Department of Earth Science and Engineering

Imperial College London

Non-linear Cross-equalization of Time-lapse Seismic Surveys using Artificial Neural Networks

Submitted for the degree of Doctor of Philosophy

and Diploma of Imperial College

by

Ibrahim Yahya Zahir Al-Ismaili

June, 2003

BIBL. LONDON. UNIV.

To my parents and siblings

Abstract

Time-lapse seismic interpretations are hindered by the lack of repeatability between different generation surveys. Artificial neural networks (ANNs) can provide a means of cross-equalizing such surveys. Their ability to act as non-linear mapping functions can overcome the limitations of linear algorithms.

Using 4D data acquired over the Oseberg oil/gas field in 1989 and 1992, parallel arrays of ANNs were used to match a single sail line of one survey with a swath in the other with different feathering and geometry. The optimum configuration, in terms of accuracy and runtime, works on normalized pre-stack data, and uses one network per offset gather. The network is a supervised, feed-forward and fully-connected three-layer perceptron trained using resilient propagation with weight decay. The target output of the network is one time sample from a trace from one of the time-lapse surveys, whereas the input consists of geometrical parameters and windows of seismic samples from the nearest coinciding trace, and other nearby traces, from the other survey. ANNs are trained on the whole surveyed area including the reservoir.

The matching achieved by ANNs was demonstrated to be superior to some conventional linear cross-equalization methods. This is partly due to the ANNs' ability to achieve non-linear transforms, and partly to their ability to cater for changes in trace position without relying on direct trace-to-trace correspondence or initial interpolation to a common grid. The cross-equalization scheme operates pre-stack and thus enables analysis such as AVO to be carried out on time-lapse surveys. Furthermore, it has the ability to distinguish between production and non-production anomalies and performs good matching to the non-production anomalies only.

Acknowledgements

After thanking Allah, I would firstly like to thank my supervisor, Prof. Mike Warner, for providing the inspiration upon which this project is based, and for his continuous encouragement and support. I acknowledge and thank Petroleum Development Oman (PDO) for their sponsorship and continuous support; Total and partners for supplying me with the seismic data required to conduct this research; and the University of Stuttgart for providing me with the Stuttgart Neural Network Simulator.

I owe thanks to many staff and colleagues that helped me in one way or another. Jo, Kerry, Richard, and Vincent: thanks for reading through parts of my thesis and giving me your comments. Patrick Alberts: thanks for the "IC" codes and for giving me, together with Paul Davies, a head start with ANNs application in seismic.

I thank all my friends and colleagues here at the College. You helped in making life in London both rich and enjoyable. Mohammed Al-Mazroey: from Chester and A-level to London and PhD, in all you enriched me with help, discussions and friendship. Yathrib, Rifaat, Widad, Azzan, Sultan, Hamed, Al-Gharbi, and Al-Hajri: it is a pleasure knowing you and sharing many memories together. Paul and Mike: your cheery nature and ability to invent ad-hoc games made study pleasant. Paolo: your tendency to buffer your responses, although frustrating at times, is funny. Patrick: I have enjoyed many discussions with you. Caterina and Stuart: we shared PhD frustrations, but also some good times. Among others, thanks also to Jörg, Ivan, Richard, Gareth, Tim, Mustafa, S. Al-Busaidi, Djamel, Vincent, Hoonyol, Saif, Tor, Peggy, and John.

I thank also friends and family outside the College. Special thanks to my cousin Ali, uncle Mohammed, aunt Raiya, uncle Hilal, aunt Nasra, and brother-in-law Zahir.

Last but not least, I would like to thank my family. Mother, father, and siblings: your love and encouragement are a treasure I cherish. I hope I can continue to make you feel proud and share good times with you. Inshirah: may we continue in life together with more love and joy. Daughter Naema: you are truly a bliss, you give me lots of joy and I love you so much. may Allah bless you and us all.

Contents

Abstract 3

Acknowledgements 4

Contents 5

List of Figures 10

List of Tables 22

1 Time-lapse and Cross-equalization **23**

1.1 Time-lapse 24

1.1.1 Seismic reservoir monitoring 24

1.1.2 Petrophysical basis 26

1.1.3 Feasibility 29

1.1.4 Repeatability problem 35

1.2 Cross-equalization 38

1.2.1 Misfit elements 39

1.2.2 Conventional methods 41

1.2.3 Proposed method: cross-equalization using ANNs 49

1.3 Thesis objective and outline 51

1.3.1 Objective and methodology 51

1.3.2 Report outline 52

2 Artificial Neural Networks (ANNs) **54**

2.1 Biological neural networks 55

2.1.1 Structure and information handling 55
2.1.2 Properties of biological neural networks 57
2.1.3 Abstract neuron 58
2.2 Artificial neural networks 59
2.2.1 Perceptron 59
2.2.2 Multi-layer perceptron (MLP) 61
2.3 Learning 69
2.3.1 Patterns 71
2.3.2 Back-propagation 71
2.3.3 Stopping criteria and generalization 75
2.3.4 Problems with back-propagation 78
2.3.5 Resilient propagation (RPROP) 81
2.4 Practical issues 84
2.4.1 Error surface complexity 84
2.4.2 Size and topology 86
2.4.3 Pattern size and choice 87
2.4.4 Computational training time 88
2.4.5 Generalization 88
2.4.6 Pattern pre-processing 90
2.4.7 ANNs in this study 93

3 Time-lapse Seismic Datasets 95

3.1 Oseberg field 95
3.1.1 Structure and stratigraphy 96
3.1.2 Reservoir character 101
3.1.3 Production history and future 102
3.1.4 Time-lapse project at Oseberg 103
3.2 Time-lapse datasets 103
3.2.1 Analysis of the two datasets 103

3.2.2 Summary of previous work on the datasets 117
3.2.3 Repeatability and cross-equalization 125

4 Implementation **126**

4.1 Method 127
4.2 Patterns 131
4.2.1 Traces involved 131
4.2.2 Amount of data used 132
4.2.3 Domain 132
4.3 Topology and ANN design 138
4.4 Pattern pre-processing 142
4.4.1 Normalization 142
4.5 Training 147
4.5.1 Training algorithm 147
4.5.2 Local minima consideration 148
4.5.3 Stopping criteria 152
4.6 From pre-training to application 156
4.6.1 Preparing the seismic datasets 162
4.6.2 Preparing the training patterns and scripts 164
4.6.3 Training ANNs 165
4.6.4 Applying the trained ANNs 168
4.7 Quality control (QC): analysing the results 168
4.7.1 Initial qualitative analysis 168
4.7.2 Further quantitative analysis 169
4.8 Is the network memorizing? 173
4.9 Record: list of conducted experiments 174
4.10 Summary 178

5 Matching the Time-lapse Datasets **179**

5.1 Matching in the wavelet domain 180

5.1.1 Initial tests 180

5.1.2 Excluding a CDP range from training 189

5.1.3 Excluding a box area from training 190

5.1.4 Discussion 195

5.1.5 Final test 198

5.2 Matching in the time domain 199

5.2.1 Configuration 201

5.2.2 Initial tests and results 202

5.2.3 Further test and analysis 202

5.2.4 Conclusion 202

5.3 Analysing ANN's input and increasing performance 205

5.3.1 Geometrical information 205

5.3.2 Comparing different input-configurations 206

5.3.3 Connection weight analysis 207

5.3.4 Assessing the results of the weight analysis 208

5.3.5 Reducing the number of ANNs per offset 212

5.3.6 Final test 212

5.4 ANN-matching in comparison with linear-matching 214

5.4.1 Matching using ANNs 214

5.4.2 Matching using linear filters 221

5.4.3 Quantitative comparison 225

5.4.4 Conclusion 234

5.5 Summary 234

6 Further Investigation on ANN cross-equalization **239**

6.1 Preserving desired anomalies 240

6.1.1 Designing the anomaly 242

6.1.2 Quantifying the anomaly 244
6.1.3 Assessing preservation of a desired 23.9% anomaly 245
6.1.4 Assessing preservation of a desired 14.5% anomaly 254
6.1.5 Assessing preservation of a desired 9.6% anomaly 257
6.1.6 Discussion and conclusion 259
6.2 Post-stack cross-equalization 262
6.2.1 Matching post-stack using ANNs 263
6.2.2 Linear post-stack cross-equalization 267
6.2.3 Post-stack versus pre-stack cross-equalization:
a quantitative analysis 271
6.2.4 Desired anomaly preservation 274
6.2.5 Discussion and conclusions 277
6.3 Summary 282
7 Summary and Conclusions **284**
7.1 Summary 284
7.2 Conclusions 289
7.3 Outlook 290
Bibliography **295**

List of Figures

1.1 Oil discovery by decade 25

1.2 World giant-field's oil discovery 25

1.3 Lab-measured versus Gassmann-calculated effects of fluid displacement on velocity 30

1.4 Effect of time-delay between two traces as seen on 30Hz Ricker wavelet . 40

1.5 Effect of amplitude difference between two traces as seen on 30Hz Ricker wavelet 40

1.6 Effect of phase mismatch between two traces as seen on Ricker wavelets 42

1.7 Effect of spectrum bandwidth difference between two traces as seen on Ricker wavelets 42

1.8 Linear cross-equalization system of two time-lapse traces 47

2.1 Biological neuron 56

2.2 Abstract model of a biological neuron 58

2.3 Perceptron 60

2.4 Multi-layer perceptron — a typical feed-forward ANN 62

2.5 Two conventions for illustrating ANNs 65

2.6 The propagation rule 66

2.7 Sigmoid activation transfer function 67

2.8 Capability of a one, two, and three layer ANN 69

2.9 Patterns for ANNs and their nomenclature 72

2.10 Error propagation in back-propagation training algorithm 75
2.11 Back-propagation training algorithm 76
2.12 Network error and cross-validation stopping criterion 78
2.13 Local minima problem in back-propagation 79
2.14 Plateaus problem in back-propagation 80
2.15 Oscillation problem in back-propagation 81
2.16 Error surface in an MLP 85
2.17 Performance versus ANN's size 87

3.1 Oseberg field location in the Norwegian North Sea 97
3.2 Oseberg field geological map and cross-section 98
3.3 Oseberg field map outlining the three major blocks 99
3.4 Schematic stratigraphic column of Mesozoic reservoir rocks at Oseberg . 100
3.5 Oseberg oil/gas field planned long term production rates 104
3.6 Geometry of the Base and the monitor survey 105
3.7 Plot of receiver position of the base and the monitor survey 107
3.8 Plot comparing offset versus channel number in the two time-lapse surveys 108
3.9 Seismic stack plot of one in-line from the base survey NH8906 109
3.10 Seismic stack plot of one in-line from the monitor survey NH9201 110
3.11 Well log and stack section of the base survey 111
3.12 Well log and stack section of the monitor survey 112
3.13 Comparison between the two time-lapse datasets at CDP 1293 114
3.14 Close comparison between the two time-lapse datasets at the near offset of CDP 1293 115
3.15 Close comparison of the two time-lapse datasets with linear cross-equalized traces shown 116
3.16 Spectra of the two time-lapse datasets at CDP 1293 118
3.17 Spectra of the two time-lapse datasets at CDP 1293 after top mute . . . 119

3.18 Spectra of the two time-lapse datasets at time section 2100–2612 ms of CDP 1293 after top mute 120
3.19 Oseberg time-lapse seismic sections from Johnstad et al. (1995) 123
3.20 Oseberg time-lapse seismic sections from Johnstad et al. (1993) 124

4.1 Cross-equalization using ANN 128
4.2 Schematic stacking chart defining gathers 129
4.3 Each offset gather is matched separately 129
4.4 Schematic stacking chart defining neighbouring traces 133
4.5 Illustration of the input and the output of the network 133
4.6 Cascade algorithm in the wavelet transform 137
4.7 Time series of 64 samples after the application of the wavelet transform 137
4.8 Example of topology size investigations 139
4.9 Matching error versus network size 141
4.10 Error as a function of the normalization scheme 143
4.11 Error as a function of the non-clipped normalization schemes 144
4.12 Error graph of non-clipped normalization (channel 121) 146
4.13 Error plot comparing three different weight initialization for many tests 149
4.14 Error graph comparing the effect of weight jogging 151
4.15 Difference plot when using a 3:1 ratio of training to validation patterns 153
4.16 Difference plot when using a 1:1 ratio of training to validation patterns 153
4.17 Error progress during training in the early stages of this study 155
4.18 Error progress during training in the later stages of this study 155
4.19 Graph of Figure 4.18 re-plotted at different scale and training is continued further 157
4.20 Zoom-in on the final error progress shown in Figure 4.19 plotted in linear scale 157
4.21 Offset gather (channel 121) of the base survey for comparison with Figures 4.22–4.28 158

4.22 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 20 training cycles 158
4.23 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 50 training cycles 159
4.24 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 100 training cycles 159
4.25 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 500 training cycles 160
4.26 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 1000 training cycles 160
4.27 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 2000 training cycles 161
4.28 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after 3000 training cycles 161
4.29 Difference plot between Figure 4.21 and the cross-equalized gather from the monitor survey after reaching the lowest validation error 162
4.30 Description of structural charts 163
4.31 Structural chart of the program used to prepare training pattern 166
4.32 Structural chart of the program used to write all necessary files for training 167
4.33 Structural chart of the network training 169
4.34 Structural chart of the application of the networks after training 170
4.35 Effect of variation between two seismic sections on predictability and NRMS 172
4.36 Output of a network trained to match the base survey gather of channel 121 using random traces as input 175
4.37 Cross-equalized gather (channel 121) from the monitor survey after training the network to match the base survey gather (Figure 4.21) using input as defined in Figure 4.5 175

5.1 Data used in Section 5.1.1 182
5.2 Seismic stack plot of data considered in Figure 5.1 taken from the base survey 182
5.3 ANN topology used in the initial test 183
5.4 MSE_{valid} versus offset matched versus time 185
5.5 Seismic stack plot of data considered in Figure 5.1 taken from the monitor survey 186
5.6 Stack plot of the monitor survey after post-stack linear matching with the base survey 186
5.7 Difference plot between Figure 5.2 and Figure 5.6 187
5.8 Stack plot of the monitor survey after pre-stack cross-equalization using ANNs 188
5.9 Difference plot between Figure 5.2 and Figure 5.8 188
5.10 Interpolating over a CDP range 190
5.11 Difference stack plot between the base survey (Figure 5.2) and the ANN cross-equalized monitor survey after excluding a CDP range 191
5.12 Interpolating over a box area 191
5.13 Error graph comparing performance of a network allocated to match a full scale to the performance of networks allocated to match individual coefficients within the scale 193
5.14 Interpolating over a box area after solving the boundary problem 195
5.15 Reproducing part of a time series from a sub-set of a wavelet series . . . 196
5.16 Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor after excluding a box from training and allowing for neighbouring input data 197
5.17 Plot of the wavelet coefficients illustrating those that are matched by ANNs 200

5.18 Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor after setting one ANN per scale and excluding no area from training 201
5.19 Difference plot of the offset gather corresponding to channel 225 between the base survey and the cross-equalized monitor survey matched using the wavelet domain 203
5.20 Difference plot of the offset gather corresponding to channel 225 between the base survey and the cross-equalized monitor survey matched using the time domain 203
5.21 Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor using the time domain 204
5.22 Plot comparing predictability and NRMS of the two matching schemes: time domain scheme versus wavelet domain scheme 204
5.23 Error plot comparison between including and excluding geometry information within input patterns 206
5.24 Error plot comparison between the different input configurations shown in Table 5.1 208
5.25 Analysis of the connection weights of a trained ANN that used the configuration of the "Window D" experiment 209
5.26 ANN topology used in the "Window D" experiment 210
5.27 Error plot comparison between different input configurations before and after using only common-offset traces for input 211
5.28 Diagram shows how neighbouring traces from the same offset gather scan an area of geology 213
5.29 Error plot comparison between different input configurations before and after using only common-offset traces for input and one ANN within 512 ms 213

5.30 Difference between the base survey stack and the stack of the cross-equalized monitor survey when only common-offset traces are used for input and one ANN within 512 ms 215
5.31 ANN topology used in the final matching test 215
5.32 The seismic input-output to the network during training to perform pre-stack cross-equalization 217
5.33 Seismic stack plot of all the offset gathers of channels 100–230 of the base survey 218
5.34 Seismic stack plot of the cross-equalized monitor survey using the best configuration 219
5.35 Difference between the base survey stack (Figure 5.33) and the stack of the cross-equalized monitor survey using the best configuration (Figure 5.34) 220
5.36 Spectrum of the ANN cross-equalized monitor survey in comparison with that of the base survey at pre-stack CDP 1293 222
5.37 Pre-stack CDP 1293 of the ANN cross-equalized monitor survey in comparison with that of the base survey 223
5.38 Difference between the base survey stack and the stack of the cross-equalized monitor survey using linear cross-equalization method with local mode 226
5.39 Difference between the base survey stack and the stack of the cross-equalized monitor survey using linear cross-equalization method with global mode 227
5.40 Difference between the base survey stack and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 51 traces 228
5.41 Close comparison between the different datasets at the near offset of CDP 1293 229

5.42 Graph of predictability between the cross-equalized monitor stack traces and the base survey stack traces 231
5.43 Graph of normalized root mean square (NRMS) between the cross-equalized monitor stack traces and the base survey stack traces 232
5.44 Graph of residual root mean square (RRMS) between the cross-equalized monitor stack traces and the base survey stack traces 233
5.45 Cross-plot of predictability versus NRMS between the ANN cross-equalized monitor stack traces and the base survey stack traces 235
5.46 Cross-plot of predictability versus NRMS between the local linear cross-equalized monitor stack traces and the base survey stack traces 236
5.47 Cross-plot of predictability versus NRMS between the global linear cross-equalized monitor stack traces and the base survey stack traces 237

6.1 Location of the introduced synthetic anomaly on the base survey stack . 242
6.2 Structural chart showing how the synthetic anomaly is designed and applied to the base survey 243
6.3 Seismic stack plot of all the offset gathers of channels 100–230 of the original base survey (*BaseDataset*) 246
6.4 Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with a 23.9% anomaly (*NewBaseDataset*) 246
6.5 Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.4), showing the introduced 23.9% anomaly 247
6.6 Seismic stack plot of the *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 23.9% anomaly . 247
6.7 Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.4) and the *NewXEqMonitorDataset* stack (Figure 6.6) 248
6.8 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 23.9% 249

6.9 Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with local mode 250

6.10 Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 11 traces 250

6.11 Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 21 traces 251

6.12 Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 51 traces 251

6.13 Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with global mode 252

6.14 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when linear cross-equalization is used and an anomaly of 23.9% is introduced 253

6.15 Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with 14.5% anomaly 255

6.16 Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.15), showing the introduced 14.5% anomaly 255

6.17 Seismic stack plot of the *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 14.5% anomaly . 256

6.18 Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.15) and the *NewXEqMonitorDataset* stack (Figure 6.17) 256

6.19 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 14.5% 257

6.20 Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with 9.6% anomaly 258

6.21 Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.20), showing the introduced 9.6% anomaly 258

6.22 Seismic stack plot of the *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 9.6% anomaly 260

6.23 Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.20) and the *NewXEqMonitorDataset* stack (Figure 6.22) 260

6.24 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 9.6% 261

6.25 The seismic input-output to the network during training to perform post-stack cross-equalization 264

6.26 Seismic stack plot of all the offset gathers of channels 1–230 of the base survey 265

6.27 Difference between the base survey stack (Figure 6.26) and the stack of the ANN post-stack cross-equalized monitor cross-equalized monitor survey 265

6.28 Figure 5.35 re-plotted to the same scale as Figure 6.27 for comparison . 266

6.29 Difference between the base survey stack (Figure 6.26) and the stack of the ANN post-stack cross-equalized monitor cross-equalized monitor survey using one in-line of the monitor 266

6.30 Graph comparing predictability between the cross-equalized monitor stack and the base survey stack of different scheme configurations 268

6.31 Graph comparing normalized root mean square (NRMS) between the cross-equalized monitor stack and the base survey stack of different scheme configurations 269

6.32 Difference between the base survey stack (Figure 6.26) and the stack of the local linear post-stack cross-equalized monitor survey 269
6.33 Difference between the base survey stack (Figure 6.26) and the stack of the semi-global (51 traces) linear post-stack cross-equalized monitor survey 270
6.34 Difference between the base survey stack (Figure 6.26) and the stack of the global linear post-stack cross-equalized monitor survey 270
6.35 Comparison of normalized and residual root mean square (NRMS and RRMS, respectively) between the different modes of the linear post-stack cross-equalization 271
6.36 Figure 5.38 re-plotted to the same scale as Figure 6.32 for comparison . 272
6.37 Graph comparing predictability, between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization 272
6.38 Graph comparing normalized root mean square (NRMS), between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization 273
6.39 Graph comparing residual root mean square (RRMS), between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization 275
6.40 Seismic stack plot of all the offset gathers of channels 1–230 of the base survey with added 12.8% anomaly (*NewBaseDataset*) 275
6.41 Difference plot, between the original base survey (*BaseDataset*) stack (Figure 6.26) and the new base survey (*NewBaseDataset*) stack (Figure 6.40), showing the introduced 12.8% anomaly 276
6.42 Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the ANN post-stack cross-equalized monitor survey 276
6.43 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when post-stack cross-equalization is used and an anomaly of 12.8% is introduced 278

6.44 Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the local linear post-stack cross-equalized monitor survey 278

6.45 Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the semi-global (51 traces) linear post-stack cross-equalized monitor survey 279

6.46 Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the global linear post-stack cross-equalized monitor survey . . . 279

6.47 Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when linear post-stack cross-equalization is used and an anomaly of 12.8% is introduced 280

List of Tables

1.1 Rock physics rules of thumb 31
1.2 Properties that might impact 4D feasibility 33
1.3 Physical and seismic criteria required to make 4D feasible 34
1.4 Feasibility of seismic monitoring of water floods 36
1.5 Time-lapse intersurvey acquisition and processing variations 36

2.1 Advantages of both stochastic and batch learning mode 74

3.1 Survey parameters of the base and the monitor surveys 106
3.2 Processing flow for stacking the datasets 113

4.1 Tests conducted on topology size 140
4.2 Actual topology used in the projects defined in Table 4.1 141
4.3 Tests conducted on normalization 143
4.4 Tests conducted on weight jogging 150

5.1 Traces involved in the four input-configurations tested in the "Window" project 207

6.1 Definition of several important datasets used in Chapter 6 241

Chapter 1

Time-lapse and Cross-equalization

The time-lapse seismic method consists of two or more seismic surveys acquired over the same area at different calendar dates (Nur, 1989; Lumley, 1995). The datasets are calibrated, compared, and the inter-survey differences interpreted as changes in fluid content or pressure. This enables flood patterns to be monitored so that the timing and location of further injection and production wells can be planned for maximum benefit (Jack, 1997).

Time-lapse seismic can provide a valuable tool to the oil industry, but is in many cases hindered by lack of repeatability since differences related to acquisition and processing are likely to be significant (Ross and Altan, 1997). Cross-equalization is a method that seeks to normalize acquisition and processing differences between time-lapse surveys (Ross et al., 1996). In so doing, this will improve the use of time-lapse and bridge its limitations. However, cross-equalization for time-lapse seismic is a relatively new approach and no satisfactory general method has yet been established (Rickett and Lumley, 2001).

The aim of this research is to assess the ability of large parallel arrays of artificial neural networks to perform cross-equalization of time-lapse seismic datasets. The

study is conducted on two time-lapse swaths[1] over the Oseberg field in the North Sea. The following sections will briefly introduce time-lapse seismic, cross-equalization, and artificial neural networks as a cross-equalization tool. The chapter ends with a section defining the objective of this research and the structure of this thesis.

1.1 Time-lapse

1.1.1 Seismic reservoir monitoring

The world's hydrocarbon reserves are finite. However, a more striking fact is that there is clear evidence to suggest that global oil field discoveries have passed their peak and that most giant fields have already been discovered (Figure 1.1 and Figure 1.2). Therefore, it is no surprise that better management of existing reserves is needed.

Reservoir management is defined by Robertson (1989) as *"maximizing the economic value of a reservoir by optimizing recovery of hydrocarbons while minimizing capital operating expenses"*. In most oil fields, primary methods that use natural reservoir energy, and secondary methods that augment natural energy by fluid injection, leave behind more than one-third of the original oil in place (Satter and Thakur, 1994). Thus, even if reserves turn out to be significantly higher than currently thought, maximizing recovery and minimizing cost is still a concern and enough reason for better reservoir management.

In addition to secondary production, optimizing recovery requires the use of *enhanced oil recovery* methods (thermal and non-thermal) that are designed to tap the enormous resources left by primary production. In all the three stages (primary, secondary, and enhanced recovery), precise knowledge of the location of the fluids used to enhance recovery and hydrocarbons is required. However, even with detailed reservoir models, engineers do not know where a fluid front is until it arrives at the wells (Sonneland et al., 1997). Fortunately, geophysical techniques have improved over the

[1] swath: A strip of terrain recorded by sensor or sensors at one time or during one sweep (Sheriff, 1997). In our case it is one pass of the ship in a 3D marine seismic survey.

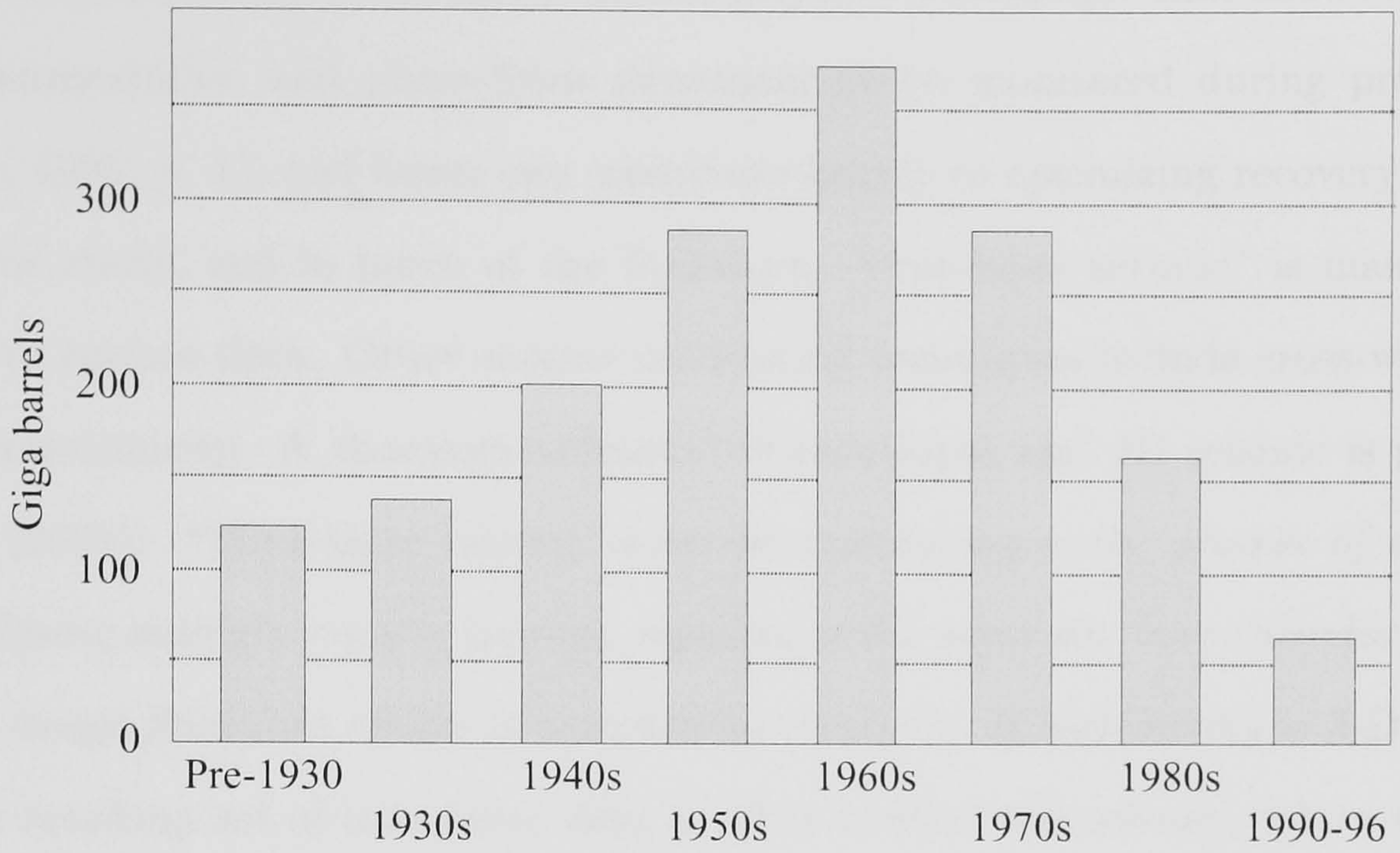


Figure 1.1: Oil discovery by decade given in giga (billion) barrels. Note that discoveries peaked in the 1960s. Redrawn from Campbell (1997, *p. 52*).

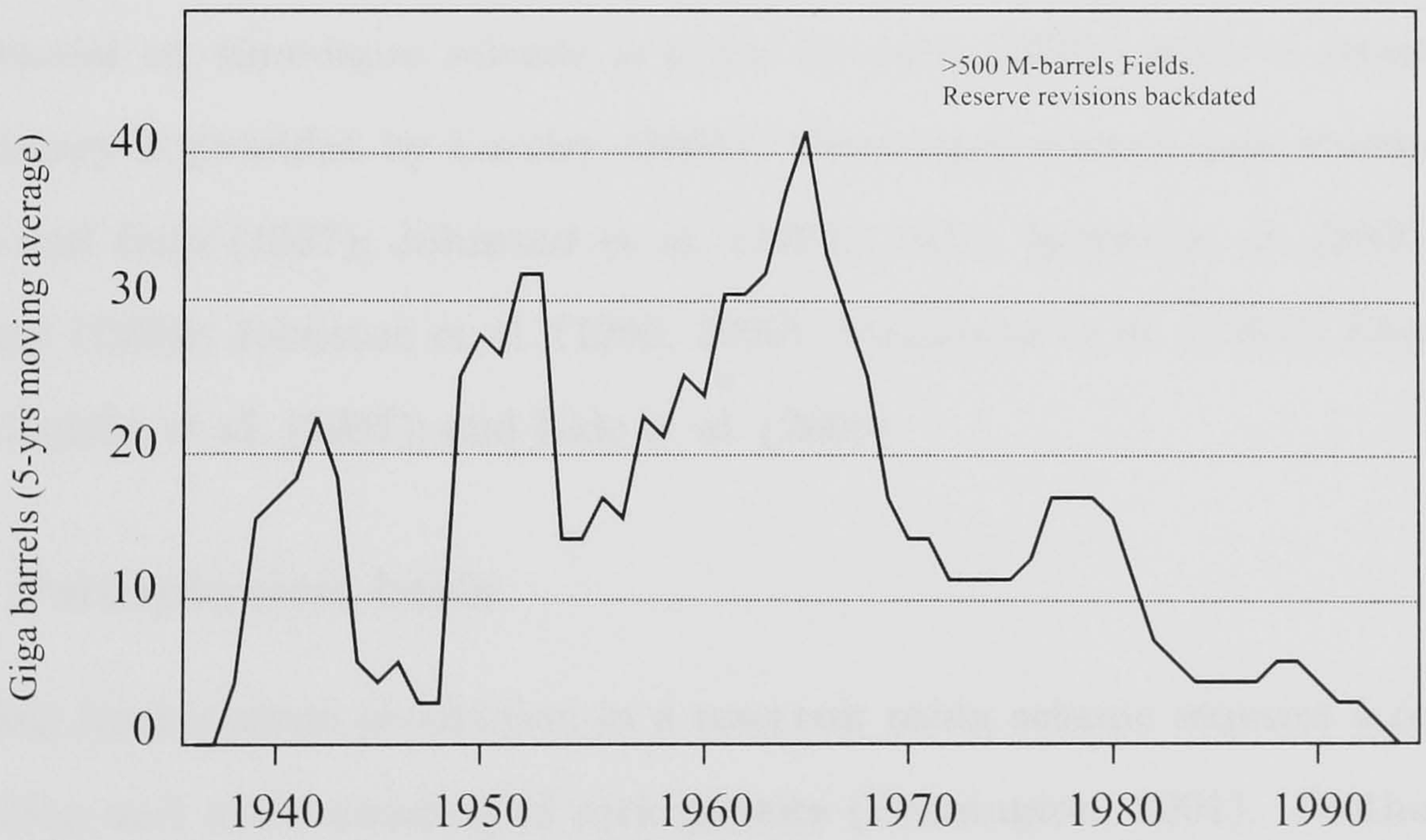


Figure 1.2: World giant-field's oil discovery graph. Discoveries peaked in the 1960s as in Figure 1.1. Redrawn from Campbell (1997, *p. 52*).

years, and thus increasingly provided a sufficient level of detail for fluid fronts to be monitored. In particular, *time-lapse seismic* provides a technique that allows fluid-flow paths, permeability, and phase-front movement to be monitored during production (Lumley, 1995, *p. 1*), and hence can contribute hugely to optimizing recovery.

In this study, and in much of the literature, "time-lapse seismic" is mainly used to refer to surface data. Other seismic monitoring techniques include cross-well, VSP and microseismicity. A thorough definition of time-lapse and 4D seismic is given by Lumley (2001): *"Time-lapse seismic reservoir monitoring is the process of acquiring and analysing multiple seismic surveys, repeated at the same site over calendar time, in order to image fluid-flow effects in a producing reservoir. If each survey is* 3-D seismic, *then the resulting set of time-lapse data is often termed* 4-D seismic, *where the extra fourth dimension is calendar time"*. Gringarten (1998) defines reservoir management as the application of available technology and knowledge to a reservoir system in order to control operations and maximize economic recovery. Hence, time-lapse seismic is a direct tool for reservoir management and is the principal tool in what is known as *Reservoir Geophysics.*

A tutorial on time-lapse seismic is given by Jack (1997), while a recent review and summary is provided by Lumley (2001). Time-lapse seismic case studies include Greaves and Fulp (1987); Johnstad et al. (1993, 1995); Koster et al. (2000); Harris and Henry (1998); Johnston et al. (1998, 2000); Sonneland et al. (1997); Ebrom et al. (1998); Jenkins et al. (1997); and Elde et al. (2000).

1.1.2 Petrophysical basis

Monitoring hydrocarbon production in a reservoir using seismic requires a reasonable appreciation and understanding of rock physics (Pennington, 2001). Furthermore, a good understanding of the effect of the physical properties of pore fluids and reservoir rocks on seismic waves is required. This understanding is required to enable time-lapse seismic to differentiate between the competing reservoir models sufficiently well

to be worth the cost and effort (Pennington, 1997). A classical paper on this subject is Batzle and Wang (1992). Rock physics and its impact specifically on time-lapse seismic is discussed in the classical paper by Nur (1989) and later in Wang (1997). Other studies and analysis of seismic monitoring, and its petrophysical basis. include Wang et al. (1991, 1998), Mavko and Mukerji (1998), and Knight et al. (1998). More extensive work on rock physics is published in Mavko et al. (1998). A good summary and review of recent advances in seismic lithology analysis is given in Castagna (2001). Wang (2001) provides a good tutorial on seismic rock physics. In some cases, seismic rock physics is termed *seismic petrophysics.* The latter is defined by Pennington (1997) as the purposeful application of rock physics theory, as calibrated by laboratory and well measurements, to the interpretation of seismic data.

In seismic petrophysics, time-lapse seismic is concerned particularly with the *fluid substitution problem.* This is defined as the prediction of seismic velocities in rocks saturated with one fluid from rocks saturated with a second fluid or saturated rock velocities from dry rock velocities, and vice versa (Mavko et al., 1998). Fluid substitution formulas are commonly based on Gassmann's (1951) equations (Berryman, 1999). Hence, Gassmann's relations are increasingly used in seismic reservoir monitoring to calculate the effect of fluid substitution on seismic properties using the frame properties (Berryman, 1999; Wang, 2001). The following are the equations as given by Wang (2001), using Mavko et al. (1998) terminology.

Gassmann's equation relates dry rock bulk modulus to saturated rock bulk modulus and states that:

$$K_{sat} = K_{dry} + \frac{(1 - \frac{K_{dry}}{K_0})^2}{\frac{\phi}{K_{fl}} + \frac{1-\phi}{K_0} - \frac{K_{dry}}{K_0{}^2}}, \tag{1.1}$$

where

K_{dry} = effective bulk modulus of dry rock;

K_{sat} = effective bulk modulus of the rock with pore fluid;

K_0 = bulk modulus of rock matrix;

K_{fl} = effective bulk modulus of pore fluid; and

ϕ = porosity.

The shear modulus of the rock is not affected by fluid saturation so that:

$$\mu_{sat} = \mu_{dry}, \tag{1.2}$$

where μ_{sat} is the effective shear modulus of the rock with pore fluid, and μ_{dry} is the effective shear modulus of the dry rock. This relation is a result of Gassmann's analysis and is not, as many confuse, an assumption (Berryman, 1999). The velocities and bulk and shear moduli are related by the following two equations:

$$K_{dry} = \rho_{dry}({V_p}^2 - \frac{4}{3}{V_s}^2), \text{ and} \tag{1.3}$$

$$\mu_{dry} = \rho_{dry}{V_s}^2, \tag{1.4}$$

where V_p is the compressional velocity, V_s is the shear velocity, and ρ_{dry} is the density of the dry rock.

As reported by Wang (2001), Gassmann's equation assume that:

1. the rock is macroscopically homogeneous;

2. all the pores are interconnected or communicating;

3. the pores are filled with a frictionless fluid;

4. the rock-fluid system under study is closed (undrained); and

5. the pore fluid does not interact with the solid in a way that would soften or harden the frame.

The first assumption requires that the rock is isotropic, but the equation has since been extended to the anisotropic case (Wang, 2001). Assumptions (2) and (3) are key assumptions, and imply that the Gassmann's equation is only strictly true for zero frequency. Therefore, the low frequencies used in seismic exploration closely meet the assumptions, while ultrasonic laboratory analysis do not (Mavko et al., 1998, *p. 176*). Assumption (5) emphasizes that the input dry frame bulk modulus to the Gassmann equation should be obtained at irreducible fluid saturation conditions (Wang, 2001).

Figure 1.3 shows an analysis on the correlation of Gassmann-calculated versus lab-measured effects of fluid displacement (water and CO_2 flood) on compressional velocity. The figure shows that the fluid displacement process approximately conforms with the Gassmann assumptions. As a result, when the input frame properties are measured at the irreducible water saturation condition, the Gassmann-predicted and the laboratory-measured effects of fluid displacements on seismic properties might be directly applied to 4D seismic feasibility studies and interpretations (Wang, 2001).

The effect of reservoir physical properties on seismic properties is derived from both the mathematical equations (e.g., Gassmann's) and the empirical trends observed on field and laboratory studies. Table 1.1 illustrates some general rules of thumbs of these effects selected from Wang (2001). Furthermore, many studies exist on the effect of environment (pressure, temperature, and composition) on rock physics (densities, bulk moduli, velocities and viscosities of common pore fluids). These studies enabled many trends and relationships to be noted between the different physical properties and seismic properties. Batzle and Wang (1992) include many of such trends that are widely used in the oil industry (Wang, 2001).

1.1.3 Feasibility

Whether a change in the reservoir state is seismically detectable, defines the *feasibility* of seismic time-lapse reservoir monitoring. The feasibility, aside from financial and business considerations, is dependent on the quality of the seismic data and the magnitude

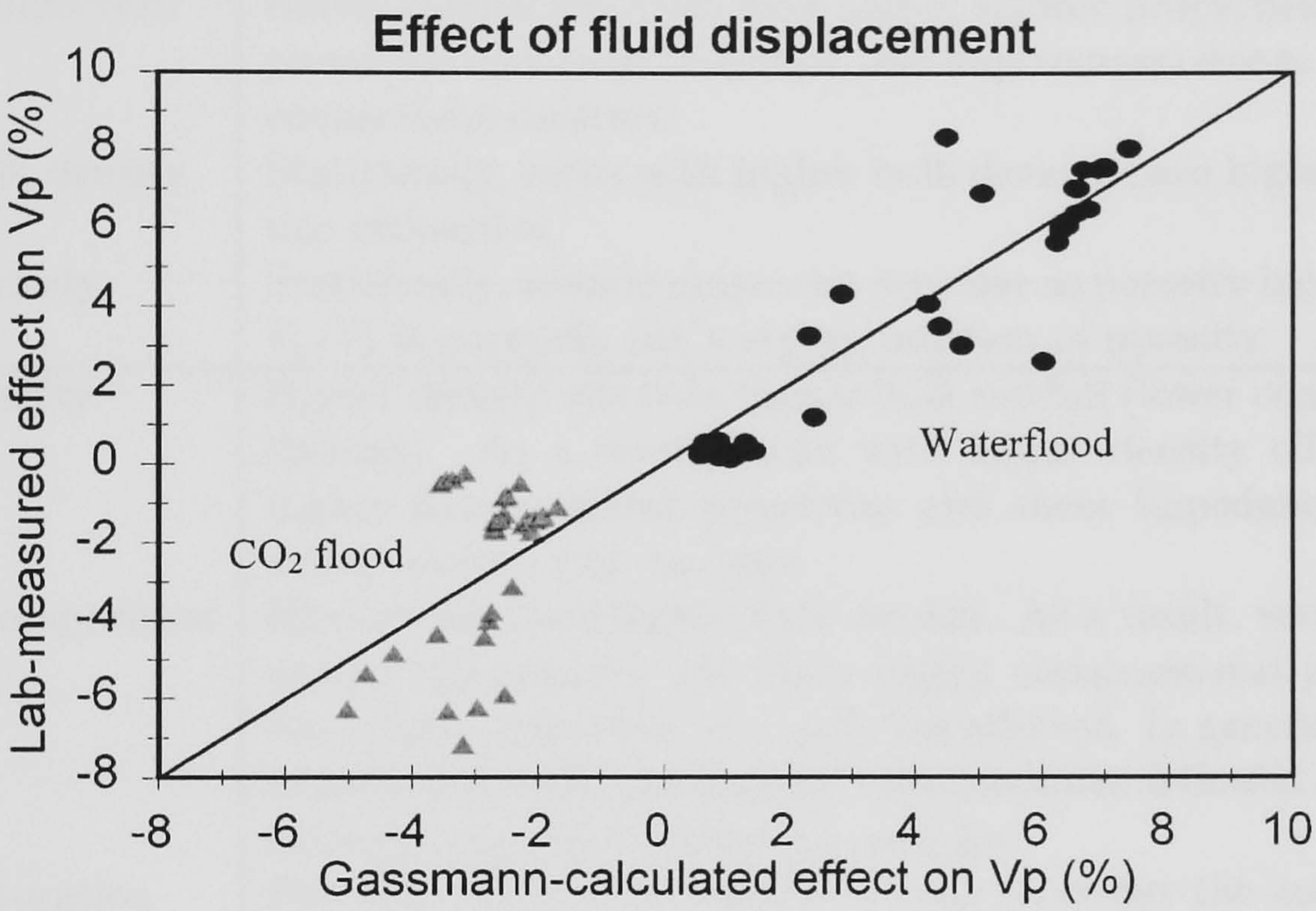


Figure 1.3: Lab-measured versus Gassmann-calculated effects of fluid displacement (water and CO_2 flood) on compressional velocity in sands, sandstones, and dolomites. Solid circles are water-flooded sandstones and sands, triangles are CO_2-flooded dolomites. From Wang (2001). Both axis show percentage change of compressional velocity V_p due to fluid displacement. Note how the two measurements lie roughly on the linear $y = x$ line and thus the fluid displacement process approximately conforms with the Gassmann assumptions. Departure from $y = x$ line could be due to errors in lab-measurements, errors on measurements needed by Gassmann's equations, and genuine departure of Gassmann's calculation from observed measurements due to assumptions not being fully satisfied. Nevertheless, the approximation is sufficient to allow the use of Gassmann's equations in 4D feasibility studies.

Table 1.1: Rock physics rules of thumb, describing the relationship between seismic properties and rock physics. The latter is split into three categories: rock properties, fluid properties, and environment. Each factor is assessed independently and assumes other factors are constant. In reality, many of these factors interact. After Wang (2001, *Tables 3,4, and 5*).

Physical Property	**Effect on seismic property**
Rock Compaction	Better compacted rocks have higher seismic properties (compressional and shear velocities, and impedances) due to better connectivity/contact.
Rock Bulk density	Statistically, rocks with higher bulk density have higher seismic properties.
Rock Porosity	Statistically, seismic properties decrease as porosity increases. V_p/V_s is normally not a strong function of porosity.
Fluid Density	Higher density oils have higher bulk moduli (lower compressibilities). As a result, rocks with higher density oils have higher compressional properties and shear impedance, but shear velocity may be lower.
Fluid composition and type	Heavier oils have higher bulk moduli. As a result, rocks saturated with heavier oils show higher compressional properties. Shear properties are much less affected. In general, high salinity brine has the highest bulk modulus, followed by low salinity brine, fresh water, oil, and gas.
Fluid Saturation	Full saturation of a liquid in a rock increases the compressional seismic properties and shear impedance, but decreases V_s, resulting in an increase in V_p/V_s . When gas is introduced into a fully liquid-saturated rock, the compressional seismic properties and shear impedance all decrease but V_s increases, resulting in a decrease in V_p/V_s. The magnitude of the saturation effect is higher in rocks with weak frames and/or flat pores (cracks, fractures).
Temperature	Dry and water-saturated rocks normally show small seismic property decreases as temperature increases. However, rocks, especially soft sands, saturated with heavy oils show large seismic property decreases as temperature increases.
Net reservoir pressure[a]	Seismic properties in all rocks increase as net reservoir pressure increases. The magnitude of such an increase depends on several other factors (pore shape, porosity, pore fluids, lithology, etc.). It can only be quantified through measurements.

[a]Net reservoir pressure is the difference pressure between overburden and reservoir pressures. Because overburden pressure does not change (or changes very little) in a reservoir, the effect of net reservoir pressure is just the opposite of the effect of reservoir pressure on seismic properties (Wang, 2001).

and nature of the changes in the reservoir. Time-lapse seismic feasibility is discussed in Lumley et al. (1997), Lumley et al. (2000), Ross et al. (1996), Wang (1997), Biondi et al. (1998), Lumley (1995, *ch. 3*), and others. Factors affecting the feasibility can be grouped into three categories:

1. *Financial and business*: Seismic monitoring should be avoided if the benefits are small because the information comes too late, extra wells are too expensive, or the remaining reserves are too small (Koster et al., 2000). There should also be a good likelihood and a plan to use the results of monitoring in reservoir management (Johnston, 1997). In addition, the timing must fit the reservoir management workflow so that a potential intervention can have an economic benefit (Christie, 2000).

