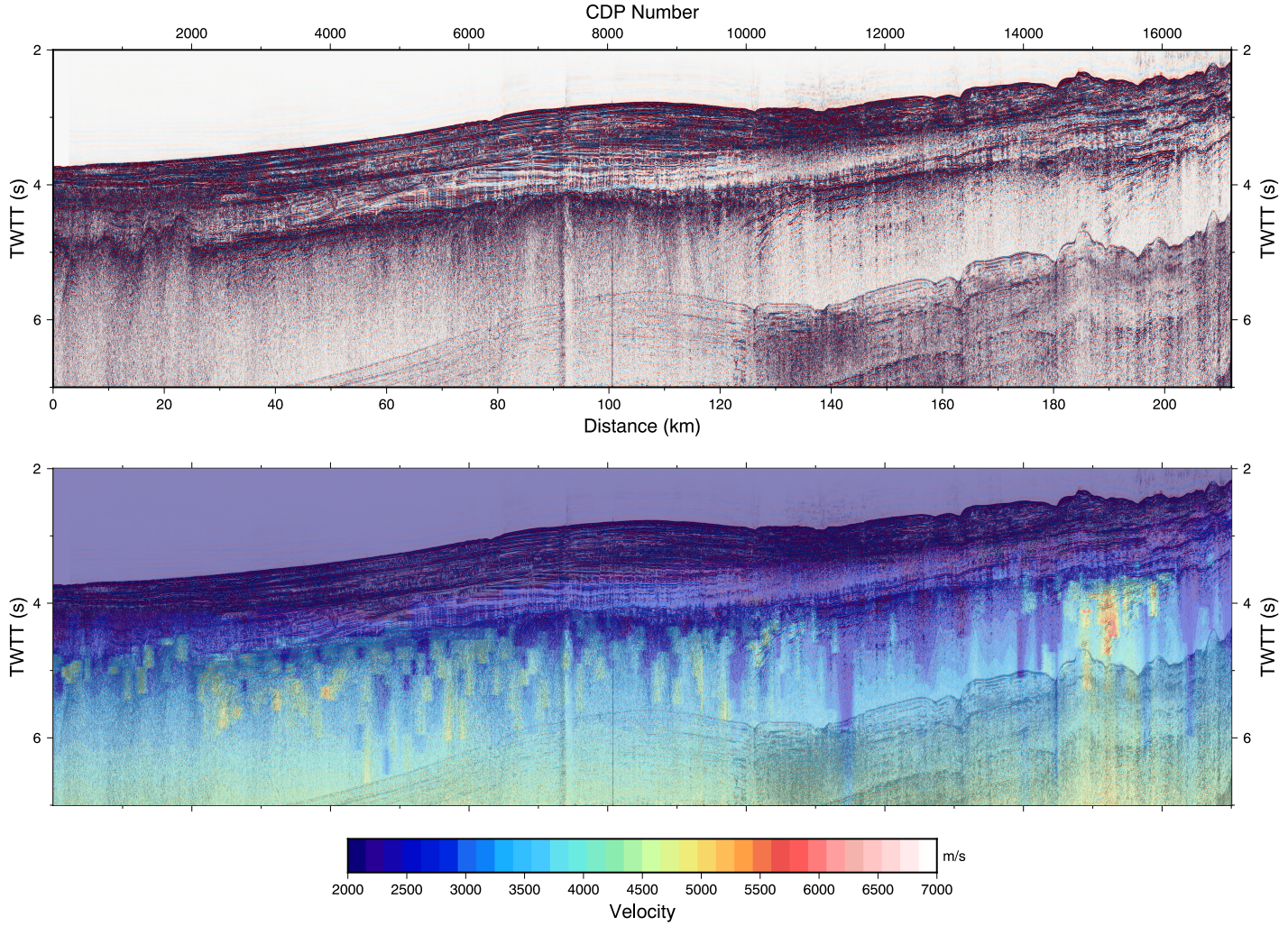


Figure 3.6: EW9607 Transect III re–processed seismic reflection profile using low stacking velocity values (also shown). Both the resulting seismic reflection profile, and the same profile with corresponding interval velocities (*Dix*, 1955) overlain are shown for clarity. The difference between the velocity values shown in this Figure and Figure 3.5 are shown in Figure 3.7.

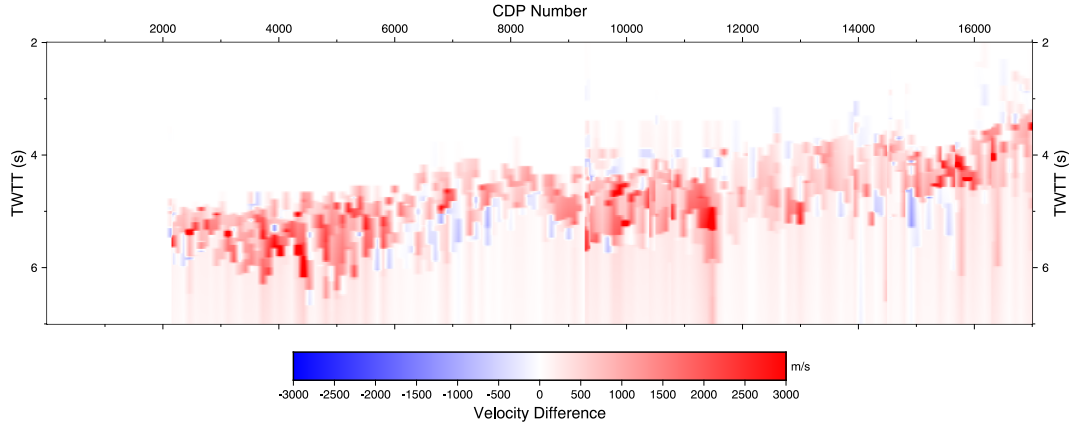


Figure 3.7: EW9607 Transect III difference in high/low envelope of corresponding interval velocities (*Dix*, 1955) derived from semblance analysis. The greatest discrepancy in values is over the SDR packages.

3.2.6 Investigating Migration

Migration is a processing technique commonly used on seismic reflection data, especially in areas with complex geology. Prior to migration, in CMP gathers, it is assumed that the event causing the reflection lies directly beneath the corresponding CMP. However, if the event corresponds to a dipping reflector, this assumption will not hold true. If the reflector is dipping in the plane of the reflection profile, the actual reflection event is displaced up–dip. If there is a component of dip across the profile, the event is displaced out of plane. In 2D reflection data, as discussed here, migration cannot correct for out of plane events due to dips across the profile.

Any non–horizontal surface will not be truly imaged without migration — assuming a zero–offset geometry, any given reflection event may lie anywhere on a locus of points. For a constant velocity above the event, this locus would be a semi–circle which passes through the maximum depth point directly beneath the CMP. For a dipping reflector, this results in an apparent dip on the non–migrated section which is smaller than the true dip.

Performing migration prior to stacking data is much more computationally expensive, but also usually produces a much superior result. When migrating a stack, it is assumed that the stacked events resemble the traces at zero–offset and contain

only normal–incidence reflection events. Migration can be performed in the time or depth domain. Depth migration requires a velocity model, and is preferable in situations where lateral velocity gradients are expected.

Hopper et al. (2003) found that post–stack migration was the most effective technique in reducing multiple energy in this dataset. As we only wish to consider pre–stack processing steps in this experiment, 2D Kirchoff pre–stack time migration is investigated using the IMAGE_K2T module in Claritas. However, in addition to being very computationally expensive, the results were poor. Due to the remaining multiple energy, the migration removed almost all of the reflections below ∼5000 ms, such that almost no SDR signal remained. This was tested with a variety of parameter choices defining the Kirchoff operator; effectively a 'bottle' shape defined by a range and angle within which traces may be migrated. Even with conservative choices of parameters for this dataset (range 3000 m, protect 250 m, maximum dip angle 35°, maximum stretch of 120%), too much of the signal was also removed.

3.2.7 Alternative Processing Methods Investigated

While processing the Transect III data, several other techniques were tested in order to see if they significantly improved the data quality. Unless this improvement was evident, these steps were not included in the final workflow in order to minimise the potential sources of uncertainty in variation in SDR geometry. In particular, given the abundance of multiple reflections in the dataset, several different multiple suppression techniques were used with variable success.

One common step in processing is swell noise attenuation, commonly achieved in Claritas using the DUSWELL process module. This process transforms the data into the wavelet domain, moving through the traces and windowing them to remove wavelets which exceed the threshold value. Swell noise does not represent a significant problem in this dataset; we chose not to continue using the Duswell process here.

The removal of multiples was first attempted using a predictive deconvolution filter, in Claritas using module DECONW. This method is most successful on datasets in which the primary multiple has a period shorter than 200 ms; in this dataset, the SDRs lie in a water depth of ∼4000 ms, so the long period multiple is of much greater wavelength. A number of different filter and gap lengths were investigated, but all had little impact on the multiple energy. The Radon transform proved to be much more successful in removing multiple energy.

3.3 Investigating the Impact of Stacking Velocities on SDR Geometries

3.3.1 Results from Transect III

Once the final stack was processed for each of the high and low velocity cases, plots were produced of the 'inner' and 'outer' SDR packages for analysis (Figures 3.8 and 3.9 respectively). For details on the method of this analysis, see Chapter 2 on SDR Observations; the same methodology is used on this dataset. In short, key reflectors are picked throughout the package by visual inspection, then various geometric measurements are taken using functions written in Python (see Appendix B). For the majority of published data, and for this dataset, the vertical dimension is time not depth – this requires a conversion to depth, for which we use a range of plausible SDR velocities (Chapter 2) between 3–6 $\mathrm{km\,s^{-1}}$. In this case, we do have access to the velocity information, but this is *not* typically available for published data. This analysis is therefore performed in the same way as in Chapter 2 in order to assess independently whether significantly different results would be obtained due to only a difference in stacking velocities.

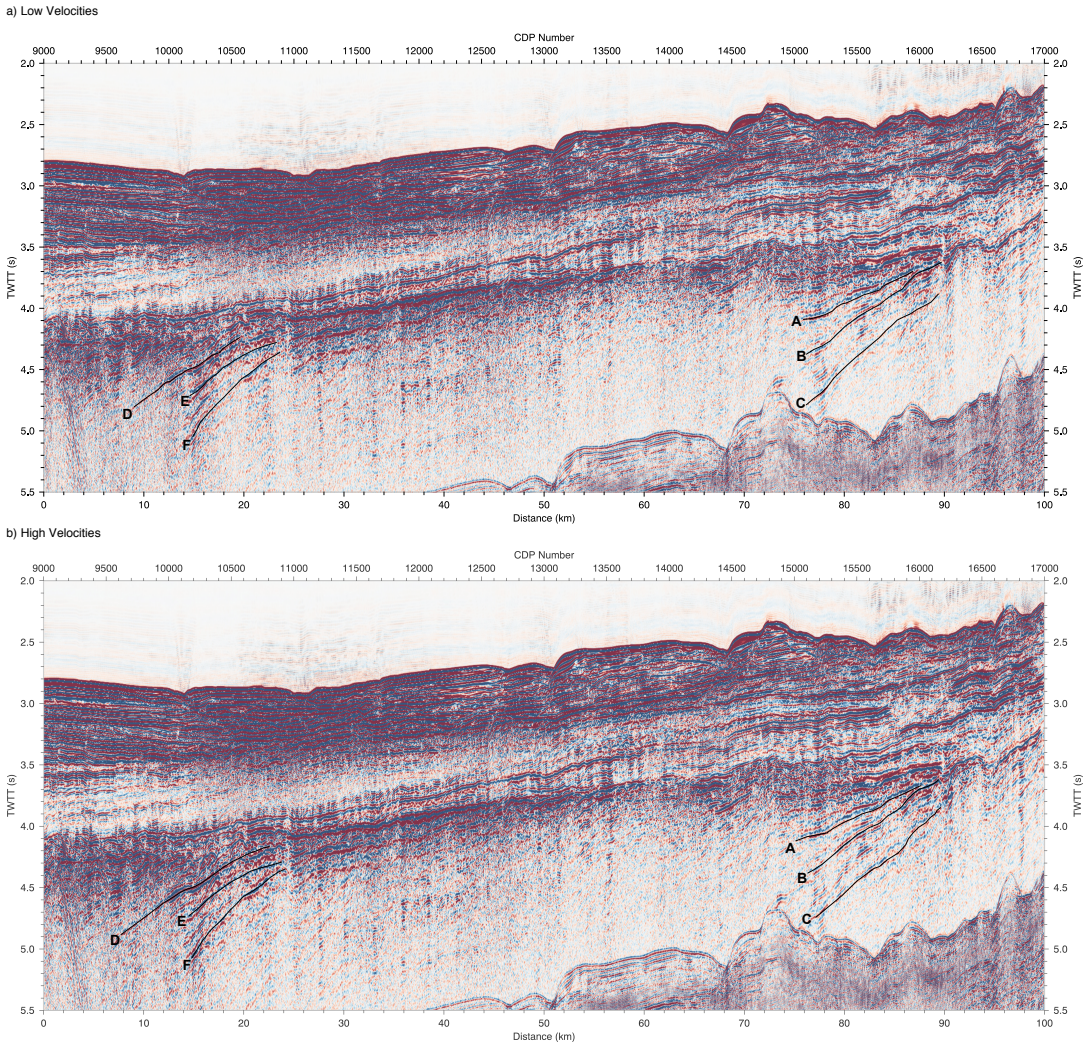


Figure 3.8: Inner section of EW9607 Transect III seismic reflection profile, comparing results of high and low velocities a) stacked using low velocities and b) stacked using high velocities. In both cases, we have attempted to identify corresponding SDRs for measurement (annotated in black, A through E). Note still significant seafloor multiple below the SDR package.

3.3.1.1 Comparing High/Low Velocity Geometries

The dip distribution for the two velocity distributions in the outer package is shown in Figure 3.10, and for the inner package in Figure 3.11. The minimum dip is the same in all four distributions (3°). The maximum dip, however, is affected, increasing by 3° for the higher velocity case when compared to the low in each of the inner and outer comparisons.

The mean dip is unaffected by velocity choice on the inner package (7°), and by

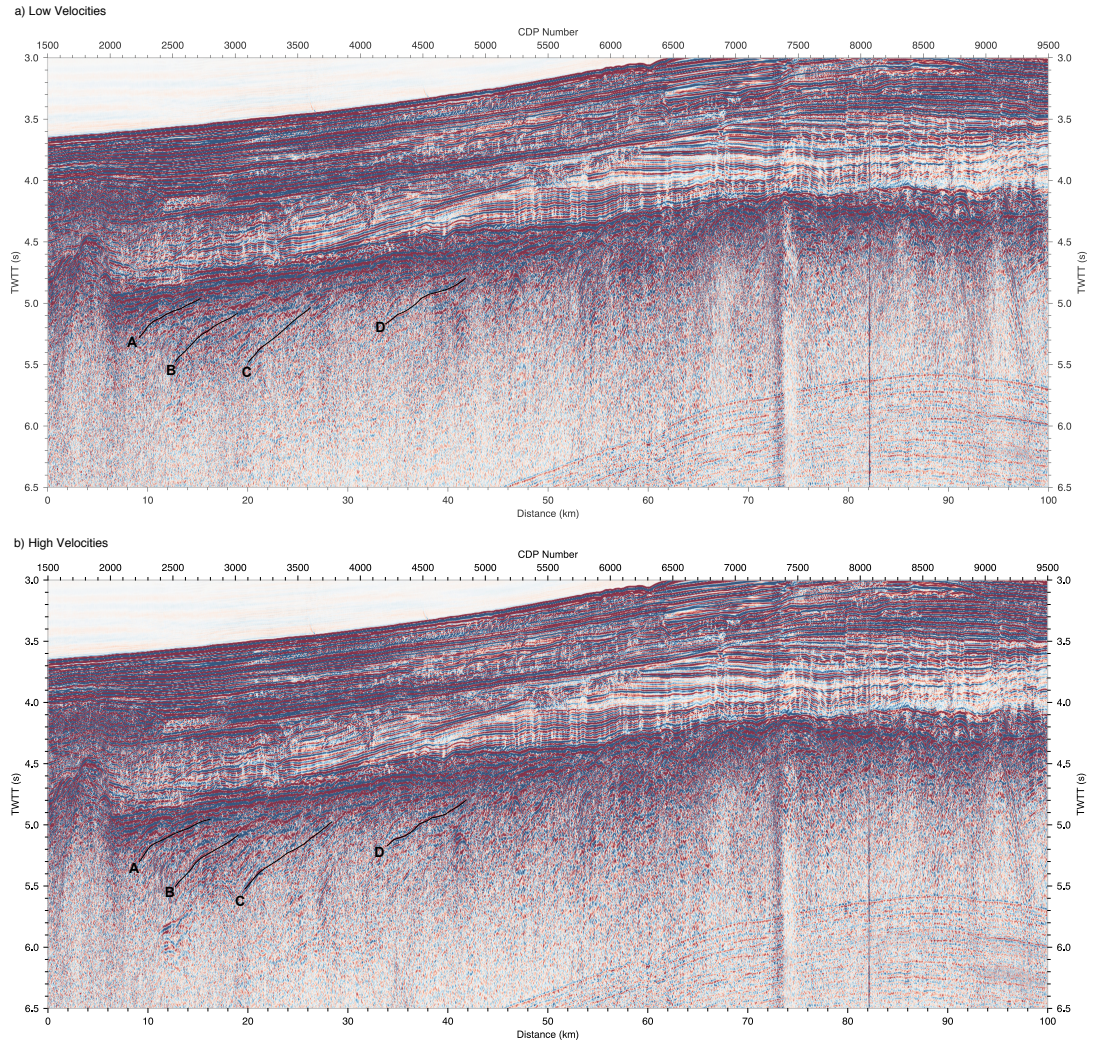


Figure 3.9: Outer section of EW9607 Transect III seismic reflection profile, comparing results of high and low velocities a) stacked using low velocities and b) stacked using high velocities. In both cases, we have attempted to identify corresponding SDRs for measurement (annotated in black, A through D).

$<1°$ on the outer package (6.4° for low velocities compared to 6.7°) for high. The median dip is unaffected in both inner and outer packages by velocity choice; 7° in the inner package, 6° in the outer. The standard deviation of dips is higher in both cases for the higher velocity stack.

Considering the thickness of SDR packages, the minimum thickness is unaffected by velocity choice in the outer package (0.5 km), and only by 0.1 km in the inner package (0.5 km for low velocities compared to 0.6 km for high). The maximum thickness is unaffected by velocity choice in the inner package (2.3 km) but differs

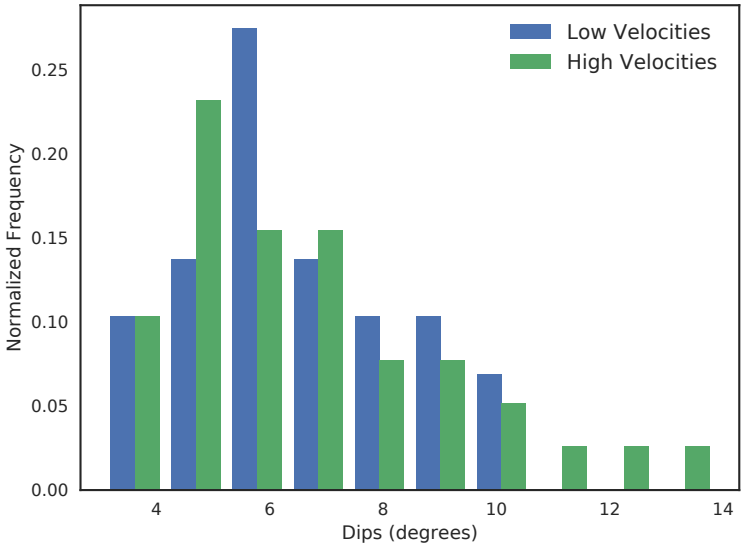


Figure 3.10: Histogram of outer SDR package dip distribution for high (green bars) and low (blue bars) stacking velocities. As anticipated, high velocities skew the distribution to higher values.

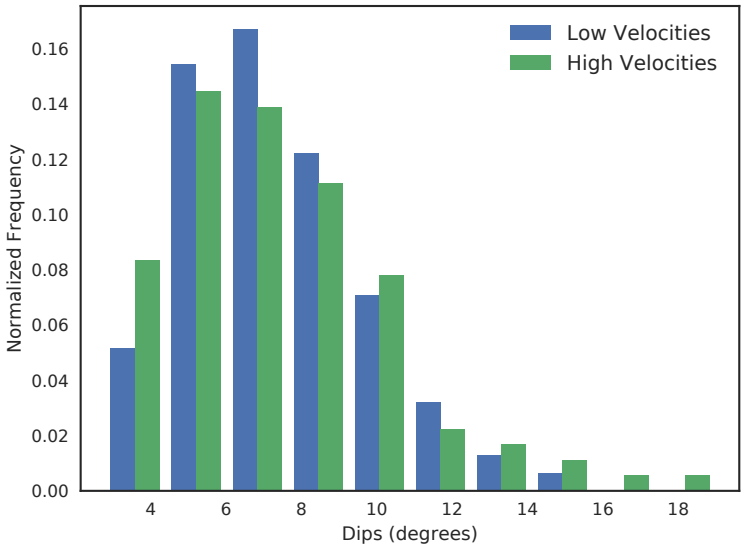


Figure 3.11: Histogram of inner SDR package dip distribution for high (green bars) and low (blue bars) stacking velocities

by ~30% in the outer package (1 km for low velocities compared to 1.3 km for high). The mean and median thicknesses have some variability with velocity in both the inner and outer packages.

3.3.1.2 Statistical Significance of Results

We use a two–sample Kolmogorov–Smirnov test (K–S test) to determine whether the dip distributions from the high and low velocity cases are significantly different. The null hypothesis in this test is that the two distributions are drawn from the same reference distribution, with no assumptions made about the nature of this underlying distribution. In other words, despite the difference in stacking velocities, in terms of dip distribution the two resulting SDR packages are statistically indistinguishable.

The K–S test is performed using the stats package in SciPy (*Jones et al.*, 2001), on dip distributions derived using the methods outlined in Chapter 2. The results show (Figure 3.12) that for both the outer and inner packages, we cannot reject the null hypothesis at a 80% or lower alpha since the p–values of 0.986 and 0.807 respectively (Table 3.1) are higher than 0.8. The high/low stacking velocity dip distributions are therefore statistically indistinguishable. Further, comparing each of the high and low velocity cases to the published result from *Hopper et al.* (2003), we cannot reject the null hypothesis at a 75% level for the inner package. The outer package does not appear statistically similar; this is likely because only three of the four reflectors identified in the high/low comparison could be picked on the *Hopper et al.* (2003) profile as it was partially obscured by an annotation. The SDRs are generally harder to resolve in the outer package than the inner in all three datasets.

The cosine similarity is also given as another measure of similarity between two distributions, calculated using a normalised dot product between two vectors. This requires that the dip distributions first be regularly binned. In general, there is a good agreement between the K–S test p value and cosine similarity (Figure 3.12), but the cosine similarity appears to be less robust in the case of small sample size (e.g. outer package *Hopper et al.* (2003) measurements).

Table 3.1: Statistical summary of dip distributions for EW9707 Transect III, including comparison to Figure 3 of *Hopper et al.* (2003) — The D value is a measure of the maximum distance between the two empirical distribution functions in each case.

Package	Dataset 1	Dataset 2	K–S D value	K–S p value	Cos. Sim.
Outer	High	Hopper	0.278	0.265	0.953
Outer	Low	Hopper	0.278	0.321	0.740
Outer	High	Low	0.111	0.986	0.905
Inner	High	Hopper	0.090	0.787	0.995
Inner	Low	Hopper	0.094	0.777	0.974
Inner	High	Low	0.088	0.807	0.986

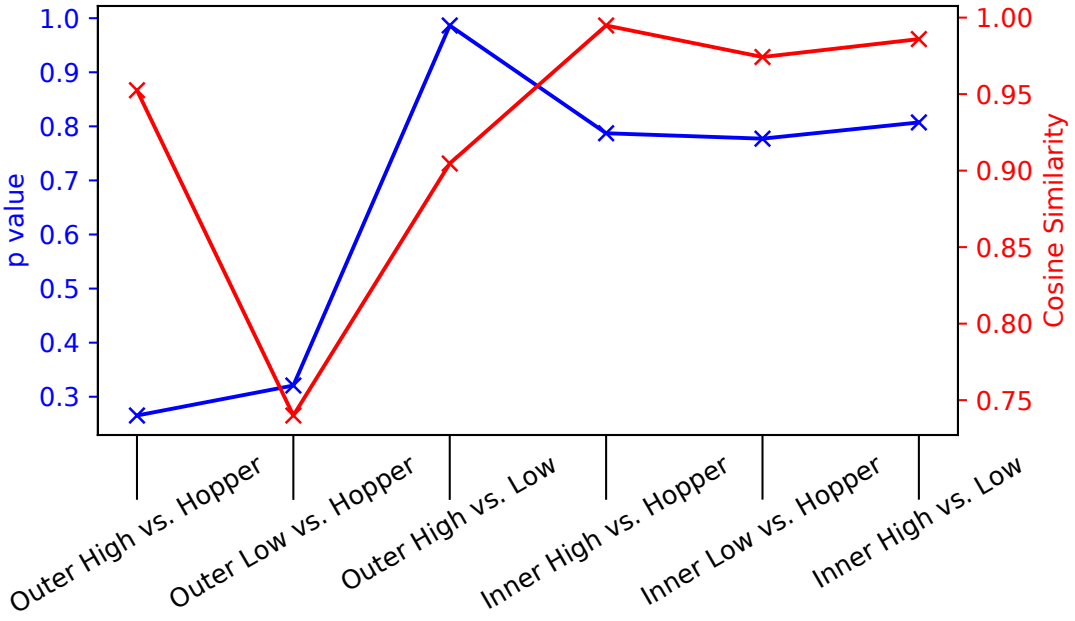


Figure 3.12: Graphical comparison of dip distribution statistical metrics for EW9707 Transect III velocities. The K–S test p value and cosine similarity are shown for each pairwise comparison between the high and low velocities used here, and the original processing conducted by *Hopper et al.* (2003). Note the good general correspondence between these two measures of distribution similarity.

3.3.2 Conclusions

- Measures of average of dip seem fairly robust to velocity choice, while the extremes are less so.
- Conversely, minimum and maximum thickness are also reasonably robust, with some variability in the averages.

- Mean dip and dip distribution, and minimum and maximum thicknesses, have been used as the main constraints in my later modelling chapters; this selection of metrics do appear to be the most robust, given a significant range of semblance velocity choices for a given dataset.

Chapter 4

Seismic Interpretation — Namibian Margin

4.1 Introduction to the Namibian margin

The Namibian margin lies in the South Atlantic, which in this region opened approximately 135 Ma (*Hall et al.*, 2018), with seafloor spreading and the propagation of the ridge axis from South to North (Figure 4.1). This area has been extensively studied both in academia and industry, since gas was first discovered in the Kudu 9A-1 well in 1974 (*McLachlan*, 1990). The most notable bathymetric feature of the margin is the Walvis Ridge, an aseismic ridge stretching from the northern Namibian coast towards Tristan de Cunha, following the path of the hotspot track.

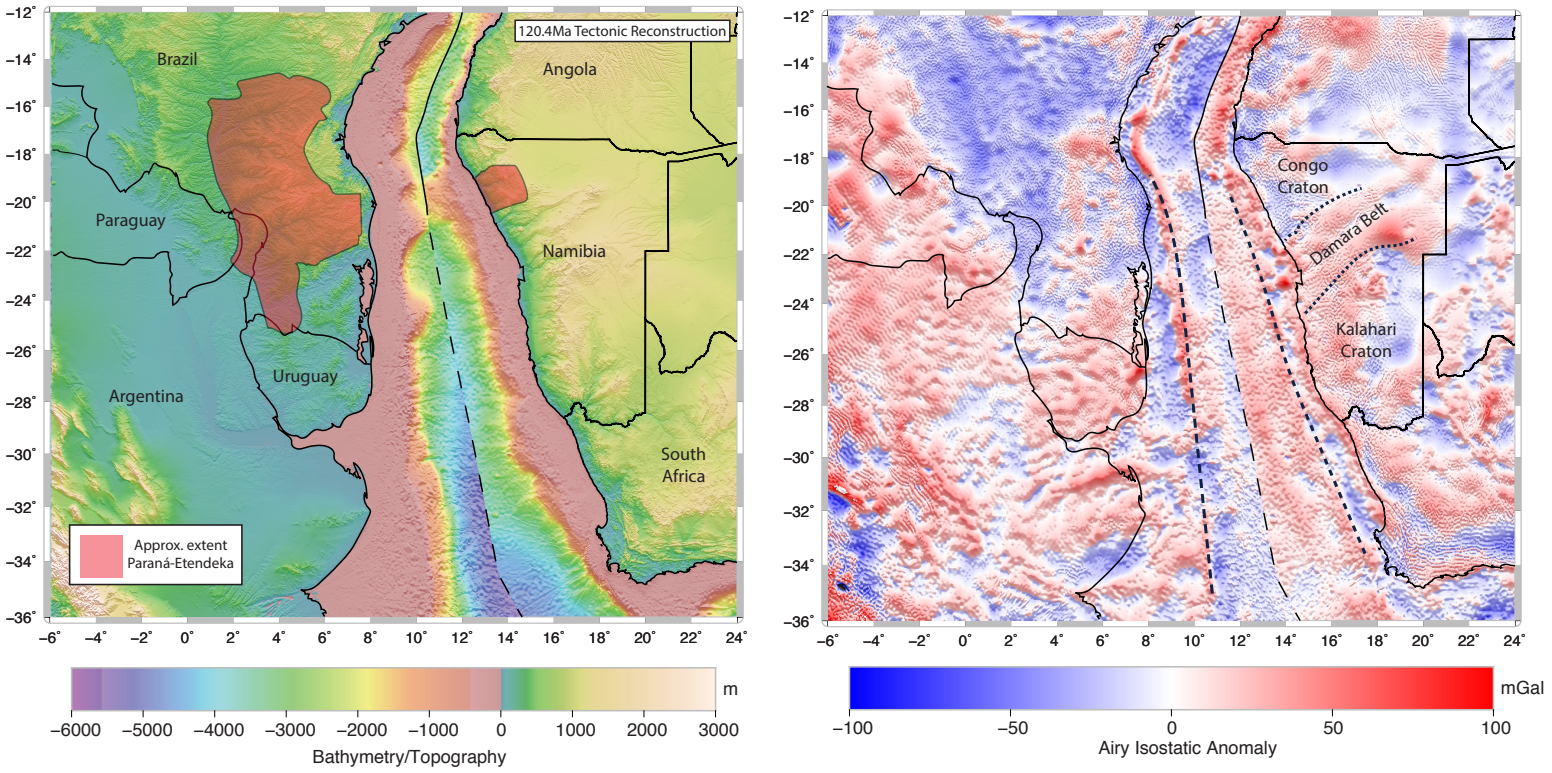


Figure 4.1: South Atlantic bathymetric and topographic (left, GEBCO 2014, (*BODC*, 2014)) and isostatic gravity anomaly (right, see Chapter 6) reconstruction at $120.4\,\mathrm{Ma}$ using rotation poles from *Torsvik et al.* (2008). Coordinates shown relative to a fixed African plate. Calculation of isostatic anomaly described in Chapter 6. On the left panel, the approximate present day extent of the Paranà–Etendeka LIP is shown in red; on the right panel, the cratonic structure of Namibia shown.

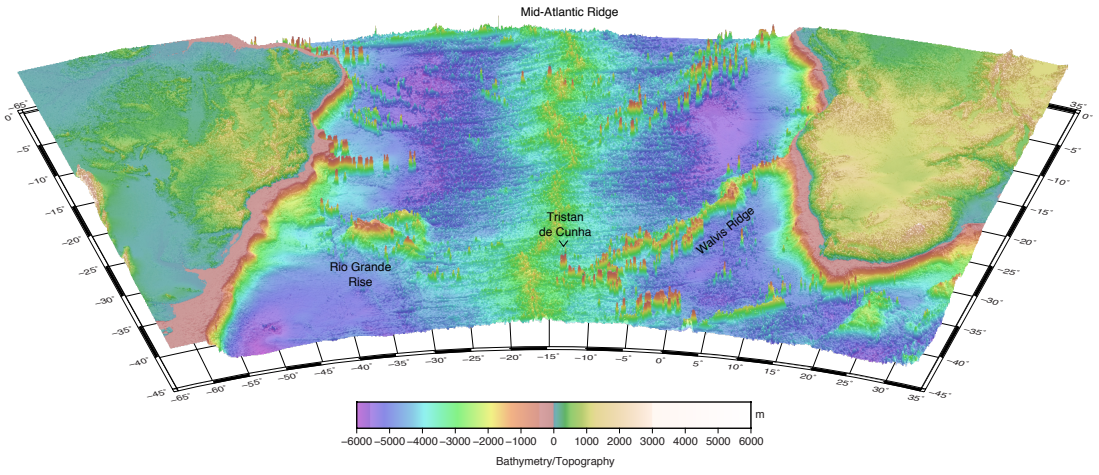


Figure 4.2: South Atlantic bathymetry perspective view. Key bathymetric features annotated include the Mid–Atlantic Ridge, the Rio Grande Rise and the Walvis Ridge.

This chapter discusses the interpretation of a set of three closely spaced 2D seismic reflection surveys from the Orange Basin, offshore Namibia. This area has well–developed SDRs, the 'inner' package of which has been drilled (*McLachlan*, 1990) to confirm their subaerial basaltic origin.

Conjugate to the Namibian margin is the Uruguay–Argentina margin (Figure 4.2), which has also been associated with magmatism. Although not the focus of this thesis, ongoing work suggests a distribution of SDRs similar to those found in Namibia (*McDermott et al.*, 2018) which likely erupted at a similar, if not coincident, time.

4.2 Comparison of Imaging across Different Surveys

Each seismic reflection profile image is a product of numerous factors, not a direct image of the geological features targeted in the subsurface. First and foremost, the design of the original acquisition survey and equipment used will be a major influence on what features are visible. Importantly, the technological capability of reflection surveying has increased rapidly over the last several decades, resulting

in significant improvements in resolution and depth of imaging. Secondly, once acquired, the data must be processed in order to produce the desired image of the geological section in question. The rise in computational power has likewise permitted more intensive processing steps to be used. For more detail and discussion on seismic processing, see Chapter 3.

In this section, four seismic reflection datasets from the Namibian margin are compared, and the differences between these surveys in terms of imaging SDRs are investigated. As SDRs and the oceanic crust are unlikely to be the primary targets of industrial surveys, there may be greater disparity between surveys than we might expect to observe, for example, in a sedimentary package.

4.2.1 Description of Surveys

Courtesy of Spectrum Geo, we have access to four distinct 2D seismic reflection datasets from the Orange Basin. Two of these datasets are derived from the same acquisition survey, on the M.V. Patrick E. Haggerty in December 1989, but have different processing histories (ECL-89 and ECL-89-Repro). The remaining two datasets are more recent, acquired in 2012 on the M/V Northern Explorer (SCOB-12) and in 2014 on the BGP Challenger (NAMZA-14). The area covered by these datasets is shown in Figure 4.3. Full example headers for each of the surveys are given in Appendix A.

4.2.1.1 ECL-89

The oldest dataset available here, ECL-89 was acquired in 1989 by Western Geophysical and processed by Veritas Geophysical. The original recording length was between 5–7 s with a 3.555–3.735 km cable at an interval of 2 ms. Notable processing steps include FX trace interpolation to 7.5 m channels, a high resolution Tau–P domain dip filter, initial 2 km velocity analysis, pre–stack time migration

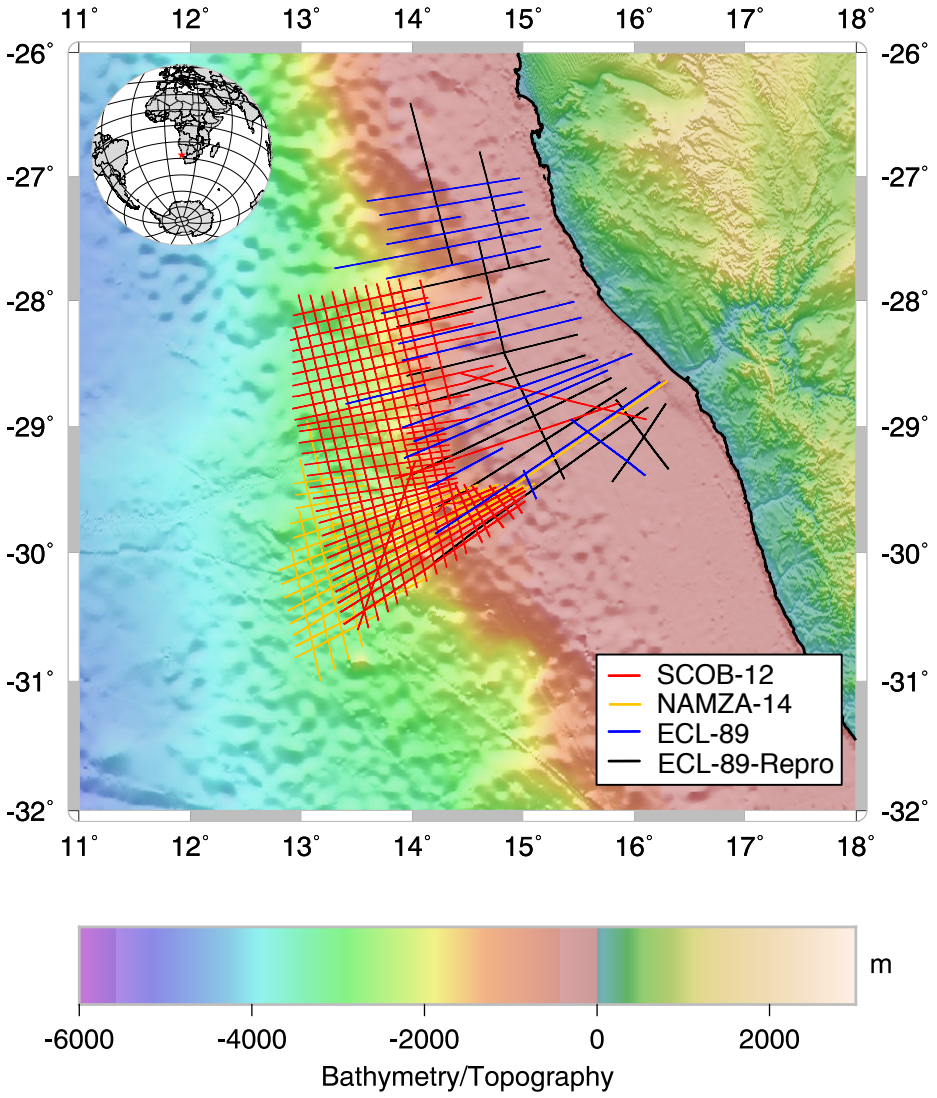


Figure 4.3: Location of seismic surveys, Orange Basin, Namibia, provided by Spectrum Geo. Note that ECL-89 and ECL-89-Repro are two subsets of the same original survey, with different processing histories. Red star on globe indicates position of this, and subsequent, localised figures. Bathymetry from GEBCO 2014 grid (*BODC*, 2014).

(PSTM) using full curved–ray Kirchoff migration, final velocity model picked every 1 km and high resolution Radon multiple attenuation. The resulting 60–fold stack has a CDP spacing of 7.5 m.

4.2.1.2 ECL-89-Repro

The ECL-89-Repro dataset is from the same 1989 cruise, but has been re–processed by Spectrum Geo in 2012. These data were processed with two passes of PSTM and a high resolution Radon filter. The processed CDP interval is 15 m. The two datasets from the ECL-89 cruise do not contain any of the same lines; the reprocessed lines are interspersed with lines from the originally processed survey (see blue and black lines in Figure 4.3). The ECL-89 survey is the most landward of the available datasets, with data predominantly over the continental shelf.

4.2.1.3 SCOB-12

Unlike ECL-89, the SCOB-12 survey lies predominantly seaward of the continental shelf and the line spacing is much more dense. This is the largest of the available datasets and images the outer SDR package extensively. Acquired with a 10 km streamer with a record length of 10 s and sample interval of 2 ms, the final processed stack has a CDP interval of 12.5 m. Notable processing includes two passes of Kirchoff PSTM and high resolution Radon demultiple filtering.

4.2.1.4 NAMZA-14

Like SCOB-12, this dataset lies largely seaward of the continental shelf, in water depths of ∼2–3 km. The dip lines typically lie between the SCOB-12 grid, such that the closest lines lie less than 5 km apart. Acquired with a 10 km streamer with a record length of 10 s and sample interval of 2 ms, the final processed stack has a CDP interval of 12.5 m. Processing is in general very similar to the Spectrum Geo processed SCOB-12, with notable processing including two passes of Kirchoff PSTM, high resolution Radon demultiple filtering and residual gain correction.

4.2.2 Comparison of Reflector Imaging on Adjacent Lines from Different Surveys

The area of interest in this thesis lies over the outer SDR package and the transition into oceanic crust, covered by the SCOB-12 and NAMZA-14 surveys. As these surveys have similar, but not identical, acquisition and processing histories, and very close spatial locations, this represents an ideal opportunity to compare imaging of SDRs and associated geological features.

In order to locate the closest possible lines, overlapping sections in longitude of dip lines were identified from all datasets using the navigation data. Within these overlapping sections, coordinates were interpolated onto regularly spaced points in longitude, then the root–mean squared (RMS) difference between the coordinates of every possible pair of seismic lines was calculated. Ignoring a line from the NAMZA-14 survey, which contained a curved section to avoid a seamount, the two closest lines are from the NAMZA-14 and SCOB-12 surveys — here termed NAMZA-14-ADJ1 and SCOB-12-ADJ2 (line names in this Chapter are omitted for confidentiality reasons).

The coincident sections of these two lines (in longitude) are shown in Figures 4.4 and 4.5, displayed with the same aspect ratio, with an annotated comparison shown in Figure 4.6. The greatest distance between the coordinates of the two lines is at the seaward extreme, at 5.1 km, with only 1.7 km between the lines at the landward termination (shown in Figure 4.7). Both lines image the outer SDR package and the transition to normal thickness oceanic crust.

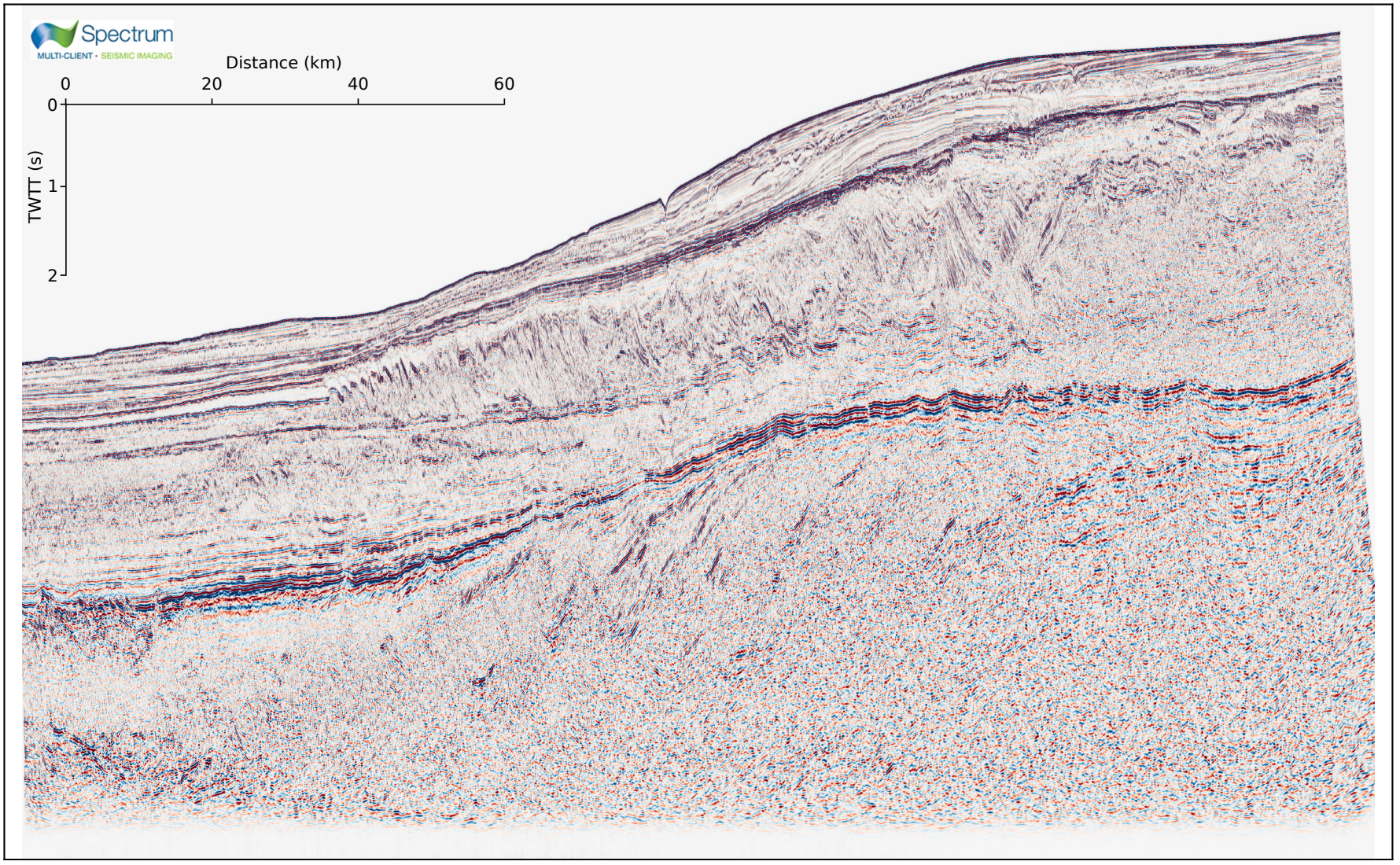


Figure 4.4: SCOB-12-ADJ2 seismic reflection profile. Annotated profile shown in Figure 4.6(i). Note horizontal and vertical scale bars.

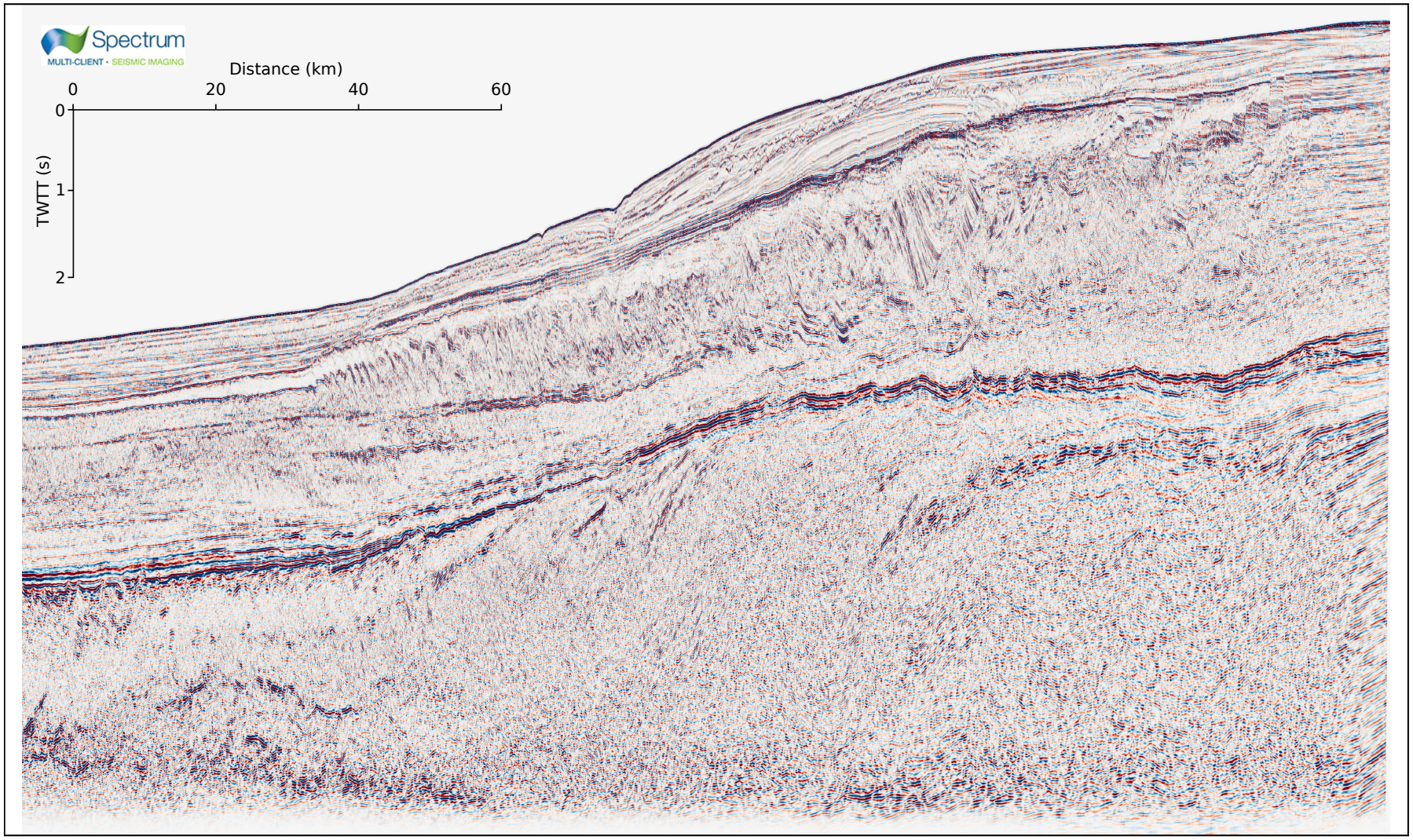
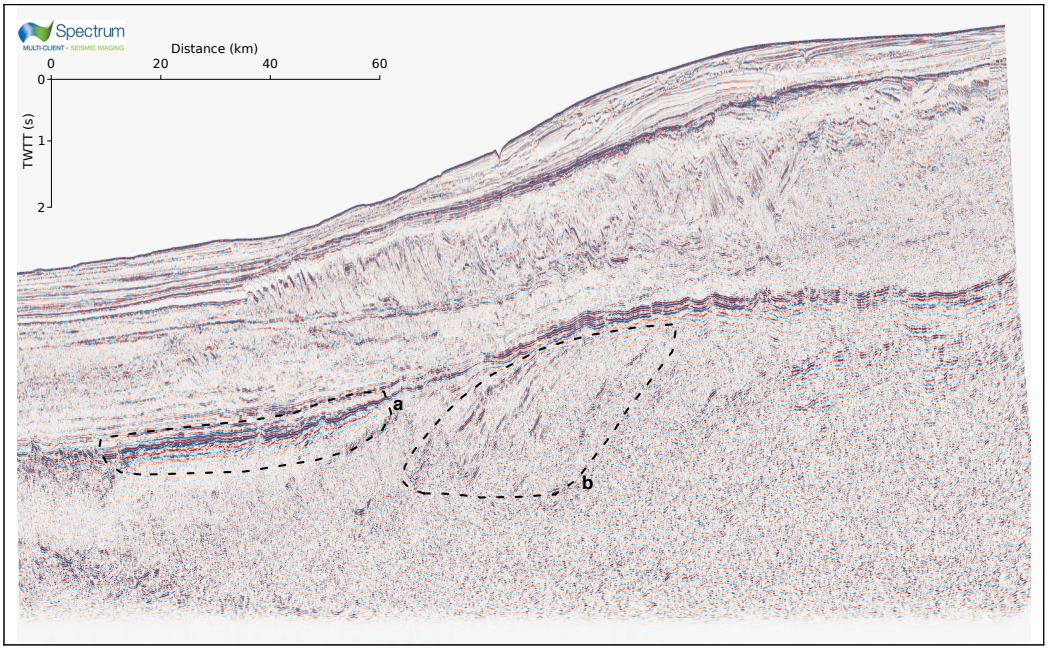


Figure 4.5: NAMZA-14-ADJ1 seismic reflection profile. Annotated profile shown in Figure 4.6(ii). Note horizontal and vertical scale bars.

i) SCOB-12-ADJ2



ii) NAMZA-14-ADJ1

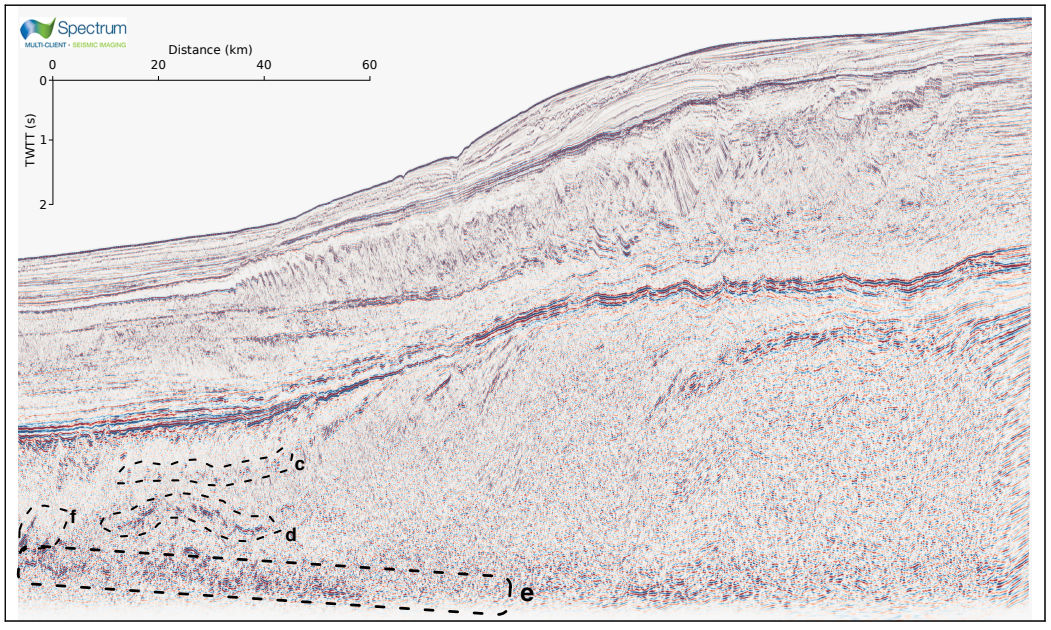


Figure 4.6: Annotated comparison of SCOB-12-ADJ2 (i) and NAMZA-14-ADJ1 (ii) seismic reflection profiles. Unannotated profiles shown in Figures 4.4 and 4.5 respectively. Features highlighted by dashed lines and labelled with letters discussed in text: a) top of transitional package, b) reflectors within SDR package, c) base of transitional package, d) irregular mid–crustal reflector, e) Moho and f) sub–vertical reflectors within oceanic crust.

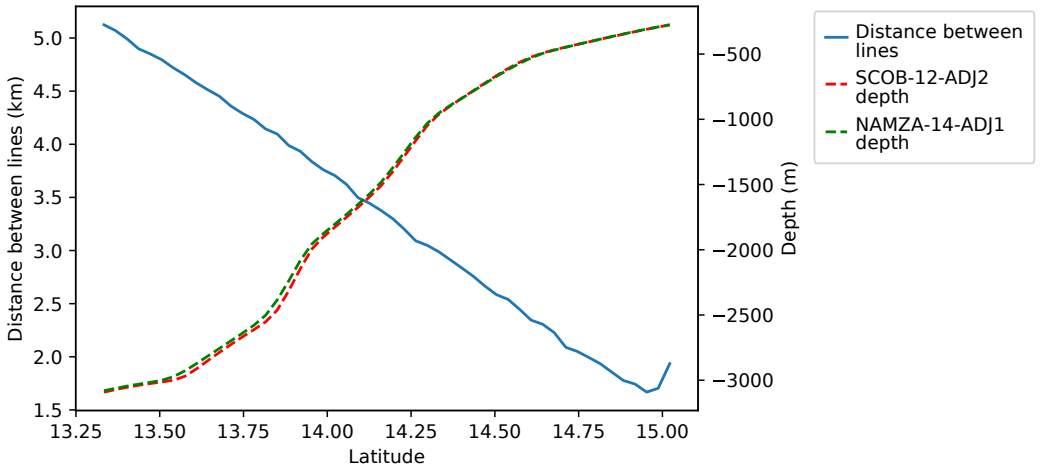


Figure 4.7: Distance between seismic lines NAMZA-14-ADJ1 and SCOB-12-ADJ2. Shown on the right axis and in dashed lines is the corresponding bathymetry for each line, extracted from the 1-minute GEBCO 2008 grid (*BODC*, 2014).

4.2.2.1 Results

The post–rift stratigraphy of these two closely spaced seismic reflection profiles is very similar, dominated in the shallow section by a mass–transport deposit. Likewise, the overall geometry of the SDR wedge in terms of width and thickness is similar on both lines. However, there are several key differences in the details of the volcanic rifting features, as highlighted in Figure 4.6. While it is possible that these differences represent true geological variations over a distance of several kilometres, it is likely that some of these differences can be attributed to the seismic imaging.

There are notable differences of interest within and adjacent to the SDR package. Within the SDR package, prominent reflectors are more frequently observed on line SCOB-12-ADJ2 (Figure 4.6(i)), particularly those extending to greater depths (around 8 s). This is an important consideration when investigating the along–strike continuity of SDRs on a 2D seismic reflection survey. With a greater number of prominent SDRs visible, we would expect to observe a greater range of measured dips when comparing these lines using the methods discussed in Chapter 2. However, the mean value from the most prominent reflectors is likely to be similar.

Between the seaward termination of the SDR package and oceanic crust, the transitional unit (discussed in Section 4.4 as 'transition package') is imaged very differently. The top of the unit is better resolved on NAMZA-14-ADJ1, whereas it is dominated by multiple energy in the SCOB-12-ADJ2 reflection profile (Figure 4.6a). There is no discernible base to the unit on SCOB-12-ADJ2, while there is a reflector at around 8.5 s depth on NAMZA-14-ADJ1 (Figure 4.6c). Beneath the transition package, a prominent but irregular reflection is visible on NAMZA-14-ADJ1 (Figure 4.6d), which is entirely absent on SCOB-12-ADJ2. The Moho, gradually dipping landward between 9–10 s, is much better imaged on NAMZA-14-ADJ1 (Figure 4.6e). Within the small section of oceanic crust visible on these two lines, two steep, sub–vertical reflections are visible at the base of the crust in NAMZA-14-ADJ1 (Figure 4.6f), which are poorly resolved on SCOB-12-ADJ2.

4.2.3 Discussion

The transition package is most susceptible to differences in imaging; with access only to the SCOB-12 survey, the base of the unit and mid–crustal reflector could be misinterpreted. Assuming these features, which are both ∼40 km in length down–dip, persist at least 5 km along strike, it is clear that absence of these features on reflection profiles does not guarantee their geological absence. It is therefore possible that the transition package may be more widespread than imaged. Other features, like the deep, sub–vertical reflections within the oceanic crust on profile (Figure 4.6b) are more likely to reflect true geological variation. Although the Moho reflection is of greater amplitude in NAMZA-14-ADJ1, the depth and dip of the Moho is similar on both surveys, and therefore not likely a major concern for interpretation.

Within the SDR package, the variation in prominent reflectors may reflect true geological variation or imaging. The continuity of single flows within the SDR package is explored later in this Chapter. In general, this variation suggests that

extreme dip measurements from within the SDR package may not be representative and should be interpreted with caution. The overall width, thickness and average dip of reflectors, conversely, appear reliable.

4.3 Identifying SDRs on a Dense 2D Grid

The combination of NAMZA-14 and SCOB-12 surveys provides a dense 2D grid of seismic reflection profiles over the outer SDR package in the Orange Basin. A typical reflection profile over the SDR package and transition to oceanic crust with annotated key features, is shown in Figure 4.8.

The SDR package discussed here is approximately 2–3 s thick (TWTT). This package is overlain unconformably by 2–4 s TWTT of post–rift sediments; this unconformity is commonly referred to as the acoustic basement (*Bauer et al.*, 2000). In some instances, both the syn– and post–rift deposits are disrupted by later intrusions, as shown on Figure 4.8.

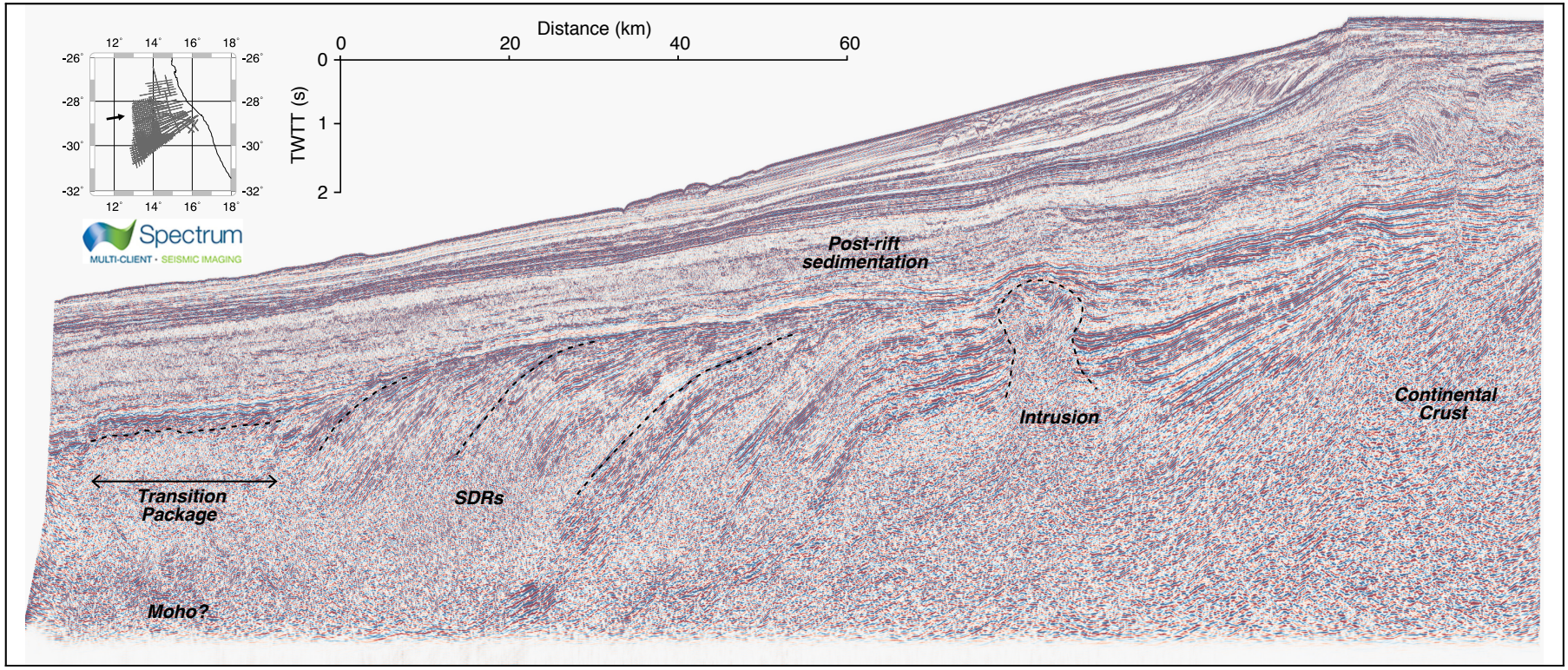


Figure 4.8: Typical Orange Basin seismic reflection profile — survey SCOB-12. Key margin features annotated in black; not horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines).

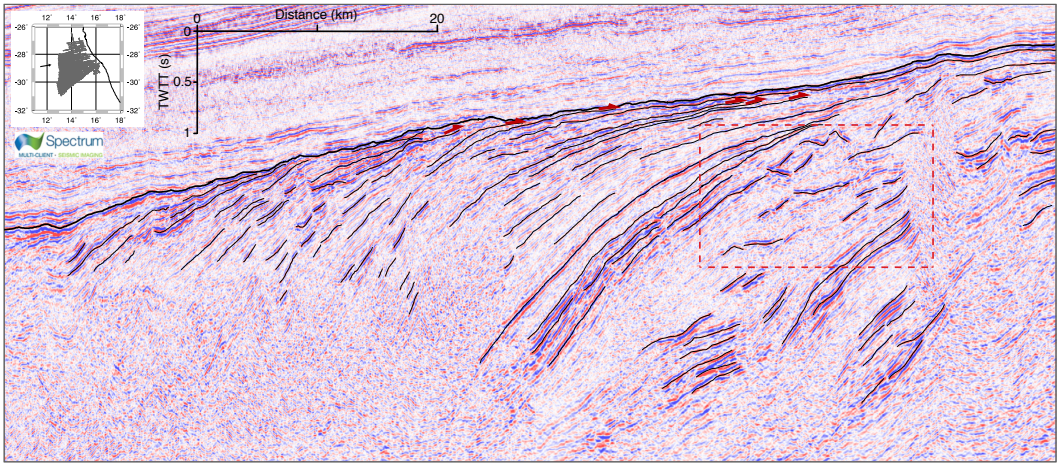


Figure 4.9: Outer SDR package line drawing — seismic reflection profile from survey SCOB-12. Note the offlapping terminations shown with the red arrows. Within the dashed red box, the arcuate SDRs are disrupted by short, sub–horizontal reflectors which may represent later intrusions. Note horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines).

4.3.1 Overall SDR Package Geometry

The term 'seaward dipping reflector', interpreted literally, may describe many features present on a passive margin; here, we restrict analysis to those which are arcuate and down–dip thickening. This distinguishes what is commonly referred to as the 'outer' SDRs, discussed here, from the 'inner' SDR, which lacks the characteristic arcuate shape.

A line drawing of an example of the outer SDR package from the Orange Basin is shown in Figure 4.9. Although not present in all seismic reflection profiles, Figure 4.9 shows that in some places the SDRs are disrupted by short, discontinuous sub–horizontal reflectors which may represent a second period of magmatic intrusion. Seaward of this, however, the SDRs are classically arcuate and down–dip thickening, with the total thickness of the package decreasing toward the transition to oceanic crust.