2. *Physical*: The effect of physical reservoir properties and production methods on feasibility. The expected physical changes — in either rock or fluids properties, pressure, or temperature — should have the potential to cause a resolvable seismic change.

3. *Seismic*: The effect of seismic survey quality and configuration on feasibility. Seismic methods employed, acquisition design and parameters, and repeatability of the surveys should warrant a good resolvability of expected changes.

Lumley et al. (1997) analysed factors that make a 4D project feasible, and developed a scheme by which one can assess this feasibility prior to undertaking the monitoring surveys. He described both the physical and seismic properties that affect the feasibility of 4D monitoring. The physical properties include all those related to the reservoir, rock, oil, water, gas, and expected 4D fluid changes. Table 1.2 summarizes those properties and gives the ideal case for 4D.

Lumley et al. (1997) selected six physical properties and four seismic properties as the main properties affecting the feasibility of time-lapse reservoir monitoring. His scheme uses those properties to quantitatively assess the feasibility. The six main physi-

Table 1.2: Physical and seismic properties that might impact 4D feasibility with ideal situation, for 4D, indicated. Note that the physical properties include all those related to the reservoir, rock, oil, water, gas and expected 4D fluid changes. After Lumley et al. (1997).

Property	ideal	Property	ideal
Reservoir		**Seismic**	
depth	shallow	dominant frequency	high
overburden pressure	low	avg. resolution (ft)	low
pore pressure	high	image quality	high
net pressure	low	repeatability	high
unit thickness	high	fluid contact visibility	high
temperature	high	mapped travel-time change (samples)	> 4
bubble point	crossed	mapped impedance change (%)	> 4
Rocks		**Water**	
dry bulk modulus	low	salinity	high
dry density	low	density	high
porosity	high	bulk modulus	high
Oil		**Gas**	
gas-to-oil ratio	high	density	low
gravity	high	bulk modulus	low
density	low		
bulk modulus	low		
4D Fluids			
fluid saturation change	high		
fluid compression contrast	high		

cal properties are the rock bulk modulus, fluid compressibility contrast, fluid saturation change, porosity, impedance change, and travel-time change. The four main seismic survey properties are seismic image quality, seismic resolution, observability of the seismic fluid contacts, and seismic repeatability. Table 1.3 lists the criteria set by Lumley et al. (1997) to determine the feasibility of 4D. He points out that criteria related to the physical properties need to favour 4D monitoring before analysing the seismic survey properties and their related criteria.

The feasibility assessment provided by Lumley et al. (1997) is a general one. There are other studies of feasibility on specific type of production. A good example is that of Wang et al. (1991) who conducted a study on the feasibility of seismic monitoring of water floods. His conclusions are summarized in Table 1.4.

Table 1.3: Physical and seismic criteria required to make 4D feasible as outlined by Lumley et al. (1997).

Property	Criterion
Dry rock bulk modulus	This needs to be compressible to allow any fluid change or pressure change to be monitored seismically. A reasonable value is 5–10GPa.
Fluid compressibility contrast	This needs to have contrastingly different compressibility values to be distinguished seismically from each other. A reasonable value is 100–150%.
Fluid saturation change	More change is favourable. A reasonable value is 30–40%.
Porosity	Fluids need to be displaced in large volumes to be observed seismically. Therefore, high rock porosity is required. A reasonable value is 15–25%.
Predicted impedance/traveltime change	This needs to be at least 4% in impedance or 4 samples travel-time.
Seismic image quality	A high-quality and repeatable image is needed for 4D seismic applications.
Seismic resolution	This needs to be high enough not to allow the fluid effects to be averaged out over a seismic wavelength, and become invisible below the background noise.
Seismic fluid contacts	If seismic images show confirmed fluid contacts, then the chances are extremely good that 4D seismic will be able to monitor production fluid flow.
Seismic repeatability	If seismic images are not highly repeatable outside the reservoir zone where no changes are expected, then it will be difficult to convince anyone that changes observed within the reservoir are real and meaningful.

1.1.4 Repeatability problem

In time-lapse seismic, differences are often introduced by the seismic method itself. Variations could be caused by feathering, cable drift, shooting azimuth, source variation, sea state conditions, etc... (Ross et al., 1996). Other non-obvious causes of difference are variations in noise and source wavelet (Bunch and White, 1985a). Hence, two traces representing the same reflection points may differ, even before any processing. In addition to acquisition differences, other differences are introduced by processing.

Table 1.5 illustrates examples of time-lapse differences related to both acquisition and processing. If uncorrected for, these differences can hinder the use of time-lapse and render interpretation ambiguous (Ross and Altan, 1997). The differences between two time-lapse traces can be simplified in terms of static shift, amplitude, phase mismatch, and bandwidth difference, the effect of those four is described later in Section 1.2.1.

The repeatability problem, although new, is well known. Methods employed to overcome the repeatability problem include special sea-bed acquisition (Ebrom et al., 1998), iterative 4D processing of pre-stack data (Elde et al., 2000), joint reprocessing of post-stack data (Huang et al., 1999), and cross-matching and cross-equalization (see e.g. Rickett and Lumley (2001)). All those methods can be summarized and classified into two categories

1. Those that aim to get rid of the repeatability problem by increasing the repeatability during acquisition. This might require new shooting configuration and more expensive apparatus.

2. Those that attempt to match the time-lapse surveys and normalize the differences between them. This is a widely used method, but there are many approaches to it.

Although increasing repeatability can make 4D seismic interpretation reasonable, it is hard to achieve in practice and requires very well planned surveys, ideal circumstances, and a considerable amount of money. In fact, time-lapse surveys can never be

Table 1.4: Feasibility of seismic monitoring of water floods. From Wang et al. (1991).

Reservoir Rocks	**Reservoir Fluids**	**Is seismic monitoring of water flooding feasible?**
Carbonate	Gas	Yes in many reservoirs.
	Light Oil	Probable in some shallow reservoirs.
	Medium Oil	No.
	Heavy Oil	No.
Sandstone	Gas	Yes in most reservoirs.
	Light Oil	Probable in some shallow reservoirs.
	Medium Oil	No in most reservoirs.
	Heavy Oil	No.
Unconsolidated Sand	Gas	Yes in most reservoirs.
	Light Oil	Yes in most reservoirs.
	Medium Oil	Yes in many reservoirs.
	Heavy Oil	No.

Table 1.5: Some time-lapse intersurvey acquisition and processing variations. After Ross and Altan (1997).

Survey Differences	**Examples**
Equipment	Recording technique e.g. OBC vs. streamer, dynamite vs. vibroseis. Recording system. Source/receiver impulse response. Lead-in distance (source to near trace). Source/receiver depths. Offsets recorded (cable length). Shot/group/CMP interval, fold and azimuth.
Geometry	Acquisition orientation. Infill/feathering. Positioning accuracies.
Processing	Algorithm variation / parameterizations. Migration velocities differences.
Quality control	Source consistency. Live/dead trace or channel variations.

fully repeatable and there is always a level of non-repeatable noise even in best-case scenarios (Rickett and Lumley, 2001). In addition, there will always be *legacy surveys* shot before any plan of time-lapse. Such legacy surveys can be very useful in 4D interpretation, but require rigourous matching to enable time-lapse analysis.

There are two main methods for matching time-lapse seismic surveys:

Deterministic Matching: Time-lapse surveys are matched by correcting for known sources of mismatch. Effect of *known* tidal state, equipment response, and survey geometry and parameters can often be corrected. However, the information necessary for these corrections is not always present and sufficient. For example, information about the exact survey parameters of legacy datasets is usually not present, or — if present — not within easy access. In addition, many causes of variation are not easy to decouple, and hence it is difficult to deterministically correct for each cause. Furthermore, even for a known cause, correcting for its effect might not always be possible (e.g. correcting simultaneously for the effect on primary events and multiples might not be possible).

Statistical Matching: A statistical matching filter is designed between the two time-lapse datasets. This removes the necessity of knowing the sources of misfit and has the potential for noise suppression (Bunch and White, 1985a).

It is apparent from the short description that complete deterministic matching is not usually possible, and thus it often offers only a crude first step to cross-matching. Statistical matching is often called *cross-equalization* and is usually applied as a final step in cross-matching.

In most time-lapse datasets, there are simple deterministic corrections that can easily be applied but those are not sufficient to closely match the two time-lapse surveys. Therefore, additional methods are employed to cross-match the two surveys and often include designing and applying a statistical matching filter. Hence, cross-equalization is now widely used as a term that includes both statistical and deterministic methods to cross-match time-lapse surveys. Cross-equalization should match the two surveys

outside the reservoir, and leave only production related differences within the reservoir. This brings an extra challenge in time-lapse, namely, how to separate changes in the recorded seismic that are due to production from those that are not.

1.2 Cross-equalization

Cross-equalization has long been used as a tool to match two intersecting seismic profiles. In general, *cross-equalization consists of designing a filter that, when applied to one trace, makes it look like another* (Bunch and White, 1985a). There are three main purposes for using cross-equalization: to check the position of line intersections, to extract a wavelet that matches a well log synthetic seismogram to seismic data, and to find a compensatory filter to match data from two intersecting seismic lines (Bunch and White, 1985a; Thomas, 1999, *p. 49*). In time-lapse seismic, and hence in this study, we are concerned with the last of the aforementioned purposes.

Time-lapse seismic cross-equalization is defined by Ross et al. (1996) as *"the catch-all industry term for the match-filtering, amplitude scaling and static corrections necessary for seismic monitoring"*. In effect, it is a filter designed to match reflection data from one survey to another. To avoid matching and obliterating production related changes, the filter is usually designed over static reflectors that exclude the reservoir. Published case studies of time-lapse cross-equalization include Burkhart et al. (2000), Harris and Henry (1998), Ross and Altan (1997), Druzhinin and MacBeth (2001), Rickett and Lumley (1998, 2001).

This section describes the main misfit elements that should be addressed by seismic cross-equalization. This is followed by a summary of the conventional methods employed to match two traces (or datasets). Finally, an alternative method, which has the potential for carrying out the necessary cross-equalization with added advantages, is proposed.

1.2.1 Misfit elements

When two traces imaging the same subsurface profile differ, there are four elements that require correction and hence should be addressed by cross-equalization. These are timing or static shift, amplitude or RMS energy difference, phase mismatch, and bandwidth difference (Ross et al., 1996). Each of these, if not corrected, can cause a significant residual energy in the time-lapse difference section, where residual energy of as low as 5% can easily obscure production related anomalies.

The following is a description of how each element might affect the difference observed between two traces as outlined by Ross et al. (1996):

Timing shift: This is easily caused by differences in sea state (temperature and salinity), tide, and acquisition geometry. Differences of as low as 2 ms (1.5 m at sea-water velocity of around 1500 ms^{-1}) can cause a considerable amplitude difference as illustrated in Figure 1.4.

Amplitude difference: This occurs due to differences in source wavelet and in instruments and their responses. Figure 1.5 illustrates the impact of amplitude variation on the difference between seismic traces. In general, a global amplitude scaling is performed between two time-lapse surveys, but often a more complex time- and/or trace-variant scaling is required.

Phase mismatch: Source wavelet phase difference is the prime cause of this mismatch. Figure 1.6 shows the impact of phase shift on the difference between seismic traces as illustrated in Ricker wavelets of different frequencies. The impact is approximately independent of frequency, and is significant even at low phase mismatch where 15° causes 20% amplitude difference. Since the human eye finds it difficult to see phase shifts of 15° or less (Ross et al., 1996), it is crucial that any cross-equalization method employed corrects for phase without relying on visual judgement.

Bandwidth difference: Often the bandwidth used in each time-lapse survey differs.

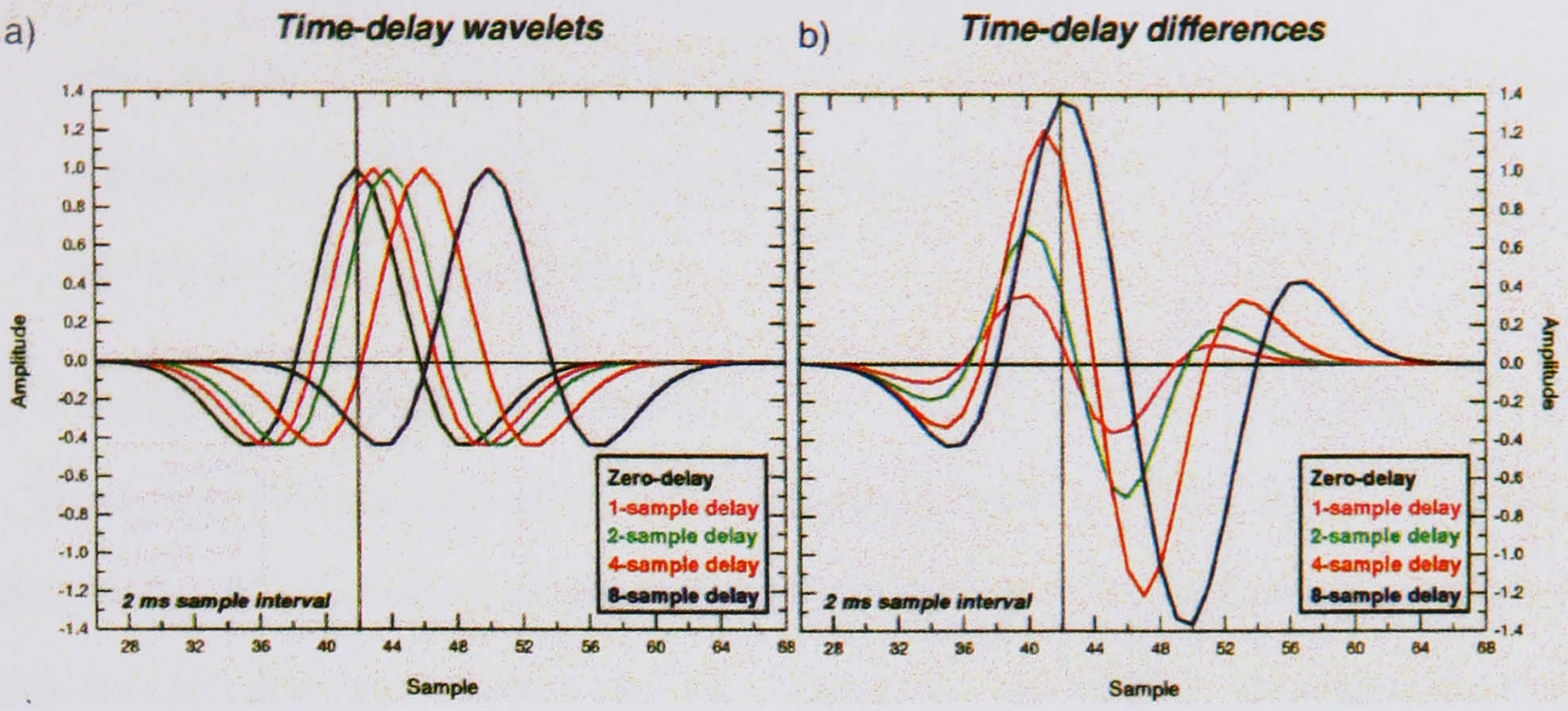


Figure 1.4: a) 30Hz Ricker wavelets with unit amplitude delayed by zero, one, two, four, and eight samples (0, 1, 2, 4, and 8 ms). b) The differences of the zero-delay wavelet and the delayed wavelets proving that significant errors can occur because of time differential between wavelets (or surveys). From Ross et al. (1996).

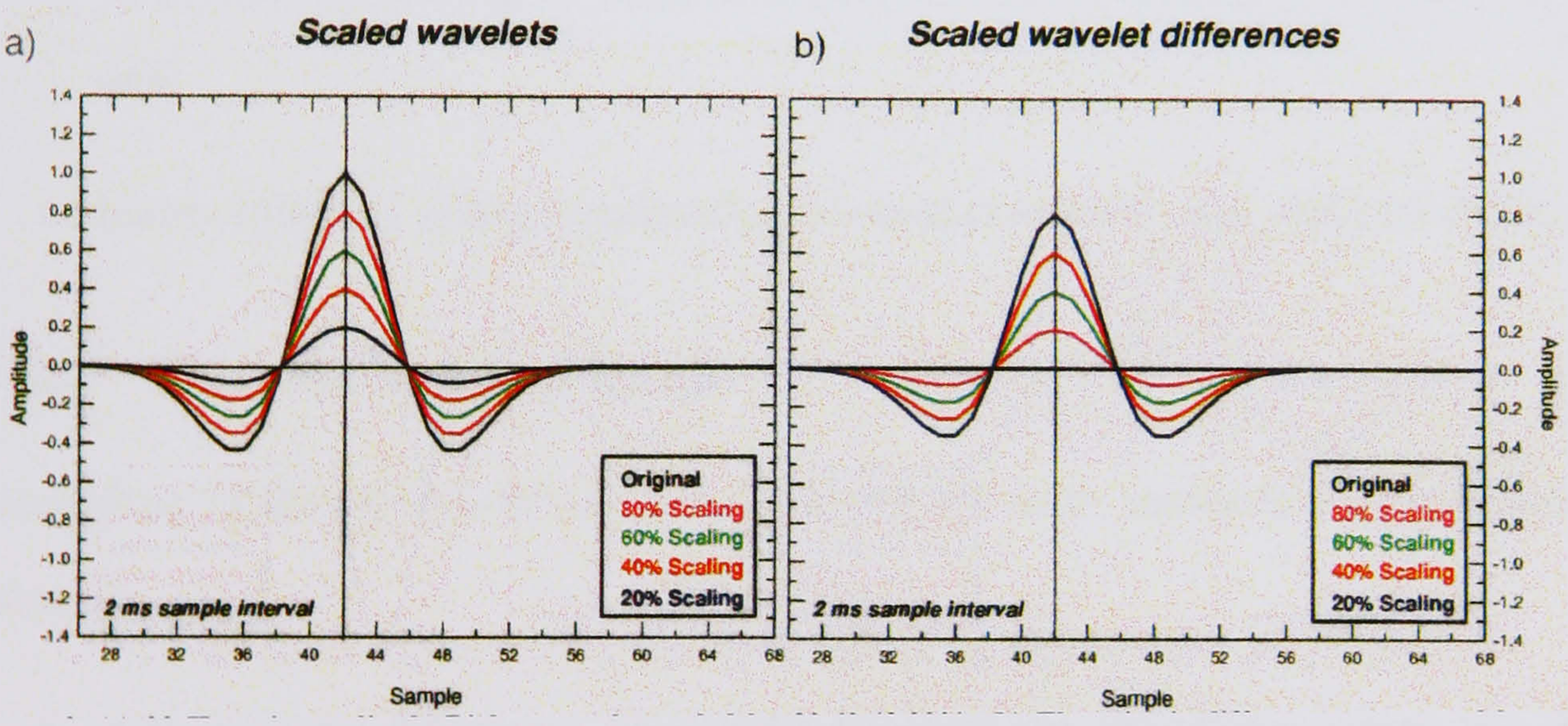


Figure 1.5: a) 30 Hz unit amplitude Ricker wavelet scaled by 20, 40, 60, and 80%. b) The seismic differences generated by subtracting each scaled wavelet from the original unit amplitude wavelet. From Ross et al. (1996).

caused primarily by differences in the source wavelets. This could easily cause residual energy in the difference between the two surveys. Figure 1.7 illustrates the impact of bandwidth differences and shows that the residual reflector energy ranges from 7% to 40% of input amplitude for centre bandwidth differences of 2.5 Hz to 12.5 Hz, respectively.

For a further description consult Ross et al. (1996) and Duranti (2001, *ch. 5*).

In addition to the four mismatch elements described, differences could also be complicated by differences in noise and alignment of the two traces. The latter might be more serious and occur when the two traces do not exactly measure the response of the same profile. Hence, time-lapse cross-equalization typically requires the following elements (Rickett and Lumley, 2001; Johnstad et al., 1995):

1. interpolation or resampling to a common grid to compensate for different geometries and acquisition parameters;
2. temporal (time) shift to compensate for environment state or survey design;
3. amplitude balancing to scale the data to the same amplitude (or energy) level; and
4. bandwidth and phase equalization to compensate for, e.g., different source wavelets.

1.2.2 Conventional methods

According to Bunch and White (1985a) there are three methods for cross-equalization, namely:

1. *deterministic filtering*: This is best suited for situations where the differences are caused by a known origin and in such cases cross-equalization corrects for those known causes. However, deterministic filtering has no capability of noise suppression. In addition, it is usually insufficient and hindered by the lack of sufficient information (see earlier description in Section 1.1.4).

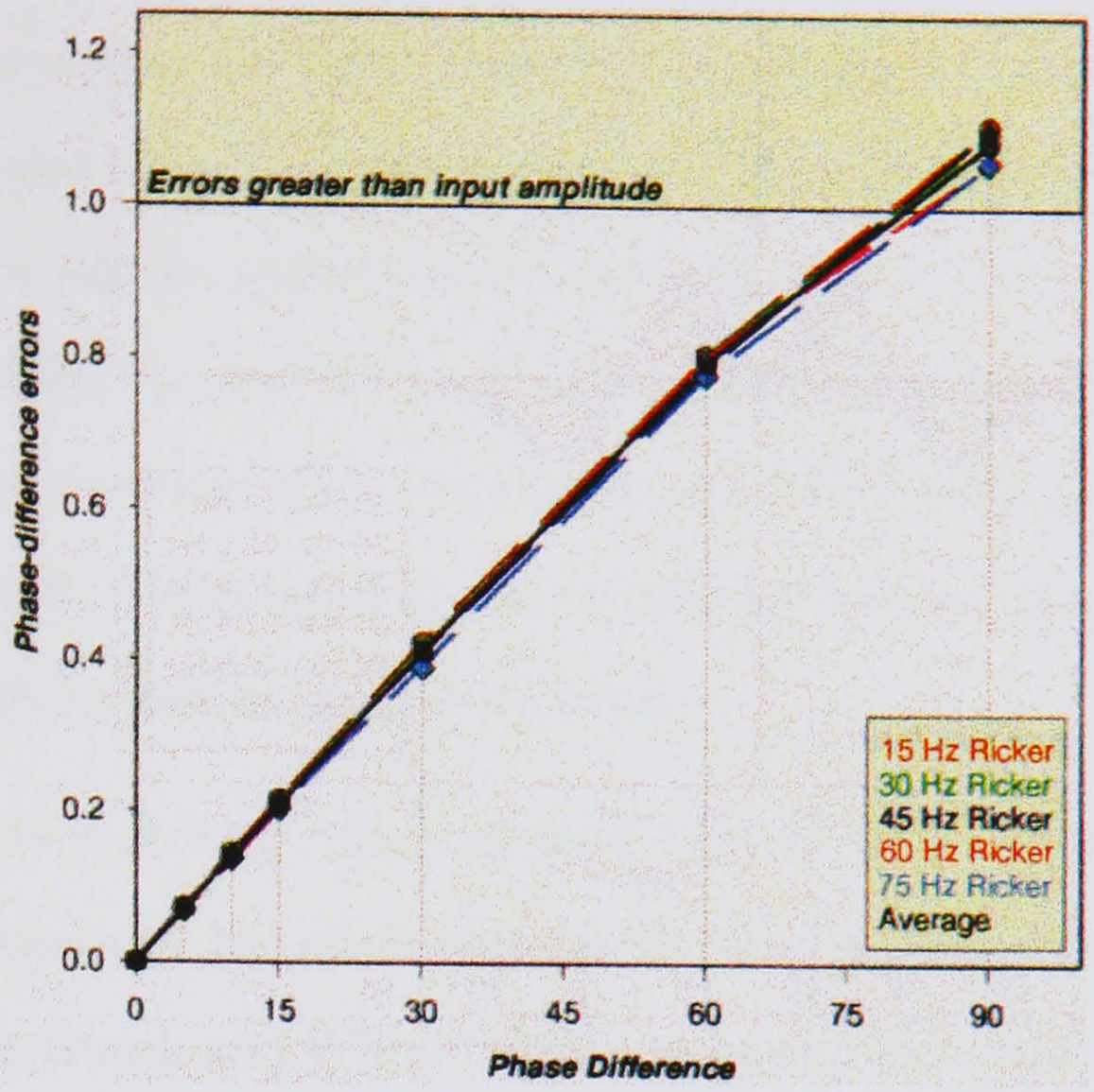


Figure 1.6: Phase difference errors as a function of phase rotation for synthetic Ricker wavelets with centre frequencies of 15-75 Hz (in 15 Hz increments). From Ross et al. (1996).

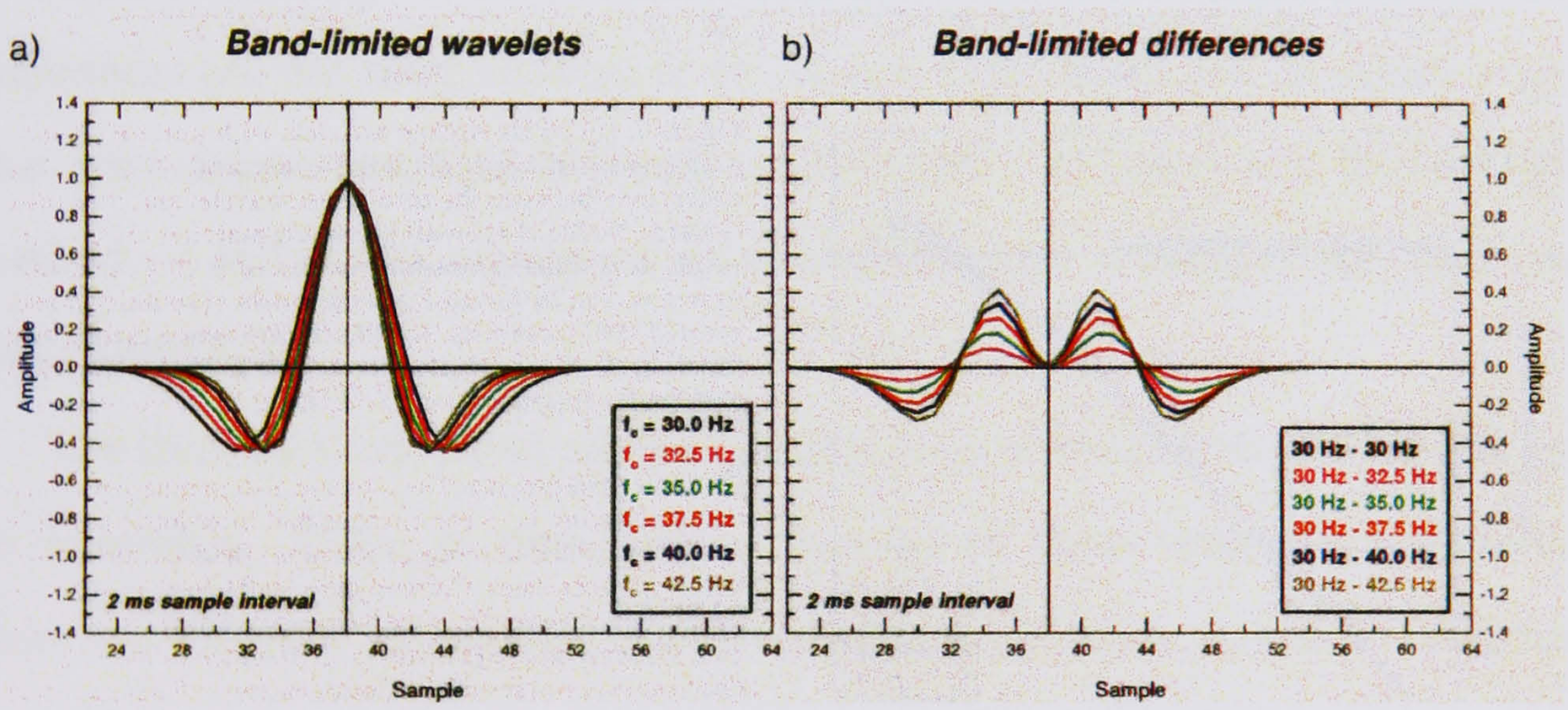


Figure 1.7: a) Unit amplitude wavelets with centre frequencies of 30-42.5Hz in increments of 2.5Hz. The base wavelet has a centre frequency of 30Hz. b) Wavelet differences obtained by subtracting broader bandwidth from the 30 Hz base wavelet. From Ross et al. (1996).

2. *wavelet processing*: Assuming the differences between two seismic traces are due to different source wavelets, this method attempts to remove the effect of the source wavelet using convolution to a zero phase wavelet. However, although the correction is simple, source wavelet determination is not (Hobbs and Jakubowicz, 2000). Usually, the wavelet is extracted assuming a simple isolated reflection can be found, or the trace is the convolution of a white reflection series with minimum phase wavelet. These assumptions usually are not met in practice and hence this method fails to give a satisfactory result.

3. *least-squares matching*: This method tries to create a matching filter that attempts to minimize the misfit between the cross-matched traces. This last method is shown by Bunch and White (1985a) to be superior over the other methods and has a capability of automatic noise suppression. It is further discussed in Bunch and White (1985b).

In time-lapse, differences between the two seismic lines usually have many interacting causes as discussed in Section 1.1.4. Therefore, deterministic methods are often incomplete although feasible in some cases, and wavelet processing is not sufficient since differences are not solely related to the source wavelet. Hence, in addition to the fact that least-square matching is shown to be superior by Bunch and White (1985a), this method — and others relying on error-minimization — remains the main cross-equalizing technique.

The three methods listed address the matching of two traces, but not the particulars of time-lapse. Often in time-lapse, a mixture of these three methods is used, and there are different approaches to error-minimization when creating a matching filter. In addition, the need to realign traces in some time-lapse cases commonly requires interpolating to a common grid. Example of methods employed in time-lapse cross-equalizations include:

- Multi stage matching of Rickett and Lumley (1998, 2001) which include spatial realignment to common grid, least-square match-filtering and wrapping (a tech-

nique outlined in Wolberg (1996)).

- Constrained cross-matching by Druzhinin and MacBeth (2001) which follows a technique developed by Monk (1993) for multiple suppression.

- Interpolation to a common grid and space-variant wavelet matching by Johnstad et al. (1995).

Here, I summarize two approaches to finding a match between two seismic traces. The first is based on correlation and a scalar quantity estimate, and the second is based on spectrum coherence and a filter estimate. A thorough description is provided for the correlation scalar estimate method since it is used as a comparison with the proposed method in later chapters. Both methods are conventional linear cross-equalization methods. Their use varies from localized, i.e. matching trace by trace, to global where the filter or scalars are averaged and applied globally. Their estimation also varies and is done on selected time windows with high signal-to-noise ratios, selected traces, or from all available traces. At the end of this section, a short note is provided on matching using a commercial package, which is also used as a comparison.

Scalars estimate

In the scalars estimate method, three scalar quantities related to the misfit between any two traces are calculated. These quantities are time misfit, phase misfit, and amplitude scalar. The bandwidth between the two traces is assumed near identical. Once the scalar misfit quantities are calculated, the necessary correction can easily be applied to counter them. The method is outlined in Bishop and Nunns (1994).

The *cross-correlation*, $\psi_{12}(t)$, between two traces $r_1(t)$ and $r_2(t)$ is defined as:

$$\psi_{12}(t) = \frac{1}{T}\sum_{t=0}^{T-1} r_1(t)r_2(t+l), \tag{1.5}$$

where T is the total number of samples considered in the two traces and l is the time lag used in calculating the correlation. The time and phase misfits are calculated from

the instantaneous attributes of the cross-correlation. Those attributes are defined by Taner et al. (1979), who regards a recorded signal $s(t)$ as the real part of an analytical or complex signal, $S(t)$, where

$$S(t) = s(t) + is^*(t). \tag{1.6}$$

The quadrature (also called conjugate or imaginary) component, $s^*(t)$, is the Hilbert transform of $s(t)$. Hence, the *complex cross-correlation*, $\varphi_{12}(t)$ is defined as:

$$\varphi_{12}(t) = \psi_{12}(t) + i\psi^*_{12}(t). \tag{1.7}$$

Two attributes can be calculated for $\varphi_{12}(t)$: the *modulus or envelope*, $a(t)$, and the *instantaneous phase*, $\vartheta(t)$. These are defined by the following two equations:

$$a(t) = |\varphi_{12}(t)| = \sqrt{{\psi_{12}}^2(t) + {\psi^*_{12}}^2(t)}, \text{and} \tag{1.8}$$

$$\vartheta(t) = \arctan \frac{\psi^*_{12}(t)}{\psi_{12}(t)}. \tag{1.9}$$

The peak of $a(t)$ at the time lag $t = \tau$ gives the optimum time shift between the two traces $r_1(t)$ and $r_2(t)$. The *phase misfit* θ follows from the choice of *time misfit* since both phase and time show linear dependence (Bishop and Nunns, 1994). Hence, θ is calculated from the instantaneous phase such that $\theta = \vartheta(\tau)$.

The amplitude scaling, γ_{corr}, between the two traces is the ratio of the energy in the cross-correlation to the energy on the input trace, such that

$$\gamma_{corr} = \frac{|\psi_{12}(\tau)|}{\psi_{11}(0)}, \tag{1.10}$$

where $|\psi_{12}(\tau)|$ is the modulus of the cross-correlation defined in Equation 1.5 at time lag τ and $\psi_{11}(0)$ is the auto-correlation of $r_1(t)$ at zero lag. Auto-correlation is defined

as a similar process to cross-correlation, but the same trace is correlated with itself such that $r_2(t)$ is replaced with $r_1(t)$ in Equation 1.5.

The scaling factor γ_{corr} is not affected by random noise which does not correlate. A simpler approach to amplitude balancing is to estimate a scalar based on the root mean square (RMS) energy in the two surveys. This assumes either the noise energy in both traces is of the same magnitude or that it is much smaller than the signal energy (Rickett and Lumley, 2001). The scaling factor based on RMS, γ_{rms}, is calculated using the following equation:

$$\gamma_{rms} = \frac{\sum_{t=0}^{T-1} {r_1}^2(t)}{\sum_{t=0}^{T-1} {r_2}^2(t)} \tag{1.11}$$

The scalars estimate matching method is selected to compare results at the different stages of development of the proposed cross-equalization using artificial neural networks. Our preference to use this method stems from two main facts: firstly, the method is simple, robust, and computationally fast; secondly, there are three scalar corrections: bulk static shift, an amplitude gain factor, and a phase rotation — all three are intuitive and easily understood (Bishop and Nunns, 1994).

Filter estimate

The filter estimate method is discussed in White (1980), Walden and White (1984) and more recently in Walden and White (1998). This method describes a procedure to match a synthetic seismogram with a surface seismic trace, both imaging the same reflection series. The same approach could easily be adopted to match two time-lapse traces, $r_1(t)$ and $r_2(t)$, imaging the same series.

The situation the method addresses is outlined in Figure 1.8. Thus, each trace is composed of a signal series $s(t)$ and random noise $\epsilon(t)$, so that:

$$r_1(t) = s_1(t) + \epsilon_1(t), \text{and} \tag{1.12}$$

$$r_2(t) = s_2(t) + \epsilon_2(t). \tag{1.13}$$

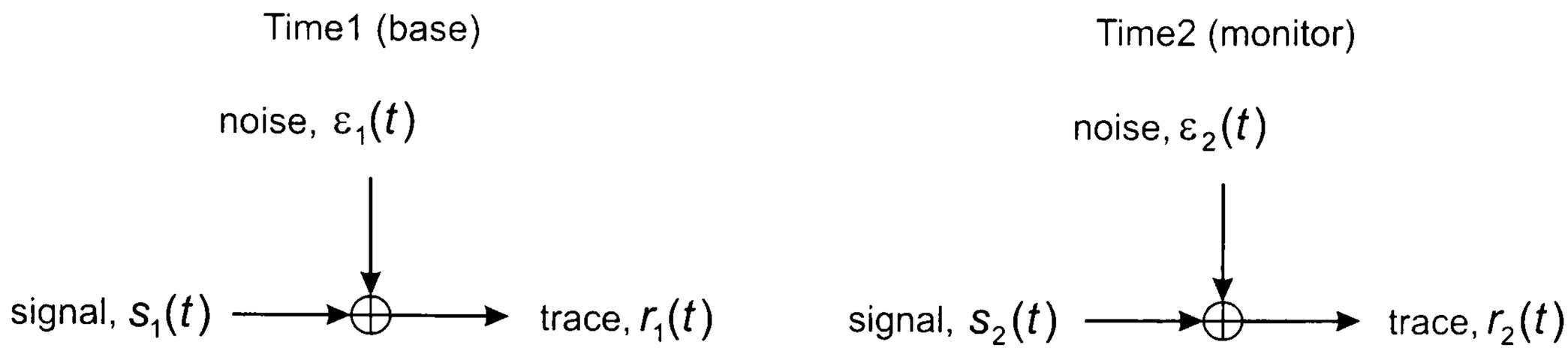


Figure 1.8: Two time-lapse traces $r_1(t)$ and $r_2(t)$ can be represented each by a signal and a noise component. In linear cross-equalization system a filter $h(t)$ is assumed to exist that can linearly predict one from another, such that $s_2(t) = h(t) * s_1(t)$. Literally, $h(t)$ will apply the necessary corrections to bring the two signals to a closer match. Adapted from Walden and White (1998).

The two signals, $s_1(t)$ and $s_2(t)$, are related linearly so that:

$$s_2(t) \quad = \quad h(t) * s_1(t), \tag{1.14}$$

where $*$ denotes convolution, and $h(t)$ is the impulse response of the filter defining the linear system.

The estimation of $h(t)$ is done in the frequency domain by calculating its Fourier transform $H(f)$. $H(f)$, commonly known as the *transfer function*, can be calculated from the frequency equivalent of Equation 1.14. However, since s_1 and s_2 are not known quantities, one could calculate an estimate of the transfer function, $\hat{H}(f)$, by assuming that there is no noise (Walden and White, 1998, *p. 289*), so that:

$$\hat{H}(f) \quad = \quad \frac{\Phi_{12}(f)}{\Phi_{11}(f)}, \tag{1.15}$$

where $\Phi_{12}(f)$ is the cross-spectrum of the two traces, $r_1(t)$ and $r_2(t)$, and $\Phi_{11}(f)$ is the auto-spectrum of $r_1(t)$. In fact, $\hat{H}(f)$ is actually $H_{2|1}(f)$ the frequency equivalent of the filter that predicts $r_2(t)$ from $r_1(t)$ in the least-square sense. To calculate $H(f)$,

Walden and White (1998) show that $H_{2|1}(f)$ should be scaled by a function of the signal-to-noise ratio so that:

$$H(f) = \frac{1 + \rho_1(f)}{\rho_1(f)} H_{2|1}(f), \tag{1.16}$$

where $\rho_1(f)$ is the signal-to-noise ratio in $r_1(t)$. This ratio can be estimated using multiple coherence analysis as in Walden and White (1998).

The filter requires two parameters to be chosen: the length of the lag window and its central point. The lag window must be centred on the wavelet-dominated portions of the cross-correlations between the two traces (Equation 1.5), and long enough to enclose them without serious distortion while tapering out the noise dominated parts (White, 1980). Once the length is fixed, the optimum alignment is found empirically by scanning a series of lag positions selecting the one which gives the best match (White, 1980).

Matching in a commercial package

The commercial seismic processing package *ProMAX* is used throughout this study. One of the uses of ProMAX involves creating a matching filter as an additional comparison with the scalar estimate method.

The derivation of match filters and their application is described in the package's manual: Landmark (1997). Basically, a least-square (L2) filter is derived between each two traces within the two datasets or globally. In the latter case, traces are vertically summed before deriving the filter. The estimated filter can later be convolved with one dataset to make it match the other. L2 filter design and application is similar to that described earlier in the filter estimate method. The time shift is determined by picking the maximum amplitude envelope function of the cross-correlation as described in the scalar estimate method. In addition to L2 filters, one could derive an optimal phase rotation. In this study, I found it useful to derive a global phase rotation estimate and apply it before deriving and applying the L2 filter.

1.2.3 Proposed method: cross-equalization using ANNs

Earlier discussion shows that in time-lapse there is a need to find a cross-equalization tool with certain characteristics. The tool should not require exact knowledge of the reasons behind the lack of repeatability. It should correct not only for the four basic misfit elements (time, amplitude, phase, and bandwidth), but also for other misfits such as trace misalignments and noise. In addition, the tool needs to be able to handle interactions between the misfit elements and should not be hindered by the possible non-linearity of the relationship. This motivated me to investigate artificial neural networks as a possible tool for time-lapse seismic cross-equalization with all the aforementioned merits.

Artificial Neural Networks, ANNs, are computer programs capable of learning by experience, which have analogies to neurons in the mammalian brain (Bishop, 1995; Gurney, 1997; Looney, 1997). In the last decade, ANNs have made their way into geophysics and have since been used to solve many geophysical problems (van der Baan and Jutten, 2000). Examples of such uses are extrapolation of well parameters by Liu and Liu (1998), seismic wavelet estimation by Liu and Castagna (2000), inversion by Roy-Chowdhury et al. (1997), prediction of porosity and saturation from time-lapse by Tanja Oldenziel and Kvamme (2000), deconvolution by Wang and Mendel (1992), multiple attenuation by Essenreiter et al. (1998), and tracking of seismic reflectors across faults by Alberts et al. (2000).

ANNs, in addition to their uses in pattern recognition, are also ideal for problems where a mapping function between some input parameter vector $\mathbf{x}$ and an output parameter vector $\mathbf{v}$ is to be found. An ANN can act as a mapping function where a numerical solution is not easy to formulate algorithmically, provided that many examples of input vector $\mathbf{x}$ and their corresponding desired output vector $\mathbf{v}$ exist. The *Kolmogorov theorem* proves the existence of a three-layer mapping ANN that can map any continuous function. It is quoted here in its form applied to neural networks as derived by Hecht-Nielsen (1990, *p. 122*):

LONDIN. UNIV.

"*Given any continuous function* $f : [0,1]^n \longrightarrow \mathbb{R}^m$, $f(\mathbf{x}) = \mathbf{y}$, f *can be implemented exactly by a three-layer feedforward neural network having* n *fanout processing elements in the first* ($\mathbf{x}$ — *input) layer,* $(2n+1)$ *processing elements in the middle layer, and* m *processing elements in the top* ($\mathbf{y}$ — *output) layer.*"

In this thesis I show that cross-equalization is a problem that can be solved by ANNs. Time-lapse surveys outside production areas can provide many examples of relationships between the two surveys as is needed by ANNs. In simple terms, if we desire one survey to look like the second, then one survey can act as input vector $\mathbf{x}$ and the other as the corresponding desired output vector $\mathbf{v}$. During the design of ANNs (*training stage*), ANNs are shown examples of seismic data from one survey (vector $\mathbf{x}$) and their corresponding seismic data from the other survey (vector $\mathbf{v}$). ANNs will learn to map one vector from the other, and will hence learn to normalize for acquisition misfit (since examples are taken from outside production zones). ANNs can later be applied to the whole survey and thus normalize it for acquisition differences.

The use of ANN will be shown to have the following merits:

1. targeting the reason for the lack of repeatability becomes unnecessary;
2. ANNs have the capability of automatic noise suppression as is the case in least-square matching technique described by Bunch and White (1985a);
3. ANNs' ability to solve non-linear problems will give them a huge advantage over other linear cross-equalization schemes; and
4. the scheme can deal with trace realignment.

Cross-equalization using ANNs is proposed to be done pre-stack and prior to any processes that might cause unnecessary loss of information. This is for the following main reasons: firstly, more training examples are available pre-stack and will hence cater for ANNs' need to learn the necessary relationship; secondly, effects of sources

of misfit are easily complicated by processing and some useful information might be lost during processing; thirdly, pre-stack analysis such as AVO can be performed and compared on the time-lapse datasets; fourthly, after cross-equalization, the user will have the flexibility to process the cross-equalized time-lapse data in parallel and however perceived necessary.

Davies and Warner (1999) and Davies (2000) attempted a similar application of ANNs to that presented in this study. Their methodology involved mapping higher-frequency data from lower-frequency data, i.e. cross-equalize lower-frequency data to match higher-frequency data. Thus, their work attempted to extrapolate missing resolution. Since, theoretically, lower-frequency data cannot contain enough information to reproduce higher-frequency data, the problem can only work where the network can interpolate within the training space. Therefore, their method did not work when a designated reservoir area was excluded from training. Although I am trying to solve a similar problem, in this case an ANN can generalize[2] better because it is not trying to extrapolate missing resolution. Furthermore, this project involves more data, which can be employed in training. In addition, a more thorough investigation is required on the exact training procedure and parameter choice to that attempted in limited time in Davies and Warner (1999) and Davies (2000).

1.3 Thesis objective and outline

1.3.1 Objective and methodology

The objective of this study is to develop a scheme to cross-equalize two time-lapse seismic datasets using artificial neural networks (ANNs).

The procedure involves: firstly, developing a scheme to match seismic data using ANNs, which involves investigating the different criteria and parameters for training ANNs and developing the codes necessary; secondly, assessing the ability of the scheme to preserve a production related anomaly within the two time-lapse seismic datasets

[2]Generalization is the ability of ANN to perform well in data not shown during training.

while matching otherwise. Furthermore, using ANNs as a cross-equalization tool for time-lapse seismic is compared with other more conventional methods.

1.3.2 Report outline

This first chapter introduced time-lapse seismic and the lack of repeatability problem. Cross-equalization was shown to be necessary to solve the aforementioned problem. This was introduced and discussed. Finally, artificial neural networks (ANNs) were shown as a potential candidate for the cross-equalization scheme. The research study was put forward to investigate the utilization of ANNs in the cross-equalization for 4D seismic.

Chapter 2 introduces ANNs and their related parameters. ANNs' learning capability and related issues, such as training algorithms, are explained and analysed. Finally, several issues involved in utilizing ANNs for this study, such as pre-processing, generalization, and parametrization, are discussed.

Chapter 3 introduces the time-lapse datasets used in this project. It gives an outline of the survey area in terms of the geological setting, production history, and seismic time-lapse project. In addition, it analyses the datasets in terms of their acquisition and seismic character. Finally, work done on the dataset, with regards to time-lapse, is reviewed, and expected fluid changes and signature are outlined.

Chapter 4 presents and discusses the implementation of ANNs in time-lapse cross-equalization. It describes how ANNs are used to cross-equalize the time-lapse datasets. In addition, it introduces the nature and type of possible input and target output of the networks. Furthermore, the chapter assesses the different parameterizations required, and, with support of experimental results, puts forward a choice. The chapter also presents the codes and procedures implemented, while using ANNs as a cross-equalization tool. In addition, it presents the methods used to quality control the results of cross-equalization. Finally, it presents a test that ensures that, while training to perform cross-equalization, the ANNs are not memorizing a copy of the dataset required to be

matched, but in fact are learning a function that normalizes acquisition differences.

Application and results are discussed in Chapter 5 and Chapter 6. Chapter 5 describes the historic development of the scheme. It outlines the procedure followed to find the configuration that gives the best results. Finally, it presents the best scheme configuration, and matching results. Chapter 6 describes further investigation on ANN cross-equalization. It presents a methodology to assess the ability of ANNs to preserve production-related anomalies while matching elsewhere. In addition, it assesses post-stack cross-equalization as an alternative to pre-stack cross-equalization, and compares the two. Both chapters present and discuss the results while making comparisons with linear cross-equalization methods.

Finally, Chapter 7 summarizes the research results and conclusions, and provides an outlook for further possible work.

Chapter 2

Artificial Neural Networks (ANNs)

The creation of *artificial neural networks* (ANNs) stems from human desire to imitate biological neurons in their learning and building upon experience. Although scientists only managed to represent biological neurons in the most abstract form, nevertheless this has proved very useful. Today, ANNs are used in many diverse fields including economics, finance, mathematics, chemistry, physics, engineering, robotics, and cryptography.

ANNs have the ability to make extensive use of a priori knowledge, find non-linear relationships, and are not restricted by assumptions about the underlying model (Essenreiter et al., 1998). Therefore, ANNs have the potential to work in areas where conventional methods fail. They are particularly well suited for non-linear problems (Hush, 1989). Their capability allows them both to partition a pattern space for classification problems, e.g. hand written text recognition (LeCun et al., 1990a, 1998a), and to implement non-linear transformations for functional approximation, e.g. locating subsurface targets (Poulton et al., 1992). A brief historical overview of ANN can be found in Hecht-Nielsen (1990). Lippmann (1987) and Hush and Horne (1993) provide a good tutorial on ANNs, while Bishop (1995) gives a more extensive analysis.

The most commonly used ANN architecture is the *multi-layer perceptron* (MLP). In addition, applications similar to those in this research employ the supervised feed-forward MLP. This undergoes a supervised training stage and an application stage.

This chapter introduces the basic concepts and theoretical background of ANN. The chapter is subdivided into four sections: the first introduces the biological neural networks with emphasis on how learning is achieved, the second describes the supervised feed-forward MLPs and their character, the third describes how learning is achieved through training the ANNs and aspects related to this training, and the fourth analyses several practical issues related to the utilization of ANNs in this study.

In writing this chapter, I found Hush and Horne (1993) particularly useful in their analysis of the MLP and followed their guidance to the related issues. I also found Alberts (2002) useful in introducing ANNs and their training algorithms.

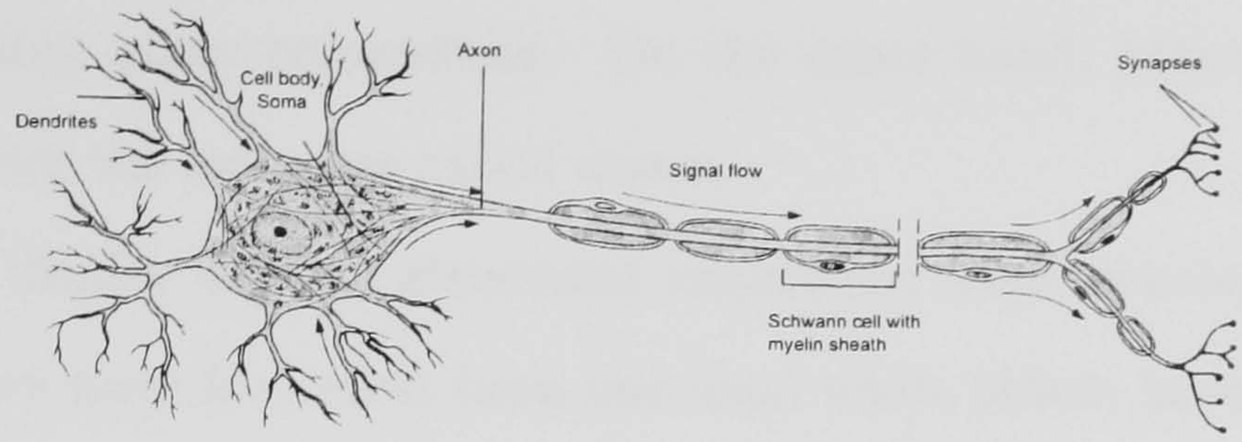
2.1 Biological neural networks

ANN design stems from our understanding of biological neurons. Thus, it is appropriate to introduce the topic through a discussion of the latter. This introduction will emphasize information transmission within the biological neural networks. The general properties of those networks will be outlined and an abstract base to ANNs will be discussed.

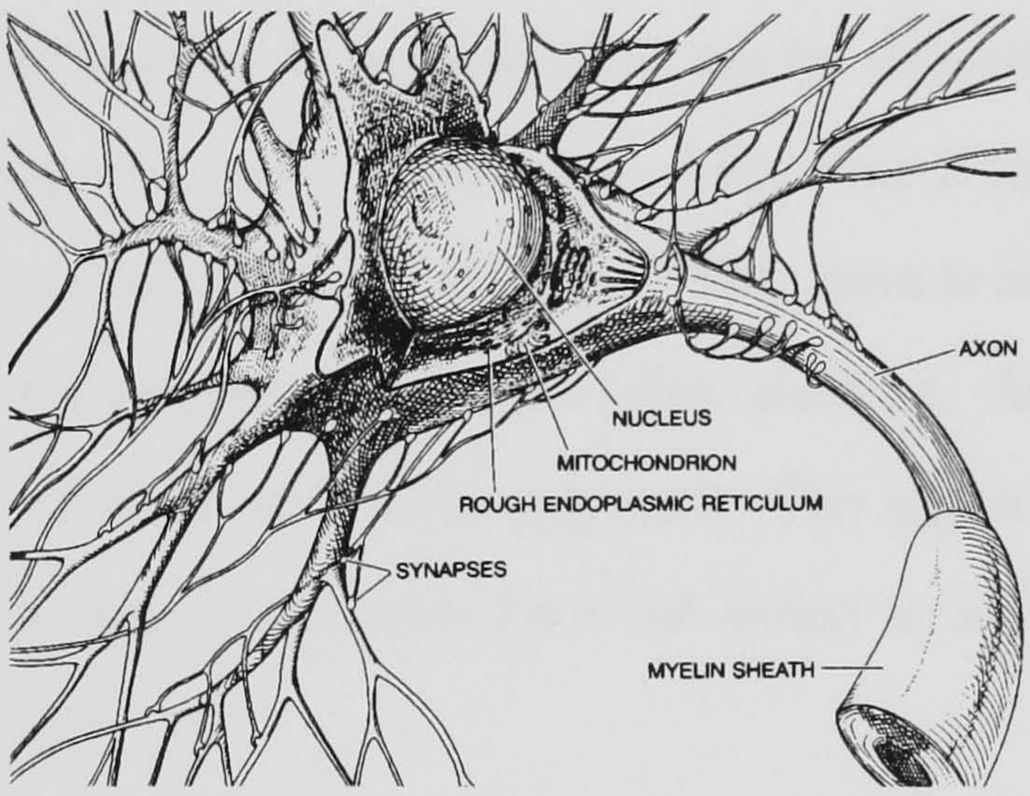
2.1.1 Structure and information handling

Nerve cells which make up the brain are biological neurons and the brain itself is a complex biological neural network. Those neurons are responsible for information handling. They continuously receive and send signals in the form of an electrical impulse.

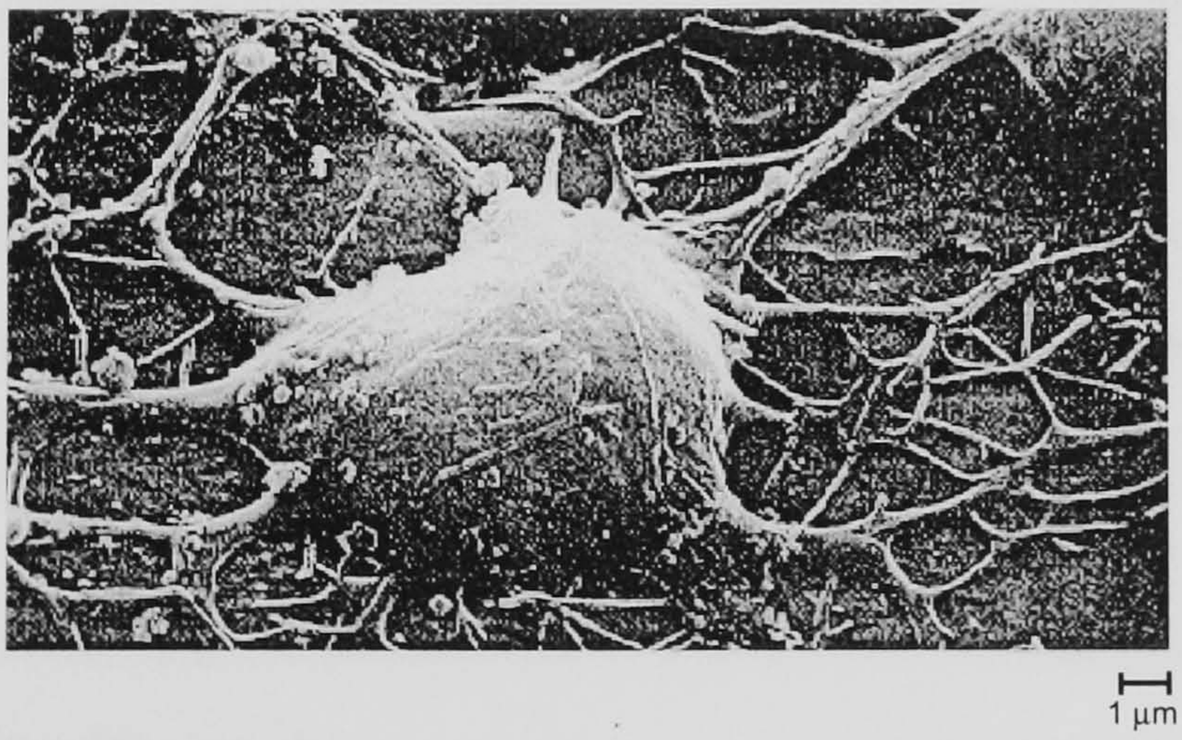
The schematic structure of a nerve cell is illustrated in Figure 2.1. The cell body (*soma*) is similar to that of other cells, with a nucleus and most of the expected *organelles* present. In addition, however, neurons contain extensions or branches called *processes* (Becker, 1986, *p. 695*). Processes that receive impulses and transmit them



(a) A schematic of a nerve cell showing the signal flow through the main nerve parts. After Becker (1986)



(b) Another schematic of a nerve cell showing the amount of connections that can exist. From Alberts (2002)



(c) A scanning electron micrograph of the cell body and its dendrites. From Becker et al. (2000)

Figure 2.1: The nerve cell is a biological neuron consisting of dendrites, cell body, and an axon, which branches and terminates with synaptic knobs. The dendrites collect incoming signals from other cells and pass them inward to the cell body (soma). The cell, if activated by the incoming signals, pass a signal through its single axon. This latter transmits the cell's activity to other cells, which are connected via synapses on their incoming dendrites. The myelin sheath act as an electric insulator.

inward to the soma are called *dendrites*. These are hand-like extensions that branch repeatedly, enabling many connections. On the other hand, processes that conduct impulses away from the soma are called *axons*.

The neurons display a lot of structural variability; some neurons have many processes while others have few; most have one axon while others have many. Typically, axons are much longer than dendrites. The axon terminates in structures called *synaptic knobs*. These are responsible for transmitting impulses to the next cell. The junction between the synaptic knobs and other cells is called a *synapse*.

A nerve cell receives impulse from other cells via its dendrites. The nerve fires a signal in response to impulses received over some delay time if those impulses exceed a certain threshold. The signal is sent via the axon. This axon is insulated by the *myelin sheath* wherever it is not supposed to trigger other neurons. At the far end it splits and connects to other cells' dendrites via *synapses*. The signal is transmitted across the synapses chemically, and the synapses have the ability to influence the signal in an *excitatory* or an *inhibitory* way.

2.1.2 Properties of biological neural networks

The nervous system of simple invertebrates might contain 100,000 cells, while a human being is estimated to have about 100 billion nerve cells (Alberts, 2002). This great number of cells make up a biological neural network that both processes senses and learns. The ability to learn enables organisms to respond to varying environment and needs. The success of biological neural networks is facilitated by four main characteristics:

1. A great number of connections exist in the network.

2. Biological neurons work in parallel. This enables the biological network to process information simultaneously.

3. Learning is not only established by activity change within neurons but also by adaptation of the whole structure. Hence knowledge is stored in the whole network.

4. There is a great tolerance toward individual neuron failure. The network can work near optimum even with some neuron degradation.

2.1.3 Abstract neuron

Units in ANNs can only represent their biological counterparts in an abstract and simple way. This abstract representation is illustrated in Figure 2.2. The main character that ANNs try to inherit from biological networks is *connectionism*, which is achieved through the great number of interconnected and communicating units.

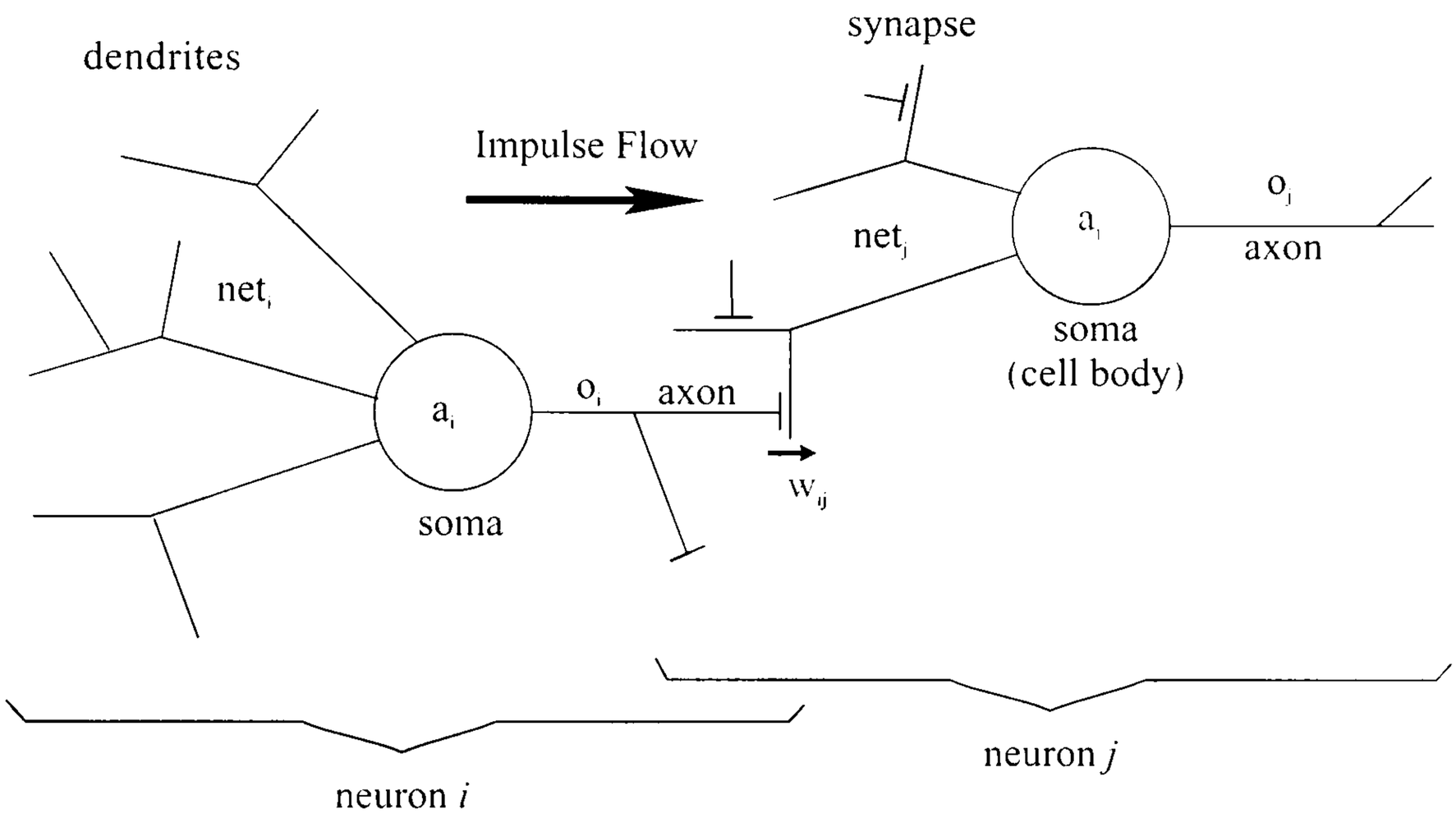


Figure 2.2: Abstract model of a biological neuron. The cell receives information net_i from its dendrites. Consequently, it builds up a membrane potential a_i. This causes an activation potential o_i to be output via the axon. The axon is finally connected to further dendrites by means of synapses, which can act excitatory or inhibitory on the signal by applying a factor w_{ij}. After Alberts (2002).

The *membrane potential* $a_j(t)$ is a measure for the degree of *activation* at time t.

The transfer function f_{act} determines the new state of activation for time-step $t+1$ based on the old activation $a_j(t)$ and the input $net_j(t)$:

$$a_j(t+1) = f_{\text{act}}(a_j(t), net_j(t), \Theta_j). \tag{2.1}$$

Θ_j is the *threshold* of neuron j below which no signal is fired. The activation potential

or output $o_j(t+1)$ is then a function of the membrane potential:

$$o_j(t+1) = f_{\text{out}}(a_j(t+1)). \tag{2.2}$$

Most artificial models build on this abstract form. Furthermore, in this research the following definitions are used for simplification:

1. the possible activation a_i is restricted to the interval $[0, 1]$;

2. an identity function is employed to move from activation a_j to output o_j, i.e. $o_j(t) = a_j(t)$; and

3. the time delay used in biological neurons is not modelled, i.e. activation at time t is not dependent on previous activations.

Hence, a neuron j after receiving some input net_j outputs a signal o_j, where:

$$o_j(t+1) = f_{\text{act}}(net_j(t), \Theta_j). \tag{2.3}$$

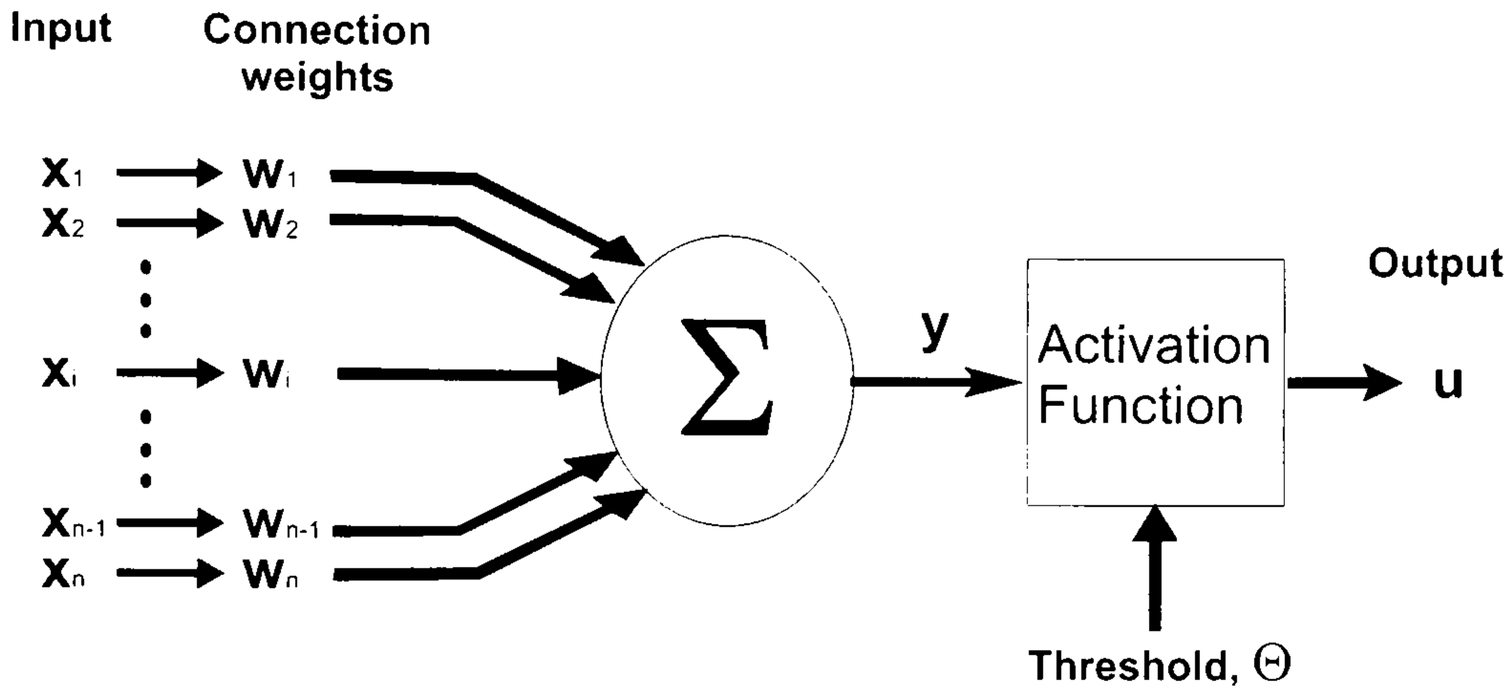
2.2 Artificial neural networks

2.2.1 Perceptron

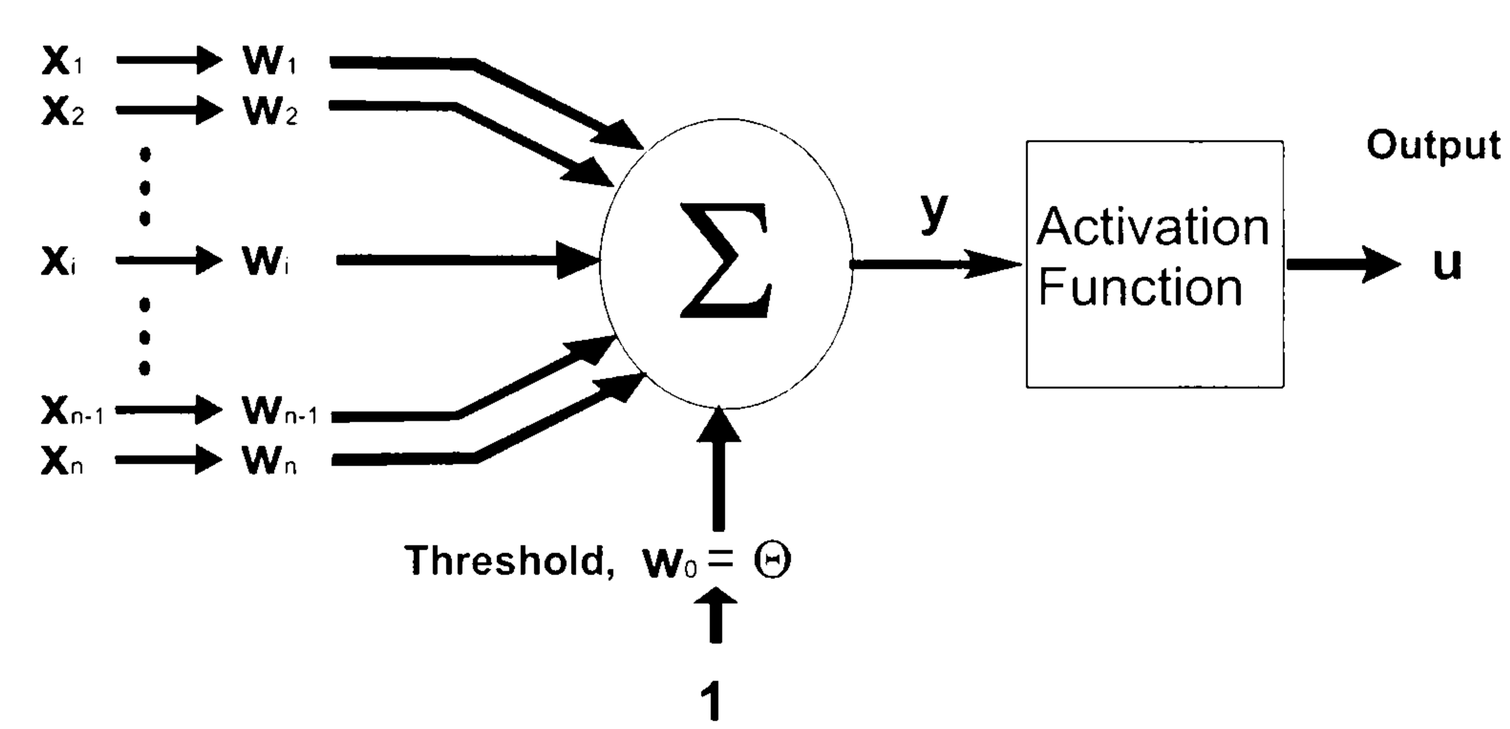
The *perceptron* was conceived in 1958 by Rosenblatt (Hush and Horne, 1993). It mimics the action of the abstract biological neuron (Section 2.1.3). Figure 2.3 shows the basic structure of a perceptron. The input y to the activation function is the dot product of the input vector $\mathbf{x}$ and the weight vector $\mathbf{w}$, so that:

$$y = \sum_{i=1}^{n} w_i x_i. \tag{2.4}$$

The original perceptron used a hard-limiting activation function, whereby the out-



(a) A diagram of the perceptron showing the main components



(b) A slight modification of (a) where the threshold is now considered part of the input. Thus, a connection is modelled with an input $x_0 = 1$ and a weight $w_0 = \Theta$.

Figure 2.3: The perceptron is the base unit of ANNs and mimics an abstract biological neuron (*cf.* Figure 2.2). It receives an input vector $\mathbf{x}$, which is scaled by the weight vector $\mathbf{w}$. The dot product is the scaled input y (corresponds to net_j in Figure 2.2). The scaled input y and a threshold Θ are passed through an activation function to produce an output u. In the original perceptron the activation function is a hard-limiting boolean function, but most commonly it is a sigmoid.

put u is calculated by the following equation:

$$u = \begin{cases} 0 & \text{if } y \geq \Theta, \\ 1 & \text{if } y < \Theta. \end{cases} \tag{2.5}$$

Θ is the threshold of the perceptron that has to be reached before an output is produced. This is similar to the biological threshold that has to be exceeded before a neuron fires. Usually, the threshold is illustrated as part of the total input to the unit (Figure 2.3(b)), such that:

$$y = \sum_{i=0}^{n} w_i x_i, \text{ and} \tag{2.6}$$

$$u = \begin{cases} 0 & \text{if } y \geq 0, \\ 1 & \text{if } y < 0. \end{cases} \tag{2.7}$$

The perceptron can be viewed as a *binary logical function* (Hush and Horne, 1993). It is capable of implementing numerous logical functions, including the three fundamental operations of Boolean algebra: *AND*, *OR*, and *COMPLEMENT*. While using a hard-limiting activation function, the perceptron is simply a *linear threshold unit* with a bias or threshold unit added (Gurney, 1997). However, most commonly, as in this study, a continuous sigmoid function is used as an activation function. This allows the perceptron to perform tasks other than classifications.

2.2.2 Multi-layer perceptron (MLP)

A *multi-layer perceptron* (MLP) is a cascade of two or more layers of perceptrons. Figure 2.4 shows a typical MLP, where perceptrons are arranged in a layered fashion and each layer is fully connected to the next.

An MLP is the most widely used ANN. Therefore, from now on, I will be using ANN to mean this form of artificial neural network. However, there are still many models and variations of networks. The study will be using the most common of those:

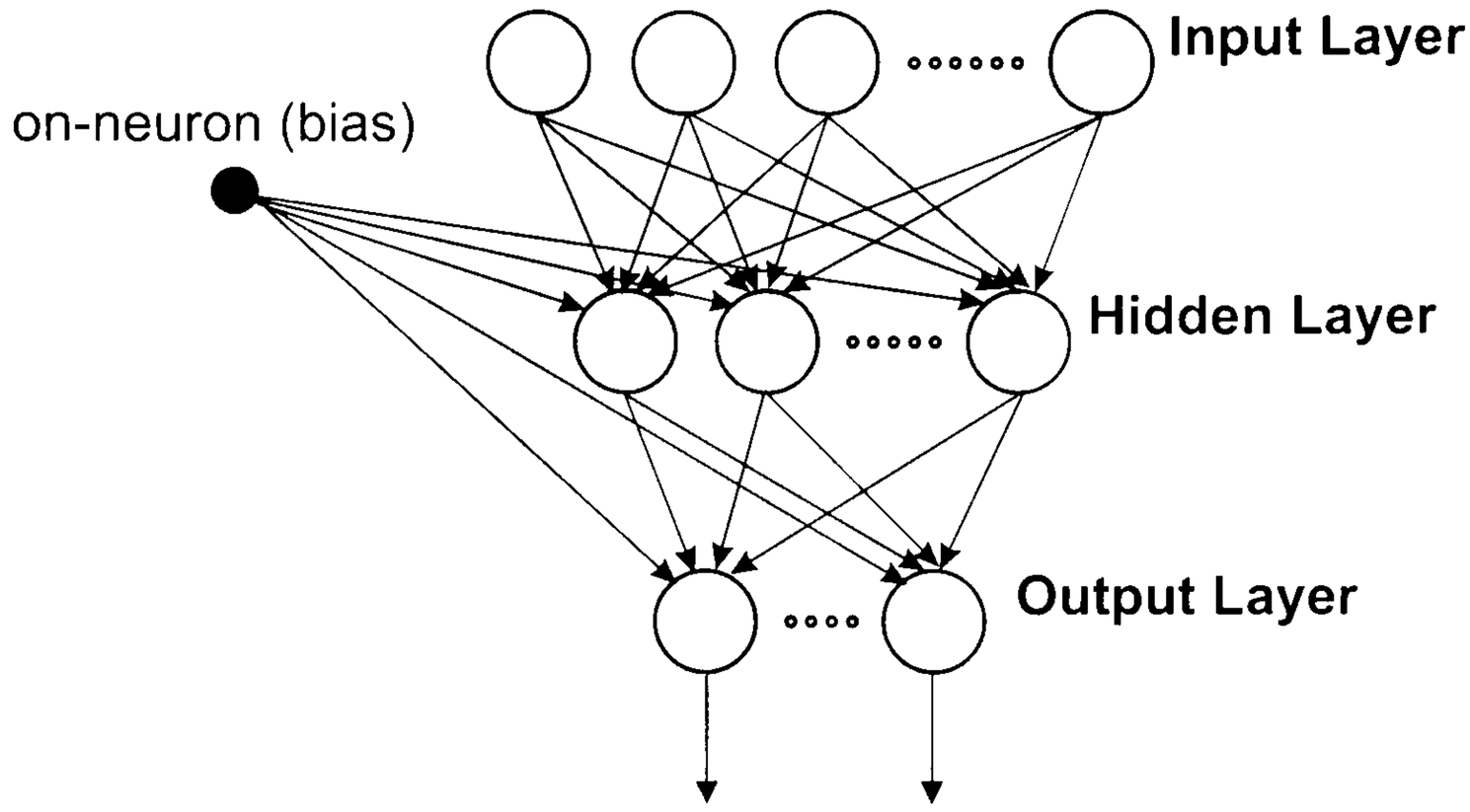


Figure 2.4: *Multi-layer perceptron* (MLP). The MLP is a cascade of two or more layers of perceptrons and is a typical ANN. The network here is organized into three layers: one input layer, one hidden layer and one output layer. Since the input layer only passes input from the outside world, the figure actually represents two layers of perceptrons. The black *on-neuron* provides the bias or threshold Θ by supplying an input of 1 and varying the connection weight to provide Θ, see Figure 2.3(b). This on-neuron might be omitted from many ANN–diagrams and is assumed to exist as an integrated part of the processing elements. Note how the network is depicted as a directed graph with a set of points (nodes) linked by a set of line segments (links). The nodes represent the processing units and links represent the connections between them.

the *static*, *supervised*, *feed-forward*, and *fully-connected* networks (MLPs). Here is the definition of those terms:

- *static* means the networks are memoryless, whereby their output is a function only of the current input, not past or future inputs or outputs (Hush and Horne, 1993).

- *supervised* means the networks are presented with set of example input-output pairs and trained to implement a mapping that matches the examples as closely as possible (Hush and Horne, 1993). This is contrary to *unsupervised* networks, which are only presented with input examples.

- *feed-forward* means that information (signal) is only passed along one direction, i.e. there is no network output that is fed back to the same network.

- *fully-connected* means that each neuron is fully connected with all neurons of the following layer.

An artificial neural network is defined by Hecht-Nielsen (1990, *see p. 22*) as "*a parallel distributed information processing structure in the form of a directed graph*[1] ", see Figure 2.4. However as described by Hecht-Nielsen (1990) the following restrictions and sub-definitions apply:

1. The nodes are the *processing elements.*

2. The links are called the *connections.* These connections act as instantaneous unidirectional signal-conducting paths.

3. Each processing element can receive any number of incoming connections.

4. Each processing element has one output connection that can fan out into many copies.

5. Each processing element possesses a *transfer function* which uses input signals to produce the processing element output.

6. Inputs to the network arrive via connections that originate in the outside world. Outputs from the network to the outside world are connections that leave the network.

In addition, the processing elements are sometimes loosely called neurons. The latter actually means the node and all the connection weights leading to it.

Topology

An MLP has an *input layer*, a number of *hidden layers* (usually none, one, or two), and an *output layer.* Each of these layers can have any number of *processing elements.* The input layer receives signals from the outside world and does not have processing elements, i.e. it does not alter the signal through an activation function. The hidden layers are used for the internal processing of the information. The output layer does the

[1] A *directed graph* is a geometric object made of a set of points (nodes) linked by a set of line segments (links)(Hecht-Nielsen, 1990).

final processing and outputs to the outside world. Therefore, all units are perceptrons except the input layer units, which perform no processing.

There are two conventions usually used when illustrating a network's topology. The first depicts inputs with no nodes while the other depicts those inputs with a layer of nodes. Figure 2.5 illustrates those two conventions. Hecht-Nielsen (1990) uses the first of those two conventions (Figure 2.52.5(a)), and thus his definition of nodes only apply to those in the hidden layers and output layer. Contrary to Hecht-Nielsen (1990) and as a convention followed in this report, I use the second convention (Figure 2.52.5(b)). where input is represented by a layer of nodes (input layer). These nodes only represent the input passed forward to the network and do not have a transfer function.

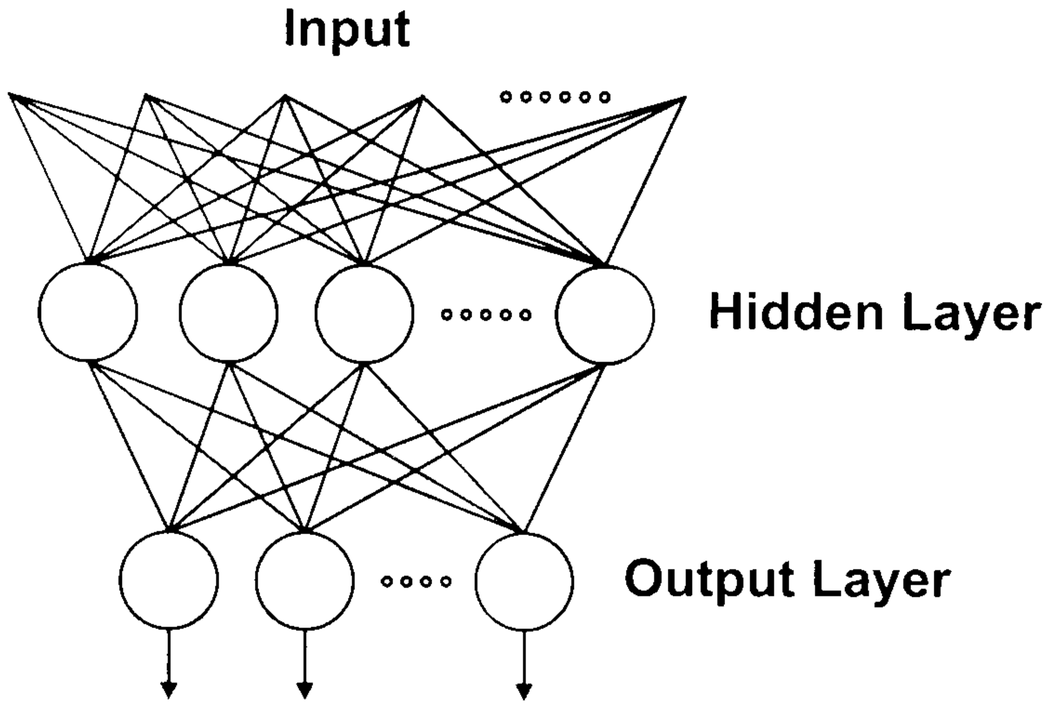
Signal flow

Figure 2.6 illustrates how a signal propagates unidirectionally through the feed-forward ANN. The connection from a node i to a node j is scaled by the weight w_{ij}. The order of indices indicate the direction of the signal transfer. The matrix of all connections is called weight matrix W.

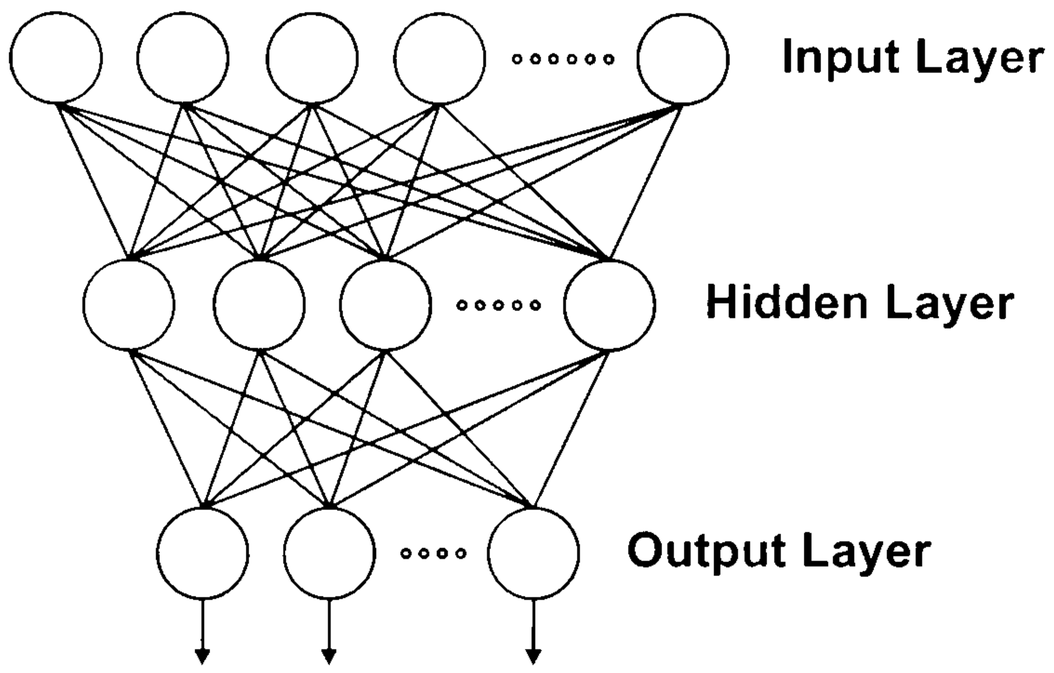
The input to node j is called net_j. This is calculated from all the output o_i of N feeding nodes by taking into account the weight w_{ij}. The calculation follows the *propagation rule*:

$$net_j(t) = \sum_{i=1}^{N} o_i(t)\, w_{ij}. \tag{2.8}$$

As described in Figure 2.4 and following from many authors such as Bishop (1995, *p. 118*), the threshold Θ_j introduced in equation 2.3 is integrated as part of the input to a neuron. This is done by introducing on-neurons or biases. The bias is usually considered as an integrated part of the processing elements. This on-neuron o_0 always produces an output of 1. The threshold Θ_j is then modelled by adapting the weights w_{0j} that connect the on-neuron 0 with neuron j. Therefore, the transfer function in



(a) ANN is illustrated with only those nodes that actually process their input



(b) ANN is illustrated with the nodes that process their input and with a layer of nodes representing input from the outside world

Figure 2.5: Two conventions for illustrating ANNs. The convention used in this report is illustrated in (b). Note that in both figures the bias on-neuron is not shown as explained in Figure 2.4.

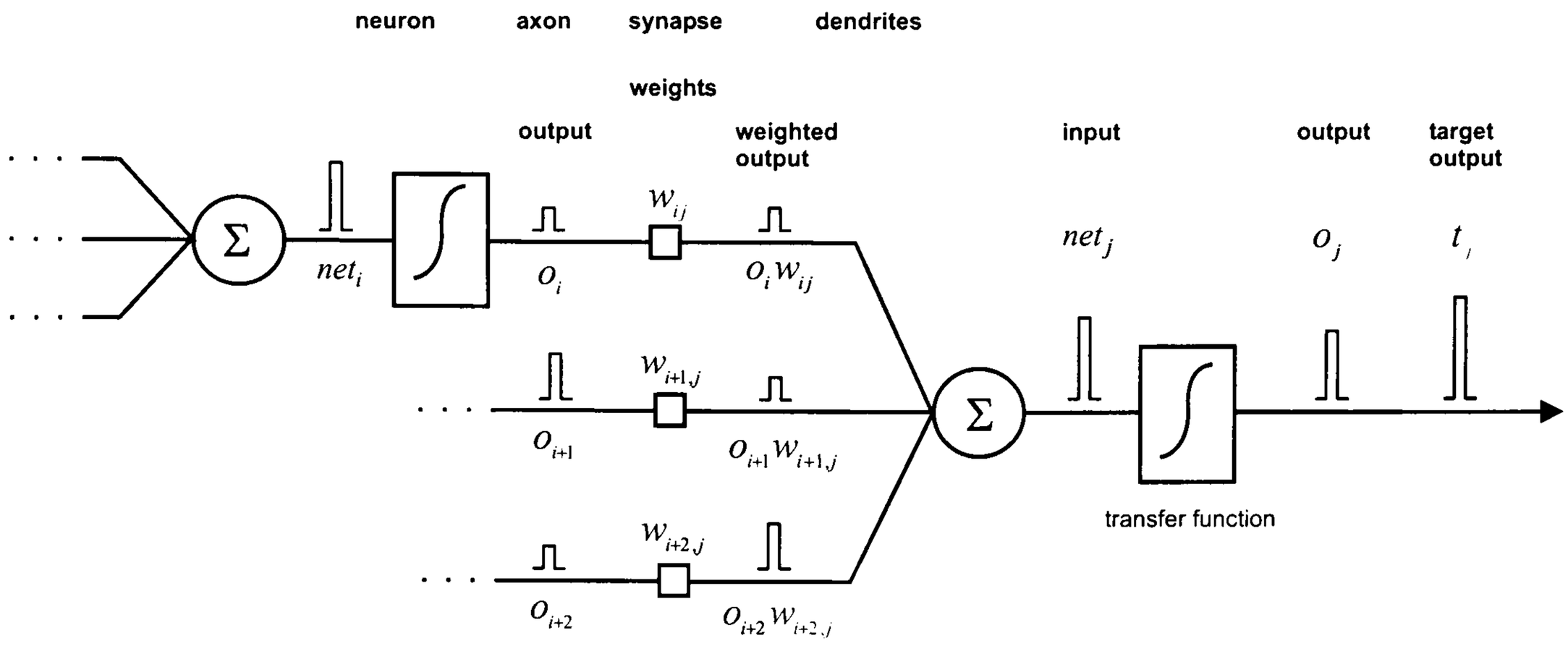


Figure 2.6: The propagation rule. A node's input is the sum of the weighted output of the feeding nodes. The signal flows unidirectionally from left to right. Compare with the abstract biological network shown in Figure 2.2. After Alberts (2002). During training, the error between the network's output o_j and the target output is used to calculate the adjustments for each connection weight.

Equation 2.3 can be calculated without the threshold Θ_j and hence simplifies to:

$$o_j(t+1) = f_{\text{act}}(net_j(t)). \tag{2.9}$$

In addition, the index i of the sum in Equation 2.8 now starts with 0:

$$net_j(t) = \sum_{i=0}^{N} o_i(t)\, w_{ij}. \tag{2.10}$$

The activation function used in most ANNs, such as the networks used in this study, is the sigmoidal function $f_{\text{act}}(x) = \frac{1}{1+e^{-x}}$ (Figure 2.7). This function determines the new state of activation based on the input net_j.