It is possible to investigate the variation in overall package width (down–dip) along strike of the margin by picking the most landward and seaward SDR on each dip line. For this purpose, an SDR is defined as an arcuate, down–dip thickening,

seaward dipping reflector. As the grid spacing is relatively small, this is an opportunity to investigate the degree of continuity of the SDR package along strike of the margin.

First, the most landward and seaward SDRs were picked on all reflection profiles in the dataset. Examples of particularly interesting cases are shown in Figures 4.10 and 4.11. Typically, the most seaward SDR is much easier to pick than the most landward; it is defined here as the most seaward arcuate reflector which has a thickness equal to, or greater than, the transition package discussed in section 4.4. The approximate base of this transition package is shown with a dashed line in Figures 4.10 and 4.11.

Defining the most landward SDR was more challenging. In some cases, a bowl–shaped 'sag' basin landward of the outer SDRs complicated this process, as shown in Figure 4.11. In such cases, the most landward prominent reflector that was *not* continuous with a sub–horizontal reflector within this basin was picked. Typically, SDRs within the package offlap; however, the landward–most SDR and subsequent $\sim$10 km reflectors onlap, producing a rise in the acoustic basement.

A map view of the extent of these picked horizons is shown in Figure 4.13, with the overall survey geometry for all datasets shown for reference. The most landward SDR is remarkably consistent along strike of the margin, with correlation along strike lines (e.g. Figure 4.12) possible in the middle of the survey. The position of the seaward termination of the outer SDR package is more variable.

The width of the SDR package may be calculated as the total horizontal distance on each profile between the most landward and most seaward reflectors. A histogram of these measured widths is shown in Figure 4.14. This shows a broad distribution of widths between 45–75 km, with a modal value of 60 km.

Considering both Figures 4.13 and 4.14, there is no systematic trend in width of the SDR package along the margin. However, there do appear to be smaller scale variations — for example, between latitudes $-28°$ and $-29°$, the width is much more consistent than further south on the margin, around latitude $-30°$.

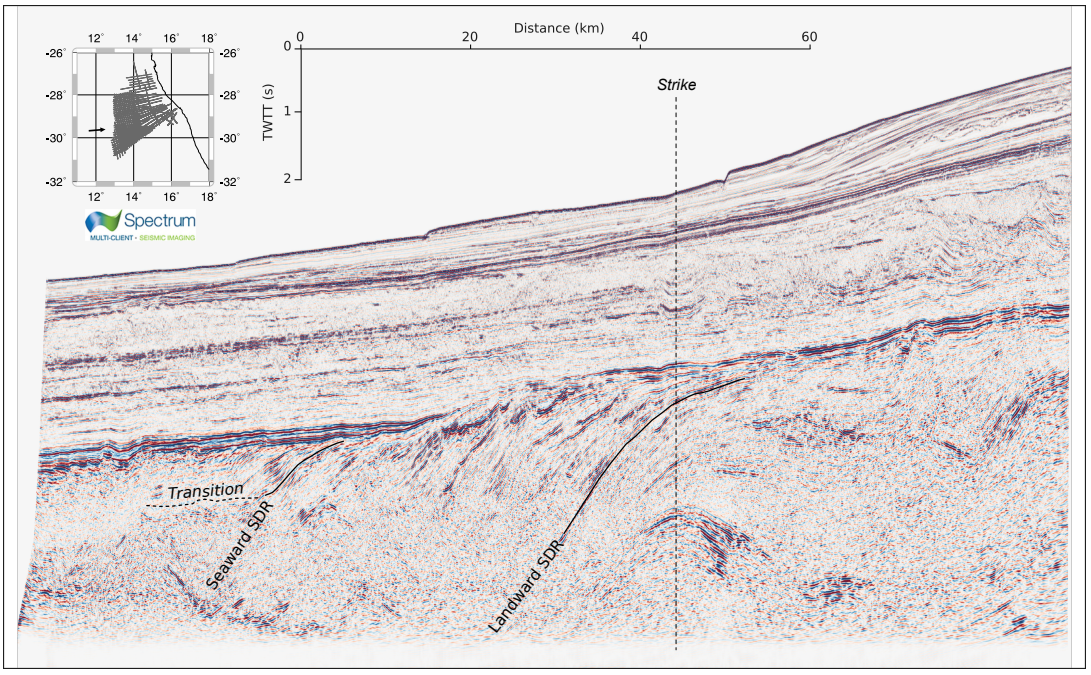


Figure 4.10: Seaward and landward SDRs definition, seismic profile from SCOB-12 survey. Note horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines). Intersection with Figure 4.12 indicated by vertical dashed line.

Considering the length of the reflectors, there is significant variation along this section of the margin, particularly in the case of the most landward SDR (shown in blue on Figure 4.17). This could reflect differences in thickness of the package, reflector dip, or both. In general, there is a good correlation between the most seaward SDR and magnetic anomaly M4.

4.3.1.1 Discussion

The mapped SDR geometry suggests that although the onset of SDR formation may be approximately synchronous along the Namibian margin, the termination of flows producing SDRs is distinctly asynchronous. However, the termination of the SDRs may not represent the initiation of full seafloor spreading — between the most seaward SDR and normal thickness oceanic crust lies the 'transition package', shown on Figures 4.10 and 4.11 discussed further in Section 4.4.

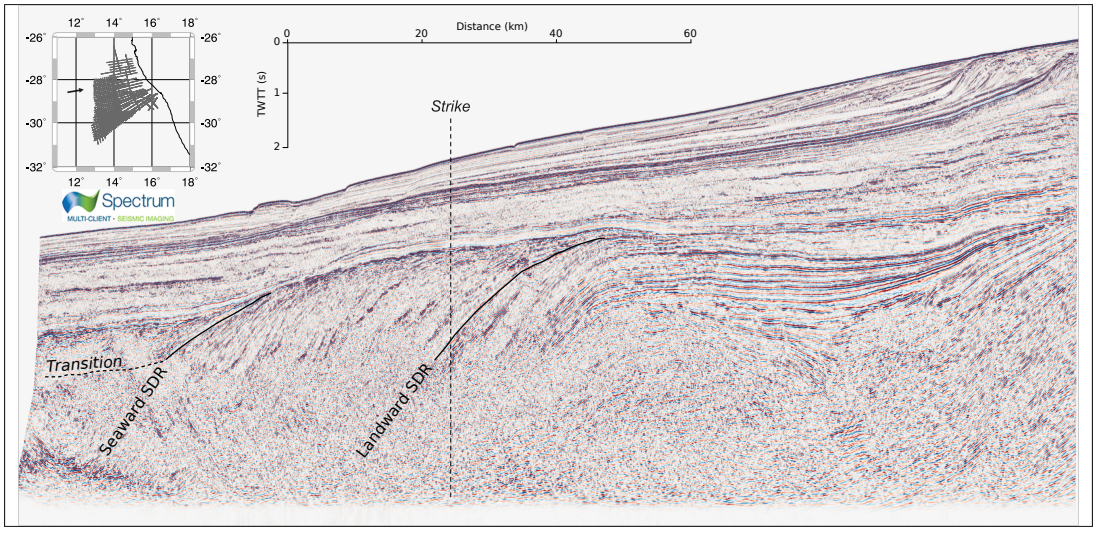


Figure 4.11: Seaward and landward SDRs definition with 'sag' basin, seismic profile from SCOB-12 survey. Note horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines). Note the bowl–shaped sag basin landward of the SDRs, not included here in the definition of most landward (see text). Intersection with Figure 4.12 indicated by vertical dashed line.

The lack of overall trend in SDR package width suggests no systematic change in effusive volume along strike of the margin; a larger scale study of the Namibian margin by *Koopmann et al.* (2014) supports this conclusion. The correlation of the SDR package with magnetic anomaly M4 (Figure 4.13, *Rabinowitz* (1976); *Collier et al.* (2017)) suggests that M4 may not represent a true seafloor spreading anomaly, as previously suggested by *Koopmann et al.* (2014).

In the north of the Orange Basin, the landward part of the SDR package shifts from onlapping to more typical offlapping stratigraphy seaward (Figure 4.11). This may suggest a change in the extrusion rate of flows forming SDRs, or change in the rate of intrusive feeder dyke injection. This aspect of stratigraphy will be revisited in Chapter 5, exploring the role of the elastic thickness (T_e) in controlling SDR formation.

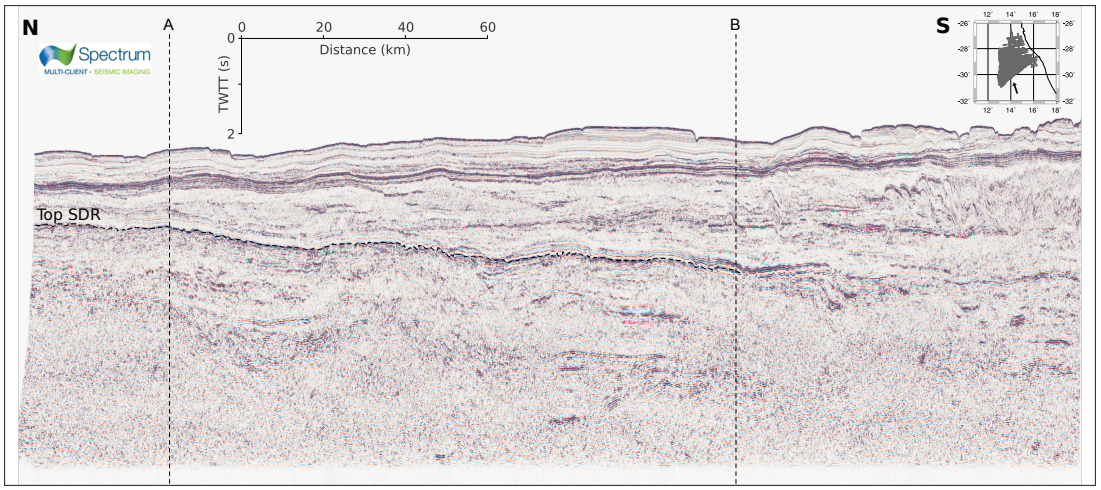


Figure 4.12: Strike line showing top SDR surface, seismic profile from SCOB-12 survey, Orange Basin, Namibia. Note horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines). Dashed vertical lines indicate the intersection with two dip lines shown in this Chapter (A, Figure 4.11 and B, Figure 4.10). The top SDR surface is also shown, dipping gently southwards.

4.3.2 Single SDR Continuity

This section explores the potential for tracing a single reflector over multiple lines on this densely spaced grid. Typically, the spacing between lines (notably not always within the same survey) is <10 km. Investigating the possibility of confidently identifying a single reflector across these spatial scales may inform discussion concerning the origin of these basaltic flows.

The spatial extent of four prominent picked SDRs in map view is shown in Figure 4.15. Two of these reflectors are entirely coincident in aerial extent with the most landward SDR (red and blue), although the most landward SDR was picked independently on each line. The longest reflector, indicated in green, can be traced ∼60 km along strike, with the aid of two strike lines. However, the other three reflectors may only be traced across a maximum of 4 dip lines, and 10–40 km along strike. Given the width of this SDR package, and the number of dip lines available, it is clear that very few SDRs can be confidently correlated across >10 km distance along–strike in the Orange Basin.

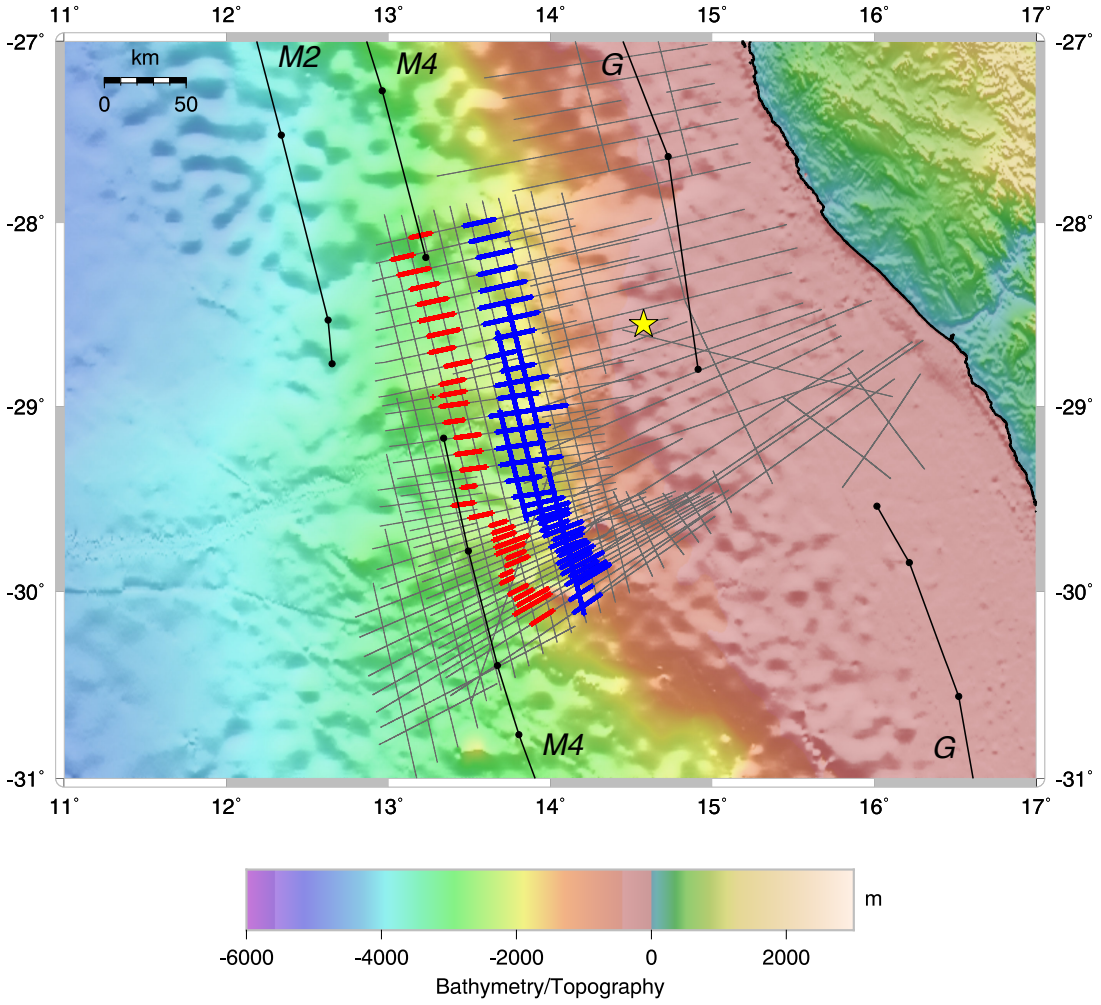


Figure 4.13: Seaward and landward SDR extent map — Orange Basin, Namibia. The most landward (blue) and most seaward (red) arcuate, seaward dipping reflectors, with all available seismic reflection profiles shown in grey. Annotated in black, dotted lines with locations of M2 ($\sim$124 Ma), M4 ($\sim$127 Ma) and the 'G' anomaly or 'LMA' (*Rabinowitz and LaBrecque*, 1979; *Collier et al.*, 2017; *Moulin et al.*, 2010). Yellow star indicates location of Kudu 9A-1 well (*McLachlan*, 1990). Bathymetry from GEBCO 2014 grid (*BODC*, 2014).

4.3.2.1 Discussion

Whether the difficulty in reliably correlating single SDRs is a limitation of the imaging quality, or a fundamental feature of the geology, remains a point for discussion. In the absence of a fully 3D seismic reflection survey, speculation is permitted as to the cause of this observation. Considering potential analogue geological settings in which basalt flow distances can be readily quantified can inform geological limits on expected flow distances under various conditions. Flood

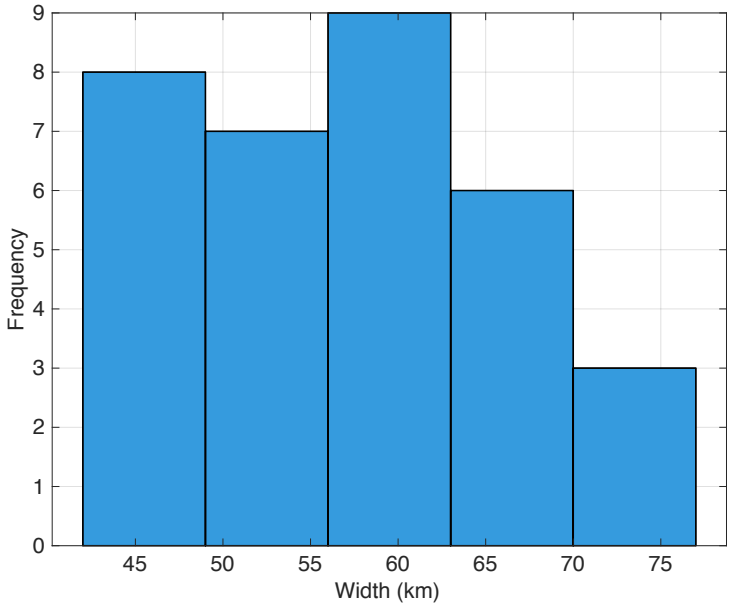


Figure 4.14: Orange Basin, Namibia, SDR package width histogram.

basalts in large igneous provinces (LIPs) can travel thousands of kilometres, limited only by topography and the size of the land mass (*Self et al.*, 2008). At Mount Etna, flow lengths of lavas range from 0–16 km, with a median value of 3.2 km (*Walker et al.*, 1973). It is therefore reasonable to expect basaltic flows which form SDRs to flow continuously over several tens of kilometres. It is perhaps surprising that we do not observe this continuity more frequently in this survey.

If this lack of continuity reflects true stratigraphic relationships, flow lengths of SDRs fall closer to the stratovolcano–like end member than the LIP analogue. This may suggest more episodic effusive activity. However, if this observation is an artefact of imaging, we would not be able to draw the same conclusion.

4.4 Transition Package

At the seaward termination of the classic arcuate, down–dip thickening package of SDRs lies a thinner, seismically transparent package that separates the SDRs from true, normal thickness oceanic crust. This package (shown in detail in Figure

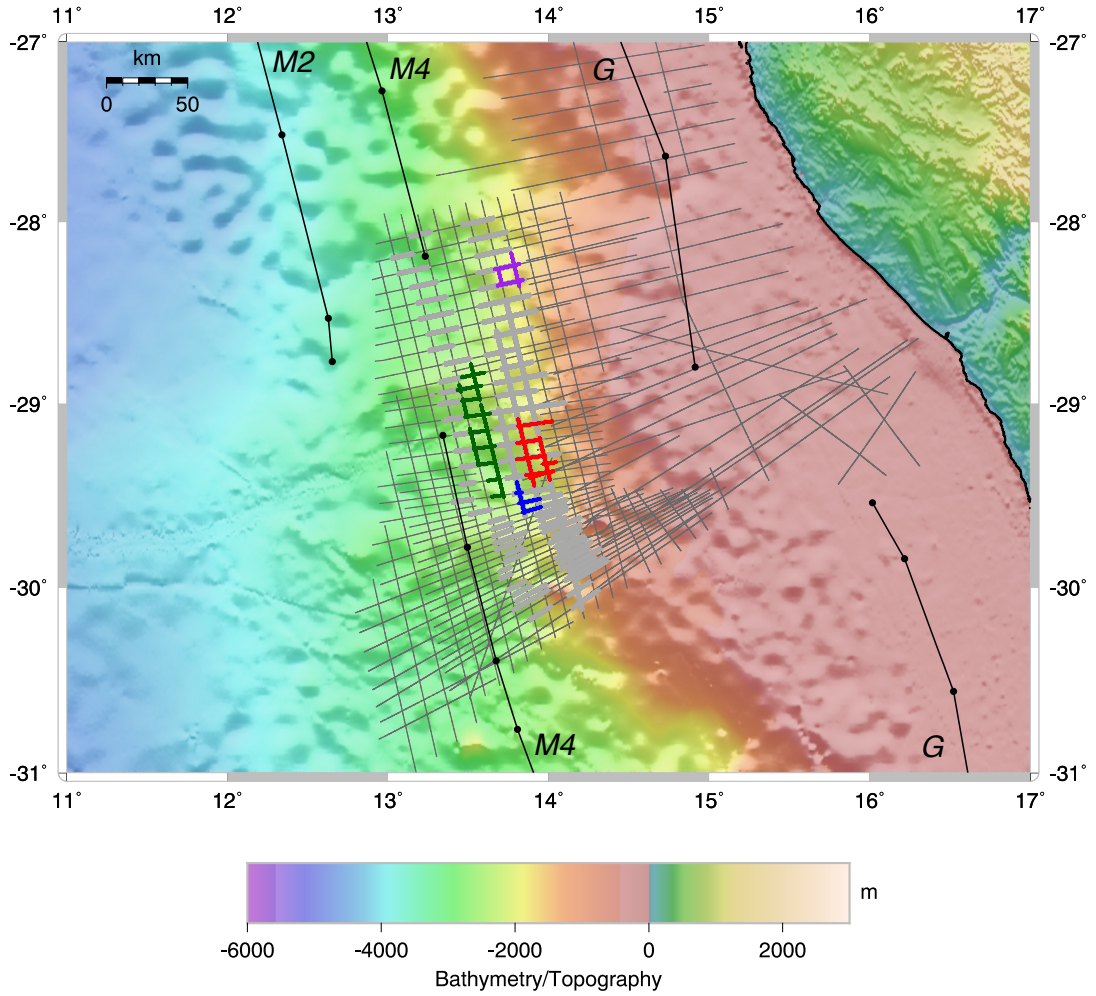


Figure 4.15: Orange Basin, Namibia picked continuous SDRs map. Thin grey lines show the overall survey geometry indicating available data, thick grey lines indicate the landward and seaward SDR horizons, shown in Figure 4.13. Green, red, blue and purple lines each represent a continuous SDR which has been correlated across multiple dip and strike lines. Annotated in black, dotted lines with locations of M2 (∼124 Ma), M4 (∼127 Ma) and the 'G' anomaly (*Rabinowitz and LaBrecque*, 1979) or large marginal anomaly 'LMA' (*Collier et al.*, 2017; *Moulin et al.*, 2010). Bathymetry from GEBCO 2014 grid (*BODC*, 2014).

4.16) is typically 0.5–1.1 s thick, 10–30 km wide (down–dip), and is common to all the seismic reflection surveys from the Namibian margin. On some profiles, there is a small scarp, offsetting oceanic crust downwards — this feature has also been described from migrated BGR profiles by *Koopmann et al.* (2014). Herein, this package is referred to as 'transition package'.

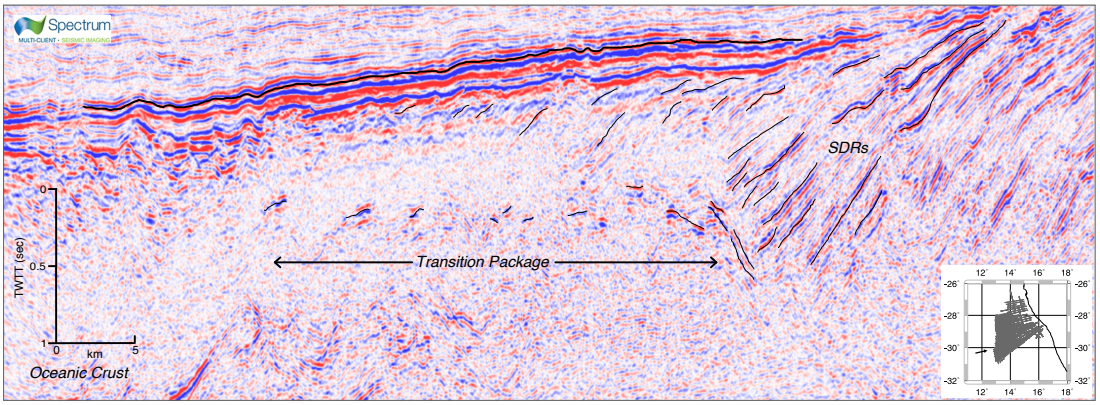


Figure 4.16: Transition package seismic image with line drawing, reflection profile from survey SCOB-12. Note horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines).

4.4.1 Detailed Description

The top of the transition package shown in Figure 4.16 is highly reflective, much more so than the unconformity which overlies the SDRs themselves and forms the base of the post–rift sedimentary sequence. There are multiples of this overlying reflector present within the package on some of the seismic reflection profiles, particularly in survey SCOB-12, which makes interpreting the internal nature of the package challenging. There appear to be some short, gently seaward dipping reflectors at the top of the package, particularly at the landward end, closest to the termination of the SDRs. Beneath this, there are few features to distinguish.

The base of the package is marked on some profiles by an irregular reflector in terms of both lateral continuity and reflectivity. A few seconds beneath this unit, the Moho is visible on many of the reflection profiles, with the crust visibly thickening landward (e.g. Figure 4.4). There is a relatively abrupt termination to this transition package unit, where it passes into classic oceanic crust of normal thickness. The highly reflective horizon at the top of the package is largely unchanged in character as it transitions to top oceanic crust.

It was noted in section 4.3.1 that the seaward limit of the outer SDR package was variable in position along strike of the margin. Figure 4.17 shows that the seaward limit of this transition package is remarkably strike–parallel. The boundary

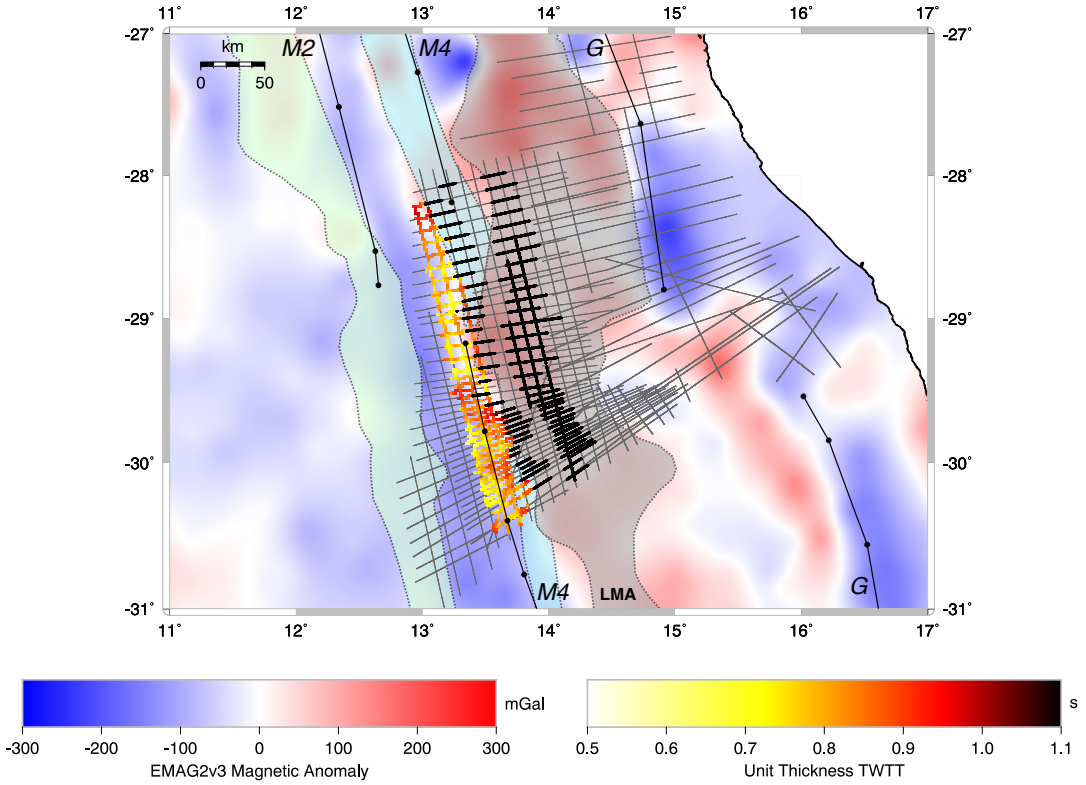


Figure 4.17: Orange Basin, Namibia transition package thickness map with EMAG2v3 magnetic anomaly (sea level). Thick black lines indicate the outcrop of the most seaward and landward SDRs, as shown in Figure 4.13. Annotated in thin, black, dotted lines with locations of M2 (∼124 Ma), M4 (∼127 Ma) and the 'G' anomaly (*Rabinowitz and LaBrecque*, 1979). Shaded polygons with dotted lines boundaries indicate anomalies of *Moulin et al.* (2010); large marginal anomaly ('LMA') in grey, M4 in blue and M2 in green.

between the SDR package and the transition package also correlates well with the seaward edge of the M4 magnetic anomaly (*Moulin et al.*, 2010). The thickness of the transition package, derived from picking the top and base horizons and shown in Figure 4.17, is inversely proportional to the width of the package.

4.4.2 Comparison with South Atlantic Margins

It is widely accepted from drilling that SDRs are subaerial in origin; this has been confirmed by drilling of the inner SDR in the Orange Basin at the Kudu 9A-1 well (*McLachlan*, 1990). Interpretation in this Chapter has focussed on the arcuate,

down–dip thickening outer SDR package; in contrast, the inner SDR package in this basin is characterised by linear flows (Figure 4.18).

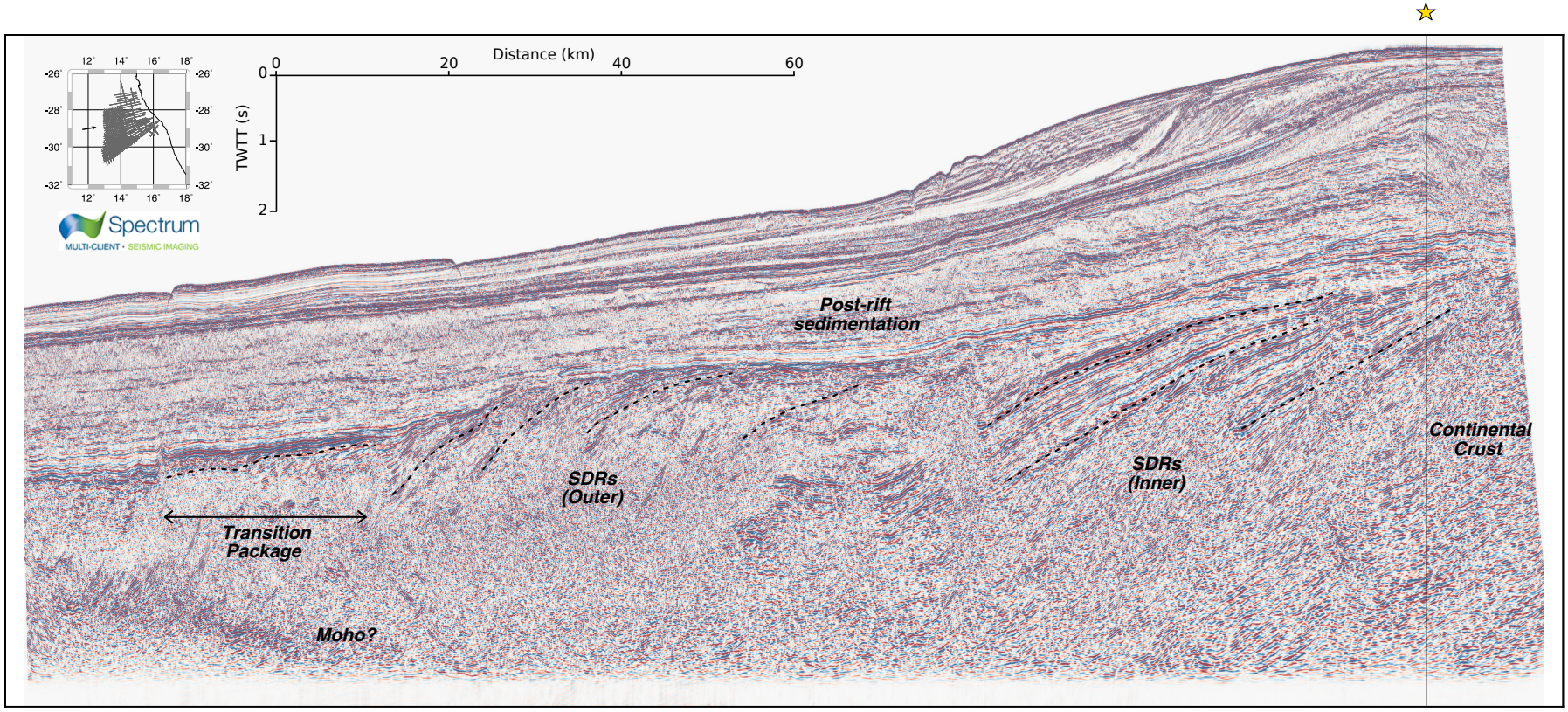


Figure 4.18: Inner and outer SDR comparison, Orange Basin over Kudu Well location, Namibia from seismic reflection profile within SCOB-12 survey. Key margin features annotated in black; horizontal and vertical scale bars and map indicating profile location (black arrow) and overall survey coverage (thin grey lines). Yellow star indicates location of well Kudu 9A-1 (*McLachlan*, 1990) also shown on Figure 4.13.

It is plausible that the transition package seaward of the outer SDRs, described in this section, represents the submarine transition in the Orange Basin — the beginning of the period during which the margin subsides, approaching the depth at which we expect the formation of normal thickness oceanic crust. The lack of clear, linear reflections within the package supports shallow submarine deposition of pillow–like basalts, rather than the longer continuous flows which could form submarine SDRs. Such continuous flows can only form either in a subaerial environment or a deep marine environment; *Gregg and Fornari* (1998) suggest a minimum water depth of 1.5 km in order to prevent degassing of volatiles and therefore permit greater flow lengths. Further, there is no clear package between the inner and outer SDRs in the Orange Basin which could represent a period of significant deepening — in Figure 4.18, the two packages are less than 20 km apart.

4.4.2.1 Comparison with Offshore Damara Belt, Northern Namibian Margin

At margins with two distinct SDR packages such as Namibia, the transition between them has typically been interpreted as the subaerial to submarine transition (*Hopper et al.*, 2003; *Franke et al.*, 2010). A seismic refraction study on the Namibian margin north of the Orange Basin in the region of the Damara Belt by *Bauer et al.* (2000) (see Chapter 7) concluded that a zone of 'flat–lying flows' between the inner and outer SDR corresponded to oceanic crust layer 2 (*Christensen and Salisbury*, 1975), with P–wave velocities in the range 4.8–5.5 km/ sec consistent with extrusive basalts. *Bauer et al.* (2000) infer that the 'flat–lying flows' represent the submarine transition, such that the outer SDR in this region must be formed in a submarine environment.

However, in the Orange Basin, we do not observe a zone of flat–lying reflectors between the two SDR packages (Figure 4.18). The zone of flat–lying flows is much thinner on the southern transect in the *Bauer et al.* (2000) study (120 km in the north, 30 km in the south); it is therefore plausible that this unit is no

longer present in the Orange Basin. This considerable along–margin thinning, and apparent absence of this unit in the surveys studied here, raises questions about the interpretation of these flat–lying flows as a characteristic marker of the submarine transition.

As the outer SDR package described by *Bauer et al.* (2000) is characterised by arcuate flows, are of comparable width (∼50 km) to the SDRs in this study, and also corresponds well with M4, we suggest that, like in the Orange Basin, this package may also be subaerial. As noted in section 4.2, the transition package is not well imaged in one of the recent surveys; it is therefore plausible that it is present seaward of the outer SDR in the Damara Belt region, but has not been imaged in the *Bauer et al.* (2000) study. A recent study on a new magnetic anomaly compilation (*Collier et al.*, 2017) found that SDRs do overlie linear magnetic anomalies, and that the anomaly geometry suggests a south to north propagation of rifting in the South Atlantic.

We therefore suggest that the submarine transition migrates seaward as rifting migrates northward, always seaward of the outer SDR package, resulting in a wider zone of igneous crust between the inner and outer SDR in the Damara belt region that is not present in the Orange Basin. It is possible that the zone of flat–lying flows recognised by *Bauer et al.* (2000) are still subaqueous, and perhaps lie in a semi–persistent lake, or developed within an area of topography that did not support long flow distances.

4.4.2.2 Comparison with Conjugate Margin — Argentina

The irregular base of the transition package is similar in seismic character to the upper crustal reflections ('UCR') described by *Franke et al.* (2010) on the South American margin, lying between the inner and outer SDRs. These reflections are irregular and hummocky, and do not mirror the overlying basement topography, much like the base of the transition package. This is interpreted as marking the submarine transition on the conjugate Argentinian margin, with the outer SDRs

therefore interpreted as submarine. The decreasing width of the UCR zone is interpreted by *Franke et al.* (2010) as increasing subsidence.

The outer SDR wedge on the Argentinian margin is much narrower (∼20 km) than in the Orange Basin, and an outer high is not always present up–dip of the SDRs. While the SDR wedge discussed in this Chapter correlates well with the M4 magnetic anomaly (Figure 4.17), no such correlation is present for the outer SDR package on the Argentinian margin. While the outer SDR package in Argentina may be submarine, it is likely that the outer SDR package in the Orange Basin is subaerial. The transition package therefore corresponds with the UCR in Argentina (*Franke et al.*, 2010), representing the submarine transition.

4.4.3 Discussion

The presence of SDRs seaward and/or landward of the submarine transition could provide insight into the rate of subsidence and spreading at volcanic rifted margins. Seismic refraction derived crustal velocities from both Greenland (*Hopper et al.*, 2003) and Namibia (*Bauer et al.*, 2000) suggest that the continent–ocean boundary lies landward of, or beneath, the inner SDR package, with all igneous crust seaward of this produced by spreading. It is clear that SDRs can form subaerially within this igneous crust, and in some locations may also form in submarine conditions.

However, the conditions under which submarine reflectors could form are unclear; assuming a depth of 1.5 km is required (*Gregg and Fornari*, 1998), *Hopper et al.* (2003) found an exceptional subsidence rate of 700 m Myr^{-1} over the period during which rough basement was formed on the Greenland margin. Alternatively, if the outer SDR formed at depths equivalent to the Reykjanes Ridge (∼900 m), which is of similar crustal thickness (*Smallwood et al.*, 1999), a smaller subsidence rate would result. Assuming the transition package does represent the submarine transition in the Orange Basin, this section of the Namibian margin never satisfied the conditions for submarine SDR formation and instead passed directly into normal oceanic crust production. This may suggest greater subsidence rates through the

window of depths under which these reflectors may form, although the absolute values are unclear.

The variation in width of the transition package along strike may represent localised differences in subsidence rates, while the strike–parallel termination of this unit shows synchronous onset of full seafloor spreading. The inverse correlation between package width and thickness suggests a constant volume of melt erupted during this final stage of rifting. Assuming this melt is produced by decompression melting, constant melt volume may therefore suggest that the stretching factor (β) has some constant value along the landward limit of the transition package, and the known terminal value of 6.24 at the seaward limit. However, as shown by the width/thickness variation, the stretching *rate* along strike is not constant. This suggests that the seaward termination of the outer SDR package is controlled by crustal thinning.

4.5 Comparison with Potential Field Data

It is possible to gain insight into the origin of significant features which may be difficult to identify in the absence of direct drilling by plotting seismic reflection profiles together with coincident gravity and magnetic anomalies. Although using potential field data alone is inherently ambiguous and non–unique, combining these data with seismic reflection profiles provides great insight into the development of the margin.

In the Orange Basin, a second wedge of seaward dipping reflectors is present landward of the package discussed in this chapter. However, unlike the outer SDR package, the inner package does not contain arcuate reflectors. Both packages are shown on a composite seismic reflection profile combining data from two surveys shown in Figure 4.19 with the corresponding isostatic gravity anomaly (calculation described in Chapter 6). It is clear that the inner and outer packages have very different gravity anomalies, in addition to this difference in geometry. Over the outer SDR package, the isostatic anomaly is at a minima at the landward edge,

rising towards a peak value, reached over the transition package. Over the inner SDR package, the local maxima is reached close to the landward termination, with a negative gradient seaward. This difference will be explored further with flexural modelling approaches in Chapter 5.

Although both packages are therefore 'SDRs' and of basaltic composition, it is unlikely they share the same formation mechanism. It has been suggested (*Barton and White*, 1997a) that inner SDRs may be fault–bond, while outer SDRs are formed by subsidence and loading (*Mutter et al.*, 1982).

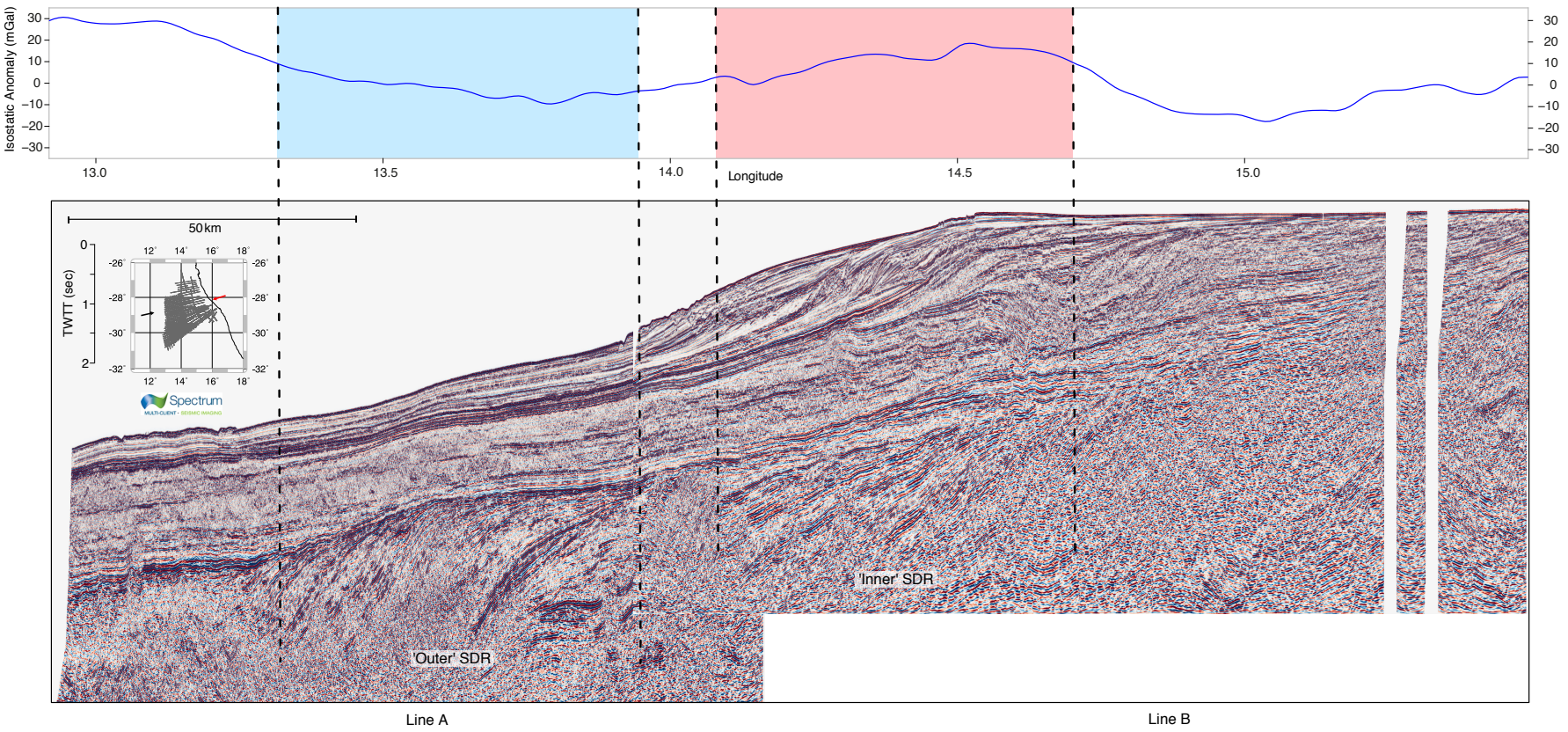


Figure 4.19: Orange Basin, Namibia composite seismic reflection profile and isostatic gravity anomaly. Two seismic reflection profiles, Line A from SCOB-12 and Line B from ECL-89 are combined here and shown with the corresponding isostatic anomaly (calculation described in Chapter 6). The blue highlighted section coincides with the outer SDR package discussed in this Chapter, while the red section coincides with the inner package. Note horizontal and vertical scale bars, map indicating location (Line A black arrow; Line B red arrow) and overall survey coverage (thin grey lines).

4.6 Chapter Summary

This Chapter focuses on a number of different aspects of VRMs, namely outer and inner SDR packages, the submarine transition and the nature of the oceanic crust. By mapping and analysing these features across multiple seismic reflection surveys with different acquisition and processing histories, the following insights into SDR imaging and the development of the Namibian margin can be obtained:

- Although the overall geometry of the outer SDR package is consistently imaged across different surveys, details in associated features and particularly within the transition to oceanic crust may not always be observed.
- The seaward termination of the SDR package is variable in position along strike of the margin, although the marked transition to normal thickness oceanic crust seaward of the transition package is more consistent.
- The width of the (outer) SDR package in the Orange Basin varies between 45–75 km, with no overall trend, suggesting no systematic change in effusive volume along strike of the margin.
- It is rarely possible to correlate any single SDR over a distance greater than 10 km along–strike; it is unclear whether this is a limitation of 2D seismic reflection surveying or a geological observation.
- A package, here named transition package, is observed between the seaward termination of SDRs and normal thickness oceanic crust. This package varies in thickness between 0.5–1.1 s TWTT in the Orange Basin and likely represents the submarine transition in the Orange Basin.
- The seaward termination of the outer SDR package correlates well with the M4 magnetic anomaly; this is also observed elsewhere on the Namibian margin (*Bauer et al.*, 2000).
- A second, inner SDR package is present in the Orange Basin, but reflectors are linear, not arcuate, with a different isostatic anomaly signature, suggesting the two packages are formed by different mechanisms.

Chapter 5

Flexural Modelling

A substantial part of the work presented in this Chapter has previously been published in *Morgan and Watts* (2018); the text has been modified to more clearly demonstrate the development of this model and expanded upon for the purpose of this thesis.

5.1 Theoretical Introduction to Flexure

The concept of lithospheric flexure has been used in a wide array of geologic contexts in order to better understand the strength of the Earth's lithosphere through time. In general, on timescales greater than 1 Myr (Figure 5.1), the Earth's steady–state response to an imposed load can be modelled by treating the upper part of the lithosphere as an elastic plate overlying an inviscid fluid. The thickness of the elastic plate, T_e which produces the best–fit flexure to the observed crustal deformation and/or gravity anomaly, is characteristic of the strength of the lithosphere. It is important to note that the T_e is not a direct measure of any particular seismic or lithological boundary within the lithosphere, and typically incorporates the crust and uppermost mantle.

When the crust is subject to a load, there are two end–member responses between which flexure operates. In the zero strength case, we expect an Airy isostatic response — namely, deformation is entirely localised directly beneath the load.

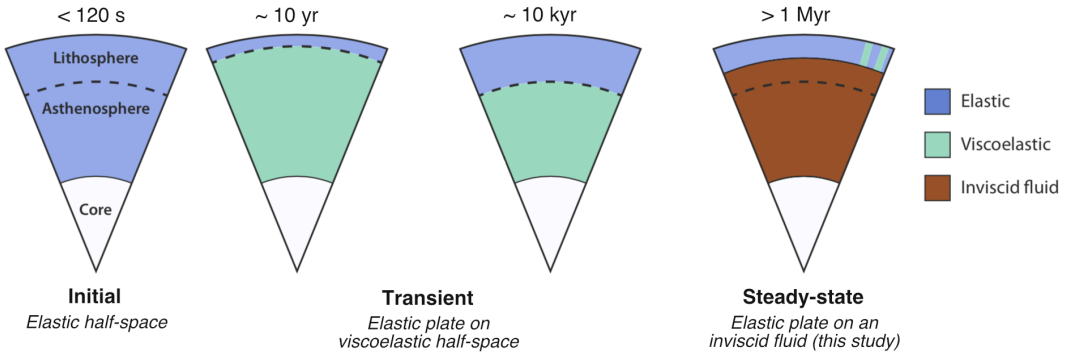


Figure 5.1: Schematic of the Earth's response to deformation on different timescales, modified from *Watts et al.* (2013).

Symbol	Variable Name
g	acceleration due to gravity
G	gravitational constant
$h(x)$	topography
$y(x)$	flexure
E	Young's modulus
D	flexural rigidity
ν	Poisson's ratio
ρ_w	density of water
ρ_m	density of mantle
ρ_c	density of load
ρ_{infill}	density of infill
ρ_c	density of crust

Table 5.1: T_e parameters — variables for flexure and gravitational derivations and calculations

Conversely, in the case where the crust is infinitely strong (Bouguer response), there is zero deformation. In all intermediate cases, in which the crust has finite strength, we observe flexure of characteristic wavelength (Figure 5.2). The lithosphere appears infinitely strong at short wavelengths and infinitely weak at long wavelengths, irrespective of its actual strength.

The nature of the flexural response differs on both temporal and spatial scales. Considering a simplified case, the response of a 2D elastic plate overlying a weak fluid substratum to a periodic load of wavenumber k, height h and density ρ_c has associated displacement described by equation (5.1). Variables in the following derivations are defined in Table 5.1.

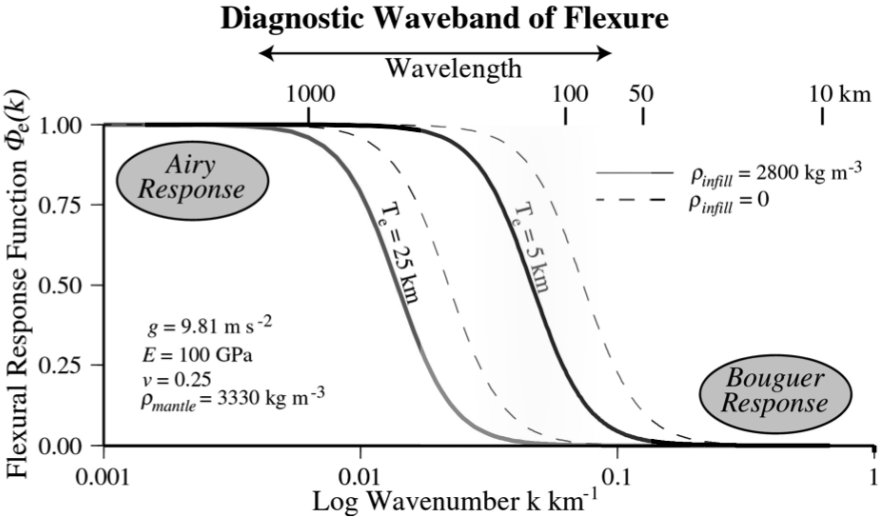


Figure 5.2: Diagnostic waveband of flexure, modified from *Watts* (2001) p.178. Note dependence of the flexure on the T_e of plate and the densities and wavelength of the load. Dashed lines represent an infill density of zero, while solid lines indicate an infill density of $2800\,\mathrm{kg\,m^{-3}}$ respectively for each T_e.

$$D\frac{\partial^4 y}{\partial x^4} + (\rho_m - \rho_{infill})yg = (\rho_c - \rho_w)gh\cos(kx) \tag{5.1}$$

This may be solved by substitution, generating a periodic solution:

$$y(x) = \frac{(\rho_c - \rho_w)h\cos(kx)}{(\rho_m - \rho_{infill})}\left[\frac{Dk^4}{(\rho_m - \rho_{infill})g} + 1\right]^{-1} \tag{5.2}$$

where $y(x)$ indicates flexure and x horizontal distance. The relationship between flexure and Airy isostasy is clear in the limit when the flexural rigidity $D \to 0$; the plate has zero strength and the Airy response (local compensation) is recovered.

$$\lim_{D\to 0}[y(x)] = \frac{(\rho_c - \rho_w)h\cos(kx)}{(\rho_m - \rho_{infill})} \tag{5.3}$$

Another case of interest, the limit of $D \to \infty$, represents the Bouguer response for an infinitely strong plate.

$$\lim_{D\to \infty}[y(x)] = 0 \tag{5.4}$$

For any intermediate value of D, the plate has finite strength and responds by flexure. The relationship between the flexural rigidity and the effective elastic thickness was stated by *Walcott* (1970) (equation (5.5)).

$$D = \frac{ET_e^3}{12(1-\nu^2)} \tag{5.5}$$

The *flexural response function*, $0 \leq \Phi_e(k) \leq 1$ (*Walcott*, 1970), is defined as a parameter which determines the flexure modifying the Airy response within the diagnostic waveband for a given wavenumber k (Figure 5.2). To derive this, the lithosphere is considered as a linear filter, with an input of the Airy response (5.3) and output of the observed flexure (equation (5.2)).

$$\Phi_e(k) = \frac{Output}{Input}$$

By substitution of equations (5.3) and (5.2), we can derive the flexural response function:

$$\therefore \Phi_e(k) = \left[\frac{Dk^4}{(\rho_m - \rho_{infill})g} + 1\right]^{-1} \tag{5.6}$$

Flexure is only significant at wavelengths within this diagnostic waveband, resulting in regional compensation to the imposed load (or deformation).

5.1.1 Elastic Thickness in the Oceans and Continents

Ocean basins have a strong age – depth dependence, demonstrated by the cooling plate model for thermal evolution of oceanic lithosphere (*Parsons and Sclater*, 1977). First identified by *Watts* (1978), the T_e of oceanic lithosphere corresponds with the 450 °C isotherm, shown in Figure 5.3. This correlation results in a thicker elastic thickness, and therefore stronger crust, with age. A similar relationship between temperature and T_e has been proposed in the continents using data from

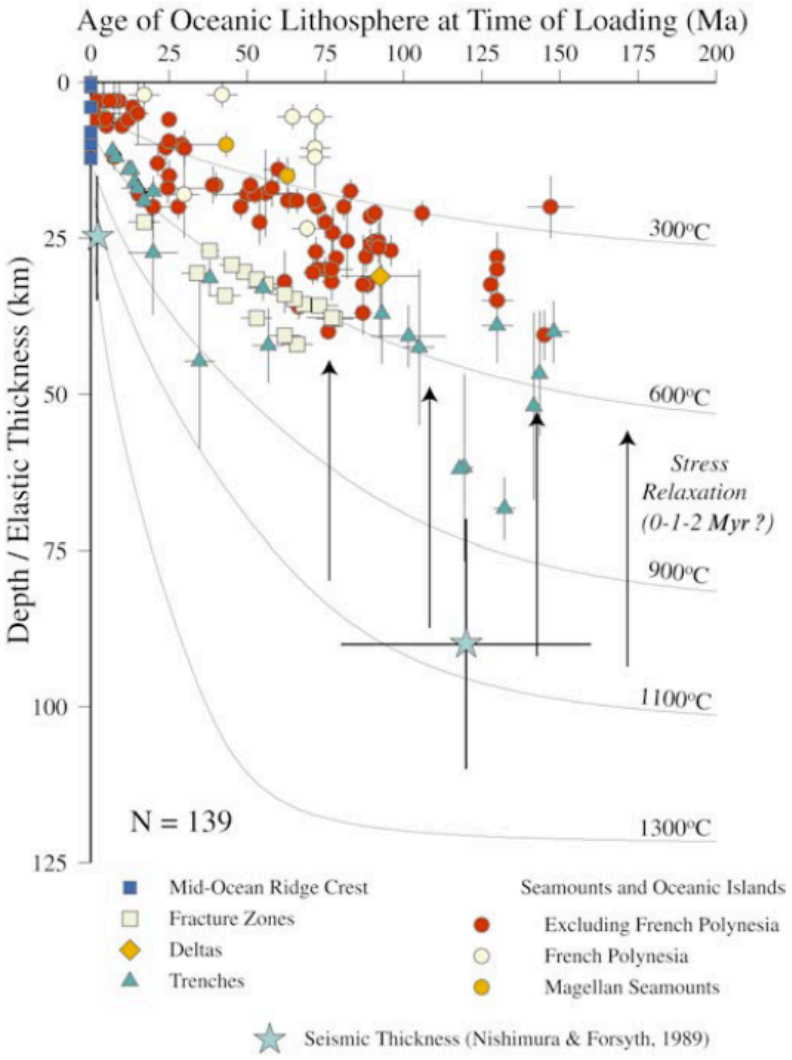


Figure 5.3: Global T_e against age of oceanic crust at the time of loading, reproduced from *Watts and Zhong* (2000). Blue star indicates the seismic thickness; on such short timescales, the plate retains more strength in the upper mantle. Over a timescale of several million years, this relaxes to produce the long–term correlation with the 450 °C isotherm.

the Canadian Shield (*Pinet et al.*, 1991), although the heterogenous nature of the crust and, in particular, the role of radiogenic heating makes this more challenging.

When considering the formation of SDRs, the nature for the crust in question is unclear. The location of the continent–ocean boundary (COB) is often poorly known at VRMs, such that the crust underlying SDRs may be purely continental, heavily intruded continental, or perhaps entirely magmatic in origin. If magmatic, the crust would be oceanic in composition and of approximately zero age

at the time of loading, but maybe significantly thicker than normal oceanic crust. Determining the T_e in this setting could provide useful insight into this problem.

5.1.2 Potential SDR Analogues

5.1.2.1 Oceanic Islands

A classic application of the flexural loading model is that of volcanic oceanic islands, or seamounts. These typically isolated structures can form on the interior of oceanic plates and are perfect test cases for flexure. The Hawaiian–Emperor seamount chain, for example, which lies on the Pacific plate far from the influence of plate boundaries, provides multiple loads of varying size emplaced at different times over the last 80 Ma on crust of approximately 80–90 Ma (*Van Ark and Lin*, 2004).

The seamount edifice constitutes a load on the lithosphere, causing flexural subsidence, which results in flanking depressions, followed by bulges. This can be clearly observed in free air gravity observations over the islands (*Watts*, 2001). The distance from the load to the flexural node, between the depression and the bulge, is indicative of the elastic thickness of the underlying lithosphere.

Seismic reflection data over the moats flanking surface volcanic loads at hot–spot generated seamount chains show a layered sequence of volcaniclastic deposits (Figure 5.4), thickening toward the edifice. Although these moats are typically much thinner and of shallower dip than SDRs, the arcuate geometry and down–dip thickening observed in moats are two of the defining SDR stratigraphic characteristics (Chapter 4). *Watts and ten Brink* (1989) modelled the stratigraphy observed in the flanks of the Hawaiian Islands using a progressive loading flexural model, finding a best–fit effective elastic thickness (T_e) of 40 km. This value is higher than that expected given the age of the oceanic lithosphere (80 Ma), which the authors suggest may be due to subsurface (buried) loading at the base of the flexed crust. This load acts to 'unbend' the surface loaded plate, resulting in the plate appearing stronger than its true value.

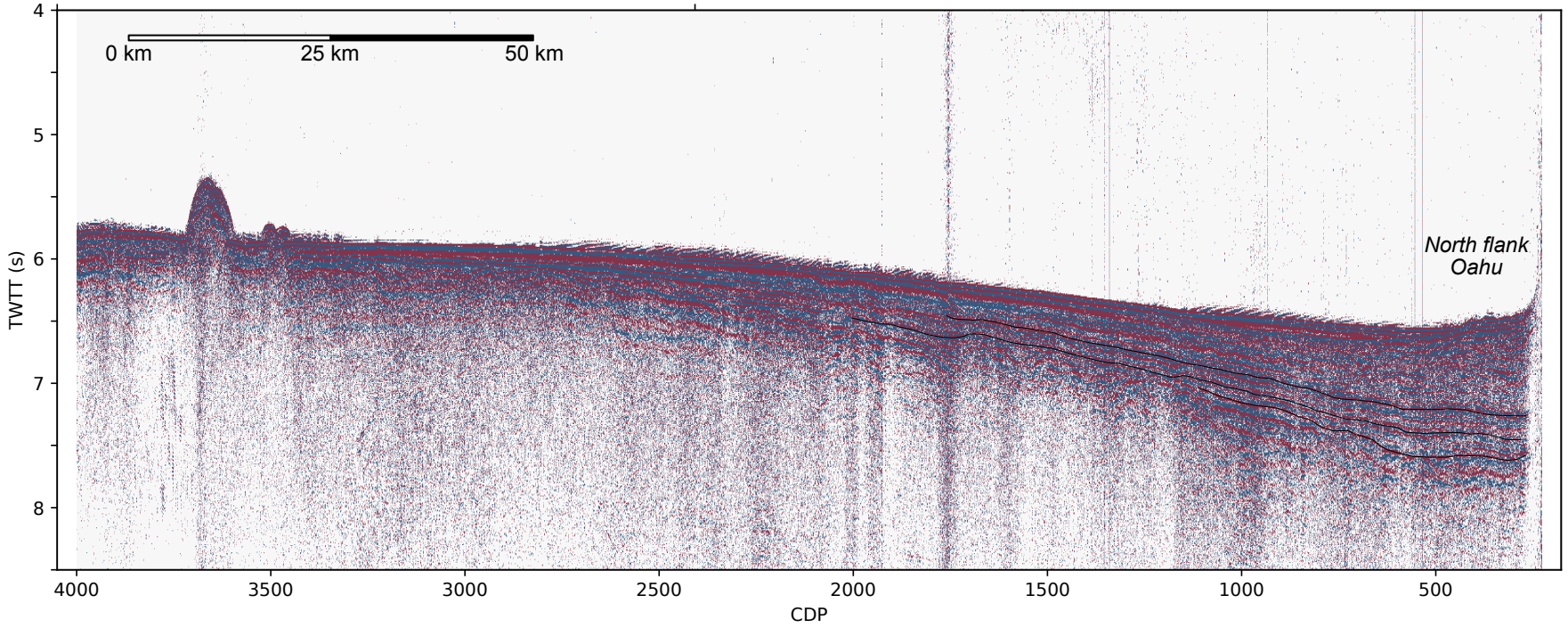


Figure 5.4: Annotated seismic reflection profile over seamount moat, Oahu, Hawaii. Originally published in *ten Brink and Watts* (1985), reflection profile 314 obtained by the RV Conrad in August 1982 (RC2308). Located north of Oahu, with scale bar shown, annotated to show reflectors within the moat with stratigraphic similarities (namely, down–dip thickening) to SDRs.

5.1.2.2 Large Igneous Povinces

A similar progressive flexural loading model has been invoked to explain the geochemically defined stratigraphy in Large Igneous Provinces (LIPs). *Watts and Cox* (1989), for example, modelled the stratigraphy of the Deccan LIP by assuming a migrating volcanic surface load superimposed on an elastic plate that responded by flexure. The load was assumed to have the approximate dimensions of La Reunion where the plume that generated the flows is believed to be presently located. The 'Bombay high' beneath the West Indian shelf and the Chagos–Laccadive Ridge, which links La Reunion to the Deccan lava flows, mark the surface expressions of this load. There is little evidence, however, in the deep seismic structure of volcanic rifted margins for the equivalent surface loads that could have generated SDRs.

5.1.2.3 Discussion

The stratigraphic similarities noted between these settings in which material infilling flexure–driven subsidence and SDRs led us to explore a surface loading driven model. Although down–dip thickening can be readily explained by syn–depositional fault movement as an alternative to flexure, the arcuate shape of SDRs cannot be. Assuming basalts do not flow up significant gradients ($\geq 5°$), this progressive increase in dip must be due to a rotation acquired after deposition. Flexure due to subsequent loading in the region *can* produce this rotation.

5.2 Role of Surface Loading

Established geological analogues demonstrate that the flexure associated with volcanic loading can produce stratigraphy that resembles SDRs. In order to explore the potential for surface volcanism as the driving load producing SDRs, we model a 3D elastic plate onto which we built a succession of volcanic cones. The flexure associated with the cone load, and the material which infills this flexure, is

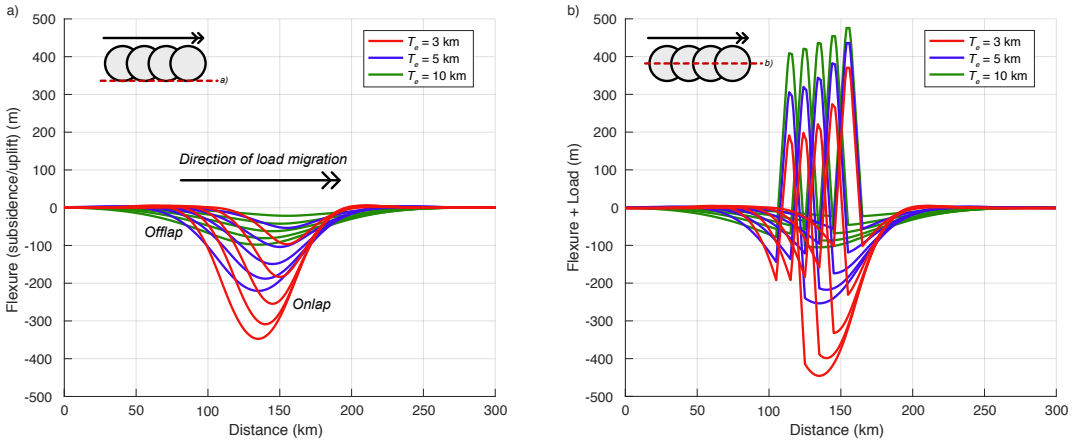


Figure 5.5: Cone surface load migration and associated flexure for different T_e values – a) over the flanks within the flexural moat and b) over the peak of the surface load cone. In each figure, an indicative top–down view shows the relative location of the cross–section shown with the red dashed line, with direction of migration indicated by the black arrow. Cone flattened height 500 m with diameter 20 km and the following densities: load density (ρ_l) 2800 kg m^{-3}, infill density (ρ_i) 2400 kg m^{-3}, crustal density (ρ_c) 2800 kg m^{-3} where the load and infill both displace air (assuming subaerial deposition). The T_e value for each example of 5 horizons is shown in the legend.

calculated using a Fast Fourier Transform (FFT) method. Figure 5.5 shows the stratigraphy across the flanks and peak (a and b respectively) of a migrating 500 m high flattened cone load. This model assumes an elastic plate of constant thickness and a constant rate of migration of the volcanic load across this plate.

Over the flanks of the load, the stratigraphy is clinoform–like and also resembles SDRs. Offlap is observed in the upper part of the package 'behind' the migrating load, whereas in the direction of load migration, onlap is observed at the bottom of the package. Progressive offlap is consistent with observations of 'trains' of SDR packages; however, the onlap and associated change in gradient of the modelled reflector at the base of the package is not observed in SDRs. Although the deepest part of the SDR package, typically with the steepest dips, is the most difficult to image, the lack of onlap suggests surface loading alone may not be responsible for SDR formation.