The choice of sigmoidal transfer function for activation allows dealing with large domains while still mapping to a bounded range of $[0, 1]$. It is also differentiable and therefore it is possible to derive a gradient search for the optimum weights (Hush and Horne, 1993). In addition, it is a continuous real monotonically increasing function as demanded by the Kolmogorov theorem, quoted on Page 49. Where classification is intended, the sigmoid function in the interval $[0, 1]$ allows the output to be interpreted as

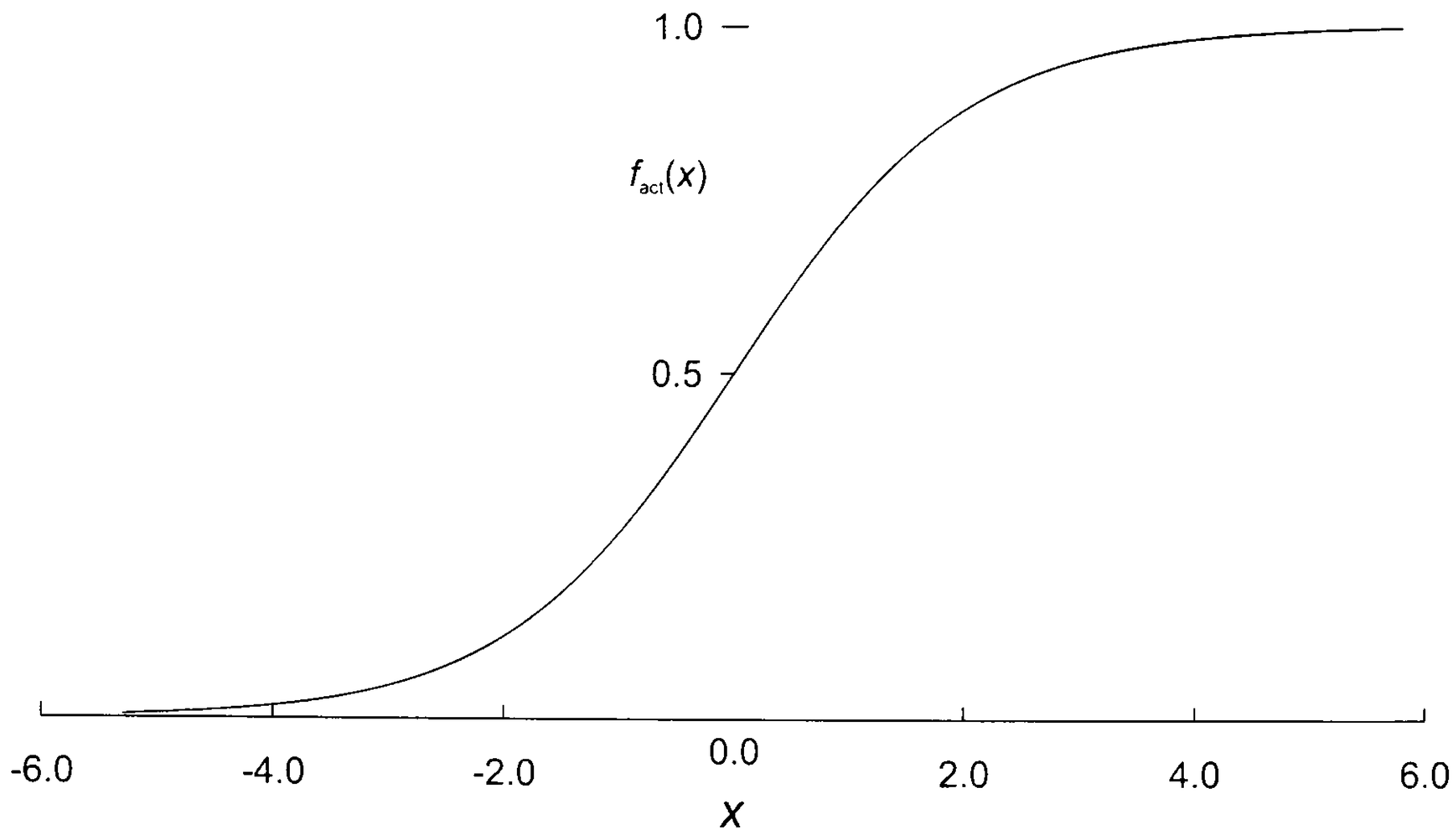


Figure 2.7: Activation transfer function, $f_{\text{act}}(x) = \frac{1}{1+e^{-x}}$. The input x is the weighted neuron input net_j, see Figure 2.2.

a posterior probability (Bishop, 1995, *see p. 83*). On the other hand, since the sigmoid is continuous, it also allows a continuous valued output required by applications other than classification.

Property and capability

ANNs mimic their biological counterparts and thus inherit their characteristics described in Section 2.1.2. In particular, they work by breaking problems into many parts that can be processed in parallel by simple processing elements, and they store information in the whole structure and have therefore great tolerance toward individual neuron degradation. Of interest is their ability for *abstraction* and *generalization.* Abstraction is the ability to retain salient features of an input pattern and discard irrelevant ones (Poulton et al., 1992). Generalization is the ability to recognize patterns on which no training occurred.

According to Hush and Horne (1993), the capability of ANNs lies in their ability to: implement boolean logic functions, partition pattern space for classification, and implement non-linear transformation for functional approximation.

To demonstrate the capability of an ANN, one could take its ability to classify as

a guideline. A single perceptron cannot implement an *exclusive-OR* function, whereas an MLP can implement complex decision boundaries and boolean functions (Hush and Horne, 1993). Its ability is determined by how many layers of perceptrons it consists of and how many neurons are in each layer. Figure 2.8 illustrates the capability of an ANN as a function of its number of layers. The figure describes the type of decision region attainable by each ANN structure. It is clear from the figure that a three layer ANN is sufficient for an arbitrary complexity while two layer ANN is sufficient to enclose a certain region.

The results in Figure 2.8 are for ANNs with hard-limiting activation functions. If the ANN has nodes with a sigmoidal activation function, then the capability is demonstrated by Bishop (1995) to be even greater. In these typical ANNs with a sigmoidal activation function, a two layer network can approximate arbitrarily well any functional continuous mapping from one finite-dimensional space to another, provided that the number of hidden nodes is sufficiently large (Bishop, 1995, *p. 130*).

The ability of a two layer ANN to map any continuous function follows the Kolmogorov theorem quoted on Page 49 and is discussed in many publications, e.g. Irie and Miyake (1988); Hecht-Nielsen (1990); Huang and Huang (1991); Hornik (1991); Leshno et al. (1993); Hush and Horne (1993); Bishop (1995); and Huang and Babri (1998). In fact, Irie and Miyake (1988) prove a kind of universality to the ability of the two layer ANNs (called three layer in Irie and Miyake (1988)). However, all publications require that sufficient number of perceptrons exist. Hush and Horne (1993) say that although a two layer MLP is capable of mapping any arbitrary continuous function, this does not imply that there is no benefit to a three layer MLP. The latter can be used for example where a two layer network requires an infinite number of nodes.

A two layer ANN is composed of an input layer, one hidden layer, and an output layer. Usually, the number of input and output nodes is fixed by the nature of the problem. Only the number of nodes in the hidden layer is usually open for variation. The hidden layer might be required to contain a very large number of nodes to enable

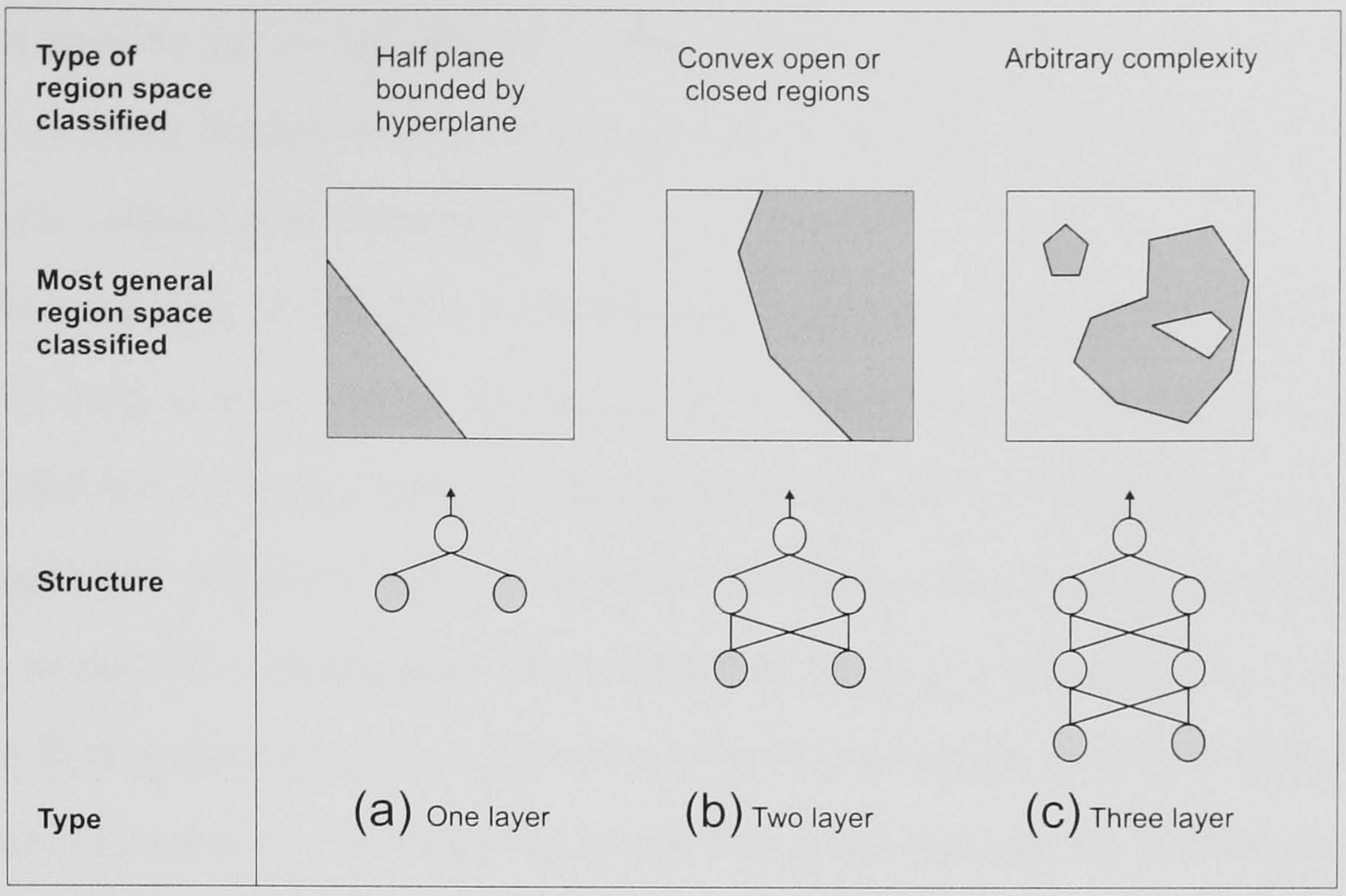


Figure 2.8: The capability of one, two, and three layer ANN. Adapted from Bishop (1995, *Figure 4.4*) and Lippmann (1987, *Figure 14*). The figure demonstrates the capability of an ANN to partition a region space into two classes as a function of the ANN's structure (number of layers). The ANNs illustrated contain hard-limiting activation functions. The shading within the region space denotes the decision region. The first layer in the network's structure denotes the input layer, where no processing is made and hence this layer is not counted. The learning space shown features two dimensions, but the space could be of any dimensionality. In summary, a one-layer ANN can split a region into two parts, a two-layer ANN can enclose a region, and a three layer ANN can form arbitrary decision regions.

the network to solve a given problem. In such cases, a three layer network, i.e. with two hidden layers, can reduce the number of connections significantly while still being able to solve the problem. Reducing the number of connections has the benefit of speeding up learning and reducing the number of training examples required to solve the problem.

2.3 Learning

The power of ANNs lies in their ability to learn. The Kolmogorov theorem quoted on Page 49 proves the existence of an ANN that can map a wide range of functions. Unfortunately, this theorem does not give an indication on how to construct an ANN

to map a specific $(x :\rightarrow y)$ relation. The ability of an ANN to map a relationship depends on many factors concerning the problem in question and the way the network is designed, trained and implemented.

ANNs are tuned to perform a desired mapping during the *learning* phase. If this learning is done in a supervised fashion, as is the case in this research, then this phase is also called the *training* phase. During training, examples of *input patterns* and their coinciding target *output patterns* are used to tune the parameters in the ANN in order to enable it to map the relationship. The training is done by minimizing an *error function* E, where E is a measure of the difference between the target output $\mathbf{v}$ and the actual output $\mathbf{u}$ of the network. E is most generally expressed through the Euclidean distance, which allows the error to be calculated for multi-dimensional output. However, most authors use a slightly modified version. The training algorithm chosen for this study expresses E as the square of the Euclidean distance:

$$E = \sum_i (v_i - u_i)^2. \tag{2.11}$$

There are several parameters that can be tuned to teach an ANN. The most common parameters that can be adjusted during training are the connecting weights, including the biases representing thresholds, and topology. The former is done by minimizing the error function E through changing the weight matrix W. The latter is tuned by either trying different topologies or by using an algorithm that dynamically grows or prunes the topology until a suitable topology is found.

In this research, it is primarily the weight matrix W that is tuned to obtain a suitably trained ANN. This not only requires a sophisticated training algorithm, but also sufficient number of training examples (*training patterns*) to optimize the weight matrix W.

2.3.1 Patterns

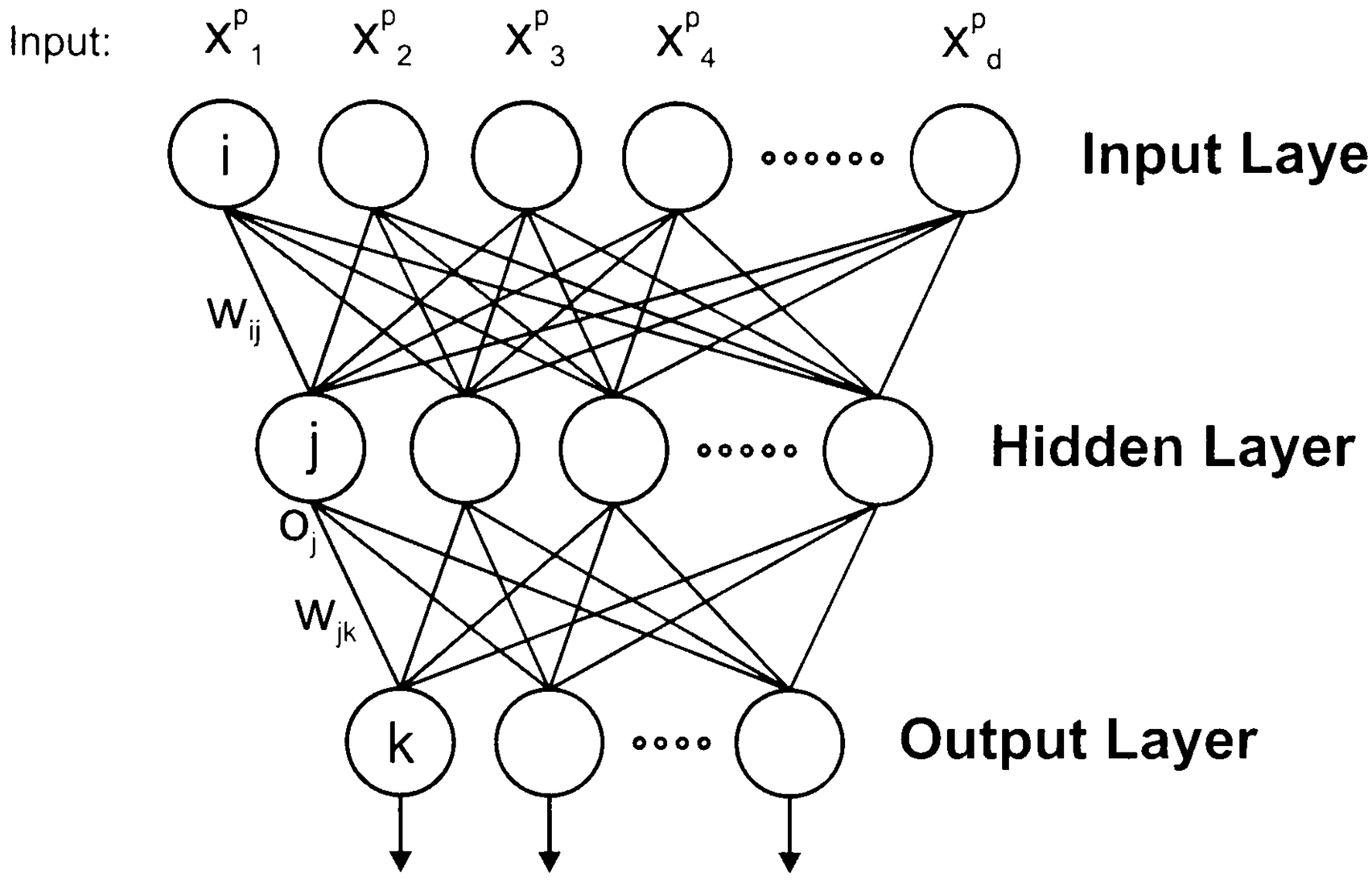
Figure 2.9 illustrates the parameters involved in ANN training and the terminology and abbreviations used. *Patterns* are example datasets of network input and associated target output. Patterns could all be used to tune the network and are then called the *training patterns.* However, it is practical to split the available patterns into two or three distinct datasets. The training patterns are used for the actual training, where the weight matrix W is adjusted depending on the misfit between the network's output and the known target output. The second dataset, which is used to test the network's performance during training, is called the *validation patterns.* This dataset is only fed to the network for testing purposes (*validation*), whereby only the error is calculated but no parameter tuning occurs. Validation is necessary for the cross-validation stopping criterion, as will be discussed later.

It is also useful to define a third pattern dataset, namely: the *testing patterns.* This dataset contains input and output patterns but is not involved in network training or cross-validation. The testing patterns are usually prepared to test the final trained network to examine its generalization and accuracy. Since the testing dataset does not influence training at all, its use leads to a better indication of generalization than through the use of the validation dataset. However, the testing dataset is not always required and used.

2.3.2 Back-propagation

During learning, ANNs need to adapt their weight matrix W. This adaptation is a function of the error in the output and the effect of each weight w_{ij} on this error. A rule is needed to calculate the required weight adaptation during learning. The most widely used algorithm for supervised learning with multi-layered feed-forward networks is the *back-propagation* learning algorithm. This is also known as the *generalized δ-rule.*

The basic idea of back-propagation is the repeated application of the chain rule to compute the influence of each weight in the network with respect to the error function



Parameters	Abbreviation	Name
$\{x_1, x_2, \ldots, x_d\}$ or $\{x_1^p, x_2^p, \ldots, x_d^p\}$	$\mathbf{x}$ or $\mathbf{x}^p$	An input pattern
$\{v_1, v_2, \ldots, v_c\}$ or $\{v_1^p, v_2^p, \ldots, v_d^p\}$	$\mathbf{v}$ or $\mathbf{v}^p$	An output pattern
$\{\{x_1, x_2, \ldots, x_d\}, \{v_1, v_2, \ldots, v_c\}\}$	$\{\mathbf{x}, \mathbf{v}\}$	A training pattern
$\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N\}$	X	Set of all input patterns
$\{\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^N\}$	V	Set of all output patterns
$\{\{\mathbf{x}^1, \mathbf{v}^1\}, \{\mathbf{x}^2, \mathbf{v}^2\}, \ldots, \{\mathbf{x}^N, \mathbf{v}^N\}\}$	$\{X, V\}$	Set of all training patterns

Figure 2.9: Patterns for ANNs and their nomenclature. A three layer network (two layer MLP) is shown. The nodes in the first, second, and third layer are labelled with indices i, j, and k, respectively. The choice of indices is arbitrary, but the order indicates the order of layers and thus the direction of the signal transfer. Connection weights and nodes' output follow the convention discussed earlier. Therefore, e.g., node j produces output o_j that fans to all nodes in the next layer and is scaled by weight w_{jk} before arriving at node k.

During training, a network is shown a set of parameter inputs (vector $\mathbf{x}$) and their target outputs (vector $\mathbf{v}$). The network itself produces an output vector $\mathbf{u}$. d is the total number of inputs and c is the total number of outputs. $\mathbf{x}$ is called an *input pattern*, whereas $\mathbf{v}$ is called an *output pattern*. The set of input and corresponding target output is called a *training pattern* or simply a *pattern*. In practice, there are many patterns and hence the superscript p is used as an index to $\mathbf{x}$, $\mathbf{u}$, and $\mathbf{v}$ indicating that each parameter belongs to a certain pattern p. This index could be any value from 1 to the total number of patterns N.

E (Riedmiller and Braun, 1993):

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_i} \frac{\partial o_i}{\partial net_i} \frac{\partial net_i}{\partial w_{ij}}, \tag{2.12}$$

where w_{ij} is the weight connecting neuron i with neuron j, o_i is the output of neuron i, and net_i is the sum of the weighted inputs to neuron i. Once the partial derivative for each weight is known, the aim of minimizing the error function is achieved by performing a gradient descent:

$$w_{ij}(t) = w_{ij}(t-1) - \epsilon \frac{\partial E}{\partial w_{ij}}(t), \tag{2.13}$$

where $w_{ij}(t-1)$ is the old connection weight, $w_{ij}(t)$ is the new connection weight, and ϵ is the *learning rate*. The learning rate ϵ determines the amount of change of the weight w_{ij} in the direction of the steepest error descent.

The derivation of the back-propagation algorithm can be found in Bishop (1995) and Alberts (2002). In summary, each connection weight w_{ij} is varied by an amount Δw_{ij} with every training pattern p according to the equation:

$$\Delta^p w_{ij} = \epsilon \, o_i^p \, \delta_j^p. \tag{2.14}$$

The superscript p indicates that each quantity refers to one training pattern p. δ_j^p is the *error signal* at neuron j, i.e. the error contributed by neuron j to the error E^p. For ANNs with sigmoidal activation function, δ_j^p is given by (Alberts, 2002):

$$\delta_j^p = \begin{cases} o_j^p \cdot (1 - o_j^p) \cdot \sum_k \delta_k^p w_{jk}, & \text{if neuron } j \text{ is hidden,} \\ o_j^p \cdot (1 - o_j^p) \cdot (v_j^p - o_j^p), & \text{if neuron } j \text{ is an output neuron,} \end{cases} \tag{2.15}$$

where v_j^p is the desired output of neuron j (if j is an output neuron) after passing pattern p. Equation 2.15 tells us that the value of δ for a particular hidden neuron can be obtained by propagating the δ's backward from the neurons further along in the

network (Bishop, 1995). The back-propagation process is illustrated in Figure 2.10.

Figure 2.11 shows the back-propagation training procedure. There are two modes of training explained in the figure. The adaptation of the weights can be done on a per-pattern basis, i.e. training is done in the *stochastic mode*. Here, weights are changed according to the gradient of the error E^p for each individual training pattern p. However, it is not guaranteed that this change follows the gradient of the total error E for all patterns. To overcome this limitation, all partial errors E^p can be calculated before weights are changed, i.e. training is done in the *batch mode*. This is accomplished by considering the true gradient of the total error, whereby:

$$E(W) = E(..., w_{ij}, ...) = \sum_p E^p(..., w_{ij}, ...). \tag{2.16}$$

Table 2.1 lists advantages of each learning mode. The stochastic learning mode has certain advantages. In particular, it is much faster with large training datasets and with large redundancy (Davies, 2000, *p. 52*). While using the stochastic mode, the training patterns are usually shuffled after each cycle through the patterns. This is because networks learn fastest from the most unexpected (and different) patterns (LeCun et al., 1998a).

Table 2.1: Advantages of both stochastic and batch learning mode. After LeCun et al. (1998b).

Advantages of Stochastic Learning	Advantages of Batch Learning
Usually much faster than batch learning. Often results in better solutions. Can be used for tracking changes.	Conditions of convergence are well understood. Many acceleration techniques (e.g. conjugate gradient) only operate in batch learning. Theoretical analysis of the weight dynamics and convergence rates are simpler.

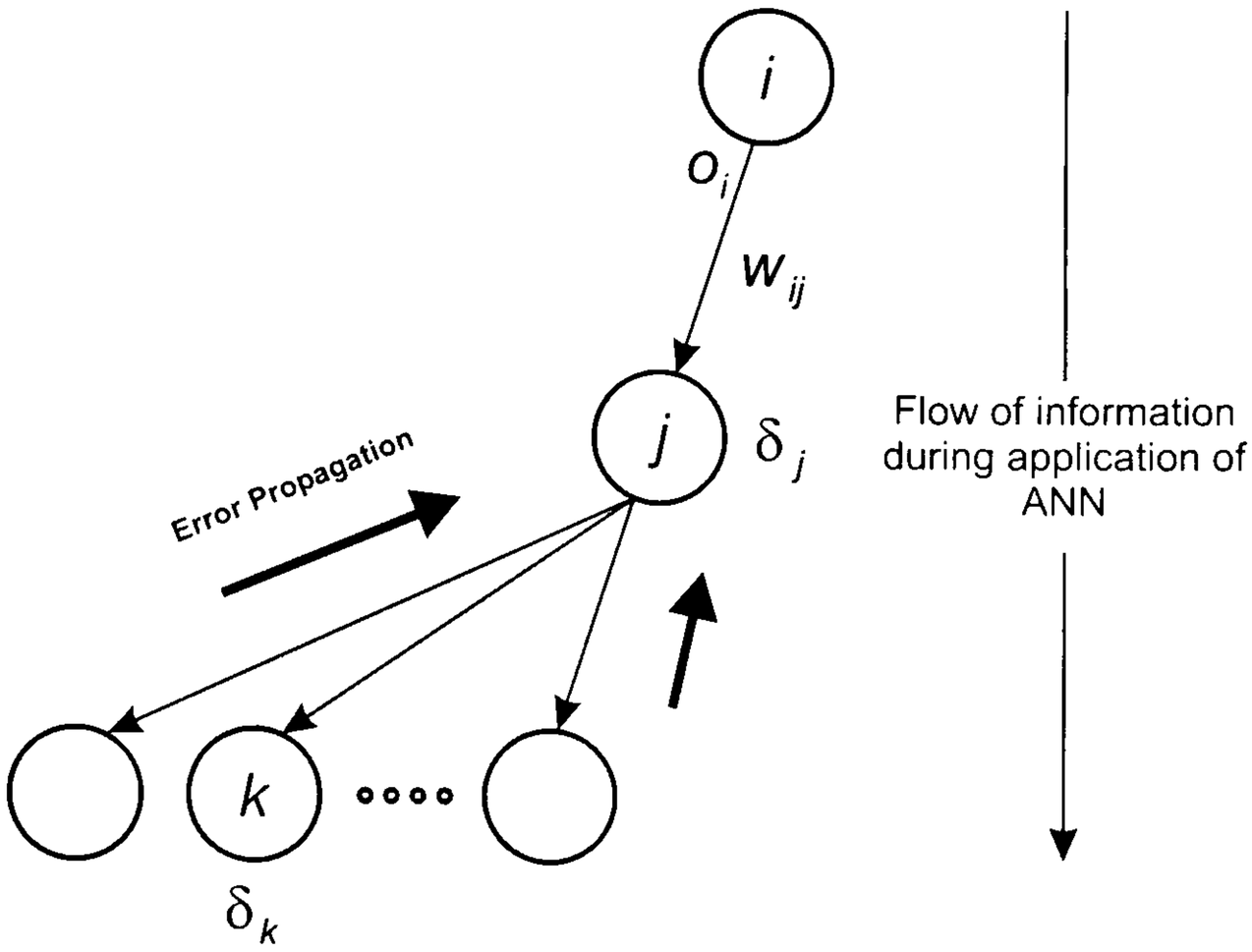


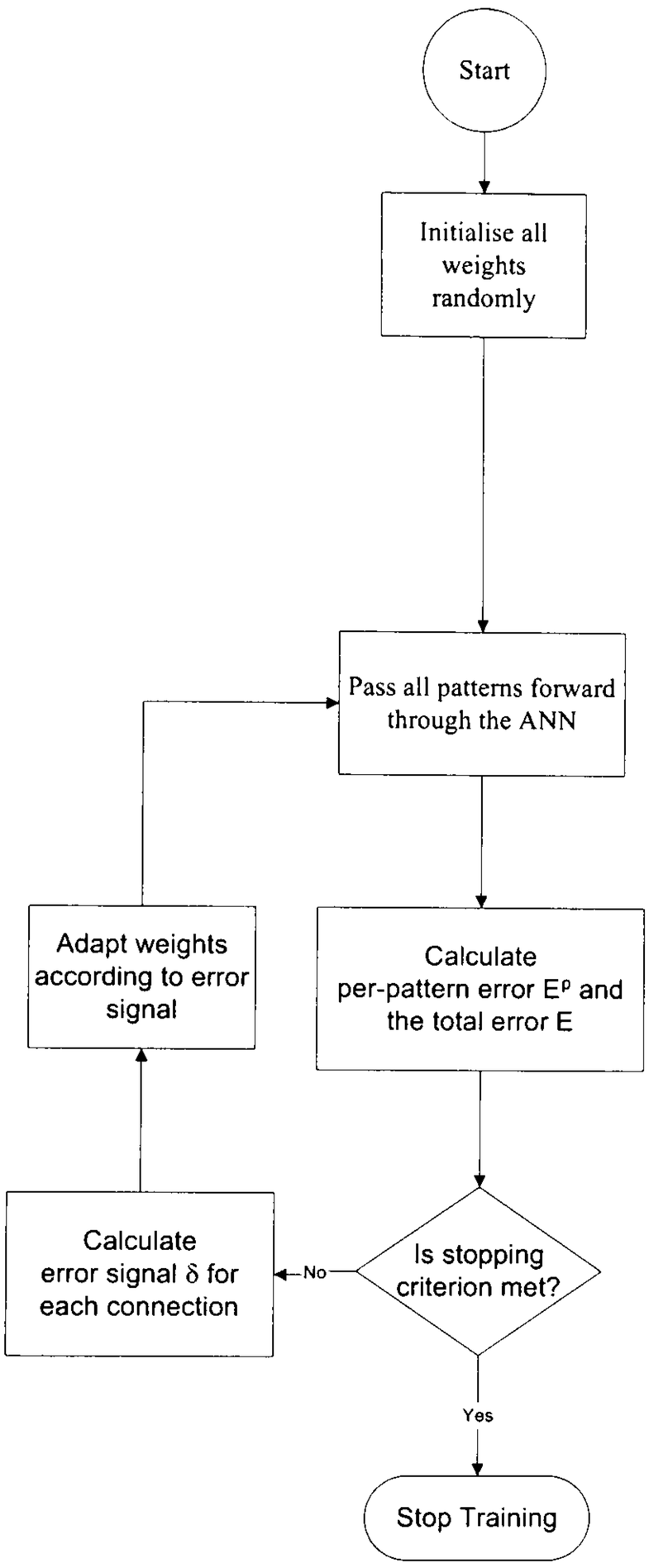
Figure 2.10: Error propagation in back-propagation training algorithm. The value of the error signal δ_j for the hidden neuron j is obtained through the back-propagation of the δ's from those neurons k to which neuron j sends connections. Adapted from Bishop (1995). The connection weight w_{ij} is then modified by an amount Δw_{ij} calculated as a function of δ_j and of the output o_i of neuron i.

2.3.3 Stopping criteria and generalization

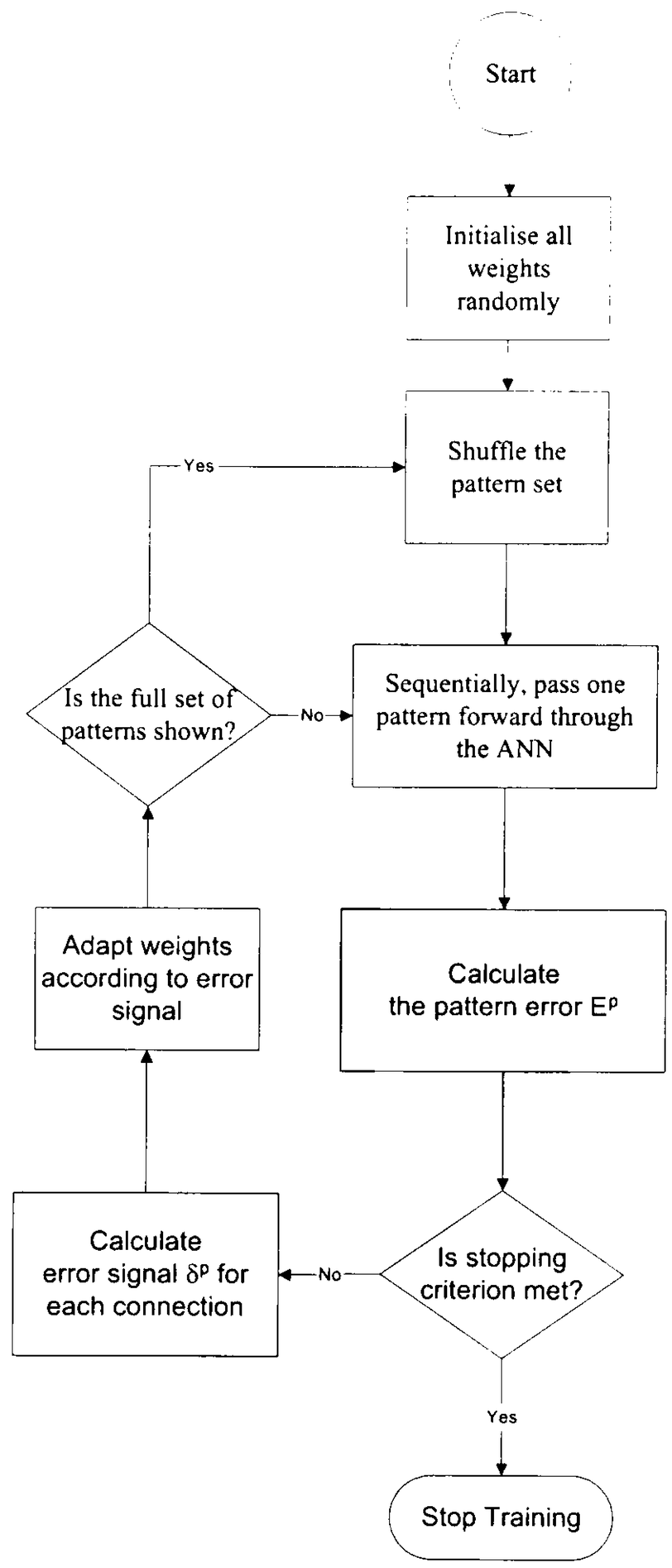
Training ANNs rarely produces a solution whereby perfect input to a target output mapping is done. In practice, training algorithms are searching for a lowest error solution. This rises the question: when should training be stopped, i.e. what is the *stopping criterion*? The question is complicated by two facts: firstly, it is not guaranteed that the training error converges to a minimum or does so in reasonable time; and secondly, training an ANN requires it, in most cases, to learn the mapping of unseen input and thus the lowest error on the training patterns is not necessarily the best solution for the unseen patterns (from the same problem space).

The ability of an ANN to perform well on unseen patterns is called its ability to *generalize*. This generalization is more important in ANN's applications than the network's ability to map the training patterns correctly, since the network is usually intended to solve the unknown case.

There are several methods that can be considered for a stopping criterion. Here are five possible stopping criteria:



(a) Batch back-propagation



(b) Stochastic back-propagation

Figure 2.11: Back-propagation training algorithm. The flow charts illustrate the two possible modes of training for comparison. The algorithm is a cycle in which weights are adapted until the stopping criterion is reached (usually dependent on the error calculated). In batch modes, all errors due to individual patterns are processed before adapting weights. On the other hand, in stochastic mode, the weight adaptation is done after calculating the error due to each individual pattern.

1. Stop when the magnitude of the descent gradient is sufficiently small. However, what if the network is at a flat error surface, see Section 2.4.1.

2. Stop when the error is below a certain threshold. However, this assumes knowledge of the minimum error.

3. Stop when all training examples are correctly mapped. However, this assume the network could do the mapping.

4. Stop after a fixed number of training iterations[2]. However, there is no guarantee to the state of the network at the end.

5. Stop using *cross-validation* where training is monitored using a separate dataset and stopped at the minimum error of ANN application to that dataset. The cross-validation criterion is also known as the *early stopping* criterion. Here, the aim is not to let the network train for too long and hence *over-fit* the training data, whereby the ANN learns to map the training dataset very well but any slight deviation from the training examples causes it to perform badly (poor generalization). The second dataset is obtained by splitting the available patterns into two different sets: a *training dataset* and a *validation dataset* (Section 2.3.1). Using the early stopping criterion, training is stopped at the point of lowest validation error, i.e. the error when the validation dataset is passed through the network. This is illustrated in Figure 2.12. While the training error decreases until the end of the learning as expected, the validation error shows an increase after a certain number of training cycles due to the network's over-fitting of the training dataset.

The first four criteria are sensitive to the choice of parameters and can result in premature termination or unnecessary long training time. On the other hand, cross-validation not only avoids premature termination, but can also improve generalization (Hush and

[2]A training cycle or iteration is completed when the whole training dataset is shown to the network once, see Figure 2.11.

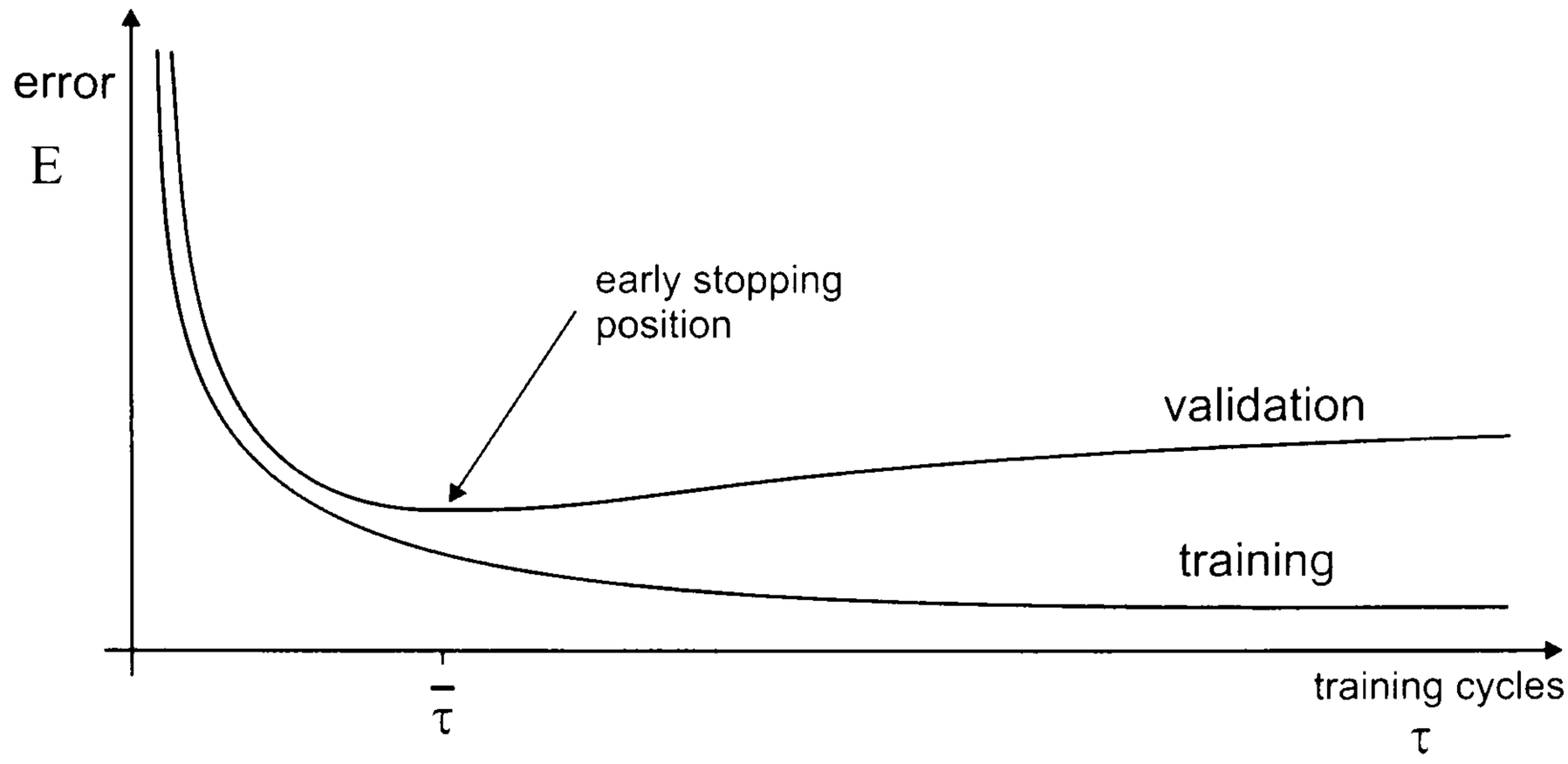


Figure 2.12: The cross-validation stopping criterion. The graph illustrate the development of the validation error and training error during the training phase, as a function of the training cycles τ. To achieve the best generalization training should be stopped at the point $\bar{\tau}$ corresponding to the minimum of the validation error. After Bishop (1995, *p. 345*).

Horne, 1993). However, cross-validation adds an extra computational load. In addition, it reduces the possible training examples and might be a problem if those examples are limited.

2.3.4 Problems with back-propagation

This section summarizes several problems associated with back-propagation and describes a solution. The problems mainly stem from the fact that back-propagation is a gradient descent method and hence works on local information without any knowledge of the global error surface (Alberts, 2002).

1. *Local minimum:* Back-propagation might find a solution at a local error minimum (Figure 2.13). To avoid this, training can be performed a few times each with different weight initialization to ensure starting at a different point on the error surface. The run with the lowest error is then selected as a solution. In addition, *weight jogging*, whereby weights are changed by a small random amount, can help to push the network out of local minima.

2. *Symmetry:* If all connection weights of one neuron have the same value, they will

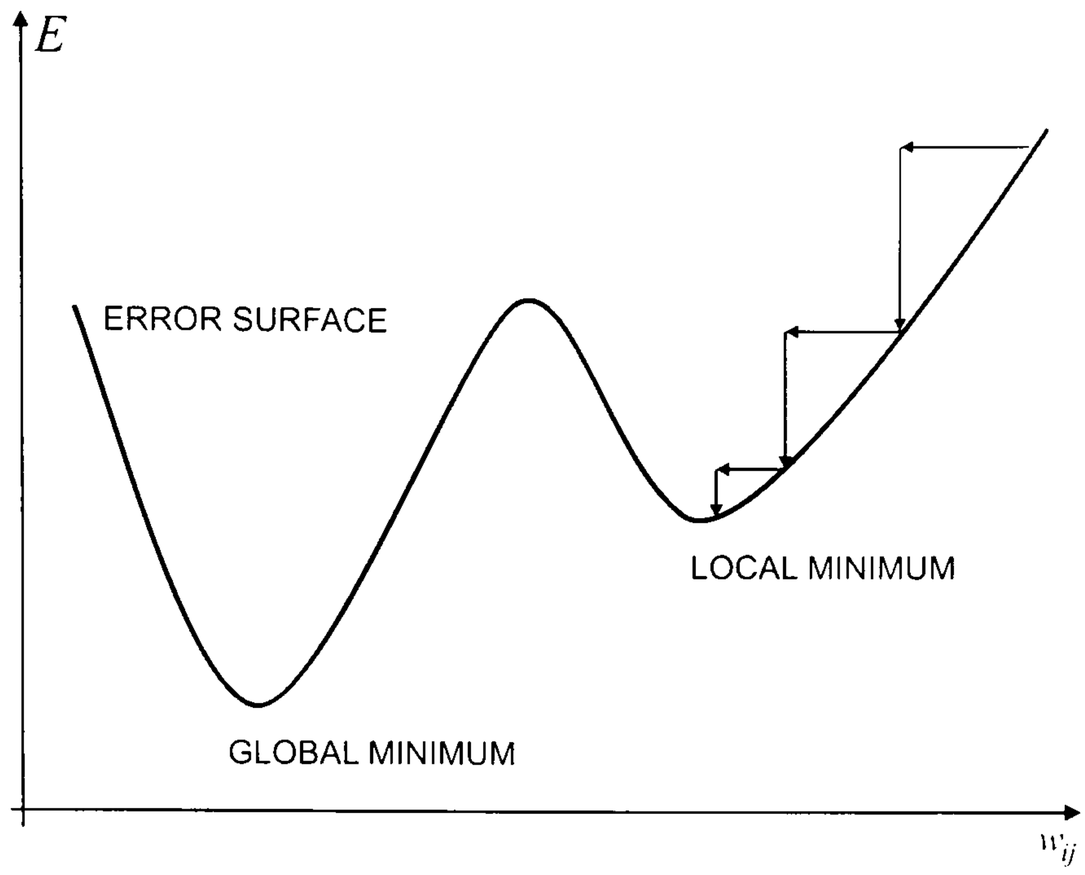


Figure 2.13: Local minima are a problem in back-propagation training algorithm. The graph shows the progress of training along the error surface of one connection weight, where E is the error and w_{ij} is the weight. The gradient descent finds a solution at a local minimum, which is not the true global minimum required. From Alberts (2002).

receive the same weight adaptation. Hence, back-propagation will fail to break this symmetry. To avoid this problem the weights are usually initialized by small random numbers.

3. *Plateaus:* The error surface might be very flat at some points making training very slow (Figure 2.14). A method proposed to get rid of this problem is to introduce a *momentum-term*. Thus, in addition to the weight adaptation due to the error signal, the weight is also changed by a factor μ of the previous weight adaptation $\Delta w_{ij}(t-1)$. Hence, Equation 2.14 is modified so that the new weight adaptation $\Delta w_{ij}(t)$ follows:

$$\Delta w_{ij}(t) = \epsilon\, o_i\, \delta_j + \mu \Delta w_{ij}(t-1). \tag{2.17}$$

Resilient propagation explained in the next section also offers a solution.

4. *Oscillation:* If the error surface has steep and narrow valleys, back-propagation can start to oscillate (Figure 2.15). In this case, the state of the ANN will be

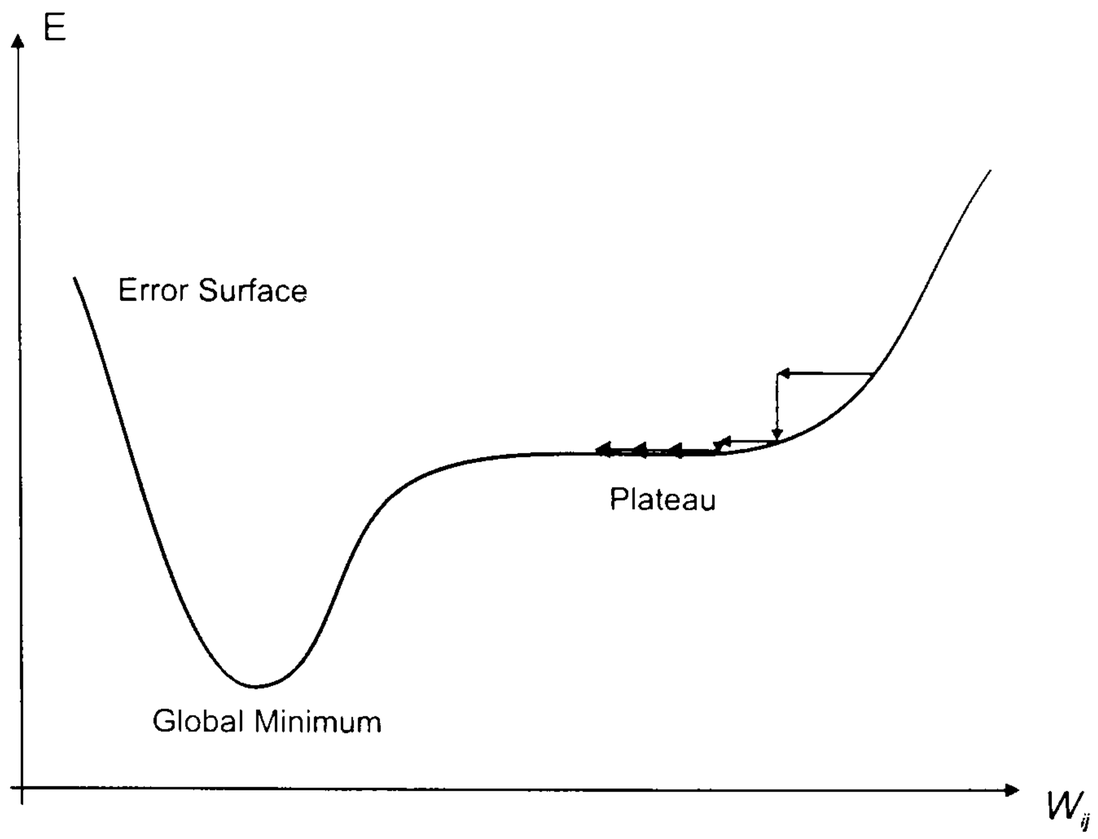


Figure 2.14: Plateaus are a problem in back-propagation training algorithm. The graph shows the progress of training along the error surface of one connection weight, where E is the error and w_{ij} is the weight. The gradient descent progress is a function of the error gradient and hence a plateau acts as a barrier. After Alberts (2002).

jumping from one side of the error surface to the other and hence either gets stuck or descends very slowly. Resilient propagation explained in the next section offers a solution.

5. *Learning rate:* The choice of the learning rate ϵ has an important effect on the rate of convergence and hence training time. Setting this parameter can be difficult. If it is set too small, too many steps are needed to reach an acceptable solution; on the contrary a large learning rate will possibly lead to oscillation, preventing the error from falling below a certain value.

   The momentum-term can accelerate convergence in shallow regions of the error function. However, Riedmiller and Braun (1993) say that the optimal value of the momentum parameter μ is equally problem dependent as the learning rate ϵ, and that no general improvement can be accomplished. On the other hand, resilient propagation explained in the next section and other adaptive training algorithms offer a solution.

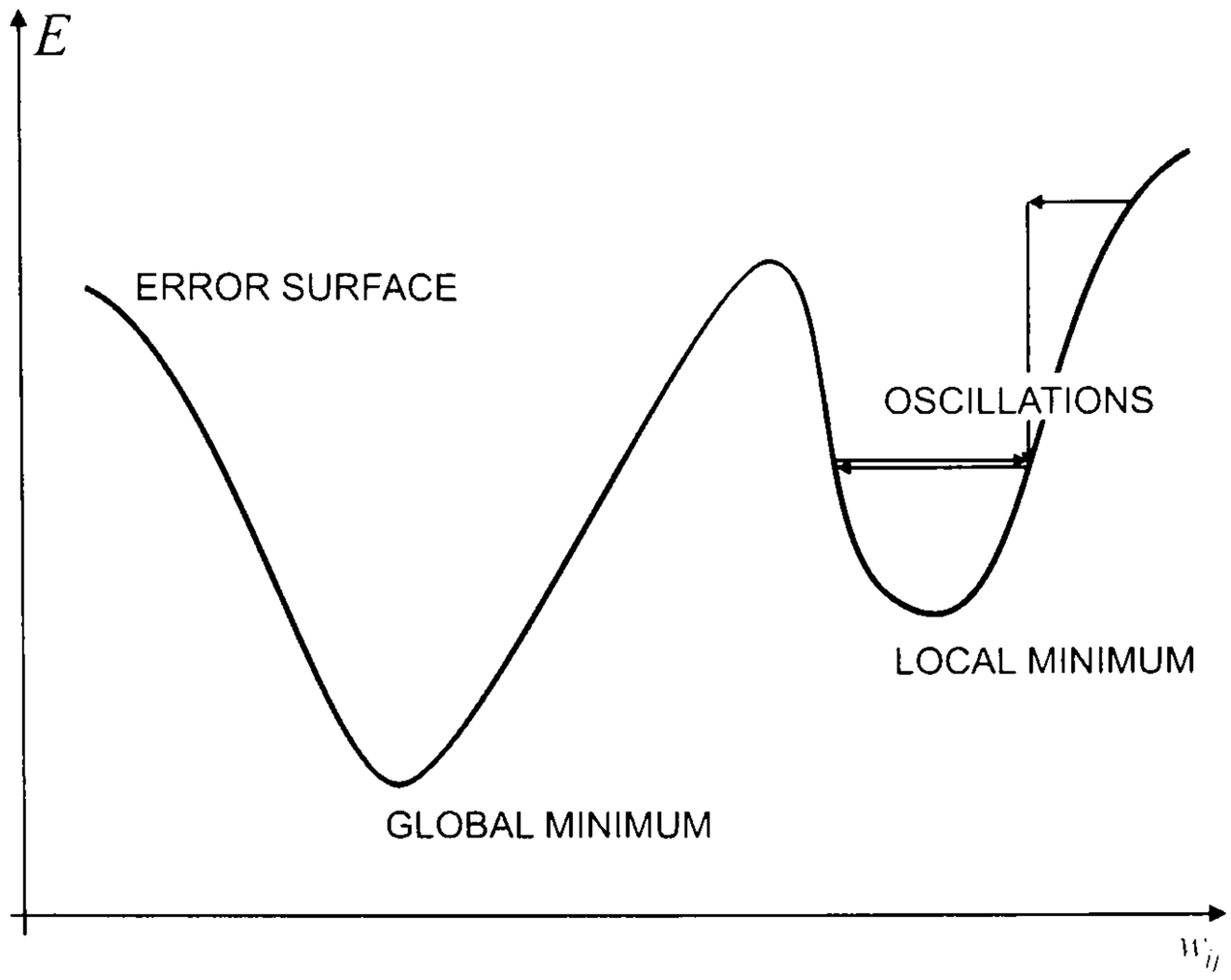


Figure 2.15: Oscillation problem in back-propagation training algorithm. The graph shows the progress of training along the error surface of one connection weight, where E is the error and w_{ij} is the weight. Steep and narrow valleys in the error surface cause the back-propagation training algorithm to oscillate, and hence it either gets stuck or descends very slowly. From Alberts (2002).

2.3.5 Resilient propagation (RPROP)

The problems associated with back-propagation led to the design of other algorithms that can overcome those problems. In addition, algorithms were brought forward that might speed up training and reduce the time and effort in finding suitable learning parameters. Examples of such algorithms is *Quickprop* (Fahlmann, 1989) and *SuperSAB* (Tollenaere, 1990). Schiffmann et al. (1992, 1994) investigated and compared many of such algorithms.

Many algorithms were developed where parameter adaptation is possible during learning. These can roughly be split into global adaptation, which make use of the knowledge of the whole state of the network; and local adaptation, where weight-specific information (e.g. the partial derivative) is used to adapt weight-specific parameters. Riedmiller and Braun (1993) explain that local strategies are superior and better suited in parallel implementations. In addition, Schiffmann et al. (1994) say that local adaptation strategies greatly reduce the training time and also improve network performance.

In most adaptive training algorithms, the size of the actually weight-step Δw_{ij} is not only dependent on the (adapted) learning-rate, but also on the partial derivative $\frac{\partial E}{\partial w_{ij}}$. Although the learning-rate is carefully adapted, it is nevertheless drastically disturbed by the unforeseeable behaviour of the derivative itself (Riedmiller and Braun, 1993).

Resilient propagation (RPROP) is an efficient learning scheme introduced by Riedmiller and Braun (1992, 1993). It performs a direct adaptation of the weight step based on local gradient information. Its crucial difference to previously developed adaptation techniques is that the weight adaptation does not depend on the magnitude of the gradient; instead, RPROP changes the size of the weight-update Δw_{ij} directly without considering the size of the partial derivative. Riedmiller and Braun (1993) show how the RPROP algorithm compares with other learning strategies and prove its superiority in most cases. Furthermore, RPROP builds on the experience made with other adaptive training algorithms such as *Manhattan-Training*, *Quickprop* and *SuperSAB* and unites their advantages (Alberts, 2002, *p. 66*). RPROP achieves its adaptive weight update by introducing an individual *adaptive update-value* Δ_{ij} for each weight w_{ij}. This Δ_{ij} solely determines the size of the weight-update. It evolves during the learning process based on its local knowledge of the error function E.

The RPROP algorithm, as explained by Riedmiller and Braun (1993), works as follows: Every time the partial derivative of the corresponding weight w_{ij} changes its sign, which indicates that the last update was too big and the algorithm has jumped over a minimum, the update-value Δ_{ij} is decreased by a *decrease factor* η^-. If the derivative retains its sign, the update-value is slightly increased by an *increase factor* η^+ in order to accelerate convergence in shallow regions. Therefore, the adaptation works according to the following *learning-rule*:

$$\Delta_{ij}(t) = \begin{cases} \eta^+ \times \Delta_{ij}(t-1) & \text{if } \frac{\partial E}{\partial w_{ij}}(t-1) \times \frac{\partial E}{\partial w_{ij}}(t) > 0, \\ \eta^- \times \Delta_{ij}(t-1) & \text{if } \frac{\partial E}{\partial w_{ij}}(t-1) \times \frac{\partial E}{\partial w_{ij}}(t) < 0, \\ \Delta_{ij}(t-1) & \text{otherwise.} \end{cases} \tag{2.18}$$

$$\text{where} \quad 0 < \eta^- < 1 < \eta^+.$$

Once the update-value for each weight is adapted, the weight-update itself follows a simple rule: if the derivative is positive (increasing error), the weight is decreased by its update-value, if the derivative is negative, the update-value is added. Therefore,

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t), \text{ where} \tag{2.19}$$

$$\Delta w_{ij}(t) = \begin{cases} -\Delta_{ij}(t) & \text{if } \frac{\partial E}{\partial w_{ij}}(t) > 0, \\ +\Delta_{ij}(t) & \text{if } \frac{\partial E}{\partial w_{ij}}(t) < 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.20}$$

There is one exception: If the partial derivative changes sign, i.e. the previous step was too large and the minimum was missed, the previous weight-update is reverted:

$$\Delta w_{ij}(t) = -\Delta w_{ij}(t-1) \quad \text{if } \frac{\partial E}{\partial w_{ij}}(t-1) \times \frac{\partial E}{\partial w_{ij}}(t) < 0 \tag{2.21}$$

Due to this *backtracking* weight-step, the derivative will change its sign once again in the following step. In order to avoid a double punishment of the update-value, there is no adaptation of the update-value in the succeeding step. In practice, this is done by setting $\frac{\partial E}{\partial w_{ij}}(t-1) = 0$ in the Δ_{ij} adaptation-rule (Equation 2.18).

The update-values and the weights are changed every time the whole pattern set has been presented once to the network, i.e. learning in batch mode (Figure 2.11).

Riedmiller and Braun (1993) explain that all update-values Δ_{ij} are initially set to an initial value Δ_0. A good choice of Δ_0 is 0.1, but this parameter is not critical, since the update-values are adaptive. Furthermore, Riedmiller and Braun (1993) introduce an upper limit of $\Delta_{\max} = 50.0$ and a lower limit of $\Delta_{\min} = 1e^{-6}$ for the update-values. A reasonable choice of the decrease factor η^- and the increase factor η^+ is 0.5 and 1.2, respectively. For further discussion on these choices refer to Riedmiller and Braun (1993).

2.4 Practical issues

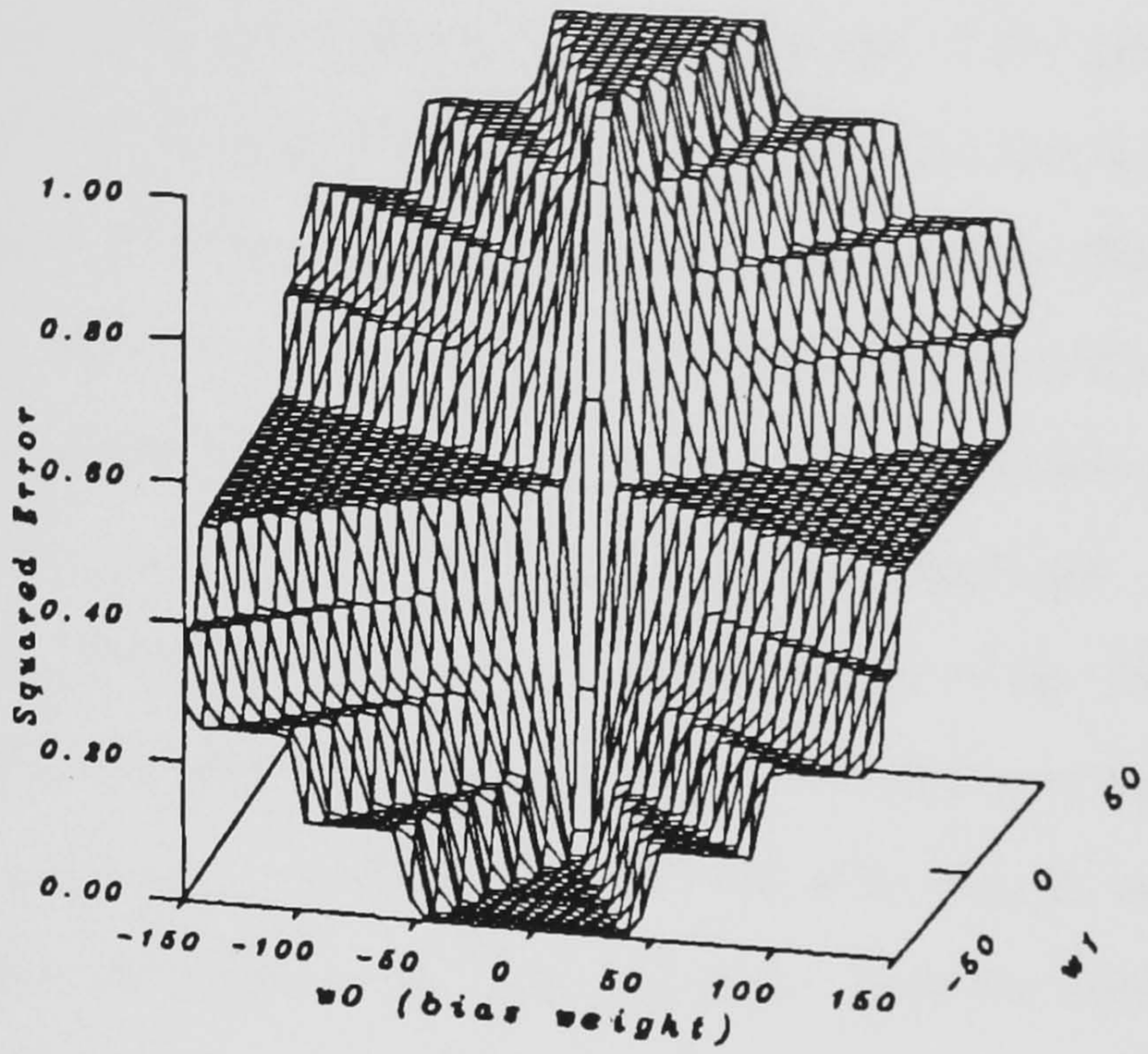
2.4.1 Error surface complexity

Although, multi-layer perceptrons (MLPs) are the most popular ANNs and are successful in a variety of problems, they have their own weaknesses. Their major weakness lies in their slow learning rates (Hush et al., 1991). Hush et al. (1991, 1992) demonstrate that the slow rate of learning is a consequence of the complexity of the error surface in the MLPs.

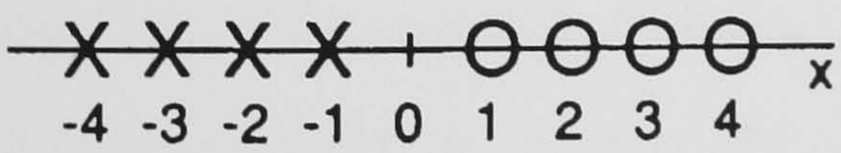
Figure 2.16 illustrates the character of an error surface of an MLP. Although this figure shows one particular example, Hush et al. (1991, 1992) showed that the error surface of an MLP, with sigmoid activation function, always have the same appearance. In particular, the surface has a stair-step appearance with many very flat and very steep regions. Thus, the magnitude of the gradient varies by several orders of magnitude. In addition, as the number of training patterns increase the surface becomes smoother. Furthermore, the surface has flat regions that extend to infinity.

Hush et al. (1991, 1992) worked on pattern recognition, and hence some of their observations might not be in line with other uses. However, their work give some insight. In particular, back-propagation will suffer largely due to the plateaus. Therefore, back-propagation with momentum and RPROP function better. The existence of flat regions that extend to infinity can cause training to continue without escaping this region. The variation of the gradient by several orders of magnitude, requires the error to be stored with floating point representation. In addition, the more patterns available the smoother the error surface and hence the faster the training.

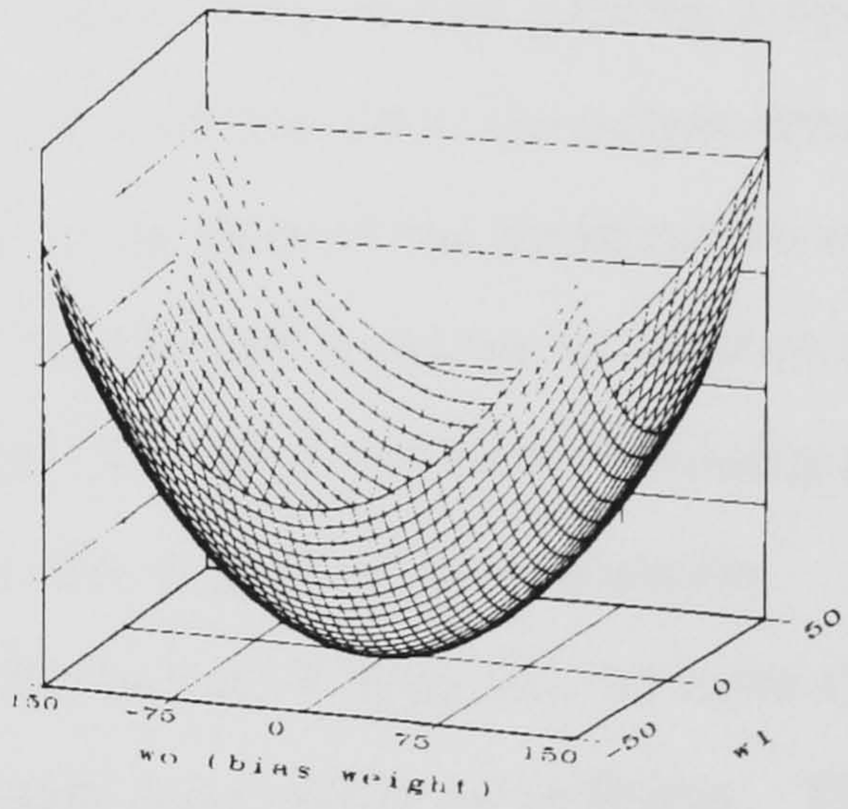
Hush et al. (1992) also found that if the desired output is changed from $[0, 1]$ to a lower range e.g. $[0.1, 0.9]$, this will introduce a dip at the bottom of each stair step. This is because the perceptron error is minimum at exactly 0.1 or 0.9 but larger at either side. There are motivations for choosing $[0.1, 0.9]$, as will be described on Page 92, but it is useful to be aware of the implication.



(a)



(b)



(c)

Figure 2.16: (a) The error surface of a one-node MLP network as a function of the connection weight w_1 and the threshold or bias weight w_0 (from Hush et al. (1992)). (b) The training samples used, where the input x is one of two classes denoted by either crosses or circles (from Hush et al. (1992)). (c) The error surface if a linear activation function was used — put only for comparison with (a) (from Hush and Horne (1993)). Note the simple bowl shape of (c) and in comparison the complexity and stair-step appearance in (a). The error surface displays a form of symmetry on w_0 and w_1 and the steps are a function of the number of patterns used.

2.4.2 Size and topology

The choice of the network size and topology is important. If the network is too small, it will not be capable of forming a good model of the problem (Hush and Horne, 1993). On the other hand, if it is too large then it may be too capable (Baum and Haussler, 1989), whereby the network can implement numerous solutions to the training patterns but with poor generalization. In general, one cannot pre-determine a network size for a given problem since each problem demands a different capability.

Hush and Horne (1993) say that with some specific knowledge about the structure of the problem and a fundamental understanding of the MLP, one can sometimes form a good estimate of the proper network size. However, with little or no prior knowledge, one must determine the network by trial and error. Usually, one will try different networks in a methodological way, whereby the network size is varied systematically by either starting from a small network and moving to a large one or vice versa.

Hush (1989) recommends starting with a small network and growing it systematically. He says if the performance of the network is monitored as the output error versus network size, then the optimal network size is the knee of the performance curve, as illustrated in Figure 2.17. This is because the size will continue to improve the performance significantly until some optimal size is reached and then increasing size will only increases performance slightly but will start to degrade generalization.

Following from the previous discussion (Section 2.2.2, page 67), no more than two hidden layers are required. In fact, one hidden layer might be sufficient. There are many studies that discuss the bounds to the number of nodes that should be used in an MLP, e.g. Baum and Haussler (1989), Huang and Huang (1991), and Huang and Babri (1998). One conclusion that can be drawn from such studies is that one should never use more hidden nodes than training patterns, otherwise the network will simply memorize the training dataset and become too specific to it.

Other than trial and error, methods and algorithms exist that can help in finding a suitable network size. These include growing algorithms that start from a small network

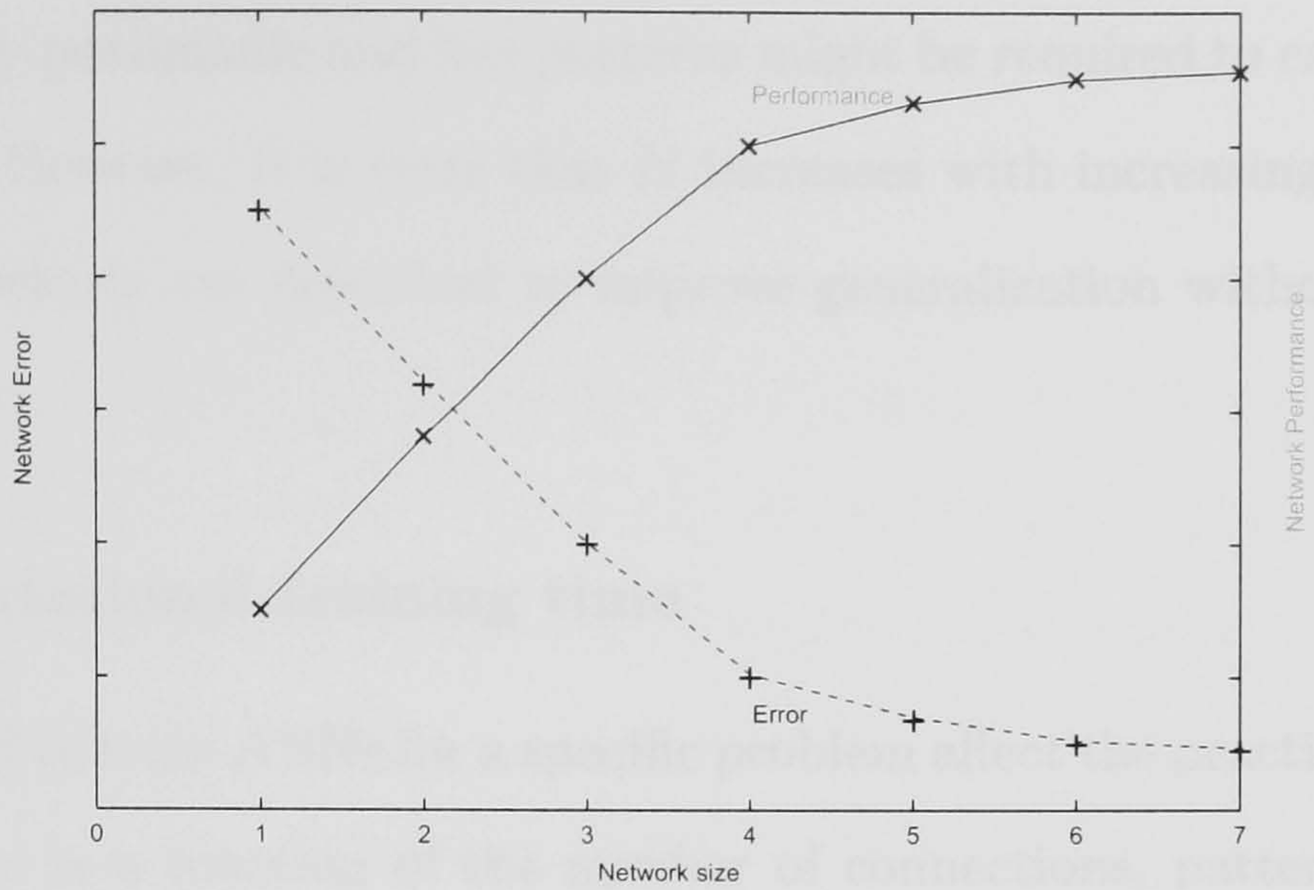


Figure 2.17: Performance versus ANN's size. The graph is a general illustration of the method recommended by Hush (1989), see text. The graph shows how the error (inverse of performance) varies with different network sizes. Here, the 4th network size forms the knee of the error curve, whereby the performance increase significantly before the 4th size but does not do so after it. Hence, the 4th network seems the most appropriate network size to use.

and pruning algorithms that start from a large network. An example of the former is the Cascade Correlation (Fahlman and Lebiere, 1990) and of the latter is the Optimal Brain Damage (LeCun et al., 1990b).

2.4.3 Pattern size and choice

The choice of training and validation patterns is also important. This choice will both determine the speed and likelihood of finding a solution and achieving generalization. These patterns should be representative of the whole sample space where the network is to be used. To be representative, examples should be chosen randomly from all possible selections within the problem space.

On the issue of pattern size, this is both related to the complexity of the problem and the size of the network used. If the best network size is predetermined, then results from Vapnik and Chervonenkis (1971) suggest that for a good generalization and as a rule of thumb, the number of training patterns N should be approximately ten times (or more) the number of weights (Hush and Horne, 1993). These results are addressing classification problems where one needs to capture all possible classes. Therefore, the

results are slightly pessimistic and less patterns might be required to capture functional approximations. However, it is clear that N increases with increasing connections. In Section 2.4.5, methods are described to improve generalization without solely relying on increasing N.

2.4.4 Computational training time

The time required to train ANNs for a specific problem affect the practicality of utilizing ANNs. This time is a function of the number of connections, patterns, and training cycles. In this study, I utilize a cluster of 16 machines running in parallel. These are 500 MHz EV5 21164 Alpha Workstations. It is estimated that a typical project (matching 512 ms from 2000 CMPs along the two time-lapse datasets) involving around 130 separate networks takes around one week on the cluster. This could increase to two weeks or decrease to two days depending on how the problem is set. In effect, each network requires 2–10 CPU hours to train. The speed is around six times faster in a 1.5 GHz Pentium IV machine. A more exact training time is given in Chapter 5.

2.4.5 Generalization

Generalization is most heavily influenced by three factors: the network size, the number of training examples and how representative these are, and the complexity of the underlying problem (Hush and Horne, 1993). The first two have already been discussed. The third cannot be chosen, but can be manipulated during pattern pre-processing as will be discussed in the next section.

In addition to the three factors outlined above, one could use algorithms that help to improve generalization during training. There are two principal methods: firstly, through reducing the number of connection weights since larger networks are more likely to over-fit training patterns; and secondly, through reducing the weight size.

Reducing the number of weights, and hence the network size, can be done through pruning algorithms such as Optimal Brain Damage (LeCun et al., 1990b). Here the

network is firstly trained using back-propagation. Then the weights with the smallest influence are deleted and the network is retrained. The process is usually repeated.

The method, whereby the weight size is reduced, is called *weight decay* (Werbos, 1988). It is a form of driving the network to less complexity. This is achieved by adding a term to Equation 2.14 to modify the weight update from being just a function of the output error signal to that which includes a penalty to the weight's size. Thus,

$$\Delta w_{ij} = \epsilon\, o_i\, \delta_j - d\, w_{ij}, \tag{2.22}$$

where d is a constant. Usually, this is achieved through modifying the error function of the whole network. In my use of RPROP, the error function (Equation 2.11) is modified to the following (Zell et al., 2000):

$$E = \sum (v_i - u_i)^2 + 10^{-\alpha} \sum w_{ij}^2, \tag{2.23}$$

where α is a constant called the *weight-decay term.* The first sum in Equation 2.23 defines the usual network's error that back-propagation (or RPROP) tries to minimize, i.e. the misfit between the target output v and the actual output u. The second sum defines a cost function to the weight size. Hence, the weight-decay term α determines the relationship of two goals, namely to reduce the output error (the standard goal) and to reduce the size of the weights (to improve generalization). Note that the weight decay parameter α denotes the exponent, to allow comfortable input of very small weight-decay. A choice of $\alpha = 2$ corresponds to a ratio of weight decay term to output error of $1 : 100$ $(1 : 10^2)$.

Hush and Horne (1993) give an explanation to how weight decay improves generalization. In summary, the weights of the network fall into two categories: those that have a large influence on the solution, and those that have little or no influence on the solution. The second, which can be called *excess weights*, can take wide range of values without significantly affecting the solution. They are not likely to take values near zero

unless encouraged to do so. They are more likely to take values that are either arbitrary values, or that cause the network to over-fit the training data to slightly reduce training error. Therefore, either case leads to poor generalization. The addition of the second term in Equation 2.23 encourages excess weights to take on zero or near zero values and hence this improves generalization.

In addition to improving generalization, weight decay has another advantage, which I found particularly useful. After learning with weight decay, the magnitude of each weight is directly proportional to its influence on the mapping error (Hush and Horne, 1993). Therefore, connections with relatively small weights can be deleted in order to speed up training in similar uses and to improve generalization. Furthermore, if all connections stemming from one particular input node have small weights then this signifies that this input is not useful and can be omitted. On the other hand, if all connections from one input node have large weights then this input is important. These facts allows one to decide what inputs are important and might lead to deciding the nature of input that is required for a particular problem. I also found out that if many connection weights within the hidden layers are near zero, then this signifies that the topology used is larger than necessary.

2.4.6 Pattern pre-processing

The ability of ANNs to perform arbitrary non-linear functional mapping between sets of variables may encourage some to use ANNs to map raw input data directly onto the required target output data. In practice, this gives poor performance.

The poor performance is as a result of what is termed the *curse of dimensionality* associated with increasing number of input data (Bishop, 1995, *p.* 7). As the number of input data increases the number of connection weights increases many orders of magnitude more, and hence more pattern examples are needed to determine the optimum weight matrix W. Since most applications have a limited number of training patterns, there is always a point beyond which adding more input information only serves to

deteriorate the performance of the network.

Another cause of poor performance is patterns data not being in a value range at which the ANNs behave best. Therefore, for most applications, it is necessary first to transform the data into a new representation before passing them through the network (Bishop, 1995, *p. 295*). This is done in the *pre-processing* stage that precedes the training stage.

Pre-processing can be a simple linear transform of pattern data to allow ANNs to process these data more efficiently (see the following section). One might also transform the data to another domain, e.g. from time domain to frequency domain. This is done when the new representation offers a more compressed feature representation or when it allows the problem to be split into many smaller ones.

More complex pre-processing involves the reduction of the dimensionality of the input data. This can be done through *principal component analysis*. In addition, one could reduce the dimensionality by making an intelligent choice of the input and output patterns that best represent the problem. This *feature extraction* is either done through prior knowledge of the problem or through assessing the performance of the network when given a particular set of features as input and output.

Normalization

Input and output patterns are usually composed of different attributes and features from the problem space. These could have values differing by several orders of magnitude. Furthermore, in Section 2.1.3 it was shown that many ANN models (including ANNs used in this study) restrict the possible activations to an interval of $[0, 1]$. It is therefore desirable to have all the patterns in a range of similar order of magnitude by applying a form of normalization to these patterns. This also allows the initialization of the weights of the network to be random with values in a similar range. Preferably, the input variables should have an average close to zero, where convergence can be expected to be faster (LeCun et al., 1998b).

The sigmoid activation function produces an output in the range $[0, 1]$ (Figure 2.7). Thus, the target outputs have to be normalized to this interval. This can be done through linear transformation, where a target output v_i for a particular output node i and all patterns 1 to N is transformed to a normalized representation $\tilde{v_i}$ through the following equation:

$$\tilde{v_i} = \frac{v_i - v_i^{\min}}{v_i^{\max} - v_i^{\min}}. \tag{2.24}$$

$v_i^{\min}$ and $v_i^{\max}$ are the minimum and maximum value of v_i, respectively.

The output is sometimes desired to be normalized to the interval $[0.05, 0.95]$. This is to avoid requesting the network to output the sigmoid (activation function) asymptotic values 0 and 1. This can be done through the following modification of Equation 2.24:

$$\tilde{v_i} = 0.9\left(\frac{v_i - v_i^{\min}}{v_i^{\max} - v_i^{\min}}\right) + 0.05. \tag{2.25}$$

The input patterns can be normalized in a similar fashion to the output. Therefore, an input x_i for a particular input node i and all patterns 1 to N is transformed to a normalized representation $\tilde{x_i}$ through the following equation:

$$\tilde{x_i} = \frac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}}, \tag{2.26}$$

There is no desire to limit the input to the range $[0.05, 0.95]$. However, for the input x_i many authors, e.g. Bishop (1995), suggest normalizing it by its mean $\overline{x_i}$ and standard deviation σ_i, where:

$$\tilde{x_i} = \frac{x_i - \overline{x_i}}{\sigma_i}. \tag{2.27}$$

If there are outliers within the pattern data, then most values normalized through Equation 2.24 and Equation 2.26 will end up localized to a very small range and hence lead to a lowering of the network's performance. If we define y_i to be either input x_i or output v_i, then the problem can be avoided by setting the minimum and maximum of y_i to be three standard deviations σ_i away from the mean $\overline{y_i}$. Thus, we define for y_i

a new minimum $y_i^{\tilde{\text{min}}}$ and maximum $y_i^{\tilde{\text{max}}}$, where:

$$\begin{aligned} y_i^{\tilde{\text{min}}} &= \overline{y_i} - 3\sigma_i, \text{ and} \\ y_i^{\tilde{\text{max}}} &= \overline{y_i} + 3\sigma_i. \end{aligned} \tag{2.28}$$

Values of y_i that are lower than y_i^{min} are clipped to y_i^{min} and values of y_i that are higher than $y_i^{\tilde{\text{max}}}$ are clipped to $y_i^{\tilde{\text{max}}}$. Therefore, Equation 2.24 and Equation 2.26 are modified by setting y_i^{min} and y_i^{max} to $y_i^{\tilde{\text{min}}}$ and $y_i^{\tilde{\text{max}}}$, respectively.

Normalization is discussed further in Section 4.4.1, where an optimal choice is given.

2.4.7 ANNs in this study

The artificial neural networks implemented in this study are trained using the *Stuttgart Neural Network Simulator* (SNNS), v4.2 (Zell et al., 2000). This simulator is a software package developed at the University of Stuttgart to create an efficient and flexible simulation environment for research on ANNs and their applications. SNNS found its way into many other studies, e.g. van der Baan and Jutten (2000), Davies (2000), and Alberts (2002).

SNNS allows for the use of many training algorithms, e.g. back-propagation, Quick-prop, and RPROP. It also allows and caters for different optimizations such as momentum (Equation 2.17), weight decay (Equation 2.23), and weight jogging (Section 2.3.4(1)). In addition, it caters for topology growing and pruning algorithms. In fact, the simulator is designed to cater for most popular training and optimization algorithms and to provide for their parameter choices.

SNNS has a graphical interface that allows an interactive way of designing and training ANNs. It contains visual tools to show the progress of the training and the validation error, and to help analyse the ANNs (e.g. to analyse the connection weights, topology, and activation of each node). In addition to the graphical interface, there is an additional interface to the SNNS kernel that allows easy background execution. This is done through the *batchman* program provided by SNNS. This interface is the most

practical way of using SNNS, since it allows the setting of jobs in batch mode. It is primarily made to train SNNS networks and set all the parameters required. However, it also allows the user to make wide range of analysis, to design specific training and stopping criteria, and allows sufficient input–output functionality.

The networks created and used in SNNS are saved as ASCII files, which allows a direct inspection of the structure of these files. This was very useful in this study as many ANNs are created automatically using programs outside SNNS. Once training is completed, SNNS provides a program called *snns2c* that creates a *C*-language function of the trained ANNs. This makes it easy to implement the trained networks. Patterns for training, validation, and application are stored in *pattern files* and should be designed in a special format in ASCII mode to allow SNNS to deal with them. These pattern files are easy to understand and can easily be prepared by other programs outside SNNS.

The preceding discussion clearly shows that SNNS provides a very capable and flexible environment for research on ANNs. All these merits helped to decide on using SNNS. The only disadvantage I note is that SNNS use of ASCII pattern files demands large computer memory (hard-disk) space. If these pattern files are stored in binary mode the space needed can easily drop by an order of magnitude or more.

This study employed the RPROP training algorithm with weight decay as described in Zell et al. (2000, *p. 148*). This algorithm basically implements RPROP in batch learning as described in Section 2.3.5 and is modified to include weight decay as in Equation 2.23. The use of RPROP allows the overcoming of some of the problems associated with back-propagation. In addition, it speeds up training, and reduces the time and effort in finding suitable learning parameters.

In this project, codes outside SNNS, written in *C++* language, are used to create and prepare training patterns, split them into training and validation datasets. write the patterns in the format required by SNNS, and create the required ANNs and batchman training scripts.

Chapter 3

Time-lapse Seismic Datasets

This study is conducted on data from two time-lapse 3D seismic surveys over the Oseberg field, North Sea. The two 3D seismic lines, NH8906 and NH9201 were acquired during 1989 and 1992, respectively.

A copy of the seismic data of the two surveys were supplied by *Total*, to whom we are thankful, in SEG-Y format. Initially it was intended that the project would include two full 3D surveys. However, only one time-lapse swath from each survey was delivered and hence involved in this study. Nevertheless, the two time-lapse swaths were sufficient to enable a scheme to be developed and an analysis to be made.

In this chapter, the Oseberg field and its time-lapse project are introduced. The introduction is followed by analysis of the two datasets used in this study and a summary of previous work on them. Finally the need for cross-equalization will be outlined.

3.1 Oseberg field

The Oseberg field is located in the Norwegian North Sea 140 km northwest of Bergen, see Figure 3.1. This field is located in blocks 30/6 and 30/9, at 100 m water-depth, and is about 26 km long and 6 km wide. With around 527.5 million Sm^3 of initial oil in place and estimated recoverable reserves of 319.3[1] million Sm^3, Oseberg field is

[1]Revised later by Erlandsen (2000) to 336 million Sm^3, see Section 3.1.3

the second largest hydrocarbon field in the Norwegian North Sea sector (Ryseth et al., 1998).

Recent studies on this field include that on stratigraphy by Ryseth et al. (1998), stratigraphy and production by Johnsen et al. (1995), 4D seismic by Burns (1998), and reservoir management by Sognesand (1997) and Burns (1997).

3.1.1 Structure and stratigraphy

The Oseberg field is located on the Bergen High at the western margin of the Horda Platform. It comprises three major fault blocks called Alpha, Gamma, and Alpha North with a gentle dip of 6–10° east–northeast, see Figure 3.2 and Figure 3.3. The reservoir rocks are deltaic sandstones of the Early to Mid Jurassic Brent Group, which varies in thickness from 45 to 190 m (Johnsen et al., 1995). The cross-section in Figure 3.2 shows that the reservoir rocks are thickest on the down-faulted structure and are overlain by 2100–2700 m of sediment cover.

The field lies within the Brent province, which is geographically part of the Viking Graben and its flanks (Figure 3.2). In this province, structural evolution controlled not only rock deposition but also hydrocarbon migration and trapping. In fact hydrocarbon trapping within the Brent province is almost exclusively related to the Mesozoic fault blocks (Ryseth et al., 1998).

Two phases of rifting represent the main events in the structural evolution of the area (Ryseth et al., 1998). The first phase of rifting and crustal extension started in the Late Permian–Early Triassic and was followed by postrift thermal subsidence until Middle Jurassic. The second phase of crustal extension occurred in the Late Jurassic–Early Cretaceous, followed by postrift thermal subsidence.

The stratigraphic column at Oseberg is illustrated in Figure 3.4. Following is the geological history of the area as summarized from Ryseth et al. (1998):

> Continental conditions prevailed throughout the Triassic and the Early Jurassic during which more than 2 km of mainly fluvial sediments of Hegre

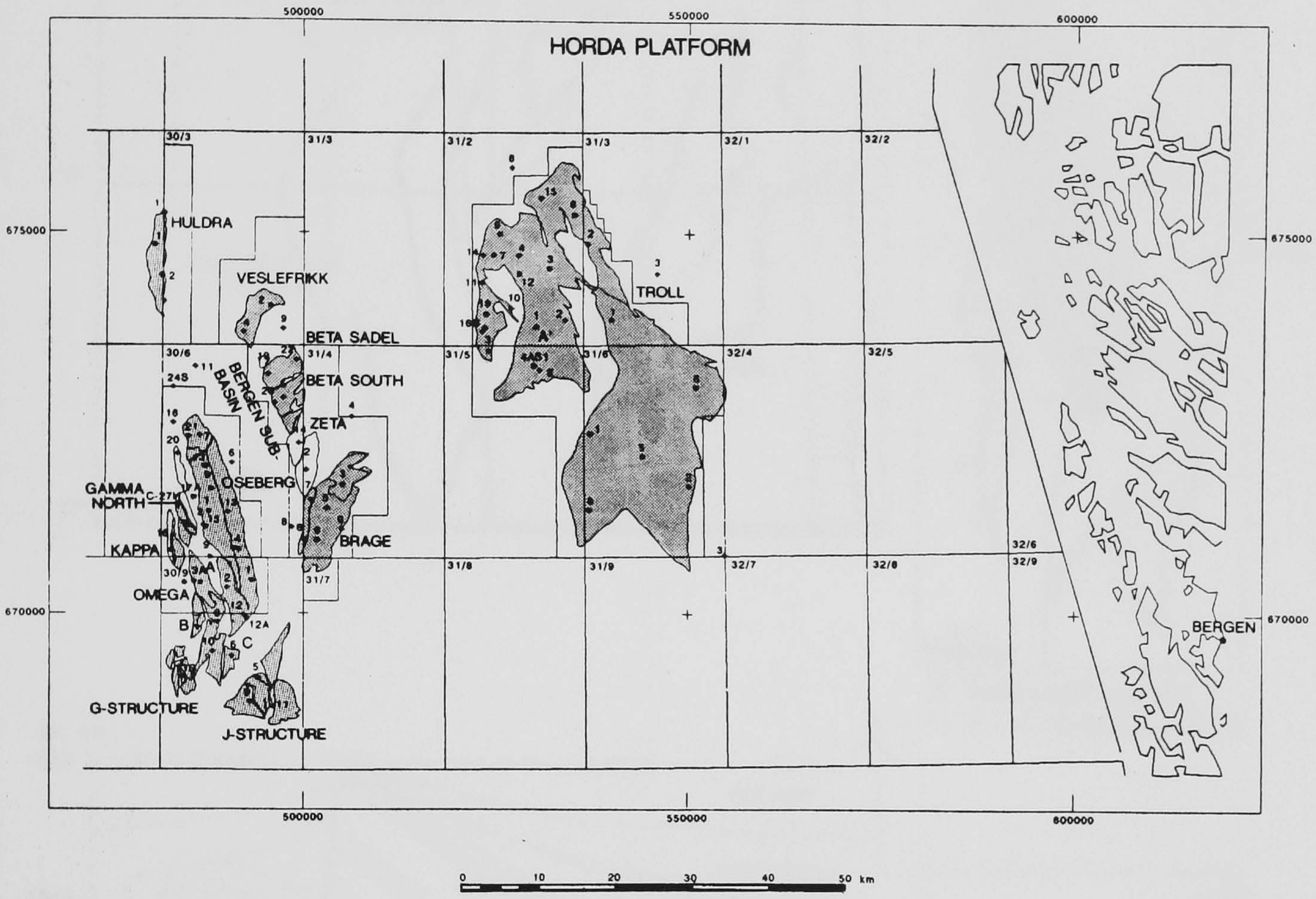


Figure 3.1: The Oseberg field is located in the Norwegian North Sea in blocks 30/6 and 30/9, approximately 140 km northwest of Bergen. Troll field that supplies Oseberg with gas for injection is approximately 50 km northeast. Shaded areas are hydrocarbon fields. From Johnstad et al. (1993).