Even for very low T_e values <5 km, the magnitude of flexure is much smaller than the typical thicknesses of observed SDRs (>1 km, see Chapter 2). Further,

the surface loads driving the flexure, required by this model, are not observed in the geological record; significant subaerial erosion of this material would therefore be required. While we do not exclude the possibility that surface loads could contribute to the development of SDRs, it is clear that surface loading alone is not sufficient to provide the required subsidence. It has been suggested that feeder dykes are found beneath SDRs (*Mutter et al.*, 1982; *Paton et al.*, 2017); such high density bodies will act as buried loads in the same fashion as volcanic cone loads on the surface. We therefore explore instead a dyke–driven flexural model further in this Chapter.

5.3 Dyke–Driven Flexural Model

SDRs are believed to comprise high extrusion rate lava flows that formed at a subaerial spreading centre, during the early stages of continental rifting (*Mutter et al.*, 1982; *Planke and Eldholm*, 1994). The flows are fed by dykes, which are focused at the spreading centre. As extension proceeds, flows further from the spreading axis are progressively loaded by younger flows, overlapping them and initially producing a landward dip, which is then rotated to form a seaward dip. Such a model does not require faulting and instead creates the accommodation space into which the volcaniclastic sediments and basaltic lava flows accumulate by surface loading.

5.3.1 Summary of Existing Models

Recent approaches to modelling SDR formation have suggested that the arcuate shape of reflectors may also be of flexural origin. *Corti et al.* (2015) showed that downward flexure resulting from a combination of internal (buried) and external (surface) loading can tilt initially horizontal flows toward the injection axis. *Paton et al.* (2017) sequentially restored SDR packages from an ION Geophysical seismic reflection profile offshore Argentina to demonstrate the progressive rotation of

volcanic flows. In their conceptual model, this rotation is driven by a combination of flexural loading due to extrusive igneous material that was fed by sheeted dykes in the upper crust and magmatic intrusions in an over–thickened lower crust.

Whereas *Paton et al.* (2017) consider the general state of isostatic balance of oceanic and continental crust, *Buck* (2017) proposed a model in which subsidence is driven only by dykes which act as intra–crustal loads. These sub–surface dyke loads intrude the pre–rift crust and cause a surface flexure into which volcaniclastic sediments and basaltic lava flows accumulate, forming SDRs. *Buck* (2017) used analytical expressions for the flexure of a semi–infinite (i.e. broken) elastic plate overlying an inviscid fluid due to a line load at the plate break to show that surface flexure due to dyke loads could explain the stratigraphic 'architecture' of individual SDR packages, in particular their offlap patterns and down–dip thickening.

5.3.2 Introduction to Model

We consider here a similar model to that of *Buck* (2017). The main difference is that we use a Finite–Difference Method (FDM) of solving the general equation for plate flexure (e.g. *Contreras-Reyes and Osses* (2010). By using FDM we are able to model distributed rather than single line loads as modelled by *Buck* (2017). We are also able to consider spatial as well as temporal variations in T_e during SDR formation, as would be expected to occur, for example from numerical models of the dynamics of rifting (*Burov and Poliakov*, 2001). Finally, we consider both infinite and semi–infinite (i.e. broken) plate flexure. In the latter case, the plate break is simulated by $T_e = 0$ at the point of intrusion.

5.3.2.1 Finite–Difference Method

Following *Contreras-Reyes and Osses* (2010), consider the ordinary differential equation for the deflection of an elastic plate by an applied load with a known horizontal force (F) and bending moment (M):

$$-\frac{d^2M}{dx^2}+\frac{d}{dx}\left(F\frac{dw}{dx}\right)+\Delta\rho wg=q\left(x\right) \tag{5.7}$$

where w is the deflection of the plate surface, x is the horizontal coordinate, g is acceleration due to gravity, $\Delta\rho=\rho_{compensation}-\rho_{infill}$ is the density difference which provides the restoring force and $q(x)$ is the load acting on the plate. Typically, $\rho_{compensation}=\rho_{mantle}$, except where T_e is much smaller than the crustal thickness.

In order to vary elastic thickness, equation 5.7 may be re–written as a system of first order equations, as follows:

$$\frac{dw}{dx}=z \tag{5.8}$$

$$\frac{dz}{dx}=-D^{-1}\left(x\right)M \tag{5.9}$$

$$\frac{dM}{dx}=R \tag{5.10}$$

$$\frac{dR}{dx}=-F\left(x\right)D^{-1}\left(x\right)M+\Delta\rho gw-q\left(x\right) \tag{5.11}$$

Subsequently, this system can be condensed into a form which may be discretised using centered differences:

$$\frac{dS}{dx}+P\left(x\right)S=Q\left(x\right) \tag{5.12}$$

where $S=\left(w,z,M,R\right)^t$, $Q\left(x\right)=\left(-q,0,0,0\right)^t$ and

$$P = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & D^{-1}(x) & 0 \\ 0 & 0 & 0 & -1 \\ -\Delta\rho & 0 & F(x)\,D^{-1}(x) & 0 \end{bmatrix}.$$

Once discretised at N equidistant points, the unknowns in the centered scheme may be re–arranged as a tridiagonal block matrix. This $4N$ x $4N$ matrix is solved for S, from which the flexure $w(x)$ and bending moment $M(x)$ can be found at each point in the interval. The bending moment $M(x)$ allows us to calculate the curvature. We solve using the Numpy `linalg` module, employing the LAPACK routine `_gesv`. *Hunter* (2016) tested this numerical approach against the analytical solution for a broken elastic plate loaded by a distributed rectangular load (*Hetényi*, 1946) and found the RMS error between the two solutions, as a percentage of the maximum deflection (PRMS error), is $< 10^{-2}\%$.

5.3.2.2 Modelling Approach

Our model comprises two main stages (Figure 5.6, detailed geometry shown in Figure 5.7). In the first stage, a dyke with a specified height, width and density, ρ_{load}, is emplaced at the axis of spreading, with the dyke's flat top set at zero depth. We assume that the dyke is the load which drives the initial flexure. The initial flexure is calculated assuming an air infill (i.e. $\rho_{infill} = 0$) in accord with the observations in ODP site 642 (*Planke and Eldholm*, 1994) and DSDP sites 342 (*Scientific Party*, 1976) and 553 (*Scientific Party*, 1984) of cores with sub–aerial, weathered flow tops in SDR packages. In the second stage, the initial flexure is infilled to the assumed horizontal pre–deformation surface by volcaniclastic sediments and basaltic lava flows, (i.e. $\rho_{infill} \neq 0$) which drives further flexure. The total subsidence for this combined dyke and infill load represents a single SDR flow unit. After calculating the flexure, the crust is extended by splitting the model at the spreading axis and intruding the next dyke load, which will drive further flexure and form a second SDR flow unit.

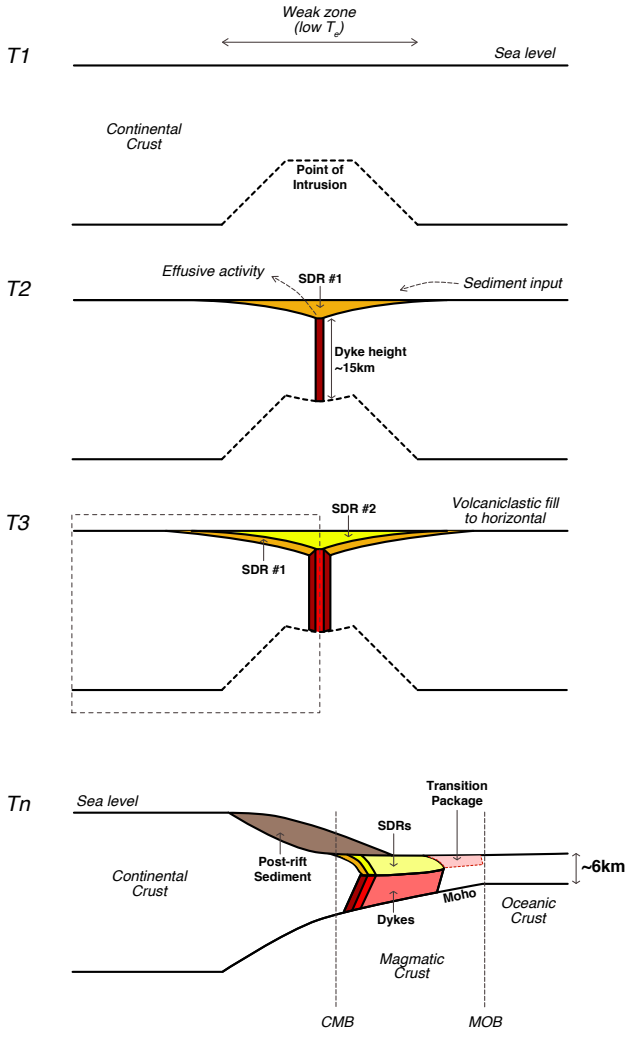


Figure 5.6: Full schematic of progression of dyke–driven SDR model, shown at four distinct time steps. At step *T1*, we identify a zone of weakness which will localise dyke intrusion. Adjacent to this zone on each side is normal continental crust. At step *T2*, the first dyke is intruded, with the associated flexure and infill forming the first SDR. This volcaniclastic fill is a combination of effusive activity and sediment input from the continental interior. At step *T3*, a second dyke has intruded, splitting the first dyke and SDR package, and forming a second. Again, the volcaniclastic infill fills the flexural depression to the horizontal surface. After many dykes have intruded, and rifting has progressed, step *Tn* shows the left hand side (indicated by the dashed grey box on *T3*) only in its final configuration. Key features discussed in this thesis are labelled, and the first two dyke and SDR packages shown in steps *T1-3* are distinguished as the oldest, and most seaward. Of particular interest is the distinction between the transition from continental to magmatic crust (CMB) and magmatic to oceanic crust (MOB), rather than a clear and distinct continent–ocean boundary (COB).

We assume that the material infilling the flexure is of uniform density at each time step. The infill in the region of the subsidence will differ, however, from the infill of

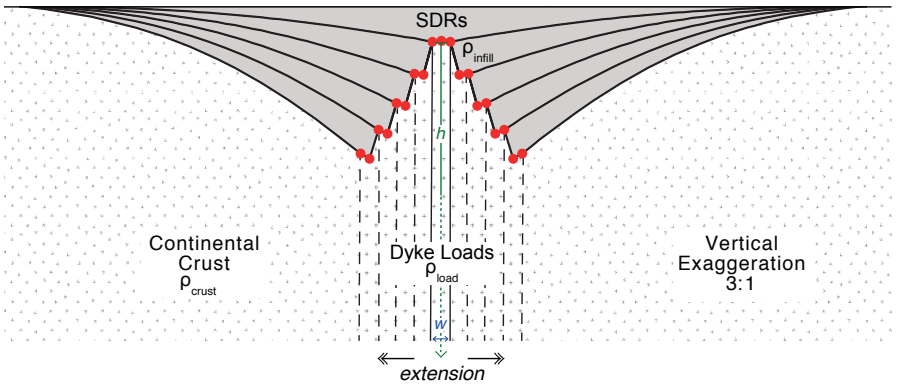


Figure 5.7: Simple model of flexure for the formation of SDRs in volcanic rifted margins. Sub–surface dyke loads intrude the crust and cause a surface flexure that is subsequently infilled by volcaniclastic and basaltic lava flows (shaded grey). The red circles show the tilting of the dyke tops that occurs as a result of progressive dyke loading and associated flexure, with the vertical dyke boundaries shown in dashed lines. The dyke tops are rotated as flexure progresses, such that the sides of the dykes may also be rotated. However, this cannot be accurately resolved within our thin elastic plate model.

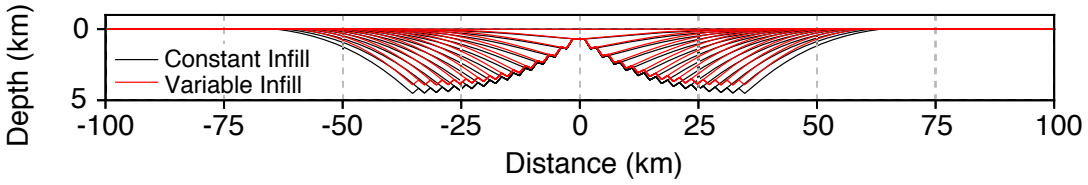


Figure 5.8: Pre–rift stratigraphy comparing the end–member cases of constant infill across the model domain (black) with variable infill (red). Note the change in width and thickness of the package, with the two model cases diverging for increasing numbers of loads.

the uplift regions; this is difficult to model without iterating for the flexure at each stage as it is dependant on the elastic thickness. In the first dyke–driven stage, the infill used in the subsidence (air) is the same as in the uplift regions and therefore our assumption is valid. However, in the second stage, where volcaniclastic infill is used, this is no longer true, subduing the flexural bulge and increasing the width of the SDR package. Alternatively, we may use non–zero infill values in the subsided areas at the infill driven stage, and use the density of air in all other regions. In this case, we underestimate the overall infilling density and therefore the size of the package as the total subsidence is wider than the initial load driven subsidence. The difference between these two end member cases for the typical model inputs is, however, small (Figure 5.8).

The least well constrained input parameter in our model is the dyke height, which defines the load and hence is a major control on the thickness of the SDR package. The total thickness is determined by the height and width of the load, the densities of infill, ρ_{infill}, substrate, ρ_{mantle}, dyke load, ρ_{load} and host rock, ρ_{crust}, and T_e. As densities can be prescribed within plausible ranges (Table 5.2) and T_e is constrained by the width of the package, we can use the observed thickness of the SDR package to inform the choice of dyke height. We assume here 15 km for the dyke height, which is of the order of the seismogenic layer thickness and vertical extent of faulting and earthquakes in continental rifts (*Ebinger*, 2005). This load size also produces SDR packages of reasonable thicknesses (1–5 km).

In a rifted, thermally subsiding, margin setting sub–aerial SDR packages will eventually be driven below sea–level, and because extension varies laterally, they may be rotated. We have therefore superimposed the modelled SDR package that forms following dyke loading onto a rifted margin profile, calculated assuming uniform stretching, finite rifting and lateral heat flow (*Cochran*, 1983). For modelled narrow margins, this process adds a few degrees of rotation to the dips of the SDRs. The final SDR stratigraphy after thermal subsidence was then analysed using the same moving average dip calculation as was used on the global seismic reflection data (Figure 2.6).

5.3.3 Results

In all following sections, unless otherwise stated, model stratigraphic output figures are shown at a fixed aspect ratio of 3 : 1 for direct visual comparison with typical seismic reflection profiles.

5.3.3.1 Impact of Elastic Thickness

We find that the overall stratigraphy of a SDR package depends on the type of flexure model used and the T_e assumed. Using a numerical, rather than an analytical, approach to the calculation of flexure for each load allows consideration

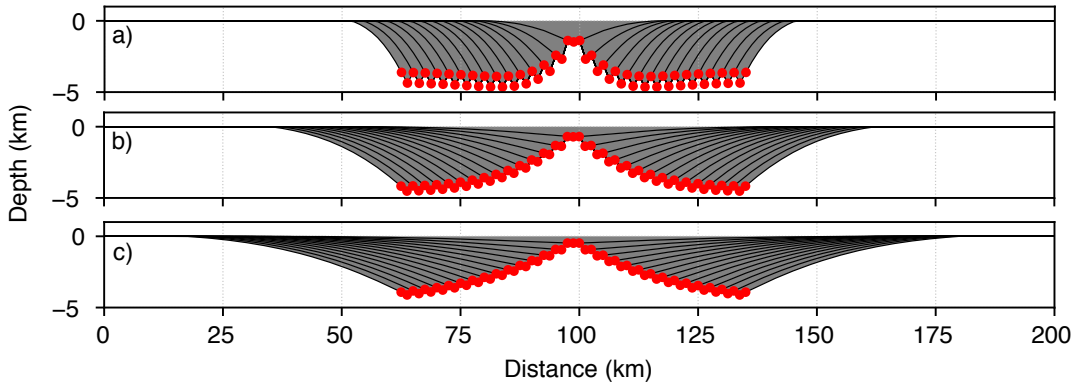


Figure 5.9: Flexure SDR model — examples of varying elastic thickness (T_e) - a) 1 km, b) 3 km, c) 5 km. All other parameters given in Table 5.2.

of the lateral variation of T_e. The model accepts three arguments – the maximum T_e, minimum T_e, and a distance over which a linear gradient between these two values should be implemented, from the position of dyke intrusion outwards.

First, we explore the implications of changing the maximum T_e value used, which represents the strength of the majority of the modelled elastic plate. We know from the flexural response function (equation 5.6) that smaller values of T_e will result in more localised compensation, converging on the Airy response for a T_e value of zero. Figure 5.9 demonstrates this clearly; as the T_e value increases, the width of the package increases significantly while the average thickness correspondingly decreases.

5.3.3.2 Broken vs. Continuous Plate

The arcuate shape of SDRs can only be produced by using a broken plate model or continuous plate model which is significantly weaker at the point of dyke injection. This weakening may, of course, be the cause of dyke localisation, before the initiation of seafloor spreading.

The default value for the distance over which the T_e decreases is the half–width of the dyke, such that the zone of weakness is the total width of the dyke. Increasing the width of this zone of weakness results in changes to the stratigraphy of the package; in general, a wider weak zone relative to the dyke width results in greater

Table 5.2: Table with standard parameters used in flexural SDR model section 5.3.3, unless otherwise stated. Note that the dyke width stated is that which produces the marker horizons; this is a representative value, larger than any single dyke for computational reasons.

Name	Symbol	Value
Young's Modulus	E	1.0×10^{11} Pa
Poisson's ratio	σ	0.25
Acceleration due to gravity	g	$9.8\,\mathrm{m\,s^{-2}}$
Zero–elevation crustal thickness	-	31.2 km
Density of crust	ρ_{crust}	$2800\,\mathrm{kg\,m^{-3}}$
Density of upper mantle	ρ_{mantle}	$3330\,\mathrm{kg\,m^{-3}}$
Density of load	ρ_{load}	$2950\,\mathrm{kg\,m^{-3}}$
Density of volcaniclastic infill	ρ_{infill}	$2700\,\mathrm{kg\,m^{-3}}$
Dyke height	h	15 km
Dyke width	w	5 km
Maximum elastic thickness	T_{emax}	3 km
Minimum elastic thickness	T_{emin}	0 km
Width of weak zone	-	2.5 km

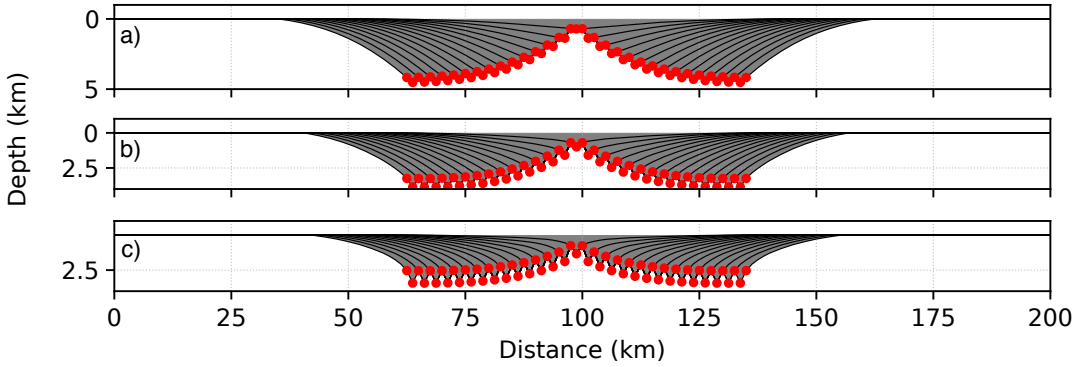


Figure 5.10: Varying width of weak zone — a) 2.5 km, b) 10 km, c) 15 km. All other parameters given in Table 5.2.

down–dip thickening and more pronounced increases in dip as depth increases. This results from the localisation of the subsidence associated with each dyke in a narrower zone compared to a plate with a more uniform T_e (Figure 5.10).

In addition to changing the width of this weak zone, changing the minimum value of T_e at the weakest point also produces changes in the stratigraphy of the modelled SDRs. With a constant distance over which the gradient is imposed, Figure 5.11 shows how changing the minimum T_e value from zero (the broken plate case) to 3 km (continuous plate) impacts the SDR dip. Whereas changing the width

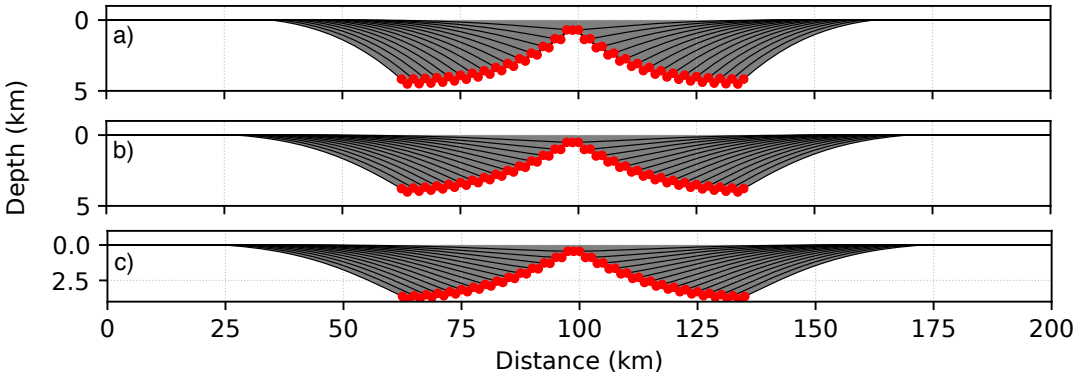


Figure 5.11: Varying T_e gradient over width of dyke, from a maximum value of 3 km to a minimum of — a) 0 km (broken), b) 1 km, c) 3 km (continuous). All other parameters given in Table 5.2.

of the weak zone does not change the width of the overall package, changing the magnitude of the weakening in this case does. Again we observe a greater magnitude of rotation, in this case for the lower minimum T_e values. A corresponding increase in thickness of the overall package is observed for the broken plate case.

In general, observations of a pronounced arcuate shape and down–dip thickening are most consistent with a broken plate model with a narrow weak zone localised to the point of intrusion. Wider zones of weakness (e.g. Figure 5.10c) produce extreme rotation at the base of the package relative to the upper part of the stratigraphy, not typically observed.

5.3.3.3 Rotation of Dykes

As subsequent SDR flow units form, our model implies that there should be some rotation of the feeder dykes, increasing with distance from the spreading axis, as shown by the red coloured markers of the dyke tops in Figure 5.7. However, thin elastic plate theory as we have assumed here only produces vertical flexures, so the deflection of an initially vertical surface cannot be accurately modelled. We have therefore shown the dykes as vertical in Figure 5.7, although we recognise that some rotation and internal deformation may occur as SDR packages evolve and the margin subsides. Nearly vertical and inclined reflections attributed to

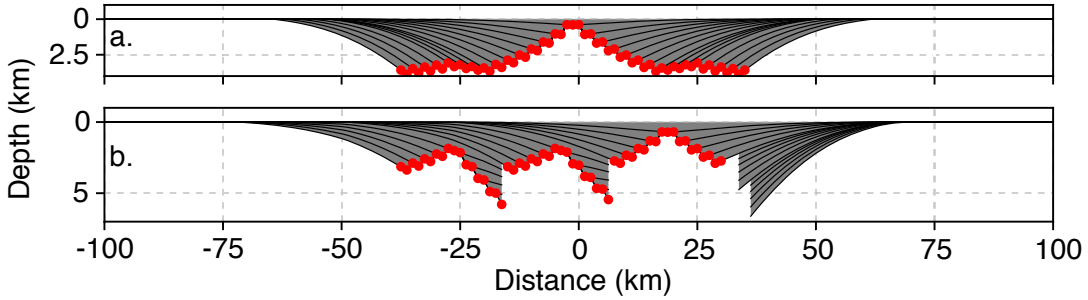


Figure 5.12: Pre–rift stratigraphy cases including a) two periods of reduced infill density and b) two jumps in position of dyke intrusion. All parameters given in 5.2, except for in case a) the infill density, which is periodically reduced to $2000\,\mathrm{kg\,m^{-3}}$. In case b), the parameters are again the same, with the addition of two jumps of $10\,\mathrm{km}$ after the intrusion of the 5^{th} and 10^{th} dyke. Note the asymmetry produced by this sequence, with a wider, thinner set of SDR packages on the left–hand side and narrower, thicker SDR packages on the right.

SDR feeder dykes have been identified in seismic reflection data from the Vøring margin (*Abdelmalak et al.*, 2015). In onshore field analogues, *Abdelmalak et al.* (2015) identify multiple generations of cross–cutting feeder dykes which have been progressively rotated, consistent with this flexural model.

5.3.3.4 'Jumps' and 'Breaks'

A period of reduced infill density, as shown in Figure 5.12a, can produce a condensed section dividing the package into two subpackages. This reduction of infilling density from $2700\,\mathrm{kg\,m^{-3}}$ to $2000\,\mathrm{kg\,m^{-3}}$ can be used to model a shift in the ratio of sediment to volcanic input into the volcaniclastic sedimentary infill. Stepped packages like this are commonly observed on volcanic rifted margins.

As demonstrated by *Buck* (2017) in his analytical modelling, the dyke loading model can be extended to shifts in the loci of intrusion, as occurs, for example, during a ridge jump in mature mid–ocean ridge settings. These jumps will produce asymmetry on the eventual conjugate margins, in both thickness and dip of a SDR package. Figure 5.12b shows that in the case of a ridge jump our numerical models produce similar geometries to those described by *Buck* (2017).

5.4 Chapter Summary

We draw the following conclusions from this modelling approach:

- Flexural models driven by surface loads produce SDR–like stratigraphy in the flanking moats, although buried loading produces flexure of a greater magnitude and is more consistent with observations.
- Offshore SDRs can be explained by a simple model of flexure in which sub–surface dyke loads create surface flexures that are subsequently infilled by volcaniclastic and basaltic lava flows.
- Flexure models generally account for the width, thickness and down–dip thickening of outer SDR packages as revealed in seismic reflection profile data over volcanic rifted margins.
- A flexural origin for outer SDRs is supported by isostatic gravity anomaly 'highs' and flanking 'lows' which reflect the dyke and infill loads and resulting flexures respectively.

Chapter 6

Gravity Anomalies and Modelling

This Chapter introduces the methods associated with calculation of gravity anomalies from existing global datasets, and modelling these anomalies using two distinct approaches. The results derived from this Chapter will be used in conjunction with results from Chapter 4: Seismic Interpretation to constrain a flexural model for SDR formation, developed in Chapter 5.

6.1 Sources of Data

Throughout this thesis, we predominantly use two global gridded datasets when considering bathymetry/topography and gravity anomalies. For elevation, GEBCO bathymetry and topography (*BODC*, 2014) grids are used, while for gravity anomaly calculations we use the Sandwell and Smith v23.1 gravity grid (*Sandwell et al.*, 2014).

The GEBCO 2014 grid is composed primarily of quality controlled shipborne depth soundings, between which data is interpolated using satellite derived gravity data. Except for the poles, land data is based on topography from the SRTM30 digital elevation model (DEM) v2.0. Source identifiers are available in the metadata for the grid, providing detailed knowledge of the origin of the data. Where more localised regional data compilations have improved on the data, these have been incorporated into the GEBCO grid using a 'remove–restore' method (*Weatherall*

et al., 2015). In some applications, particularly those in which the bathymetry is used with corresponding gravity anomaly grids, the 1–minute resolution GEBCO 2008 dataset is used.

The Sandwell and Smith v23.1 global free–air gravity anomaly grid (*Sandwell et al.*, 2014) is used as the basis for all gravity anomaly calculations. This grid is calculated from satellite radar altimetry data and has a spatial resolution of 1 minute. Where two grid are used in conjunction in a calculation in this thesis, the grid with the finer resolution is downsampled to match that of the coarser resolution grid to prevent aliasing.

6.2 Airy Isostatic Anomaly

The concept of isostatic equilibrium, the ideal theoretical balance of the Earth's lithosphere as though 'floating', can be traced back to R. G. Boscovich, an Italian astronomer, in the 18th century (*Watts*, 2001). Later, in 1855, two end–member hypotheses were proposed; the Airy–Heiskanen (henceforth Airy) model and the Pratt–Hayford model (henceforth Pratt). In both models, at some depth known as the depth of compensation, the total pressure associated with any vertical column of material will be constant and therefore equilibrium achieved. In the Airy model, isostatic compensation is achieved by columns of constant density with expanded 'roots' beneath elevated areas and 'anti–roots' beneath depressed areas, floating above a denser mantle. Conversely, in the Pratt model, compensation is achieved by laterally variable densities within columns of material which have the same maximum depth. An elevated area would therefore lie above a less dense column than an area with lower elevation.

Here, we consider the Airy isostatic anomaly as a measure of the degree of local compensation of an area. An Airy isostatic anomaly of zero indicates the region is fully compensated, in a manner equivalent to the crustal root envisaged by Airy isostasy. A positive anomaly indicates a mass excess relative to this presumed model, and therefore that the Moho is not as deep as the Airy model would

predict, while a negative anomaly indicates a mass deficit, and therefore a deeper Moho than predicted. The Airy isostatic anomaly is particularly useful when studying rifted margins, as the thinning of the crust from continental to oceanic produces a significant component of the gravity anomaly, which will be largely removed by the Airy model. Both Airy and Pratt represent completely local isostatic compensation of loads on the crust; as discussed in Chapter 5, where the crust has finite strength, we expect to observe regional compensation by flexure. The isostatic anomaly can therefore provide insight as to where flexure may be acting.

6.2.1 Method of Calculation

Computationally, the gravity anomaly calculations performed here were carried out in the wavenumber domain. We use the `fft2` MATLAB function in order to calculate the global Airy isostatic anomaly from satellite–derived gravity anomaly and topography data; bold font indicates the variable in the Fourier transform domain.

Equation (6.1) demonstrates the exponential decay of the gravity anomaly $\Delta\mathbf{g}_Q(k)$ with height P above the point Q, where $k = \frac{2\pi}{\lambda}$ and λ is the wavelength. The anomaly decays more rapidly for high wavenumbers than for low wavenumbers.

$$\Delta\mathbf{g}_P(k) = e^{-kp}\Delta\mathbf{g}_Q(k) \tag{6.1}$$

Using Green's equivalent layer theorem (*Grant and West*, 1965), $\Delta\mathbf{g}_Q(k)$ is replaced by a surface mass distribution $\mathbf{M}_Q(k)$ (equation (6.2)), which can be decomposed into an undulating interface with an associated density contrast $\Delta\rho$, equation (6.3).

$$\mathbf{M}_Q(k) = \frac{\Delta \mathbf{g}_Q(k)}{2\pi G} \tag{6.2}$$

$$\Delta \mathbf{g}_Q(k) = 2\pi G \Delta\rho \mathbf{H}(k) \tag{6.3}$$

$$\Delta \mathbf{g}_P(k) = e^{-kp} 2\pi G \Delta\rho \mathbf{H}(k) \tag{6.4}$$

Finally, when measured at height P, the exponential decay from equation 6.1 is incorporated, resulting in equation 6.4 for the first order term. This equation is used in the MATLAB function `grv3dFFT` (Appendix C) to calculate the component of the gravity anomaly produced by the topography of the Earth's crust and its isostatic compensation. This topography is balanced against a reference column (e.g. zero–elevation crustal thickness of 31.2 km and density 2800 kg m^{-3}) following *Simpson et al.* (1986) (see Figure 6.1) in order to locate the Moho which would achieve Airy–style compensation. The gravity anomaly due to this surface is also calculated using equation 6.4, and both the basement and Moho components are subtracted from the observed free–air anomaly.

We have derived the observed Airy isostatic gravity anomaly from the 1 × 1 minute free air gravity anomaly grid (Version 23.1) of *Sandwell et al.* (2014) and the 1 × 1 minute GEBCO 2008 bathymetry grid of *BODC* (2014), assuming a zero–elevation crustal thickness of 31.2 km and an average crustal density of 2800 kg m^{-3}. Spherical harmonics from degree and order 2 to 20 (EGM 2008, *Pavlis et al.* (2012), shown in Figure 6.2) have been removed from the anomaly in order to remove long wavelength effects associated, for example, with dense subducted slabs.

6.2.2 Global Map

The result of the calculation described above in section 6.2.1 globally is shown in Figure 6.3. While many areas of the globe have an isostatic anomaly close to zero, there are significant deviations from equilibrium in some geological settings.

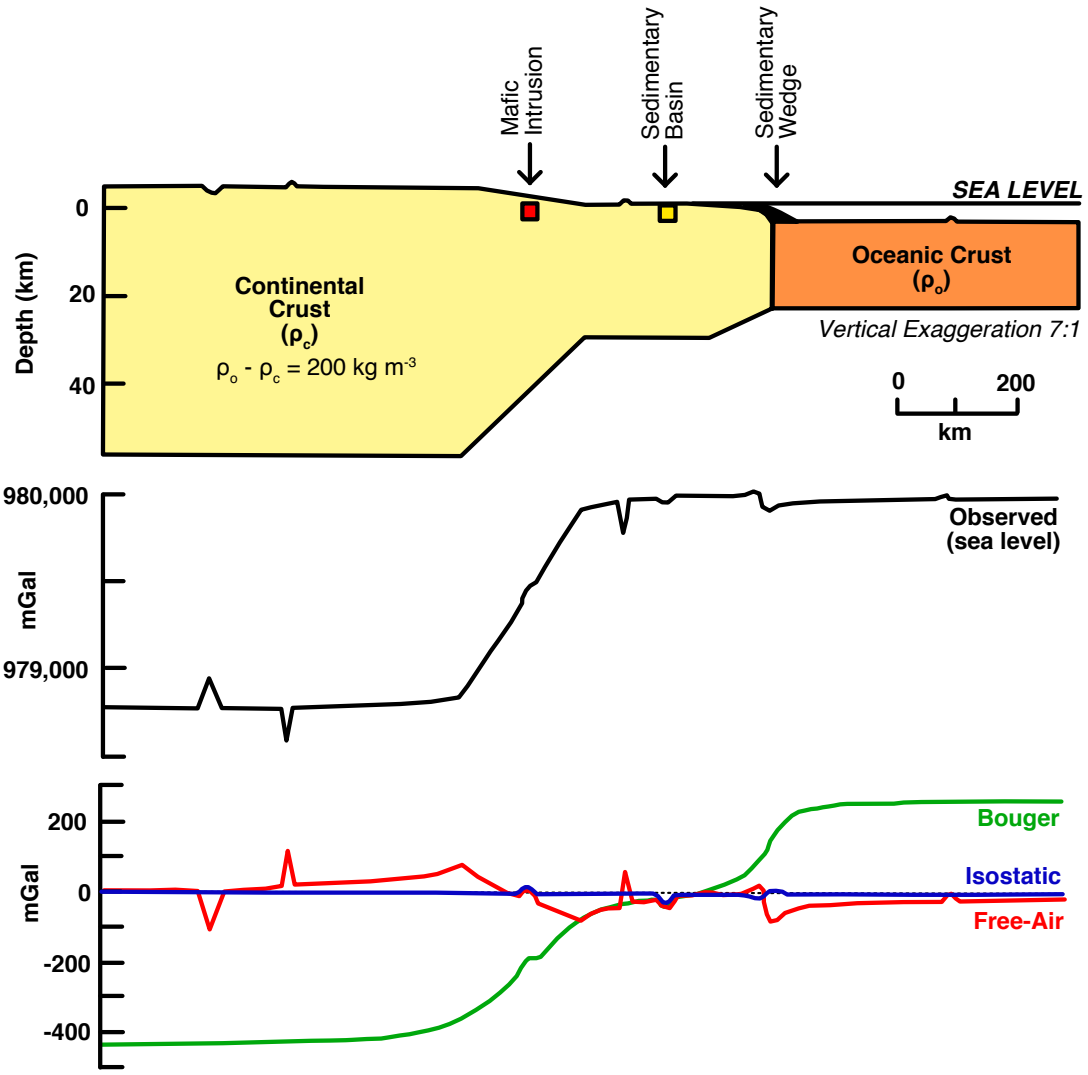


Figure 6.1: Comparison of the observed, free–air, Bouger and Airy isostatic anomaly over a 2D rifted margin model, modified from *Simpson and Jachens* (1989). Simplified geometry demonstrates the impact of small scale (e.g. basins and intrusions) and large scale (e.g. crustal thinning) processes. The model assumes local Airy isostatic compensation, with oceanic crust $200\,\mathrm{kg\,m^{-3}}$ denser than continental. As the isostatic anomaly calculation assumes crust has a constant density, this results in a small low–high couple at the COB. The high–low couple in the Free Air anomaly at the same location results is dominated by the change in water depth.

Subduction zones have large negative isostatic anomalies, while ocean islands and mid–ocean ridges have large positive anomalies. Of interest in this thesis are the origin of the strike–parallel highs and lows along many of the volcanic rifted margins — notably South West Africa, South America and Greenland. In addition to providing context, the grid shown in this Figure will be later used to constrain

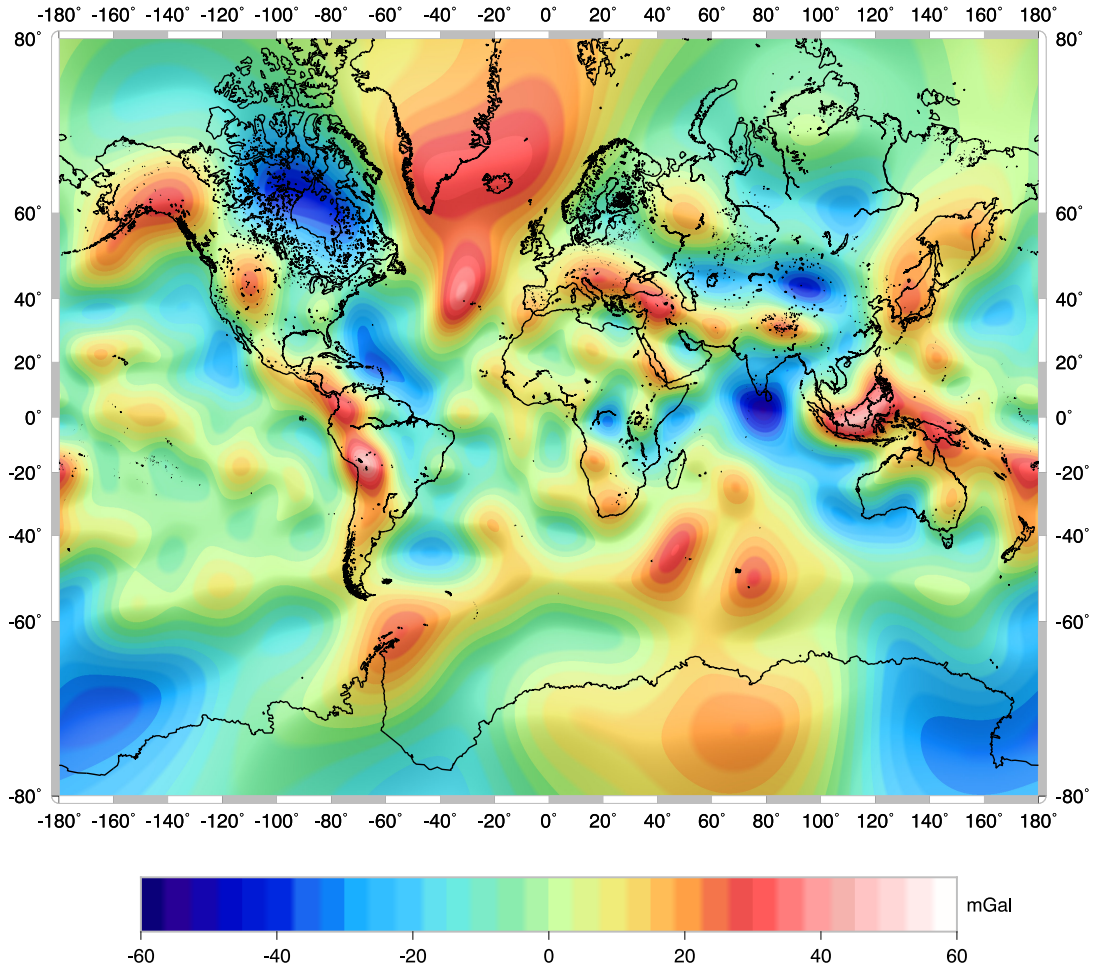


Figure 6.2: Global gravity spherical harmonics, from degree and order 2 to 20 (EGM 2008, *Pavlis et al.* (2012).

flexural modelling (Chapter 7).

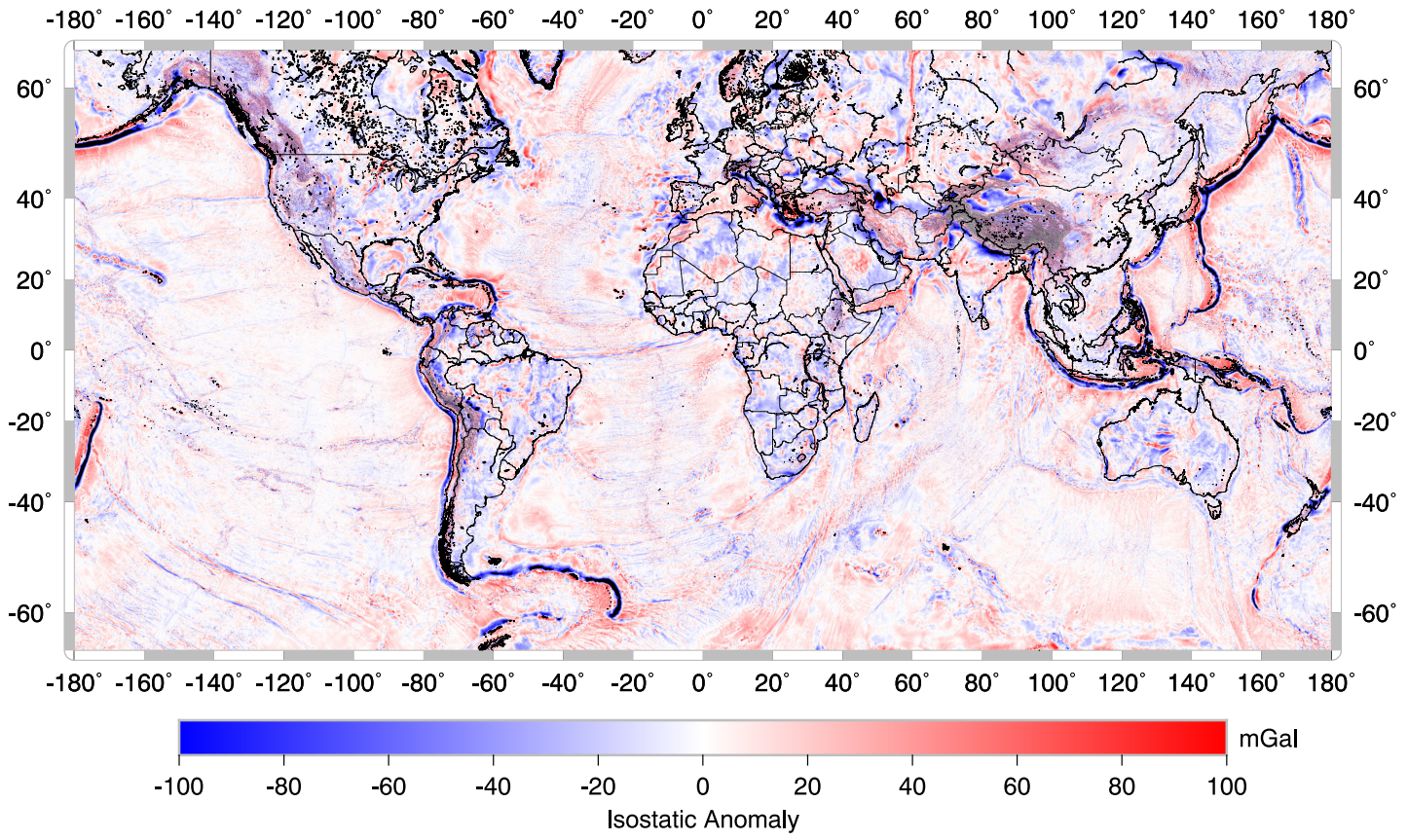


Figure 6.3: Global Airy Isostatic Anomaly map, calculated using method described in section 6.2.1. Note the extreme lows (e.g. over subduction zone trenches) in black, exceeding the bottom of the scale bar. Bathymetric features such as mid–ocean ridges and fracture zones are clearly visible.

6.3 Introduction to Process–Oriented Gravity Modelling

In order to calculate the gravity effect of SDRs, we take a Process–Oriented Gravity Modelling (POGM) approach, similar to that used by *Watts* (1988) to calculate the contributions of rifting, sedimentation and erosion to the free–air gravity 'edge effect' anomaly at the non–volcanic Baltimore Canyon Trough, East Coast USA rifted margin. This approach has also been used to calculate the contribution of magmatic underplating to the 'edge effect' anomaly at the British Isles rifted margin (*Watts and Fairhead*, 1997). In principle, POGM proposes different configurations of the study area, isolating the distinct contributions of different geological processes (e.g. rifting and sedimentation) to the resulting gravity anomaly (Figure 6.4).

Here, we extend the POGM methodology to include the gravity effect of sub–surface intrusive dyke loads and their associated extrusive infill loads, isolating in the process the so–called dyke and SDR gravity anomalies. These processes may have occurred synchronously, or may have occurred at different stages during the development of the area; in either case, the theoretical isolated contributions may be identified using POGM. These anomalies can therefore be summed and compared to observed gravity anomalies.

The rifting anomaly, or edge–effect, is calculated in POGM by evaluating the gravity anomaly on two interfaces — the basement and Moho, after backstripping (*Watts*, 1981) to locate the position of the basement in the absence of sediment. By removing the calculated rifting anomaly from the observed gravity anomaly, it is possible to isolate interesting features on a margin where other processes are significantly affecting the observation. This can highlight, for example, intrusive volcanic bodies or localised extensional basins, which may warrant further study.

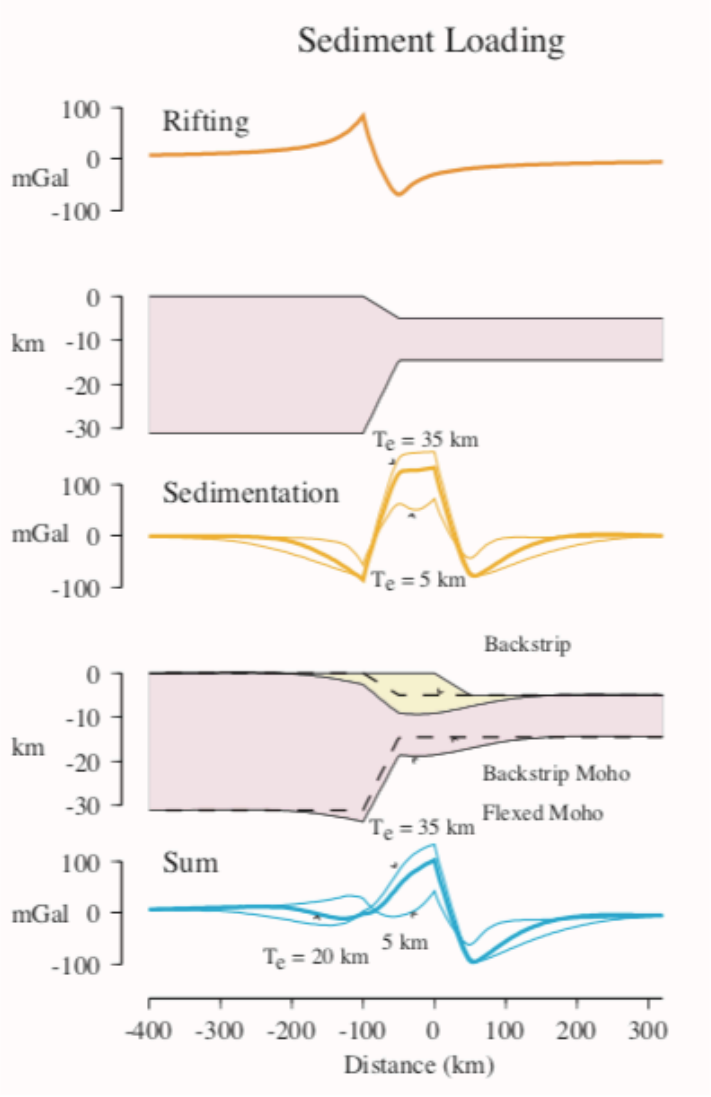


Figure 6.4: Schematic of POGM sediment anomaly calculation, from *Watts and Fairhead* (1999). The rifting anomaly, composed purely of the thinning of the crust, is distinguished from the effect of the sediment wedge (and its associated flexure). The width and magnitude of the sediment anomaly is dependent upon the elastic thickness (T_e).

6.3.1 Methods of Calculating Gravity Anomalies

There are two distinct approaches to calculating the gravity anomaly associated with a density contrast in the Earth's crust. As described in section 6.2.1, a FFT method is often used. This is particularly useful when dealing with large–scale continuous surfaces, like the crustal boundaries at a rifted margin. However, the FFT behaves poorly when dealing with sharp discontinuities (i.e. interfaces that vary rapidly about the mean depth), so this method is often unsuitable for

smaller scale features and steep boundaries. The FFT method also requires a constant density contrast across the entire surface in question, requiring additional corrections when this value is not constant in the model.

An alternative approach uses a line integral method (*Talwani et al.*, 1959; *Bott*, 1960). Although this can be more computationally expensive than the FFT, there are several advantages. Firstly, the density contrast can be defined on each segment of the polygonal boundary, rather than a single constant value. Once the polygon is defined, the same boundaries may be used to calculate the magnetic anomaly. The polygon may be open or closed, so the same method can be used for both discrete bodies and continuous surfaces.

The FFT method has been used to calculate the isostatic anomaly, but the line integral method will be used to calculate the magmatic contributions to the process–oriented gravity anomaly.

6.4 Comparison to Object–Oriented Gravity Modelling

In contrast, object–oriented gravity modelling, or OOGM, considers only the final, present–day configuration of the subject area. By fitting this to the observed gravity anomaly, we can constrain the geometry and density of the contributing bodies (given the non–uniqueness of gravity modelling). However, OOGM does not provide any additional information about the formation of these bodies.

Contributions to the POGM calculated anomaly may appear counter–intuitive: for example, the sediment anomaly is a positive anomaly, not a negative one. In a process–oriented sense, sediments fill an existing space, displacing air or water, and therefore the major contribution to their anomaly is a positive one. This is modified by the compensation at both the basement and Moho interfaces, which will be small flanking negatives, as less dense material displaces more dense material in both cases. In an object–oriented sense, a sediment anomaly may be

expected to be negative as sediments are typically less dense than surrounding crust.

6.4.1 Evaluating POGM and OOGM

It can be shown that the results of POGM and OOGM are equivalent for a rifted margin with sedimentary wedge case (*Watts*, 2018). A satisfactory fit in either case is therefore not confirmation of an exact geometry, or set of contributing processes, but it *is* a plausible one. Many of the free parameters in gravity modelling can be well constrained by a combination of seismic reflection, refraction and drilling observations. Assuming the approximate lithological boundaries and densities are known from one of these datasets, POGM can be used to provide additional information about the crust — namely elastic thickness (T_e) during the process in question. Similarly, it retains all the benefits of OOGM, confirming that the densities and boundaries used are a satisfactory fit to the observed gravity anomaly.

POGM and OOGM are rarely applied in tandem. In addition to the difference in approach to modelling, the two methods typically use a different approach to calculate the gravity anomaly for each body. In OOGM, a line integral method is usually used to calculate the contribution to the overall gravity anomaly of each body, modelled as an open or closed polygon. In contrast, POGM often uses a Fast Fourier Transform (FFT) method to calculate the gravity anomaly of continuous surfaces which separate each geological boundary. This approach is very useful for large–scale features such as rifted margins, with few boundaries.

6.5 Ensemble Averaging of Gravity and Magnetics along Margins

The isostatic anomaly has a significant along–strike trend along rifted margins. It is therefore insightful to extract this component of the anomaly, both to constrain

models using the average value and to consider the residual. A similar approach was used by *Bassett and Watts* (2015) to calculate and remove the ensemble average of the free–air anomaly (FAA) at subduction zones. Ensemble averaging is a spectral method that is equivalent to arithmetic averaging if the number of profiles is large (*Bassett and Watts*, 2015). In this study, once calculated, the ensemble average value along a section of a margin can be used in conjunction with the flexural model to constrain the T_e of a margin. As SDRs have no bathymetric expression and lie within the upper crust, the isostatic anomaly is particularly suitable in understanding these features.

6.5.1 Method of Calculation

The ensemble average and corresponding residual can be calculated from any gridded dataset — in this case, we use the isostatic anomaly and the EMAGv2 magnetic anomaly. First, an along–strike track must be provided, from which orthogonal perpendicular profiles will be projected using the `project_profiles` function (Appendix C). This function projects a user–defined number of profiles using great circle paths.

The gridded values are extracted along each of these profiles, from which an ensemble average is calculated in the wavenumber domain using function `calc_averages`. This average is used to constrain the flexural model described in Chapter 5. In order to find the residual, the ensemble average values are projected back onto the orthogonal profiles using the function `project_ensembles` and exported for gridding in GMT.

Using the values calculated in MATLAB, the GMT programs `blockmean` and `surface` are used to produce grids of the ensemble data (see Appendix D). These grids are subtracted from the corresponding input grids using GMT `grdmath` to find the residual; both the ensemble and residual grids are masked to cover only the spatial extent of the profiles using `grdmask`. Where the ensemble profiles intersect

LIPs/SDRs as identified in the database compiled by *Coffin et al.* (2006), this is shown on the appropriate figure.

6.5.2 Results

The ensemble average method is applied to both the Greenland and Namibian margins, discussed elsewhere in this thesis.

Figure 6.5 shows the result from the Greenland margin, applied to the isostatic anomaly, bathymetry and magnetic anomaly grids. The corresponding gridded values are shown in map view in Figure 6.6. The dashed line in Figure 6.6a. is derived from the EMAG2v3 grid, chosen over using a bathymetric contour as a better proxy for margin structure.

There is a significant isostatic anomaly high which is persistent along–strike, shown clearly in the ensemble grid. While a large part of this is removed by the ensemble, a component of this remains in the residual and closely mirrors the coastline. The magnetic anomaly is more irregular, with small ensemble values. A high is present on the coastline, which may be associated with the coastal dyke swarm.

The location of SDRs (*Coffin et al.*, 2006) does not correlate well with either the isostatic or magnetic anomaly; both are variable and approximately zero over this region. Results from Chapter 3 shows multiple discrete SDR packages on the southern part of the Greenland margin; if this continued along–strike, the ensemble process could mask smaller scale variation.

Figure 6.7 shows the result from the Namibian margin, again applied to the isostatic anomaly, bathymetry and magnetic anomaly grids. The corresponding gridded values are shown in map view in Figure 6.8. The dashed line in Figure 6.8a. is again derived from the EMAG2 grid.

In comparison to the Greenland margin, there are much clearer similarities between the isostatic anomaly and magnetic anomaly on the Namibian margin. A broad high in the magnetic anomaly, corresponding with the 'LMA' (*Moulin et al.*, 2010)

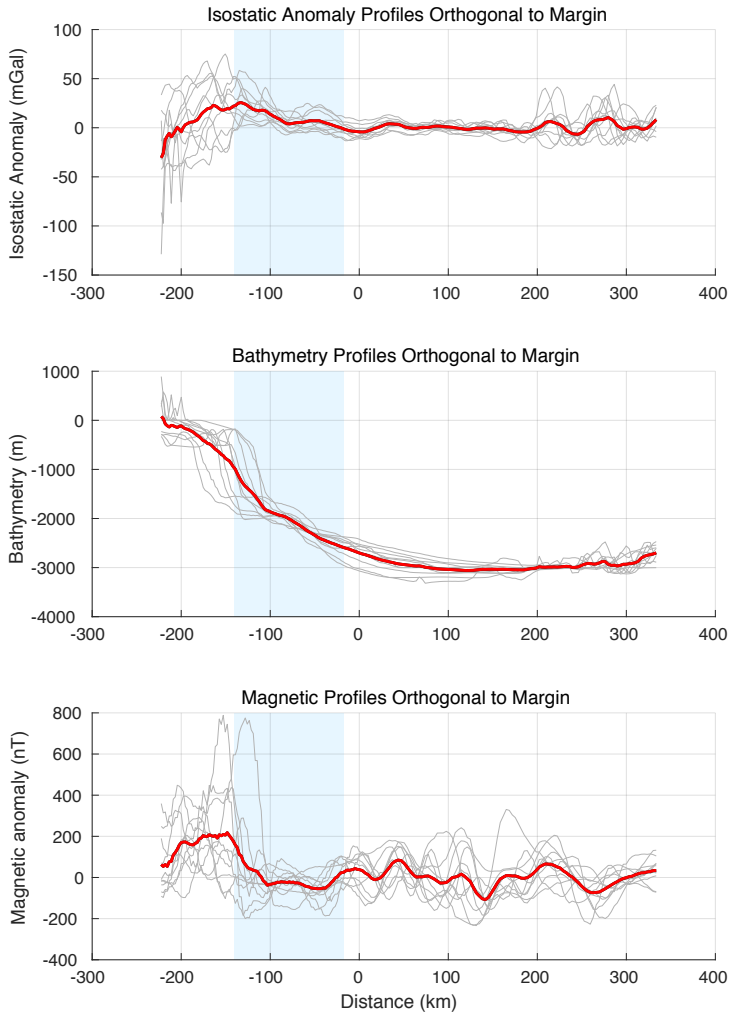


Figure 6.5: Orthogonal profiles from the Greenland margin: Isostatic gravity anomaly, bathymetry (GEBCRO 2008) and magnetic anomaly with calculated ensemble averages.The blue shaded region indicates intersection with location of SDRs from *Coffin et al.* (2006). In each panel, thin grey lines represent individual profile track data (with location of profiles shown in Figure 6.6 and the thick red line is the ensemble average of these profiles. Zero marks an along–strike defined path found using the EMAG2 global grid as a consistent reference point. Negative distance is landward, while positive distance is seaward (with east/west orientation preserved).

dominates the signal, which, once removed using the ensemble average, reveals a residual which resembles the isostatic anomaly residual. Notably, the study region in this case is smaller than in Greenland, increasing the likelihood that ensemble averaging will correctly extract a correlatable along–strike signal.

The isostatic anomaly ensemble average shows a strong coastal high and two offshore highs. The most landward offshore high corresponds with the landward

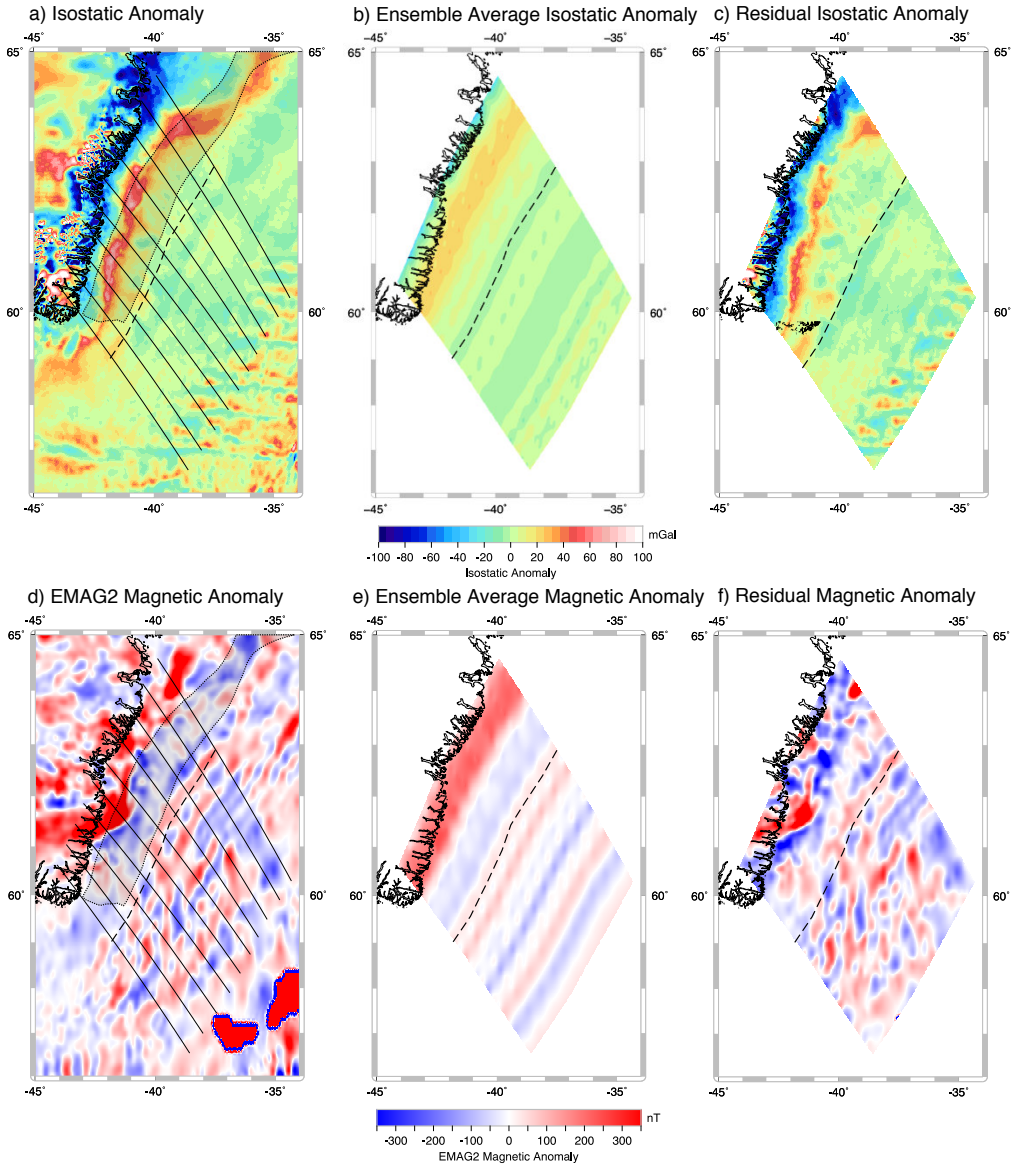


Figure 6.6: Map view of gridded ensemble average and residual isostatic anomaly and magnetic anomaly data, Greenland margin. Thin black lines show location of projected orthogonal profiles; bold dashed line shows the along–strike marker from which the profiles were projected. Coastlines and political boundaries shown for context. The scale used in each case is consistent across the input, ensemble and residual panels for direct visual comparison. In panels a) and d), the light grey shaded region outlined by the black dotted lines indicates the extent of known SDRs from *Coffin et al.* (2006).

gradient of the LMA, with the seaward high corresponding with the seaward gradient of the LMA. This seaward gradient also corresponds well with the location of the SDR package, according to *Coffin et al.* (2006). These features will be explored further in Chapter 7.

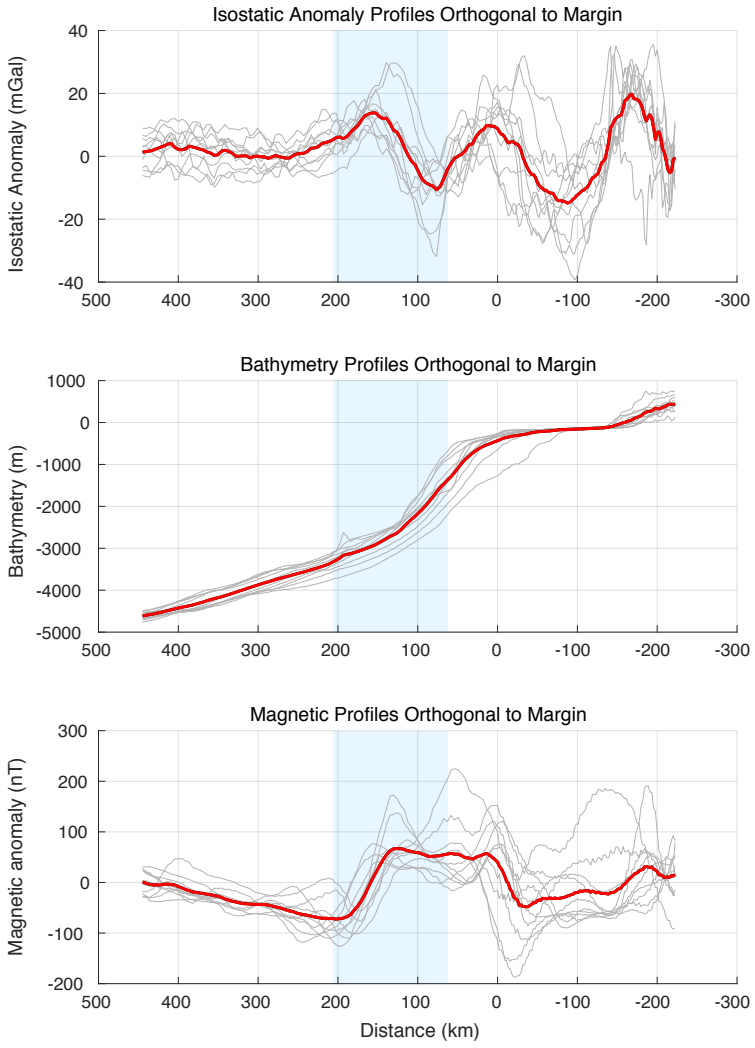


Figure 6.7: Orthogonal profiles from the Namibian margin: Isostatic gravity anomaly, bathymetry (GEBCO 2008) and magnetic anomaly with calculated ensemble averages. The blue shaded region indicates intersection with location of SDRs from *Coffin et al.* (2006). In each panel, thin grey lines represent individual profile track data (with location of profiles shown in Figure 6.8 and the thick red line is the ensemble average of these profiles. Zero marks an along–strike defined path found using the EMAG2 global grid as a consistent reference point. Negative distance is landward, while positive distance is seaward (with east/west orientation preserved).