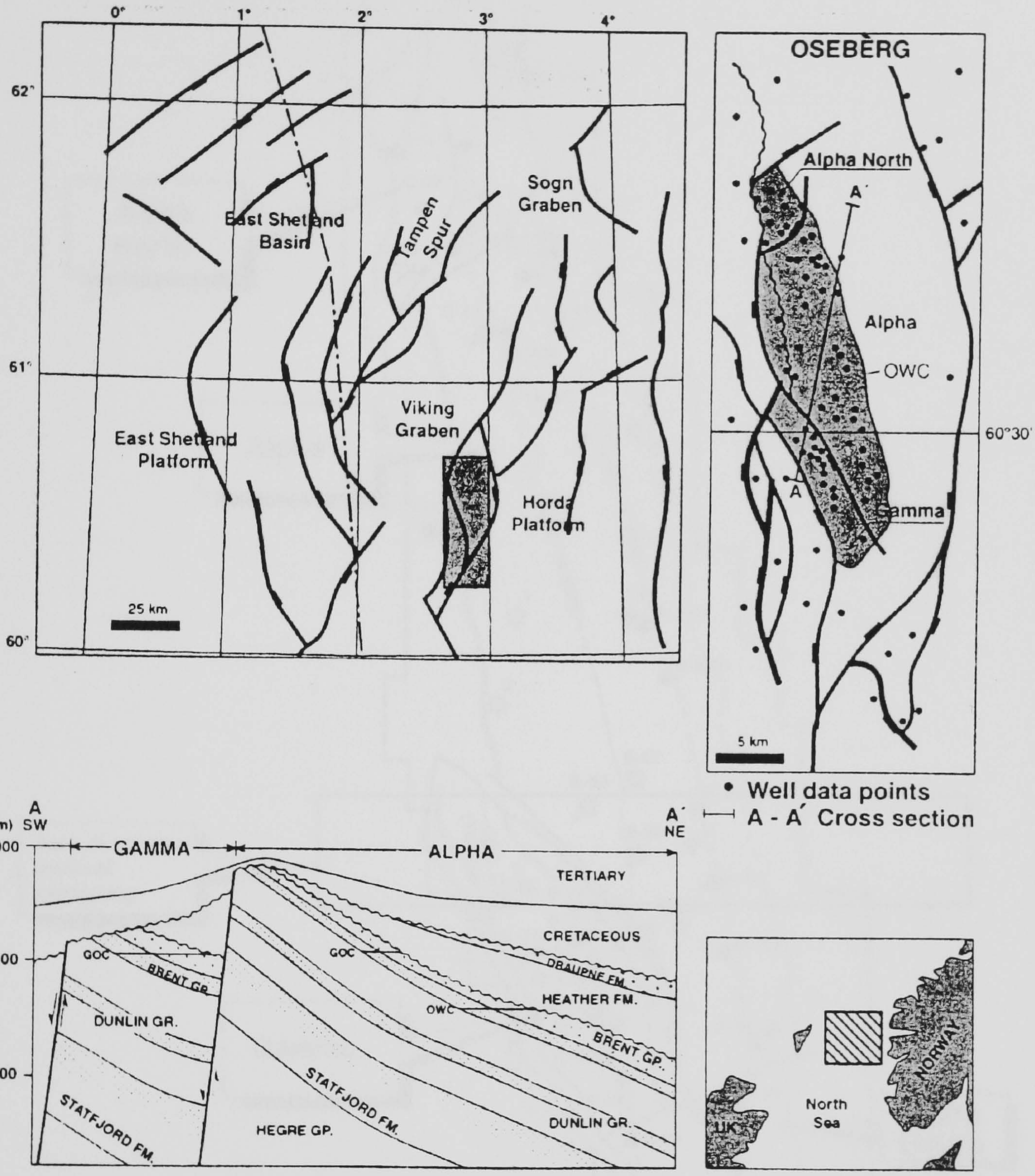


Figure 3.2: Location and geological maps and cross-section of the Oseberg field. The field is located offshore in the Norwegian North Sea at the western margin of the *Horda Platform*. The main reservoir interval (Brent Group) is truncated by the Base Cretaceous unconformity in the crustal parts of the fault blocks. From Ryseth et al. (1998)

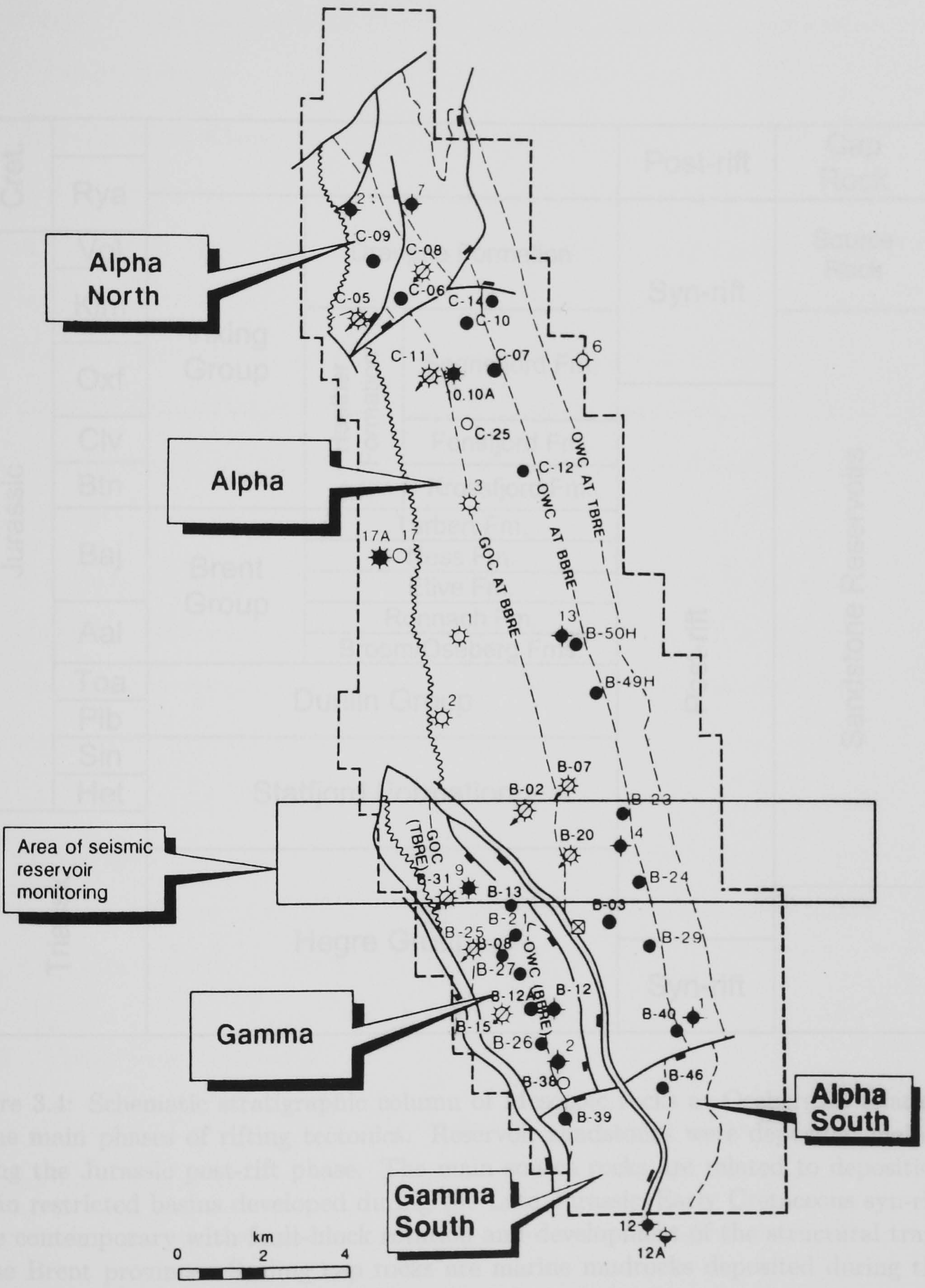


Figure 3.3: Oseberg field map, outlining the three major eastward tilted fault blocks (Alpha, Gamma, and Alpha North). Area of seismic reservoir monitoring surveys is indicated. From Johnstad et al. (1995)

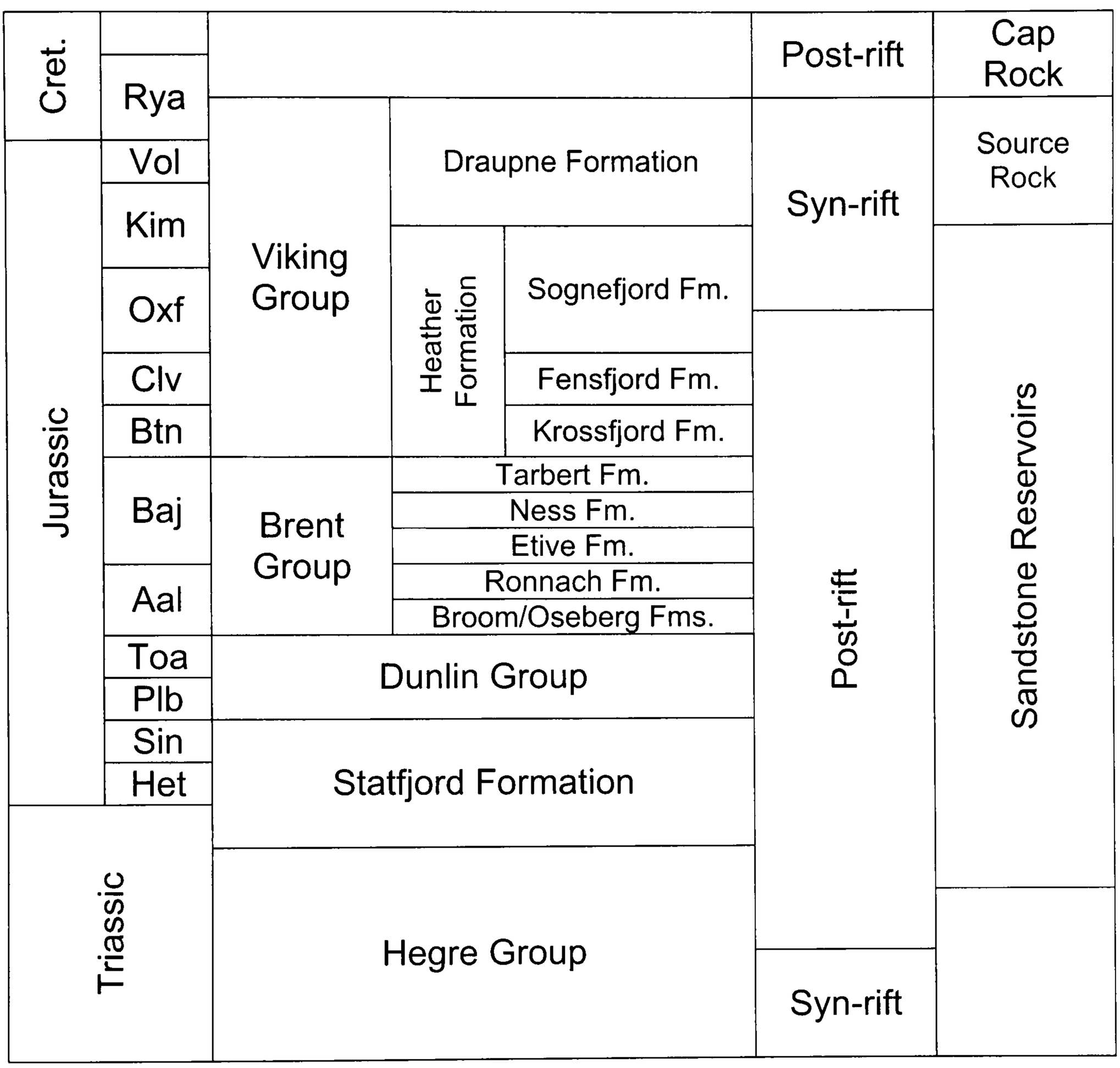


Figure 3.4: Schematic stratigraphic column of Mesozoic rocks at Oseberg in relation to the main phases of rifting tectonics. Reservoir sandstones were deposited mainly during the Jurassic post-rift phase. The main source rocks are related to deposition within restricted basins developed during the Late Jurassic–Early Cretaceous syn-rift stage contemporary with fault-block rotation and development of the structural traps of the Brent province. Sealing cap rocks are marine mudrocks deposited during the Cretaceous post-rift phase. Redrawn from Ryseth et al. (1998).

Group and Statfjord Formation were deposited. Rising sea level in the Early Jurassic led to the deposition of mudrocks and marginal marine sandstone of Dunlin Group.

An uplift of a volcanic dome south of the Viking Graben during the Toarcian (Early–Middle Jurassic) led to the deposition of the Brent Group as a major deltaic unit. Brent group comprises shallow-marine fan-delta sandstone (Broom and Oseberg formations), marine shoreface deposits (Rannoch and Etive formations), delta plain deposits (Ness formation), and prograding shoreline and coastal plain deposits (Tarbert formation). The Brent Group deposits indicates delta progradation and retreat caused by major regressive-transgressive cycles.

The Brent Group deposits were followed by the deposition of the marine mudrocks of Heather Formation. During the Late Jurassic, a rift phase caused the development of restricted basins and the deposition of highly organic mudrocks (Draupne Formation).

After the second phase of rifting and during post-rift, Cretaceous depositions were mostly of marine mudrocks. These marine conditions persisted throughout the Tertiary leading to the deposition of deep-marine and shallow-marine sandstone and mudrock facies.

3.1.2 Reservoir character

The Late Jurassic Draupene Formation represents the only significant source rock of hydrocarbons, Figure 3.4. From these source rocks, hydrocarbon migration into the Oseberg structure commenced during the second post-rift subsidence in the Late Cretaceous (Ryseth et al., 1998). Oil migrated into the Brent group, which represents the main reservoir interval. The seal to these tilted reservoir intervals is provide by the Heather formation and the Cretaceous limestone and shales.

Recoverable oil within Oseberg is mainly stored in the Oseberg Formation of the

Brent Group. It comprises excellent reservoir character with porosity of around 24% and permeability averaging 1 to 4 Darcys and ranging from 0.5 to more than 10 Darcys (Johnsen et al., 1995). However, a considerable amount of recoverable oil is stored within the Ness Formation (Ryseth et al., 1998). This is heterogeneous in nature, and is made of mainly elongated and laterally restricted sandstone bodies. Hence, although the Ness Formation accounts for 25% of total oil in place, it only accounts for around 10% of the reserves (Johnsen et al., 1995).

All structures within the Oseberg Oil Reservoir contain a gas cap, with a 380 m vertical gas column on Alpha and 150 m on Gamma. The depth of the original gas-oil contact was 2497 m corresponding to about 2340 ms TWTT, and the initial oil column was around 210 m (Johnstad and Ahmed, 1991; Johnstad et al., 1995). This oil column had been reduced to 20–40 m thickness after production of 75% of the total oil reserves by the year 2000 (Erlandsen, 2000).

3.1.3 Production history and future

The Oseberg field was discovered in 1979, and came onstream in December 1988 (Johnstad et al., 1993). The estimated reserves are 336 million Sm^3 of which 266 million Sm^3 had been produced by January 2000 (Erlandsen, 2000).

In 1993, Oseberg oil production was 470 thousands barrels per day (Johnstad et al., 1993), and by 1995 this had increased to 510 thousands barrels per day (Johnstad et al., 1995). Of the aforementioned oil production, 72% was from the field centre, and the rest was from the northern area (Johnstad et al., 1993, 1995), see Figure 3.5. This plateau oil production of 510 thousand barrels per day continued until the end of 1997. By May 2000 oil production had dropped to 290 thousand barrels per day (Erlandsen, 2000).

According to reservoir engineering studies, gas flooding can potentially recover a greater volume of oil than water flooding. Hence, pressure maintenance is implemented by re-injection of associated gas, and of gas from the nearby Troll field (Johnstad et

al., 1993).

In the near future, there are plans to start water injection and turn Oseberg into a gas producing field as illustrated in Figure 3.5 (Johnsen et al., 1995).

3.1.4 Time-lapse project at Oseberg

Between 1982 and 1999, several seismic surveys were acquired over the Oseberg Field as part of a time-lapse project. This aimed to analyse fluid movement to enable effective reservoir management. In particular, seismic monitoring focused on pressure recording, gas-front observation and production logging.

The base survey (NH8906) was acquired from December 1989 to January 1990. A monitor survey was collected using the same vessel in early 1991. In April to July 1992 the field was re-shot using one source and five cables to acquire the monitor survey NH9201. The aim was to derive a dataset compatible with 1989 and 1991 surveys (Johnstad et al., 1995). The three surveys (1989, 1991 and 1992) were discussed by Johnstad et al. (1995) who extended on previous work by Johnstad et al. (1993) on the first two sets (1989 and 1991).

3.2 Time-lapse datasets

This study is conducted on a time-lapse swath from the base survey (NH8906) and the monitor survey (NH9201). Therefore, this section will analyse the two aforementioned datasets and summarize previous studies conducted on them.

3.2.1 Analysis of the two datasets

Survey parameters

The base survey NH8906 was acquired with two streamers 100 m apart, while the monitor survey NH9201 was acquired with five streamers 75 m apart. However in both cases each streamer towed 240 receivers, the shot to shot spacing was 25 m and

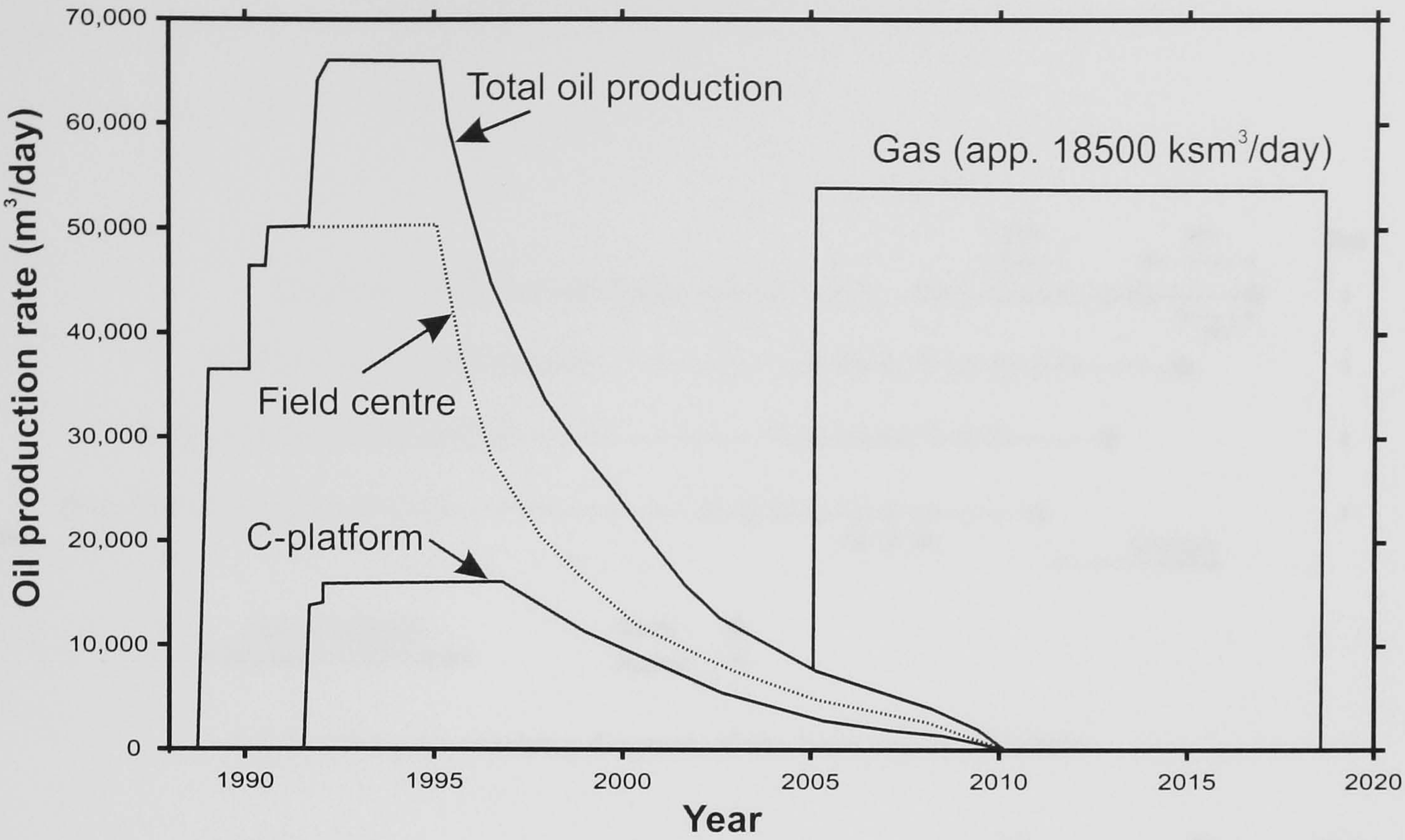
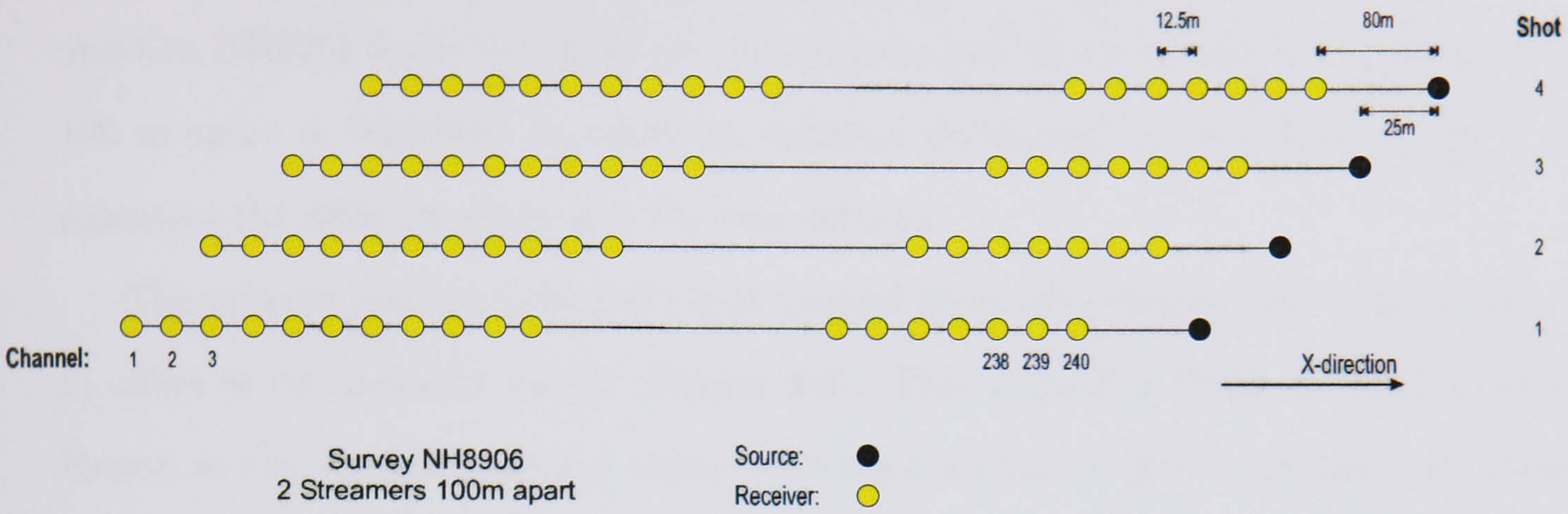


Figure 3.5: Oseberg oil/gas field planned long term production rates. Note the move from oil to gas production expected to start in 2005. C-platform is located in the northen part of Oseberg and account for around 28% of production. Redrawn from Johnsen et al. (1995)

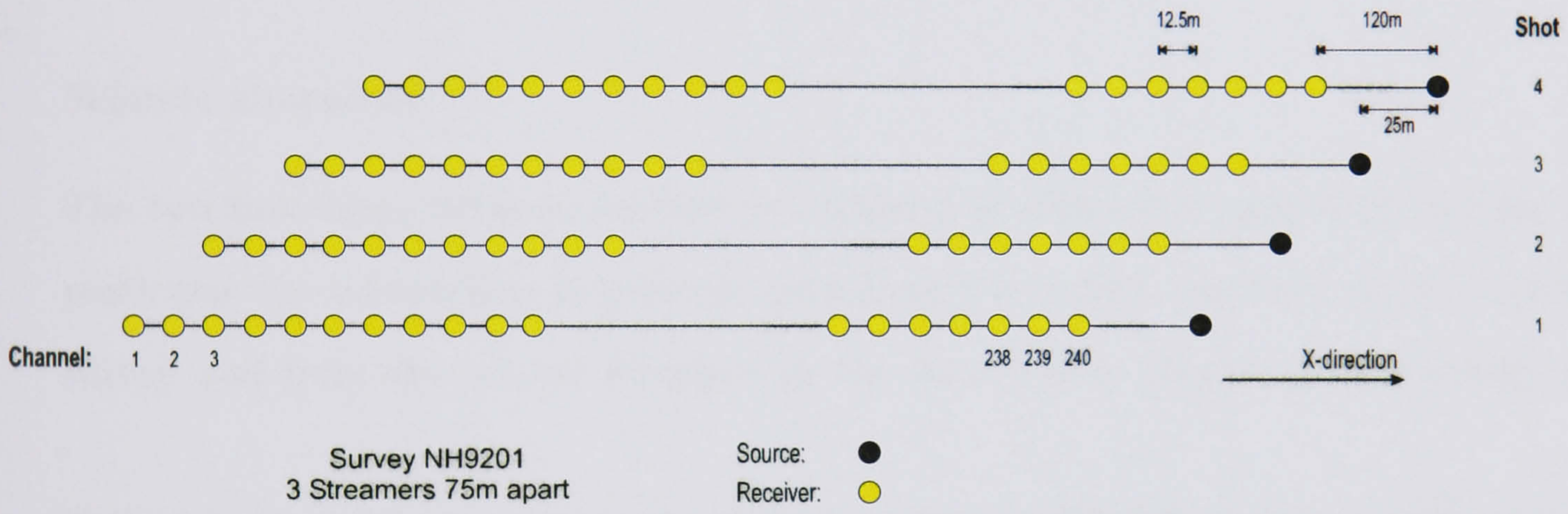
the receiver spacing was 12.5m. Table 3.1 and Figure 3.6 summarize the basic survey parameters of both NH8906 and NH9201.

One source was used within each survey. The source was located between the two streamers in the base survey and between the three streamers towed by Geo Explorer (Table 3.1) in the monitor survey.

In this study, the receivers (channels) in each streamer are numbered 1 to 240 sequentially from the furthest offset to the nearest offset. In addition, only the three streamers towed by Geo Explorer in the monitor survey are considered. This is because those three were lain at similar position to the base survey, whereas the other two towed by Malene Ostervold were lain further away. Furthermore, in this study, the seismic data acquired by one streamer in one survey swath is called an *in-line*. This is because, in this study, there is one source within each survey, which means that there is one CMP seismic profiles acquired by each streamer in each of the two time-lapse swaths,



(a) A schematic stacking diagram of the base survey, NH8906



(b) A schematic stacking diagram of the monitor survey, NH9201

Figure 3.6: A schematic diagram of the shooting geometry of the two time-lapse surveys, NH8906 and NH9201. The diagram shows one streamer from each. All streamers within one survey have the same geometry. In both surveys, each streamer tows 240 receivers, the shot to shot spacing is 25 m, and the receiver spacing is 12.5m. However, the streamer separation is 100 m in the base survey and 75 m in the monitor survey. In addition, the nearest channel offset is around 80 m in the base survey and 120 m in the monitor survey (*cf.* Figure 3.8). In this study, the receivers (channels) in each streamer are numbered 1 to 240 sequentially from the furthest offset to the nearest offset. In addition, in each survey there are approximately 465 shots, which were acquired along the same x-profile within the two time-lapse surveys.

and this profile is parallel to the in-line direction of the 3D survey.

Although the basic survey parameters of NH8906 and NH9201 as outlined by Table 3.1 are similar, nevertheless they are not identical. The use of five streamers 75 m apart in NH9201 leads to a different survey geometry than that given by two streamers 100 m apart in NH8906. In addition, different feathering, as indicated in Figure 3.7, enhances the difference between the two surveys.

The nearest receiver (channel) is at around 80 m offset in the base survey and 120 m offset in the monitor survey (Figure 3.6). This is further illustrated in Figure 3.8. Hence, at the same shooting position, and when trying to find the nearest comparable trace to the monitor survey from the base survey, one needs to select a channel three to four channel positions further along the streamer.

Seismic character

The two time-lapse datasets are further analysed at different stages of processing. In particular the comparison is between data from the middle streamer in the monitor survey and from the second streamer in the base survey (higher in the y-axis, see

Table 3.1: Basic survey parameters of the base survey NH8906 and the monitor survey NH9201. After Johnstad et al. (1995)

	NH8906	NH9201
Survey Date	Dec 1989 to Jan 1990	Apr to Jul 1992
Vessel	Polar Princess	Geo Explorer (3 streamers) & Malene Ostervold (2 str.)
Source	2400 cu. in.	3400 cu. in.
Source Depth	5 m	6 m
No. Streamers	2	5 (3+2)
Streamer Depth	5-6 m	6 m
Streamer Separation	100 m	75 m
Receivers per Streamer	240	240
fold	60	60
Shot spacing	25 m	25 m
Receiver spacing	12.5 m	12.5 m
Record Length	6 sec	6 sec
Sampling Rate	2 ms resampled to 4 ms	2 ms resampled to 4 ms
Record Filter	180hz high, 8hz low	218hz high, low out

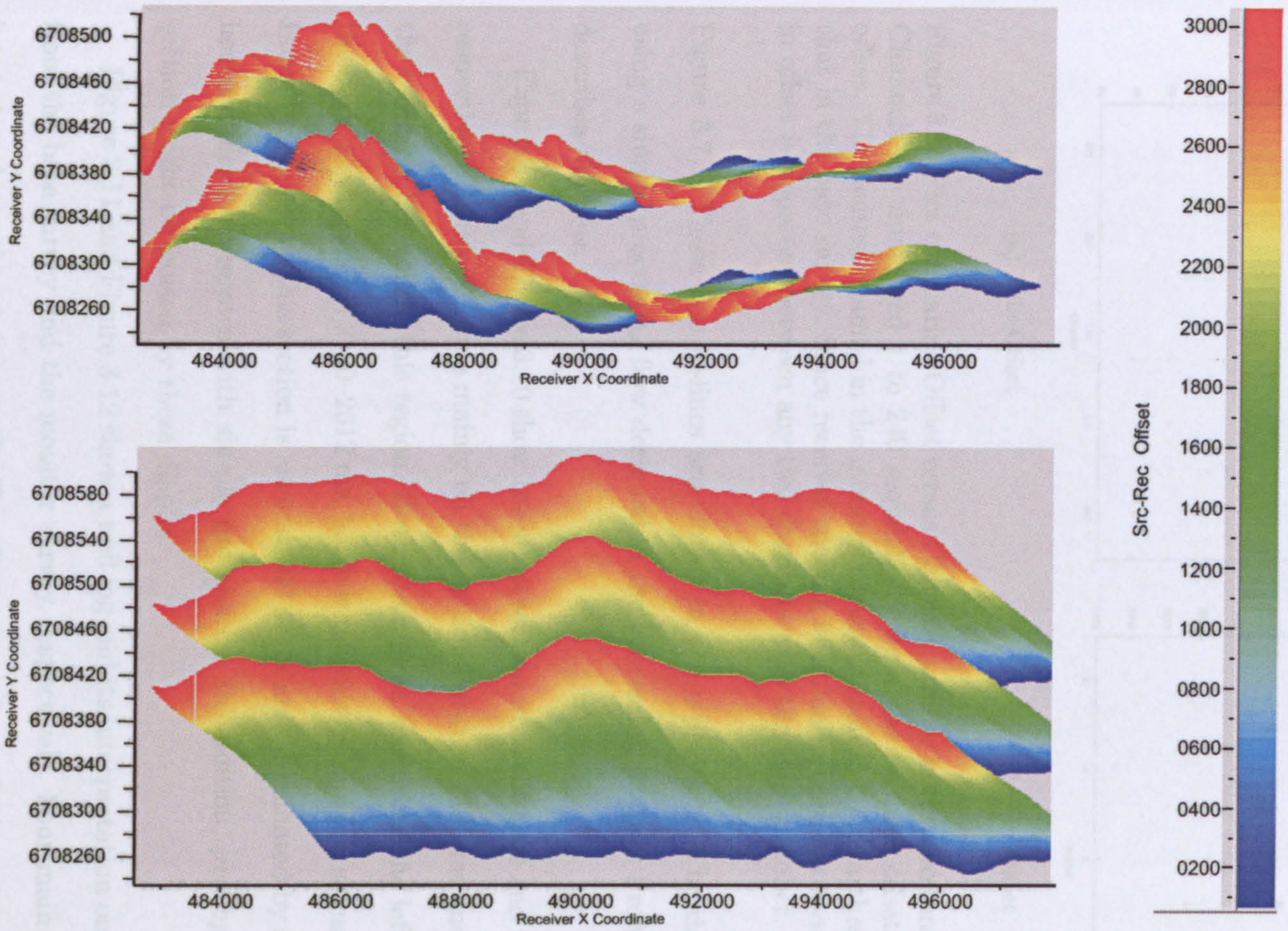
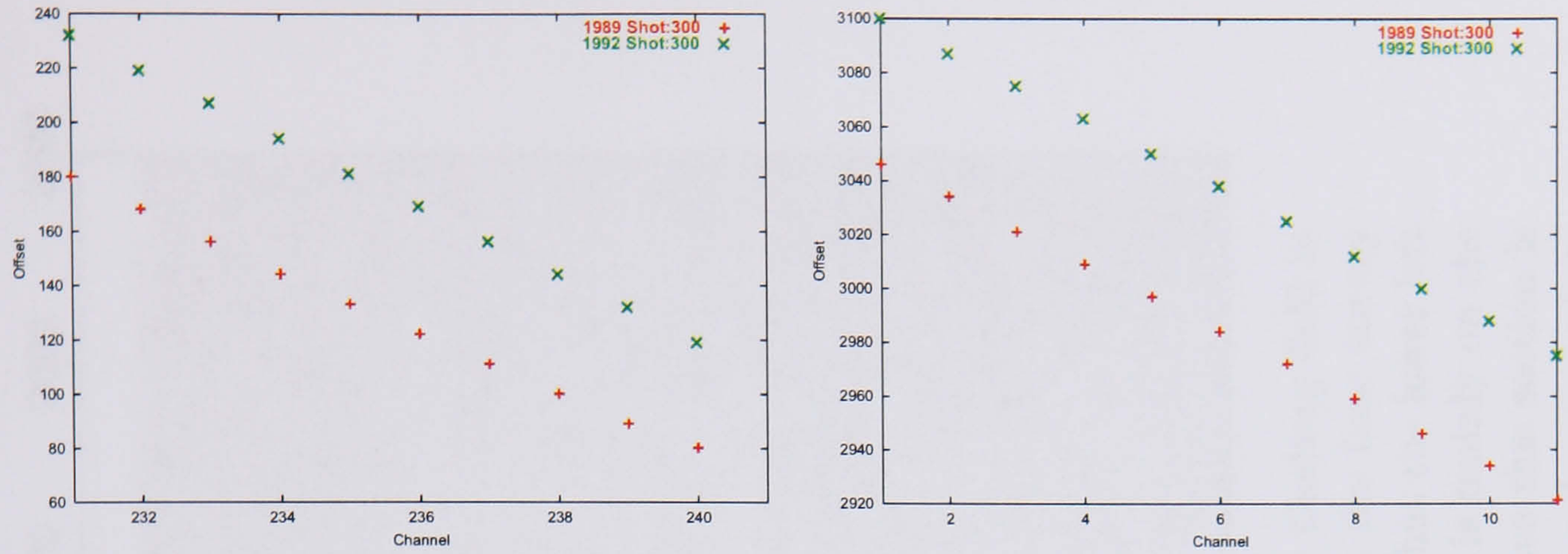


Figure 3.7: Receiver position of the base and the monitor survey. Receiver x-position is plotted against receiver y-position with colour indicating offset. Top: NH8906 base survey line with two streamers 100 m apart. Bottom: NH9201 survey lines with three streamers 75 m apart (there are two additional streamers towed by another vessel). There is clear difference in feathering between the two surveys. Note that the y-axis is exaggerated.



(a) Near Offset

(b) Far Offset

Figure 3.8: Plot comparing Offset versus Channel number in the two time-lapse surveys. Channels are numbered 1 to 240 sequentially from the furthest offset to the nearest offset. The nearest channel in the monitor survey is around 40 m farthest in offset from that in the base survey. Since receiver spacing is 12.5m in both surveys, this difference in offset is apparent between any two channels with the same number.

Figure 3.7). Those two in-lines are closest in the y-position. The data are stacked using a simple processing flow designed in the processing package ProMAX. Table 3.2 describes the flow.

Figure 3.9 and Figure 3.10 show the stack section from the base and monitor survey, respectively. Our interest is mainly on the section between 2100 ms and 2612 ms since the reservoir lies within this region. Both stack plots show that the left (west) part of the data, at the section 2100–2612 ms, has low reflectivity and low signal-to-noise ratio. Immediately above this section is around 800 ms of area dominated by salt diapirs and hence difficult to resolve with simple processing. In addition, *pull-ups* on the lower section might be caused by these diapirs.

Figure 3.11 and Figure 3.12 show a well log and its interpretation on a stack section from the base survey and the monitor survey, respectively. Four main horizons are of interest and are thus interpreted: *Top Cretaceous*, *Base Cretaceous*, *Top Brent*, and *Base Brent*. These are clearly marked on Figure 3.11(a) and Figure 3.12(a). The reservoir lies within the Brent group which truncates at the unconformity of the Base Cretaceous.

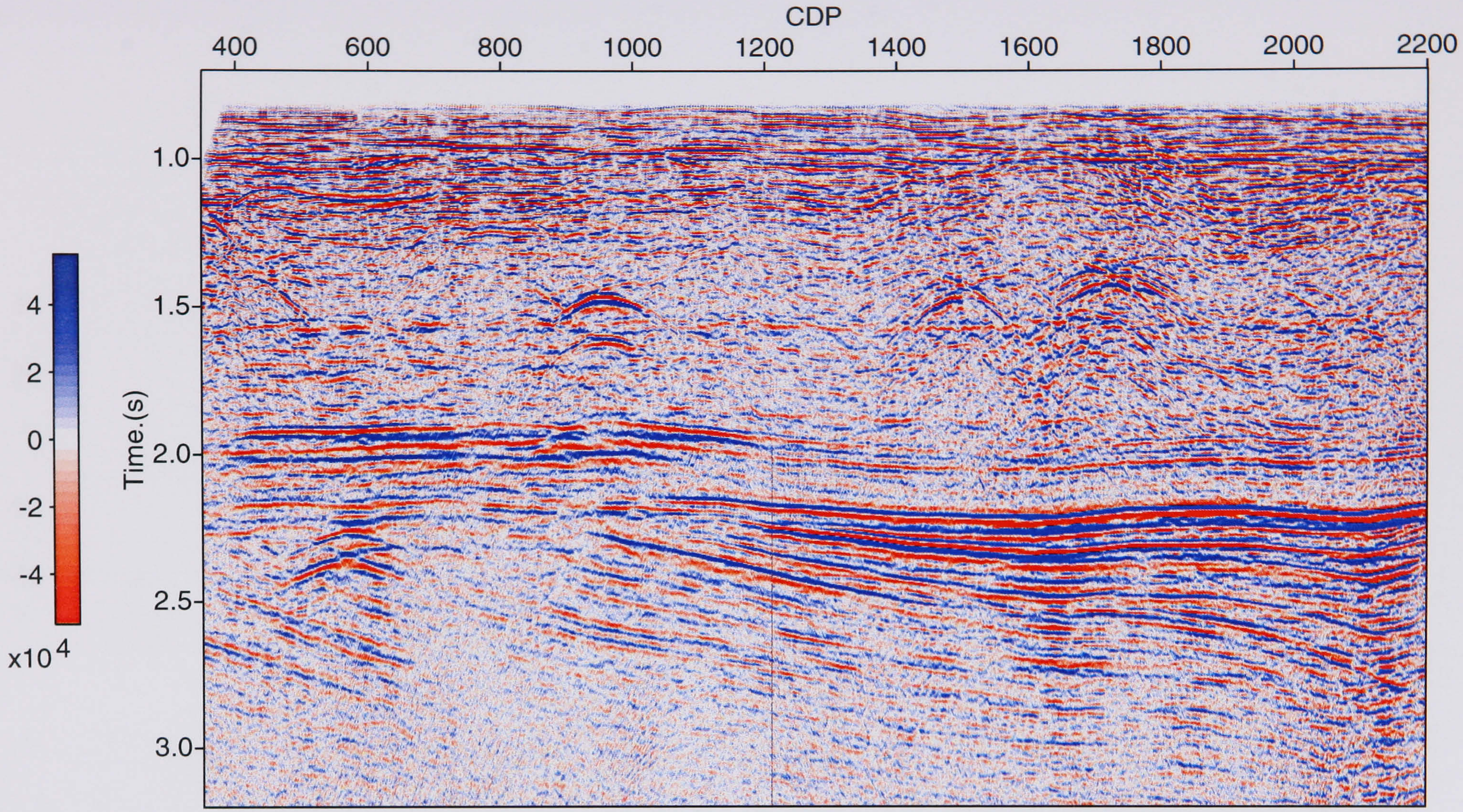


Figure 3.9: Two time-lapse seismic lines, NH8906 and NH9201, were acquired over the Oseberg field in 1989 and 1992 respectively. The figure shows a seismic stack plot of one in-line from the base survey NH8906. A similar plot of the monitor survey is shown in Figure 3.10. The plot shows that the lower left (west) part of the data has low reflectivity and low signal-to-noise ratio. Our interest is mainly on the time section 2100–2612 ms since the reservoir lies within this region. Immediately above this section is around 800 ms of area dominated with salt diapirs and hence difficult to resolve with simple processing.