6.6 Applying POGM to Dykes and SDRs

A critical test of the validity of all process–based formation models, and particularly flexural models, is the gravity anomaly, since it is sensitive to both the driving loads and the resulting flexures. We have shown that a flexure model of dyke and infill loading can predict geometries that match those of SDRs observed on seismic

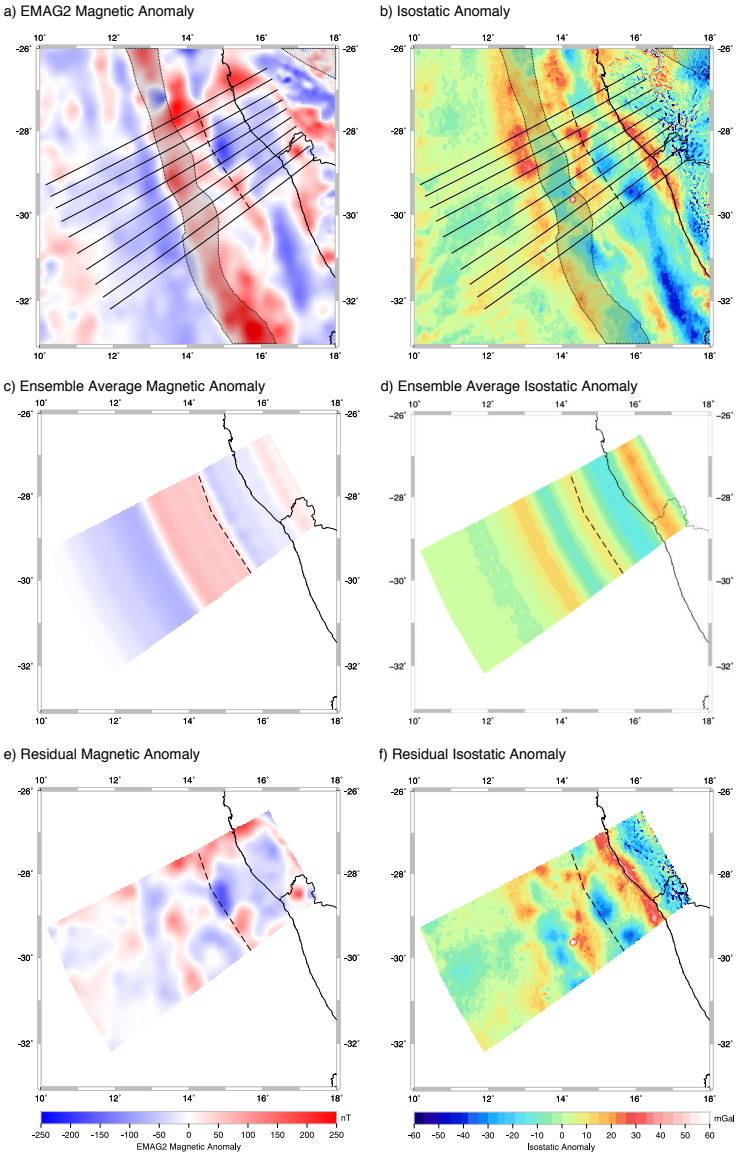


Figure 6.8: Map view of gridded ensemble average and residual isostatic anomaly and magnetic anomaly data, Namibian margin. Thin black lines show location of projected orthogonal profiles; bold dashed line shows the along–strike marker from which the profiles were projected. Coastlines and political boundaries shown for context. The scale used in each case is consistent across the input, ensemble and residual panels for direct visual comparison. In panels a) and b), the light grey shaded region outlined by the black dotted lines indicates the extent of known SDRs from *Coffin et al.* (2006).

reflection data, both in terms of overall width and thickness and stratigraphic dip. However, the internal velocity structure and consequently the thickness of SDRs is generally not well constrained from seismic data, so it is useful to use the gravity anomaly as an additional constraint on models for SDR formation.

Table 6.1: Summary of the main parameters used in the flexural and gravity modelling in this Chapter. The values used in this table apply to all figures except where explicitly stated,

Name	Symbol	Value
Young's Modulus	E	1.0×10^{11} Pa
Poisson's ratio	σ	0.25
Acceleration due to gravity	g	$9.8\,\mathrm{m\,s^{-2}}$
Zero–elevation crustal thickness	-	31.2 km
Density of crust	ρ_{crust}	$2800\,\mathrm{kg\,m^{-3}}$
Density of upper mantle	ρ_{mantle}	$3330\,\mathrm{kg\,m^{-3}}$
Density of load	ρ_{load}	$2950\,\mathrm{kg\,m^{-3}}$
Density of volcaniclastic infill	ρ_{infill}	$2700\,\mathrm{kg\,m^{-3}}$
Dyke height	h	15 km
Dyke width	w	5 km
Elastic thickness	T_e	3 km

6.6.1 Calculating Gravity in the SDR Model

We assume, following *Mutter et al.* (1982), that SDRs originate at sea–level and then subside following rifting, cooling and thermal contraction. There will therefore be a rifting gravity anomaly in addition to a dyke and SDR anomaly. The rifting anomaly depends on the mode of extension and is given by the combined gravity effect of the water–filled tectonic subsidence and the isostatically balanced Moho uplift. The dyke anomaly subdivides into three main contributions: a gravity high associated with the dyke load, and flanking gravity lows associated with the load–driven flexure at both the top and base of the crust. The SDR anomaly subdivides similarly: a gravity high associated with the lava and volcaniclastic material infilling the pre–existing, air–filled subsidence, and flanking lows associated with the additional flexure at the top and base of the crust. All anomalies can be computed from the implied mass distributions using a line–integral method (e.g. *Talwani et al.*, 1959; *Bott*, 1960) and the attenuating effects of the increase in water depth that follows rifting can be included in the dyke and SDR anomaly calculation.

We can use the same approach to investigate how different configurations of SDR packages produced by the flexural model impact the SDR and dyke anomaly.

First, we consider a model with periods of reduced infill, producing three sub–packages of SDRs (Figure 6.9). Despite this significant difference in the shape of the package, there is comparatively little impact on the sum anomaly. The wider, flatter base of the SDR package reduces the amplitude of the dyke and sum anomalies. However, the same amplitude change is observed in Figure 6.10 without the periods of reduced infill.

The sum anomaly in all previous figures has been shown after rifting, i.e. with the package subsided to observed depths of ∼6 km. Figure 6.10c. shows the pre– and post–rift sum anomaly for the same SDR configuration, a long 'train' of SDRs. The exponential decay with depth as the margin subsides decreases the amplitude of the sum anomaly, leaving only non–zero values at the edges of the dyke body.

6.6.2 Fitting Modelled Gravity to the Airy Isostatic Anomaly

In order to assess the goodness of fit between modelled gravity as described in this Chapter and with observations, we seek the minimum root–mean squared error (RMSE). Using this method, the 2D modelled sum anomaly is compared to the ensemble average of the Airy Isostatic Anomaly over the SDR package. The RMS is calculated for a range of shifts in the y position of the modelled values, as the absolute value is not important. The shift which obtains the minimum RMS value is used in each case.

This technique is applied to the Namibian margin, described in more detail in Chapter 7.

6.7 Gravity for Alternative SDR Formation Models

If feeder dykes were not present immediately beneath the SDR body, the dyke component of the modelled anomaly would not be present. A benefit of the

process–oriented approach is this isolation; we can consider the SDR anomaly alone if considering a formation model in which the dykes are not present.

For example, if the SDR package was instead bound by a fault, and both the footwall and the hanging wall of the fault were of continental crust density as modelled here, the SDR anomaly calculated would apply (see Figure 6.11c for comparison). This is dominated by the high of the infill load of volcaniclastic fill, which is denser than the surrounding crust. Although the magnitude and high and the wavelength will vary with the density contrast, the maxima of the SDR anomaly will always be landward of the sum anomaly. We would therefore expect a fault–bound SDR package to have a gravity anomaly maxima over the thickest part of the package, while a dyke–driven SDR package would have the maxima close to the seaward termination, where the dyke load dominates.

Considering the inner and outer SDR packages discussed in Chapter 4 in the Orange Basin, Namibia, Figure 4.19 shows the Airy Isostatic Anomaly with corresponding seismic reflection data. The inner SDR corresponds with a broad high, with a maxima over the centre of the package, suggesting it may be fault bound. The outer package will be investigated further in Chapter 7.

6.8 Chapter Summary

- A global grid of the Airy isostatic anomaly is calculated, highlighting key geologic features which may not be easily studied with the free–air or Bouger anomalies alone.
- An ensemble averaging technique is used to extract along–strike signals from margins and study both the resulting average and residual maps for key VRMs.
- Applying this averaging technique to Greenland and Namibia highlights trends in the isostatic and magnetic anomalies that may be attributed to magmatic features.

- A process–oriented gravity modelling technique is developed for the proposed flexural model, differentiating between the contributions of the dyke intrusion and volcaniclastic infill processes. This is contrasted with the more traditional object–oriented gravity modelling approach (OOGM).

- Given a configuration of surfaces and corresponding density contrasts, two methods of calculation of the gravity anomaly are available: FFT, commonly used in POGM, and line integral, commonly used in OOGM.

- An RMS–based technique is used to quantify the goodness of fit between the modelled sum anomaly and observed, in order to estimate the best–fit parameters.

- The POGM approach suggests that the shape of the gravity anomaly for a fault–bound SDR package will differ from that of a dyke–driven SDR package, with a characteristic shift in the position of the maxima.

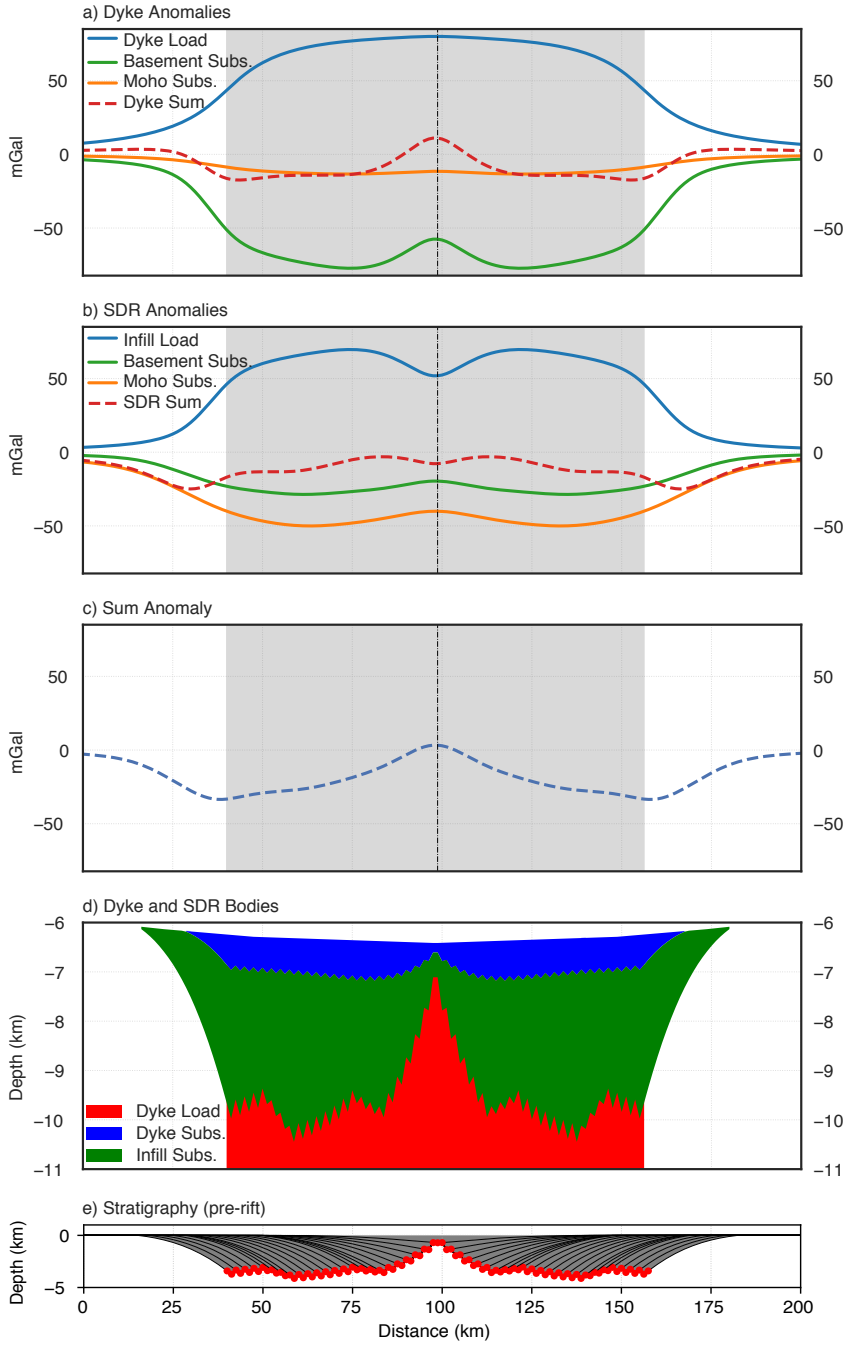


Figure 6.9: Process–oriented gravity anomalies for an SDR package, demonstrating impact of reduced infill periods on the sum anomaly (stratigraphy also shown). Using values from Table 6.1. a) Dyke gravity anomaly. b) SDR gravity anomaly. c) Sum gravity anomaly (a + b) with pre–rift shown with a dashed line and post–rift shown with the solid line. d) Dyke and SDR bodies. Red shade = dyke load. Blue shade = dyke driven subsidence and subsequent infill load. Green shade = infill driven flexure. The final SDR package will incorporate both the blue and brown regions. Vertical dashed line in figures (a–c) delineates the location of future continental rifting and break–up; grey shaded rectangle indicates the lateral extent of dyking. e) Stratigraphy with marker horizons in black, clearly demonstrating the condensed sections of reduced infill density.

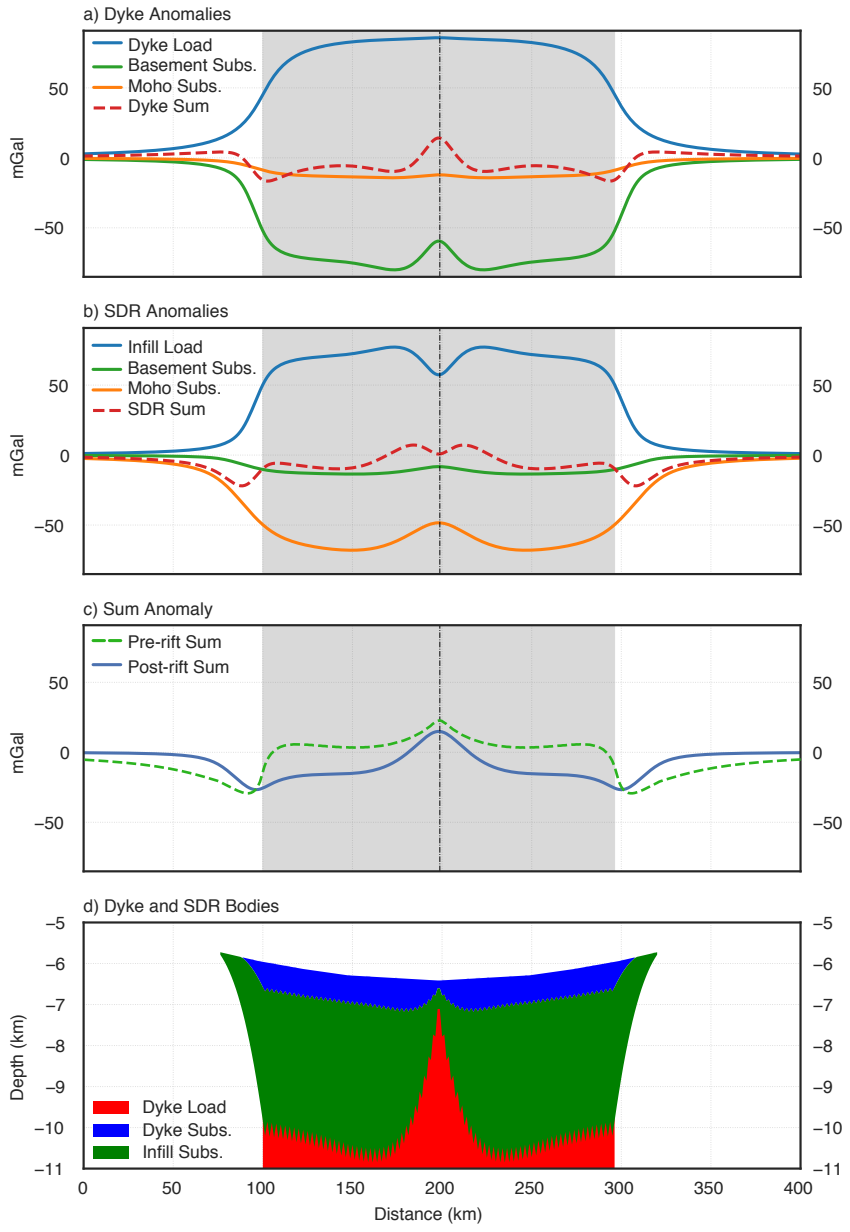


Figure 6.10: Process–oriented gravity anomalies for an SDR package, demonstrating impact of rifting and subsidence on the sum anomaly for a wide SDR package. Using values from Table 6.1. a) Dyke gravity anomaly. b) SDR gravity anomaly. c) Sum gravity anomaly (a + b) with pre–rift shown with a dashed line and post–rift shown with the solid line. d) Dyke and SDR bodies for final post–rift configuration. Red shade = dyke load. Blue shade = dyke driven subsidence and subsequent infill load. Green shade = infill driven flexure. The final SDR package will incorporate both the blue and brown regions. Vertical dashed line in figures (a–c) delineates the location of future continental rifting and break–up; grey shaded rectangle indicates the lateral extent of dyking.

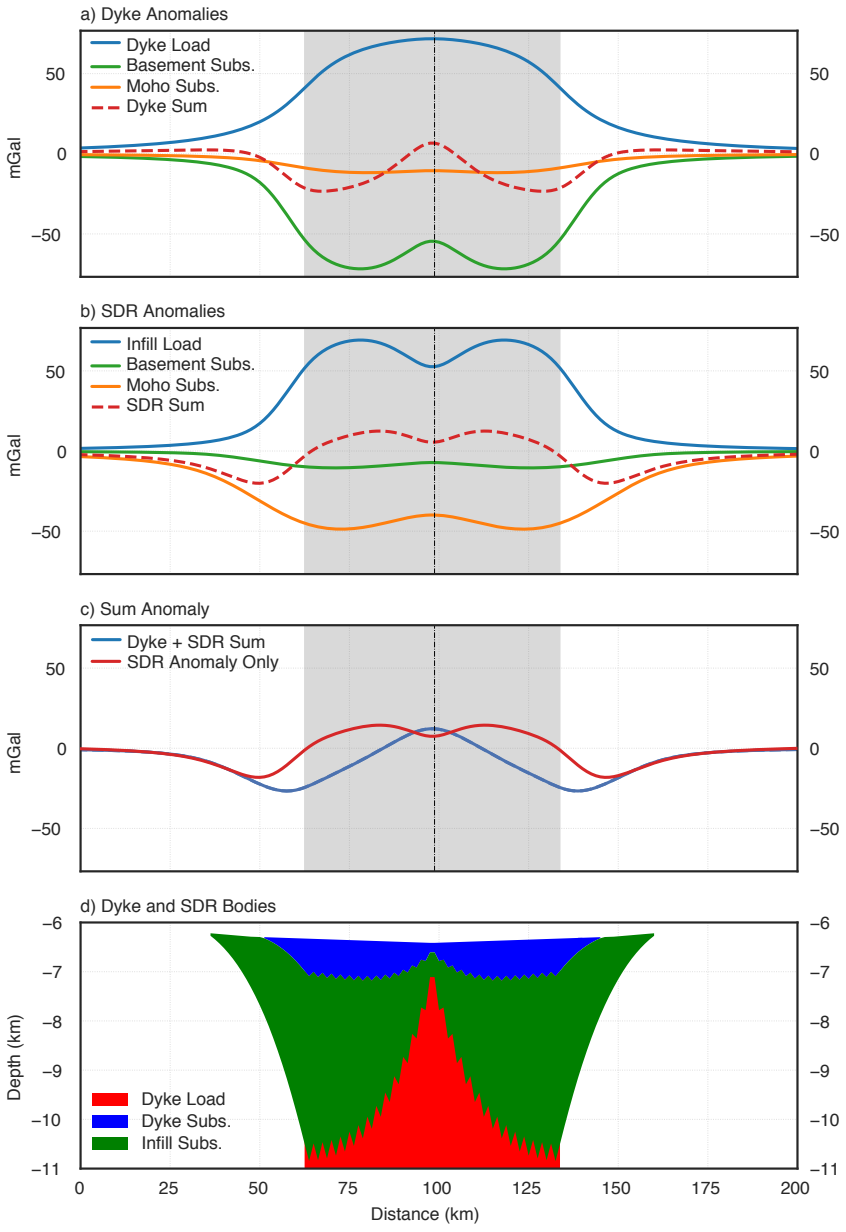


Figure 6.11: Process–oriented gravity anomalies for an SDR package, comparing shape of sum anomaly with the SDR component of the anomaly alone. Using values from Table 6.1. a) Dyke gravity anomaly. b) SDR gravity anomaly. c) Sum gravity anomaly (blue, a + b) contrasted with the SDR anomaly alone (red); note shift in position of gradient relative to SDR package. d) Dyke and SDR bodies. Red shade = dyke load. Blue shade = dyke driven subsidence and subsequent infill load. Green shade = infill driven flexure. The final SDR package will incorporate both the blue and brown regions. Vertical dashed line in figures (a–c) delineates the location of future continental rifting and break–up; grey shaded rectangle indicates the lateral extent of dyking.

Chapter 7

Namibia Case Study

We have applied our model of SDR formation to the Namibia volcanic rifted margin (Figure 7.1). This margin was selected as a type example of a VRM, south of the Rio Grande Rise and Walvis Ridge. Spectrum Geo provided us access to a closely spaced (∼5–10 km) seismic reflection profile data set on which SDRs can be easily identified and their dips measured; this dataset is interpreted and discussed in Chapter 4. Primarily, the polygon used to define the SDR body in this section encloses completely all picked most seaward and most landward SDRs.

7.1 Application of Flexural Model

The model parameters that best–fit the geometry of the offshore Namibia SDRs are T_e in the range (1.25–3 km), a dyke height of 15 km and a total dyke intrusion width of 75 km. The maximum thickness of the modelled package of 5 km is also consistent with observations, assuming the internal velocity of the package is in the range 3–5 km s^{-1}, consistent with the results of Chapter 3.

The SDR package produced by the best–fit model and the dips associated with this are shown in Figure 7.2, with parameters given in Table 7.1. We find dip values here are very close to the global average, typically in the range 5–15°. The observed dips were derived from the Spectrum Geo seismic reflection data set, interpreted in Chapter 4. The calculated dips were derived from the best–fit model output

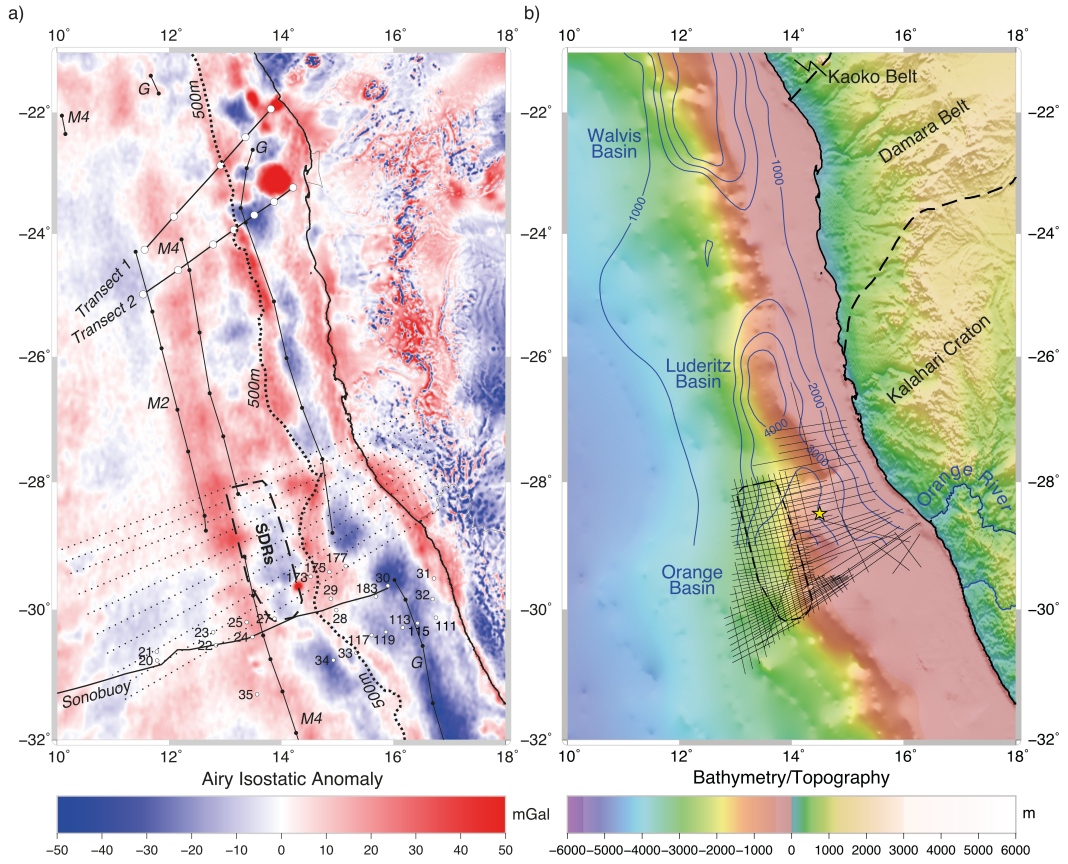


Figure 7.1: Airy isostatic gravity anomaly and bathymetry (GEBCO 2008, *BODC* (2014)) of the volcanic rifted margin offshore Namibia. The extent of the SDR package identified from the seismic reflection data is shown by the polygon with dashed black line, derived from Chapter 4. a) Airy isostatic gravity anomaly with the degree and order 20 gravity field removed. Heavy dotted black line shows the $500\,\mathrm{m}$ isobath. Thin solid lines with filled black circles show magnetic lineations M2 ($\sim$124 Ma) and M4 ($\sim$127 Ma) and the 'G' anomaly or 'LMA' (*Rabinowitz and LaBrecque*, 1979; *Collier et al.*, 2017; *Moulin et al.*, 2010). Thick solid line with associated white circles and numbers show the Sonobuoy profile of *Rabinowitz* (1976). Transects 1 and 2 show the seismic refraction profiles of *Bauer et al.* (2000). Light dotted black lines indicate the location of the profiles used to calculate the ensemble average shown in Figure 7.2c. b) Bathymetry. Thin solid black lines show the location of available Spectrum Geo seismic reflection survey. Contours at $1000\,\mathrm{m}$ interval in blue show the total post–rift (Aptian to present) sediment thickness from *Stewart et al.* (2000). Yellow star indicates the location of the Kudu 9A-1 well. Key geological features also labelled, from *Bauer et al.* (2000).

using the same method as was used in the observed seismic data. There is a good general agreement between the two data sets, both of which show increasing dip with depth beneath the top of the SDR package. The most common values of dip are similar ($\sim$5°) and occur at shallow depths. The main discrepancies are for high dips and deep depths. The calculated dip vs. depth plot flattens with

Table 7.1: Summary of the main parameters used in the flexural and gravity model applied to the Orange Basin, Namibia.

Name	**Symbol**	**Value**
Young's Modulus	E	1.0×10^{11} Pa
Poisson's ratio	σ	0.25
Acceleration due to gravity	g	$9.8\,\mathrm{m\,s^{-2}}$
Zero-elevation crustal thickness	-	31.2 km
Density of crust	ρ_{crust}	$2800\,\mathrm{kg\,m^{-3}}$
Density of upper mantle	ρ_{mantle}	$3330\,\mathrm{kg\,m^{-3}}$
Density of load	ρ_{load}	$2950\,\mathrm{kg\,m^{-3}}$
Density of volcaniclastic infill	ρ_{infill}	$2700\,\mathrm{kg\,m^{-3}}$
Dyke height	h	15 km
Dyke width	w	5 km
Elastic thickness	T_e	*Variable*

depth while the observed dip vs. depth continues to increase. This is probably the consequence of uncertain velocities near the base of observed SDR packages. In order to quantify the best–fit T_e in terms of the resulting dip distribution, we calculate the cosine similarity between the normalised, binned datasets shown in the histogram (Figure 7.2b). We find a best–fit T_e to the observed dips in the range 2–3 km.

7.2 Application of POGM

Figure 7.3 shows the dyke and SDR anomaly, together with the sum anomaly which reflects the gravity effect of all the magmatic processes to have modified the margin since rifting. The dyke and SDR anomalies (Figure 7.3a, b) comprises a high flanked by lows which reflects the mass excess of the loads and the mass deficiency of the associated flexures. The sum anomaly (Figure 7.3c) also comprises a high flanked by lows with the high delineating the extent of the dyke loads and the lows the extent of the SDRs. The sum anomaly increases in amplitude and wavelength with increase in T_e. Irrespective of T_e, the model predicts that the maximum gradient of the sum anomaly corresponds to the contact between the dyke load and the SDRs.

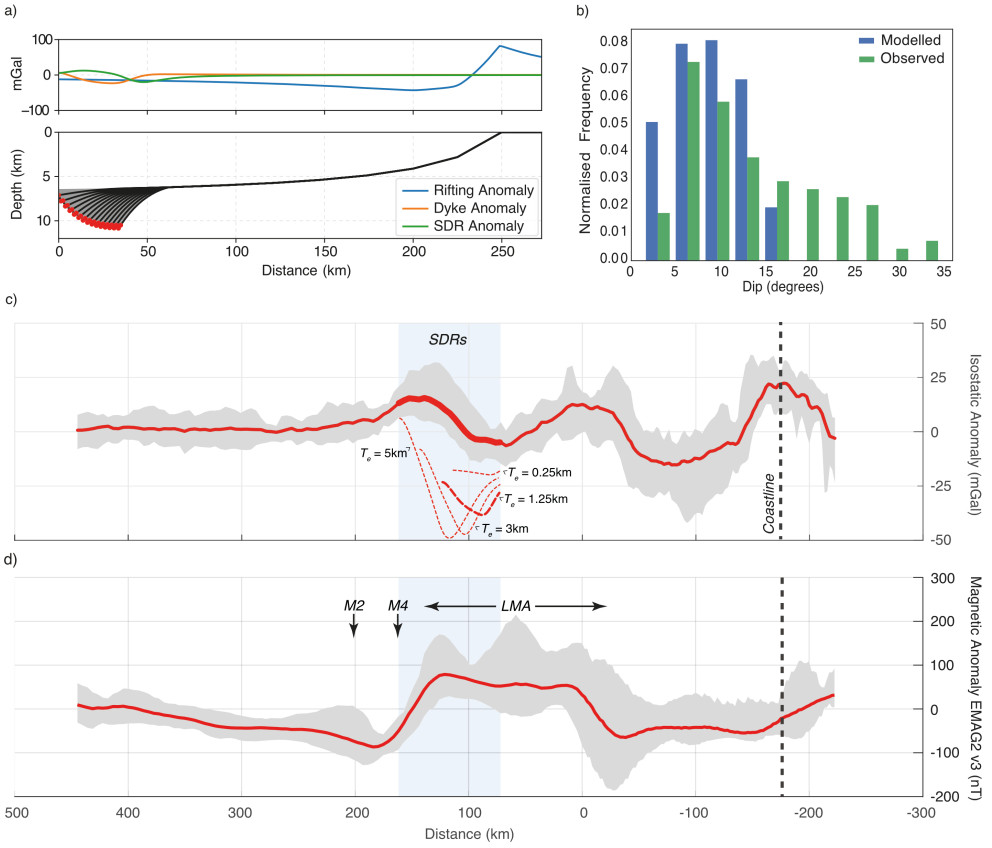


Figure 7.2: a) Post–rift model stratigraphy and components of gravity anomaly for $T_e = 3\,\mathrm{km}$; the rifting anomaly is shown in blue, indicating that the 'edge effect' lies far from the SDRs. The rifting anomaly was calculated from the gravity effect of the water–filled tectonic subsidence and the compensating oceanward thinning of the continental crust, assuming a uniform stretching model and Airy isostasy. The calculated Dyke and SDR anomalies shown are from Figure 7.3, which incorporate the water depth. b) Normalised histogram of observed vs. modelled dips for a T_e of $3\,\mathrm{km}$. c) Ensemble average (red) of observed Airy Isostatic Anomaly profiles (range shown in grey) shown in Figure 7.1. The blue shaded vertical bar indicates the region in which SDRs have been located on seismic reflection data; the dashed black line indicates the intersection with the coastline. The dashed red lines show the modelled result (offset vertically for comparison) for T_e of $0.25\,\mathrm{km}$, $1.25\,\mathrm{km}$, $3\,\mathrm{km}$ and $5\,\mathrm{km}$, with best–fit $1.25\,\mathrm{km}$ in bold. These curves are shown only for the region in the model which corresponds with the extent of the SDRs and are positioned such that the well–constrained landward edge of the SDR package matches the observed. d) Corresponding ensemble average of the observed magnetic anomaly (EMAG2 v3, (*Meyer et al.*, 2017)). 'LMA', 'M4' and 'M2' indicate the location of the magnetic anomalies (*Rabinowitz*, 1976; *Moulin et al.*, 2010). The seaward gradient of the 'LMA' has been interpreted by *Collier et al.* (2017) as the seaward limit of the outer SDRs, consistent with our seismic and gravity anomaly observations.

The sum anomalies in Figure 7.3 do not include the rifting anomaly and so they can be compared directly to observed Airy isostatic gravity anomalies. This comparison is valid because the Airy isostatic gravity anomaly is the free–air anomaly

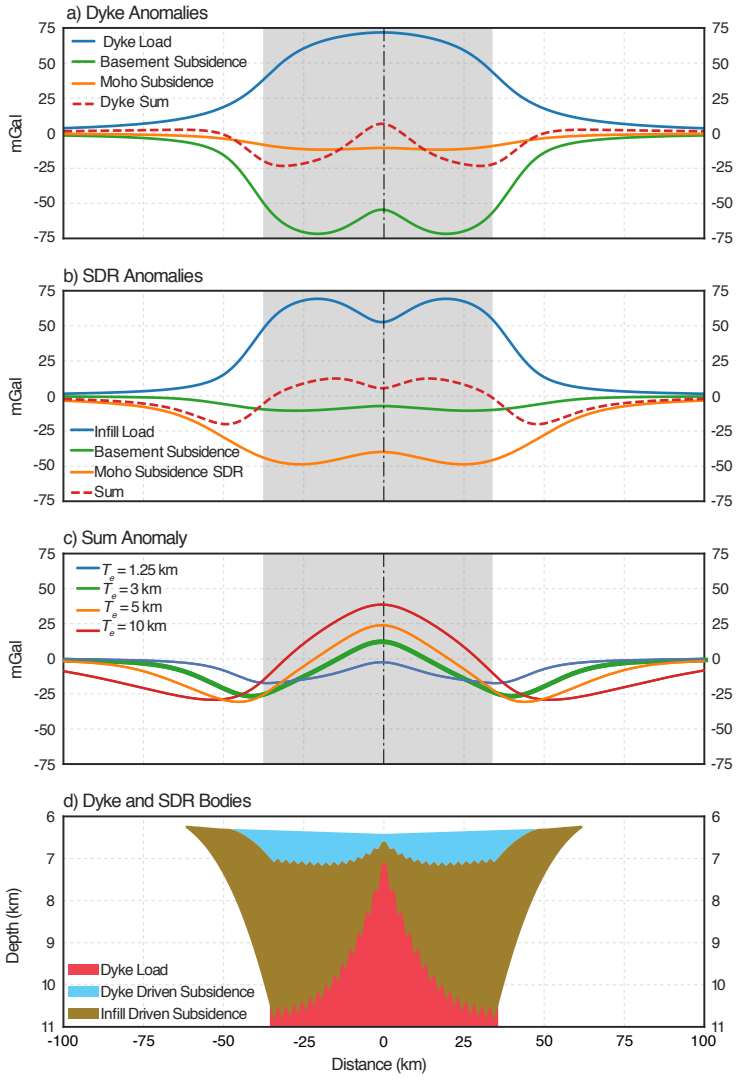


Figure 7.3: Process–oriented gravity modelling of SDR formation using values from Table 7.1 with a dyke height of 15 km and a T_e of 3 km. a) Dyke gravity anomaly. b) SDR gravity anomaly. c) Sum gravity anomaly (a + b) with $T_e = 3$ km shown in the thick line, other T_e values (all other parameters equal) shown for comparison. d) Dyke and SDR bodies. Red shade = dyke load. Blue shade = dyke driven subsidence and subsequent infill load. Green shade = infill driven flexure. The final SDR package will incorporate both the blue and brown regions. Vertical dashed line in figures (a–c) delineates the location of future continental rifting and break–up; grey shaded rectangle indicates the lateral extent of dyking.

less the gravity effect of the topography and its Airy–type compensation, so the anomaly already includes a correction for the thinning of the crust that results from rifting.

We have not corrected the observed Airy isostatic gravity anomaly for the effect of the post–rift sediments and their compensation. However, these sediments

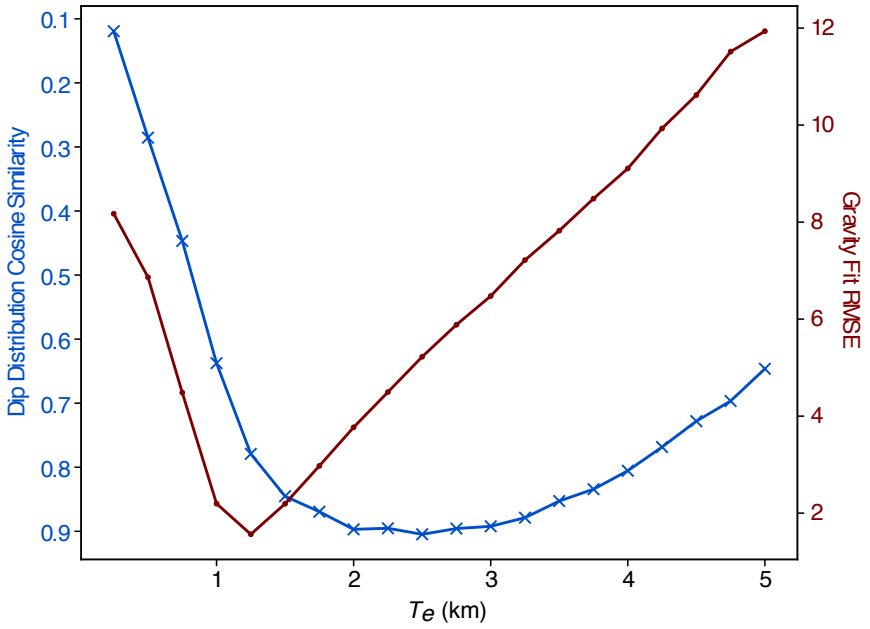


Figure 7.4: Measure of best–fit for different T_e values to observed gravity anomaly (red circles, root–mean squared error) and dip distribution (blue crosses, cosine similarity) in Orange Basin, Namibia. The cosine similarity is calculated from normalised vectors of binned dip measurements.

have accumulated over an area of the continental slope and rise that is much larger than individual SDR packages and so their gravity effect will be significantly longer in wavelength than the contributions associated with magmatism. The calculated gravity anomaly also fits observations (Figure 7.2c), matching the Airy isostatic gravity anomaly gradient at the landward edge of the SDRs reasonably well. Quantifying the fit to the observed isostatic anomaly using a root–mean squared error, the best–fit T_e is 1.25 km (Figure 7.4). This fits the gradient over the dyke–SDR contact very well, but the width of the package is better fit by a slightly higher T_e, in the range 3–5 km. This produces a larger amplitude gravity anomaly, within the range of the observed profiles shown in Figure 7.2c, but larger than the ensemble average.

7.3 Discussion

Unusually for a volcanic rifted margin, a second gravity anomaly high is present offshore Namibia, inboard of the isostatic gradient coincident with the M4 magnetic anomaly. The landward gradient of this high correlates with the 'G' magnetic anomaly. This has previously been modelled as an edge effect associated with the Ocean/Continent boundary (*Talwani and Eldholm*, 1973; *Rabinowitz*, 1976). *Moulin et al.* (2010) redefined the 'G' anomaly, when grouped with the M11 and M9 anomalies of *Rabinowitz and LaBrecque* (1979), as the Large Marginal Anomaly (LMA). The LMA (Figure 7.2d) is characterised by a large amplitude and irregular width (*Moulin et al.*, 2010), unlike linear seafloor spreading anomalies, and corresponds well with SDRs mapped in seismic studies of in the Walvis Basin (*Bauer et al.*, 2000; *Koopmann et al.*, 2014).

Magnetic modelling of SDR packages in the South Atlantic by *Collier et al.* (2017) demonstrated that the inner SDRs, which lie above transitional continental crust, would not be associated with detectable linear anomalies. In contrast, the outer SDRs, underlain by igneous crust composed of continuous dykes, as we propose here, are associated with linear magnetic anomalies. We find a second, landward package of sub–horizontal, dipping reflectors in the Orange Basin, but this package does not correspond with a gradient in isostatic anomaly, as expected within our proposed flexural model. Nor does the package show the typical arcuate down–dip thickening of reflectors characteristic of outer SDRs. This supports an alternative origin for the inner SDRs, likely developed at an earlier stage of rifting, which may be fault controlled.

7.4 Implications of Model

We have assumed that the architecture of SDRs reflects magmatic processes that have modified the margin since the initiation of extension. Further, we assume that only an initial mechanical subsidence followed by a gentle thermal subsidence have

modified them subsequent to formation. Therefore, we have ignored the possible contributing effects of post–rift sedimentation, underplating, faulting and inversion to the present day dips of SDRs. We point out, however, that we have been able to explain the observations without invoking other processes suggesting that the combined effects of these other factors on SDR geometry are small. Therefore, a relatively simple model of dyke–driven flexure in combination with an initial and thermal subsidence can explain the observations and this is despite the fact that the initial flow of basalt during SDR formation may have been in a landward direction, away from the axis of the future rift.

7.4.1 Isostatic Gravity Anomalies

The Airy isostatic gravity anomaly is a useful parameter with which to examine the structure of rifted margins. *Talwani and Eldholm* (1973), for example, used the landward gradient of a prominent Airy isostatic anomaly high to locate a change in basement age and depth which they attributed to the ocean/continent boundary. Other explanations of the high include flexure due to sediment loading (*Karner and Watts*, 1982). We have interpreted the landward gradient of the high as marking the boundary of the seaward limit of the SDR package at the contact between the low density (compared to crustal rocks) infill material and the high–density dykes. The landward gradient does not, therefore, correspond to the Ocean/Continent boundary per se but, lies somewhat landward of it and marks the transition from 'magmatic' intruded stretched continental crust with associated SDRs to true continental crust.

7.4.2 Flexure and the Role of Faulting

We have assumed thus far a model in which outer SDRs that abut oceanic crust form by dyke–driven loading and associated flexure. Listric faults and landward dipping detachments (*Gibson et al.*, 2006; *Eldholm et al.*, 1995; *Geoffroy et al.*, 2015) which could help accommodate SDRs within the brittle upper crust are not

required in order to explain the seismic and gravity observations. Some form of faulting may, however, accompany SDR formation as is evidenced by the high plate curvatures (up to $\sim 10^{-5}\,\mathrm{m}^{-1}$) which would normally be indicative of significant yielding of the plate.

We note that at volcanic rifted margins with distinct sets of inner and outer SDRs, it has been proposed that they form by different mechanisms active during different stages of rifting. Outer SDRs form by loading and flexure, as modelled here, while inner SDRs may be fault controlled (*Barton and White*, 1997a; *Planke et al.*, 2000). We do not know if the inner and outer SDRs at these margins are contemporaneous. One possibility is that inner SDRs are older than outer SDRs and reflect brittle deformation in the shallow crust, while outer SDRs reflect flexure of a weakened lithosphere with an elastic 'core' that has been thinned by both brittle and ductile deformation.

7.4.3 Long–term Strength of Extended Continental Lithosphere

We know from numerical geodynamic modelling (*Burov and Poliakov*, 2001) that rifting changes the integrated strength of the lithosphere: weakening it during the syn–rift and strengthening it during the early post–rift. SDR formation could occur during either or both of these contrasting strength regimes. We found that decreasing T_e for each dyke as loading progresses produces offlap at the edges of a SDR package and a flattening of its base (Figure 7.5b). This tendency to offlap could produce long 'trains' of near–horizontal SDRs with no single pinch out point, or flexural node, as is frequently observed. This is consistent with the pre–existing crust in the region of intrusion weakening as rifting progresses.

In contrast, we found that increasing T_e for successive dyke loads results in onlap at the edges of a SDR package and a marked relief at its base (Figure 7.5a). However, onlap is not typically observed in SDR packages, suggesting they are emplaced during the early weakening phase rather than during the later strengthening phase

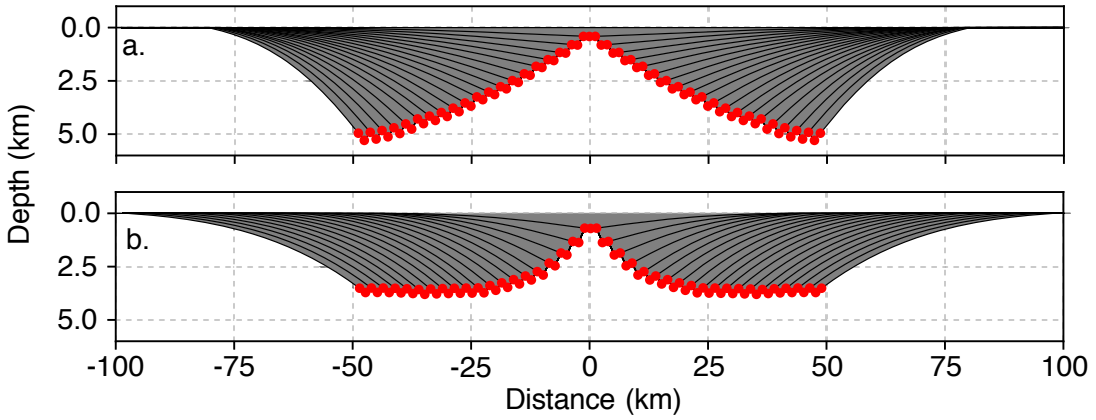


Figure 7.5: a). Increasing T_e and b). Decreasing T_e with successive loads resulting in onlap and offlap respectively. Parameters used are shown in Table 7.1, with a dyke height of 15 km. In both cases, the total width of dyking is 100 km, with the T_e varying between 3–6 km.

of rift margin evolution. In Chapter 4, it was noted that onlap was observed over a small part of the most landward end of the SDR package (Figure 4.11). This could indicate a brief period of initial strengthening of the crust at the beginning of significant intrusive activity, followed by prolonged weakening.

7.5 Synthetic Seismic Reflection Data

An outstanding question is whether our models for SDR formation are compatible with the internal structure and reflectivity of an individual SDR package. In order to address this question, we have used a convolutional method (*Yilmaz*, 2001) to ray trace flexural model outputs and construct a preliminary set of synthetic zero–offset Common Depth Point gathers that can be compared to observed reflection profiles.

We assumed a velocity/density model in which the dyke–driven infill comprises volcaniclastic sediment with intercalated lava flows, by alternating the infill value (2600–2700 kg m^{-3}) between dykes. We found that such a layered sequence, as shown in Figure 7.6, which is compatible with direct sampling data (*Teagle and Alt*, 1999), is necessary within the infill to provide the impedance contrast required. We also found that the dyke loads that generate the infill will not be imaged due

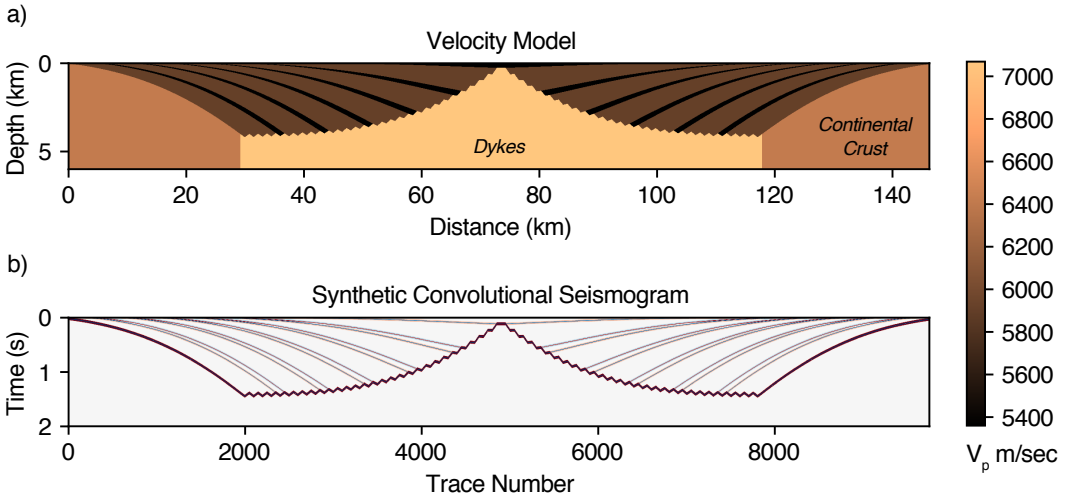


Figure 7.6: a. Velocity model produced by flexural SDR model and b. corresponding zero–offset convolutional gather (*Yilmaz*, 2012) produced using a Ricker wavelet source with a sample time of $2\,\mathrm{ms}$. Input model with infill density alternating between $2700\,\mathrm{kg\,m^{-3}}$ for $12\,\mathrm{km}$ in width of dyke intrusion and $2600\,\mathrm{kg\,m^{-3}}$ for the next $3\,\mathrm{km}$ of dyke intrusion. Other parameters are given in Table 7.1, with a T_e of $3\,\mathrm{km}$. Velocities are obtained from the densities in the flexural model using the relationships of *Brocher* (2005). The results presented here were obtained with the help of the open–source software Fatiando a Terra by *Uieda and Oliveira Jr* (2013).

to their steep, near vertical, contact with the surrounding crust. If there is no significant variability in the density of the intruding dykes through time, there will also be a lack of density/velocity contrasts to image within the dyke body.

7.6 Rheological Implications of Model Results

The thin elastic plate model is inherently limited by the assumption of pure elasticity, such that the plate is able to accumulate infinite stress. The response of the Earth's lithosphere to imposed stresses (in tension or compression) can be more accurately represented using a yield strength envelope (YSE), which combines brittle and ductile regimes into a single strength profile. Often referred to as a 'sailboat diagram' due to the characteristic shape (Figure 7.7), in both the continents and oceans the upper part of the layered rheology is characterised by brittle failure and the lower by ductile, separated by the brittle–ductile transition.

In the crust, a notch below this transition marks the shallower onset of ductility in typical crustal compositions compared to that of mantle olivine.

Like T_e estimates, the YSE provides an estimate of the integrated strength of the lithosphere. This can be calculated by considering the total area under the 'sail'. With understanding of the appropriate failure conditions and composition of the lithosphere, the YSE therefore provides an estimate of the maximum stress difference the lithosphere can support by flexure before yielding.

Goetze and Evans (1979) demonstrated using a YSE model that the bending moment had a non–linear dependance on the curvature of the plate, while an elastic model assumes a linear dependance. The bending moment for an elastic plate is found by integrating the fibre stresses with distance from the neutral surface; in a YSE model, the moment has contributions from three different regimes within the plate — the brittle upper part, the elastic core, and the ductile lower.

McAdoo et al. (1985) found an analytical solution for the bending moment using a layered rheology YSE model, and showed that unlike the elastic plate model, bending moments converged on a limiting value. Using deflection profiles from trenches and fitting the elastic and layered rheology solutions in a least squared sense, *McAdoo et al.* (1985) found the elastic model derived T_e was around 30% less than the value estimated from the layered rheology model. As a result, elastic plate derived estimates of T_e may underestimate the true elastic thickness of the plate in settings where sufficient yielding has occurred. In regions where curvature is convex up, such as this buried loading case, yielding is increased further by interactions between flexural and residual thermal stresses (*Wessel*, 1992).

The T_e values resulting from the Namibia Case Study are small; however, *Buck* (1988) and *Roberts et al.* (1993) have used similar small values in order to constrain the geometry of faulting and syn–rift basins during the initial stages of continental break–up. The T_e values imply curvatures of up to 10^{-5}–$10^{-6}\,\mathrm{m}^{-1}$ which, depending on the crust and upper mantle rheology, suggest significant yielding ($< 60 - 70\%$) of the flexed plate and a reduction in thickness as the plate thins from its short–term mechanical thickness to its long–term elastic thickness. In

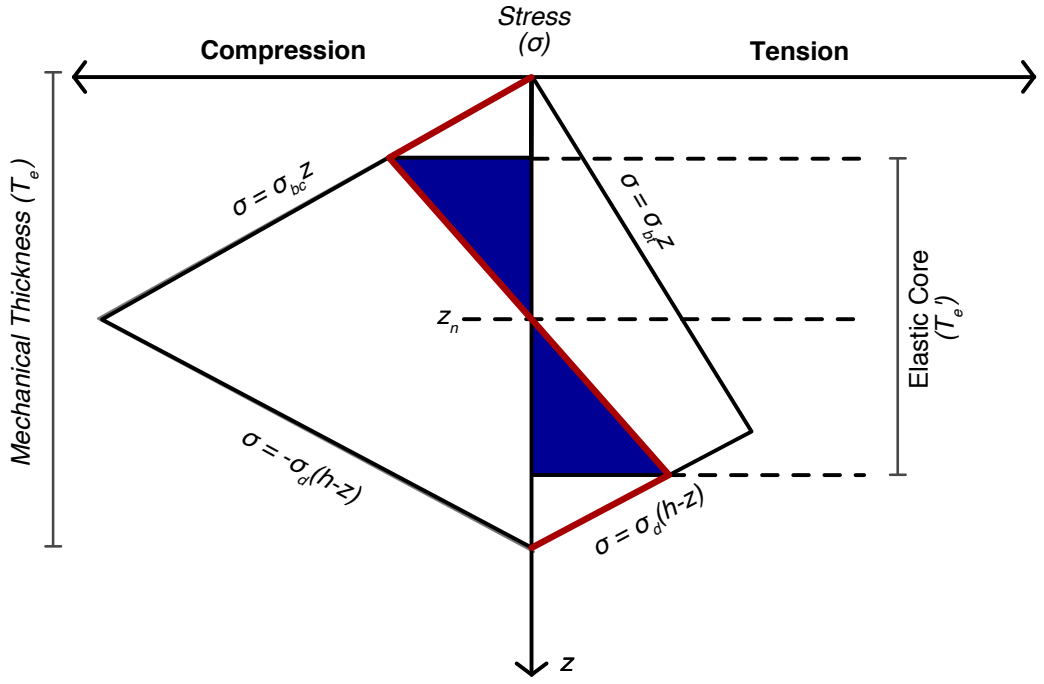


Figure 7.7: Schematic of Yield Strength Envelope (YSE) demonstrating effective elastic thickness (T_e'), modified from *Wessel* (1993); *Watts and Burov* (2003). Shaded blue areas represent the elastic core, where imposed stress is less than the yield strength. Bold red line over envelope represents the deviatoric bending stresses as a function of depth. z_n is the depth to the neutral plane, where stress is equal to zero. The full mechanical plate thickness (T_e) is contrasted with the thickness of the elastic core (T_e'), which is the value which would be calculated using a constant thickness plate model as presented in this chapter. Each boundary of the YSE shows the failure criterion, brittle in the upper part and ductile in the lower, as a function of depth. Typically, Byerlee's law is used for brittle failure while a power law will be used for ductile.

addition, the effective T_e calculated in our model using a constant elastic thickness plate model is likely to be a significant underestimate, by 30–50 %, (*McAdoo et al.*, 1985) of the true mechanical thickness of the plate.

7.7 Chapter Summary

- The flexure parameters that best–fit seismic and gravity anomaly data suggest either a broken plate or a continuous plate that has been significantly weakened at the point of intrusion, and is associated with relatively low values of elastic thicknesses (1–3 km) with yielding.

- The lack of onlap at the landward edges of SDRs suggests that they are emplaced when the long–term strength of the lithosphere is progressively weakening.
- Curvatures of up to 10^{-5}–$10^{-6}\,\mathrm{m}^{-1}$ imply significant yielding and a corresponding reduction of the effective T_e of the plate.

Chapter 8

Conclusions and Future Work

8.1 Summary of Results

8.1.1 SDR Observations

In Chapter 2, a representative dataset of SDR measurements is compiled from seismic reflection profiles covering the major volcanic rifted margins. In each case, the dip, thickness and width of the package is measured and averaged at different scales, from which SDR formation models may be constrained. The measured dip and thicknesses of SDR packages does not correlate with the year of acquisition, demonstrating that changes in methodology and processing power over the years has not introduced a sampling bias. Using a suite of 1D velocity profiles, a windowed moving average is used to calculate the dip which fully captures the arcuate shape. Dips measured with this method are typically in the range 5–15°, with few values greater than 30°. Widths (perpendicular to strike) are more variable, but typically <75 km. Package thicknesses (in depth/time) are again subject to the uncertainty of conversion to depth, but are typically 1–6 km. SDRs typically lie in several kilometres of water depth, beneath significant thicknesses of post–rift sediment, making drilling challenging. Where the landward edge of (typically inner) SDR packages has been drilled, basalts interbedded volcaniclastic sediments are found.

8.1.2 Seismic Processing

In Chapter 3, a publicly accessible dataset of seismic reflection field data from the Greenland margin is re–processed to test the sensitivity of the final SDR geometry to, in particular, semblance analysis. We found that measures of average of dip seem fairly robust to velocity choice, while the extremes are less so. Conversely, minimum and maximum thickness measurements are reasonably robust, with some variability in the averages. Later in the thesis, mean dip and dip distribution, and minimum and maximum thicknesses, have been used as the main geometric constraints in flexural modelling. This selection of metrics do appear to be the most robust, given a plausible range of semblance velocity choices for this dataset.

8.1.3 Seismic Interpretation

In Chapter 4, a set of three 2D seismic reflection surveys from the Orange Basin, Namibia, is interpreted, focussing on the volcanic features of the margin. Comparing SDR imaging across the surveys of different vintage, the overall geometry of the outer SDR package is consistently imaged. However, individual reflectors within the package and particularly within the transition to oceanic crust may not always be consistently observed. We find that the seaward termination of the SDR package is variable in position along strike of the margin, although the marked transition to normal thickness oceanic crust seaward of the transition package is more consistent and correlates well with magnetic lineations. Measuring the width of the outer SDR package (perpendicular to margin strike) is quite consistent along strike, varying between 45–75 km, with no overall trend. Considering individual reflectors, it is rarely possible to correlate any horizon over a distance greater than 10 km. However, it is unclear whether this is a limitation of 2D seismic reflection surveying or a geological observation.

A package here termed the ‘transition package’ is identified between the seaward termination of SDRs and normal thickness oceanic crust. This package varies in thickness between 0.5–1.1 s TWTT in the Orange Basin and likely represents the

submarine transition in the Orange Basin. A second, inner SDR package is present in the Orange Basin, but reflectors are linear, not arcuate. This more landward package has a different isostatic anomaly signature in comparison with the outer, suggesting the two packages are formed by different mechanisms.

8.1.4 Flexural Model

In Chapter 5, a dyke–driven flexural loading model with progressive extension is developed. While we find that a surface loading based model can produce SDR–like stratigraphy in the flanking moats, buried loading produces flexure of a greater magnitude and is more consistent with observations. The sub–surface dyke load produces a surface flexure, which are subsequently infilled by volcaniclastic and basaltic flows. This infilling material also constitutes a load on the lithosphere, producing further flexure and infill. This model can account for the width, thickness and down–dip thickening of outer SDR packages as revealed in seismic reflection profile data over volcanic rifted margins. A flexural origin for outer SDRs is supported by isostatic gravity anomaly 'highs' and flanking 'lows' which reflect the dyke and infill loads and resulting flexures respectively.

8.1.5 Gravity Anomalies

In Chapter 6, a global grid of the Airy isostatic anomaly is calculated, highlighting key geologic features which may not be easily studied with the free–air or Bouger anomalies alone. This grid, in addition to EMAGv2, is used as input to an ensemble averaging technique. This method is used to extract along–strike signals from margins and study both the resulting average and residual maps for key VRMs. Applying this averaging technique to Greenland and Namibia highlights trends in the isostatic and magnetic anomalies that may be attributed to magmatic features.

A process–oriented gravity modelling technique is developed for the proposed flexural model, differentiating between the contributions of the dyke intrusion and

volcaniclastic infill processes. This is contrasted with the more traditional object–oriented gravity modelling approach (OOGM). Using this POGM approach suggests that the shape of the gravity anomaly for a fault–bound SDR package will differ from that of a dyke–driven SDR package, with a characteristic shift in the position of the maxima.

Given a configuration of surfaces and corresponding density contrasts, two methods of calculation of the gravity anomaly are available: FFT, commonly used in POGM, and line integral, commonly used in OOGM. In order to constrain the modelling results with observed isostatic anomalies, an RMS–based technique is used to quantify the goodness of fit and therefore estimate the best–fit parameters.

8.1.6 Namibia Case Study

In Chapter 7, the flexural and process–oriented gravity model developed in Chapters 5 and 6 are applied to the Namibian margin, using the interpretation of the seismic reflection dataset in Chapter 4. We find that the flexure parameters that best–fit seismic and gravity anomaly data suggest either a broken plate, or a continuous plate that has been significantly weakened at the point of intrusion, and is associated with relatively low values of elastic thicknesses (1–3 km) with yielding. The lack of onlap at the landward edge of the SDR package suggests that they are emplaced when the long–term strength of the lithosphere is progressively weakening. The best–fit modelled output has curvatures of up to 10^{-5}–$10^{-6}\,\mathrm{m}^{-1}$, which imply significant yielding and a corresponding reduction of the effective T_e of the plate.

8.2 Future Work

8.2.1 Applying the Modelling Approach to Other VRMs

With access to more seismic reflection data, an approach like that used on the Namibian margin in Chapter 7 could be applied to other VRMs. This requires precise knowledge of the location and geometry of the SDR package in order to constrain the flexural model and establish goodness of fit for different T_e values. Once the best–fit T_e has been found, the corresponding process–oriented gravity model result can be compared to the expected gravity anomaly profile for alternative models; if the SDRs were fault bound, for example, the dyke component of the anomaly, which causes the high beneath the termination of the SDRs, would not be present. Understanding where SDRs are formed by faulting and where they are purely dyke load driven would improve our understanding of regional margin development.

Similarly, if we expect inner SDRs and outer SDRs to be formed by different mechanisms on the same margin, applying the model in tandem on the same margin could provide evidence for this. If we found gravity modelling suggested that inner SDRs are indeed consistently fault–bound within continental crust, while outer SDRs are formed by dyke–driven flexure, this could provide a better understanding of the continent–ocean transition, which is often difficult to precisely locate in the absence of seismic refraction. Understanding the origin of the crust beneath the SDRs has important implications for heat flow on the margin, particularly when syn– and post–rift sedimentation may be of interest for petroleum exploration.

Applying the model to SDR packages on conjugate margins could also provide insight into variations in strength at the early stages of rifting. If T_e values are not symmetric for similarly aged SDR packages on conjugates, it is likely that other factors contribute to this strength e.g. pre–existing crustal weaknesses or intrusions. In this study, we concluded that a very low effective T_e was required in order to produce, in particular, the magnitude of dips observed in SDR packages.

The results of the global compilation suggest that dips in the range used in the Namibian margin case study of 5–10° are typical; it is therefore reasonable to consider whether such small values are required to form SDRs. Whether alternative volcanic structures are present in higher T_e settings, or volcanism is restricted to low T_e settings, would be difficult to ascertain purely from seismic reflection alone as the characteristic dip of SDRs would be absent.

8.2.2 Expanding PySDR Model Code

In this thesis, the SDR modelling approach used a 2D approach in which properties such as the T_e vary in one dimension only. Expanding the modelling approach to three dimensions would permit further exploration of the geometry of SDRs in the along strike direction. A 3D approach to flexure would allow variation of T_e in two dimensions, investigating, for example, the role that intersecting transform faults may have on SDR formation.

The core functionality of the Python code allows loading of a mass of a pre–defined height, width and density onto a plate, from which the associated gravity anomaly is calculated. Secondly, the plate can be split at a defined coordinate and undergo an amount of defined extension. This has been applied to dyke loading and subaerial spreading in this thesis, but could be modified relatively easily to provide a useable framework for any flexure–based calculation and associated process–oriented gravity model. By using these functions in an iPython notebook with inline data visualisation, this could provide a quick method to explore the parameter space around new loading problems in many different geological settings (e.g. seamounts, intrusive activity in continental settings). In addition, adding the role of an underplate as a 'buoyant' load, which could flex the plate upwards. In the context of VRMs, this additional uplift may cause interesting interactions with the formation of SDRs.

One important aspect of margin development that is not captured by the existing model is the impact of changes in heat flow, which are likely to be significant

in this setting. This could be investigated using numerical modelling, which is frequently applied to more mature spreading at mid–ocean ridges.

8.2.3 Acquiring Additional Data

In order to establish the origin of SDRs, we consider a hypothetical new seismic acquisition survey. Coupled with deployment of OBS, a seismic reflection and refraction survey over SDRs which have not previously been studied in this detail could be very informative. For example, two profiles in the Orange Basin, targeting a wider and narrower section of the SDR package (see Figure 4.13) and extending out over the transition package into true oceanic crust could provide crucial velocity information. Coupled with the results of *Bauer et al.* (2000), this would enable an along–strike comparison the magmatism on the Namibian margin and help determine conclusively the nature of what in this thesis is referred to as the transition package.

While drilling through a typical outer SDR package is currently beyond the abilities of modern engineering, the ultimate test of SDR formation models would be to directly the sample the basal surface of the SDR package. The IODP's Chikyu, already being used to investigate near future projects to drill the Moho near Hawaii, may soon have this capability.

It is difficult to conclusively prove a flexure model in either case; the presence of high velocity magmatic crust and/or directly sampled dykes beneath the SDR would support the model proposed in this thesis, but would not be conclusive. However, the presence or absence of a fault is easier to ascertain. If a fault surface is present, it would then be important to quantify the throw on the fault and therefore the amount of rotation of the SDR package that is can account for.

8.2.4 What Makes a Margin Magmatic?

The effective T_e values suggested by the Namibian margin case study in Chapter 7 are very low, <5 km. In the proposed flexural model for SDRs the crust beneath the SDR package is entirely magmatic in origin. This is supported by many other studies which have identified magmatic or over–thickened oceanic crust in these settings. As SDRs are one of the defining features of VRMs, is in fact the presence of this magmatic crust which distinguishes volcanic from non–volcanic margins? We could therefore view SDRs as subaerial equivalent of oceanic crust layer 1. This reframes the distinction between volcanic and non–volcanic margins, focussing instead on the nature of the continental–oceanic transition. A VRM may then be viewed as a margin which has undergone a period of subaerial extension, before subsiding to depths at which submarine extension produces normal thickness oceanic crust.

Appendix A

Namibia Seismic Data Headers

Please see below for example SEGY headers for each dataset.

A.1 ECL-89

```
C01VERITAS DGC LIMITED LINE ECL-89-** SP 1 - 23463
C02 RECORDED BY HGS PARTY 2938  APRIL - FEBRUARY 1990
C03 REEL: ECL_FINMIG_STV     DATASET: FINAL MIGRATION
C04VESSEL: M.V. PATRICK E. HAGGERTY BEARING:  56.5 DEGREES
C05 RECORDING FORMAT: SEG-D
C06 SAMPLE RATE 2 MS; RECORDING LENGTH : 7 OR 5 SEC
C07 LOW CUT FILTER: 8 HZ 18DB/OCT; HIGH CUT FILTER 180 HZ 72DB/OCT
C08 240/228 GROUPS,GROUP INT. 15M,  CABLE LENGTH 3735/3555M, CABLE DEPTH 8M
C09 NEAR OFFSET: 150 M SOURCE: SLEEVE GUNS 2180 CU. IN. SP INT: 30M, 6M DEPTH
C10 CDP INT: 7.5M 60 FOLD, POS PRESSURE AT HYDROPHONE PRODUCES NEG NO. ON TAPE
C11 VESSEL POSITIONING PRIMARY: ARGO/SYLEDIS SECONDARY: TRANSIT SATELLITE
C12 PROCESSING HISTORY: REFORMAT FROM SEG-D AND EDIT
C13 ANTI-ALIAS FILTER THEN RESAMPLE FROM 2MS TO 4MS
C14 ZERO PHASE DESIGNATURE, TT SPHERICAL DIVERGENCE CORRECTION
C15 SWELL NOISE AND SPIKE ATTENUATION, FX TRACE INTERPOLATION TO 7.5M CHANNELS,
C16 TAU-P DOMAIN MUTE, HIGH RESOLUTION TAU-P DIP FILTER,
C17 FREE SURFACE MULTIPLE ATTENUATION, 2KM VELOCITY ANALYSIS
C18 PREDICTIVE GAP DECONVOLUTION IN TAU-P DOMAIN, 250MS OPERATOR PLUS 64 MS GAP ,
C19 DUAL GATE DESIGN AND APPLY,
C20 REMOVAL OF INTERPOLATED TRACES, BACKGROUND AMPLITUDE CORRECTION,
C21 PRE-STACK TIME MIGRATION USING FULL CURVED RAY KIRCHHOFF,
C22 NMO USING FINAL VELOCITY MODEL PICKED EVERY 1KM,
C23 HIGH RESOLUTION RADON MULTIPLE ATTENUATION,
C24 OUTER TRACE MUTES AND WATER DEPTH DEPENDENT INNER MUTE,
C25 60 FOLD STACK WITH 1/N SCALING,
C26 EXPONENTIAL GAIN RECOVERY: WB DEPENDENT,
C27 LOCALISED REMNANT MULTIPLE ATTENUATION,
C28 COHERENCY ENHANCEMENT IN F/K, SPECTRAL BALANCING
C29 TIME VARIED BAND PASS FILTER, DUAL GATE DESIGN BALANCE GATES.
C30 11MS SHOT AND RECEIVER STATIC
C31 SHOTPOINT    : BYTE  17 (4)
C32 CDP          : BYTE  21 (4)
C33 FIELDRECORD  : BYTE   9 (4)
C34 SEGTYPE      : BYTE  29 (4)
C35 SRATE        : BYTE 117 (4)
C36 X            : BYTE  73 (4)
C37 Y            : BYTE  77 (4)
C38
C39
C40 PROCESSED BY VERITAS GEOPHYSICAL SERVICES A/S, NORWAY.
```

A.2 ECL-89-Repro

```
C 1 Multi-Client Division
C 2 Survey Area               Offshore Namibia
C 3 Line                      ECL-89-**
C 4 Shot Point Range          1 - 3400/23401 - 27644
C 5 CDP Range                 1 - 15406
C 6 Data Type                 Final PSTM Stack
C 7 ************************************************************************
C 8 Acquisition:
C 9 Acquired By    Western Geophysical         Shot Point Interval    30 m
C10 Vessel         P.E. Haggerty               Group Interval         15 m
C11 Date           December 1989               Record Length          7  s
C12                                            Sample Interval        2 ms
C13 ************************************************************************
C14 Processing:
C15 Processed By       Spectrum Geo            Processed CDP Interval     15.0 m
C16 Project Number     W3206085                Processed Record Length    7  s
C17 Date               2012                    Processed Sample Interval  4 ms
C18 TIME PROCESSING
C19   Reformat from SEGY / Geometry / Zero Phase Designature / Resample to 4ms
C20   -51.2 Start of Data Delay / Tsqu Spherical Divergence / 3Hz lo-cut
C21   Swell Noise Attenuation in Shot Domain / SRME Demultiple / 2km Velocity
C22   Trace Drop, o/p 120 Channels / High Resolution Radon
C23   TVF / Remove sph div / 1st pass PSTM / 1km Velocity / 2nd pass PSTM
C24   VsquT Spherical Divergence / High Resolution Radon
C25   Automatic Continuous  Vels, 4th order / NMO / Outer & inner Trace Mute
C26   Stack /  Gun & Cable +9.5ms / Coherency Enhance / F/K Filter
C27   TVF / Expanding Window AGC
C28 ************************************************************************
C29 Trace Header Information:
C30 Shot Point Number (multiplied by 100)     Bytes 17 - 20 (32 bit integer)
C31 CDP Number                                Bytes 21 - 24 (32 bit integer)
C32 Co-ordinates:
C33      Geodetic Datum     WGS-84, meter     Projection   UTM Zone 33S
C34      CDP X (multiplied by 100)            Bytes 189 - 192 (32 bit integer)
C35      CDP Y (multiplied by 100)            Bytes 193 - 196 (32 bit integer)
C36
C37
C38
C39
C40 USE IS SUBJECT TO ACCEPTANCE OF SPECTRUM'S DATA USE AGREEMENT.
```

A.3 SCOB-12

```
C 1 Multi-Client Division
C 2 Survey Area               Orange Basin, Offshore Namibia
C 3 Line                      SCOB12-***
C 4 Shot Point Range          4883 - 798
C 5 CDP Range                 1 - 8572
C 6 Data Type                 Final PSTM Stack (Equalised Amplitudes)
C 7 ************************************************************************
C 8 Acquisition:
C 9 Acquired By    Seabird Exploration         Shot Point Interval    25 m
C10 Vessel         M/V northern Explorer       Group Interval         12.5 m
C11 Date           January 2012                Record Length          10 s
C12                                            Sample Interval        2 ms
C13 ************************************************************************
C14 Processing:
C15 Processed By       Spectrum Geo            Processed CDP Interval     12.5 m
C16 Project Number     W3201017                Processed Record Length    10 s
C17 Date               2012                    Processed Sample Interval  4 ms
C18 TIME PROCESSING
C19   Reformat from SEGY / Geometry / Zero Phase Designature / Resample to 4ms
C20   -40ms Start of Data Delay / Tsqu Spherical Divergence / 3Hz lo-cut
C21   Swell Noise Attenuation in Shot & CMP Domains / SRME Demultiple
C22   Tau-P Linear Filtering / SNA / 2km Velocity / Trace Drop, o/p 402  Channels
C23   High Resolution Radon / TVF / K-Filter / Remove sph div / 1st pass PSTM
C24   1km Velocity / 2nd pass PSTM / VsquT Spherical Divergence
C25   High Resolution Radon / Automatic Continuous  Vels, 4th order / NMO
```

```
C26   Outer & inner Trace Mute / Stack / Coherency Enhance / F/K Filter / TVF
C27   Expanding Window AGC
C28 ***************************************************************************
C29 Trace Header Information:
C30 Shot Point Number (multiplied by 100)     Bytes 17 - 20 (32 bit integer)
C31 CDP Number                                Bytes 21 - 24 (32 bit integer)
C32 Co-ordinates:
C33      Geodetic Datum     WGS-84, meter     Projection   UTM Zone 33S
C34      CDP X (multiplied by 100)           Bytes 189 - 192 (32 bit integer)
C35      CDP Y (multiplied by 100)           Bytes 193 - 196 (32 bit integer)
C36
C37
C38
C39
C40 USE IS SUBJECT TO ACCEPTANCE OF SPECTRUM'S DATA USE AGREEMENT.
```

A.4 NAMZA-14

```
C 1 Multi-Client Division Spectrum Geo
C 2 Survey Area      Orange Basin, Offshore Namibia/South Africa
C 3 Line             NAMZA-OB-2D14-***
C 4 Shot Point Range 8372 - 1094
C 5 CDP Range        1 - 14958
C 6 Data Type        Kirchhoff PSTM Stack Filtered and Scaled M-KPSTM-FS
C 7 ********Acquisition ************************************************************
C 8 Acquired By     BGP CHALLENGER               Shot Point Interval    25 m
C 9 Date            2014                         Group Interval         12.5 m
C10 Record Length   10050 ms                     Source Depth           7 m
C11 Sample Interval 2 ms                         Streamer Depth         10-15  m
C12 No. of Channels 804                          Cable Length           10050  m
C13 ********Processing ************************************************************
C14 Processed By       Spectrum Geo   Processed CDP Interval      12.5 m
C15 Project Number     W1405053       Processed Rec.Length        10000 ms
C16 Date               2014           Processed Sample Interval   4 ms
C17 TIME PROCESSING
C18   Reformat / Zero Phase Designature / LC Filt. 3Hz / Resample to 4ms
C19   Tsq Spherical Div. / Start of Data Correction -50ms / Geometry
C20   Shot & Trace Edit / Swell Noise Attenuation / Demultiple - SRME
C21   Vel. Analysis 2km / Tau-P Lin.Noise Attenuation Shots / Adj.Tr.Sum 2 to 1
C22   Demult HR Radon / Diffr. Noise Attn / FX Filter / Spatl.AA Filter / TVF
C23   Targeted Polygonal Gain / Remove Tsq Sph Div / KPSTM / Vel. Analysis 1km
C24   Final KPSTM / VsqT Spherical Div. / Automatic Cont.Vels,4th order
C25   Demult HR Radon / NMO / Outer & Inner Trace Mute / Stack / QFilt Ph only
C26   FK Coher.Enhance / Spatl.AA Filter / TVF / Residual Gain Correction
C27   Gun & Cable Stat 11ms / AGC / SEGY Output
C28  Polarity:Zero Phase SEG Rev; -ve no.(Trough)=increase in Acoustic  Impedance
C29 ********Trace header information ******************************************
C30 Shot Point Number (multiplied by 100)     Bytes 17 - 20 (32 bit integer)
C31 CDP Number                                Bytes 21 - 24 (32 bit integer)
C32 Co-ordinates:
C33      Geodetic Datum       WGS84, meter    Projection   UTM Zone 33S
C34      CDP X (multiplied by 100)           Bytes 181 - 184 (32 bit integer)
C35      CDP Y (multiplied by 100)           Bytes 185 - 188 (32 bit integer)
C36
C37 CDP = (ABS(SP-FSP))*2+402 Delivery Date ***3SEPT14***
C38 Use Is Subject To Acceptance Of Spectrum's Data Use Agreement
C39 SEG Y REV1
C40 END TEXTUAL HEADER
```

Appendix B

Python Code

B.1 PySDR Module Setup

```
from setuptools import setup

setup(name='pysdr',
      version='0.1',
      description='SDR Flexural and Gravity Modelling',
      author='Rebecca Morgan',
      author_email='beccamorgan92@gmail.com',
      packages=['pysdr', 'pysdr.data'],
      install_requires=['numpy', 'matplotlib', 'seaborn',
                        'scipy', 'moviepy'],
      scripts=['scripts/runModel.py'],
      )
```

```
[[source]]
url = "https://pypi.python.org/simple"
verify_ssl = true
name = "pypi"

[dev-packages]

[packages]
numpy = "*"
moviepy = "*"
matplotlib = "*"
seaborn = "*"
flask = "*"
scipy = "*"
pandas = "*"

[requires]
python_version = "3.6"
```

B.2 Flexural and Gravity Modelling

```
#!/usr/bin/env python
from pysdr import SDRs as sdr
from pysdr.dips import dips_from_strat, compare_dips
from pysdr.data.constants import zero_ct
import configparser
import sys
import os
from shutil import copy2


def extract_dyke_param(parser, section, param, loads):
    """
    Get value from parser as list of floats or float

    Assumes list comma seperated with no braces

    Returns val, either float or list of floats
    """
    str_val = parser[section][param]

    if ',' in str_val:
        val = str_val.split(',')
        val = [float(i) for i in val]
    else:
        v = float(str_val)
        val = [v for _ in range(loads)]

    return val


def run_model(file=None, config=None):
    """
    Calculate the response of the lithosphere to dyke intrusions

    Input:
    file = str, name of configuration file in this directory
    config = config variable

    To run as script:
    python runModel.py 'config.ini'
    """
    # Get current directory
    cur_dir = os.getcwd()

    try:
        # Parse configuration file
        if file is None:
            p = config
        else:
            p = configparser.ConfigParser()
            p.read(file)

        # Get output directory
        out_dir = p['Calculation']['OutputDir']

        # If out_dir doesn't exist, make it
        if not os.path.exists(out_dir):
            os.makedirs(out_dir)

        # Place a copy of config in output dir
        copy2(file, out_dir)

        # Change directory to out_dir
        os.chdir(out_dir)

        # Define some shortcuts
        p_int = p.getint
        p_bool = p.getboolean

        # Calculate rifting
        if p_bool('Rifting', 'Rift') is True:
            R = sdr.Rift(p['Rifting']['XRiftOutput'], p['Rifting']['BetaFile'],
                         p_int('Rifting', 'RiftWidthKm') * 1000)
        else:
            R = None

        # Construct jump list
        jump1 = (p_int('Jumps', 'JumpPos1'), p_int('Jumps', 'JumpSize1'))
        jump2 = (p_int('Jumps', 'JumpPos2'), p_int('Jumps', 'JumpSize2'))
        jump3 = (p_int('Jumps', 'JumpPos3'), p_int('Jumps', 'JumpSize3'))

        if sum(jump1) == 0:
            jumps = []

        if sum(jump1) != 0 and sum(jump2) == 0:
```

```
            jumps = [jump1]

        if sum(jump1) != 0 and sum(jump2) != 0 and sum(jump3) == 0:
            jumps = [jump1, jump2]

        if sum(jump1) != 0 and sum(jump2) != 0 and sum(jump3) != 0:
            jumps = [jump1, jump2, jump3]

        # Extract dyke parameters
        loads = p_int('Plate', 'Loads')

        height = extract_dyke_param(p, 'Dykes', 'Height', loads)
        rho_infill = extract_dyke_param(p, 'Dykes', 'RhoInfill', loads)
        rho_load = extract_dyke_param(p, 'Dykes', 'RhoLoad', loads)
        rho_comp = extract_dyke_param(p, 'Dykes', 'RhoComp', loads)
        rho_ti = extract_dyke_param(p, 'Dykes', 'RhoTempInfill', loads)
        te_max = extract_dyke_param(p, 'Dykes', 'MaxTe', loads)
        te_min = extract_dyke_param(p, 'Dykes', 'MinTe', loads)
        te_width = extract_dyke_param(p, 'Dykes', 'WidthTe', loads)

        # Initiate plate
        P = sdr.Plate(p_int('Dykes', 'Width'), loads, jumps, max(te_max),
                      rift=R, bulge_wipe=p_bool('Plate', 'BulgeWipe'))

        # Add dykes to plate
        for i in range(P.loads):
            # Instantiate new dyke
            D = sdr.Dyke.add_dyke(P, height[i], rho_infill[i], rho_load[i],
                                  rho_comp[i], rho_ti[i], te_max[i], te_min[i],
                                  te_width[i], p_int('Plate', 'dx'))

            # Add dyke to plate
            P.add_dyke(D)
            print("Dyke complete: ", i + 1)

        # Build, plot and export stratigraphy
        P.build_strat()
        P.plot_strat(P.strat, P.top_dyke,
                     animate=p_bool('Calculation', 'Animate'))
        P.export_stratigraphy()

        # Gravity Process Oriented
        if p_bool('Calculation', 'GravityPreRift'):
            topped_strat = P.add_zero_top(P.strat)
            topped_load_flex = P.add_zero_top(P.load_flex)
            prerift_anoms = \
                P.calc_volc_POGM(topped_strat, topped_load_flex,
                                 P.top_dyke_lf, P.base_dyke_lf, zero_ct)
            prerift_anoms.plot_pogm_components("prerift_pogm")

        # Get dips
        if p_bool('Calculation', 'DipsPreRift'):
            dips_from_strat(P.strat[:, :, -1], P.top_dyke[:, :, -1],
                            "prerift_model", P.x_plate)

        # Rifting
        if p_bool('Rifting', 'Rift'):
            P.rift_SDR_packages()
            P.calc_rifting_anomaly()
            P.plot_rift_packages()

        # Get dips after rifting
        if p_bool('Calculation', 'DipsPostRift'):
            dips_l = dips_from_strat(P.sdr_l, P.top_dyke_l, "postrift_l_model",
                                     P.x_plate_l, rift_top=P.rift_top_l)
            dips_r = dips_from_strat(P.sdr_r, P.top_dyke_r, "postrift_r_model",
                                     P.x_plate_r, rift_top=P.rift_top_r)

        if p_bool('ObservedDips', 'ObsDips'):
            if p['ObservedGravity']['Orientation'] == 'E':
                modelled = dips_r
            elif p['ObservedGravity']['Orientation'] == 'W':
                modelled = dips_l

            obs_dips_dir = p['ObservedDips']['DirPath']
            compare_dips(obs_dips_dir, modelled)

        # Recalculate and Plot Process Oriented Gravity After Rifting
        if p_bool('Calculation', 'GravityPostRift'):
            postrift_anoms = P.calc_pogm_after_rift()
            postrift_anoms.plot_pogm_components("postrift_pogm") #, ylim_grav=[-75, ↩
    75], xlim=[100, 300])
            if p_bool('ObservedGravity', 'ObsGrav'):
                obs_grav_file = p['ObservedGravity']['FilePath']
```

```
                orient = p['ObservedGravity']['Orientation']
                postrift_anoms.compare_obs_grav(obs_grav_file, orient)

            # Replot rift packages with volcanic anomalies
            P.plot_rift_packages(pogm_obj=postrift_anoms,
                                 title="postrift_with_volcanic_anoms")

        # Calculate curvature
        if p_bool('Calculation', 'Curvature'):
            P.plot_curvature(P.strat[:, :, -1], P.top_dyke[:, :, -1])

    except Exception as e:
        raise e

    finally:
        # Move back to calling directory
        os.chdir(cur_dir)


if __name__ == '__main__':
    run_model(file=sys.argv[1])
```

```
from .data import constants as c
from .dips import dips_from_strat, compare_dips
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import gridspec
from copy import copy
import math
import csv

# Imports needed for running from config file
import configparser
import os
from shutil import copy2

# Imports needed for animation only
# TODO: imageio currently causing issues, so import off and animate wont work
# from moviepy.editor import ImageSequenceClip

# Imports needed for export only
try:
    import cPickle as pickle
except ImportError:
    import pickle

# Set matplotlib global settings for figures
plot_params = {
    # 'axes.titlesize': 10,
    # 'axes.labelsize': 10,
    'legend.loc': 'best',
    # 'legend.fontsize' : 8,
    'legend.fancybox': True,
    # 'legend.labelspacing' : 0.25,
    # 'ytick.labelsize' : 10,
    # 'xtick.labelsize' : 10,
    'pdf.fonttype': 42,
    'ps.fonttype': 42,
    'svg.fonttype': 'none',
    'axes.grid': True,
    'grid.linestyle': 'dotted',
    'font.sans-serif': 'Helvetica',
    'grid.linewidth': 0.5,       # in points
    'grid.alpha': 0.7,       # transparency, between 0.0 and 1.0
}

# Main run function


def run_model(file=None, config=None, save=True):
    """
    Calculate the response of the lithosphere to dyke intrusions

    Input:
    file = str, name of configuration file in this directory
    config = config variable (e.g. from web form)
    save = bool, whether to save output files
    """
    # Get current directory
    cur_dir = os.getcwd()

    try:
```

```
61          # Parse configuration file
62          if file is None:
63              p = config
64          else:
65              p = configparser.ConfigParser()
66              p.read(file)
67
68          # Get output directory
69          out_dir = p['Calculation']['OutputDir']
70
71          # If out_dir doesn't exist, make it
72          if not os.path.exists(out_dir):
73              os.makedirs(out_dir)
74
75          if save is True:
76              # Place a copy of config in output dir
77              copy2(file, out_dir)
78
79              # Change directory to out_dir
80              os.chdir(out_dir)
81
82          # Define some shortcuts
83          p_int = p.getint
84          p_bool = p.getboolean
85
86          # Calculate rifting
87          if p_bool('Rifting', 'Rift') is True:
88              R = Rift(p['Rifting']['XRiftOutput'], p['Rifting']['BetaFile'],
89                       p_int('Rifting', 'RiftWidthKm') * 1000)
90          else:
91              R = None
92
93          # Construct jump list
94          jump1 = (p_int('Jumps', 'JumpPos1'), p_int('Jumps', 'JumpSize1'))
95          jump2 = (p_int('Jumps', 'JumpPos2'), p_int('Jumps', 'JumpSize2'))
96          jump3 = (p_int('Jumps', 'JumpPos3'), p_int('Jumps', 'JumpSize3'))
97
98          if sum(jump1) == 0:
99              jumps = []
100
101         if sum(jump1) != 0 and sum(jump2) == 0:
102             jumps = [jump1]
103
104         if sum(jump1) != 0 and sum(jump2) != 0 and sum(jump3) == 0:
105             jumps = [jump1, jump2]
106
107         if sum(jump1) != 0 and sum(jump2) != 0 and sum(jump3) != 0:
108             jumps = [jump1, jump2, jump3]
109
110         # Extract dyke parameters
111         loads = p_int('Plate', 'Loads')
112
113         height = extract_dyke_param(p, 'Dykes', 'Height', loads)
114         rho_infill = extract_dyke_param(p, 'Dykes', 'RhoInfill', loads)
115         rho_load = extract_dyke_param(p, 'Dykes', 'RhoLoad', loads)
116         rho_comp = extract_dyke_param(p, 'Dykes', 'RhoComp', loads)
117         rho_ti = extract_dyke_param(p, 'Dykes', 'RhoTempInfill', loads)
118         te_max = extract_dyke_param(p, 'Dykes', 'MaxTe', loads)
119         te_min = extract_dyke_param(p, 'Dykes', 'MinTe', loads)
120         te_width = extract_dyke_param(p, 'Dykes', 'WidthTe', loads)
121
122         # Extract magnetic parameters
123         he = p_int('Magnetics', 'he')
124         anginc = p_int('Magnetics', 'anginc')
125         angstr = p_int('Magnetics', 'angstr')
126         suscept = p_int('Magnetics', 'suscept')
127
128         # Initiate plate
129         P = Plate(p_int('Dykes', 'Width'), loads, jumps, max(te_max),
130                   rift=R, bulge_wipe=p_bool('Plate', 'BulgeWipe'))
131
132         # Add dykes to plate
133         for i in range(P.loads):
134             # Instantiate new dyke
135             D = Dyke.add_dyke(P, height[i], rho_infill[i], rho_load[i],
136                               rho_comp[i], rho_ti[i], te_max[i], te_min[i],
137                               te_width[i], p_int('Plate', 'dx'))
138
139             # Add dyke to plate
140             P.add_dyke(D)
141             print("Dyke complete: ", i + 1)
142
143         # Build, plot and export stratigraphy
144         P.build_strat()
```

```
        P.plot_strat(P.strat, P.top_dyke, animate=p_bool('Calculation', 'Animate'), ↩
    save=save)
        if save is True:
            P.export_stratigraphy()

        # Gravity Process Oriented
        if p_bool('Calculation', 'GravityPreRift'):
            topped_strat = P.add_zero_top(P.strat)
            topped_load_flex = P.add_zero_top(P.load_flex)
            prerift_anoms = \
                P.calc_volc_POGM(topped_strat, topped_load_flex,
                                 P.top_dyke_lf, P.base_dyke_lf, c.zero_ct)
            prerift_anoms.plot_pogm_components("prerift_pogm", save=save)

        # Get dips
        if p_bool('Calculation', 'DipsPreRift'):
            dips_from_strat(P.strat[:, :, -1], P.top_dyke[:, :, -1],
                            "prerift_model", P.x_plate)

        # TODO Get magnetics before rifting
        if p_bool('Magnetics', 'Calculate'):
            print("Calculating magnetics prerift")

        # Rifting
        if p_bool('Rifting', 'Rift'):
            P.rift_SDR_packages()
            P.calc_rifting_anomaly()
            P.plot_rift_packages(save=save)

        # Get dips after rifting
        if p_bool('Calculation', 'DipsPostRift'):
            dips_l = dips_from_strat(P.sdr_l, P.top_dyke_l, "postrift_l_model",
                                     P.x_plate_l, rift_top=P.rift_top_l)
            dips_r = dips_from_strat(P.sdr_r, P.top_dyke_r, "postrift_r_model",
                                     P.x_plate_r, rift_top=P.rift_top_r)

        if p_bool('ObservedDips', 'ObsDips'):
            if p['ObservedGravity']['Orientation'] == 'E':
                modelled = dips_r
            elif p['ObservedGravity']['Orientation'] == 'W':
                modelled = dips_l

            obs_dips_dir = p['ObservedDips']['DirPath']
            compare_dips(obs_dips_dir, modelled)

        # TODO Get magnetics after rifting
        if p_bool('Magnetics', 'Calculate'):
            print("Calculating magnetics post-rift")

        # Recalculate and Plot Process Oriented Gravity After Rifting
        if p_bool('Calculation', 'GravityPostRift'):
            postrift_anoms = P.calc_pogm_after_rift()
            postrift_anoms.plot_pogm_components("postrift_pogm", save=save)
            if p_bool('ObservedGravity', 'ObsGrav'):
                obs_grav_file = p['ObservedGravity']['FilePath']
                orient = p['ObservedGravity']['Orientation']
                postrift_anoms.compare_obs_grav(obs_grav_file, orient)

            # Replot rift packages with volcanic anomalies
            P.plot_rift_packages(pogm_obj=postrift_anoms,
                                 title="postrift_with_volcanic_anoms", save=save)

        # Calculate curvature
        if p_bool('Calculation', 'Curvature'):
            P.plot_curvature(P.strat[:, :, -1], P.top_dyke[:, :, -1], save=save)

    except Exception as e:
        raise e

    finally:
        # Move back to calling directory
        os.chdir(cur_dir)


# General utility functions
def extract_dyke_param(parser, section, param, loads):
    """
    Get value from parser as list of floats or float

    Assumes list comma seperated with no braces

    Retruns val, either float or list of floats
    """
    str_val = parser[section][param]
```

```
228 |
229 |     if ',' in str_val:
230 |         val = str_val.split(',')
231 |         val = [float(i) for i in val]
232 |     else:
233 |         v = float(str_val)
234 |         val = [v for _ in range(loads)]
235 |
236 |     return val
237 |
238 |
239 | def rmse(predictions, targets):
240 |     """
241 |     Calculates Root Mean Square Error between two curves
242 |
243 |     Take two 2D numpy arrays, regularly spaced in x, [x, y] , predictions
244 |     and targets, and returns RMSE as scalar value. If arrays are mismatched in
245 |     size, it pads with NANs before calculating RMSE.
246 |     """
247 |
248 |     global_xmin = int(math.floor(min(min(predictions[:, 0]),
249 |                                      min(targets[:, 0]))))
250 |     global_xmax = int(math.ceil(max(max(predictions[:, 0]),
251 |                                     max(targets[:, 0]))))
252 |
253 |     p_xspace = predictions[1, 0] - predictions[0, 0]
254 |     t_xspace = targets[1, 0] - targets[0, 0]
255 |     x_sp = int(math.floor(min(p_xspace, t_xspace)))
256 |
257 |     x = range(global_xmin, global_xmax + x_sp, x_sp)
258 |
259 |     p_interp = np.interp(x, predictions[:, 0], predictions[:, 1], left=np.nan,
260 |                          right=np.nan)
261 |     t_interp = np.interp(x, targets[:, 0], targets[:, 1], left=np.nan,
262 |                          right=np.nan)
263 |
264 |     return np.sqrt(np.nanmean(((p_interp - t_interp) ** 2)))
265 |
266 |
267 | def find_first_nparray(item, array):
268 |     """
269 |     Find first occurrence of item in 1D numpy array, return index
270 |     """
271 |     for i in range(array.shape[0]):
272 |         if item == array[i]:
273 |             return i
274 |     return -1
275 |
276 |
277 | # Functions for gravity
278 | def surf2poly(base, top, ind):
279 |     """
280 |     Builds x or z vector around polygon for gbott
281 |
282 |     base = horizon at bottom
283 |     top = horizon at top
284 |     ind = indices to extract from those horizons, which satisfy polygon rules
285 |     """
286 |     coord = np.empty(2 * len(ind) + 1)
287 |     coord[:len(ind)] = base[ind]
288 |     coord[len(ind):-1] = np.flipud(top[ind])
289 |
290 |     # Wrap around to close polygon
291 |     coord[-1] = base[ind[0]]
292 |
293 |     return coord
294 |
295 |
296 | def surf2polyrho(rho_i, rho_b, rho_a, rho_sides, ind_len):
297 |     """
298 |     Builds rho vector around polygon for gbott
299 |
300 |     rho_i = density between horizons
301 |     rho_b = density below base horizon
302 |     rho_a = density above top horizon
303 |     ind_len = NUMBER of indices extracted from each horizon
304 |     """
305 |     coord = np.empty(2 * ind_len)
306 |     coord[:ind_len - 1] = (rho_i - rho_b) * np.ones(ind_len - 1)
307 |     coord[ind_len - 1] = (rho_i - rho_sides)
308 |     coord[ind_len:2 * ind_len - 1] = (rho_i - rho_a) * np.ones(ind_len - 1)
309 |     coord[2 * ind_len - 1] = (rho_i - rho_sides)
310 |
311 |     return coord
312 |
```

```
313 |
314 | def grv_gbott(x, z, rho, nsta, xstart, xstep, zp, p=0, v=0):
315 |     """
316 |     Calculate gravity for polygon with given density contrast
317 |
318 |     Function modified from gbott Matlab code and into Python by R. Morgan 2016
319 |
320 |     Chapter 5 - Isostasy and Flexure of the Lithosphere
321 |     A. B. Watts
322 |     Matlab file to calculate the gravity anomaly of an
323 |     arbitrary-shaped 2D body using the line integral method
324 |     of Talwani (1959) and Bott (1965).
325 |     The method is based on the formula for the gravity effect of
326 |     a horizontal sloping semi-infinite sheet of mass.
327 |     Set up body and assign each interface a density contrast
328 |     rho. The script can be modified to take into account a different
329 |     contrast for each x, z pair.
330 |
331 |     Note: The body x,z co-ordinates should be input in
332 |     an anti-clockwise direction. The density contrast between
333 |     the body and surroundings, rho, then refers to the material to the
334 |     left of the line connecting two successive pairs of co-ordinates.
335 |     Bodies can be closed (i.e. the first and last pairs of co-ordinates
336 |     are the same) or open. If open, extend the open end to great distance
337 |     to avoid end effects.
338 |
339 |     INPUTS:
340 |     x and z - body coordinates in km, with z +ve down, x and z same length,
341 |     in an anticlockwise direction, such that rho refers to left of line
342 |     rho - vector of density contrast across each line segment kg/m3, length
343 |     one less than x and z such that first contrast refers to length between
344 |     first 2 points, rho refers to left of line. Numpy arrays of same length.
345 |     p - if 1, plot outputs, if 0 don't (default)
346 |     nsta - Number of calculated gravity stations
347 |     xstart - First station position in km
348 |     xstep - Station interval in km
349 |     zp - Station height, -ve above the zero plane (km)
350 |     v - verbose mode if set to 1, default 0
351 |
352 |     OUTPUTS (in dictionary, indexed by variable name below, numpy arrays):
353 |     grv - output gravity profile in km
354 |     xs - station points in output gravity profile in km
355 |     """
356 |
357 |     # Universal gravitational constant Nm2/kg2
358 |     G = 6.67e-11
359 |
360 |     # Number of coordinate points defining body
361 |     npts = x.size
362 |
363 |     # Fix for z == 0 leaving body open TODO: still necessary?
364 |     for i in range(npts):
365 |         if z[i] == 0:
366 |             z[i] = 0.001
367 |
368 |     if v == 1:
369 |         print("Number of points = {}".format(npts))
370 |
371 |     # Convert to mGal (10^5) and m (10^3)
372 |     constant = 2 * G * 1.0e8
373 |     slab = 0
374 |
375 |     # Preallocate looped variables with NANs
376 |     grv = np.zeros(nsta) * np.nan
377 |     s = np.zeros(npts - 1) * np.nan
378 |     c = np.zeros(npts - 1) * np.nan
379 |
380 |     #  Set up field stations
381 |     zs = np.ones(nsta) * zp
382 |     xs = np.array([xstart + xstep * j for j in range(nsta)])
383 |
384 |     #  Set up geometry of each sloping interface
385 |     brho = -rho * constant
386 |
387 |     for i in range(npts - 1):
388 |         h = np.sqrt(np.square(x[i] - x[i + 1]) + np.square(z[i + 1] - z[i]))
389 |         if h == 0.0:
390 |             h = 0.001
391 |         s[i] = (z[i + 1] - z[i]) / h
392 |         c[i] = (x[i] - x[i + 1]) / h
393 |
394 |     #  Loop around all field stations
395 |     for j in range(nsta):
396 |         x1 = x[0] - xs[j]
397 |         z1 = z[0] - zs[j]
```

```
        # if z1 < 0: # previously commented out if block
        #     z1 = z1 - z[j]
        r1 = np.square(x1) + np.square(z1)
        fi1 = np.pi / 2 - np.arctan2(z1, x1)
        gtemp = 0

        #  Loop around all interfaces
        for i in range(npts - 1):
            x2 = x[i + 1] - xs[j]
            z2 = z[i + 1] - zs[j]
            # if z2 < 0:   # previously commented out if block
            #     z2 = z2 - z[j]
            r2 = np.square(x2) + np.square(z2)
            fi2 = np.pi / 2 - np.arctan2(z2, x2)

            #  Calculate gravity effect of each sloping interface
            if brho[i] != 0:
                W = 0.5 * s[i] * np.log(r1 / r2) + c[i] * (fi2 - fi1)
                gtemp = gtemp + brho[i] * \
                    (z2 * fi2 - z1 * fi1 - W * (x1 * s[i] + z1 * c[i])) + slab
            x1 = x2
            z1 = z2
            r1 = r2
            fi1 = fi2

        # Add effect of interface to total
        grv[j] = gtemp

    #  Display the gravity anomaly and body (if flagged)
    if p == 1:
        if v == 1:
            print("Plotting")

        # Calculate limits
        zllim = -(np.ceil(np.max(z)) + 10)
        zulim = -(np.floor(np.min(z)) - 10)

        # Plotting
        fig = plt.figure()
        ax1 = fig.add_subplot(211)
        ax1.plot(xs, grv, 'b-')
        ax1.set_xlim(np.min(xs), np.max(xs))
        ax1.set_title("Gravity Anomaly")
        ax1.set_ylabel("mGal")

        ax2 = fig.add_subplot(212)
        ax2.plot(x, -z, 'r-')
        ax2.set_xlim(np.min(xs), np.max(xs))
        ax2.set_ylim(zllim, zulim)
        ax2.set_title("Body")
        ax2.set_xlabel("Distance (km)")
        ax2.set_ylabel("Depth (km)")

        plt.tight_layout()
        plt.show()

    # Make dictionary to return
    grv_out = {"grv": grv, "xs": xs}

    # Return variables
    return grv_out


# Functions for magnetics
def mag_poly(xv, zv, he, nsta, xstart, xstep, zp, anginc, angstr, suscept,
             pl=False):
    """
    Computes the magnetic anomaly at one or more stations due to an infinite
    polygonal cylinder magnetized by the earth's magnetic field. The cylinder
    strikes parallel to the Y-axis, and has a polygonal cross-section in the
    X,Z plane. The anomalous magnetic field strength depends on X and Z, but
    not on Y.

    Note: the polygon is closed!

    Modified from Won and Bevis (1987) Fortran subroutine by R Morgan (2018).

    ARGUMENTS:
    nsta - Number of calculated gravity stations
    xstart - First station position in km
    xstep - Station interval in km
    zp - Station height, -ve above the zero plane (km)
    xv, zv     Vectors containing the coordinates of the polygon vertices. The
               vertices must be ordered in clockwise sequence as viewed
               looking towards the negative Y-axis.
```

```
    he          The Earth's total magnetic field strength
    anginc      The angle of inclination of the Earth's field (degrees)
    angstr      The strike of the cylinder (in degrees) measured clockwise
                (looking down) from magnetic north to the negative Y-axis
    suscept     The magnetic susceptibility of the cylinder in emu, or four pi
                times the magnetic susceptibility given in SI units
    p           If True, results are plotted, otherwise not (default False)

    OUTPUTS:
    anom_z      A vector containing the vertical magnetic anomaly at each
                station
    anom_x      A vector containing the horizontal magnetic anomaly at each
                station
    anom_t      A vector containing the total magnetic anomaly at each station

    If the routine is used to model one station per call (i.e. nstn = 1) then
    (undimensioned) variables can be used to pass the arguments xs, zs,
    anom_z, anom_x and anom_t.

    SIGN CONVENTION:
    A right-handed coordinate system is assumed. Z values are positive
    downwards. The X- and Z- components of the anomalous field strength are
    positive if these component fields point in the positive X and Z
    directions respectively, and negative otherwise.

    UNITS:
    The component and total anomalies will be returned in the same units used
    to specify the Earth's total field strength, and thus the user is free to
    choose any units deemed convenient. Note that the way in which the
    argument 'suscept' is interpreted does depend in the system of units
    being employed. Spatial coordinates may be given in any length unit, as
    long as the unit is employed consistently throughout, although I have
    assumed km here in defining the station points.
    """

    # Define parameters
    dtr = np.pi / 180
    c1 = np.sin(anginc * dtr)
    c2 = np.sin(angstr * dtr) * np.cos(anginc * dtr)
    c3 = 2 * suscept * he

    #  Set up field stations in the same way as gbott
    zs = np.ones(nsta) * zp
    xs = np.array([xstart + xstep * j for j in range(nsta)])
    nstn = xs.size

    # Preallocate output vectors
    anom_z = np.zeros(nstn) * np.nan
    anom_x = np.zeros(nstn) * np.nan
    anom_t = np.zeros(nstn) * np.nan

    # Check for overlapping points - unlike gbott, fails if start/end points
    # are identical
    if xv[0] == xv[-1] and zv[0] == zv[-1]:
        xv = xv[:-1]
        zv = zv[:-1]
    nvert = xv.size

    # Loop over the stations
    for i in range(nstn):
        xst = xs[i]
        zst = zs[i]

        hz = 0
        hx = 0

        # Loop over vertices
        for ic in range(nvert):

            # Translate origin
            x1 = xv[ic] - xst
            z1 = zv[ic] - zst

            if ic == nvert - 1:
                x2 = xv[0] - xst
                z2 = zv[0] - zst
            else:
                x2 = xv[ic + 1] - xst
                z2 = zv[ic + 1] - zst

            if x1 == 0 and z1 == 0:   # gz = 0
                break   # terminate this pass through vertex loop
            else:
                th1 = np.arctan2(z1, x1)
```

```
            if x2 == 0 and z2 == 0:   # gz = 0
                break   # terminate this pass through vertex loop
            else:
                th2 = np.arctan2(z2, x2)

            if np.sign(z1) != np.sign(z2):
                test = x1 * z2 - x2 * z1
                if test > 0:
                    if z1 >= 0:
                        th2 = th2 + 2 * np.pi
                elif test < 0:
                    if z2 >= 0:
                        th1 = th1 + 2 * np.pi
                else:    # station on polygon side: gz = 0
                    break   # teminate this pass through vertex loop

            t12 = th1 - th2
            z21 = z2 - z1
            x21 = x2 - x1
            x21s = x21 * x21
            z21s = z21 * z21
            xz12 = x1 * z2 - x2 * z1
            r1s = x1 * x1 + z1 * z1
            r2s = x2 * x2 + z2 * z2
            r21s = x21 * x21 + z21 * z21
            rln = 0.5 * np.log(r2s / r1s)
            p = (xz12 / r21s) * ((x1 * x21 - z1 * z21) /
                                 r1s - (x2 * x21 - z2 * z21) / r2s)
            q = (xz12 / r21s) * ((x1 * z21 + z1 * x21) /
                                 r1s - (x2 * z21 + z2 * x21) / r2s)

            if x21 == 0:
                dzz = -p
                dzx = q - z21s * rln / r21s
                dxz = q
                dxx = p + z21s * t12 / r21s
            else:
                fz = (t12 + (z21 / x21) * rln) / r21s
                fx = (t12 * (z21 / x21) - rln) / r21s
                dzz = - p + x21s * fz
                dzx = q - (x21 * z21) * fz
                dxz = q - x21s * fx
                dxx = p + (x21 * z21) * fx

            hz = c3 * (c1 * dzz + c2 * dzx) + hz
            hx = c3 * (c1 * dxz + c2 * dxx) + hx

        anom_z[i] = hz
        anom_x[i] = hx
        anom_t[i] = c1 * hz + c2 * hx

    #  Display the magnetic anomaly and body (if flagged)
    if pl is True:
        # Calculate limits
        zllim = -(np.ceil(np.max(zv)) + 10)
        zulim = -(np.floor(np.min(zv)) - 10)

        yllim = np.floor(np.min(anom_t)) - 10
        yulim = np.ceil(np.max(anom_t)) + 10

        # Plotting
        print("Plotting Magnetics")

        fig = plt.figure()
        ax1 = fig.add_subplot(211)
        ax1.plot(xs, anom_t, 'b-')
        ax1.set_xlim(np.min(xs), np.max(xs))
        ax1.set_ylim(yllim, yulim)
        ax1.set_title("Magnetic Anomaly")
        ax1.set_ylabel("nT")

        ax2 = fig.add_subplot(212)
        ax2.fill(xv, -zv, 'r')
        ax2.set_ylim(zllim, zulim)
        ax2.invert_yaxis()
        ax2.set_xlim(np.min(xs), np.max(xs))
        ax2.set_title('Body')
        ax2.set_xlabel('Distance (km)')
        ax2.set_ylabel('Depth (km)')

        plt.tight_layout()
        plt.show()

    return [anom_z, anom_x, anom_t]
```

```
653
654 # Functions for flexure
655 def flex_2d(xload=0, zload=0, x_te=0, z_te=0, prof_length=0, dx=0,
656             rho_load=0, rho_displace=0, rho_infill=0,
657             rho_compensation=0, p=False):
658     """
659     Calculates flexure along a 2D profile of a continuous elastic plate.
660
661     Based on Contreras-Reyes (2010,GJI). Modified by J. Hunter to
662     include a continuous plate rather than a broken plate boundary
663     condition. Elastic thickness (Te) can vary along the profile.
664     Modified from Matlab script to Python function by R. Morgan. Assumes
665     horizontal force F is zero everywhere, V0 and M0 are zero.
666
667     Inputs
668     xload,zload = n length lists, x and z coordinates of load (m)
669     xTe, zTe    = n length lists, x and z values of Te (m)
670     prof_length  = scalar, profile length (m)
671     dx           = scalar, distance step (m)
672     rho_load, rho_displace, rho_compensation = densities, scalar, kg/m3
673     p            = plot if True
674     rho_infill = density kg/m3, vector, sampled at same x positions as load
675
676     Outputs: dict containing numpy.ndarrays, same length and pos as xload
677     flex = flexed plate along profile in m
678     load_out =  load along profile in m
679     moments = bending moments along profile
680     shear = shear force along profile
681     """
682
683     # Create x-array
684     x = np.arange(xload[0], prof_length + 1, dx)
685
686     # Calculate number of points
687     # npts = int(np.ceil(prof_length/dx))
688     npts = x.shape[0]
689
690     # Interpolate load to load array
691     Q1 = np.interp(x, xload, zload)
692
693     # Calculate weight of the load
694     Q = Q1 * (rho_load - rho_displace) * c.g
695
696     # Interpolate Te array
697     Ted = np.interp(x, x_te, z_te)
698
699     # Interpolate rho_infill array, use x positions of load
700     rho_infill_samp = np.interp(x, xload, rho_infill)
701
702     # Correct for problem with first entry zero in interpolation
703     if np.isnan(Ted[0]):
704         Ted[0] = Ted[1]
705
706     # Preallocate
707     qx = np.zeros((4 * (npts - 2), 1))
708
709     for i in range(2, npts):
710         qx[3 + (i - 2) * 4] = Q[i]
711
712     qx = np.transpose(qx)
713
714     # Convert Te array to flexural rigidity array
715
716     # Initialize array
717     D = np.empty(Ted.shape[0])
718
719     for i in range(npts):
720         D[i] = (c.E * (Ted[i] ** 3)) / (12 * (1 - c.nu ** 2))
721
722     # Forming matrices
723     # Defining A_one and A_N
724     A_N = np.array([[0, 0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0, 0],
725                     [0, -1, 0, 0, 0, 0, dx * (1 / D[npts - 1]), 0],
726                     [0, 0, -1, 0, 0, 0, 1, -dx]])
727     A_one = np.array([[-1, -dx, 0, 0, 1, 0, 0, 0], [0, -1, 0, 0, 0, 1, 0, 0],
728                       [0, 0, 1, 0, 0, 0, 0, 0],
729                       [0, 0, 0, 1, 0, 0, 0, 0]])
730
731     # Defining B_one and B_N, assuming M0 and V0 zero
732     B_N = np.array([[0], [0], [0], [0]])
733     B_one = np.array([[0], [0], [0], [0]])
734
735     # Forming matrix array P
736     P = np.zeros((4, 4, npts))
737
```

```
    for n in range(npts):
        P[:, :, n] = [[0, -1, 0, 0], [0, 0, (1 / D[n]), 0], [0, 0, 0, -1],
                      [-(rho_compensation - rho_infill_samp[n]) *
                      c.g, 0, 0, 0]]

    # Identity matrix
    Ident = np.eye(4)

    # Forming Mat_A
    Y = 4 * npts

    Mat_A = np.zeros((Y, Y))
    Mat_A[0:4, 0:8] = A_one
    Mat_A[Y - 4:Y, Y - 8:Y] = A_N

    for i in range(2, npts):
        n = 5 + (i - 2) * 4
        Mat_A[n - 1:n + 3, n - 1:n + 3] = 2 * dx * P[:, :, i]
        Mat_A[n - 1:n + 3, n - 5:n - 1] = -Ident
        Mat_A[n - 1:n + 3, n + 3:n + 7] = Ident

    # Forming Mat_B
    Mat_B = np.zeros((Y, 1))
    Mat_B[0:4] = B_one
    Mat_B[Y - 4:Y] = B_N
    Mat_B[4:Y - 4] = np.transpose(2 * dx * qx)

    # Solving for Mat_S
    Mat_S = np.linalg.solve(Mat_A, Mat_B)

    # print("Shape: " + str(Mat_A.shape))
    # print("Rank: " + str(np.linalg.matrix_rank(Mat_A)))
    # print("Norm: " + str(np.linalg.norm(np.dot(Mat_A,Mat_S) - Mat_B)))

    # Initialize arrays
    y_fd1 = np.empty(npts)
    mom1 = np.empty(npts)
    V1 = np.empty(npts)

    # Pulling values out of Mat_S
    for i in range(1, npts + 1):
        # Flexure profile
        n = 1 + (i - 1) * 4
        y_fd1[i - 1] = Mat_S[n - 1]

        # Bending moment profile
        k = 3 + (i - 1) * 4
        mom1[i - 1] = Mat_S[k - 1]

        # Shear force profile
        j = 4 + (i - 1) * 4
        V1[i - 1] = Mat_S[j - 1]

    # Re-sample outputs onto same vector as load profile input
    flex = np.interp(xload, x, y_fd1, left=np.nan, right=np.nan)
    load_out = np.interp(xload, x, Q1, left=np.nan, right=np.nan)
    moments = np.interp(xload, x, mom1, left=np.nan, right=np.nan)
    shear = np.interp(xload, x, V1, left=np.nan, right=np.nan)
    rigidity = np.interp(xload, x, D, left=np.nan, right=np.nan)

    # Find curvature
    curv = - moments / rigidity

    # Make dictionary to return
    out = {"flex": flex, "load_out": load_out,
           "moments": moments, "shear": shear, "rigidity": rigidity,
           "curv": curv}

    # Simple test plotting
    if p is True:
        x_plot = [i for i in range(flex.shape[0]) if flex[i] < -1]
        x_plot = range(x_plot[0] - 10, x_plot[-1] + 10)
        print("Plotting flexure")

        plt.figure()
        plt.subplot(3, 1, 1)
        plt.plot(xload[x_plot] / 1000, flex[x_plot], "bo-")
        plt.ylabel("Flexure (m)")
        plt.xlabel("Distance (km)")
        plt.title("Flexure")

        plt.subplot(3, 1, 2)
        plt.plot(x_te[x_plot] / 1000, z_te[x_plot], "ro-")
        plt.ylabel("Te (m)")
```

```
822         plt.xlabel("Distance (km)")
823         plt.title("Te Profile")
824 
825         plt.subplot(3, 1, 3)
826         plt.plot(xload[x_plot] / 1000, load_out[x_plot], "r--")
827         plt.ylabel("Load Height (m)")
828         plt.xlabel("Distance (km)")
829         plt.title("Load")
830 
831         plt.tight_layout()
832         plt.show()
833 
834     return out
835 
836 
837 def calc_curvature(x, y):
838     """
839     Calculates the curvature of a thin elastic plate
840 
841     Assumes twice differentiable, same units used in x and y
842     See McNutt and Menard 1982 eq (4)
843 
844     Inputs:
845     x, y = numpy array
846 
847     Returns:
848     C = curvature at corresponding points, numpy array, same length as
849     input x and y
850     """
851 
852     dy = np.gradient(y, x)
853     ddy = np.gradient(dy, x)
854 
855     C = np.absolute(ddy)
856 
857     return C
858 
859 
860 # Main classes
861 class Plate(object):
862     def __init__(self, dyke_w, loads, jumps, te_max, rift=None,
863                  bulge_wipe=True):
864         # Allocate variables
865         self.bulge_wipe = bulge_wipe
866         self.rift = rift  # class
867         self.dyke_width = dyke_w  # m
868         self.loads = loads
869         self.te_max = te_max
870 
871         # Define jumps - input list of tuples, with first entry the load
872         # number after which the jump will take place and the second entry the
873         # magnitude of the jump in metres
874         self.jumps = jumps
875 
876         # Force x resolution to be 1/4 of dyke width for ease of indexing;
877         # can be 1/any even integer
878         self.x_step = dyke_w / 4  # m
879 
880         # Set flexural properties and plate size
881         self._flex_setup()
882 
883         # Set emplacement point
884         self.emp = int(0.5 * self.npts)
885 
886         # Declare lists to store flexures from dykes
887         self.flexes_infill = list()
888         self.flexes_load = list()
889         self.morphos = list()
890 
891         # Declare lists to store densities
892         self.infill_ds = list()
893         self.load_ds = list()
894         self.temp_infill_ds = list()
895 
896         # Track dyke heights (to track base)
897         self.dyke_heights = list()
898 
899     def _flex_setup(self):
900         """
901         Calculates flexural parameters for all objects on this plate
902         """
903 
904         # Calculate flexural parameter
905         D = (c.E * self.te_max ** 3) / (12 * (1 - c.nu ** 2))
906         alpha = np.power((4 * D) / (c.g * (c.rho_mantle - c.rho_water)), 1 / 4)
```

```
907 |
908 |         # Calculate flexural wavelength in terms of grid points
909 |         lambda_pts = np.ceil((2 * np.pi * alpha) / self.x_step)
910 |
911 |         # Define plate side length in m, allowing for sufficient space for
912 |         # FFT, or rifting, whatever is greater (need width for rifting
913 |         # anomaly to go to zero at the edges
914 |         # TODO: Remove rift_width input, calculate it from beta_profile
915 |         if self.rift is not None:
916 |             if self.rift.rift_width * 6 < (np.ceil(6 * np.pi * alpha) +
917 |                                             np.ceil(self.loads *
918 |                                                     self.dyke_width)):
919 |                 plate_size = np.ceil(self.loads * self.dyke_width) + \
920 |                     np.ceil(6 * np.pi * alpha)
921 |             else:
922 |                 plate_size = self.rift.rift_width * 6
923 |         else:
924 |             plate_size = np.ceil(self.loads * self.dyke_width) + \
925 |                 np.ceil(6 * np.pi * alpha)
926 |
927 |         # Force npts to be even TODO: Would like to remove this
928 |         if plate_size / self.x_step % 2 == 1:
929 |             plate_size += 1
930 |
931 |         # Set up plate index vectors
932 |         x_plate = np.arange(1, plate_size + 1, self.x_step)  # m
933 |         npts = x_plate.shape[0]
934 |         prof_length = np.max(x_plate)
935 |
936 |         # Set attributes of plate
937 |         self.npts = npts
938 |         self.x_plate = x_plate
939 |         self.lambda_pts = lambda_pts
940 |         self.prof_length = prof_length
941 |
942 |     def add_dyke(self, dyke):
943 |         """
944 |         Keep track of dyke variables as they are added to plate
945 |         """
946 |
947 |         # Track flexures
948 |         self.flexes_infill.append(dyke.infill_flex)
949 |         self.flexes_load.append(dyke.load_flex)
950 |         self.morphos.append(dyke.morpho)
951 |
952 |         # Track densities
953 |         self.infill_ds.append(dyke.rho_infill)
954 |         self.load_ds.append(dyke.rho_load)
955 |         self.temp_infill_ds.append(dyke.rho_ti)
956 |
957 |         # Track dyke heights (to track base)
958 |         self.dyke_heights.append(dyke.height)
959 |
960 |     def find_dyke_base(self, top_dyke):
961 |         """
962 |         Given the positions of the top of the dykes and heights, find base
963 |
964 |         Inputs:
965 |         top_dyke = stratigraphy tracking variable, 2D or 3D
966 |         """
967 |         L = self.loads
968 |         array_fill = np.ones(top_dyke.shape[0])
969 |
970 |         dimension = len(top_dyke.shape)
971 |
972 |         if dimension == 3:
973 |             base_dyke = np.empty([top_dyke.shape[0], L, L])
974 |
975 |             # Restart loop generator
976 |             strat_loop = ((l, t) for l in range(L) for t in range(L))
977 |
978 |             for l, t in strat_loop:
979 |                 base_dyke[:, l, t] = np.add(top_dyke[:, l, t], -
980 |                                             self.dyke_heights[l] * array_fill)
981 |
982 |             return base_dyke
983 |
984 |         elif dimension == 2:
985 |
986 |             base_dyke = np.empty([top_dyke.shape[0], L])
987 |
988 |             for l in range(L):
989 |                 base_dyke[:, l] = np.add(top_dyke[:, l], -
990 |                                          self.dyke_heights[l] * array_fill)
991 |
```

```
            return base_dyke

    def build_strat(self):
        """
        Build resulting stratigraphy for given dykes and jumps
        """

        def stitch_strat(sh_strat, st, loads):
            """
            Stitch together shifted half #loads to form symmetric package
            """
            # Preallocate
            strat = np.zeros([self.npts, loads, loads]) * np.nan
            strat[:st, :, :] = sh_strat[:st, :, :]

            # -1 offset as middle point should be reflected and therefore
            # the same, not artificially duplicated
            strat[st - 1:2 * st - 1, :, :] = np.flipud(sh_strat[:st, :, :])

            # As plate >> than the load anyway, fill the final point with the
            # same value as penultimate (zero before flipping)
            strat[-1, :, :] = strat[0, :, :]

            return strat

        def build_sym_package(load_range):
            """
            For loads between jumps, build symmetric package

            Takes list load_range of the load indexes to work on
            """
            # Should be 2, still calculated in case I no longer force x_step
            x_shift = int(np.ceil(0.5 * self.dyke_width / self.x_step))
            L = len(load_range)

            # Preallocate stratigraphy tracking variables
            a = np.zeros([self.npts, L, L])
            sh_strat = copy(a)
            sh_morpho = copy(a)
            sh_load_flex = copy(a)
            sh_inf_flex = copy(a)
            sum_sh_load_flex = copy(a)
            sum_sh_inf_flex = copy(a)

            # Preallocate top dyke with NAN's - one for overall stratigraphy,
            # one for load flex only for gravity calculation
            top_dyke = copy(a) * np.nan
            top_dyke_lf = copy(a) * np.nan

            # Set up loop generator
            strat_loop = ((l, t) for l in range(L) for t in range(L))

            # Loop through loads and build stratigraphy
            for l, t in strat_loop:
                # Calculate size of shift for each load, at each time
                sh = (t - l) * x_shift
                if l <= t:
                    # Already exists, modify by subsequent flexure
                    # Shifted, will be summed in subsequent loop
                    for k in range(self.emp - sh):
                        sh_morpho[k, l, t] = self.morphos[
                            load_range[l]][k + sh]
                        sh_load_flex[k, l, t] = self.flexes_load[
                            load_range[l]][k + sh]
                        sh_inf_flex[k, l, t] = self.flexes_infill[
                            load_range[l]][k + sh]

            # Sum final stratigraphy, looping through loads (l) and time (t)
            # Restart loop generator
            strat_loop = ((l, t) for l in range(L) for t in range(L))
            for l, t in strat_loop:
                if l + 1 == t:
                    for k in range(self.emp - sh):
                        sum_sh_load_flex[k, l, t] = sh_load_flex[k, l + 1, t]
                        sum_sh_inf_flex[k, l, t] = sh_inf_flex[k, l + 1, t]
                elif l + 1 < t:
                    for k in range(self.emp - sh):
                        sum_sh_load_flex[k, l, t] = np.sum(
                            sh_load_flex[k, l + 1:t + 1, t])
                        sum_sh_inf_flex[k, l, t] = np.sum(
                            sh_inf_flex[k, l + 1:t + 1, t])

            # Restart loop generator
            strat_loop = ((l, t) for l in range(L) for t in range(L))
```

```
            # Set dyke half width
            half_w = int((self.dyke_width / 2) / self.x_step)

            # sh_morpho includes all flexure associated with load in question;
            # need to add on subsequent flexure which is split up into
            # load/infill. Repeat for sh_infill_flex and sh_load_flex.
            for l, t in strat_loop:
                if l <= t:
                    for k in range(self.emp - sh):
                        sh_strat[k, l, t] = sh_morpho[k, l, t] + \
                            sum_sh_load_flex[k, l, t] + \
                            sum_sh_inf_flex[k, l, t]
                        sh_inf_flex[k, l, t] = sh_inf_flex[k, l, t] + \
                            sum_sh_inf_flex[k, l, t]
                        sh_load_flex[k, l, t] = sh_load_flex[k, l, t] + \
                            sum_sh_load_flex[k, l, t]

                    # Find top_dyke position, fill with half width of dyke
                    end = self.emp - ((t - l) * x_shift)
                    top_dyke[end - half_w:end, l, t] = \
                        sh_strat[end - half_w:end, l, t]
                    top_dyke_lf[end - half_w:end, l, t] = \
                        sh_load_flex[end - half_w:end, l, t]

            # Wipe bulges (if required, minimum required, set to 35km)
            if self.bulge_wipe is True:
                sh_strat = np.clip(sh_strat, -c.zero_ct, 0)
                sh_inf_flex = np.clip(sh_inf_flex, -c.zero_ct, 0)
                sh_load_flex = np.clip(sh_load_flex, -c.zero_ct, 0)

            # Stitch point is right-most written value, given by emp
            strat_out = stitch_strat(sh_strat, self.emp, L)
            inf_flex_out = stitch_strat(sh_inf_flex, self.emp, L)
            load_flex_out = stitch_strat(sh_load_flex, self.emp, L)
            top_dyke = stitch_strat(top_dyke, self.emp, L)
            top_dyke_lf = stitch_strat(top_dyke_lf, self.emp, L)

            return strat_out, inf_flex_out, load_flex_out, \
                top_dyke, top_dyke_lf

        def dyke_jump(s_pack, i_pack, l_pack, t_d_pack, t_d_lf_pack,
                      j_arr, j_pack):
            """
            Build overall stratigraphy by jumping symmetric packages
            """

            def _fill_strat(sz, pack, f=None):
                '''
                Fill strat with contents of sub packages only

                sz = list of number of loads in each package
                pack = list containing arrays of strat of each package
                f = preallocate with this, default zeros
                '''

                # Preallocate
                if f is None:
                    strat = np.zeros([self.npts, self.loads, self.loads])
                else:
                    strat = np.zeros([self.npts, self.loads, self.loads]) * f

                # Sum together stratigraphy
                st = 0
                for i in range(len(sz)):
                    # Add in new sub-package for load range
                    strat[:, st:st + sz[i], st:st + sz[i]] = pack[i]

                    # Also add final configuration of load to all subsequent
                    # times
                    for j in range(st + sz[i], self.loads):
                        strat[:, st:st + sz[i], j] = pack[i][:, :, -1]

                    # Iterate start point
                    st += sz[i]

                return strat

            def _shift_sum_packs(strat, sz, pack, j_sz, sm=True):
                '''
                Perform sum operations with time on jumped packages AND shift

                strat = existing stratigraphy to work on
                sz = list of number of loads in each package
                pack = list containing arrays of strat of each package
```

```
                j_sz = list of length equal to the number of jumps, with each
                       entry the total sum dist of jumps
                sm = bool, if True also sum horizons
                '''

                # Convert j_sz entries to ints for use in indexing
                j_sz = [int(i) for i in j_sz]

                # Define shift
                x_shift = int(np.ceil(0.5 * self.dyke_width / self.x_step))

                for i in range(1, len(sz)):
                    # For subsequent packages, add the deepest horizon of the
                    # current time of new package to all horizons beneath
                    for l in range(sum(sz[:i])):
                        # Loop over times after deposition of new sub package
                        # Add relevant horizon for those times during subpack
                        for t in range(sum(sz[:i]), sum(sz[:i + 1])):
                            # If it exists:
                            if l <= t:
                                # Define shift size
                                sh = (t - sum(sz[:i]) + 1) * x_shift

                                for k in range(self.emp + j_sz[i - 1] - sh):
                                    strat[k, l, t] = strat[k + sh, l, t]

                                for k in range(self.npts - 1, self.emp +
                                               j_sz[i - 1] + sh, -1):
                                    strat[k, l, t] = strat[k - sh, l, t]

                                # 'nan' over where now filled by dyke
                                dyke_l = self.emp + j_sz[i - 1] - sh
                                dyke_r = self.emp + j_sz[i - 1] + sh + 1
                                strat[dyke_l:dyke_r, l, t] = np.nan

                                # Shift down
                                if sm is True:
                                    strat[:, l, t] += \
                                        pack[i][:, 0, t - sum(sz[:i])]

                    if i != len(sz) - 1:
                        # Add end case for all times AFTER this sub package
                        for t in range(sum(sz[:i + 1]), sum(sz)):
                            # If it exists:
                            if l <= t:
                                # Define shift size (maximum)
                                sh = sz[i] * x_shift

                                for k in range(self.emp +
                                               j_sz[i - 1] - sh):
                                    strat[k, l, t] = strat[k + sh, l, t]

                                for k in range(self.npts - 1,
                                               self.emp +
                                               j_sz[i - 1] + sh, -1):
                                    strat[k, l, t] = strat[k - sh, l, t]

                                # 'nan' over where now filled by dyke
                                dyke_l = self.emp + j_sz[i - 1] - sh
                                dyke_r = self.emp + j_sz[i - 1] + sh + 1
                                strat[dyke_l:dyke_r, l, t] = np.nan

                                # Shift down
                                if sm is True:
                                    strat[:, l, t] += pack[i][:, 0, -1]

                return strat

            # Scale jump size by x_step
            j_arr = j_arr / self.x_step

            # Calculate sizes of jumps - j_sz is a list of length equal to
            # the number of jumps, with each entry the total sum dist of jumps
            # completed so far. List length = #packages - 1
            j_sz = list()
            for i in range(len(j_arr)):
                if j_arr[i] != 0:
                    j_sz.append(np.sum(j_arr[:i + 1]))

            # Jump each package by sum of all jumps up to the time it was
            # deposited
            def j(pack, jump):
                return np.roll(pack, int(jump))

            for i in range(len(j_sz)):
```

```
                # Jump package after jump (i+1) by given amount
                for l in range(s_pack[i + 1].shape[1]):
                    for t in range(s_pack[i + 1].shape[1]):
                        # If it exists:
                        if l <= t:
                            s_pack[i + 1][:, l, t] = \
                                j(s_pack[i + 1][:, l, t], j_sz[i])
                            i_pack[i + 1][:, l, t] = \
                                j(i_pack[i + 1][:, l, t], j_sz[i])
                            l_pack[i + 1][:, l, t] = \
                                j(l_pack[i + 1][:, l, t], j_sz[i])
                            t_d_pack[i + 1][:, l, t] = \
                                j(t_d_pack[i + 1][:, l, t], j_sz[i])
                            t_d_lf_pack[i + 1][:, l, t] = \
                                j(t_d_lf_pack[i + 1][:, l, t], j_sz[i])

            # Extract number of loads in each sub package to be used in loop
            # Length of sz = number of packages
            sz = list()
            for i in range(len(j_pack)):
                sz.append(len(j_pack[i]))

            # Fill new 3D arrays with contents of sub packages
            strat = _fill_strat(sz, s_pack)
            infill_flex = _fill_strat(sz, i_pack)
            load_flex = _fill_strat(sz, l_pack)
            top_dyke = _fill_strat(sz, t_d_pack, f=np.nan)
            top_dyke_lf = _fill_strat(sz, t_d_lf_pack, f=np.nan)

            # Shift stratigraphy by extension due to dyke intrusion
            top_dyke = _shift_sum_packs(top_dyke, sz, l_pack, j_sz, sm=False)
            top_dyke_lf = _shift_sum_packs(top_dyke_lf, sz, l_pack, j_sz,
                                           sm=False)

            # Sum AND shift together jumped packages through time
            strat = _shift_sum_packs(strat, sz, s_pack, j_sz)
            infill_flex = _shift_sum_packs(infill_flex, sz, i_pack, j_sz)
            load_flex = _shift_sum_packs(load_flex, sz, l_pack, j_sz)

            # Place values in top_dyke based on non-nan values in top_dyke
            # and taking the jumped and summed values from relevant
            # stratigraphy
            for i in range(self.npts):
                for l in range(self.loads):
                    for t in range(self.loads):
                        if not np.isnan(top_dyke[i, l, t]):
                            top_dyke[i, l, t] = strat[i, l, t]
                        if not np.isnan(top_dyke_lf[i, l, t]):
                            top_dyke_lf[i, l, t] = load_flex[i, l, t]

            # Return final arrays
            return strat, infill_flex, load_flex, top_dyke, top_dyke_lf

        def split_point(strat, top_dyke):
            """
            Find split point of strat at position of final dyke intrusion
            using top_dyke
            """

            inds = list()
            for j in range(strat.shape[0]):
                if not np.isnan(top_dyke[j, -1, -1]):
                    inds.append(j)

            # Split point is mid of returned indices marking top of final dyke
            sp = inds[int(len(inds) / 2 - 0.5)]

            return sp

        # If no jumps, save vars to plate and return at this point
        if not self.jumps:
            (strat, infill_flex, load_flex, top_dyke, top_dyke_lf) = \
                build_sym_package(np.arange(self.loads))

            base_dyke = self.find_dyke_base(top_dyke)
            base_dyke_lf = self.find_dyke_base(top_dyke_lf)

            # Save values to plate
            self.strat = strat
            self.infill_flex = infill_flex
            self.load_flex = load_flex
            self.top_dyke = top_dyke
            self.top_dyke_lf = top_dyke_lf
            self.base_dyke = base_dyke
            self.base_dyke_lf = base_dyke_lf
```

```
1331            self.split_point = split_point(strat, top_dyke)
1332
1333            return
1334
1335        # Identify symmetric packages between jumps and build
1336
1337        # Build jump array from tuple input - this is a numpy array of length
1338        # self.loads which contains the
1339        # magnitude of the jump at the index of the load it happens after, and
1340        # if no jump, zeros
1341        j_array = np.zeros(self.loads)
1342
1343        # j_i is used to build j_pack, below
1344        j_i = np.arange(self.loads)
1345
1346        # j_pack is a list of numpy arrays, each containing the range of
1347        # indices of the loads in each sub package
1348        j_pack = list()
1349
1350        for i in range(len(self.jumps)):
1351            j_array[self.jumps[i][0] - 1] = self.jumps[i][1]
1352
1353        # Split at jumps into subsets of symmetric packages
1354        # Initialize counter at first load
1355        k = 0
1356        for i in range(len(j_array)):
1357            # If jump happens:
1358            if j_array[i] != 0:
1359                # Split indices for package
1360                j_pack.append(j_i[k:i + 1])
1361                # Set counter to next load after jump
1362                k = i + 1
1363
1364        # Append everything after final jump
1365        j_pack.append(j_i[k:])
1366
1367        # Initialize lists to store package stratigraphy
1368        s_pack = list()
1369        i_pack = list()
1370        l_pack = list()
1371        t_d_pack = list()
1372        t_d_lf_pack = list()
1373
1374        # Store symmetric packages in a list
1375        for i in range(len(j_pack)):
1376            (s, i_f, l_f, t_d, t_d_lf) = build_sym_package(j_pack[i])
1377            s_pack.append(s)
1378            i_pack.append(i_f)
1379            l_pack.append(l_f)
1380            t_d_pack.append(t_d)
1381            t_d_lf_pack.append(t_d_lf)
1382
1383        # Jump around
1384        (strat, infill_flex, load_flex, top_dyke, top_dyke_lf) = \
1385            dyke_jump(s_pack, i_pack, l_pack, t_d_pack, t_d_lf_pack, j_array,
1386                      j_pack)
1387
1388        base_dyke = self.find_dyke_base(top_dyke)
1389        base_dyke_lf = self.find_dyke_base(top_dyke_lf)
1390
1391        # Save values to plate and return
1392        self.strat = strat
1393        self.infill_flex = infill_flex
1394        self.load_flex = load_flex
1395        self.top_dyke = top_dyke
1396        self.top_dyke_lf = top_dyke_lf
1397        self.base_dyke = base_dyke
1398        self.base_dyke_lf = base_dyke_lf
1399        self.split_point = split_point(strat, top_dyke)
1400
1401        return
1402
1403    def plot_timeslice(self, strat_plot, dyke_plot, limits, save=True,
1404                       title=None, animate=False, index=None):
1405        '''
1406        Plot stratigraphy at a given time
1407
1408        limits = tuple of 3 values: deepest (minimum y), x_range, x_min
1409        strat_plot and dyke_plot: 2D slices, 3rd index is relevant time-slice
1410        index = index used in 3rd dimension of strat/dyke, used for plot title
1411        animate = True if part of animation sequence, otherwise False
1412        save = True to output file
1413        '''
1414
1415        # Update plotting parameters
```

```
        plt.rcParams.update(plot_params)

        # Extract limits
        deepest = limits[0]
        x_range = limits[1]
        x_min = limits[2]

        # Save these values for future plots
        self.x_min = x_min
        self.x_range = x_range

        # Start new figure
        fig = plt.figure()
        ax = fig.gca()

        # Plot every load i at given time
        for i in range(self.loads):
            plt.plot(self.x_plate / 1000 - x_min, strat_plot[:, i] / 1000,
                     lw=0.5, color='black')
            if dyke_plot is not None:
                plt.plot(self.x_plate / 1000 - x_min, dyke_plot[:, i] / 1000,
                         'r.')
            if i < self.loads - 1:
                plt.fill_between(self.x_plate / 1000 - x_min,
                                 strat_plot[:, i] / 1000,
                                 strat_plot[:, i + 1] / 1000,
                                 facecolor='grey', interpolate=False)

        # Fill final package
        plt.fill_between(self.x_plate / 1000 - x_min, strat_plot[:, -1] / 1000,
                         0, facecolor='grey', interpolate=False)

        # Set limits
        plt.ylim(deepest, 1)
        plt.xlim(0, x_range)

        # Labeling
        plt.xlabel('Distance (km)')
        plt.ylabel('Depth (km)')

        # Fix layout issues e.g. tick labels overlapping
        if animate is True:
            plt.yticks([deepest, 1])
        plt.tight_layout()

        # Force aspect ratio
        ax.set_aspect(3)

        # Save, padding title with zeros, if not given as string
        if title is None:
            if animate is False:
                plt_title = "strat_time_" + str(index).zfill(3) + ".pdf"
                plt_path = os.path.join('Strat_Plots', plt_title)
            if animate is True:
                plt_title = "strat_time_" + str(index).zfill(3) + ".png"
                plt_path = os.path.join('Strat_Plots', plt_title)
        else:
            if animate is False:
                plt_path = title + ".pdf"
            if animate is True:
                plt_path = title + ".png"

        if save is True:
            if animate is True:
                plt.savefig(plt_path, dpi=400, format='png')
            if animate is False:
                plt.savefig(plt_path, dpi=400, format='pdf')
            plt.close()

    def plot_strat(self, strat, top_dyke, animate=False, save=True):
        '''
        Plot stratigraphy at all times and save animation or only final figure

        save=True/False; whether to save figs output to file
        '''

        # Calculate limits
        deepest = 1000 * np.floor(np.nanmin(strat) / 1000) / 1000

        x_vals = np.transpose((strat < -50).nonzero())

        left = int(100000 * np.floor(self.x_plate[x_vals[0][0]] / 100000))
        right = int(100000 * np.ceil(self.x_plate[x_vals[-1][0]] / 100000))

        x_range = (right - left) / 1000
```

```
        x_min = left / 1000

        limits_vals = (deepest, x_range, x_min)

        # Make directory and plots for animation
        if animate is True:
            # TODO: grid not showing on gif but is on png's?
            if not os.path.exists('Strat_Plots'):
                os.makedirs('Strat_Plots')

            for j in range(self.loads):
                # If animate is true, must save timeslices
                self.plot_timeslice(strat[:, :, j], top_dyke[:, :, j],
                                    limits_vals, animate=True, index=j, save=True)

            # Animate
            clip = ImageSequenceClip('Strat_Plots', fps=4)
            clip.write_gif('strat_animate.gif')

        # Only plot end case if no animation
        else:
            self.plot_timeslice(strat[:, :, -1], top_dyke[:, :, -1],
                                limits_vals, title="final_stratigraphy", save=save)

    def isolate_flexure(self, grav_strat, densities):
        """
        Extract bounding horizons and average densities between jumps for strat

        grav_strat = 3D numpy array, modified for gravity to include zero top
        densities = list of infill densities

        Return list of class packages with base, top and average densities
        """
        # Prepare list of extracted polygons
        subsidence = list()

        # Record position of last jump (start at 0)
        start = 0
        end = 0

        for j in range(len(self.jumps) + 1):
            # Extract base
            package_base = grav_strat[:, start]

            # Move end to the next jump position
            if j == len(self.jumps):
                # For last sub package, set end to final index
                end = self.loads
            else:
                end = self.jumps[j][0]

            # Extract top
            package_top = grav_strat[:, end]

            # Find average of infill densities in this package
            tot = 0
            density_sum = 0

            # Only contribute non-zero values to average densities, as SDRs
            # with only dyke load driven subsidence will be volumetrically
            # much smaller and this would skew the average
            for l in range(end - start):
                if densities[start + l] != 0:
                    density_sum += densities[start + l]
                    tot += 1

            average_density = density_sum / tot

            # Append package to list
            subsidence.append(Package(package_base, package_top,
                              average_density))

            # All done - move start on
            start = end

        # Return subsidence, list of Packages class
        return subsidence

    def isolate_dyke_load(self, top_dyke, base_dyke, load_densities):
        """
        Isolate dyke load for gravity between jumps

        top_dyke = positions of tops of dykes, 2D or 3D numpy array
        base_dyke = positions of tops of dykes, 2D or 3D numpy array
        load_densities = list of load densities for all dykes