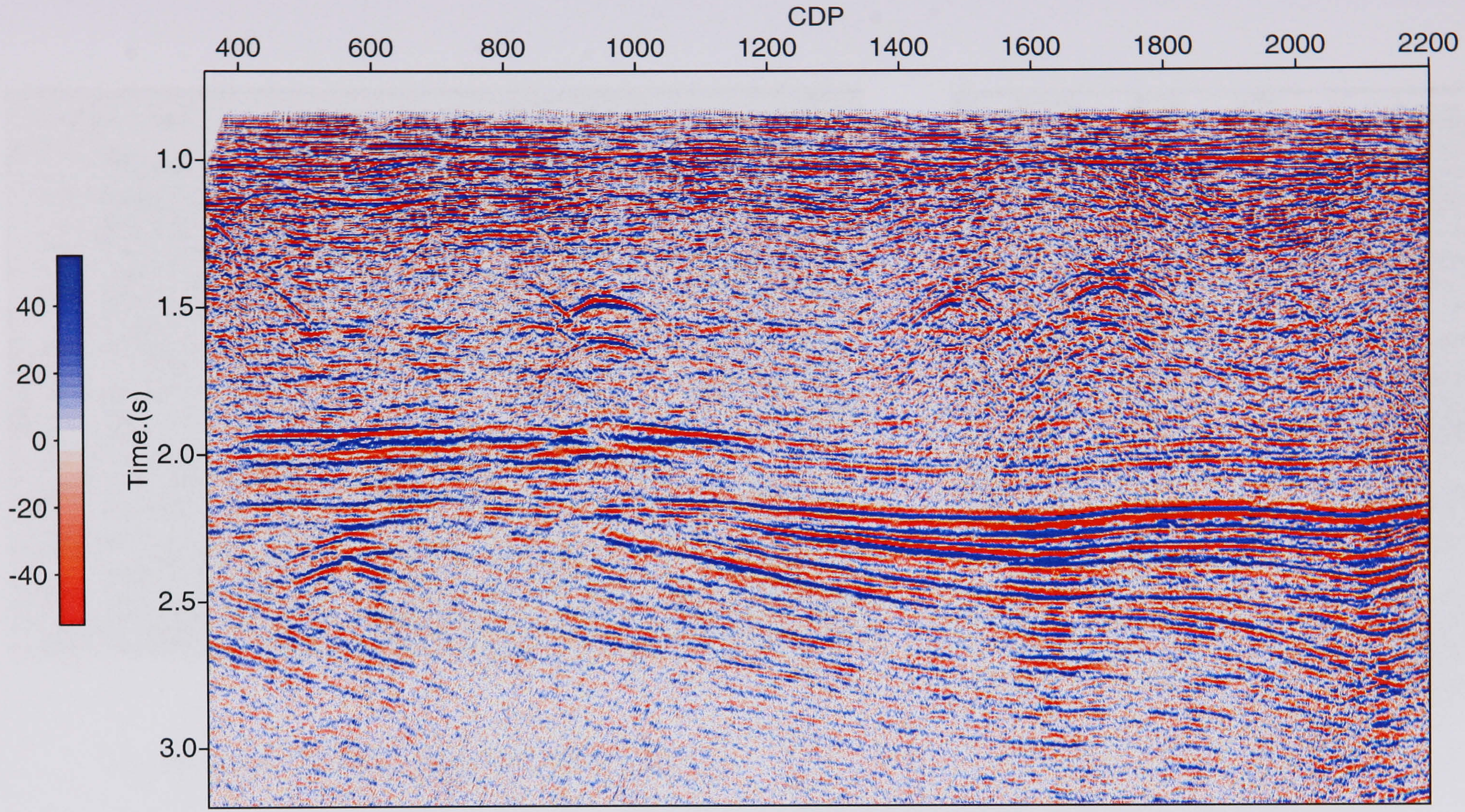
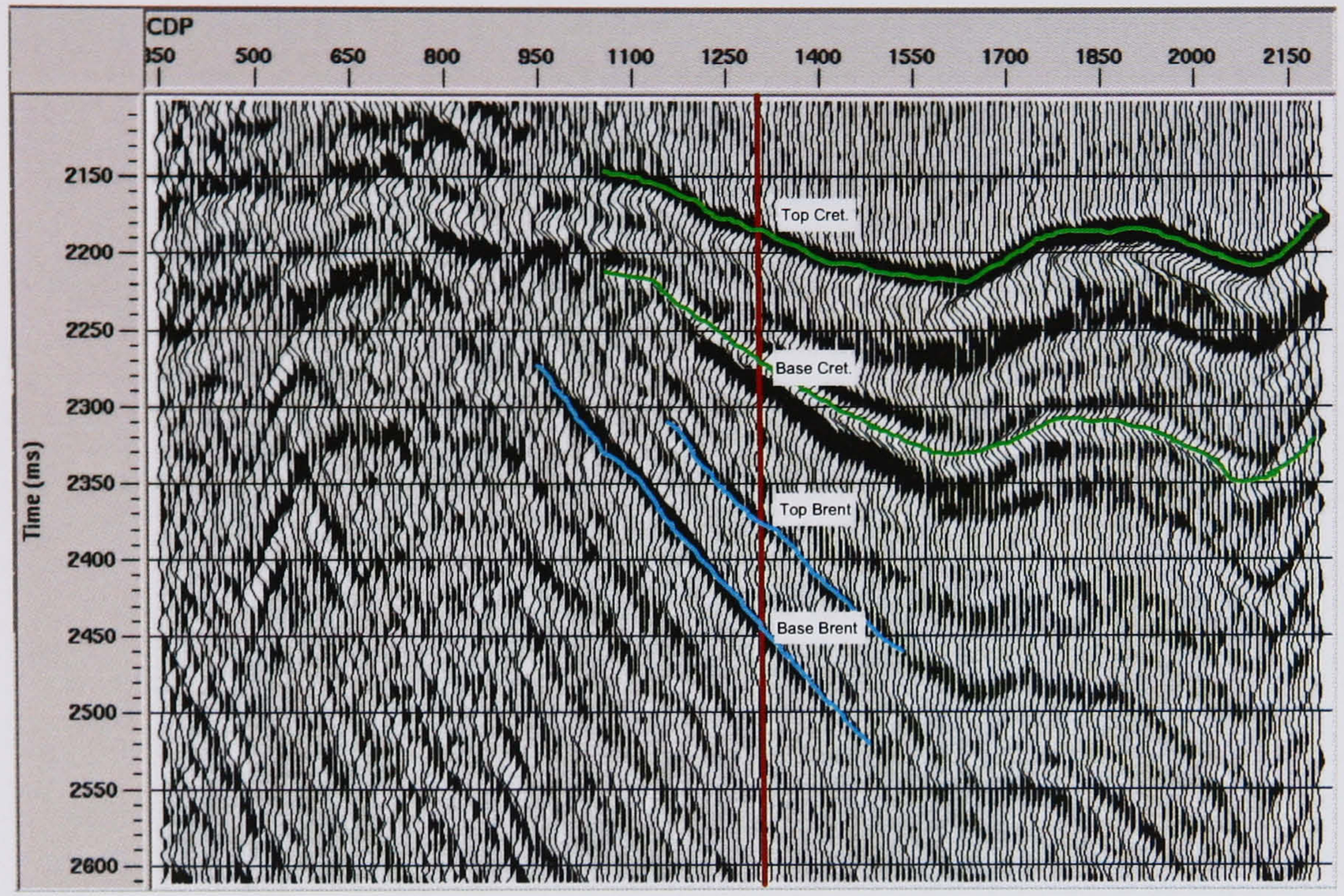
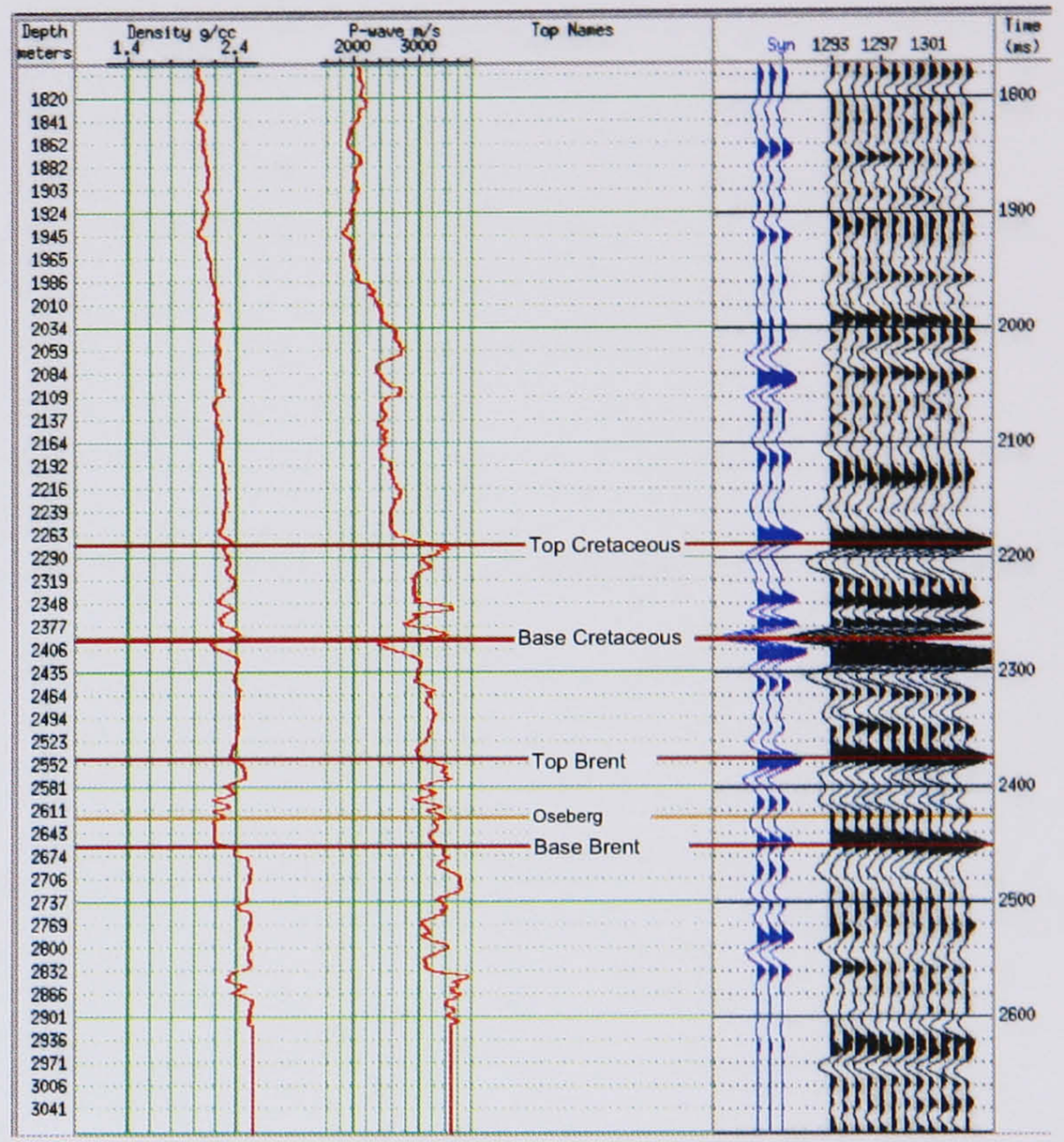


Figure 3.10: Two time-lapse seismic lines, NH8906 and NH9201, were acquired over the Oseberg field in 1989 and 1992 respectively. The figure shows a seismic stack plot of one in-line from the monitor survey NH9201. A similar plot of the base survey is shown in Figure 3.9. Note that this plot is scaled by a factor of 1000 relative to Figure 3.9 due to difference in amplitude scale between the two datasets. As in Figure 3.9, note the low reflectivity of the lower left part and the salt diapirs dominating a significant part of the top half.

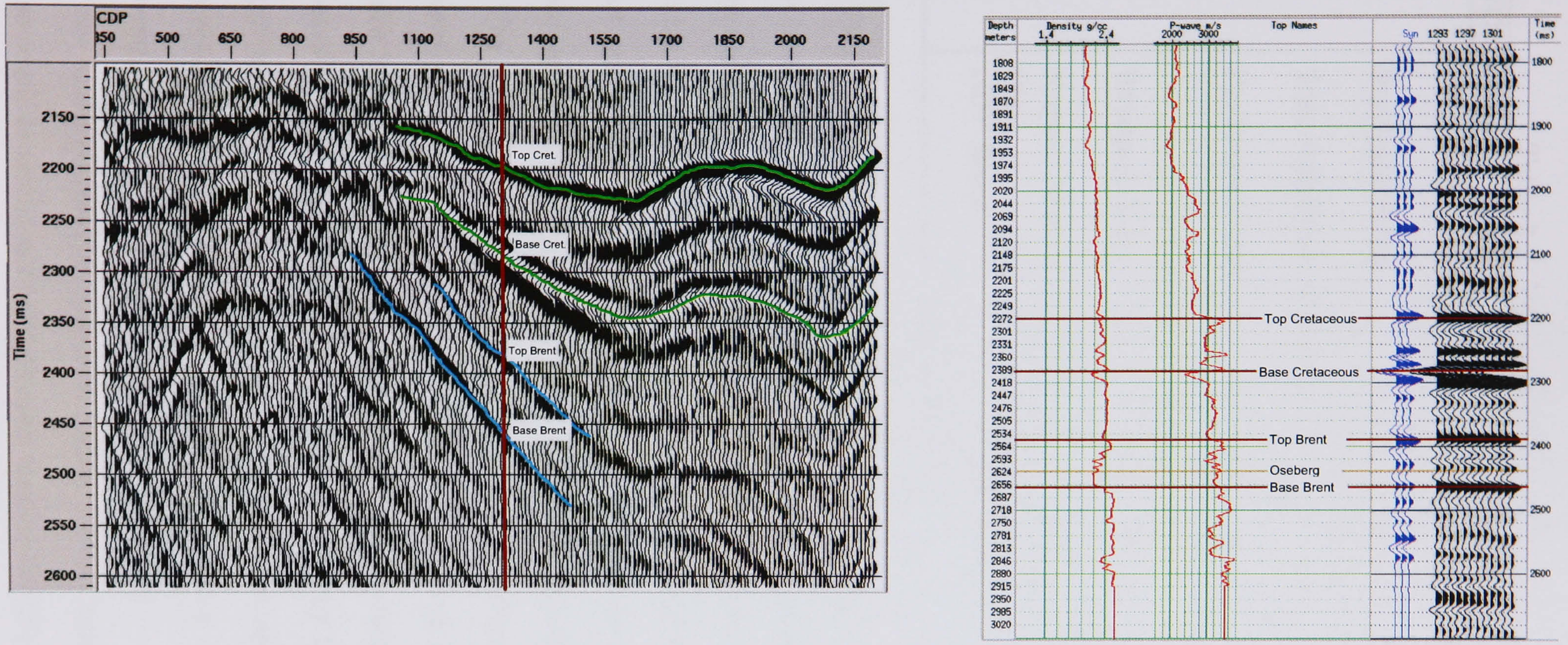


(a) Base survey stack section between 2100 ms and 2612 ms



(b) Base survey logs from well B-30-6-4

Figure 3.11: (a) shows a zoom of the stack section of the base survey shown in Figure 3.9. Only every 10^{th} trace is plotted. (b) shows the logs from well B-30-6-4. This well lies at CDP 1293 and its position is marked on the stack plot in (a). The logs are interpreted for the major horizons of interest: Top Cretaceous, Base Cretaceous, Top Brent, and Base Brent. This interpretation is correlated with the seismic, and in turn those horizons are marked on the stack section. The reservoir lies within the Brent group which truncates at the unconformity of the base Cretaceous.



(a) Monitor survey stack section between 2100 ms and 2612 ms

(b) Monitor survey logs from well B-30-6-4

Figure 3.12: (a) shows a zoom of the stack section of the monitor survey shown in Figure 3.10. Only every 10^{th} trace is plotted. (b) shows the logs from well B-30-6-4. This well lies at CDP 1293 and its position is marked on the stack plot in (a). The logs are interpreted for the major horizons of interest: Top Cretaceous, Base Cretaceous, Top Brent, and Base Brent. This interpretation is correlated with the seismic, and in turn those horizons are marked on the stack section. The reservoir lies within the Brent group which truncates at the unconformity of the base Cretaceous.

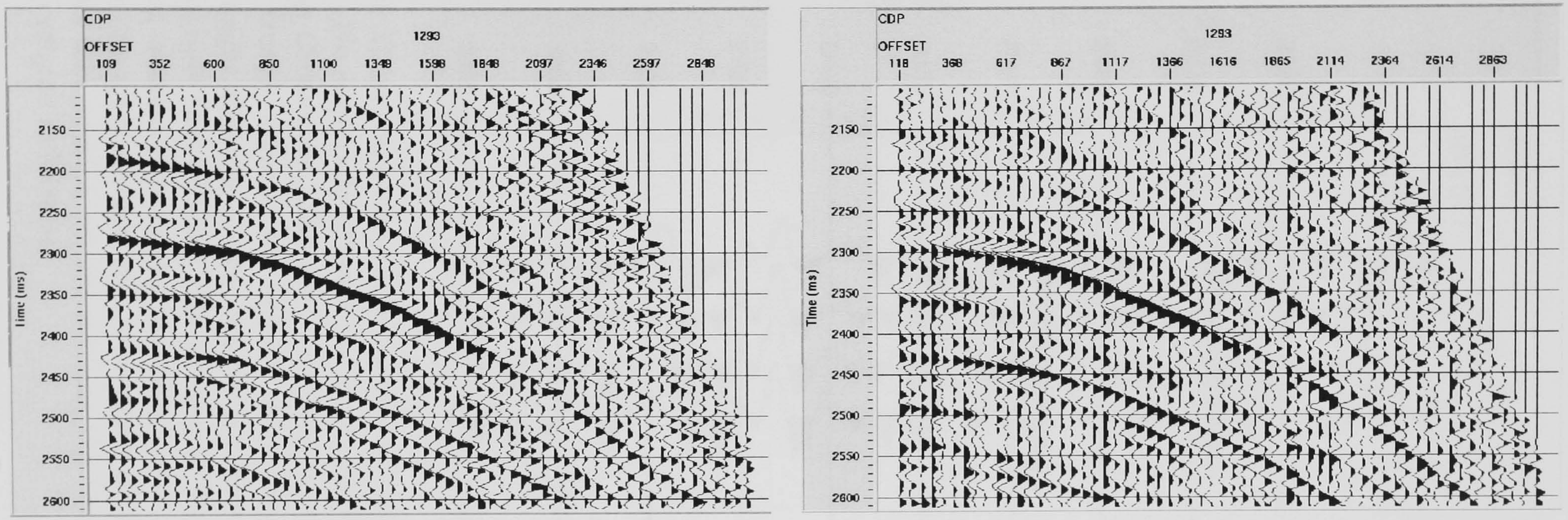
Table 3.2: Processing flow used for stacking the datasets.

NO.	Process Name	Description
1	Top Mute	mutes the top part of traces to remove direct arrivals.
2	Bandpass Filter	using Ormsby bandpass, passing 8Hz–90Hz and tapering both sides.
3	True Amplitude Recovery	applies a time dependent amplitude recovery with time^2 correction.
4	Predictive Decon	applies a minimum phase predictive deconvolution with 250 ms operator and 70 ms prediction lag.
5	Bandpass Filter	using Ormsby bandpass, passing 8Hz–90Hz and tapering both sides.
6	Normal Moveout Correction	corrects for normal moveout using stacking velocities supplied by *Total*.
7	CDP Stack	sums all traces within one CDP to produce a stack.

The two time-lapse datasets shown on the previous stacked sections might look near identical. However, at close comparison, differences do appear. Figure 3.13 shows a comparison between one particular CDP gather from the two datasets. The figure shows a few differences. A closer analysis is shown in Figure 3.14, where only four near offset traces are compared. In this latter figure, we can see clear indication of time, phase, and amplitude misfit. In addition, slight spectral difference might exist.

The differences between the traces shown in Figure 3.14 are further analysed by estimating the three misfit parameters as described in Section 1.2.2, page 44. From Figure 3.14, each trace in the monitor survey is matched to its closest equivalent (in terms of offset) in the base survey. This is done by first estimating time, phase, and amplitude misfit between the two traces and then applying the necessary correction. The results are summarized in Figure 3.15, where a comparison is given between the cross-equalized traces and those of the base survey.

The misfit parameters shown in Figure 3.15(b) indicate, as expected, both time and phase misfit in addition to the obvious amplitude difference. The linear cross-equalization, applied through the three scalar corrections, improved the matching significantly. In addition, it allowed a possible production related difference to be noted (Figure 3.15).



(a) Base Survey

(b) Monitor Survey

Figure 3.13: Comparison between the two time-lapse datasets at CDP 1293. A top mute and an amplitude recovery are applied. Traces are scaled individually. At the Top Cretaceous (2200 ms at near offset), the base survey shows a strong peak followed by a stronger trough, whereas the monitor shows a weak peak and an even weaker trough. Between 2300 ms and 2350 ms at near offset (and lower down along the same horizon on far offsets), we note a long period trough on the base survey, while only a shorter trough is seen on the monitor survey. Closer comparison is given in Figure 3.14.

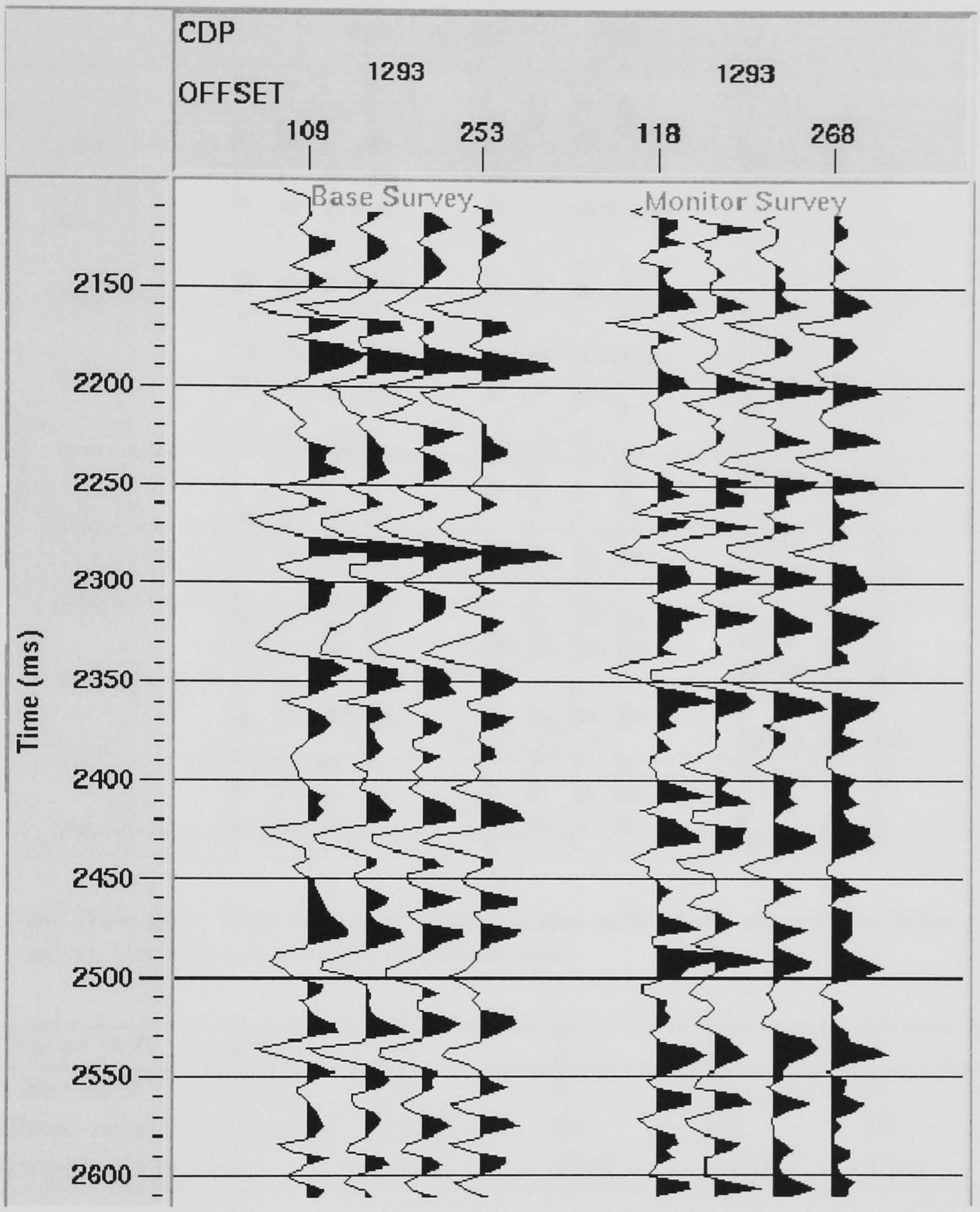
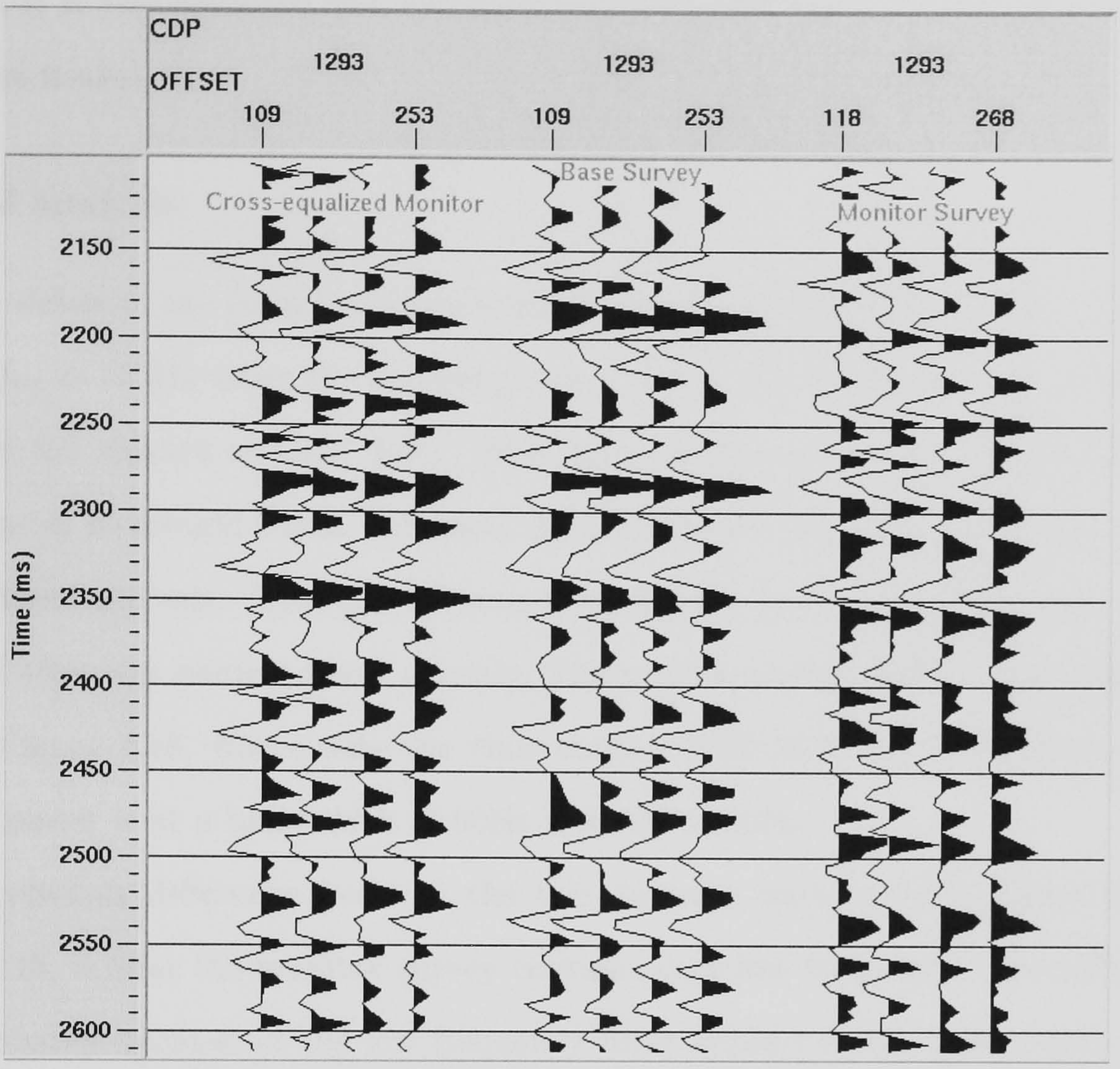


Figure 3.14: Close comparison between the two time-lapse datasets at the near offset of CDP 1293. No amplitude recovery has been applied. Each trace is scaled individually. At the Top Cretaceous (2200 ms), the base survey shows a strong peak followed by a stronger trough, whereas the monitor shows weak peak and even weaker trough. A time shift might align the events, but the period (or frequency) might also be different. In addition, note that between 2300 ms and 2350 ms, a long period trough exist in the base survey, whereas it is shorter on the monitor survey. Here, a phase shift might be required to correct the event. At around 2400 ms a more complicated difference exists. However, at this position we are entering the Brent group and hence there might be production related differences.



(a) Trace plot. Note that each trace is scaled individually since the monitor survey traces have a lot lower amplitude scale.

Traces (left to right)	1	2	3	4
Time shift (samples)	2	2	2	2
Phase rotation (degrees)	73	61	60	59
Amplitude scaling	901	1012	1288	1204

(b) Parameters used to match each monitor survey trace to its equivalent in the base survey

Figure 3.15: (a) shows a close comparison of the two time-lapse datasets at the near offset of CDP 1293. In addition to the same traces shown in Figure 3.14, linear cross-equalized traces of the monitor are also shown. The cross-equalized traces are calculated by applying a time shift, phase rotation, and amplitude scaling to each trace in the monitor to best match it to its nearest in the base. The three parameters applied are noted in (b). The cross-equalization applied improved the match significantly pointing to the fact that there are clear misfits in terms of time, phase, and amplitude. However, it is also apparent that the three misfit parameters are not sufficient to fully match the traces. Note also that the cross-equalization matched the traces significantly less between time 2360 ms and 2420 ms. This could be because there is a production related change, since the reservoir (Top Brent) starts at around this time (*cf.* Figure 3.11 and Figure 3.12).

However, it is also apparent that the three misfit parameters are not sufficient to fully match the traces.

Spectral analysis

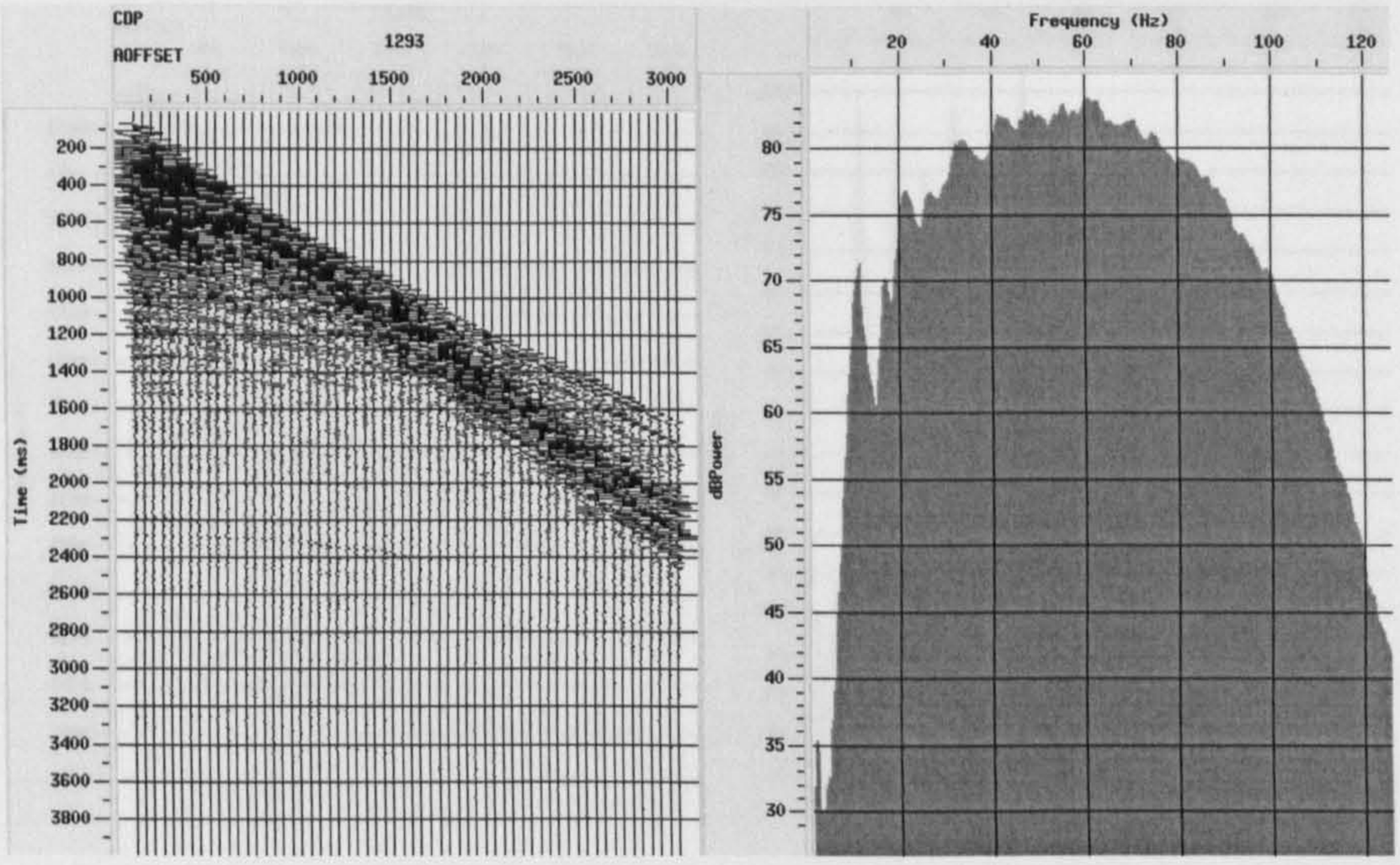
The two datasets are further analysed for their frequency spectra. The Nyquist frequency f_{nq} is 125Hz since the sampling rate $\triangle t$ is 4 ms and $f_{nq} = \frac{1}{2\triangle t}$. Figure 3.16 compares the spectra at CDP 1293. Both have similar spectra and bandwidth. The spectra peak at around 60Hz. However, the peak frequency is shifted to the high end due to direct arrivals. This can be seen from Figure 3.17, where the same data are analysed after top muting direct arrivals. The peak is around 40Hz. This peak is also seen in Figure 3.18, where only the time section 2100–2612 ms is considered. Most spectral power is at a bandwidth of 60Hz from 20 to 80Hz.

One obvious difference between the two datasets seen in both Figure 3.17 and Figure 3.18, is that the monitor survey contains very low frequency. The difference is most probably because of the low frequency filter applied to the base survey dataset during acquisition (see Table 3.1).

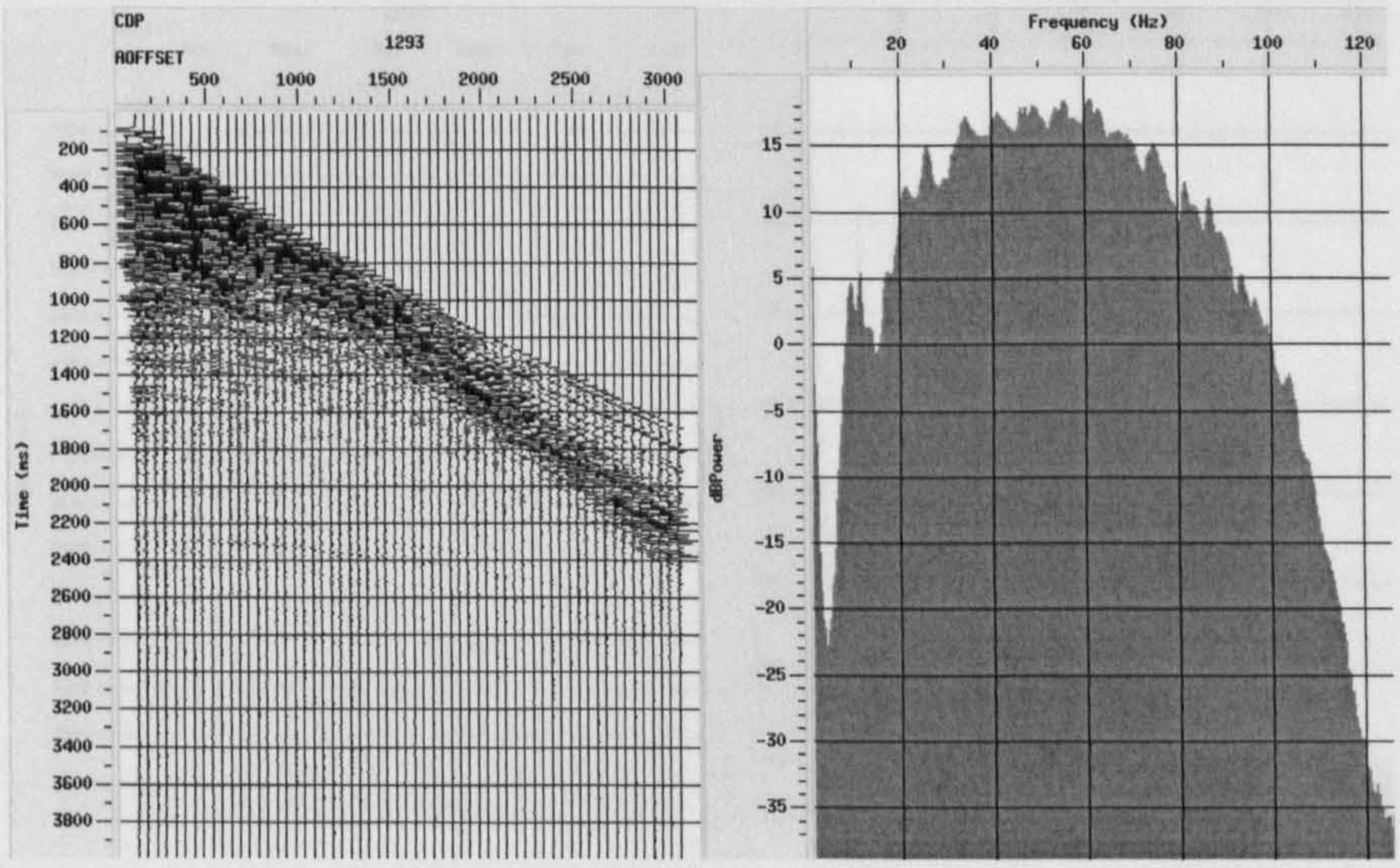
3.2.2 Summary of previous work on the datasets

Previous analysis and comparison of the two time-lapse surveys NH8906 and NH9201 is reported in Johnstad et al. (1995). Here, I will summarize their matching criteria and interpretation.

Their work was on stacked volumes of the two datasets that have undergone slightly different processing. Their spectral analysis of the two datasets indicate very similar spectrum, but more ringing was observed on the monitor survey. They also report minor phase difference on the stacks. Johnstad et al. (1995) concluded that the two datasets are similar in phase. In addition, they explained that although two different deconvolution functions were used on the two datasets, these did not change this similarity. This is contrary to my finding (on pre-stack data), where there is a clear

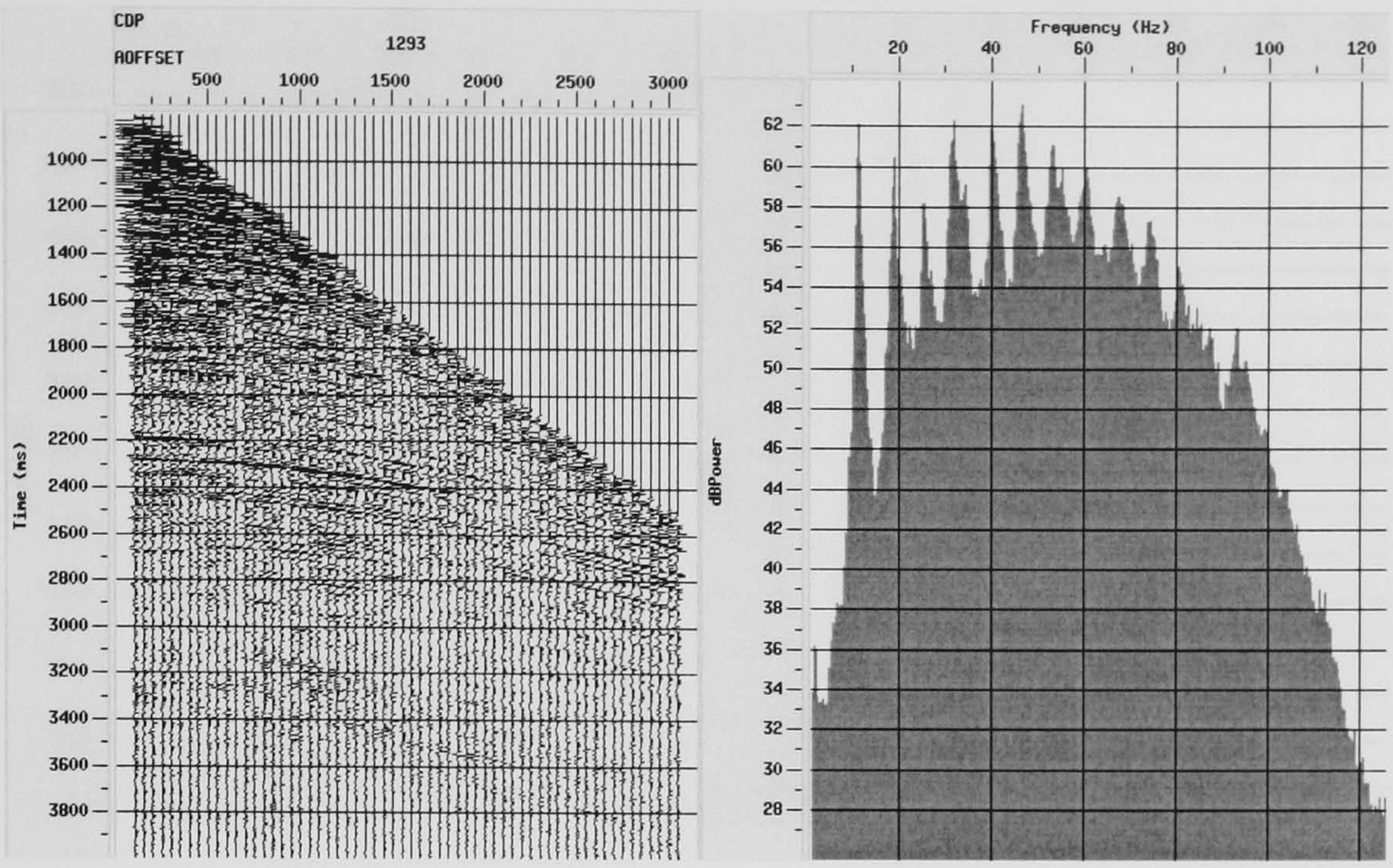


(a) Base Survey

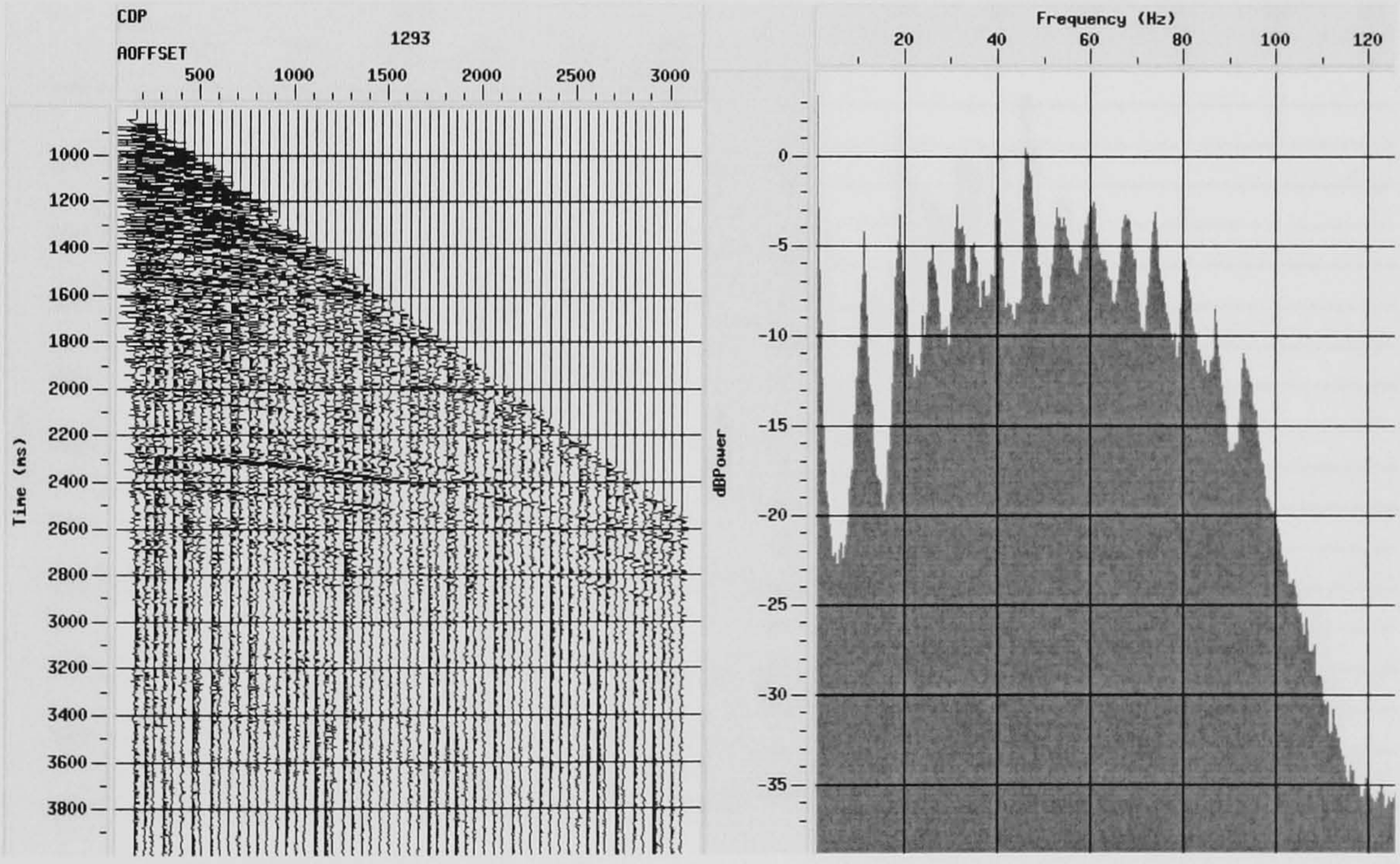


(b) Monitor Survey

Figure 3.16: Spectra of the two time-lapse datasets at CDP 1293. Each plot shows the spectrum on the right and the seismic section from which the spectrum is calculated on the left. The spectrum is plotted in dB log scale with the reference power being 1. Both spectra are similar and peak around 60Hz. However, direct arrivals have shifted the spectrum to the high end as can be seen when these are muted as in Figure 3.17 and Figure 3.18.

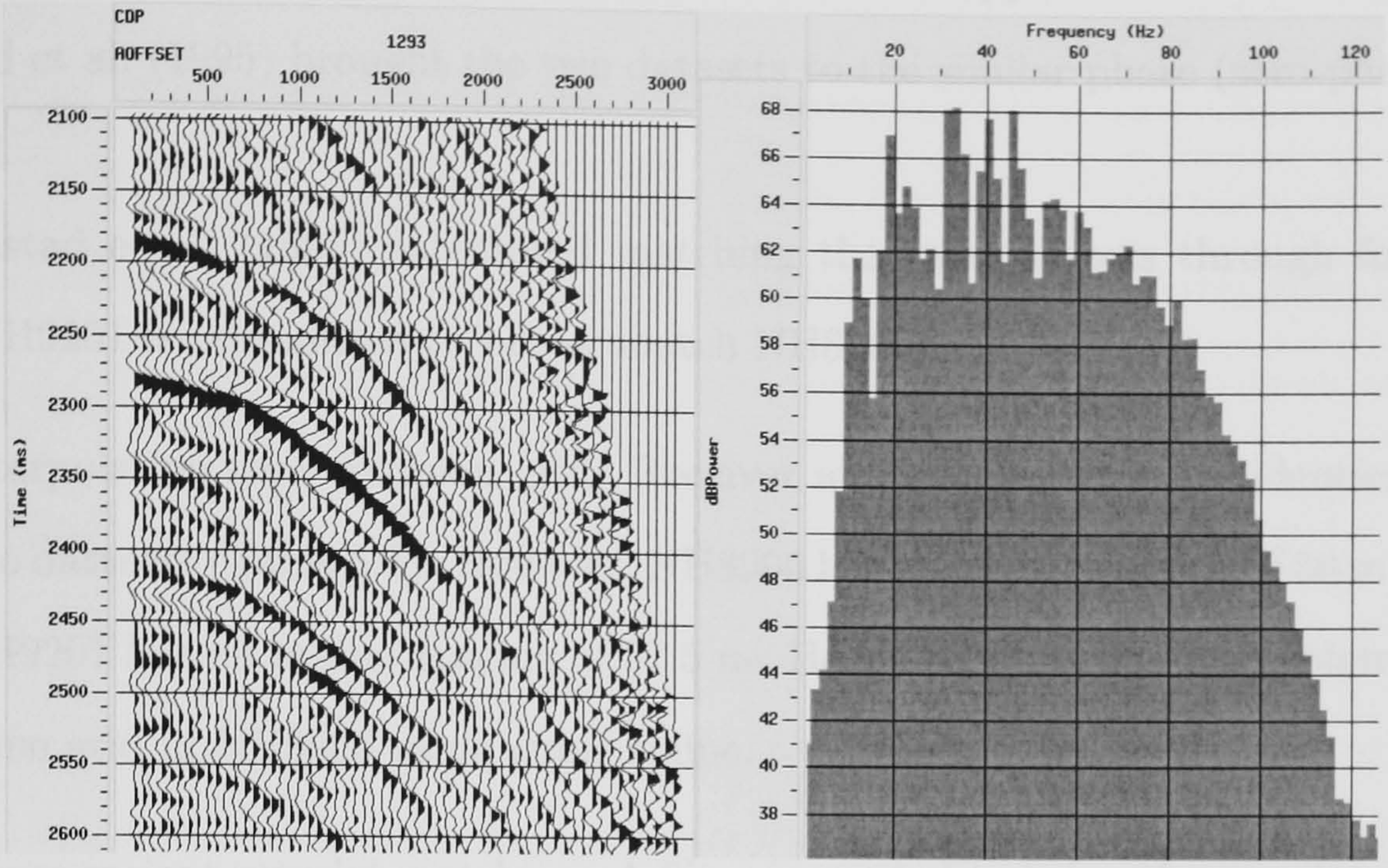


(a) Base Survey

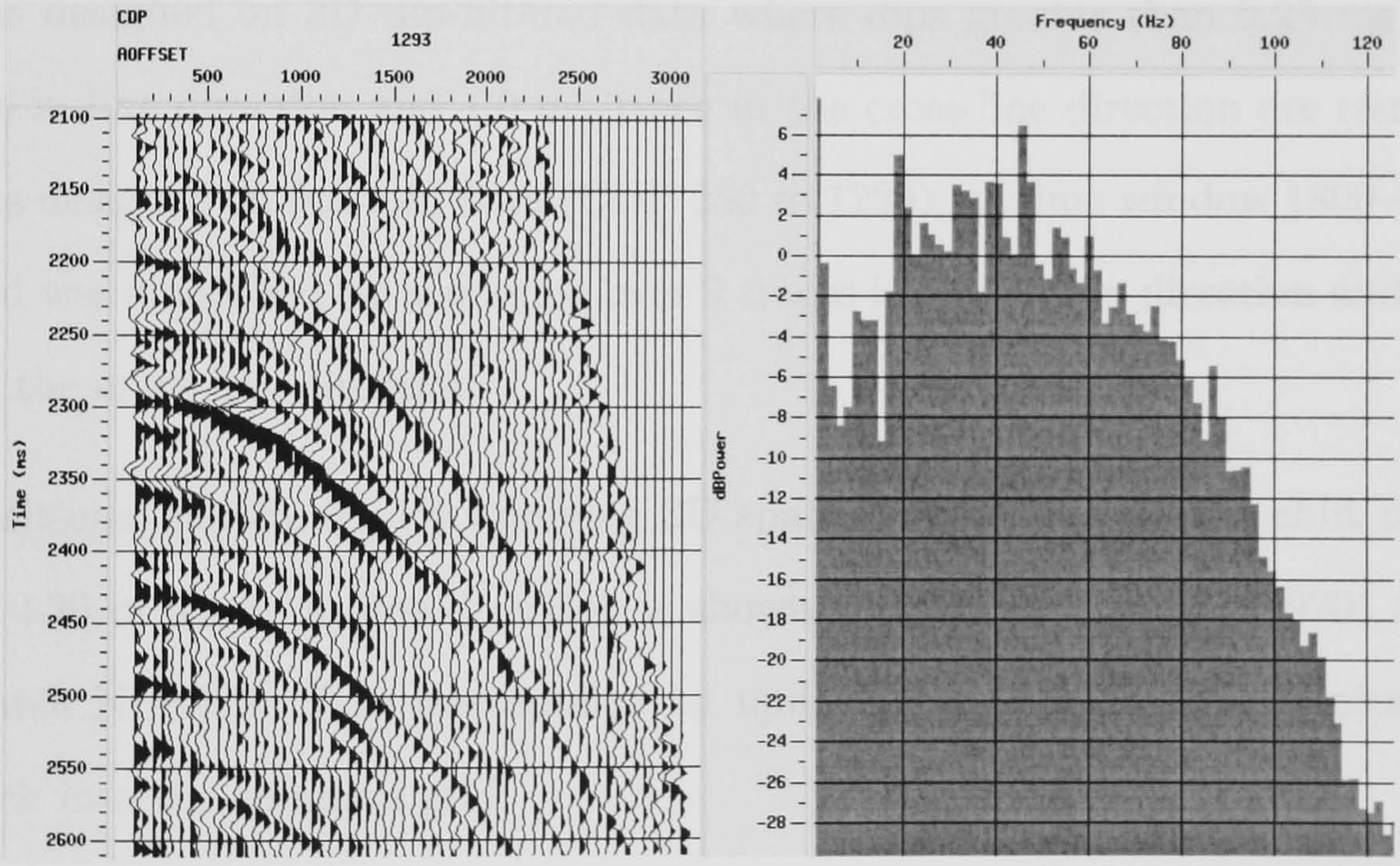


(b) Monitor Survey

Figure 3.17: Spectra of the two time-lapse datasets at CDP 1293 after applying top mute to remove direct arrivals. Each plot shows the spectrum on the right and the seismic section from which the spectrum is calculated on the left. The spectrum is plotted in dB log scale with the reference power being 1. The removal of the direct arrivals shifted the spectrum to the lower end (*cf.* Figure 3.16). Both spectra peak at around 40Hz. Contrary to the base survey, the monitor survey has a very low frequency component (less than 8Hz). Note also the power difference between the two spectra.