```

```
        If 3D arrays given, only the last time slice will be used

        Return list of class packages with base, top and average densities
        """

        # If passed 3D numpy array, extract final time slice only
        if len(top_dyke.shape) == 3:
            top_dyke = top_dyke[:, :, -1]

        if len(base_dyke.shape) == 3:
            base_dyke = base_dyke[:, :, -1]

        # Prepare list of extracted polygons
        dyke_body = list()

        # Record position of last jump (start at 0)
        start = 0
        end = 0

        pts = top_dyke.shape[0]

        for j in range(len(self.jumps) + 1):
            # Preallocate base and top
            package_base = np.nan * np.ones(pts)
            package_top = np.nan * np.ones(pts)

            # Move end to the next jump position
            if j == len(self.jumps):
                # For last sub package, set end to final index
                end = self.loads - 1
            else:
                # Otherwise end on final dyke BEFORE jump (-1)
                end = self.jumps[j][0] - 1

            # Extract base
            for i in range(end - start):
                for n in range(pts):
                    if not np.isnan(base_dyke[n, start + i]):
                        package_base[n] = base_dyke[n, start + i]

            # Extract top
            for i in range(end - start):
                for n in range(pts):
                    if not np.isnan(top_dyke[n, start + i]):
                        package_top[n] = top_dyke[n, start + i]

            # Find average of infill densities in this package
            tot = 0
            density_sum = 0

            # Calculate average load density for sub-package
            for l in range(end - start):
                density_sum += load_densities[start + l]
                tot += 1

            average_density = density_sum / tot

            # Append package to list
            dyke_body.append(Package(package_base, package_top,
                                     average_density))

            # All done - move start on to dyke at start of next package
            start = end + 1

        # Return subsidence, list of Packages class
        return dyke_body

    def add_zero_top(self, strat):
        """
        Extract final stratigraphy and add zero top horizon for gravity
        """
        strat_out = np.zeros([strat.shape[0], self.loads + 1])

        if len(strat.shape) == 3:
            strat_out[:, :-1] = strat[:, :, -1]

        elif len(strat.shape) == 2:
            strat_out[:, :-1] = strat

        return strat_out

    def grav_from_horizons(self, base, top, rho_b, rho_i, rho_a, rho_sides,
                           z_tol, x=None):
        """
        Calculates gravity between horizons with given densities using gbott
```

```
1672 |
1673 |         Inputs:
1674 |         base = base horizon
1675 |         top = top horizon
1676 |         rho_b = density below base horizon
1677 |         rho_i = density inside or between top and base horizons
1678 |         rho_a = density above top horizon
1679 |         rho_sides = density to use on sides, length of which will be determined
1680 |         by z_tol
1681 |         z_tol = minimum height difference in m between top and base at which
1682 |         to close polygon
1683 |
1684 |         Returns
1685 |         xs: 1D numpy array, station coordinates in km
1686 |         grav_feature: 1D numpy array of gravity profile with...
1687 |         p_x and p_z: lists of 1D numpy arrays, corresponding polygons coords km
1688 |         """
1689 |
1690 |         # Set default x value
1691 |         if x is None:
1692 |             x = copy(self.x_plate)
1693 |
1694 |         # Set up stations for gbott
1695 |         # Station variables - set to be the same as x
1696 |         nsta = x.shape[0]
1697 |         zp = 0  # zero elevation
1698 |
1699 |         # Convert to km for gbott
1700 |         xstart = (x[0]) / 1000
1701 |         xstep = (x[1] - x[0]) / 1000
1702 |
1703 |         # Make polygons from base, top; set densities
1704 |         # Find npts in horizon
1705 |         npts = base.shape[0]
1706 |         dyke_width_ind = self.dyke_width / self.x_step
1707 |
1708 |         # List of indices that satisfy z_tol
1709 |         all_poly_index = []
1710 |
1711 |         # Positions in above list to split at based on dyke_width
1712 |         split = []
1713 |
1714 |         # List of lists of polygon indices
1715 |         poly_index = []
1716 |
1717 |         # Loop through and find points in polygons that satisfy z_tol
1718 |         for i in range(npts):
1719 |             if (top[i] - base[i]) > z_tol:
1720 |                 all_poly_index.append(i)
1721 |
1722 |         # For points satisfying z_tol, check x tolerance
1723 |         for i in range(1, len(all_poly_index)):
1724 |             if (all_poly_index[i] - all_poly_index[i - 1]) > dyke_width_ind:
1725 |                 split.append(i)
1726 |
1727 |         # If only one body, extract this, otherwise loop through
1728 |         if len(split) == 0:
1729 |             poly_index.append(all_poly_index)
1730 |         else:
1731 |             for i in range(len(split)):
1732 |                 if i == 0:
1733 |                     poly_index.append(all_poly_index[:split[i]])
1734 |                 else:
1735 |                     poly_index.append(all_poly_index[split[i - 1]:split[i]])
1736 |
1737 |         # Build x, z and rho based on these indices
1738 |         # Lists of numpy arrays
1739 |         x_list = []
1740 |         z_list = []
1741 |         rho_list = []
1742 |
1743 |         for poly in poly_index:
1744 |             xpts = surf2poly(x, x, poly)
1745 |             zpts = surf2poly(base, top, poly)
1746 |             rho = surf2polyrho(rho_i, rho_b, rho_a, rho_sides, len(poly))
1747 |
1748 |             # Append to output
1749 |             x_list.append(xpts)
1750 |             z_list.append(zpts)
1751 |             rho_list.append(rho)
1752 |
1753 |         # Make dictionary of return values
1754 |         polygon_dict = {"x_list": x_list, "z_list": z_list,
1755 |                         "rho_list": rho_list}
1756 |
```

```
1757            # Prepare lists for export, to allow for generality where there may be
1758            #  multiple distinct polygons for a single package (if split by a jump)
1759            p_x = list()
1760            p_z = list()
1761
1762            # For all polygons extracted, calculate gravity using gbott
1763            for i in range(len(polygon_dict["x_list"])):
1764                grav_dict = grv_gbott(polygon_dict["x_list"][i] / 1000,
1765                                      -polygon_dict["z_list"][i] / 1000,
1766                                      polygon_dict["rho_list"][i], nsta, xstart,
1767                                      xstep, zp, p=0)
1768                if i == 0:
1769                    grv = grav_dict["grv"]
1770
1771                    # For first station (arbitrary choice) save xs too and prepare
1772                    # zero vector for summing of gravity
1773                    grv_xs = grav_dict["xs"]
1774                    grav_feature = np.zeros(grv_xs.shape)
1775                else:
1776                    grv = grav_dict["grv"]
1777
1778                # Store results only if maximum absolute value of grv exceeds 1mGal
1779                if np.amax(np.absolute(grv)) > 1:
1780                    grav_feature = np.add(grav_feature, grv)
1781                    p_x.append(polygon_dict["x_list"][i] / 1000)
1782                    p_z.append(polygon_dict["z_list"][i] / 1000)
1783
1784            # Return gravity, stations and corresponding polygon coordinates
1785            # grav_feature and grv_xs are single np arrays
1786            # p_x and p_z are lists of arrays
1787            return grav_feature, grv_xs, p_x, p_z
1788
1789        def calc_volc_POGM(self, strat, load_flex, top_dyke_lf,
1790                           base_dyke_lf, crust_thick):
1791            """
1792            Calculate components of volcanic POGM anomaly and save as object
1793            """
1794
1795            # TODO: Fix for load/load fill in dyke anomaly for pre-jump packages
1796
1797            def _calc_dyke_grav():
1798                """
1799                Calculate the dyke component of the process-oriented gravity
1800                anomaly
1801
1802                Three components:
1803                1. High from dyke load itself, displacing crust
1804                2. Low from basement downward flexure, displacing crust, infilling
1805                temporary (air/water)
1806                3. Low from moho downward flexure, crust into mantle
1807
1808                Repeat process for each sub-package and dyke complex between jumps.
1809
1810                """
1811
1812                # 1. High from dyke load itself, displacing crust
1813                # Isolate dyke load using load_flex, known dyke-width (constant)
1814                # and dyke_height for each dyke to find bottom of dyke
1815                dyke_bodies = self.isolate_dyke_load(top_dyke_lf, base_dyke_lf,
1816                                                     self.load_ds)
1817
1818                # For each package in dyke_bodies, number of which will be equal
1819                # to number of jumps + 1, calculate gravity anomaly using gbott.
1820                # Set ALL boundaries for ALL packages to c.rho_crust
1821
1822                # Prepare list to contain 1D numpy arrays of gravity profile for
1823                # each package in dyke_bodies
1824                dyke_bodies_grv = list()
1825                dyke_bodies_x = list()
1826                dyke_bodies_z = list()
1827
1828                # Calculate gravity for each sub package - density contrast of load
1829                # density average against c.rho_crust on all sides
1830                for package in dyke_bodies:
1831                    grv, xs, p_x, p_z = \
1832                        self.grav_from_horizons(package.base,
1833                                                package.top, c.rho_crust,
1834                                                package.rho, c.rho_crust,
1835                                                c.rho_crust, 10)
1836
1837                    # Save gravity to list, which will have same length as
1838                    # dyke_bodies
1839                    dyke_bodies_grv.append(grv)
1840                    dyke_bodies_x.append(p_x)
1841                    dyke_bodies_z.append(p_z)
```

```
1842
1843            # 2. Low from basement downward flexure, displacing crust,
1844            # infilling temporary (air/water)
1845
1846            # Use load_flex to find 'empty bowl'; set all boundaries to crust
1847            # Extract bodies of SDR packages between jumps for use in polygons
1848            load_driven_subs = self.isolate_flexure(load_flex, self.
1849                                                    temp_infill_ds)
1850
1851            # Prepare list to contain 1D numpy arrays of gravity profile for
1852            # each package in load_driven_subs
1853            load_driven_subs_grv = list()
1854            load_driven_subs_x = list()
1855            load_driven_subs_z = list()
1856
1857            # Calculate gravity for each sub package - density contrast set to
1858            # average temp infill density against c.rho_crust on all sides
1859            for package in load_driven_subs:
1860                # Make polygon from base, top; set densities
1861                grv, xs, p_x, p_z = \
1862                    self.grav_from_horizons(package.base, package.top,
1863                                            c.rho_crust, package.rho,
1864                                            c.rho_crust, c.rho_crust, 10)
1865
1866                # Save vars to list, which will have same length as
1867                # load_driven_subs
1868                load_driven_subs_grv.append(grv)
1869                load_driven_subs_x.append(p_x)
1870                load_driven_subs_z.append(p_z)
1871
1872            # 3. Low from Moho downward flexure, crust into mantle
1873            # Use load_flex and known crustal thickness to calculate size of
1874            # crustal polygon in mantle
1875
1876            # Calculate as single polygon with density contrast of crust to
1877            # mantle on base, using same polygon as step 2.
1878
1879            # Prepare list to contain 1D numpy arrays of gravity profile for
1880            # each package in load_driven_subs
1881            moho_subs_grv = list()
1882            moho_subs_x = list()
1883            moho_subs_z = list()
1884
1885            # Calculate gravity for each sub package - density contrast set to
1886            # c.rho_crust against c.rho_mantle on all sides
1887            for package in load_driven_subs:
1888                # Modify base, top to add crustal thickness
1889                top = package.top - crust_thick
1890                base = package.base - crust_thick
1891                # Make polygon from base, top; set densities, crust inside,
1892                # mantle all outside boundaries?
1893                grv, xs, p_x, p_z = \
1894                    self.grav_from_horizons(base, top, c.rho_mantle,
1895                                            c.rho_crust, c.rho_mantle,
1896                                            c.rho_mantle, 10)
1897
1898                # Save gravity to list, which will have same length as
1899                # moho_subs
1900                moho_subs_grv.append(grv)
1901                moho_subs_x.append(p_x)
1902                moho_subs_z.append(p_z)
1903
1904            # Build dictionaries for output
1905            dyke_bodies = {"dyke_bodies_x": dyke_bodies_x, "dyke_bodies_z":
1906                           dyke_bodies_z, "load_driven_subs_x":
1907                           load_driven_subs_x, "load_driven_subs_z":
1908                           load_driven_subs_z, "moho_subs_x": moho_subs_x,
1909                           "moho_subs_z": moho_subs_z}
1910
1911            dyke_grav = {"dyke_load": dyke_bodies_grv, "load_subs":
1912                         load_driven_subs_grv, "moho_flex": moho_subs_grv,
1913                         "xs": xs}
1914
1915            # Return list of gravity anomalies for each component, where each
1916            # list is the same length as the number of sub packages (jumps + 1)
1917
1918            return dyke_grav, dyke_bodies
1919
1920        def _calc_sdr_grav():
1921            """
1922            Calculate the SDR component of the process-oriented gravity anomaly
1923
1924            Three components:
1925            1. High from SDR load itself, filling existing depression
1926            2. Low from added basement downward flexure, displacing crust,
```

```
            infilling volcaniclastic
            3. Low from added moho downward flexure, crust into mantle

            Repeat process for each sub-package and dyke complex between jumps.

            """

            # 1. High from SDR load itself, filling existing depression

            # Extract bodies of SDR packages between jumps for use in polygons
            filled_load_driven_subs = self.isolate_flexure(load_flex, self.
                                                           infill_ds)

            # Prepare list to contain 1D numpy arrays of gravity profile for
            # each package in filled_load_driven_subs
            filled_load_driven_subs_grv = list()
            filled_load_driven_subs_x = list()
            filled_load_driven_subs_z = list()

            # Calculate gravity for each sub package - set density contrast to
            # average infill density against temp infill on all sides
            for package in filled_load_driven_subs:
                # Make polygon from base, top; set densities
                # TODO: Assumes temp infill density same for all dykes - makes
                # physical sense
                grv, xs, p_x, p_z = \
                    self.grav_from_horizons(package.base, package.top,
                                            self.temp_infill_ds[0],
                                            package.rho, self.temp_infill_ds
                                            [0], self.temp_infill_ds[0], 10)

                # Save gravity to list, which will have same length as
                # filled_load_driven_subs
                filled_load_driven_subs_grv.append(grv)
                filled_load_driven_subs_x.append(p_x)
                filled_load_driven_subs_z.append(p_z)

            # 2. Low from added basement downward flexure, displacing crust,
            # infilling volcaniclastic

            # Use the difference between the load driven depression above and
            # the total depression.
            total_infilled_subs = self.isolate_flexure(strat, self.infill_ds)

            # Prepare list to contain 1D numpy arrays of gravity profile for
            # each package in total_infilled_subs
            total_infilled_subs_grv = list()
            total_infilled_subs_x = list()
            total_infilled_subs_z = list()

            # Calculate gravity for each sub package - set density contrast to
            # average infill against crust on all sides
            for p in range(len(filled_load_driven_subs)):
                # Modify base, top to get only EXTRA space between load driven
                # subsidence and the total subsidence
                top = filled_load_driven_subs[p].base
                base = total_infilled_subs[p].base

                # Offset top by future jumped packages
                for r in range(1, len(filled_load_driven_subs) - p):
                    top = np.add(top, total_infilled_subs[p + r].base)

                grv, xs, p_x, p_z = \
                    self.grav_from_horizons(base, top, c.rho_crust,
                                            total_infilled_subs[p].rho,
                                            c.rho_crust, c.rho_crust, 10)

                # Save gravity to list, which will have same length as
                # total_infilled_subs
                total_infilled_subs_grv.append(grv)
                total_infilled_subs_x.append(p_x)
                total_infilled_subs_z.append(p_z)

            # 3. Low from added moho downward flexure, crust into mantle

            # Polygon is same as step 2 but with addition of crustal thickness.

            # Prepare list to contain 1D numpy arrays of gravity profile for
            # each package in moho_infill_subs
            moho_infill_subs_grv = list()
            moho_infill_subs_x = list()
            moho_infill_subs_z = list()

            # Calculate gravity for each sub package - set density contrast to
            # crust against mantle on all sides
```

```
            for p in range(len(total_infilled_subs)):
                # Modify base, top to get only EXTRA space between load driven
                # subsidence and the total subsidence, shift down by crustal
                # thickness to get to moho
                top = filled_load_driven_subs[p].base - crust_thick
                base = total_infilled_subs[p].base - crust_thick

                # Offset top by future jumped packages
                for r in range(1, len(filled_load_driven_subs) - p):
                    top = np.add(top, total_infilled_subs[p + r].base)

                # Make polygon from base, top; set densities
                grv, xs, p_x, p_z = \
                    self.grav_from_horizons(base, top, c.rho_mantle,
                                            c.rho_crust, c.rho_mantle,
                                            c.rho_mantle, 10)

                # Save gravity to list, which will have same length as
                # moho_infill_subs
                moho_infill_subs_grv.append(grv)
                moho_infill_subs_x.append(p_x)
                moho_infill_subs_z.append(p_z)

            # Build dictionary for output
            sdr_bodies = {"filled_load_driven_subs_x":
                          filled_load_driven_subs_x,
                          "filled_load_driven_subs_z":
                          filled_load_driven_subs_z, "total_infilled_subs_x":
                          total_infilled_subs_x, "total_infilled_subs_z":
                          total_infilled_subs_z, "moho_infill_subs_x":
                          moho_infill_subs_x, "moho_infill_subs_z":
                          moho_infill_subs_z}

            sdr_grav = {"load_fill": filled_load_driven_subs_grv,
                        "extra_fill": total_infilled_subs_grv,
                        "moho_flex": moho_infill_subs_grv, "xs": xs}

            # Return list of gravity anomalies for each component, where each
            # list is the same length as the number of sub packages (jumps + 1)
            return sdr_grav, sdr_bodies

        # Calculate dyke gravity
        (dyke_grav, dyke_bodies) = _calc_dyke_grav()
        (sdr_grav, sdr_bodies) = _calc_sdr_grav()

        # Build and return object
        return POGM_Anomalies(dyke_grav, dyke_bodies, sdr_grav,
                              sdr_bodies, self)

    def rift_SDR_packages(self, strat=None, top_dyke=None):
        """
        Split package at final dyke intrusion, drop both sides onto rift

        Inputs:
        strat - stratigraphy through time, 3D numpy array
        top_dyke - corresponding top_dyke positions through time, 3D array

        Default values are that of the current plate run's full, final,
        pre-rift stratigraphy

        """

        # Set default values if not given, set option to save outputs to plate:
        if top_dyke is None and strat is None:
            save = True
        else:
            save = False

        if strat is None:
            strat = copy(self.strat)

        if top_dyke is None:
            top_dyke = copy(self.top_dyke)

        # Split into 'left' and 'right' packages at end time, either side of
        # mid point
        sp = self.split_point
        sdr_l = strat[:sp, :, -1]
        sdr_r = strat[sp + 1:, :, -1]

        top_dyke_l = top_dyke[:sp, :, -1]
        top_dyke_r = top_dyke[sp + 1:, :, -1]

        # Use xrift output to make rift on same x scale as SDR model
        # Find x position in km of first max beta as pin point for end of SDR
```

```
        max_beta = np.amax(self.rift.beta_y)
        cob_index = find_first_nparray(max_beta, self.rift.beta_y)
        cob_x = self.rift.beta_x[cob_index] * 1000  # convert to m

        # Generate x vector of required length with same spacing as x_plate
        x_rift = np.arange(0, int(cob_x) + 1, self.x_step)

        # Sample xrift output onto this new x vector, convert to metres for
        # consistency with SDR model
        y_rift = np.interp(x_rift, self.rift.rift_x * 1000, -
                           self.rift.rift_y * 1000)

        # Add values in y_rift at relevant indices to extracted packages
        for i in range(self.loads):
            sdr_l[-y_rift.shape[0]:, i] = \
                np.add(sdr_l[-y_rift.shape[0]:, i], y_rift)
            top_dyke_l[-y_rift.shape[0]:, i] = \
                np.add(top_dyke_l[-y_rift.shape[0]:, i], y_rift)

            sdr_r[:y_rift.shape[0], i] = \
                np.add(sdr_r[:y_rift.shape[0], i], y_rift[::-1])
            top_dyke_r[:y_rift.shape[0], i] = \
                np.add(top_dyke_r[:y_rift.shape[0], i], y_rift[::-1])

        # Modify a version of x_plate to go with them
        x_plate_l = self.x_plate[:sp]
        x_plate_r = self.x_plate[sp + 1:]

        # Find rift_top explicitly for each side
        rift_top_l = np.zeros(sdr_l.shape[0])
        rift_top_r = np.zeros(sdr_r.shape[0])

        rift_top_l[-y_rift.shape[0]:] = \
            np.add(rift_top_l[-y_rift.shape[0]:], y_rift)
        rift_top_r[:y_rift.shape[0]] = \
            np.add(rift_top_r[:y_rift.shape[0]], y_rift[::-1])

        # Find rift_moho explicitly for each side using Airy isostasy
        rift_moho_l = -(c.zero_ct + rift_top_l *
                        ((c.rho_crust - c.rho_water) / (c.rho_mantle -
                                                        c.rho_crust)))
        rift_moho_r = -(c.zero_ct + rift_top_r *
                        ((c.rho_crust - c.rho_water) / (c.rho_mantle -
                                                        c.rho_crust)))

        # If running for default stratigraphy, save results to plate:
        if save is True:
            # Save rift surface to plate
            self.rift_top_l = rift_top_l
            self.rift_top_r = rift_top_r

            # Save rift moho to plate
            self.rift_moho_l = rift_moho_l
            self.rift_moho_r = rift_moho_r

            # Find mean depth of resulting rifted package for use in gravity
            # calculation
            self.rift_min_depth = np.min(y_rift)

            # Save values to plate
            self.sdr_l = sdr_l
            self.sdr_r = sdr_r
            self.top_dyke_l = top_dyke_l
            self.top_dyke_r = top_dyke_r
            self.x_plate_l = x_plate_l
            self.x_plate_r = x_plate_r

            return 1

        else:
            rift_results = {"sdr_l": sdr_l, "sdr_r": sdr_r, "rift_top_l":
                            rift_top_l, "rift_top_r": rift_top_r,
                            "rift_moho_l": rift_moho_l, "rift_moho_r":
                            rift_moho_r, "top_dyke_l": top_dyke_l,
                            "top_dyke_r": top_dyke_r, "x_plate_l": x_plate_l,
                            "x_plate_r": x_plate_r, "split_point": sp}

            return rift_results

    def calc_rifting_anomaly(self):
        """
        Calculate rifting anomaly using gbott, assuming water filled

        All inputs 1D numpy arrays
        """
```

```
2182 |
2183 |         # Pad out edges of array
2184 |         rift_top_l_pad = np.pad(self.rift_top_l, (100, 100), 'edge')
2185 |         rift_top_r_pad = np.pad(self.rift_top_r, (100, 100), 'edge')
2186 |         rift_moho_l_pad = np.pad(self.rift_moho_l, (100, 100), 'edge')
2187 |         rift_moho_r_pad = np.pad(self.rift_moho_r, (100, 100), 'edge')
2188 |
2189 |         # Extend x vectors accordingly - only spacing is important
2190 |         x_plate_l_pad = np.arange(1, rift_top_l_pad.shape[0] *
2191 |                                   self.x_step + 1, self.x_step)
2192 |         x_plate_r_pad = np.arange(1, rift_top_r_pad.shape[0] *
2193 |                                   self.x_step + 1, self.x_step)
2194 |
2195 |         # Make sea level top (zero) for water fill
2196 |         sl_l = np.zeros(rift_top_l_pad.shape)
2197 |         sl_r = np.zeros(rift_top_r_pad.shape)
2198 |
2199 |         # Make base for moho contribution
2200 |         base_l = -c.zero_ct * np.ones(rift_top_l_pad.shape)
2201 |         base_r = -c.zero_ct * np.ones(rift_top_r_pad.shape)
2202 |
2203 |         # Calculate water anomaly for each side
2204 |         (grv_water_l, grv_xs_l, _, _) = \
2205 |             self.grav_from_horizons(rift_top_l_pad, sl_l, c.rho_crust,
2206 |                                     c.rho_water, c.rho_air, c.rho_water, 10,
2207 |                                     x=x_plate_l_pad)
2208 |         (grv_water_r, _, _, _) = \
2209 |             self.grav_from_horizons(rift_top_r_pad, sl_r, c.rho_crust,
2210 |                                     c.rho_water, c.rho_air, c.rho_water, 10,
2211 |                                     x=x_plate_r_pad)
2212 |
2213 |         # Calculate moho anomaly for each side
2214 |         (grv_moho_l, _, _, _) = \
2215 |             self.grav_from_horizons(base_l, rift_moho_l_pad, c.rho_mantle,
2216 |                                     c.rho_mantle, c.rho_crust, c.rho_mantle,
2217 |                                     0.1, x=x_plate_l_pad)
2218 |         (grv_moho_r, _, _, _) = \
2219 |             self.grav_from_horizons(base_r, rift_moho_r_pad, c.rho_mantle,
2220 |                                     c.rho_mantle, c.rho_crust, c.rho_mantle,
2221 |                                     0.1, x=x_plate_r_pad)
2222 |
2223 |         # Sum
2224 |         rift_anom_l = np.add(grv_water_l, grv_moho_l)
2225 |         rift_anom_r = np.add(grv_water_r, grv_moho_r)
2226 |
2227 |         # Remove padding, save to plate
2228 |         grv_water_l = grv_water_l[100:-100]
2229 |         grv_moho_l = grv_moho_l[100:-100]
2230 |         grv_xs_l = grv_xs_l[100:-100]
2231 |
2232 |         self.rift_anom_l = rift_anom_l[100:-100]
2233 |         self.rift_anom_r = rift_anom_r[100:-100]
2234 |
2235 |         # Return
2236 |         return 1
2237 |
2238 |     def plot_rift_packages(self, pogm_obj=None, title=None, save=True):
2239 |         """
2240 |         Plot final stratigraphy of rifted packages
2241 |
2242 |         Optional arguments:
2243 |         pogm_obj = POGM_Anomalies object if plotting of volcanic anomaly
2244 |         desired
2245 |
2246 |         title = optional filename without extension, otherwise default value
2247 |         of rifted_margin.pdf
2248 |
2249 |         save = bool, whether to save figure output to file
2250 |         """
2251 |
2252 |         # Update plotting parameters
2253 |         plt.rcParams.update(plot_params)
2254 |
2255 |         # Set y limits, use same minima for both sides
2256 |         minimum = np.nanmin(self.sdr_l)
2257 |
2258 |         if np.nanmin(self.sdr_r) < minimum:
2259 |             minimum = np.nanmin(self.sdr_r)
2260 |
2261 |         deepest = 1000 * np.floor(minimum / 1000) / 1000
2262 |
2263 |         # Set x limits
2264 |         x_l_vals = np.transpose((self.sdr_l < -75).nonzero())
2265 |         x_r_vals = np.transpose((self.sdr_r < -75).nonzero())
2266 |
```

```
        min_xind_l = int(100 * np.floor(x_l_vals[0][0] / 100))
        max_xind_l = self.x_plate_l.shape[0] - 1

        min_xind_r = 0
        max_xind_r = int(100 * np.ceil(x_r_vals[-1][0] / 100))

        x_range_l = (self.x_plate_l[max_xind_l] -
                     self.x_plate_l[min_xind_l]) / 1000
        x_min_l = self.x_plate_l[min_xind_l] / 1000
        x_range_r = (self.x_plate_r[max_xind_r] -
                     self.x_plate_r[min_xind_r]) / 1000
        x_min_r = self.x_plate_r[min_xind_r] / 1000

        # Set up axes
        fig = plt.figure()
        gs = gridspec.GridSpec(2, 2, width_ratios=[x_range_l / x_range_r, 1])
        ax_stratl = fig.add_subplot(gs[2])
        ax_stratr = fig.add_subplot(gs[3])
        ax_gravl = fig.add_subplot(gs[0])
        ax_gravr = fig.add_subplot(gs[1])

        # Plot left hand side
        ax_stratl.set_aspect(5)
        ax_stratl.set_ylim(deepest, 0)
        ax_stratl.set_ylabel('Depth (km)')
        ax_stratl.set_xlabel('Distance (km)')
        ax_stratl.set_xlim(0, x_range_l)

        for i in range(self.loads):
            ax_stratl.plot(self.x_plate_l / 1000 - x_min_l, self.sdr_l[:, i] /
                           1000, lw=1.5, color='black')
            ax_stratl.plot(self.x_plate_l / 1000 - x_min_l,
                           self.top_dyke_l[:, i] / 1000, 'r.')

            if i < self.loads - 1:
                ax_stratl.fill_between(self.x_plate_l / 1000 - x_min_l,
                                       self.sdr_l[:, i] / 1000,
                                       self.sdr_l[:, i + 1] / 1000,
                                       facecolor='grey', interpolate=False)

        # Fill final package
        ax_stratl.fill_between(self.x_plate_l / 1000 - x_min_l,
                               self.sdr_l[:, -1] / 1000, self.rift_top_l /
                               1000, facecolor='grey', interpolate=False)

        # Plot right hand side
        ax_stratr.set_ylim(deepest, 0)
        ax_stratr.set_aspect(5)
        ax_stratr.yaxis.tick_right()
        ax_stratr.set_xlabel('Distance (km)')
        ax_stratr.set_xlim(0, x_range_r)

        # Shift x labels to better match LHS
        tick_shift = int(10 * np.ceil(x_range_l / 10))
        ax_stratr.set_xticklabels([str(int(x) + tick_shift) for x in
                                   ax_stratr.get_xticks()])

        for i in range(self.loads):
            ax_stratr.plot(self.x_plate_r / 1000 - x_min_r, self.sdr_r[:, i] /
                           1000, lw=1.5, color='black')
            ax_stratr.plot(self.x_plate_r / 1000 - x_min_r,
                           self.top_dyke_r[:, i] / 1000, 'r.')

            if i < self.loads - 1:
                ax_stratr.fill_between(self.x_plate_r / 1000 - x_min_r,
                                       self.sdr_r[:, i] / 1000,
                                       self.sdr_r[:, i + 1] / 1000,
                                       facecolor='grey', interpolate=False)

        # Fill final package
        ax_stratr.fill_between(self.x_plate_r / 1000 - x_min_r,
                               self.sdr_r[:, -1] / 1000,
                               self.rift_top_r / 1000, facecolor='grey',
                               interpolate=False)

        # Plot rifting anomaly
        ax_gravl.plot(self.x_plate_l / 1000 - x_min_l, self.rift_anom_l,
                      label='Rifting Anomaly')
        ax_gravr.plot(self.x_plate_r / 1000 - x_min_r, self.rift_anom_r,
                      label='Rifting Anomaly')

        ax_gravl.set_ylabel('mGal')
        ax_gravl.set_xlabel('Distance (km)')
```

```
        ax_gravl.set_xlim(0, x_range_l)
        ax_gravl.set_aspect(0.2)

        ax_gravr.set_ylabel('mGal')
        ax_gravr.set_xlabel('Distance (km)')
        ax_gravr.set_xlim(0, x_range_r)
        ax_gravr.set_aspect(0.2)

        # If given, add plotting of rifting anomalies
        if pogm_obj is not None:

            # Extract and split Dyke/SDR Sum anomalies
            dyke_anom_l = pogm_obj.sum_grav["dyke_sum"][:self.split_point]
            dyke_anom_r = pogm_obj.sum_grav["dyke_sum"][self.split_point + 1:]

            sdr_anom_l = pogm_obj.sum_grav["SDR_sum"][:self.split_point]
            sdr_anom_r = pogm_obj.sum_grav["SDR_sum"][self.split_point + 1:]

            ax_gravl.plot(self.x_plate_l / 1000 - x_min_l, dyke_anom_l,
                          label='Dyke Anomaly')
            ax_gravl.plot(self.x_plate_l / 1000 - x_min_l, sdr_anom_l,
                          label='SDR Anomaly')

            ax_gravr.plot(self.x_plate_r / 1000 - x_min_r, dyke_anom_r,
                          label='Dyke Anomaly')
            ax_gravr.plot(self.x_plate_r / 1000 - x_min_r, sdr_anom_r,
                          label='SDR Anomaly')

        # Add legends
        ax_gravr.legend()
        ax_gravl.legend()

        # Make it bigger!
        curr_size = fig.get_size_inches()
        fig.set_size_inches(curr_size[0] * 2, curr_size[1] * 2)

        if title is None:
            filename = "rifted_margin.pdf"
        else:
            filename = title + ".pdf"
        plt.tight_layout()
        if save is True:
            plt.savefig(filename, dpi=400, format='pdf')
            plt.close()

    def calc_pogm_after_rift(self):
        """
        Calculate process-oriented gravity inc. effects of rifting
        """
        def _add_basement(strat, basement):
            """
            Add arbitrary top (basement) to a 2D strat variable, assuming
            same number of x points
            """

            # TODO: This is basically the same function as add_zero_top,
            # consolidate these!

            if len(basement.shape) == 1:
                basement_2d = np.reshape(basement, (-1, 1))
            elif len(basement.shape) == 2:
                basement_2d = basement

            strat_topped = np.concatenate((strat, basement_2d), axis=1)

            return strat_topped

        # To calculate POGM, we also need to rift the load_flexure alone:
        rift_load_flex = self.rift_SDR_packages(strat=self.load_flex,
                                                top_dyke=self.top_dyke_lf)

        # Also need to know base_dyke_lf for each side
        base_dyke_lf_l = self.find_dyke_base(rift_load_flex["top_dyke_l"])
        base_dyke_lf_r = self.find_dyke_base(rift_load_flex["top_dyke_r"])

        # Add basement to surfaces, in place of what would otherwise be
        # zero-topped in the gravity calculation as the top horizon is
        # missing at this stage
        sdr_l = _add_basement(self.sdr_l, self.rift_top_l)
        sdr_r = _add_basement(self.sdr_r, self.rift_top_r)

        load_flex_l = _add_basement(rift_load_flex["sdr_l"],
                                    rift_load_flex["rift_top_l"])
        load_flex_r = _add_basement(rift_load_flex["sdr_r"],
                                    rift_load_flex["rift_top_r"])
```

```
2435 |
2436 |         # Now stitch the two halves back together for gravity calculation
2437 |         strat = np.concatenate((sdr_l, sdr_r), axis=0)
2438 |         load_flex = np.concatenate((load_flex_l, load_flex_r), axis=0)
2439 |         top_dyke_lf = np.concatenate((rift_load_flex["top_dyke_l"],
2440 |                                       rift_load_flex["top_dyke_r"]), axis=0)
2441 |         base_dyke_lf = np.concatenate((base_dyke_lf_l, base_dyke_lf_r), axis=0)
2442 |
2443 |         # Calculate gravity
2444 |         postrift_anoms = self.calc_volc_POGM(strat, load_flex, top_dyke_lf,
2445 |                                              base_dyke_lf,
2446 |                                              self.rift.crust_thick_min)
2447 |
2448 |         # Return
2449 |         return postrift_anoms
2450 |
2451 |     def plot_curvature(self, strat, top_dyke, save=True):
2452 |         """
2453 |         For the given 2D stratigraphy numpy array, plot all curvatures
2454 |
2455 |         save = bool, whether to save output figure to file
2456 |         """
2457 |
2458 |         # Prepare empty array for curvature
2459 |         curv = np.empty(strat.shape)
2460 |
2461 |         # If actually the top of dyke surface, also set to nan
2462 |         for i in range(strat.shape[1]):
2463 |             inds = list()
2464 |             for j in range(strat.shape[0]):
2465 |                 if not np.isnan(top_dyke[j, i]):
2466 |                     inds.append(j)
2467 |
2468 |             if inds:
2469 |                 strat[inds[0]:inds[-1] + 1, i] = np.nan
2470 |
2471 |         # Update plotting parameters
2472 |         plt.rcParams.update(plot_params)
2473 |
2474 |         # Set up figure geometry
2475 |         fig = plt.figure()
2476 |         ax1 = fig.add_subplot(211)
2477 |         ax2 = fig.add_subplot(212)
2478 |
2479 |         # Adjust x range according to position of SDRs
2480 |         x = self.x_plate / 1000 - self.x_min
2481 |
2482 |         # Calculate and plot curvature in log10 m^-1, stratigraphy
2483 |         for i in range(strat.shape[1]):
2484 |             curv[:, i] = np.log10(calc_curvature(self.x_plate, strat[:, i]))
2485 |             ax2.plot(x, curv[:, i])
2486 |             ax1.plot(x, strat[:, i] / 1000)
2487 |
2488 |         ax1.set_xlim([0, self.x_range])
2489 |         ax2.set_xlim([0, self.x_range])
2490 |
2491 |         # Force y limits to region of interest for yielding
2492 |         ax2.set_ylim([-10, 0])
2493 |
2494 |         ax2.set_xlabel('Distance (km)')
2495 |         ax1.set_ylabel('Depth (km)')
2496 |         ax2.set_ylabel('Curvature $\log_{10}$ m$^{-1}$')
2497 |
2498 |         plt.tight_layout()
2499 |         if save is True:
2500 |             plt.savefig("curvatures.pdf", dpi=400, format='pdf')
2501 |             plt.close()
2502 |
2503 |     def export_stratigraphy(self, strat=None, top_dyke=None, filename="strat"):
2504 |         """
2505 |         Export the final stratigraphic geometry to use elsewhere e.g. fatiando
2506 |
2507 |         Builds a dictionary and uses pickle to save file for use in other
2508 |         Python scripts/functions.
2509 |
2510 |         Inputs:
2511 |         strat - stratigraphy through time, 3D numpy array
2512 |         top_dyke - corresponding top_dyke positions through time, 3D array
2513 |         filename - (without extension) name of pickle file, string
2514 |
2515 |         Default values are that of the current plate run's full, final,
2516 |         pre-rift stratigraphy, saving as strat.p
2517 |
2518 |         """
2519 |
```

```
2520        # Set default values if not given
2521        if strat is None:
2522            strat = copy(self.strat)
2523
2524        if top_dyke is None:
2525            top_dyke = copy(self.top_dyke)
2526
2527        # Build dictionary to save
2528        export_dict = {'x_spacing': self.x_step, "strat_horizons":
2529                       strat[:, :, -1], "top_dyke": top_dyke[:, :, -1],
2530                       "infill_rho": self.infill_ds, "load_rho": self.load_ds,
2531                       "c.rho_crust": c.rho_crust, "npts": self.npts,
2532                       "x_plate": self.x_plate, "jumps": self.jumps}
2533
2534        # Pickle it
2535        file = filename + ".p"
2536        pickle.dump(export_dict, open(file, "wb"), protocol=2)
2537
2538
2539 class Dyke(object):
2540     # Save all dyke configurations in dict to reduce computation for duplicate
2541     # dyke parameter sets
2542     existing_dykes = {}
2543
2544     @classmethod
2545     def add_dyke(cls, host_plate, h, rho_i, rho_l, rho_c, rho_ti, Te_max,
2546                  Te_min, Te_width, dx):
2547         """
2548         Only create new dyke if same params doesn't already exist.
2549
2550         Use instead of directly constructing dykes for efficiency
2551         """
2552
2553         # Add to dictionary with key as tuple of params in order given
2554         params = (host_plate, h, rho_i, rho_l, rho_c, rho_ti, Te_max,
2555                   Te_min, Te_width, dx)
2556
2557         if params not in cls.existing_dykes:
2558             D = cls(*params)
2559         else:
2560             D = cls.existing_dykes[params]
2561
2562         return D
2563
2564     def __init__(self, host_plate, h, rho_i, rho_l, rho_c, rho_ti, Te_max,
2565                  Te_min, Te_width, dx):
2566         # Allocate variables
2567         self.host_plate = host_plate  # class
2568         self.height = h
2569         self.rho_infill = rho_i
2570         self.rho_load = rho_l
2571         self.rho_comp = rho_c
2572         self.rho_ti = rho_ti  # temporary infill
2573         self.Te_max = Te_max
2574         self.Te_min = Te_min
2575
2576         # half-width of linear Te change in m, 0 for
2577         # default half of dyke width
2578         self.Te_width = Te_width
2579
2580         # x resolution in 2d flexure calc, m
2581         self.dx = dx
2582
2583         # Inherit width from host plate
2584         self.width = self.host_plate.dyke_width
2585
2586         # Calculate z_load array
2587         x = np.arange(-0.5 * self.width, 0.5 *
2588                       self.width, self.host_plate.x_step)
2589         self.z_load = -self.height * np.ones(x.shape[0])
2590
2591         # Setup Te profiles
2592         self._te_setup()
2593
2594         # Flex and fill
2595         self._flex_and_fill()
2596
2597         # Add to dictionary of existing dykes
2598         params = (host_plate, h, rho_i, rho_l, rho_c, rho_ti, Te_max,
2599                   Te_min, Te_width, dx)
2600         self.existing_dykes[params] = self
2601
2602     def _te_setup(self):
2603         """
2604         Set up Te profiles for broken plate model
2605         """
```

```
2606 |
2607 |         # Calculate Te profiles
2608 |         z_te = np.zeros(self.host_plate.npts)
2609 |         half_dyke_w = int(0.5 * self.width / self.host_plate.x_step)
2610 |
2611 |         # If Te_width is set to dyke width, use that, otherwise use defined
2612 |         # value
2613 |         if self.Te_width is 0:
2614 |             te_dist = half_dyke_w
2615 |         else:
2616 |             te_dist = int(self.Te_width / self.host_plate.x_step)
2617 |
2618 |         # Fill Te profile using linear increase/decrease to mid dyke
2619 |         z_te[:self.host_plate.emp - te_dist - 1] = self.Te_max
2620 |         z_te[self.host_plate.emp - te_dist - 1:self.host_plate.emp] = \
2621 |             np.linspace(self.Te_max, self.Te_min, te_dist + 1)
2622 |         z_te[self.host_plate.emp - 1:self.host_plate.emp + te_dist] = \
2623 |             np.linspace(self.Te_min, self.Te_max, te_dist + 1)
2624 |         z_te[self.host_plate.emp + te_dist:] = self.Te_max
2625 |
2626 |         self.z_te = z_te
2627 |
2628 |     def _flex_and_fill(self):
2629 |         """
2630 |         Load a broken plate and calculate flexure due to load and infill.
2631 |
2632 |         Load a broken plate with a given elastic thickness profile with a
2633 |         number of loads. The density of the load, temporary infill and
2634 |         final infill are specified. The flexure due to the load and due to
2635 |         the infill are saved separately for each stage.
2636 |
2637 |         """
2638 |
2639 |         # Prepare new load
2640 |         load = np.zeros(self.host_plate.npts)
2641 |         infill = np.zeros(self.host_plate.npts)
2642 |
2643 |         load[self.host_plate.emp - 1:self.host_plate.emp +
2644 |              self.z_load.shape[0] - 1] = self.z_load
2645 |
2646 |         # Find infill vector, with value of one one for air for areas with no
2647 |         # dyke load (negative areas)
2648 |         infill_vector = np.ones(self.host_plate.npts)
2649 |
2650 |         for i in range(self.host_plate.npts):
2651 |             if load[i] < 0:
2652 |                 infill_vector[i] = self.rho_ti
2653 |
2654 |         # Override with constant infill
2655 |         infill_vector[:] = self.rho_ti
2656 |
2657 |         # Calculate and plot flexure due to this load only
2658 |         load_out = flex_2d(xload=self.host_plate.x_plate, zload=load,
2659 |                            x_te=self.host_plate.x_plate, z_te=self.z_te,
2660 |                            prof_length=self.host_plate.prof_length,
2661 |                            dx=self.dx, rho_load=self.rho_load,
2662 |                            rho_displace=c.rho_crust, rho_infill=infill_vector,
2663 |                            rho_compensation=self.rho_comp, p=False)
2664 |
2665 |         # Extract flex - negative due to buried
2666 |         load_flex = -load_out["flex"]
2667 |
2668 |         # Infill flexure
2669 |         for j in range(self.host_plate.npts):
2670 |             if load_flex[j] < 0:
2671 |                 infill[j] = -load_flex[j]
2672 |
2673 |         # Find infill vector, with value of one for air for areas with no
2674 |         # infill load (positive areas)
2675 |         infill_vector = np.ones(self.host_plate.npts)
2676 |
2677 |         for i in range(self.host_plate.npts):
2678 |             if infill[i] > 0:
2679 |                 infill_vector[i] = self.rho_infill
2680 |
2681 |         # Override with constant infill
2682 |         infill_vector[:] = self.rho_infill
2683 |         # TODO: Work out where I left variable infill; remove?
2684 |
2685 |         # Calculate and flexure due to infill, replace on plate
2686 |         infill_out = flex_2d(xload=self.host_plate.x_plate, zload=infill,
2687 |                              x_te=self.host_plate.x_plate, z_te=self.z_te,
2688 |                              prof_length=self.host_plate.prof_length,
2689 |                              dx=self.dx, rho_load=self.rho_infill,
2690 |                              rho_displace=self.rho_ti,
```

```
                                rho_infill=infill_vector,
                                rho_compensation=self.rho_comp)

        # Extract flex only
        infill_flex = infill_out["flex"]

        # Add together to sum morphology
        morpho = load_flex + infill_flex

        # Assign to dyke attributes
        self.morpho = morpho
        self.load_flex = load_flex
        self.infill_flex = infill_flex
        self.curv = load_out["curv"] + infill_out["curv"]


class Package(object):
    def __init__(self, base, top, rho):
        # Allocate variables
        self.base = base    # numpy array
        self.top = top  # numpy array
        self.rho = rho  # scalar value


class POGM_Anomalies(object):
    def __init__(self, dyke_grav, dyke_bodies, sdr_grav, sdr_bodies, plate):
        """
        Allocate variables
        """

        self.dyke_grav = dyke_grav
        self.dyke_bodies = dyke_bodies
        self.sdr_grav = sdr_grav
        self.sdr_bodies = sdr_bodies

        # Extract plotting ranges from plate
        self.x_min = plate.x_min
        self.x_range = plate.x_range
        self.split_point_val = plate.x_plate[plate.split_point] / 1000

        # Sum anomalies
        self._sum_pogm_anomalies()

    def _sum_pogm_anomalies(self):
        """
        Sum POGM anomalies for dyke and SDR grav
        """

        # Extract variables with colliding keys
        xs = self.sdr_grav["xs"]
        sdr_moho_flex = self.sdr_grav["moho_flex"]
        dyke_moho_flex = self.dyke_grav["moho_flex"]

        # Calculate sums for plotting for SDR
        load_fill_sum = np.zeros(xs.shape)
        extra_fill_sum = np.zeros(xs.shape)
        moho_SDR_sum = np.zeros(xs.shape)

        for array in self.sdr_grav["load_fill"]:
            load_fill_sum = np.add(load_fill_sum, array)

        for array in self.sdr_grav["extra_fill"]:
            extra_fill_sum = np.add(extra_fill_sum, array)

        for array in sdr_moho_flex:
            moho_SDR_sum = np.add(moho_SDR_sum, array)

        SDR_sum = np.add(load_fill_sum, extra_fill_sum)
        SDR_sum = np.add(SDR_sum, moho_SDR_sum)

        # Calculate sums for plotting for dyke
        dyke_load_sum = np.zeros(xs.shape)
        load_subs_sum = np.zeros(xs.shape)
        moho_dyke_sum = np.zeros(xs.shape)

        for array in self.dyke_grav["dyke_load"]:
            dyke_load_sum = np.add(dyke_load_sum, array)

        for array in self.dyke_grav["load_subs"]:
            load_subs_sum = np.add(load_subs_sum, array)

        for array in dyke_moho_flex:
            moho_dyke_sum = np.add(moho_dyke_sum, array)

        dyke_sum = np.add(dyke_load_sum, load_subs_sum)
```

```
        dyke_sum = np.add(dyke_sum, moho_dyke_sum)

        overall_sum = np.add(SDR_sum, dyke_sum)

        # Calculate residuals
        integral = np.trapz(overall_sum)
        integral_sdr = np.trapz(SDR_sum)
        integral_dyke = np.trapz(dyke_sum)

        # Print residuals for inspection
        print("Integrated sum of gravity: ", integral)
        print("Integrated sum of SDR gravity: ", integral_sdr)
        print("Integrated sum of Dyke gravity: ", integral_dyke)

        # Build dictionary of outputs with matching keys
        outputs = {}

        for i in ('dyke_load_sum', 'load_subs_sum', 'moho_dyke_sum',
                  'dyke_sum', 'overall_sum', 'load_fill_sum', 'extra_fill_sum',
                  'moho_SDR_sum', 'SDR_sum', 'xs'):
            outputs[i] = locals()[i]

        # Save as class variable
        self.sum_grav = outputs

    def _get_modelled_gravity_sdrs(self):
        """
        Export the modeled gravity directly over the SDRs

        Returns 2x2D numpy arrays, model_left and model_right, where the first
        column is the horizontal position in km and the second column is the
        calculated gravity over this region. These are split at the eventual
        rift split point; for no jumps case, should be symmetrical.
        """

        # Extract extreme edges fo SDR wedge
        for i, array in enumerate(self.sdr_bodies["total_infilled_subs_x"]):
            left_edge = np.amin(array)
            right_edge = np.amax(array)
            if i == 0:
                min_left_edge = left_edge
                max_right_edge = right_edge
            if left_edge < min_left_edge:
                min_left_edge = left_edge
            if right_edge > max_right_edge:
                max_right_edge = right_edge

        mid_point = self.split_point_val

        # Get model values over this region
        model = np.hstack((self.sum_grav["xs"].reshape((-1, 1)),
                           self.sum_grav["overall_sum"].reshape((-1, 1))))
        model_left = model[(min_left_edge < model[:, 0]) &
                           (mid_point > model[:, 0])]
        model_right = model[(mid_point < model[:, 0]) &
                            (max_right_edge > model[:, 0])]

        return model_left, model_right

    def compare_obs_grav(self, file, orientation, save=True):
        """
        Given path of a .csv file containing observed gravity, calculate fit

        file = path to .csv file containing two columns, first with distance in
               km and second with isostatic gravity anomaly, over SDRs.
               Landward side should be the lower x values.
        orientation = 'E' for land on the East and 'W' for land on the West.
                      Determines which half of the model output to use.
        save = bool, whether to save outputs as files
        """

        # Read modeled and observed data
        model_left, model_right = self._get_modelled_gravity_sdrs()
        if orientation == 'E':
            model = np.hstack((model_right[:, 0].reshape((-1, 1)),
                               np.flipud(model_right[:, 1].reshape((-1, 1)))))
        else:
            model = model_left

        # My MATLAB script currently always exports with land on the left
        obs = np.genfromtxt(file, delimiter=',')

        # Modify offset on both such that dist starts at zero
        obs[:, 0] = obs[:, 0] - obs[0, 0]
        model[:, 0] = model[:, 0] - model[0, 0]
```

```
2862            # Shift in y direction to match means as starting point
2863            obs_mean = np.mean(obs[:, 1])
2864            model_mean = np.mean(model[:, 1])
2865            model[:, 1] = model[:, 1] + (obs_mean - model_mean)
2866            rms = rmse(model, obs)
2867            shift = 0
2868
2869            model_max = np.max(model[:, 1])
2870            model_min = np.min(model[:, 1])
2871            model_halfrange = int(np.ceil((model_max - model_min) / 2))
2872
2873            for y in range(-model_halfrange, model_halfrange):
2874                shift_model = np.hstack((model[:, 0].reshape(-1, 1),
2875                                         (model[:, 1] + y).reshape(-1, 1)))
2876                new_rms = rmse(shift_model, obs)
2877
2878                if new_rms < rms:
2879                    rms = new_rms
2880                    shift = y
2881
2882            # Apply best y-shift
2883            model = np.hstack((model[:, 0].reshape(-1, 1),
2884                               (model[:, 1] + shift).reshape(-1, 1)))
2885
2886            # Save statistics
2887            if save is True:
2888                filename = 'grav_fit_stats.csv'
2889                with open(filename, 'w', newline='') as csv_file:
2890                    writer = csv.writer(csv_file, delimiter=',')
2891                    writer.writerow(['RMSE', rms])
2892
2893            # Simple plot
2894            plt.rcParams.update(plot_params)
2895            legend_model_str = "Model RMSE = " + str(rms)
2896            plt.figure()
2897            plt.plot(obs[:, 0], obs[:, 1], 'r-', label="Observed")
2898            plt.plot(model[:, 0], model[:, 1], 'b--', label=legend_model_str)
2899            plt.legend()
2900            plt.tight_layout()
2901
2902            # Save figure
2903            if save is True:
2904                filename = 'obs_grav_fit.pdf'
2905                plt.savefig(filename, dpi=400, format='pdf')
2906                plt.close()
2907
2908            return
2909
2910        def plot_pogm_components(self, title, xlim=None, ylim_grav=None,
2911                                 ylim_bodies=None, save=True):
2912            """
2913            Plot POGM dyke and SDR components with bodies
2914
2915            Inputs:
2916            title = name of output file without extensions
2917            xlim = list, x limits if you want to force [lower, upper]
2918            ylim_grav = list, y limits if you want to force for grav [lower, upper]
2919            ylim_bodies = list, y limits if you want to force for bodies
2920            plot [lower, upper]
2921            save = whether to save output file
2922            """
2923
2924            # Set x limits if not given
2925            if xlim is None:
2926                xlim = [0, self.x_range]
2927
2928            # TODO: Fix for jump cases, not plotting (and maybe not calculating)
2929            # extra infill. Also extra and load driven subsidence overlap
2930
2931            x = self.sum_grav["xs"] - self.x_min
2932
2933            # Plot results
2934
2935            # Update plotting parameters
2936            plt.rcParams.update(plot_params)
2937
2938            plt.figure(figsize=(7, 10))
2939            plt.gca().set_prop_cycle(plt.rcParams['axes.prop_cycle'])
2940
2941            # Set up subplot grid
2942            ax_dyke = plt.subplot2grid((4, 1), (0, 0))
2943            ax_sdr = plt.subplot2grid((4, 1), (1, 0))
2944            ax_sum = plt.subplot2grid((4, 1), (2, 0))
2945            ax_bodies = plt.subplot2grid((4, 1), (3, 0),)
2946
```

```
        # Find max/min based on total infill for plot
        min_in = 0
        max_in = -c.zero_ct

        # Prepare lists for shading dyke region on gravity ploys
        dyke_l = list()
        dyke_r = list()

        # Plot bodies
        for i in range(len(self.dyke_bodies["dyke_bodies_x"])):
            for j in range(len(self.dyke_bodies["dyke_bodies_x"][i])):
                ax_bodies.fill(self.dyke_bodies["dyke_bodies_x"][i][j] -
                               self.x_min, self.dyke_bodies["dyke_bodies_z"]
                               [i][j], facecolor='red', label='Dyke Load')

            dyke_l.append(min(min(self.dyke_bodies["dyke_bodies_x"][i])) -
                          self.x_min)
            dyke_r.append(max(max(self.dyke_bodies["dyke_bodies_x"][i])) -
                          self.x_min)

        for i in range(len(self.dyke_bodies["load_driven_subs_x"])):
            for j in range(len(self.dyke_bodies["load_driven_subs_x"][i])):
                ax_bodies.fill(self.dyke_bodies["load_driven_subs_x"][i][j] -
                               self.x_min, self.dyke_bodies
                               ["load_driven_subs_z"][i][j], facecolor='blue',
                               label='Dyke Driven Subsidence')

        for i in range(len(self.sdr_bodies["total_infilled_subs_x"])):
            for j in range(len(self.sdr_bodies["total_infilled_subs_x"][i])):
                ax_bodies.fill(self.sdr_bodies["total_infilled_subs_x"][i][j] -
                               self.x_min,
                               self.sdr_bodies["total_infilled_subs_z"][i][j],
                               facecolor='green',
                               label='Infill Driven Subsidence')

                # Extract maximum and minimum
                infill_temp_min = np.nanmin(
                    self.sdr_bodies["total_infilled_subs_z"][i][j])
                infill_temp_max = np.nanmax(
                    self.sdr_bodies["total_infilled_subs_z"][i][j])

                if infill_temp_min < min_in:
                    min_in = infill_temp_min

                if infill_temp_max > max_in:
                    max_in = infill_temp_max

        if ylim_bodies is None:
            ylim_bodies = [np.floor(min_in), np.ceil(max_in)]

        ax_bodies.set_title("d) Dyke and SDR Bodies", loc='left')
        ax_bodies.set_xlim(xlim)
        ax_bodies.set_ylim(ylim_bodies)
        ax_bodies.set_ylabel("Depth (km)")
        ax_bodies.set_xlabel("Distance (km)")
        ax_bodies.tick_params(labeltop=False, labelright=True)
        ax_bodies.legend(loc=3)

        # Find maximum/minimum of dyke and SDR anomalies to keep the same
        # y limits
        y_min = 0
        y_max = 0

        for anom in ("dyke_load_sum", "load_subs_sum", "moho_dyke_sum",
                     "load_fill_sum", "extra_fill_sum", "moho_SDR_sum"):
            if np.nanmin(self.sum_grav[anom]) < y_min:
                y_min = np.nanmin(self.sum_grav[anom])
            if np.nanmax(self.sum_grav[anom]) > y_max:
                y_max = np.nanmax(self.sum_grav[anom])

        # Offset slightly
        y_max = y_max + 5
        y_min = y_min - 5

        if ylim_grav is None:
            ylim_grav = [y_min, y_max]

        # Plot dyke gravity
        ax_dyke.plot(x, self.sum_grav["dyke_load_sum"], color='#1f77b4',
                     label='Dyke Load')
        ax_dyke.plot(x, self.sum_grav["load_subs_sum"], color='#2ca02c',
                     label='Basement Subsidence')
        ax_dyke.plot(x, self.sum_grav["moho_dyke_sum"], color='#ff7f0e',
                     label='Moho Subsidence')
        ax_dyke.plot(x, self.sum_grav["dyke_sum"], '--', color='#d62728',
```

```
                    label='Dyke Sum')

        ax_dyke.set_title("a) Dyke Anomalies", loc='left')
        ax_dyke.get_xaxis().set_ticklabels([])
        ax_dyke.set_ylabel("mGal")
        ax_dyke.set_xlim(xlim)
        ax_dyke.set_ylim(ylim_grav)
        ax_dyke.tick_params(labeltop=False, labelright=True)
        ax_dyke.legend(loc=3)

        # Plot SDR Anomalies
        ax_sdr.plot(x, self.sum_grav["load_fill_sum"], color='#1f77b4',
                    label='Infill Load')
        ax_sdr.plot(x, self.sum_grav["extra_fill_sum"], color='#2ca02c',
                    label='Basement Subsidence')
        ax_sdr.plot(x, self.sum_grav["moho_SDR_sum"], color='#ff7f0e',
                    label='Moho Subsidence')
        ax_sdr.plot(x, self.sum_grav["SDR_sum"], '--', color='#d62728',
                    label='SDR Sum')

        ax_sdr.set_title("b) SDR Anomalies", loc='left')
        ax_sdr.get_xaxis().set_ticklabels([])
        ax_sdr.set_ylabel("mGal")
        ax_sdr.set_xlim(xlim)
        ax_sdr.set_ylim(ylim_grav)
        ax_sdr.tick_params(labeltop=False, labelright=True)
        ax_sdr.legend(loc=3)

        # Plot sums
        ax_sum.plot(x, self.sum_grav["overall_sum"], '--')

        ax_sum.set_title("c) Sum Anomaly", loc='left')
        ax_sum.get_xaxis().set_ticklabels([])
        ax_sum.set_ylabel("mGal")
        ax_sum.set_xlim(xlim)
        ax_sum.set_ylim(ylim_grav)
        ax_sum.tick_params(labeltop=False, labelright=True)

        # Shade dyke region on gravity plots
        for i in range(len(dyke_l)):
            ax_sum.axvspan(dyke_l[i], dyke_r[i], alpha=0.3, color='grey')
            ax_dyke.axvspan(dyke_l[i], dyke_r[i], alpha=0.3, color='grey')
            ax_sdr.axvspan(dyke_l[i], dyke_r[i], alpha=0.3, color='grey')

        # Add vertical line at split point
        ax_sum.axvline(x=self.split_point_val - self.x_min, ls='dashdot',
                       color='black', lw=0.5)
        ax_dyke.axvline(x=self.split_point_val - self.x_min, ls='dashdot',
                        color='black', lw=0.5)
        ax_sdr.axvline(x=self.split_point_val - self.x_min, ls='dashdot',
                       color='black', lw=0.5)

        plt.tight_layout()

        # Save figure
        if save is True:
            filename = title + '.pdf'
            plt.savefig(filename, dpi=400, format='pdf')
            plt.close()

        # Return
        return 1

    def plot_pogm_gravonly(self, title, save=True):
        """
        Plot results of POGM, gravity anomalies only

        Inputs:
        title = output filename without extension
        save = bool, whether to save output file
        """

        # Plotting

        # Update plotting parameters
        plt.rcParams.update(plot_params)

        fig = plt.figure()
        ax1 = fig.add_subplot(311)

        # Use x values from stratigraphy plotting
        x = self.sum_grav["xs"] - self.x_min

        ax1.plot(x, self.sum_grav["load_fill_sum"],
```

```
                label="Fill Load Subsidence")
        ax1.plot(x, self.sum_grav["extra_fill_sum"],
                label="Infill Driven Subsidence")
        ax1.plot(x, self.sum_grav["moho_SDR_sum"], label="Moho Flexure")
        ax1.plot(x, self.sum_grav["SDR_sum"], 'r-', lw=2)

        ax1.set_title("SDR Anomalies")
        ax1.set_ylabel("mGal")
        ax1.set_xlim([0, self.x_range])
        ax1.legend(loc='upper left')

        ax2 = fig.add_subplot(312)

        ax2.plot(x, self.sum_grav["dyke_load_sum"], label="Dyke Load")
        ax2.plot(x, self.sum_grav["load_subs_sum"],
                label="Load Driven Subsidence")
        ax2.plot(x, self.sum_grav["moho_dyke_sum"], label="Moho Flexure")
        ax2.plot(x, self.sum_grav["dyke_sum"], 'r-', lw=2)

        ax2.set_title("Dyke Anomalies")
        ax2.set_ylabel("mGal")
        ax2.set_xlim([0, self.x_range])
        ax2.legend(loc='upper left')

        ax3 = fig.add_subplot(313)

        ax3.plot(x, self.sum_grav["overall_sum"], 'r-')
        ax3.set_title("Sum")
        ax3.set_ylabel("mGal")
        ax3.set_xlabel("Distance (km)")
        ax3.set_xlim([0, self.x_range])

        plt.tight_layout()

        # Save figure
        if save is True:
            filename = title + '.pdf'
            plt.savefig(filename, dpi=400, format='pdf')
            plt.close()


class Rift(object):
    def __init__(self, xrift_output, xrift_beta_input, rift_width):
        # Open files and extract profiles
        rift_x = list()
        rift_y = list()

        beta_x = list()
        beta_y = list()

        with open(xrift_output, 'r') as f:
            for line in f:
                rift_x.append(line.split()[0])
                rift_y.append(line.split()[1])

        with open(xrift_beta_input, 'r') as f:
            for line in f:
                beta_x.append(line.split()[0])
                beta_y.append(line.split()[1])

        # Allocate as numpy arrays
        self.rift_x = np.array(rift_x).astype(float)
        self.rift_y = np.array(rift_y).astype(float)

        self.beta_x = np.array(beta_x).astype(float)
        self.beta_y = np.array(beta_y).astype(float)

        # Allocate rift_width variable
        self.rift_width = rift_width  # indicative, used to calc plate size

        # Extract minimum crustal thickness after rifting for post-rift POGM
        self.crust_thick_min = c.zero_ct / np.max(self.beta_y)


class Vector2D(object):
    def __init__(self, x, y):
        # Coordinates of point
        self.x = x
        self.y = y

    def theta(self):
        # Angle in radians
        return np.arctan2(self.y, self.x)

```

```
3200     def length(self):
3201         # Length/magnitude of vector
3202         return np.sqrt(self.x**2 + self.y**2)
```

B.3 Dip Measurement and Analysis

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  import seaborn as sns
4  import os
5  import csv
6  from copy import copy
7  import scipy.stats as stats
8
9  # For export
10 try:
11     import cPickle as pickle
12 except ImportError:
13     import pickle
14
15 # Set matplotlib globals for pdf and font type to edit
16 plot_params = {
17     'pdf.fonttype': 42,
18     'ps.fonttype': 42,
19     'svg.fonttype': 'none',
20     'font.sans-serif': 'Helvetica',
21 }
22
23
24 def cosine_similarity(norm1, norm2):
25     """
26     Calculate the cosine similarity between two normalized vectors
27     """
28
29     cos_sim = np.dot(norm1, norm2) / (np.linalg.norm(norm1) *
30                                       np.linalg.norm(norm2))
31
32     return cos_sim
33
34
35 def split_on_nan(x):
36     """
37     Return list of arrays split on nan's in argument array (numpy 1D)
38     """
39     return [x[s] for s in np.ma.clump_unmasked(np.ma.masked_invalid(x))]
40
41
42 def flatten(l):
43     """
44     Flattens a list of lists
45     """
46     return [item for sublist in l for item in sublist]
47
48
49 def monotonic(x):
50     """
51     Check if array is monotonic increasing or decreasing
52
53     From: http://stackoverflow.com/questions/4983258/
54     python-how-to-check-list-monotonicity
55     """
56
57     dx = np.diff(x)
58     return np.all(dx <= 0) or np.all(dx >= 0)
59
60
61 def dipEst_depth(xp, yp, v, pl=False):
62     """
63     Calculate estimate of SDR dips based on input of points picked along
64     representative SDR, plot against TWT or km with depth.
65
66     Input:
67     x - horizontal position (in km), 1D numpy array, row vector
68     y - vertical position (in TWT/s or km), 1D numpy array, row vector,
```

```
    +ve down
    v - list of representative velocity of SDR packet (in km/s), can be
    vector. If v = 0, assumes data is already depth converted to km.
    pl - set equal to True for plotting, default False

    Output:
    dips - 1d numpy array of dips with variables [velocity, depth]
    rel_depth - 1d numpy array approximate km relative depth at which average
    applies from top of this SDR
    totThick - array of thickness ranges covered by reflector corresponding to
    v; single value in array if data is in depth
    breadth - inline breadth of SDR picked (km)
    """

    # Get rid of NaNs
    xp = xp[ np.isnan(xp)]
    yp = yp[ np.isnan(yp)]

    # Set y such that the first element in x and y is the shallowest i.e. the
    # 'top' of the SDR
    if yp[-1] < yp[0]:
        yp = yp[::-1]
        xp = xp[::-1]

    # Interpolate onto regular grid before windowing, assuming picks capture
    # shape changes but may be sparse in flat areas
    if v != 0:
        y = np.arange(np.min(yp), np.max(yp), 0.1)
    else:
        y = np.arange(np.min(yp), np.max(yp), 0.2)

    x = np.interp(y, yp, xp)

    # Define window for dip calculation in terms of time/depth, not number of
    # points picked using average gap between
    gap = np.abs((np.abs(y[-1]) - y[0]) / y.shape[0])

    # Define window in two ways to ensure it is always calculable - if the SDR
    # is long enough, average over 1/6 a second TWTT or 0.25km. If not,
    # average over fifth the length of the SDR. win is how many picked points
    # fit into the deisred 'gap'

    if v != 0:
        win1 = int(0.16 / gap)
        win2 = int((y[-1] - y[0] / 5) / gap)

        if (win1 > 0) and (win1 < win2):
            win = win1
        else:
            win = win2

    else:
        win1 = int(0.25 / gap)
        win2 = int(((y[-1] - y[0] / 5) / gap))

        if (win1 > 0) and (win1 < win2):
            win = win1
        else:
            win = win2

    # If y is not monotonic, not really an 'SDR', throw it away, return nans
    if not monotonic(y):
        nan_list = np.array([[np.nan]])
        return nan_list, nan_list, np.array([np.nan]), np.nan

    # Convert picked points into 'thicknesses' in each direction
    width = np.empty(x.shape[0] - win)
    tthick = np.empty(x.shape[0] - win)

    for i in range(win, x.shape[0]):
        width[i - win] = np.abs(x[i] - x[i - win])
        tthick[i - win] = np.abs(y[i] - y[i - win])

    # Find corresponding depth in time points for each tthick window
    depth_t = y[win:]
    inc_t = np.zeros(depth_t.shape[0])

    for i in range(1, depth_t.shape[0]):
        inc_t[i] = depth_t[i] - depth_t[i - 1]

    # If already depth converted, calculate velocities
    if v == 0:
        dips = np.empty([1, tthick.shape[0]])
        depth = np.empty([1, tthick.shape[0]])
        rel_depth = np.empty([1, tthick.shape[0]])
```

```
154         totThick = np.empty(1)
155 
156         for j in range(tthick.shape[0]):
157             dips[0, j] = np.arctan(tthick[j] / width[j]) * 180 / np.pi
158             if j > 1:
159                 depth[0, j] = depth[0, j - 1] + inc_t[j]
160             else:
161                 depth[0, j] = depth_t[j]
162 
163             rel_depth = depth - np.min(depth)
164 
165         totThick[0] = y[-1] - y[0]
166 
167     # If not, calculate dips using velocities
168     if v != 0:
169         dips = np.empty([len(v), tthick.shape[0]])
170         depth = np.empty([len(v), tthick.shape[0]])
171         rel_depth = np.empty([len(v), tthick.shape[0]])
172         totThick = np.empty(len(v))
173 
174         for i in range(len(v)):
175             for j in range(tthick.shape[0]):
176                 dips[i, j] = np.arctan(((tthick[j] * v[i]) / 2) / width[j]) \
177                     * 180 / np.pi
178                 if j > 1:
179                     depth[i, j] = depth[i, j - 1] + ((inc_t[j] * v[i]) / 2)
180                 else:
181                     depth[i, j] = (depth_t[j] * v[i]) / 2
182 
183             rel_depth[i, :] = depth[i, :] - np.min(depth[i, :])
184 
185             totThick[i] = (y[-1] - y[0]) * v[i] / 2
186 
187     # Define maximum 'breadth' of SDRs along seismic line, km
188     breadth = np.amax(x) - np.amin(x)
189 
190     # Plot output
191     if pl is True:
192         plt.figure()
193 
194         for i in range(len(v)):
195             plt.plot(dips[i, :], rel_depth[i, :])
196 
197         plt.title("Estimate of Local Dip Along SDR")
198         plt.ylabel("Relative Depth/km")
199         plt.xlabel("Moving Dip Estimate/deg")
200         plt.show()
201 
202     return dips, rel_depth, totThick, breadth
203 
204 
205 def compare_axis_limits(xlim1, ylim1, xlim2, ylim2):
206     """
207     Given two sets of axis limits (tuples), calculate and return a third that
208     encapsulates the range of both inputs
209 
210     Returns xlim, ylim, two tuples
211     """
212 
213     xlim = list()
214     ylim = list()
215 
216     if xlim1[0] <= xlim2[0]:
217         xlim.append(xlim1[0])
218     else:
219         xlim.append(xlim2[0])
220 
221     if ylim1[0] <= ylim2[0]:
222         ylim.append(ylim1[0])
223     else:
224         ylim.append(ylim2[0])
225 
226     if xlim1[1] >= xlim2[1]:
227         xlim.append(xlim1[1])
228     else:
229         xlim.append(xlim2[1])
230 
231     if ylim1[1] >= ylim2[1]:
232         ylim.append(ylim1[1])
233     else:
234         ylim.append(ylim2[1])
235 
236     return tuple(xlim), tuple(ylim)
237 
238 
```

```
def dips_joint_plot(name, depth, dips, depth2=None, dips2=None, name2=None,
                    save=True, **kwargs):
    """
    Plot results of dip calculation using Seaborn JointPlot

    depth = 1D numpy array of relative depth measurements
    dips = 1D numpy array of dip measurements
    name = string, name for file
    save = bool, whether to save files
    **kwargs = any keyword arguments you wish to pass to sns.jointplot
               see the documentation for options

    Optionally provde a second set of depth and dip arrays and a second name
    before **kwargs to make a second plot on the same scales
    """

    # Set styling
    plt.rcParams.update(plot_params)
    sns.set(style="white", color_codes=True)

    # Make first joint plot
    g = sns.jointplot(depth, dips, **kwargs)
    ax = g.ax_joint

    # If second dataset provided, make a second plot where both have the same
    # size and limits for compariso
    if depth2 is not None and dips2 is not None:
        # Get limits
        xlim1 = ax.get_xlim()
        ylim1 = ax.get_ylim()
        plt.close()

        # Make new plot and get second set of limits
        g2 = sns.jointplot(depth2, dips2, color="g", **kwargs)
        ax2 = g2.ax_joint
        xlim2 = ax2.get_xlim()
        ylim2 = ax2.get_ylim()

        # Now compare limits
        xlim, ylim = compare_axis_limits(xlim1, ylim1, xlim2, ylim2)
        print(xlim, ylim)

        # Modify and save plot2
        ax2.set_xlim(xlim)
        ax2.set_ylim(ylim)
        ax2.set_xlabel('Relative Depth (km)')
        ax2.set_ylabel('Dips (degrees)')
        ax2.set_title(name2)

        # Save to file
        if save is True:
            FigStr = "%s_dips_depth_joint_plot.pdf" % name2
            g2.savefig(FigStr, format='pdf')

        # Remake first plot with correct limits
        g = sns.jointplot(depth, dips, xlim=xlim, ylim=ylim, **kwargs)
        ax = g.ax_joint

    # Add axis labels
    ax.set_xlabel('Relative Depth (km)')
    ax.set_ylabel('Dips (degrees)')
    ax.set_title(name)

    # Save to file
    if save is True:
        FigStr = "%s_dips_depth_joint_plot.pdf" % name
        g.savefig(FigStr, format='pdf')


def dips_histogram(paper, dips, labels=None, stack=False, save=True):
    """
    Plot dip results as a simple histogram

    paper = string, name of paper/model run to prepend to output file
    labels = list of strings to label each dataset in corresponding dips list
    dips = list of 1d arrays, or single 1d array, containing dip data
    stack = bool, whether to stack multiple dips datasets
    save = bool, whether to save figure to file
    """

    # Set number of bins
    num_bins = 50

    # Initiate axes
    fig, ax = plt.subplots()
```

```
    ax.grid()

    # the histogram of the data
    if stack is False:
        ax.hist(dips, num_bins)
    else:
        ax.hist(dips, num_bins, histtype='bar',
                                    stacked=True, label=labels)
        ax.legend(prop={'size': 10})

    # Label
    ax.set_xlabel('Dips')
    ax.set_ylabel('Frequency')

    # Tweak spacing
    fig.tight_layout()

    # Save
    if save is True:
        histFigStr = "%s_dips_histogram.pdf" % paper
        fig.savefig(histFigStr, format='pdf')
        plt.close()


def thickness_boxplot(paper, thicks, save=True):
    """
    Plot boxplot of SDR thicknesses

    paper = string, name of paper/model run to prepend to output file
    thicks = 1d numpy array of measured thicknesses
    save = bool, whether to save figure to file
    """

    # Create a figure instance
    fig = plt.figure()

    # Create boxplot
    ax = fig.add_subplot(111)
    ax.grid(axis='y')
    ax.boxplot(thicks)

    # Add labels
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

    ax.set_xlabel('SDR')
    ax.set_ylabel('Thickness (km)')

    # Save the figure
    if save is True:
        box_name = paper + '_thickness_boxplot.pdf'
        fig.savefig(box_name, format='pdf')
        plt.close()


def sdr_stats(paper, thicks, dips, save=True):
    """
    Save small csv file with pertinent statistcis about SDR measurements

    paper = string, name of paper/model run to prepend to output file
    thicks = 1d numpy array of measured thicknesses
    dips = 1d numpy array of measured dips
    save = bool, whether to save to file
    """

    # Flatten thickness list of lists into one list for datasets with combined
    # TWTT and depth converted
    thicks = [item for sublist in thicks for item in sublist]

    # Calculate for dips
    dips_mean = np.around(np.mean(dips), decimals=3)
    dips_median = np.around(np.median(dips), decimals=3)
    dips_std = np.around(np.std(dips), decimals=3)
    dips_min = np.around(np.min(dips), decimals=3)
    dips_max = np.around(np.max(dips), decimals=3)

    # Calculate for thicknesses
    thick_mean = np.around(np.mean(thicks), decimals=3)
    thick_median = np.around(np.median(thicks), decimals=3)
    thick_std = np.around(np.std(thicks), decimals=3)
    thick_min = np.around(np.min(thicks), decimals=3)
    thick_max = np.around(np.max(thicks), decimals=3)

    dip_headings = ['Min dip (deg)', 'Max dip (deg)', 'Mean dip (deg)',
                    'Median dip (deg)', 'Std dev dip']
```

```
    dip_numbers = [dips_min, dips_max, dips_mean, dips_median, dips_std]
    thickness_headings = ['Min thickness (km)', 'Max thickness (km)',
                          'Mean thickness (km)', 'Median thickness (km)',
                          'Std dev thickness']
    thickness_numbers = [thick_min, thick_max, thick_mean, thick_median,
                         thick_std]

    # Write to file
    if save is True:
        filename = paper + '_stats.csv'
        with open(filename, 'w', newline='') as csv_file:
            writer = csv.writer(csv_file, delimiter=',')
            writer.writerow(dip_headings)
            writer.writerow(dip_numbers)
            writer.writerow(thickness_headings)
            writer.writerow(thickness_numbers)
    # Or print to screen
    else:
        print(dip_headings)
        print(dip_numbers)
        print(thickness_numbers)
        print(thickness_numbers)

    return 1


def dips_from_strat(stratigraphy, top_dykes, title, x_vector, rift_top=None, save=True,
                    **kwargs):
    """
    Given stratigraphy, calculate dips and corresponding depths

    stratigraphy = 2D numpy array, timeslice of stratigraphy
    top_dykes = 2D numpy array corresponding to strat which contains
    corresponding dyke tops, to be eliminated from calculation
    title = string to prepend to title of saved plot
    x_vector = 1D numpy array, x positions for stratigraphy
    rift_top = rift curve, to eliminate this from calculation
    save = bool, whether to save outputs, default True
    **kwargs = any keyword arguments you wish to pass to sns.jointplot

    Returns dictionary of dips, depths, thicknesses and breadths
    """

    # Explicitly copy input vars so processsing doesn't modify their external
    # value
    strat = copy(stratigraphy)
    dykes = copy(top_dykes)

    # If it's close to zero, set value in strat to be nan TODO: check magnitude
    if rift_top is None:
        for i in range(strat.shape[0]):
            for j in range(strat.shape[1]):
                if strat[i, j] > -20:
                    strat[i, j] = np.nan

    # Extract side of dyke near vertical contact using top_dyke values
    td_nonnan = list()
    for i in range(strat.shape[1]):
        inds = list()
        for j in range(strat.shape[0]):
            if not np.isnan(dykes[j, i]):
                inds.append(j)

        # Split into groups of consecutive values
        split_inds = np.split(np.array(inds),
                              np.where(np.diff(np.array(inds)) > 1)[0] + 1)

        td_nonnan.append(split_inds)

    # For each group of indices extracted, establish whether this lies on the
    # left or the right of the correpsonding dyke body
    side_orientation = list()
    for i in range(strat.shape[1] - 1):
        o = list()
        for j in range(len(td_nonnan[i])):
            if strat[td_nonnan[i][j][0], i] > strat[td_nonnan[i][j][1], i]:
                o.append('L')
            elif strat[td_nonnan[i][j][0], i] < strat[td_nonnan[i][j][1], i]:
                o.append('R')
        side_orientation.append(o)

    # If actually the side of dyke surface, also set to nan
    # Loop through the non-nan indices of top_dyke stored in td_nonnan and
```

```
    # calculate, for each group of dykes, the values corresonding to the steep
    # sides in each case which shouldn't be included in the dyke calculation
    side_dyke = list()
    for i in range(strat.shape[1]):
        curr_sides = list()
        # Deal with final top load
        if i == strat.shape[1] - 1:
            curr_sides.append(td_nonnan[i][0].tolist())
        # Deal with penultimate load
        elif i == strat.shape[1] - 2:
            for j in range(len(td_nonnan[i])):
                if side_orientation[i][j] == 'L':
                    r_start = td_nonnan[i][j][-1]
                    r_stop = td_nonnan[i + 1][int((j + 1) / 2)][0]
                elif side_orientation[i][j] == 'R':
                    r_stop = td_nonnan[i][j][0]
                    r_start = td_nonnan[i + 1][int(j / 2)][-1]
                curr_sides.append(list(range(r_start, r_stop + 1)))
        # All other loads
        else:
            for j in range(len(td_nonnan[i])):
                if side_orientation[i][j] == 'L':
                    r_start = td_nonnan[i][j][-1]
                    r_stop = td_nonnan[i + 1][j][0]
                elif side_orientation[i][j] == 'R':
                    r_stop = td_nonnan[i][j][0]
                    r_start = td_nonnan[i + 1][j][-1]
                curr_sides.append(list(range(r_start, r_stop + 1)))

        side_dyke.append(curr_sides)

    # Make side_dyke points nan, for this layer and all layers beneath which
    # 'wrap' upwards
    for i in range(strat.shape[1]):
        for j in range(strat.shape[0]):
            if j in flatten(side_dyke[i]):
                strat[j, :i + 1] = np.nan

    # If it's close to the rift top (equiv. to zero), set to nan
    if rift_top is not None:
        for i in range(strat.shape[0]):
            for j in range(strat.shape[1]):
                if abs(strat[i, j] - rift_top[i]) < 20:
                    strat[i, j] = np.nan

    # Prepare summary lists for output
    all_breadth = list()
    all_thick = list()

    # Extract x, y to pass to dip calculation
    for i in range(strat.shape[1]):
        # Split on nan's, which will isolate SDRs on either side of dyke
        sdr_y_list = split_on_nan(strat[:, i])
        for y in sdr_y_list:
            if len(y) > 1:
                # As x is only used in relative terms, generate x vector of
                # correct length, as spacing is constant
                x = x_vector[:y.shape[0]]

                # Calculate dips (pass x, y in km)
                dips, depth, thick, breadth = dipEst_depth(x / 1000,
                                                           -y / 1000, 0)

                # For those which have passed the monotonic test...
                # Test if we have all_ variables yet
                if not np.isnan(thick[0]):
                    try:
                        all_dips = np.append(all_dips, dips.flatten())
                        all_depths = np.append(all_depths, depth.flatten())
                    except NameError:
                        all_dips = dips.flatten()
                        all_depths = depth.flatten()

                    all_thick.append(thick.tolist())
                    all_breadth.append(breadth)

    # Plot dip and depth results
    dips_joint_plot(title, all_depths, all_dips, save=save, **kwargs)

    # Plot thicknesses
    thickness_boxplot(title, all_thick, save=save)

    # Plot histogram of dips
    all_dips_nonan = all_dips[ np.isnan(all_dips)]
```

```
    dips_histogram(title, all_dips_nonan, save=save)

    # Calculate stats
    sdr_stats(title, all_thick, all_dips_nonan, save=save)

    # Pickle data for later use
    if save is True:
        file = title + "_dips.p"
        export_dict = {'dips': all_dips, 'depths': all_depths, 'thick': all_thick,
                       'breadth': all_breadth}
        pickle.dump(export_dict, open(file, "wb"), protocol=2)

    return export_dict


def dips_from_picks(directory, name, save=True, **kwargs):
    """
    Calculate dips from manual picks e.g. GraphClick in a given directory

    Export picks x, y no headers, to folder

    directory = path to folder which contains exported pick txt files;
    includes subdirectories
    name = root title to use for plots
    save = bool, whether to save plots or not
    **kwargs = any keyword arguments you wish to pass to sns.jointplot

    If input text file ends with *depth.txt, assumes data in depth (km).
    Otherwise default behaviour is for TWTT (seconds). Only reads text files
    with filenames that begin with 'sdr' (case insensitive)

    Output files are saved to the given directory

    Returns:
    all_dips, all_depths: 1D numpy arrays
    all_thick: list of lists
    all_breadth: list
    """

    # Get current directory
    cur_dir = os.getcwd()

    try:
        # Change directory to out_dir
        os.chdir(directory)

        # Prepare summary list for output
        all_breadth = list()
        all_thick = list()

        # Recursively look through all subdirectories for txt files
        for root, dirs, files in os.walk(directory):
            for file in files:
                if (file.lower().endswith(".txt") and
                        file.lower().startswith("sdr")):
                    with open(os.path.join(root, file)) as infile:
                        x = list()
                        y = list()
                        for line in infile:
                            cols = line.split()
                            x.append(float(cols[0]))
                            y.append(float(cols[1]))

                        # Check for depth flag, if so set v to 0
                        if file.lower().endswith("depth.txt"):
                            v = 0
                        else:
                            v = [3, 4, 5]

                        # Calculate dips
                        dips, depth, totThick, breadth = \
                            dipEst_depth(np.array(x), np.array(y), v)

                        # For those which have passed the monotonic test...
                        # Test if we have all_ variables yet
                        if not np.isnan(totThick[0]):
                            try:
                                all_dips = np.append(all_dips, dips.flatten())
                                all_depths = np.append(all_depths,
                                                       depth.flatten())
                            except NameError:
                                all_dips = dips.flatten()
                                all_depths = depth.flatten()

                            all_thick.append(totThick.tolist())
```

```
                            all_breadth.append(breadth)

                        # Move to next file
                        continue
                    else:
                        continue

            # Plot dip and depth results
            dips_joint_plot(name, all_depths, all_dips, save=save, **kwargs)

            # Plot thicknesses
            thickness_boxplot(name, all_thick, save=save)

            # Plot histogram of dips
            all_dips_nonan = all_dips[ np.isnan(all_dips)]
            dips_histogram(name, all_dips_nonan, save=save)

            # Calculate stats
            sdr_stats(name, all_thick, all_dips_nonan, save=save)

            # Reshape numpy arrays for explicit second dimension
            all_dips.shape = (all_dips.size, 1)
            all_depths.shape = (all_depths.size, 1)

        except Exception as e:
            raise e

        finally:
            # Move back to calling directory
            os.chdir(cur_dir)

        return all_dips, all_depths, all_thick, all_breadth


def extract_dips_depths(input_dips):
    """
    Given a path to observed dips, or a dictionary containing the output of
    modelled dips, extract the dip vector only and return

    Input options for input_dips1 and input_dips2:
    For observed data - str, path to observed directory of files, passed to
                        dips_from_picks
    For modelled data - dict, result of dips_from_strat

    Returns a (n, 1) numpy array of dips where n is the number of measurements
    """

    if type(input_dips) is str:
        # Calculate observed data
        obs_path_split = input_dips.split('/')
        obs_name = obs_path_split[-1]
        dips, depths, _, _ = dips_from_picks(input_dips, obs_name, save=False)
    elif type(input_dips) is dict:
        # Get calculated data (output from dips_from_strat)
        dips = input_dips['dips']
        dips = dips.reshape(dips.shape[0], -1)
        depths = input_dips['depths']
        depths = depths.reshape(depths.shape[0], -1)

    return dips, depths


def compare_dips(input_dips1, input_dips2, name_dips1='Observed',
                 name_dips2='Modelled', save=True):
    """
    Given a folder of observed SDRs, compare to modelled dip distribution

    Input options for input_dips1 and input_dips2:
    For observed data - str, path to observed directory of files, passed to
                        dips_from_picks
    For modelled data - dict, result of dips_from_strat

    Input for name_dips1 and name_dips1:
    Optional strings to label/name output files, defaults to observed and
    modelled if not provided

    save = bool, whether to save output files

    Outputs: (saved in current directory)
    *_vs_*_dips_histogram.pdf - figure, normed histogram, where * are names
    given above
    dip_fit_stats.csv - simple csv file containing KS and cosine similarity

    """

    dips1, depths1 = extract_dips_depths(input_dips1)
```

```
    dips2, depths2 = extract_dips_depths(input_dips2)

    fig = plt.figure()
    plt.rcParams.update(plot_params)
    ax = fig.add_subplot(111)
    n, _, _ = ax.hist([dips2, dips1], stacked=False, histtype='bar', normed=True,
                      label=[name_dips2, name_dips1])
    ax.legend(prop={'size': 12})
    ax.set_xlabel('Dips (degrees)')
    ax.set_ylabel('Normalized Frequency')
    fig.tight_layout()

    # Calculate the Kolmogorov-Smirnov statistic on 2 samples
    D, p = stats.ks_2samp(dips1[:, 0], dips2[:, 0])

    # Calculate cosine similarity between normalized vectors from historgam
    cos_sim = cosine_similarity(n[1], n[0])

    # Save statistics
    if save is True:
        filename = 'dip_fit_stats.csv'
        with open(filename, 'w', newline='') as csv_file:
            writer = csv.writer(csv_file, delimiter=',')
            writer.writerow(['Kolmogorov-Smirnov statistic', D, p])
            writer.writerow(['Cosine similarity', cos_sim])
    else:
        print('Kolmogorov-Smirnov statistic', D, p)
        print('Cosine similarity', cos_sim)

    # Save figure
    if save is True:
        figure_name_hist = name_dips1 + "_vs_" + name_dips2 + "_dips_histogram.pdf"
        plt.savefig(figure_name_hist, dpi=400, format='pdf')

    # Also make joint plot
    dips_joint_plot(name_dips1, depths1, dips1, depth2=depths2,
                    dips2=dips2, name2=name_dips2, save=save)

    return
```

B.4 SEGY Plotting (using ObsPy)

```
import numpy as np
from obspy.io.segy.core import _read_segy
import matplotlib.pyplot as plt
from matplotlib.figure import figaspect
from utm import to_latlon
from os import path
from copy import deepcopy as copy


def segy_process(filename, out_path, utm_number=None, utm_letter=None):
    """
    Read in SEGY file with ObsPy and process data.

    Gets data encoding and endianess from the first trace. Assumes header and
    trace data have the same endianess.

    Set utm_number and utm_letter optionally to dump nav as lat/lon.
    This information is often given in the file header if not known.

    Returns ObsPy Stream object.
    """

    # Create file names
    segy_file = path.splitext(path.basename(filename))
    segy_name = segy_file[0]

    head_name = segy_name + "_header.txt"
    header_file = path.join(out_path, head_name)

    bin_name = segy_name + "_binary.txt"
    binary_file = path.join(out_path, bin_name)

```

```
    # Read in the file
    stream = _read_segy(filename, unpack_trace_headers=True)

    # Get headers
    txt = np.array(list(stream.stats.textual_file_header.decode()))

    with open(header_file, "w") as h_file:
        h_file.write("\n".join("".join(row) for row in txt.reshape((40, 80))))

    with open(binary_file, "w") as b_file:
        b_file.write(str(stream.stats.binary_file_header._pretty_str()))

    # Extract and save geom
    if utm_number is not None and utm_letter is not None:
        lat, lon, trace = get_geom(stream, out_path, segy_name, utm_number, ↩
    utm_letter)

    return stream


def get_geom(
    stream,
    out_path,
    file_name,
    utm_number,
    utm_letter,
    x_header="source_coordinate_x",
    y_header="source_coordinate_y",
    trace_header="ensemble_number",
):
    """
    From stream of SEGY data, extract, return and save geometry info
    """

    # Extract coordinates for each trace
    lat = list()
    lon = list()
    trace = list()

    for t in stream:
        (y, x) = to_latlon(
            t.stats.segy.trace_header[x_header] / 100,
            t.stats.segy.trace_header[y_header] / 100,
            utm_number,
            utm_letter,
        )
        lat.append(x)
        lon.append(y)
        trace.append(t.stats.segy.trace_header[trace_header])

    # Save file
    nav_name = file_name + "_nav.txt"
    nav_file = path.join(out_path, nav_name)

    with open(nav_file, "w") as n_file:
        for i in range(len(trace)):
            n_file.write("{}\t{}\t{}\n".format(trace[i], lat[i], lon[i]))

    return lat, lon, trace


def plot_segy(
    stream,
    cdp_spacing,
    out_path,
    km_tick_spacing,
    trace_label,
    plot_name,
    min_time=None,
    max_time=None,
    min_trace=None,
    max_trace=None,
    aspect=None,
    fig_width_inches=15,
    flip=False,
    trace_header="ensemble_number",
    percentile=99
):
    """
    Produce nice plot of SEGY data with many controlling options

    stream = ObsPy stream from _read_segy
    cdp_spacing = CDP/trace spacing in metres (for axis/aspect calculation)
    out_path = desired path for output file
    km_tick_spacing = int, number of kilometres to put tick labels
    trace_label = str, label for x axis e.g. "CDP"
```

```
    plot_name = str, stem of filename (no extension)
    min and max_time = ints, time limits in ms
    min and max_trace = ints, trace limits in trace number
    aspect = scalar, aspect ratio of plot in km(x)/s(y)
    fig_width_inches = fixed width of figure in inches, y varies with aspect
    flip = bool, flip l/r if you wish to change seaward side
    trace_header = str, name of attribute in trace header which represents
    trace increment
    percentile = int, percent amplitude for plotting (default 99)
    """

    # Apply limits if desired; copy stream to manipulate
    plot_stream = copy(stream)

    if min_trace is not None and max_trace is not None:
        # Adjust dataset
        plot_stream = modify_traces(plot_stream, min_trace, max_trace)

    if min_time is not None and max_time is not None:
        # Adjust dataset
        plot_stream = modify_time_axis(plot_stream, min_time, max_time)

    # Read file
    data = np.stack(t.data for t in plot_stream.traces)

    # Find nth percentile amplitude for plotting
    vm = np.percentile(data, percentile)

    # Set var names
    ps_name = plot_name + ".pdf"
    plot_file = path.join(out_path, ps_name)

    # Get y limits
    srate_ms = plot_stream[0].stats.segy.trace_header[
        "sample_interval_in_ms_for_this_trace"
    ]

    # Assumes all same length, may be modified from header for plotting
    num_samples = plot_stream[0].data.shape[0]

    # If not none, already set and stream modified accordingly
    if max_time is None:
        max_time = (srate_ms * num_samples) / 1000
    if min_time is None:
        min_time = 0

    # Get trace x limits
    if flip is True:
        r_trace = plot_stream[0].stats.segy.trace_header[trace_header]
        l_trace = plot_stream[-1].stats.segy.trace_header[trace_header]
    elif flip is False:
        r_trace = plot_stream[-1].stats.segy.trace_header[trace_header]
        l_trace = plot_stream[0].stats.segy.trace_header[trace_header]

    # Extract aspect variable and set
    min_trace_plot = min(r_trace, l_trace)
    max_trace_plot = max(r_trace, l_trace)
    trace_range = max_trace_plot - min_trace_plot

    if aspect is not None:
        aspect = aspect * (1000 / 12.5)
        fig_aspect = aspect * ((max_time - min_time) / 1000) / trace_range

        # Scale to fig_width_inches wide
        w, h = figaspect(fig_aspect)
        scale = fig_width_inches / w
        figsize = (fig_width_inches, h * scale)
    else:
        # Without aspect ratio set figsize semi-arbitrarily
        aspect = "auto"
        figsize = (fig_width_inches, fig_width_inches * 3 / 4)

    # Set up figure
    fig = plt.figure(figsize=figsize)
    ax1 = fig.add_subplot(111)
    ax2 = ax1.twiny()

    # Flip data left-right if required
    if flip is True:
        data = np.fliplr(data.T)
    else:
        data = data.T

    # Plot SEGY
    ax1.imshow(
```

```
        data,
        cmap="RdBu",
        vmin=-vm,
        vmax=vm,
        aspect=aspect,
        extent=[l_trace, r_trace, -max_time / 1000, -min_time / 1000],
    )

    # Set labels - with absolute value in seconds for y
    ax1.set_yticklabels(["%d" % abs(x) for x in ax1.get_yticks()])
    ax1.set_ylabel("IWIT (s)")
    ax1.set_xlabel(trace_label)

    # Add distance axis
    cdp_tick_spacing = km_tick_spacing / (cdp_spacing / 1000)
    new_tick_locations = np.arange(0, trace_range, cdp_tick_spacing)

    ax2.set_xlim(0, trace_range)
    ax2.set_xticks(new_tick_locations)
    ax2.set_xticklabels(tick_function(new_tick_locations, cdp_spacing))
    ax2.set_xlabel("Distance (km)")

    # Save as 400dpi pdf with transparency
    plt.savefig(plot_file, dpi=400, transparent=True, format="pdf")


def tick_function(x, cdp_spacing):
    """
    Generates km axis from cdp numbers given spacing
    """

    v = [i * cdp_spacing / 1000 for i in x]

    return ["%d" % i for i in v]


def modify_time_axis(stream, min_time, max_time):
    """
    Modifies data in stream based on min/max time in ms
    """

    srate_ms = ↩
    stream[0].stats.segy.trace_header["sample_interval_in_ms_for_this_trace"]
    num_samples = stream[0].data.shape[0]

    time_arr = np.arange(0, num_samples) * srate_ms / 1000
    min_idx = (np.abs(time_arr - min_time)).argmin()
    max_idx = (np.abs(time_arr - max_time)).argmin()

    for t in stream:
        t.data = t.data[min_idx : max_idx + 1]

    return stream


def modify_traces(stream, min_trace, max_trace, trace_header="ensemble_number"):
    """
    Modifies data in stream based on min/max trace number in file
    """

    for t in stream:
        number = t.stats.segy.trace_header[trace_header]
        if number < min_trace:
            stream.remove(t)
        if number > max_trace:
            stream.remove(t)

    return stream


def conv_coord_to_trace(coord1, coord2, coords, traces):
    """
    Given two lats or two lons, determine corresponding min/max trace number.

    coord1, coord2 = floats, lat OR lon coordinates
    coords, traces = lists to match to
    """

    coords = np.array(coords)

    idx1 = (np.abs(coords - coord1)).argmin()
    idx2 = (np.abs(coords - coord2)).argmin()

    min_trace = min(traces[idx1], traces[idx2])
    max_trace = max(traces[idx1], traces[idx2])
```

```
287
288     return min_trace, max_trace
```

B.5 Synthetic Seismic Reflection

```
1  from __future__ import print_function
2  import os
3  import numpy as np
4  import matplotlib.pyplot as plt
5  import matplotlib.gridspec as gridspec
6  from fatiando.seismic import conv
7
8  try:
9      import cPickle as pickle
10 except ImportError:
11     import pickle
12
13
14 def density_to_vp(rho):
15     """
16     Calculates Vp in m/sec given rho in kg/m^3 using Brocher 2005 eq. 5
17
18     rho = single value, or list of scalars
19
20     returns vel in m/s as list or scalar, depending on input
21     """
22
23     # Convert rho to g/cm^3 for scalar or list
24     if type(rho) is int or type(rho) is float:
25         rho = float(rho) / 1000
26
27         vel = int((39.128 * rho - 63.064 * np.power(rho, 2) + 37.083 *
28                    np.power(rho, 3) - 9.1819 * np.power(rho, 4) + 0.8228 *
29                    np.power(rho, 5)) * 1000)
30
31     elif type(rho) is list:
32         rho[:] = [float(x) / 1000 for x in rho]
33         rho = np.asarray(rho)
34
35         vel = 39.128 * rho - 63.064 * np.power(rho, 2) + 37.083 * \
36             np.power(rho, 3) - 9.1819 * np.power(rho, 4) + 0.8228 * \
37             np.power(rho, 5)
38
39         vel.tolist()
40         vel = [int(1000 * i) for i in vel]
41
42     return vel
43
44
45 def seismic_wiggle(section, dt=0.004, ranges=None, scale=1.,
46                    color='k', normalize=False):
47     """
48     Taken from fatiando as soon to be deprecated.
49     Plot a seismic section (numpy 2D array matrix) as wiggles.
50
51     Parameters:
52
53     * section :  2D array
54         matrix of traces (first dimension time, second dimension traces)
55     * dt : float
56         sample rate in seconds (default 4 ms)
57     * ranges : (x1, x2)
58         min and max horizontal values (default trace number)
59     * scale : float
60         scale factor multiplied by the section values before plotting
61     * color : tuple of strings
62         Color for filling the wiggle, positive  and negative lobes.
63     * normalize :
64         True to normalizes all trace in the section using global max/min
65         data will be in the range (-0.5, 0.5) zero centered
66
67     .. warning::
68         Slow for more than 200 traces, in this case decimate your
69         data or use ``seismic_image``.
```

```
    """
    npts, ntraces = section.shape  # time/traces
    if ntraces < 1:
        raise IndexError("Nothing to plot")
    if npts < 1:
        raise IndexError("Nothing to plot")
    t = np.linspace(0, dt * npts, npts)
    amp = 1.  # normalization factor
    gmin = 0.  # global minimum
    toffset = 0.  # offset in time to make 0 centered
    if normalize:
        gmax = section.max()
        gmin = section.min()
        amp = (gmax - gmin)
        toffset = 0.5
    plt.ylim(max(t), 0)
    if ranges is None:
        ranges = (0, ntraces)
    x0, x1 = ranges
    # horizontal increment
    dx = float((x1 - x0) / ntraces)
    plt.xlim(x0, x1)
    for i, trace in enumerate(section.transpose()):
        tr = (((trace - gmin) / amp) - toffset) * scale * dx
        x = x0 + i * dx  # x positon for this trace
        plt.plot(x + tr, t, 'k')
        plt.fill_betweenx(t, x + tr, x, tr > 0, color=color)


def seismic_image(section, dt=0.004, ranges=None, cmap=plt.cm.gray,
                  aspect=None, vmin=None, vmax=None):
    """
    Modified from fatiando as soon to be deprecated.
    Plot a seismic section (numpy 2D array matrix) as an image.

    Parameters:

    * section :  2D array
        matrix of traces (first dimension time, second dimension traces)
    * dt : float
        sample rate in seconds (default 4 ms)
    * ranges : (x1, x2)
        min and max horizontal values (default trace number)
    * cmap : colormap
        color map to be used. (see pyplot.cm module)
    * aspect : float
        matplotlib imshow aspect parameter, ratio between axes
    * vmin, vmax : float
        min and max values for imshow

    """
    npts, maxtraces = section.shape  # time/traces
    if maxtraces < 1:
        raise IndexError("Nothing to plot")
    if npts < 1:
        raise IndexError("Nothing to plot")
    t = np.linspace(0, dt * npts, npts)
    data = section
    if ranges is None:
        ranges = (0, maxtraces)
    x0, x1 = ranges
    extent = (x0, x1, t[-1:], t[0])
    if aspect is None:  # guarantee a rectangular picture
        aspect = np.round((x1 - x0) / np.max(t))
        aspect -= aspect * 0.2
    plot = plt.imshow(data, cmap=cmap, origin='upper', aspect=aspect,
                      extent=extent, vmin=vmin, vmax=vmax)

    return plot

# Define file to load
directory = "/Users/rebeccamorgan/Drive/University/" \
            "SyntheticSeismic/"
file = "strat.p"

# Name output file
outfile = "/Users/rebeccamorgan/Drive/University/SyntheticSeismic/" \
    "synth_seismic.pdf"

# Define trace spacing in metres
tr_sp = 15

# Set maximum depth to resolve in metres
max_depth = 6000
```

```
155 |
156 |# Plot every x traces
157 |plot_tr = 200
158 |
159 |# Additional arguments for seismic_image
160 |dt = 2.e-3
161 |plot_args = {'cmap': "RdBu", 'vmin': -0.05, 'vmax': 0.05}
162 |
163 |# ----------------------------------------------------------------------
164 |# Set matplotlib globals for pdf and font type to edit
165 |plot_params = {
166 |    'pdf.fonttype': 42,
167 |    'ps.fonttype': 42,
168 |    'svg.fonttype': 'none',
169 |    'font.sans-serif': 'Helvetica',
170 |}
171 |
172 |# Load stratigraphic dictionary
173 |print("Load stratigraphy")
174 |file_path = os.path.join(directory, file)
175 |model_out = pickle.load(open(file_path, "rb"))
176 |
177 |# Extract variables from dictionary
178 |strat = model_out["strat_horizons"]
179 |top_dyke = model_out["top_dyke"]
180 |infill_rho = model_out["infill_rho"]
181 |load_rho = model_out["load_rho"][0]
182 |x_plate = model_out["x_plate"]
183 |
184 |# Extract jump positions, shift to zero indexing
185 |jump_pos = [i[0] - 1 for i in model_out["jumps"]]
186 |
187 |# Calculate Vp velocities using Brocher relationship
188 |infill_vel = density_to_vp(infill_rho)
189 |load_vel = density_to_vp(load_rho)
190 |crust_vel = density_to_vp(model_out['c.rho_crust'])
191 |
192 |# --------- Build up rho/velocity structure using horizons --------------------
193 |print("Calculate density and velocity structure")
194 |
195 |# Trim down strat to get only relevant bit of stratigraphy with SDRs, and not
196 |# the rest of the padded out plate, find relevant x indices
197 |x_ind = [i for i in range(strat[:, 0].shape[0]) if strat[i, 0] < -50]
198 |
199 |# Resample x_ind between first and last values to include potential jumps
200 |x_ind = range(x_ind[0], x_ind[-1])
201 |
202 |# Set up grid for density and velocity in depth structure
203 |length = (len(x_ind) - 1) * model_out["x_spacing"]
204 |trs_x = np.arange(0, length + 1, tr_sp)
205 |n_trs = trs_x.shape[0]
206 |
207 |# Set up basic velocity and density grids with background value
208 |velocity = crust_vel * np.ones((max_depth, n_trs))
209 |rho = model_out['c.rho_crust'] * np.ones(np.shape(velocity))
210 |
211 |# Interpolate strat onto same spacing as velocity/density determined by trace
212 |# spacing
213 |strat_trs = np.empty((n_trs, strat.shape[1])) * np.nan
214 |top_dyke_trs = np.empty((n_trs, top_dyke.shape[1])) * np.nan
215 |
216 |for i in range(strat.shape[1]):
217 |    # Can index x_plate from the start as steps are all relative
218 |    strat_trs[:, i] = np.interp(trs_x, x_plate[:len(x_ind)], strat[x_ind, i])
219 |    top_dyke_trs[:, i] = np.interp(trs_x, x_plate[:len(x_ind)],
220 |                                   top_dyke[x_ind, i])
221 |
222 |# Loop from base of stratigraphy upwards updating velocity and density within
223 |# each package
224 |for i in range(strat_trs.shape[1]):
225 |    for j in range(strat_trs.shape[0]):
226 |        if not np.isnan(strat_trs[j, i]):
227 |            base = -int(np.round(strat_trs[j, i]))
228 |            # Handle edge case of top package
229 |            if i == strat_trs.shape[1] - 1:
230 |                top = 0
231 |            else:
232 |                top = -int(np.round(strat_trs[j, i + 1]))
233 |
234 |            # Assuming rho/velocity grids have 1m depth increments as defined
235 |            rho[top:base, j] = int(infill_rho[i])
236 |            velocity[top:base, j] = int(infill_vel[i])
237 |
238 |# Extract 'sides' of SDR packages by interpolating between sequential top_dyke
239 |# points in order to set density contrast to load
```

```
side_dyke_trs = np.empty((n_trs, strat.shape[1])) * np.nan

# Build by interpolation between the end points of previous and successive
# top_dyke horizons
ex_l_edge_ind = list()
ex_r_edge_ind = list()
mid_l_edge_ind = list()
mid_r_edge_ind = list()

# For each horizon, find the edges of top_dyke in terms of indices
for i in range(0, strat_trs.shape[1]):
    dyke_zone = np.where( np.isnan(top_dyke[:, i]))[0]
    ex_l_edge_ind.append(dyke_zone[0])
    ex_r_edge_ind.append(dyke_zone[-1])

    # Find 'mid' limits of dyke
    # in jump case, where they are continuous
    if i in jump_pos:
        ind = int(len(dyke_zone) / 2)
        mid_r_edge_ind.append(dyke_zone[ind])
        mid_l_edge_ind.append(dyke_zone[ind])
    # Otherwise, dind break in top_dyke between two halfs were indices are
    # discontinuous
    else:
        for j in range(1, dyke_zone.shape[0]):
            if dyke_zone[j] - dyke_zone[j - 1] != 1:
                mid_r_edge_ind.append(dyke_zone[j])
                mid_l_edge_ind.append(dyke_zone[j - 1])

# Use these indices between horizons to build side and interpolate in the gaps
side_dyke = np.empty(top_dyke.shape) * np.nan

for i in range(0, strat.shape[1] - 1):
    side_dyke[mid_l_edge_ind[i], i] = top_dyke[mid_l_edge_ind[i], i]
    side_dyke[mid_r_edge_ind[i], i] = top_dyke[mid_r_edge_ind[i], i]

    side_dyke[ex_l_edge_ind[i + 1], i] = top_dyke[ex_l_edge_ind[i + 1], i + 1]
    side_dyke[ex_r_edge_ind[i + 1], i] = top_dyke[ex_r_edge_ind[i + 1], i + 1]

    side_dyke_trs[:, i] = np.interp(trs_x, x_plate[:len(x_ind)],
                                    side_dyke[x_ind, i])

# At base of SDR-dyke contact, add dyke density contrast
for i in range(top_dyke_trs.shape[1]):
    for j in range(top_dyke_trs.shape[0]):
        # If not nan, the dyke is present here
        if  np.isnan(top_dyke_trs[j, i]):
            top = -int(np.round(top_dyke_trs[j, i]))
            rho[top:, j] = int(load_rho)
            velocity[top:, j] = int(load_vel)

        # Apply again for sides of dykes
        if  np.isnan(side_dyke_trs[j, i]):
            top = -int(np.round(side_dyke_trs[j, i]))
            rho[top:, j] = int(load_rho)
            velocity[top:, j] = int(load_vel)

# ------------- Seismic Convolution -------------------------------------------
print("Run Seismic Convolution")

# We need to convert the depth model we made above into time
vel_l = conv.depth_2_time(velocity, velocity, dt=2e-3, dz=1)
rho_l = conv.depth_2_time(velocity, rho, dt=2e-3, dz=1)

# With that, we can calculate the reflectivity model in time
rc = conv.reflectivity(vel_l, rho_l)
# and finally perform our convolution
synt = conv.convolutional_model(rc, 30, conv.rickerwave, dt=2e-3)


# Set aspect ratio using trace spacing, equiv. to 3 in km
width_km = n_trs * tr_sp

# Given sample rate, fix aspect ratio to match depth model
npts, maxtraces = synt.shape  # time/traces
max_time = dt * npts
aspect_time_trs = (max_depth / n_trs) * (max_depth / max_time) * (1 / 3)

# ------------------ Plotting ------------------------------------------------
print("Plotting")

plt.rcParams.update(plot_params)

gs = gridspec.GridSpec(2, 1, height_ratios=[1, 1])
```

```
fig = plt.figure()
ax = plt.subplot(gs[0])
ax.set_title("Velocity Model")
tmp = ax.imshow(velocity, extent=[0, width_km / 1000, max_depth / 1000,
                0], cmap="copper", aspect=3, origin='upper')
ax.set_ylabel('Depth (km)')
ax.set_xlabel('Distance (km)')
fig.colorbar(tmp, ax=ax, fraction=0.04, pad=0.04)

ax2 = plt.subplot(gs[1])
ax2.set_title("Synthetic Seismogram")
smp = seismic_image(synt, dt=dt, aspect=aspect_time_trs, **plot_args)
fig.colorbar(smp, ax=ax2, fraction=0.04, pad=0.04)
ax2.set_xlabel('Trace Number')
ax2.set_ylabel('Time (s)')
plt.tight_layout()
plt.savefig(outfile, dpi=400, format='pdf')
```

Appendix C

Matlab Code

C.1 Isostatic Anomaly

```
clear; close all;

% Produce isostatic anomaly grid, given a bathymetry grid and FAA. No
% sediment correction at present

% Zero elevation crustal thickness in m
zero_ct = 31.2 * 1e3;

% Set densities in kg/m^3
rho_c = 2800;   % crust
rho_m = 3330;   % mantle
rho_w = 1030;   % water

% Read in bathy/topo and gravity
[X,Y,FAA] = grdread2('grav.23.1.1min.grd');
[ , ,gebco] = grdread2('GEBCO_2008.grd');

% Read in spherical harmonics, to extract long wavelength signal from FAA
[ , ,spher] = grdread2('degorder20.1m.grd');

% Balance bathy/topo against zero elevation thickness crust, following
% Simpson et al 1986
moho = NaN(size(gebco));

for j = 1:size(gebco,2)
    for i = 1:size(gebco,1)
        if gebco(i,j) < 0
            moho(i,j) = -(zero_ct + gebco(i,j).*((rho_c-rho_w)/(rho_m-rho_c)));
        elseif gebco(i,j) >= 0
            moho(i,j) = -(zero_ct + gebco(i,j).*(rho_c/(rho_m-rho_c)));
        end
    end
end

% Scale topography up to account for the lower density contrast of
% crust/air so I can use a single FFT over top crust (same methodology as
% used in POGM)
bathy = NaN(size(gebco));

for j = 1:size(gebco,2)
    for i = 1:size(gebco,1)
        if gebco(i,j) > 0
            bathy(i,j) = gebco(i,j)*(rho_c/(rho_c-rho_w));
        else
            bathy(i,j) = gebco(i,j);
        end
    end
end

```

```
% Calculate gravitational effect of top of crust, for both submarine and
% subaerial cases
topCrustGrav = grv3dFFT(X,Y,bathy,rho_c,rho_w,4,0);

% Calculate gravitational effect of moho
mohoGrav = grv3dFFT(X,Y,moho,rho_m,rho_c,4,0);

% Calculate and export isostatic anomaly
IA = FAA - spher - mohoGrav - topCrustGrav;
%grdwrite2(X,Y,IA,'airy_isostatic_degorder20.grd')
grdwrite2(X,Y,IA,'IA_anomaly_degorder20_2800_1030_3330_31.2.grd')
```

C.2 Ensemble Averaging

```

clear; close all

% Find orthogonal profiles to picked shelf break, extract isostatic anomaly
% along these profiles, average and plot the results

%% Import data

% Set input definition
use_contour=0;    % 1 to use coastline, 0 to use input track
reg_box=[12, 16, -34, -22]; % for using contour
contour_depth=0; % for using contour

data = importdata('Namibia.txt');


% Output dir
out_dir = "temp/";

% Choose number of profiles
n = 10;

% Is land east or west of the shelf break? Choose 'E' or 'W'
landSide = 'E';

% Choose number of degrees landward/seaward of picks to plot
degS = 4;
degL = 2;

% Use cross correlation or not? Requires signal processing toolbox
correlate_grav = 0;  % 1 for yes, 0 for no
correlate_mag = 0;  % 1 for yes, 0 for no
plot_xcorr = 0; % 1 to plot each xcorr result, 0 to not

% Choose acceptable shift to fit cross correlation in km
shiftKm = 50;

% Plot all lines for every profile or fill range?
plotStyle = 'lines'; % Choose 'lines' or 'fill'

% Adding SDR's?
sdr = 0;    % 1 for yes, 0 for no, set polygon if 1
polygon = importdata('SDRpolygon.txt');

% Adding magnetics? Check adjustment shifts for overlaying
magnetics = 0;  % 1 for yes, 0 for no, set mag_filename if 1
mag_filename = 'mag_calc.pz';
x_mag_adj = 130;
y_mag_adj = -25;

% Save ensemble magnetic profile?
save_mag_calc = 0; % 1 for yes, 0 for no, saves in current directory

% Import Isostatic Anomaly grid
%[X,Y,isoAnom] = grdread2('isostatic_degorder20.grd');
[X,Y,isoAnom] = grdread2('IA_anomaly_degorder20_2800_1030_3330_31.2.grd');
Ria = [60 80.738 -180]; %global
%Ria = [60 -21 5]; % Namibia only

```

```
% Import bathymetry grid as check
[ , ,bathy] = grdread2('GEBCO_2008.grd');
Rbathy = [60 80.738 -180]; %global
%Rbathy = [60 -21 5]; % Namibia only

% Import magnetics grid - EMAG2 has -9999 instead of NaNs, so use
% file cleaned for this purpose
[Xmag,Ymag,mag] = grdread2('EMAG2_V3.grd');
Rmag = [30,80,-180];

%% Find profiles
if use_contour == 0
    [profLat, profLon, distKm, lat, lon, latSm, lonSm] = project_profiles(data(:,2), ↩
    data(:,1), landSide, n, degL, degS, out_dir);
elseif use_contour == 1
    [xgrid, ygrid] = meshgrid(X,Y);
    indices = find(xgrid > reg_box(1) & xgrid < reg_box(2) & ygrid > reg_box(3) & ↩
    ygrid < reg_box(4));

    bathyReg = bathy(indices);
    xReg = xgrid(indices);
    yReg = ygrid(indices);

    contour_ind = find(round(bathyReg) == contour_depth);

    coastline_lon = xReg(contour_ind);
    coastline_lat = yReg(contour_ind);

    [profLat, profLon, distKm, lat, lon, latSm, lonSm] = ↩
    project_profiles(coastline_lat, coastline_lon, landSide, n, degL, degS);
end

% Plot profiles and input track
% Set up the axes
figure
worldmap('World')
gridm on; plabel on; mlabel on
title('Profile Locations')

% Plot shelf break track
meshm(bathy,Rbathy)
plotm(lat,lon,'--g')
plotm(latSm,lonSm,'--b')
plotm(lat,lon,'xk')


for i = 1:n
    plotm(profLat(:,i),profLon(:,i),'r')
end


%% Extract data along profiles

% Extract values from grid using ltln2val
zVal = NaN(size(profLat));

for i = 1:n
    for j = 1:size(profLon,1)
        zVal(j,i) = ltln2val(isoAnom,Ria,profLat(j,i),profLon(j,i));
    end
end

% Repeat for bathymetry
bathyVal = NaN(size(profLat));

for i = 1:n
    for j = 1:size(profLon,1)
        bathyVal(j,i) = ltln2val(bathy,Rbathy,profLat(j,i),profLon(j,i));
    end
end

% Repeat for magnetics
magVal = NaN(size(profLat));

for i = 1:n
    for j = 1:size(profLon,1)
        magVal(j,i) = ltln2val(mag,Rmag,profLat(j,i),profLon(j,i));
    end
end

% For magnetics, interpolate in case of NaN values which break fft
for i = 1:n
    magVal(:,i) = inpaint_nans(magVal(:,i));
end
```

```
141 |
142 | %% Calculate cross correlation and shift on IA
143 |
144 | if correlate_grav == 1
145 |     maxLag = floor(shiftKm/(distKm(2)-distKm(1)));
146 |     [zVal, zLags] = cross_correlate(zVal, maxLag, plot_xcorr);
147 | end
148 |
149 | if correlate_mag == 1
150 |     maxLag = floor(shiftKm/(distKm(2)-distKm(1)));
151 |     [magVal, magLags] = cross_correlate(magVal, maxLag, plot_xcorr);
152 | end
153 | %% Averaging
154 |
155 | % Isostatic Anomaly
156 | [zMean, ZMean] = calc_averages(zVal);
157 |
158 | % Bathymetry
159 | [bathyMean, BathyMean] = calc_averages(bathyVal);
160 |
161 | % Magnetics
162 | [magMean, MagMean] = calc_averages(magVal);
163 |
164 | %% IF SDR polygon available, define points on profiles with SDR
165 |
166 | if sdr == 1
167 |     % Import polygon encapsulating SDRs
168 |     polyLon = polygon(:,1);
169 |     polyLat = polygon(:,2);
170 |
171 |     % Find limits of max extent of orthogonal profiles
172 |     mid = floor(size(profLat,1)/2);
173 |     minLat = profLat(mid,1);
174 |     maxLat = profLat(mid,1);
175 |     minLon = profLon(mid,1);
176 |     maxLon = profLon(mid,1);
177 |
178 |     for j =1:size(profLon,2)
179 |         for i = 1:size(profLon,1)
180 |             if profLon(i,j) > maxLon
181 |                 maxLon = profLon(i,j);
182 |             elseif profLon(i,j) < minLon
183 |                 minLon = profLon(i,j);
184 |             end
185 |             if profLat(i,j) > maxLat
186 |                 maxLat = profLat(i,j);
187 |             elseif profLat(i,j) < minLat
188 |                 minLat = profLat(i,j);
189 |             end
190 |         end
191 |     end
192 |
193 |     % Define grid query points on grid
194 |     latRange = length(minLat:0.001:maxLat);
195 |     lonRange = length(minLon:0.001:maxLon);
196 |
197 |     lonq = NaN(latRange,lonRange);
198 |     latq = NaN(latRange,lonRange);
199 |
200 |     for i = 1:latRange
201 |         lonq(i,:) = minLon:0.001:maxLon;
202 |     end
203 |
204 |     for i = 1:lonRange
205 |         latq(:,i) = minLat:0.001:maxLat;
206 |     end
207 |
208 |     % Test whether in polygon
209 |     in = inpolygon(lonq,latq,polyLon,polyLat);
210 |
211 |     % Construct referencing vector for grid
212 |     RPoly = [1000 maxLat minLon];
213 |
214 |     % Extract values from grid using ltln2val
215 |     zValPoly = NaN(size(profLat));
216 |
217 |     for i = 1:n
218 |         for j = 1:size(profLon,1)
219 |             zValPoly(j,i) = ltln2val(in,RPoly,profLat(j,i),profLon(j,i));
220 |         end
221 |     end
222 |
223 |     % Plot to check - N.B. in cartesian
224 |     figure
```

```
225     hold on
226     plot(lonq(in),latq(in),'g+')
227     plot(polyLon, polyLat,'-r')
228     for i = 1:(size(profLon,2))
229         plot(profLon(:,i),profLat(:,i),'-k')
230     end
231     xlabel('Longitude')
232     ylabel('Latitude')
233     title('Profiles over known SDR location polygon')
234 
235     % Sum and replace zeros with NaNs and value for plotting purposes
236     posSDR = sum(zValPoly,2);
237     posSDR(posSDR == 0) = NaN;
238     posSDR(posSDR > 0) = floor(min(min(zVal))) - 10;
239 
240 end
241 
242 %% Plot profiles
243 
244 % Find maximum and minimum IA values at each point
245 maxIA = NaN(size(zMean));
246 minIA = NaN(size(zMean));
247 
248 for i = 1:size(zVal,1)
249     maxIA(i) = max(zVal(i,:));
250     minIA(i) = min(zVal(i,:));
251 end
252 
253 % Find maximum and minimum bathy values at each point
254 maxBathy = NaN(size(zMean));
255 minBathy = NaN(size(zMean));
256 
257 for i = 1:size(bathyVal,1)
258     maxBathy(i) = max(bathyVal(i,:));
259     minBathy(i) = min(bathyVal(i,:));
260 end
261 
262 % Find maximum and minimum mag values at each point
263 maxMag = NaN(size(zMean));
264 minMag = NaN(size(zMean));
265 
266 for i = 1:size(zVal,1)
267     maxMag(i) = max(magVal(i,:));
268     minMag(i) = min(magVal(i,:));
269 end
270 
271 % Plot Isostatic Anomaly profiles
272 figure
273 subplot(3,1,1),hold on,
274 if landSide == 'E'
275     set(gca, 'XDir','reverse')
276 end
277 subplot(3,1,1),ylabel('Isostatic Anomaly (mGal)')
278 subplot(3,1,1),title('Isostatic Anomaly Profiles Orthogonal to Margin')
279 
280 if strcmp(plotStyle,'fill')
281     h1 = fill([distKm fliplr(distKm)],[maxIA; flipud(minIA)],[0.8 0.8 0.8]);
282     set(h1,'edgecolor','white');
283 elseif strcmp(plotStyle,'lines')
284     for i = 1:n
285         subplot(3,1,1),plot(distKm,zVal(:,i),'LineWidth',0.5,'Color',[0.7 0.7 0.7])
286     end
287 end
288 
289 subplot(3,1,1),plot(distKm,zMean,'-k','LineWidth',2)
290 subplot(3,1,1),plot(distKm,ZMean,'-r','LineWidth',2)
291 
292 if sdr == 1
293     subplot(3,1,1),plot(distKm,posSDR,'-b','LineWidth',2.5)
294 end
295 
296 % Plot bathymetry profiles
297 subplot(3,1,2),hold on,
298 if landSide == 'E'
299     set(gca, 'XDir','reverse')
300 end
301 subplot(3,1,2),ylabel('Bathymetry (m)')
302 subplot(3,1,2),title('Bathymetry Profiles Orthogonal to Margin')
303 
304 if strcmp(plotStyle,'fill')
305     h2 = fill([distKm fliplr(distKm)],[maxBathy; flipud(minBathy)],[0.8 0.8 0.8]);
306     set(h2,'edgecolor','white');
307 elseif strcmp(plotStyle,'lines')
308     for i = 1:n
```

```
309         subplot(3,1,2),plot(distKm,bathyVal(:,i),'LineWidth',0.5,'Color',[0.7 0.7 ↩
        0.7])
310     end
311 end
312
313 subplot(3,1,2),plot(distKm,bathyMean,'-k','LineWidth',2)
314 subplot(3,1,2),plot(distKm,BathyMean,'-r','LineWidth',2)
315
316 % Plot magnetic profiles
317 subplot(3,1,3),hold on,
318 if landSide == 'E'
319     set(gca, 'XDir','reverse')
320 end
321 subplot(3,1,3),ylabel('Magnetic anomaly (nT)')
322 subplot(3,1,3),xlabel('Distance (km)')
323 subplot(3,1,3),title('Magnetic Profiles Orthogonal to Margin')
324
325 if strcmp(plotStyle,'fill')
326     h3 = fill([distKm fliplr(distKm)],[maxMag; flipud(minMag)],[0.8 0.8 0.8]);
327     set(h3,'edgecolor','white');
328 elseif strcmp(plotStyle,'lines')
329     for i = 1:n
330         subplot(3,1,3),plot(distKm,magVal(:,i),'LineWidth',0.5,'Color',[0.7 0.7 0.7])
331     end
332 end
333
334 subplot(3,1,3),plot(distKm,magMean,'-k','LineWidth',2)
335 subplot(3,1,3),plot(distKm,MagMean,'-r','LineWidth',2)
336
337 % Export profile data as csv files
338 latdest  = strcat(out_dir,'profiles_lat.csv');
339 londest  = strcat(out_dir,'profiles_lon.csv');
340
341 csvwrite(latdest, profLat);
342 csvwrite(londest, profLon);
343
344 % Extract the observed IA mean over the SDRs and save
345 if sdr == 1
346     IA_SDR_observed = [distKm', ZMean];
347     IA_SDR_observed = IA_SDR_observed( isnan(posSDR(:,1)), :);
348     iasdrdest  = strcat(out_dir,'IA_SDR_observed.csv');
349     csvwrite(iasdrdest, IA_SDR_observed);
350 end
351
352 %% Plot Isostatic Only at fixed size and scale for global compilation
353
354 % Plot Isostatic Anomaly profiles
355 fig = figure;
356 if landSide == 'E'
357     set(gca, 'XDir','reverse')
358 end
359 hold on
360 ylabel('Isostatic Anomaly (mGal)')
361 xlabel('Distance (km)')
362 title('Isostatic Anomaly Profiles Orthogonal to Margin')
363 %ylim([-50 50])
364
365 if strcmp(plotStyle,'fill')
366     h1 = fill([distKm fliplr(distKm)],[maxIA; flipud(minIA)],[0.8 0.8 0.8]);
367     set(h1,'edgecolor','white');
368 elseif strcmp(plotStyle,'lines')
369     for i = 1:n
370         plot(distKm,zVal(:,i),'LineWidth',0.5,'Color',[0.7 0.7 0.7])
371     end
372 end
373
374 plot(distKm,zMean,'-k','LineWidth',2)
375 plot(distKm,ZMean,'-r','LineWidth',2)
376
377 if sdr == 1
378     plot(distKm,posSDR,'-b','LineWidth',2.5)
379 end
380
381 fig.PaperUnits = 'centimeters';
382 fig.PaperPosition = [0 0 20 8];
383
384 %% Plot magnetics at same scale in case needed
385
386 fig = figure;
387 if landSide == 'E'
388     set(gca, 'XDir','reverse')
389 end
390 hold on
391 ylabel('Magnetic anomaly (nT)')
```

```
xlabel('Distance (km)')
title('Magnetic Profiles Orthogonal to Margin')
ylim([-200 200])

if strcmp(plotStyle,'fill')
    h1 = fill([distKm fliplr(distKm)],[maxMag; flipud(minMag)],[0.8 0.8 0.8]);
    set(h1,'edgecolor','white');
elseif strcmp(plotStyle,'lines')
    for i = 1:n
        plot(distKm,magVal(:,i),'LineWidth',0.5,'Color',[0.7 0.7 0.7])
    end
end

plot(distKm,magMean,'-k','LineWidth',2)
plot(distKm,MagMean,'-r','LineWidth',2)

if sdr == 1
    plot(distKm,posSDR-140,'-b','LineWidth',2.5)
end

% Overlay calculated magnetics
if magnetics == 1
    calc_mag = importdata(mag_filename,' ',1);
    x_mag = calc_mag.data(:,1) + x_mag_adj;
    z_mag = flipud(calc_mag.data(:,2)) + y_mag_adj;
    plot(x_mag, z_mag, '--b','LineWidth',2);
end

fig.PaperUnits = 'centimeters';
fig.PaperPosition = [0 0 20 8];

%% Export calculated magnetic values
if save_mag_calc == 1
    A = [distKm; MagMean'];
    magcalcdest  = strcat(out_dir,'mag_calc_ensemble.txt');
    fileID = fopen(magcalcdest,'w');
    fprintf(fileID, '%6s    %6s\n', 'x (km)', 'z (nT)');
    fprintf(fileID, '%3.3f  %3.3f\n', A);
    fclose(fileID);
end

%% Create grid of ensemble average to find residual (see Bassett and Watts, 2015)
project_ensemble(profLon, profLat, ZMean, 'IA', out_dir);
project_ensemble(profLon, profLat, MagMean, 'mag', out_dir);
project_ensemble(profLon, profLat, BathyMean, 'bathy', out_dir);
```

```
function [zValAl, lags] = cross_correlate(zVal, maxLag, p)
% Cross-correlate along profile measurements and shift to align
%
% Inputs:
% zVal = matrix of profiles ixj, where j = # profiles
% maxLag = int, number of indices max allowable lag
% p = 1 for plotting each profile, 0 for no plotting
%
% Outputs:
% zValAl = corresponding matrix of now aligned values, with inapinted NaNs
% lags = vector of lag used for each profile

% Profile to use for master - set to middle
n = size(zVal, 2);
master = floor(n/2);

% Calculate xcorr between each profile and the first, shift
lags = zeros(1, n);
zValAl = NaN(size(zVal));  % Store aligned

for i = 1:n
    % Calculate xcorr based only on marine component
    [acor,lag] = xcorr(zVal(:,master),zVal(:,i),maxLag);
    [ ,I] = max(abs(acor));
    lags(i) = lag(I);

    % Shift values
    if lags(i) < 0
        temp = zVal(-lags(i) + 1:end,i);
        zValAl(1:length(temp),i) = temp;
    elseif lags(i) > 0
        temp = zVal(1:end-lags(i),i);
        zValAl(lags(i) + 1:end,i) = temp;
```

```
    else
        zValAl(:, i) = zVal(:,i);
    end

    % Plot (if desired)
    if p == 1
        figure
        subplot(3, 1, 1), plot(lag,acor), title(['Shifted by: ' num2str(lags(i))])
        subplot(3, 1, 2), plot(zVal(:,i),'--rx')
        subplot(3, 1, 2), hold on, plot(zValAl(:,i),'--b+')
        subplot(3, 1, 2), legend('Original', 'Aligned')
        subplot(3, 1, 3), plot(zVal(:,master),'-k')
    end

    clear acor; clear lag; clear I; clear temp;

    % Extrapolate ends in case of NaN values which break fft
    zValAl(:,i) = inpaint_nans(zValAl(:,i));

end
```

```
function [zArithMean, zEnsemble] = calc_averages(zVal)
% Calculate the arithmetic and ensemble mean
%
% Inputs:
% zval = matrix of profiles, ixj where j = # profiles
%
% Outputs:
% zArithMean = vector, ix1, arithmetic mean
% zEnsemble = vector, ix1, ensemble average in spectral domain

% Calculate arithmetic mean
zArithMean = mean(zVal,2);

% Calculate ensemble average in spectral domain
ZVal = NaN(size(zVal));

for i = 1:size(ZVal, 2)
    ZVal(:,i) = fft(zVal(:,i));
end

ZMeanS = sum(ZVal,2)./size(ZVal,2);
zEnsemble = ifft(ZMeanS);
```

```
function project_ensemble(profLon, profLat, zEnsemble, name, out_dir)
% Given location of profile points and ensemble average profile, project
% these values back onto geographic coordinates ans save as xyz file for
% gridding in GMT.
%
% Inputs:
% profLat, profLon = matrix of projected profile coordinates
% zEnsemble = vector of corresponding ensemble values
% name = str to append to output file names


% Fill in values
ensVals = NaN(size(profLon));

for i=1:size(profLon, 2)
    ensVals(:,i) = zEnsemble;
end

% Reshape into vectors for export
profLon = reshape(profLon, 1, []);
profLat = reshape(profLat, 1, []);
ensVals = reshape(ensVals, 1, []);

% Save xyz txt file
A = [profLon; profLat; ensVals];
fileID = fopen(strcat(out_dir, 'ensemble_grid_points_', name, '.txt'),'w');
fprintf(fileID, '%3.3f  %3.3f %3.3f\n', A);
fclose(fileID);

% Generate region string using min/max values for GMT
minLat = floor(min(min(profLat)));
```

```
maxLat = ceil(max(max(profLat)));
minLon = floor(min(min(profLon)));
maxLon = ceil(max(max(profLon)));

reg_string = ['-R' num2str(minLon) '/' num2str(maxLon) '/' num2str(minLat) '/' num2str(maxLat)];
fileID = fopen(strcat(out_dir, 'region_string_', name, '.txt'),'w');
L = strlength(reg_string);
fprintf(fileID, ['%' num2str(L) 's'], reg_string);
fclose(fileID);

% Define polygon which encloses points for masking in GMT
k = boundary(profLon', profLat');
B = [profLon(k); profLat(k)];
fileID = fopen(strcat(out_dir, 'polygon_bounding_', name, '.txt'),'w');
fprintf(fileID, '%3.3f %3.3f\n', B);
fclose(fileID);
```

```
function [profLat, profLon, distKm, lat, lon, latProfs, lonProfs] = project_profiles(input_lat, input_lon, landSide, n, degL, degS, out_dir)
% Project profiles perpendicular to some input track (like GMT project)
%
% Inputs
% n = int, number of profiles = n - 1
% input_lat and input_lon = vector, coords of input track to project from
% landSide = 'E' or 'W', which side of the profile is the landward side
% degL, degS = ints, number of degrees landward/seaward of input to project
%
% Outputs
% lat, lon = interpolated input track
% latSm, lonSm = smoothed input track
% profLat, profLon = matrix of projected profile coordinates
% distKm = vector, relative distance in km along profile (0 is intersection
% with input track)

% Force picks to be north -> south down the column vector
x = input_lon;
y = input_lat;

if y(1) < y(end)
    x = flipud(x);
    y = flipud(y);
end

% Perform interpolation on input points, 60 points per degree
range = max(y) - min(y);
pts = range * 60;
lat = linspace(min(y)-0.1,max(y)+0.1,pts);
[y, index] = unique(y);
lon = interp1(y,x(index),lat);

% Smooth over 2 degrees latitude with a robust linear regression
smooth_gap = round(2/(range/length(lat)));
disp(smooth_gap)

latSm = smoothdata(lat, 'rlowess', smooth_gap);
lonSm = smoothdata(lon, 'rlowess', smooth_gap);

% Now interpolate for number of profiles
latProfs = linspace(min(y),max(y),n);
lonProfs = interp1(latSm,lonSm,latProfs);

% Calculate azimuth between smoothed points
az = zeros(1,n);

for i = 1:n
    if i == 1
        az(i) = azimuth(lonProfs(i),latProfs(i),lonProfs(i+2),latProfs(i+2));
    elseif i == 2
        az(i) = azimuth(lonProfs(i-1),latProfs(i-1),lonProfs(i+2),latProfs(i+2));
    elseif i == n-1
        az(i) = azimuth(lonProfs(i-2),latProfs(i-2),lonProfs(i+1),latProfs(i+1));
    elseif i == n
        az(i) = azimuth(lonProfs(i-2),latProfs(i-2),lonProfs(i),latProfs(i));
    else
        az(i) = azimuth(lonProfs(i-2),latProfs(i-2),lonProfs(i+2),latProfs(i+2));
    end
end

```

```
% Find great circle paths perpendicular to these azimuths and plot
profLat = NaN(199,n);
profLon = NaN(199,n);

% Project seaward and landward by specified number of degrees
if strcmp(landSide,'E')
    for i = 1:n
        [lat_temp,lon_temp] = track1(latProfs(i),lonProfs(i),180-az(i),degL);
        [lat_temp2,lon_temp2] = track1(latProfs(i),lonProfs(i),180-az(i),-degS);
        profLat(1:100,i) = flipud(lat_temp);
        profLat(101:199,i) = lat_temp2(2:end);
        profLon(1:100,i) = flipud(lon_temp);
        profLon(101:199,i) = lon_temp2(2:end);
    end
elseif strcmp(landSide,'W')
    for i = 1:n
        [lat_temp,lon_temp] = track1(latProfs(i),lonProfs(i),180-az(i),degS);
        [lat_temp2,lon_temp2] = track1(latProfs(i),lonProfs(i),180-az(i),-degL);
        profLat(1:100,i) = flipud(lat_temp);
        profLat(101:199,i) = lat_temp2(2:end);
        profLon(1:100,i) = flipud(lon_temp);
        profLon(101:199,i) = lon_temp2(2:end);
    end
end

% Export profLat and profLon for later use elsewhere
% profs_out=strcat(out_dir, 'profiles.mat');
% save(profs_out, profLat, profLon);

% Calculate relative distances along profiles
dist = NaN(1,199);

for i = 1:100
    [dist_temp, ] = distance(profLat(100,1),profLon(100,1),profLat(i,1),profLon(i,1));
    if strcmp(landSide,'E')
        dist(i) = -dist_temp;
    elseif strcmp(landSide,'W')
        dist(i) = dist_temp;
    end
end

for i = 101:199
    [dist_temp, ] = distance(profLat(100,1),profLon(100,1),profLat(i,1),profLon(i,1));
    if strcmp(landSide,'E')
        dist(i) = dist_temp;
    elseif strcmp(landSide,'W')
        dist(i) = -dist_temp;
    end
end

% Convert distances from degrees to km
distKm = distdim(dist,'degrees','km');
```

C.3 Functions

Modified from program supplied by *Watts (pers. comm.)*.

```
%  Based on the gr3vd_FFT_GMT_abw.m written by A.B. Watts

%  Read a GMT grd file and calculate the 3D gravity effect
%  of a topographic interface of uniform density contrast
%  using a Fast Fourier Transform technique that takes into account
%  the higher order terms. Write output to a GMT grd file.
%
%  Topography below sea-level (i.e. bathymetry) is -ve. Topography
%  above sea-level is +ve. Gravity effect is calculated at sea-level.
%
%
%  The functions grdread and grdwrite were written by Kelsey Jordahl
%  and are available from
```

```
%  http://www.mathworks.com/matlabcentral/fileexchange/authors/73501

% ***************************************************************
%  Set the parameters

function [Grav]=grv3dFFT(X,Y,Z,rho_below,rho_above,order,s)

% rho_below = Density of the layer below the interface kg m-3
% rho_above =  Density of the layer above the interface kg m-3
% Order = order to compute gravity anomaly
% s = 1 to include slab for; 0 no slab

G=6.67e-11;                 % Universal gravitational constant
                            %  m3 kg-1 s-2

% ***********************************************************
% Read a GMT grd topo file. lat and lon are in degrees.
% Depth to the interface is -ve below sea-level

nptsx=length(X);
nptsy=length(Y);

% Find the mean of the topo and remove it from the topo. Then,
% calculate the wavenumber.

meantopo = mean(mean(Z));
interface = Z - meantopo;

avglat=(Y(1)+Y(nptsy))/2.;

dx=(X(nptsx)-X(1))*111.*1000./nptsx;
dy=(Y(nptsy)-Y(1))*111.*1000./(nptsy*cosd(avglat));

xkint=(2*pi)/(nptsx*dx);  %x wavenumber 1/m
ykint=(2*pi)/(nptsy*dy);  %y wavenumber 1/m

% Transform the topo and calculate each term in the gravity
% admittance (mGal/m) of uncompensated topography.
% Initialize arrcccys
grv_t = zeros(nptsy,nptsx);

for n=1:1:order
    c=fft2(interface.^n);
    z=zeros(nptsx,nptsy);

    for i=1:1:nptsx/2 + 1
        xk = (i-1) * xkint;
        for j=1:1:nptsy/2 + 1
            yk = (j-1) * ykint;
            rk1 = ( (xk*xk) + (yk*yk) )^0.5;
            z(j,i) = 2.*pi*G*(rho_below-rho_above)*exp(-rk1*-meantopo)...
                *1.0e5*rk1^(n-1)/factorial(n);
        end
    end

    % Put mirror planes into the z array

    for j=1:1:nptsy/2
        k = nptsy + 2 - j;
        for i=1:1:nptsy
            z(k,i)=z(j,i);
        end
    end

    for i=1:1:nptsx/2
        k = nptsx + 2 - i;
        for j=1:1:nptsx
            z(j,k)=z(j,i);
        end
    end

    % Now convolve the admittance with the fourier transform of the
    % topo to get the first-order gravity effect

    for i=1:1:nptsy
        for j=1:1:nptsx
            grv_t(i,j) = grv_t(i,j) + z(i,j)*c(i,j);
        end
    end

end

grv_topo = real(ifft2(grv_t)); %n

```

```
% The topo had a mean taken out (for the fourier transform) and so the
% gravity effect also has a mean taken out. We can correct for this by
% adding the gravity effect of a slab which has a thickness of meantopo
% and density contrast of (rho_below-rho_above). Otherwise, use
% slab = 0.

if s==1
    % Take difference between mean and regional background
    mindiff = meantopo- min(min(Z));
    maxdiff = meantopo - max(max(Z));
    if mindiff < maxdiff
        slab_thick = meantopo - min(min(Z));
    else
        slab_thick = meantopo - max(max(Z));
    end

    % Apply slab
    slab = 2*pi*G*(rho_below-rho_above)*slab_thick*1.0e5;
    Grav = grv_topo + slab;
else
    Grav = grv_topo;
end

end
```

Appendix D

Shell Scripts

D.1 Ensemble Averaging

```
#!/usr/bin/env bash

# IA or mag?
var='IA'

# Input track or use specified contour
use_contour=false   # true to use contour (specify depth, path to bathygrid) or false to use track (specify path to .txt file)
track='=Namibia.txt'
contour_depth=-500
bathygrid='GEBCO_2008.grd'
lips='UTIG_data/LIPs/LIPS.2011.gmt'

# Grid interval in GMT speak
int='-I1m'

# Name used to generate files in Matlab; same convention will be used here for outputs
# Output directory containing Matlab files, which will also be used for these outputs.
# Files required from MATLAB:
# region_string_$name.txt
# ensemble_grid_points_$name.txt
# polygon_bounding_$name.txt
# profiles_lat.csv
# profiles_lon.csv
path_to_dir='Results/Namibia'


# Original input grid (to Matlab) observed grid path
if [ "$var" == 'mag' ]; then
    in_grid='EMAG2_V3.grd'
else
    in_grid='IA_anomaly_degorder20_2800_1030_3330_31.2.grd'
fi

# Set colour/bar options depending on variable
if [ "$var" == 'mag' ]; then
    name='mag'
    palette='polar'
    bar_title="EMAG2 Magnetic Anomaly"
    bar_unit="nT"
    limit=350
    tick_int=50
else
    name='IA'
    palette='haxby'
    bar_title="Isostatic Anomaly"
    bar_unit="mGal"
    limit=100
    tick_int=20
```

```
fi

# Make colour palettes
gmt makecpt -C"$palette" -T-$limit/$limit/1 -Do > col_ens.cpt
gmt makecpt -C"$palette" -T-$limit/$limit/1 -Do > col_res.cpt
gmt makecpt -C"$palette" -T-$limit/$limit/1 -Do > col_orig.cpt

# Plotting options
projection_width_cm=8
ticks=-Baf
ens_bar_int=$tick_int
res_bar_int=$tick_int
orig_bar_int=$tick_int
coast="-N1"

#####################################################################

# Build projection size and scale bar width
projection="-JM${projection_width_cm}c"
scale_width=$(echo ${projection_width_cm}/2 | bc)

# Path to Matlab generated region string
reg_file="$path_to_dir/region_string_$name.txt"

# xyz text file containing ensemble calculated points
ens_xyz="$path_to_dir/ensemble_grid_points_$name.txt"

# Masking polygon file
mask_poly="$path_to_dir/polygon_bounding_$name.txt"

# Profiles
prof_lon="$path_to_dir/profiles_lon.csv"
prof_lat="$path_to_dir/profiles_lat.csv"

# Output ensemble grid file name
out_ens_grid="$path_to_dir/ensemble_grid_$name.grd"
out_ens_map="$path_to_dir/ensemble_grid_$name.ps"

# Output residual grid name
out_res_grid="$path_to_dir/residual_grid_$name.grd"
out_res_map="$path_to_dir/residual_grid_$name.ps"

# Output original map
out_orig_map="$path_to_dir/input_with_profs_$name.ps"

# Get region
reg=`cat $reg_file`

# Grid ensemble points
gmt blockmean $int $reg $ens_xyz | gmt surface $int $reg -Gunmasked_ens.grd
gmt grdmask $mask_poly $int $reg -Af -NNaN/1/1 -Gmask.grd
gmt grdmath unmasked_ens.grd mask.grd OR = $out_ens_grid
rm mask.grd
rm unmasked_ens.grd

# Plot ensemble grid
rm $out_ens_map
gmt psbasemap $reg $projection -K $ticks -Xc -Yc > $out_ens_map
gmt grdimage $out_ens_grid $reg $projection $ticks -Q -Ccol_ens.cpt -K -O >> ↩
    $out_ens_map
gmt pscoast $reg $projection $coast -Dh -O -K -Wthin,black >> $out_ens_map

if [ "$use_contour" = false ];
then
    gmt psxy $track $reg $projection -Wthick,black,- -O -K >> $out_ens_map
else
    gmt grdcontour -C+$contour_depth $bathygrid $reg $projection -Wthick,black,- -O ↩
    -K >> $out_ens_map
fi

gmt psscale -D${scale_width}c/-1c/${projection_width_cm}c/0.5ch -S ↩
    "-B$ens_bar_int:Ensemble $bar_title:/:$bar_unit:" -Ccol_ens.cpt -O >> $out_ens_map
open $out_ens_map

# Find residual grid
gmt grdsample $in_grid $reg $int -Gtemp.grd
gmt grdmath temp.grd $out_ens_grid SUB = $out_res_grid

# Plot residual grid
rm $out_res_map
gmt psbasemap $reg $projection -K $ticks -Xc -Yc > $out_res_map
gmt grdimage $out_res_grid $reg $projection $ticks -Q -Ccol_res.cpt -K -O >> ↩
    $out_res_map
gmt pscoast $reg $projection $coast -Dh -O -K -Wthin,black >> $out_res_map

if [ "$use_contour" = false ];
```

```
then
    gmt psxy $track $reg $projection -Wthick,black,- -O -K >> $out_res_map
else
    gmt grdcontour -C+$contour_depth $bathygrid $projection $reg -Wthick,black,- -O ↩
    -K >> $out_res_map
fi

gmt psscale -D${scale_width}c/-1c/${projection_width_cm}c/0.5ch -S ↩
    "-B$res_bar_int:Residual $bar_title:/:$bar_unit:" -Ccol_res.cpt -O >> $out_res_map
open $out_res_map

# Plot original grid with profiles
rm $out_orig_map
gmt psbasemap $reg $projection -K $ticks -Xc -Yc > $out_orig_map
gmt grdimage temp.grd $reg $projection $ticks -Q -Ccol_orig.cpt -K -O >> $out_orig_map
gmt psxy $reg $projection $lips -Wthinnest,black -Gred -O -K >> $out_orig_map
gmt pscoast $reg $projection $coast -Dh -O -K -Wthin,black >> $out_orig_map

# Extract profiles from csv
paste -d, $prof_lon $prof_lat > temp.csv

# Count number of profiles
cols=$(head -1 temp.csv | sed 's/[^,]//g' | wc -c)
profs=$(expr $cols / 2)

# Plot extracted profiles
for i in $(seq 1 $profs)
do
    m=$(expr $i + $profs)
    awk -F "\"*,\"*" -v n="$i" -v m="$m" '{print $n, $m}' temp.csv > prof_Temp
    gmt psxy prof_Temp $reg $projection -Wthinnest,black -O -K >> $out_orig_map
done

if [ "$use_contour" = false ];
then
    gmt psxy $track $reg $projection -Wthick,black,- -O -K >> $out_orig_map
else
    gmt grdcontour -C+$contour_depth $bathygrid $projection $reg -Wthick,black,- -O ↩
    -K >> $out_orig_map
fi

gmt psscale -D${scale_width}c/-1c/${projection_width_cm}c/0.5ch -S ↩
    "-B$orig_bar_int:$bar_title:/:$bar_unit:" -Ccol_res.cpt -O >> $out_orig_map
open $out_orig_map

# Clean up
rm temp.csv
rm prof_Temp
rm temp.grd
rm *.cpt
```

```
#!/bin/bash

# User inputs
field="mag" # set "mag"or "grav"
region=-R10/18/-34/-21

projection=-JM12c
ticks=-Ba2f2/a2f2
output=NamibiaTightFit/namibia_mag_profiles_contour.ps
contour="yes" # set "yes" or "no"
contint=20  # if yes, set contour interval

polygon=NamibiaTightFit/SDRpolygon.txt
prof_lon=NamibiaTightFit/profiles_lon.csv
prof_lat=NamibiaTightFit/profiles_lat.csv

# ------------------------------------------------------------------------------
# Extract relevant grid in region
if [ "$field" == "grav" ]
    then
    ↩
    grid=/Users/rebeccamorgan/Drive/University/GMT_Grids/IA_anomaly_2700.3330.1030.35.global.grd
    gmt makecpt -Cpolar -T-50/50/1 -Z > col.cpt
elif [ "$field" == "mag" ]
    then
    grid=/Users/rebeccamorgan/Drive/University/GMT_Grids/EMAG2_V2.grd
    gmt makecpt -Chaxby -T-200/200/1 > col.cpt
fi

gmt grdcut $grid $region -Ggrid_cut.grd

```

```
# Plot map
gmt psbasemap $region $projection -K $ticks -Xc -Yc -P > $output
gmt grdimage grid_cut.grd $region $projection $ticks -Ccol.cpt -K -O >> $output
gmt pscoast $region $projection -F=AF -Df -O -K >> $output
gmt psxy $polygon $region $projection -Wthin,black,-- -O -K >> $output

# If desired add contours
if [ "$contour" == "yes" ]
    then
    gmt grdcontour $grid $region $projection -C$contint -Wgrey -O -K >> $output
fi

# Extract profiles from csv
paste -d, $prof_lon $prof_lat > temp.csv

# Count number of profiles
cols=$(head -1 temp.csv | sed 's/[^,]//g' | wc -c)
profs=$(expr $cols / 2)

# Plot extracted profiles
for i in $(seq 1 $profs)
do
    m=$(expr $i + $profs)
    awk -F "\"*,\"*" -v n="$i" -v m="$m" '{print $n, $m}' temp.csv > prof_Temp
    gmt psxy prof_Temp $region $projection -Wthin,black -O -K >> $output
done

# Add scale bar
if [ "$field" == "grav" ]
    then
    gmt psscale -D6c/-1.5c/12c/1ch -S -B10:"Isostatic Anomaly":/:"mGal": -Ccol.cpt ↩
    -O >> $output
elif [ "$field" == "mag" ]
    then
    gmt psscale -D6c/-1.5c/12c/1ch -S -B50:"EMAG2 V2 Magnetic Anomaly":/:"nT": ↩
    -Ccol.cpt -O >> $output
fi

# Tidy up
rm prof_Temp
rm temp.csv
rm grid_cut.grd

# Open ps output file to check
open $output
```

Appendix E

SDR Observations Table

Reference	Fig. Number	Location	Year Acquisition	Year Pub (earliest)	Region	Ship/Profile ID (where given)	Package	# Reflectors Picked	Package Width (km)	Approx. Water Depth (km)	Approx. Overlying Sed. (TWTT/s, except where stated)	Min. dip (deg)	Max. dip (deg)	Mean dip (deg)	Median dip (deg)	Std. dev. dip	Min. thickness (km)	Max. thickness (km)	Mean thickness (km)	Median thickness (km)	Std. dev. thickness
Ajay et al 2011	2	Laccadive Ridge	1979		India	RE17	Single	4	13	2.1	1	3	26	10	9	4.9	0.6	1.8	1.2	1.1	0.4
Ajay et al 2011	3	Laccadive Ridge	1979		India	RE19	Single	4	25	1.8	1.2	7	26	15	15	4.1	1.5	3.3	2.4	2.4	0.5
Ajay et al 2011	4	Laccadive Ridge	1979		India	RE23	Single	6	25	2.5	0.9	5	57	24	22	10.4	1.1	5.5	2.9	2.7	1.2
Barton and White 1997 (Geol Soc)	2b	Hatton Bank	1985		North East Atlantic	NI8	Inner	3	14	1.3	0.4	6	40	15	14	6.2	1.2	3.3	2.1	2	0.6
Barton and White 1997 (Geol Soc)	2b	Hatton Bank	1985		North East Atlantic	NI8	Outer	4	34	2.6	0.3	2	40	14	12	7.5	1.7	4	2.8	2.8	0.7
Barton and White 1997 (JGR)	3a	Edoras Bank	1992		North East Atlantic	Cam77	Inner	3	22	2.3	0.5	4	23	11	10	4.2	1.2	3.3	2.1	2	0.6
Barton and White 1997 (JGR)	3b	Edoras Bank	1992		North East Atlantic	Cam77	Outer	3	12	3.1	0.5	2	33	16	16	7	1.2	2.5	1.9	2	0.4

Bauer et al 2000	5a	Namibia	2000	SW Africa	Transect 2	Outer	4	29	1.1	2	4	12	7	7	1.9	0.5	1.3	0.8	0.8	0.2
Bauer et al 2000	5b	Namibia	2000	SW Africa	Transect 2	Inner	3	35	1.1	2.3	7	30	14	14	4.2	1.8	5	3.3	3	1
Blaich et al 2013	4	Pelotas Basin	1998	South America	-	Single	5	100	2	1.7	1	25	9	8	4.7	1.4	3.5	2.2	2.1	0.6
Blaich et al 2013	7	Walvis Basin	2011	SW Africa	-	Single	5	70	2.4	1.8	2	21	8	8	4.2	1.2	2.5	1.7	1.6	0.4
Eccles et al 2011	4	Faroes	2002	North East Atlantic	Profile 1	Single	4	65	1.1	1.2	3	40	14	13	7	2.1	6.5	4.1	4	1.2
Franke et al 2007	6	Argentina	1998	South America	BGR98-05	Single	3	35	3.4	2.8	5	50	19	17	8.6	1.8	6.5	4	3.9	1.5
Franke et al 2010	4	Argentina	1998	South America	BGR98-20	Inner (Outer not shown)	3	53	3	3	2	24	10	9	4.4	2.4	5	3.5	3.4	0.8
Franke et al 2010	5	Argentina	1998	South America	BGR98-07	Inner (Outer not shown)	2	73	3.4	3	4	22	12	11	4.5	1.8	4.3	2.9	2.8	0.8
Gladczenko et al 1997	3	Northern Namibia	1997	SW Africa	-	Single	5	90	2.6	1.8	1	18	7	7	3.5	0.9	3.3	1.9	2	0.6
Hinz et al 1999	14	Colorado Basin, Argentina	1987	South America	BGR 87-02	Single	4	27	4.4	1.8	7	36	21	21	6.5	1.8	5.3	3.4	3.3	1
Hinz et al 1999	5	Argentina	1987	South America	BGR 87-01	Single	5	28	3	2.8	8	34	18	17	6.1	1.4	5.8	3.6	3.6	1.2
Horn 2015	12	Uruguay	2015	South America	-	Single	2	60	2.1	3.2km	5	87	18	17	11.9	s	8.2	7.6	7.6	0.6
Jackson et al 2000	1	Outer Voring Plateau	1978	North East Atlantic	-	Single	3	30	1.4	1.3	2	23	9	8	4.5	0.8	2.5	1.3	1.3	0.6

Jackson et al 2000	8	Jacuipe Basin, Brazil	1990		South America	Line 2	Single	4	15	3.8	2.5	7	40	21	21	6.6	1.5	3.3	2.4	2.4	0.5
Jackson et al 2000	9	Walvis Basin		2000	SW Africa	Line 3	Single	3	60	1.7	1.8	2	18	6	6	2.8	1.8	4	2.9	3	0.7
Koopmans et al 2014	6	South Africa	2003		SW Africa	BGR03_12	Inner	3	65	3.2	1.4	1	35	10	8	6.8	2	4.5	3.3	3.3	0.8
Koopmans et al 2014	6	South Africa	2003		SW Africa	BGR03_12	Outer	1	20	4.2	1.5	2	5	3	3	0.8	0.8	1.3	1	1	0.2
Koopmans et al 2014	7	South Africa	2003		SW Africa	BGR03_16A	Inner	4	100	2.7	2	1	41	10	9	6.5	2.3	6.8	4.1	3.9	1.2
Koopmans et al 2014	7	South Africa	2003		SW Africa	BGR03_16A	Outer	1	22	3.8	1.6	2	4	3	3	0.8	0.5	0.8	0.6	0.6	0.1
Koopmans et al 2014	9	Orange Basin, Namibia		2014	SW Africa	-	Single	2	70	0.6	3.4	1	21	9	8	3.7	4.5	8	6.2	6.2	1.3
Koopmans et al 2014	12	Namibe Basin, Namibia		2014	SW Africa	-	Single	4	210	1.1	1.5	1	32	8	7	5	4.2	12.5	8.2	7.5	2.5
Lau et al 2010	12	Fugloy Ridge, Faroes	2002		North East Atlantic	iSIMM	Single	6	70	1	0.5km	1	29	12	11	6	2.4	4.2	3.2	3.2	0.6
McDermott et al 2015	7	Walvis Basin	2014		SW Africa	Namibia SPAN	Inner	2	120	1.1	2.5	2	18	8	8	4	1.4	3.8	2.4	2.3	0.8
Pindell et al 2014	4	Pelotas Basin, Brazil		2014	South America	PS1-0010	Single	4	55	1.6	7km	4	40	15	13	6.9	1.8	5.6	3.5	3.2	1.4
Planke et al 2000	9	Exmouth Plateau	1995		Australia	GA-162/02	Outer (Inner not shown)	4	23	2	0.4	5	26	12	11	4.3	1.4	3	2	1.9	0.5
Planke et al 2000	6a	Central Vøring Margin		2000	North East Atlantic	C-166	Inner	3	16	2.3	1.1	4	22	11	11	3.8	1.1	3	2.1	1.8	0.6

Planke et al 2000	6a	Central Voring Margin	2000	North East Atlantic	C-166	Outer	3	40	3.2	0.4	2	14	7	6	2.5	1.1	2.8	1.9	1.8	0.6
Planke et al 2000	6b	North-central Cuvier Margin	1992	North East Atlantic	LDEO-670	Inner (Outer not shown)	3	33	2.7	1.3	3	37	19	18	7.7	4.8	10.8	7.8	8	1.9
Planke et al 2000	7c	Jan Mayen Fracture Zone	2000	North East Atlantic	HV96-1	Inner (Outer not shown)	2	35	3	0.9	1	7	3	3	1.3	0.9	1.5	1.2	1.2	0.2
Rey et al 2008	9	Gascoyne Margin	1995	Australia	162-02	Outer (no Inner)	5	24	4.9	0.3	3	25	12	11	5	0.9	2.3	1.4	1.4	0.4
Rey et al 2008	10	Gascoyne Margin	1995	Australia	162-08	Outer (no Inner)	3	27	4.9	1	5	26	12	12	4	1.1	2.0	1.5	1.6	0.3
Schnabel et al 2008	2	Argentine Basin	1993	South America	SO85-005	Single	3	45	3	2	4	25	11	10	4.3	1.7	4.8	3	2.9	0.9
Sheridan et al 1993	3	Norfolk Basin, USA	1990	North America	MA 801	Single	4	62	0	4	2	54	17	17	7.5	2.7	7.8	5.1	4.6	1.5
Soto et al 2011	5	Uruguay	2001	South America	COPLA-02	Single	3	60	3.8	3.2	3	17	10	10	3.5	1.4	3.3	2.3	2.3	0.6
Soto et al 2011	6	Uruguay	2007	South America	-	Single	3	54	-	-	7	32	15	14	5	1.2	4	2.4	2.4	0.8
Soto et al 2011	7	Uruguay	2007	South America	-	Single	3	40	-	-	1	16	6	6	2.8	1.8	4.8	3.1	3	0.9
Stica et al 2013	6	Pelotas Basin	2013	South America	Pelotas SPAN Line 90	Inner	3	130	0	5.7km	3	18	13	14	3	3.8	5.2	4.6	4.8	0.6

Stica et al 2013	6	Pelotas Basin	2013	South America	Pelotas SPAN Line 90	Outer	4	130	2.9	1.4km	1	33	13	13	7	4.0	7.0	4.9	4.2	1.2

Table E.1: Compilation of geometric SDR observations from representative global dataset.

Bibliography

Abdelmalak, M. M., T. B. Andersen, S. Planke, J. I. Faleide, F. Corfu, C. Tegner, G. E. Shephard, D. Zastrozhnov, and R. Myklebust (2015), The ocean-continent transition in the mid-Norwegian margin: Insight from seismic data and an onshore Caledonian field analogue, *Geol*, *43*(11), 1011–1014, doi:10.1130/G37086.1.

Ayele, A., D. Keir, C. J. Ebinger, T. J. Wright, G. W. Stuart, W. R. Buck, E. Jacques, G. Ogubazghi, and J. Sholan (2009), September 2005 mega-dike emplacement in the Manda-Harraro nascent oceanic rift (Afar depression), *Geophys. Res. Lett.*, *36*(20), 177, doi:10.1029/2009GL039605.

Barton, A. J., and R. S. White (1997a), Volcanism on the Rockall continental margin, *Journal of the Geological Society*, *154*(3), 531–536, doi:10.1144/gsjgs.154.3.0531.

Barton, A. J., and R. S. White (1997b), Crustal structure of Edoras Bank continental margin and mantle thermal anomalies beneath the North Atlantic, *J. Geophys. Res. Solid Earth*, *102*(B2), 3109–3129, doi:10.1029/96JB03387.

Bassett, D., and A. B. Watts (2015), Gravity anomalies, crustal structure, and seismicity at subduction zones: 1. Seafloor roughness and subducting relief, *Geochem. Geophys. Geosyst.*, *16*(5), 1508–1540, doi:10.1002/2014GC005684.

Bauer, K., S. Neben, B. Schreckenberger, R. Emmermann, K. Hinz, N. Fechner, K. Gohl, A. Schulze, R. B. Trumbull, and K. Weber (2000), Deep structure of the Namibia continental margin as derived from integrated geophysical studies, *J. Geophys. Res. Solid Earth*, *105*(B11), 25,829–25,853, doi:10.1029/2000JB900227.

BODC (2014), Gridded bathymetry data (General Bathymetric Chart of the Oceans).

Bodvarsson, G., and G. P. L. Walker (1964), Crustal Drift in Iceland, *Geophysical Journal of the Royal Astronomical Society*, *8*(3), 285–300, doi:10.1111/j.1365-246x.1964.tb06295.x.

Bott, M. H. P. (1960), The use of Rapid Digital Computing Methods for Direct Gravity Interpretation of Sedimentary Basins, *Geophysical Journal of the Royal Astronomical Society*, *3*(1), 63–67, doi:10.1111/j.1365-246X.1960.tb00065.x.

Brocher, T. M. (2005), Empirical Relations between Elastic Wavespeeds and Density in the Earth's Crust, *Bulletin of the Seismological Society of America*, *95*(6), 2081–2092, doi:10.1785/0120050077.

Buck, W. R. (1988), Flexural rotation of normal faults, *Tectonics*, *7*(5), 959–973, doi:10.1029/TC007i005p00959.

Buck, W. R. (2017), The role of magmatic loads and rift jumps in generating seaward dipping reflectors on volcanic rifted margins, *Earth and Planetary Science Letters*, *466*, 62–69, doi:10.1016/j.epsl.2017.02.041.

Burov, E. B., and A. Poliakov (2001), Erosion and rheology controls on synrift and postrift evolution: Verifying old and new ideas using a fully coupled numerical model, *J. Geophys. Res. Solid Earth*, *106*(B8), 16,461–16,481.

Christensen, N. I., and M. H. Salisbury (1975), Structure and constitution of the lower oceanic crust, *Rev. Geophys.*, *13*(1), 57–86, doi:10.1029/RG013i001p00057.

Cochran, J. R. (1983), Effects of finite rifting times on the development of sedimentary basins, *Earth and Planetary Science Letters*, *66*(C), 289–302, doi:10.1016/0012-821X(83)90142-5.

Coffin, M., R. Duncan, O. Eldholm, J. G. Fitton, F. Frey, H. C. Larsen, J. Mahoney, A. Saunders, R. Schlich, and P. Wallace (2006), Large Igneous Provinces

and Scientific Ocean Drilling: Status Quo and A Look Ahead, *Oceanography*, *19*(4), 150–160, doi:10.5670/oceanog.2006.13.

Coffin, M. F., and O. Eldholm (1994), Large igneous provinces: Crustal structure, dimensions, and external consequences, *Rev. Geophys.*, *32*(1), 1–36, doi:10.1029/93RG02508.

Collier, J. S., C. McDermott, G. Warner, N. Gyori, M. Schnabel, K. McDermott, and B. W. Horn (2017), New constraints on the age and style of continental breakup in the South Atlantic from magnetic anomaly data, *Earth and Planetary Science Letters*, *477*, 27–40, doi:10.1016/j.epsl.2017.08.007.

Contreras-Reyes, E., and A. Osses (2010), Lithospheric flexure modelling seaward of the Chile trench: implications for oceanic plate weakening in the Trench Outer Rise region, *Geophys. J. Int.*, pp. no–no, doi:10.1111/j.1365-246X.2010.04629.x.

Corti, G., A. Agostini, D. Keir, J. van Wijk, I. D. Bastow, and G. Ranalli (2015), Magma-induced axial subsidence during final-stage rifting: Implications for the development of seaward-dipping reflectors, *Geosphere*, *11*(3), 563–571, doi:10.1130/GES01076.1.

Dahl-Jensen, T., and W. S. Holbrook (1996), SIGMA Seismic investigation of the Greenland Margin, pp. 1–203.

Dix, C. H. (1955), Seismic Velocities from Surface Measurements, *Geophysics*, *20*(1), 68–86, doi:10.1190/1.1438126.

Dodd, S. C., C. Mac Niocaill, and A. R. Muxworthy (2015), Long duration (>4 Ma) and steady-state volcanic activity in the early Cretaceous Paraná–Etendeka Large Igneous Province: New palaeomagnetic data from Namibia, *Earth and Planetary Science Letters*, *414*(C), 16–29, doi:10.1016/j.epsl.2015.01.009.

Ebinger, C. J. (2005), Continental break-up: The East African perspective, *Astronomy and Geophysics*, *46*(2), 16–21.

Eldholm, O., J. Skogseid, S. Planke, and T. P. Gladczenko (1995), Volcanic margin concepts, in *Rifted Ocean-Continent Boundaries*, edited by E. Banda, M. Torné, and M. Talwani, Kluwer Academic Publishers, Dordrecht.

Franke, D., S. Ladage, M. Schnabel, B. Schreckenberger, C. Reichert, K. Hinz, M. Paterlini, J. de Abelleyra, and M. Siciliano (2010), Birth of a volcanic margin off Argentina, South Atlantic, *Geochem. Geophys. Geosyst.*, *11*(2), n/a–n/a, doi: 10.1029/2009GC002715.

Geoffroy, L., J. P. Gelard, C. Lepvrier, and P. Olivier (1998), The coastal flexure of Disko (West Greenland), onshore expression of the 'oblique reflectors', *Journal of the Geological Society*, *155*(3), 463–473, doi:10.1144/gsjgs.155.3.0463.

Geoffroy, L., J. P. Callot, S. Scaillet, A. Skuce, J. P. Gelard, M. Ravilly, J. Angelier, B. Bonin, C. Cayet, K. Perrot, and C. Lepvrier (2001), Southeast Baffin volcanic margin and the North American-Greenland plate separation, *Tectonics*, *20*(4), 566–584, doi:10.1029/2001TC900003.

Geoffroy, L., E. B. Burov, and P. Werner (2015), Volcanic passive margins: another way to break up continents, *Sci. Rep.*, *5*, 14,828, doi:10.1038/srep14828.

Gibson, I. L., and D. Love (1989), A Listric Fault Model for the Formation of the Dipping Reflectors Penetrated During the Drilling of Hole 642e, Odp Leg 104 , in *Proc., scientific results, ODP, Leg 104, Norwegian Sea*, pp. 979–983.

Gibson, S. A., R. N. Thompson, and J. A. Day (2006), Timescales and mechanisms of plume–lithosphere interactions: 40Ar/39Ar geochronology and geochemistry of alkaline igneous rocks from the Paraná–Etendeka large igneous province, *Earth and Planetary Science Letters*, *251*(1-2), 1–17, doi:10.1016/j.epsl.2006.08.004.

Goetze, C., and B. Evans (1979), Stress and temperature in the bending lithosphere as constrained by experimental rock mechanics, *Geophysical Journal of the Royal Astronomical Society*, *59*(3), 463–478, doi:10.1111/j.1365-246X.1979.tb02567.x.

Grant, F. S., and G. F. West (1965), Interpretation theory in applied geophysics, doi:10.1017/S0016756800050627.

Gregg, T. K. P., and D. J. Fornari (1998), Long submarine lava flows: Observations and results from numerical modeling, *103*(B11), 27,517–27,531.

Hale, D. (1984), Dip-moveout by Fourier transform, *Geophysics*, *49*(6), 741–757, doi:10.1190/1.1441702.

Hall, S. A., D. E. Bird, D. J. McLean, P. J. Towle, J. V. Grant, and H. A. Danque (2018), New constraints on the age of the opening of the South Atlantic basin, *Marine and Petroleum Geology*, *95*, 50–66, doi:10.1016/j.marpetgeo.2018.03.010.

Haupert, I., G. Manatschal, A. Decarlis, and P. Unternehr (2016), Upper-plate magma-poor rifted margins: Stratigraphic architecture and structural evolution, *Marine and Petroleum Geology*, *69*(C), 241–261, doi:10.1016/j.marpetgeo.2015.10.020.

Hetényi, M. (1946), Beams on elastic foundation: theory with applications in the fields of civil and mechanical engineering.

Hinz, K. (1981), Hypothesis on terrestrial catastrophes: wedges of very thick oceanward dipping layers beneath passive continental margins - their origins and paleoenvironmental significance., *Geologisches Jahrbuch, Series E*, *22*(12), 1–28, doi:10.1016/0198-0254(85)93886-5.

Holbrook, W. S., H. C. Larsen, J. Korenaga, T. Dahl-Jensen, I. D. Reid, P. B. Kelemen, J. R. Hopper, G. M. Kent, D. Lizarralde, S. Bernstein, and R. S. Detrick (2001), Mantle thermal structure and active upwelling during continental breakup in the North Atlantic, *Earth and Planetary Science Letters*, *190*(3), 251–266, doi:10.1016/S0012-821X(01)00392-2.

Hopper, J. R., T. Dahl-Jensen, W. S. Holbrook, H. C. Larsen, D. Lizarralde, J. Korenaga, G. M. Kent, and P. B. Kelemen (2003), Structure of the SE Greenland margin from seismic reflection and refraction data: Implications for nascent

spreading center subsidence and asymmetric crustal accretion during North Atlantic opening, *J. Geophys. Res.*, *108*(B5), 253, doi:10.1029/2002JB001996.

Horn, B. (2015), Access and exploration opportunities - A view of the potential in frontier and mature basins, *First Break*, *33*(1), 85–93.

Hunter, J. (2016), Flexure and Rheology of Pacific Oceanic Lithosphere, Ph.D. thesis.

Jones, E., T. Oliphant, P. Peterson, and others (2001), SciPy: Open Source Scientific Tools for Python.

Karner, G. D., and A. B. Watts (1982), On isostasy at Atlantic-type continental margins, *J. Geophys. Res.*, *87*(B4), 2923, doi:10.1029/jb087ib04p02923.

Kearey, P., M. Brooks, and I. Hill (2002), *An Introduction to Geophysical Exploration*, 3 ed., Oxford : Blackwell Science.

Koopmann, H., D. Franke, B. Schreckenberger, H. Schulz, A. Hartwig, H. Stollhofen, and R. di Primio (2014), Segmentation and volcano-tectonic characteristics along the SW African continental margin, South Atlantic, as derived from multichannel seismic and potential field data, *Marine and Petroleum Geology*, *50*(C), 22–39, doi:10.1016/j.marpetgeo.2013.10.016.

McAdoo, D. C., C. F. Martin, and S. Poulouse (1985), Seasat observations of flexure: Evidence for a strong lithosphere, *Tectonophysics*, *116*(3-4), 209–222, doi:10.1016/0040-1951(85)90209-4.

McDermott, C., L. Lonergan, J. S. Collier, K. G. McDermott, and P. Bellingham (2018), Characterization of seaward-dipping reflectors along the S. American Atlantic margin and implications for continental breakup, *Tectonics*, doi:10.1029/2017TC004923.

McLachlan, L. R. (1990), Introductory note on the history of the Kudu prospect, *Communications of the geological survey of Namibia*, *6*(5), 886–886, doi:10.1093/ahr/rhy009.

Menzies, M. A., S. L. Klemperer, C. J. Ebinger, and J. Baker (2002), Characteristics of volcanic rifted margins, *Geological Society of America Special Papers*, *362*, 1–14, doi:10.1130/0-8137-2362-0.1.

Meyer, B., R. Saltus, and A. Chulliat (2017), EMAG2: Earth Magnetic Anomaly Grid (2-arc-minute resolution) Version 3, doi:10.7289/V5H70CVX.

Morgan, R. L., and A. B. Watts (2018), Seismic and gravity constraints on flexural models for the origin of seaward dipping reflectors, *Geophys. J. Int.*, *214*(3), 2073–2083, doi:10.1093/gji/ggy243.

Moulin, M., D. Aslanian, and P. Unternehr (2010), A new starting point for the South and Equatorial Atlantic Ocean, *Earth Science Reviews*, *98*(1-2), 1–37, doi:10.1016/j.earscirev.2009.08.001.

Mutter, J. C., M. Talwani, and P. L. Stoffa (1982), Origin of seaward-dipping reflectors in oceanic crust off the Norwegian margin by "subaerial sea-floor spreading", *Geol*, *10*(7), 353–6, doi:10.1130/0091-7613(1982)10<353:OOSRIO>2.0.CO;2.

Parkin, C. J., and R. S. White (2008), Influence of the Iceland mantle plume on oceanic crust generation in the North Atlantic, *Geophys. J. Int.*, *173*(1), 168–188, doi:10.1111/j.1365-246X.2007.03689.x.

Parsons, B., and J. G. Sclater (1977), An analysis of the variation of ocean floor bathymetry and heat flow with age, *J. Geophys. Res.*, *82*(5), 803–827, doi:10.1029/jb082i005p00803.

Paton, D. A., J. Pindell, K. McDermott, P. Bellingham, and B. Horn (2017), Evolution of seaward-dipping reflectors at the onset of oceanic crust formation at volcanic passive margins: Insights from the South Atlantic, *Geol*, *45*(5), 439–442, doi:10.1130/G38706.1.

Pavlis, N. K., S. A. Holmes, S. C. Kenyon, and J. K. Factor (2012), The development and evaluation of the Earth Gravitational Model 2008 (EGM2008), *J. Geophys. Res.*, *117*(B4), n/a–n/a, doi:10.1029/2011JB008916.

Pinet, C., C. Jaupart, J. C. Mareschal, C. Gariepy, G. Bienfait, and R. Lapointe (1991), Heat flow and structure of the lithosphere in the eastern Canadian Shield, *J. Geophys. Res.*, *96*(B12), doi:10.1002/jgrb.v96.B12;article:article:doi\:10.1029/91JB01020;wgroup:string:Publication.

Planke, S., and O. Eldholm (1994), Seismic response and construction of seaward dipping wedges of flood basalts: Vøring volcanic margin, *99*(B5), 9263–9278, doi:10.1029/94JB00468.

Planke, S., P. A. Symonds, E. Alvestad, and J. Skogseid (2000), Seismic volcanostratigraphy of large-volume basaltic extrusive complexes on rifted margins, *J. Geophys. Res.*, *105*(B8), 19,335–19,351, doi:10.1029/1999JB900005.

Rabinowitz, P. D. (1976), Geophysical study of the continental margin of southern Africa, *Geological Society of America Bulletin*, *87*(11), 1643–1653, doi:10.1130/0016-7606(1976)87{\&}lt;1643:GSOTCM{\&}gt;2.0.CO;2.

Rabinowitz, P. D., and J. LaBrecque (1979), The Mesozoic South Atlantic Ocean and evolution of its continental margins, *J. Geophys. Res.*, *84*(B11), 5973, doi:10.1029/JB084iB11p05973.

Roberts, A. M., G. Yielding, N. J. Kusznir, I. Walker, and D. Dorn-LOPEZ (1993), Mesozoic extension in the North Sea: Constraints from flexural backstripping, forward modelling and fault populations, in *Petroleum Geology Conference Proceedings*, pp. 1123–1136, Badley Earth Sciences Ltd, Horncastle, United Kingdom, Geological Society of London, doi:10.1144/0041123.

Sandwell, D. T., R. D. Müller, W. H. F. Smith, E. Garcia, and R. Francis (2014), New global marine gravity model from CryoSat-2 and Jason-1 reveals buried tectonic structure, *Science*, *346*(6205), 65–67, doi:10.1126/science.1258213.

Saunders, A. D., H. C. Larsen, and S. W. J. Wise (Eds.) (1998), *Proceedings of the Ocean Drilling Program*, *Proceedings of the Ocean Drilling Program*, vol. 152, Ocean Drilling Program, doi:10.2973/odp.proc.sr.152.1998.

Scientific Party (1976), Sites 338-343, in *Initial Reports, DSDP*, pp. 151–387, US Govt. Printing Office.

Scientific Party (1984), Sites 552-553, in *Initial Reports*, pp. 31–233.

Scientific Party (1987), Proceedings Ocean Drilling Program Initial Report 104, *Tech. rep.*

Self, S., A. E. Jay, M. Widdowson, and L. P. Keszthelyi (2008), Correlation of the Deccan and Rajahmundry Trap lavas: Are these the longest and largest lava flows on Earth?, *Journal of Volcanology and Geothermal Research*, *172*(1-2), 3–19, doi:10.1016/j.jvolgeores.2006.11.012.

Self, S., A. Schmidt, and T. A. Mather (2014), Emplacement characteristics, time scales, and volcanic gas release rates of continental flood basalt eruptions on Earth, *Special Paper of the Geological Society of America*, *505*, 319–337, doi: 10.1130/2014.2505(16).

Simpson, R. W., and R. C. Jachens (1989), Chapter 3: Gravity methods in regional studies, in *Memoir of the Geological Society of America*, pp. 35–44, Geological Society of America, doi:10.1130/MEM172-p35.

Simpson, R. W., R. C. Jachens, R. J. Blakely, and R. W. Saltus (1986), A new isostatic residual gravity map of the conterminous United States with a discussion on the significance of isostatic residual anomalies, *J. Geophys. Res.*, *91*(B8), 8348, doi:10.1029/jb091ib08p08348.

Smallwood, J. R., R. K. Staples, K. R. Richardson, and R. S. White (1999), Crust generated above the Iceland mantle plume: From continental rift to oceanic spreading center, *J. Geophys. Res.*, *104*(B10), 22,885–22,902, doi: 10.1029/1999JB900176.

Stewart, J., A. B. Watts, and J. G. Bagguley (2000), Three-dimensional subsidence analysis and gravity modelling of the continental margin offshore Namibia, *Geophys. J. Int.*

Talwani, M., and O. Eldholm (1973), Boundary between continental and oceanic crust at the margin of rifted continents, *Nature*, *241*(5388), 325–330, doi:10.1038/241325a0.

Talwani, M., J. L. Worzel, and M. Landisman (1959), Rapid gravity computations for two-dimensional bodies with application to the Mendocino submarine fracture zone, *J. Geophys. Res.*, *64*(1), 49–59, doi:10.1029/jz064i001p00049.

Talwani, M., G. Udintsev, and et al. (1976), *Initial Reports of the Deep Sea Drilling Project*, *Initial Reports of the Deep Sea Drilling Project*, vol. 38, U.S. Government Printing Office, doi:10.2973/dsdp.proc.38.1976.

Teagle, D. A. H., and J. C. Alt (1999), *Data Report: Alteration and Vein Log of Holes 917a and 918d, Southeast Greenland Margin* , in *Proceedings of the Ocean Drilling Program, 163 Scientific Results*, Ocean Drilling Program, doi:10.2973/odp.proc.sr.163.116.1999.

Tegner, C., and R. A. Duncan (1999), 40Ar–39Ar chronology for the volcanic history of the Southeast Greenland rifted margin, in *Proceedings of the Ocean Drilling Program, 163 Scientific Results*, pp. 1–10, Ocean Drilling Program, doi:10.2973/odp.proc.sr.163.108.1999.

ten Brink, U. S., and A. B. Watts (1985), Seismic Stratigraphy of the Flexural Moat Flanking the Hawaiian-Islands, *Nature*, *317*(6036), 421–424, doi:10.1038/317421a0.

Torsvik, T. H., R. D. Müller, R. Van der Voo, B. Steinberger, and C. Gaina (2008), Global plate motion frames: Toward a unified model, *Rev. Geophys.*, *46*(3), RG3004–44, doi:10.1029/2007RG000227.

Uieda, L., and V. C. Oliveira Jr (2013), Modeling the Earth with Fatiando a Terra, doi:10.6084/m9.figshare.708390.

Van Ark, E., and J. Lin (2004), Time variation in igneous volume flux of the Hawaii-Emperor hot spot seamount chain, *J. Geophys. Res.*, *109*(B11), 1–18, doi:10.1029/2003JB002949.

Walcott, R. I. (1970), Flexural rigidity, thickness, and viscosity of the lithosphere, *J. Geophys. Res.*, *75*(20), 3941–3954, doi:10.1029/jb075i020p03941.

Walker, G. P. L., A. T. Huntingdon, A. T. Sanders, and J. L. Dinsdale (1973), Lengths of Lava Flows [and Discussion], *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *274*(1238), 107–118, doi:10.1098/rsta.1973.0030.

Watts, A. B. (1978), An analysis of isostasy in the world's oceans 1. Hawaiian-Emperor Seamount Chain, *J. Geophys. Res.*, *83*(B12), 5989–6004, doi:10.1029/JB083iB12p05989.

Watts, A. B. (1981), The US Atlantic continental margin: subsidence history, crustal structure and thermal evolution.

Watts, A. B. (1988), Gravity anomalies, crustal structure and flexure of the lithosphere at the Baltimore Canyon Trough, *Earth and Planetary Science Letters*, *89*(2), 221–238, doi:10.1016/0012-821X(88)90174-4.

Watts, A. B. (2001), *Isostasy and Flexure of the Lithosphere*, Cambridge University Press.

Watts, A. B. (2018), The use of object-oriented and process-oriented methods for gravity anomaly modeling of sedimentary basins, *Geophys. J. Int.*, pp. 1–21, doi:10.1093/gji/ggy353/5078869.

Watts, A. B., and E. B. Burov (2003), Lithospheric strength and its relationship to the elastic and seismogenic layer thickness, *Earth and Planetary Science Letters*, *213*(1-2), 113–131, doi:10.1016/S0012-821X(03)00289-9.

Watts, A. B., and K. G. Cox (1989), The Deccan Traps: an interpretation in terms of progressive lithospheric flexure in response to a migrating load, *Earth and Planetary Science Letters*, *93*(1), 85–97, doi:10.1016/0012-821x(89)90186-6.

Watts, A. B., and J. D. Fairhead (1997), Gravity anomalies and magmatism along the western continental margin of the British Isles, *Journal of the Geological Society*, *154*(3), 523–529, doi:10.1144/gsjgs.154.3.0523.

Watts, A. B., and J. D. Fairhead (1999), A process-oriented approach to modeling the gravity signature of continental margins, *The Leading Edge*, *18*(2), 258–263, doi:10.1190/1.1438270.

Watts, A. B., and U. S. ten Brink (1989), Crustal structure, flexure, and subsidence history of the Hawaiian Islands, *J. Geophys. Res.*, *94*(B8), 10,473–10,500, doi:10.1029/jb094ib08p10473.

Watts, A. B., and S. Zhong (2000), Observations of flexure and the rheology of oceanic lithosphere, *Geophys. J. Int.*, *142*(3), 855–875, doi:10.1046/j.1365-246x.2000.00189.x.

Watts, A. B., S. Zhong, and J. Hunter (2013), The Behavior of the Lithosphere on Seismic to Geologic Timescales, *Annu. Rev. Earth Planet. Sci.*, *41*(1), 443–468, doi:10.1146/annurev-earth-042711-105457.

Weatherall, P., K. M. Marks, M. Jakobsson, T. Schmitt, S. Tani, J. E. Arndt, M. Rovere, D. Chayes, V. Ferrini, and R. Wigley (2015), A new digital bathymetric model of the world's oceans, *Earth and Space Science*, *2*(8), 331–345, doi:10.1002/2015EA000107.

Wessel, P. (1992), Thermal stresses and the bimodal distribution of elastic thickness estimates of the oceanic lithosphere, *J. Geophys. Res.*, *97*(B10).

Wessel, P. (1993), A Reexamination of the Flexural Deformation Beneath the Hawaiian-Islands, *J. Geophys. Res. Solid Earth*, *98*(B7), 12,177–12,190, doi:10.1029/93JB00523.

Wessel, P., and J. F. Luis (2017), The GMT/MATLAB Toolbox, *18*(2), 811–823, doi:10.1002/2016GC006723.

White, R. S., L. K. Smith, A. W. Roberts, P. A. F. Christie, N. J. Kusznir, A. M. Roberts, D. Healy, R. Spitzer, A. Chappell, J. D. Eccles, R. Fletcher, N. Hurst, Z. Lunnon, C. J. Parkin, and V. J. Tymms (2008), Lower-crustal intrusion on the North Atlantic continental margin, *Nature*, *452*(7186), 460–464, doi:10.1038/nature06687.

Wickens, H. V., and L. R. McLachlan (1990), The stratigraphy and sedimentology of the reservoir interval of the Kudu 9A-2 and 9A-3 boreholes, *6*, 9–23.

Yilmaz, Ö. (2001), *Seismic Data Analysis*, Processing, Inversion, and Interpretation of Seismic Data, Society of Exploration Geophysicists, doi:10.1190/1.9781560801580.

Yilmaz, Ö. (2012), 2. Deconvolution, in *Seismic Data Analysis*, pp. 159–270, Society of Exploration Geophysicists, doi:10.1190/1.9781560801580.ch2.