(a) Base Survey



(b) Monitor Survey

Figure 3.18: Spectra of the two time-lapse datasets at time section 2100–2612 ms of CDP 1293. Top mute is applied to remove direct arrivals. Each plot shows the spectrum on the right and the seismic section from which the spectrum is calculated on the left. The spectrum is plotted in dB log scale with the reference power being 1. Both spectra are similar and peak around 40Hz with most power at a bandwidth of around 60Hz, from 20Hz to 80Hz. Contrary to the base survey, the monitor survey contains very low frequency with a sharp peak at the lower end of the spectrum. The power scale is also very different.

phase difference. It is probable that the deconvolution applied before the analysis by Johnstad et al. (1995) brought the two datasets to the similar phase (zero-phase) they reported.

Johnstad et al. (1995) attempted matching the two datasets through four steps where NH9201 is normalized to better match NH8906:

1. Interpolation to a common grid: Receiver and shot spacing are identical in the two datasets (Table 3.1). However, NH8906 has an in-line spacing of 50 m whereas NH9201 has an in-line spacing of 37.5 m. Hence NH9201 was interpolated to the same grid in NH8906 using cubic spline.

2. Space-variant wavelet matching: A filter is created to bring the two datasets to a closer match by applying a time shift, phase rotation, and scalar gain. The filter was designed on 3D dip-filtered data where dips greater than 0.25 ms/trace in the in-line direction and 1.0 ms/trace in the cross-line direction are removed. It was designed on full fold data (CDP 250 to 1220), at time window 1800–2300 ms, and was smoothed by averaging over 9 traces in the in-line direction and 3 traces on the cross-line direction.

3. Positional matching: By applying 3D spatial cross-correlation a shift of +10 m to +20 m in the in-line direction is shown necessary to bring NH9201 to better match NH8906. This was applied by updating trace headers and interpolating back into the common grid.

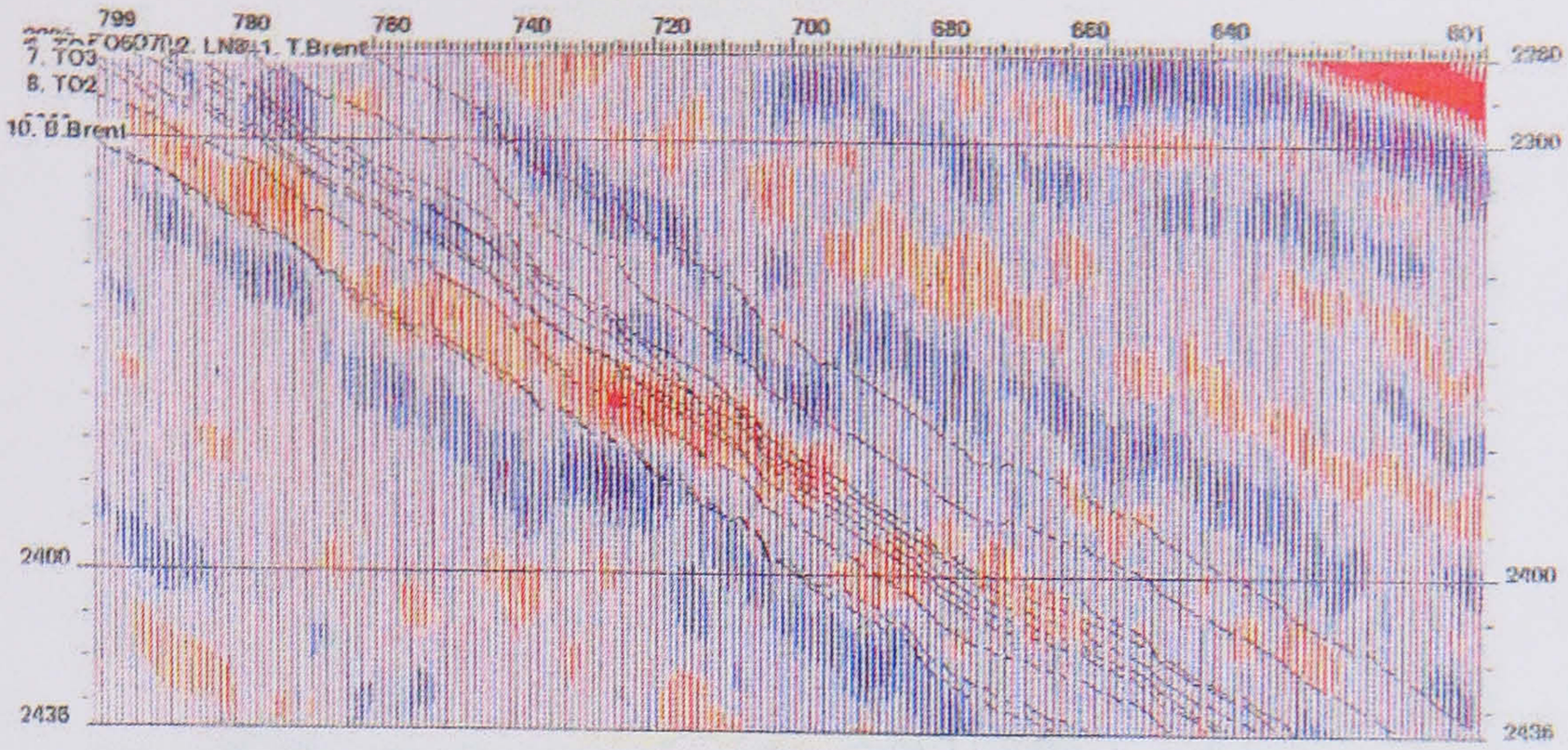
4. Local data normalization: To further improve matching and noise level, intense normalization was performed on selected lines. Time window 1900–2300 ms, just above the original gas-oil contact, was used. The procedure includes static correction and least-square cross-equalization filters, all space-variant and smoothed over 21 CDPs or 250 m. It was further found that filter performance drops significantly above 35Hz which indicates that the two time-lapse datasets (after the original processing) match least well beyond this frequency. Hence a bandpass of

12–35Hz was selected as a filter to the difference sections.

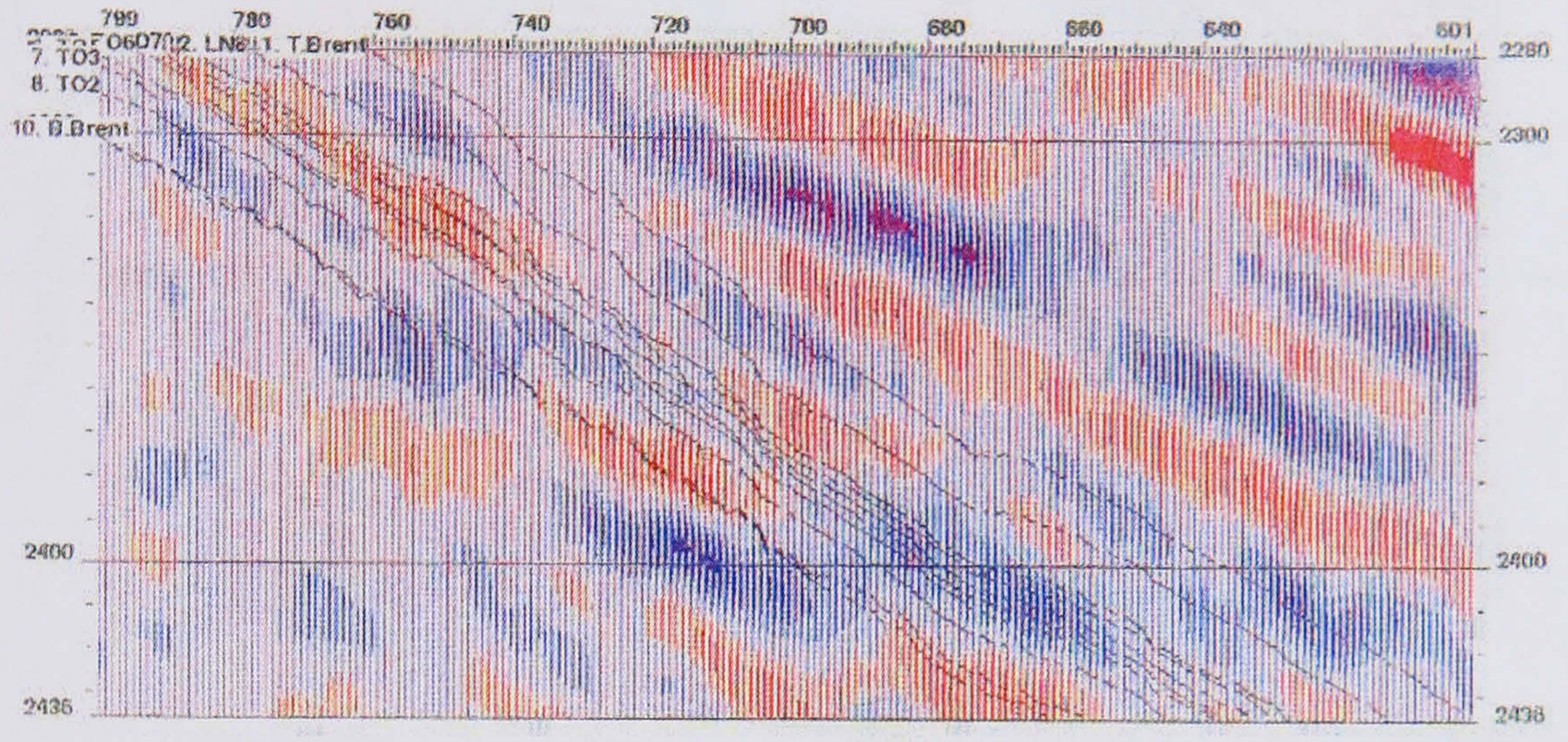
Johnstad et al. (1995) described difficulty in interpreting the difference sections between the base survey NH8906 and the monitor NH9201. These sections contained lots of noise and structural leakage, i.e. the structures seen in one dataset stack section is not fully cancelled in the difference section. They concluded that this is due partly to the noisy data of NH9201 and partly to the different processing of the two datasets. They recommended that identical acquisition and processing should be employed in time-lapse seismic monitoring. Figure 3.19 shows time-lapse seismic difference sections from Johnstad et al. (1995) and illustrates the difficulty in interpreting anomalies. Johnstad et al. (1995) still managed an interpretation of a gas front movement. This was suggested to be down dip with 50 ms vertical movement and 800 m horizontal movement. The original gas front in NH8906 was clear at around 2360 ms and vaguely recognized at 2412 ms in NH9201.

Other literature that discuss seismic monitoring at Oseberg during the period 1989 to 1992 is Johnstad et al. (1993) and Burns (1998). The latter gave a quantitative figure of change due to gas displacement of oil at Oseberg. Using Gassmann's equation, the expected changes are +0.2% in velocity and −5% change in density and hence around −5% change in acoustic impedance. Since velocity change is insignificant, no change in horizon pick-times is expected and interpretation of time-lapse is restricted to amplitude analysis. These subtle changes are inline with those found through a new time-lapse interpretation scheme by Al-Mazroey (2003).

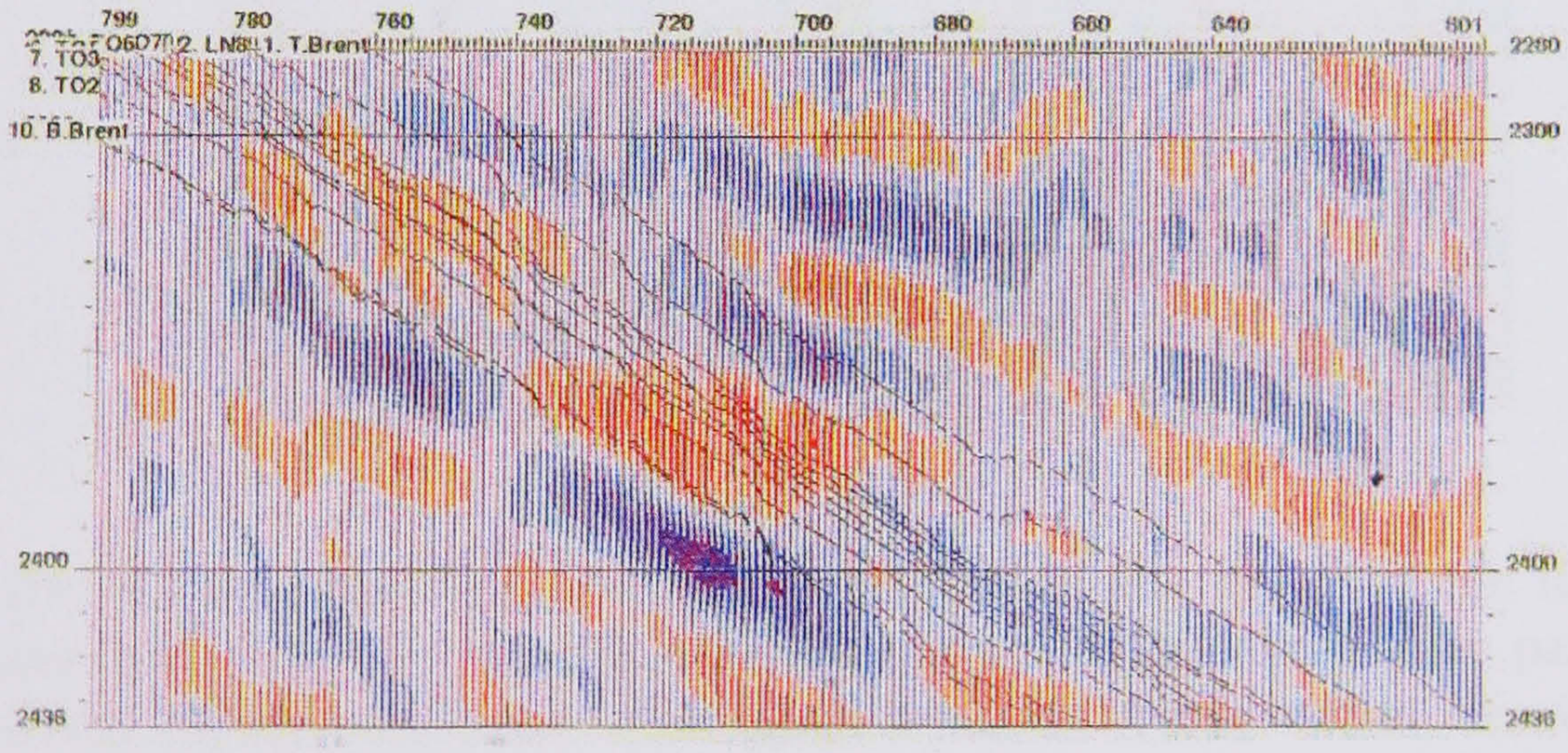
Johnstad et al. (1993) analysed the time-lapse data 1989 and 1991. The two datasets were shot using the same vessel and were processed using the same sequence. However, cross-equalization was seen to be necessary and was performed on the monitor survey of 1991. This was in the form of a filter designed over flat data and applied on the monitor to remove the effect of spatial wavelet variation. Figure 3.20 shows a seismic difference plot from Johnstad et al. (1993) with an anomaly at 2356 ms. However, structural leakage is still apparent and it is fair to say that any interpretation cannot



a) 1989/1991 difference section showing a simple red fluid front event at 2360 ms (x=720) just above base Brent (B.Brent)



b) 1991/1992 difference section showing a smaller and weaker pair of red events at around 2380 ms (x=720).



c) 1992/1989 difference section showing a large and complex red fluid event at around 2370 ms (x=720).

Figure 3.19: Oseberg time-lapse seismic sections plots. The panels show the difference seismic section between: (a) the monitor survey in 1991 and the base survey in 1989 (NH8906), (b) the monitor survey in 1992 (NH9201) and the survey in 1991, and (c) the monitor survey in 1992 (NH9201) and the base survey in 1989 (NH8906). The faint lines are horizon tracks within the reservoir. From Johnstad et al. (1995). Note that (a) shows a clear anomaly. However, the other two sections — (b) and (c) —, which involve 1992 dataset, have lots of structural leakage and are hence difficult to interpret with confidence.

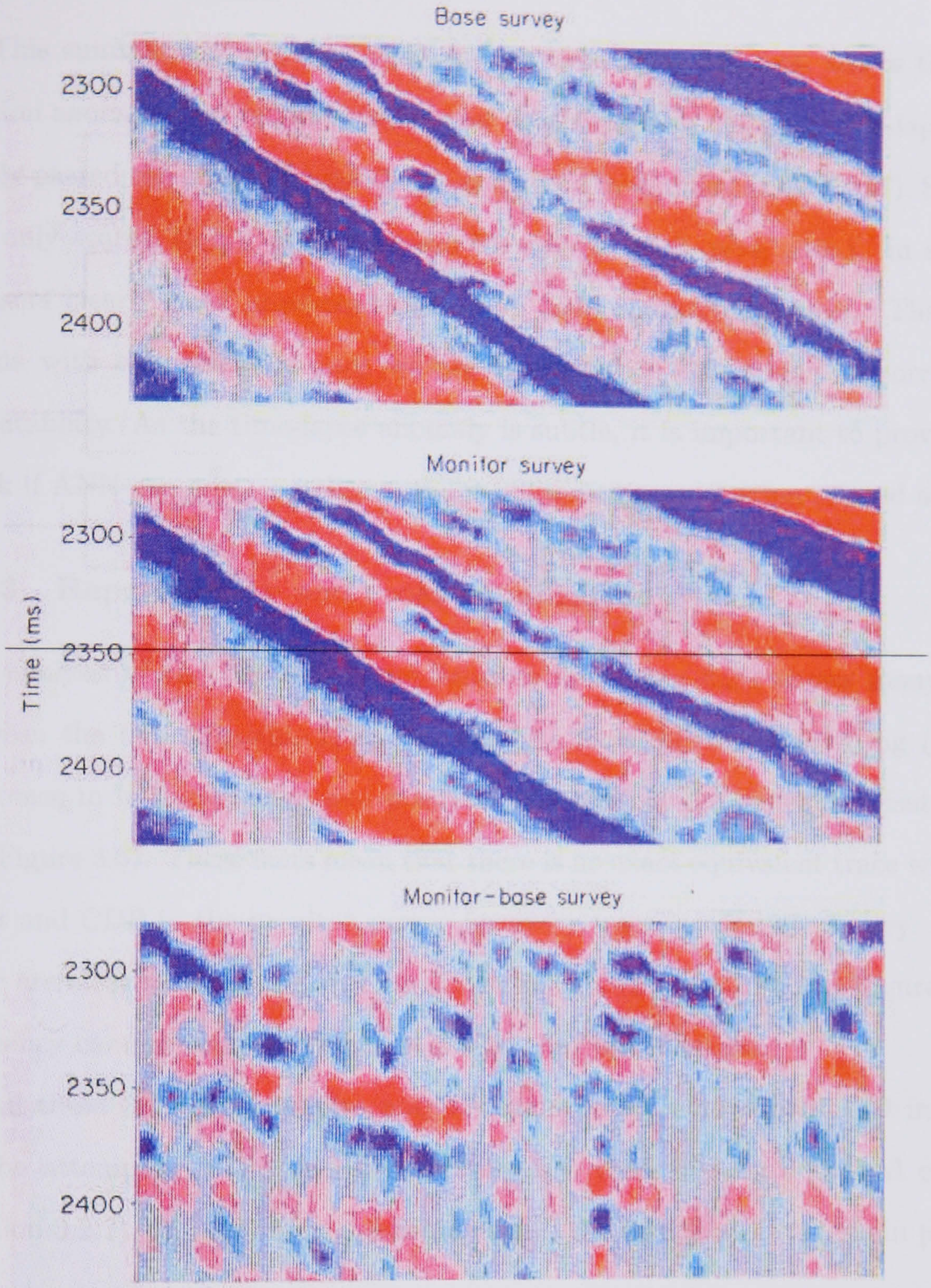


Figure 3.20: Oseberg time-lapse seismic sections from the centre of the monitoring area (Figure 3.3). The top panel is the base survey (1989), the middle panel is the monitor survey (1991), and the bottom panel is the difference section between them. The difference plot shows an anomaly, consisting of a trough (red) event followed by a peak (blue) event, within the Brent formation at time 2356 ms (one third along the section from the left). From Johnstad et al. (1993). Clearly, there is some structural leakage in the difference plot and hence it is difficult to interpret genuine production related anomalies from non-genuine ones.

be given with confidence.

This summary, of previous work on the time-lapse datasets, shows that any production anomaly that might exist is quiet subtle. In fact, Oseberg time-lapse feasibility barely passed the physical feasibility test defined by Lumley et al. (1997), Section 1.1.3, and only scored 15 out of 25 (Hamish Macintyre, 2002, *pers. com.*). In addition, the datasets clearly need cross-equalization to enable any comparison. These facts are in-line with this research since we are interested in being able to correct for non-repeatability. As the time-lapse anomaly is subtle, it is important to provide a test to check if ANN can cross-equalize without obliterating production related anomalies.

3.2.3 Repeatability and cross-equalization

Our analysis of the two time-lapse datasets in Section 3.2.1 shows many differences between the two datasets. The two datasets have different feathering (Figure 3.7), difference in in-line spacing and position, and a slight difference in offset (Figure 3.8 and Figure 3.6). These facts mean that there is no exact equivalent trace with the same offset and CDP in the monitor survey for every trace in the base survey. In addition, there are clear time, phase, and amplitude misfits (Figure 3.14 and Figure 3.15). The frequency content and bandwidth is similar, but not identical.

All these differences require normalization before a reasonable 4D interpretation can be attempted. This requirement was also described by Johnstad et al. (1995) (Section 3.2.2). In particular, Johnstad et al. (1995) suggested two main procedures:

1. spatial re-alignment and interpolation to a common grid to correct for mismatch in trace positions and variation of cross-line intervals; and
2. wavelet matching and other more local normalization in the form of least-square filters, mainly to correct for misfit in time, amplitude, and phase.

In this project, the two matching steps are attempted through ANNs. While mapping inputs from NH9201 to target outputs from NH8906, the networks correct for the misfit in position and match the traces in the least-square sense.

Chapter 4

Implementation

This study was put forward to investigate artificial neural networks (ANNs) as a time-lapse cross-equalization tool. The dataset, on which the study is conducted, was introduced in Chapter 3. In addition, earlier chapters have already discussed and introduced time-lapse, cross-equalization, and ANNs. It is now possible to discuss the procedure and implementation.

The implementation of ANNs as a time-lapse cross-equalization tool is hindered by the overwhelming number of parameters and choices that need to be made. A decision is required on ANN's topology, input, target output, and pattern pre-processing. Furthermore, a decision on how to address the problem of utilizing ANNs to cross-equalize the datasets is required to make it both practical and beneficial. The problem needs to be subdivided to allow efficient training and application while allowing the scheme to produce the desired effect, i.e. cross-equalization. All these aforementioned parameters require careful analysis and an optimum choice, which should produce a successful cross-equalization. In addition, since the use of ANNs in cross-equalization is a novel technique, guidance in the literature is fairly minimal and has only indirect implications. This is also the case with the required codes, where most necessary codes need to be written precisely for this project.

This chapter will describe the different parameter choices made and the reasons

behind these choices. It should be noted however that not all possible choices can be analysed. In addition, since many parameter choices have an impact on each other (in a non-linear fashion), it will not be possible to test all the combinations. Furthermore, during the progress of this study, I investigated many options, some of which did not produce a positive result, and some (e.g. topology, normalization, and pattern choice) are analysed many times at different stages of progress to be sure that the outcome is suitable at each particular stage with the current parameters. Therefore, it will not be useful and practical to present all these tests.

4.1 Method

While testing ANNs as a cross-equalization tool, the method is addressed as matching the monitor survey to the base survey. Figure 4.1 illustrates schematically this intention. The method is further narrowed by trying to produce a close match to one in-line[1] of the base survey from the three in-lines of the monitor survey. The base survey in-line chosen is that acquired by the second streamer (higher in y-axis, see Figure 3.7), which lies most closely to the y-position of the middle streamer in the monitor survey.

In seismic surveys, each seismic record (trace) can be considered as part of different domains or gathers, whereby each gather shares a particular common character. Figure 4.2 illustrates this idea and shows four main gathers: the common-shot gather, the common-receiver gather, the common-midpoint (CMP) gather, and the common-offset section. These subdivisions enable one to assess a trace for different characters that it shares with each gather. In addition, the subdivisions will be useful to the choice of the network's input and target output and useful when allocating an ANN to different parts of the datasets. Note that I commonly use the common-depth-point CDP as synonymous to the CMP, although, strictly speaking, this is only true for seismic sections of horizontally layered stratigraphy. In addition, I define an offset gather as the collection of seismic traces acquired by one channel (receiver) within the streamer.

[1] An in-line is defined as the seismic data collected by one streamer in one survey swath.

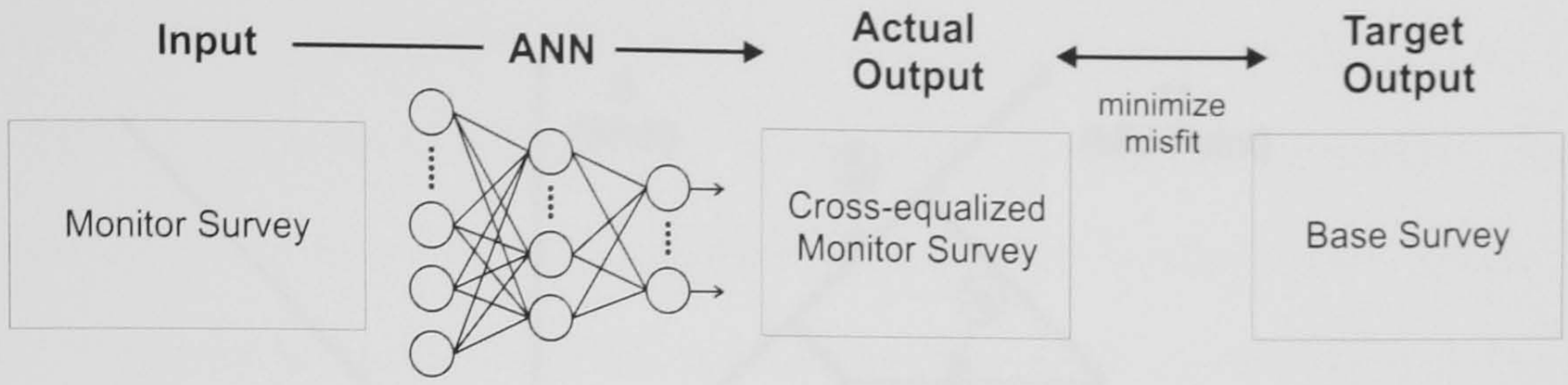


Figure 4.1: Cross-equalization using ANN. It is intended to produce an output that matches the base survey, NH8906. This output will be the cross-equalized monitor survey and will be mapped by ANN(s) from trace information from the monitor survey, NH9201. During training, ANN(s) will minimize the misfit between the cross-equalized monitor and the base survey.

although a channel may actually vary slightly in offset.

While matching two datasets, one designs a matching filter that is applied to a particular part of the dataset at each time. Here, the filter is an ANN that matches or maps an input pattern $\mathbf{x}$ from the monitor survey to a corresponding target output pattern $\mathbf{v}$ from the base survey. An ANN has to match many such input-output pairs. The set of such patterns, i.e. the matrices X and V, are used to train the network to enable it to act as a matching filter.

Usually, it is desirable to subdivide the datasets into sets of different characters where each requires a slightly different filter — or in this case, different ANNs. This simplifies the problem since each subset alone requires a simpler filter (or ANN) than if the filter was designed to cater for all. In addition, by allocating an ANN to each subset, the matching can be run more efficiently since the matching is made simpler, and the split allows parallel implementation. In this study, I chose to subdivide the problem into the matching of independent common offset gathers, where each offset gather is matched by a different network(s). Figure 4.3 illustrates schematically how the matching is subdivided where each offset is matched separately.

The choice to subdivide along the offset axis is motivated by a few facts that support this subdivision, but do not support other subdivisions like CMP and Shot gathers, nor do they favour performing no subdivision. These facts are subsequently listed.

1. Intuitively, each offset might only require one constant time correction but a range

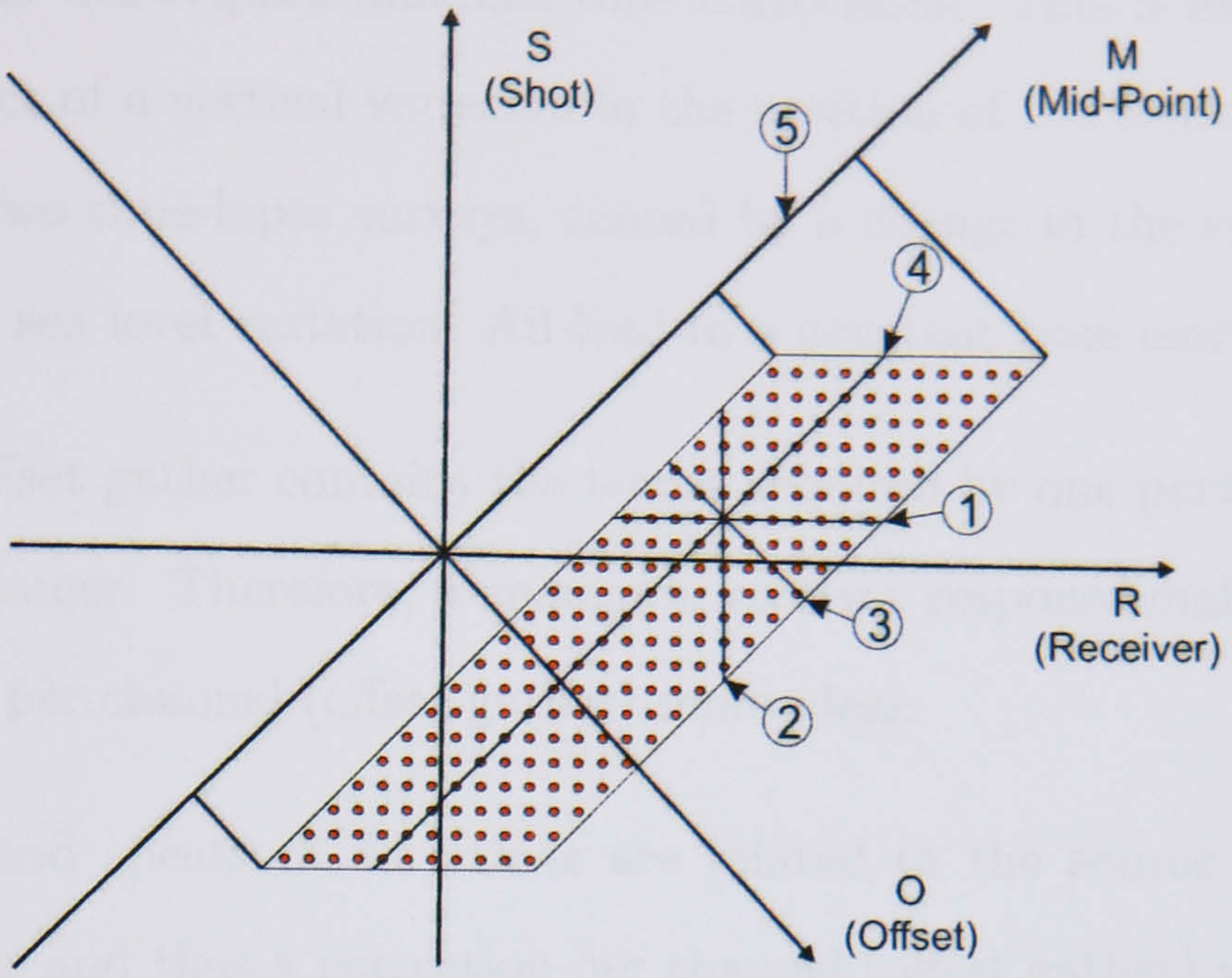


Figure 4.2: Schematic stacking chart defining gathers and showing a trace as part of a common mid point, offset, source, and receiver. Each dot represents a single trace with the time axis perpendicular to the plane of the page. Shot-receiver (S,R), and midpoint-offset (M,O) coordinates are superimposed with the (M,O) plane rotated 45 degrees with respect to the (S,R) plane. Here, (1) is a common-shot gather, (2) is a common-receiver gather, (3) is a common-midpoint (CMP) gather, (4) is a common-offset section, and (5) is a CMP stacked section. Redrawn from Yilmaz (1987, 2000).

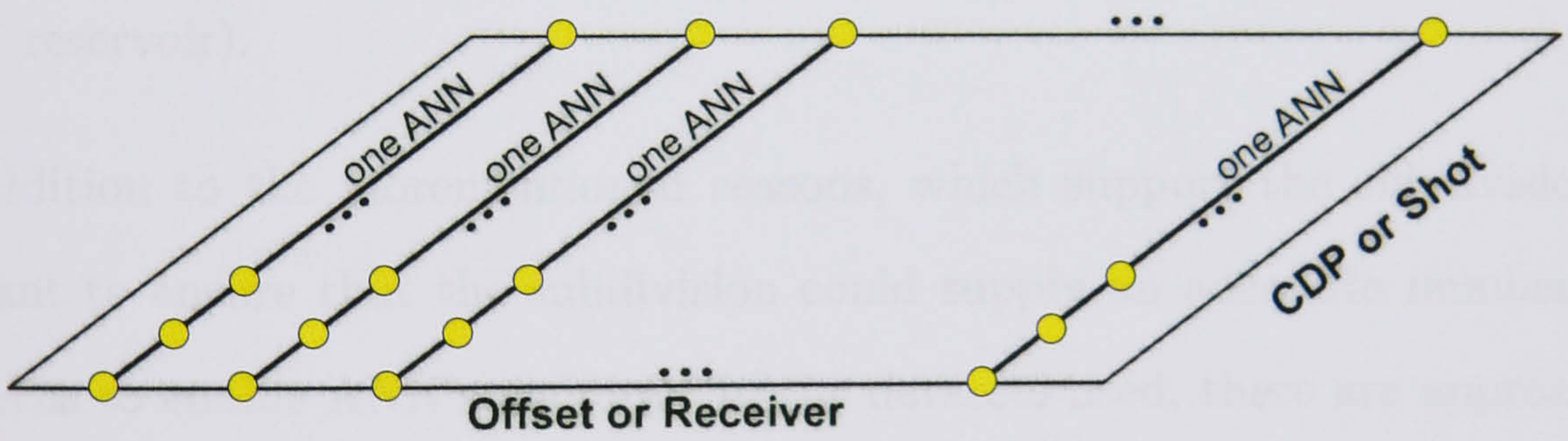


Figure 4.3: Each offset gather is matched separately. The box represents one in-line of the base seismic survey. Each large circle represent a single trace, where the time axis is perpendicular to the plane of the paper. The horizontal axis is the offset and the other axis is either the CMP or shot depending on the sorting of the dataset. Each offset gather on the base survey will be matched independently from the monitor survey by one ANN. In practice, matching might also be subdivided on the time axis and so a group of ANNs might be allocated to one offset gather. It should be noted that here each offset gather corresponds to one channel within the streamer. Strictly speaking, one channel may vary slightly in offset, but nevertheless I define an offset gather as that collected by one channel.

of offsets will require different time corrections. This is explained by analysing the effect of a vertical variation in the position of the source and/or the receiver in the two time-lapse surveys, caused by a change in the survey geometry or by tidal or sea level variation. All lead to a constant time correction for each offset.

2. Each offset gather contains the traces acquired by one particular channel within the streamer. Therefore, a change in receiver response makes an amplitude correction per channel (offset gather) more ideal.

3. Phase and spectrum variations are related to the source wavelet and receiver response and thus a correction per channel (offset gather) is more appropriate.

4. One ANN per Offset allows training across the whole survey area, and thus enabling the ANN to act as a filter that corrects for acquisition differences and still be appropriate for the reservoir area. This will not be possible if the subdivision was for shot or CMP gathers, where the networks will either be trained at a reservoir or non-reservoir area, and thus unable to acquire the ability to normalize for acquisition differences while being appropriate at the location needed (the reservoir).

In addition to the aforementioned reasons, which support the subdivision, it is important to ensure that the subdivision could supply an adequate number of training patterns to enable ANN's training. In the datasets used, there are approximately 460 shots and hence these many traces within an offset gather.

The discussion above also illustrates one important criterion I followed during this study. The choice of the input and target output of the network is not only derived from trial and analysis, but also from the thought to what might constitute a good choice. The network is constantly thought of as a filter or a function that should map a certain input $\mathbf{x}$ to its correspondent output $\mathbf{v}$. Thus, any priori knowledge of the problem should be imprinted on the selection of $\mathbf{x}$ and the expected $\mathbf{v}$. It also helps to think of an ANN as an "intelligent" being to which we are giving a set of input

information ($\mathbf{x}$) and requiring an answer ($\mathbf{v}$). By showing both $\mathbf{x}$ and $\mathbf{v}$ during training, an ANN learns any relationship that might exist between the two. Therefore, we do not want to overwhelm the network with unnecessary input or complicate the problem unnecessarily, nor restrict from it information that might be crucial (Section 5.3 will show an example of this approach).

4.2 Patterns

Patterns are the set of input and target output parameters that are associated with ANNs. These were described in Section 2.3.1 and are summarized in Figure 2.9. The choice of these patterns involves selecting the required traces, the number of required samples, the domain in which they are presented, and additional information other than the seismic data, e.g. survey geometry information, that should be incorporated. In this section, these choices are introduced, and simple and general decisions are made. Detailed investigation is left for the next chapter.

The decision of what constitutes a target output usually comes prior to selecting the input. This is because it is easier and more appropriate to select the information needed as an input given a certain target output.

4.2.1 Traces involved

The target output $\mathbf{v}$ of an ANN is selected from one particular trace r_b from the base survey. This trace should be from the same offset gather represented by the ANN. The input $\mathbf{x}$ is then selected from the trace r_m that lies on the nearest relative position to r_b but on the monitor survey. Nearest position means that it lies at the nearest offset and CMP to that of r_b. In addition, r_m is selected from the middle streamer of the monitor survey since this lies closest to the y-position of r_b.

In addition to r_m, the input $\mathbf{x}$ includes other seismic information from a group of related traces. These traces are neighbours to r_m within the different gathers. As an illustration, if r_m is at the crossing of the different gathers in Figure 4.2, then the

neighbouring traces are those closest to it within those different gathers. In addition, the neighbouring traces also contain traces from the other two streamers of the monitor survey. Initially, these neighbouring traces are selected as two positions in either direction of r_m in each gather. This group of traces is selected from each of the three streamers of the monitor survey. Figure 4.4 illustrates how these traces are defined.

Traces from the different gathers provide information about the character of the gather, e.g. traces from the CMP gather provide information about the midpoint, whereas traces from the common source gather provide information about the source. In addition, neighbouring traces allow the ANN to correct for any mis-position or feathering between the two time-lapse traces r_b and r_m.

4.2.2 Amount of data used

The amount of seismic information given from each trace is the subject of the next chapter. In general, the target output $\mathbf{v}$ contains only one time sample, frequency component, or wavelet coefficient (see next section) from trace r_b. On the other hand, the input contains a window of time, frequency, or wavelet coefficients from trace r_m and fewer coefficients from all the other neighbouring input traces.

The network might also benefit from the knowledge of the relative geometrical positions between r_b and r_m. These might include the difference in the spatial location (x- and y-position) of the sources and receivers of the two time-lapse surveys. Figure 4.5 illustrates the input and output related to one ANN.

4.2.3 Domain

When selecting input and target output to an ANN, one could consider various representations of the seismic record. In this section, I will describe the three main representations and attempt to outline their relative merits.

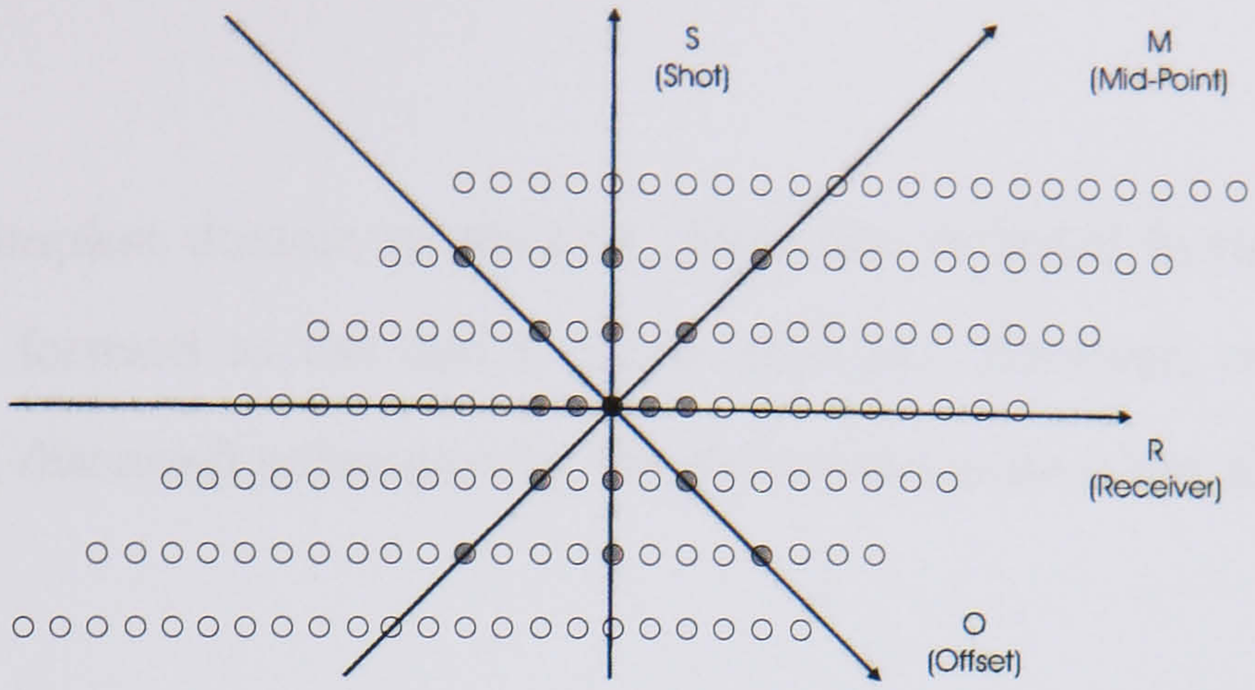


Figure 4.4: Schematic stacking chart defining neighbouring traces. The figure is a zoom from Figure 4.2, where each circle represents a single trace. This stacking chart is of one streamer from the monitor survey. The shaded circles represent those traces included as input for an ANN. If the required input trace r_m is the centre black trace, then its neighbours from the different gathers are the shaded traces. The same pattern of traces are selected from the other two streamers.

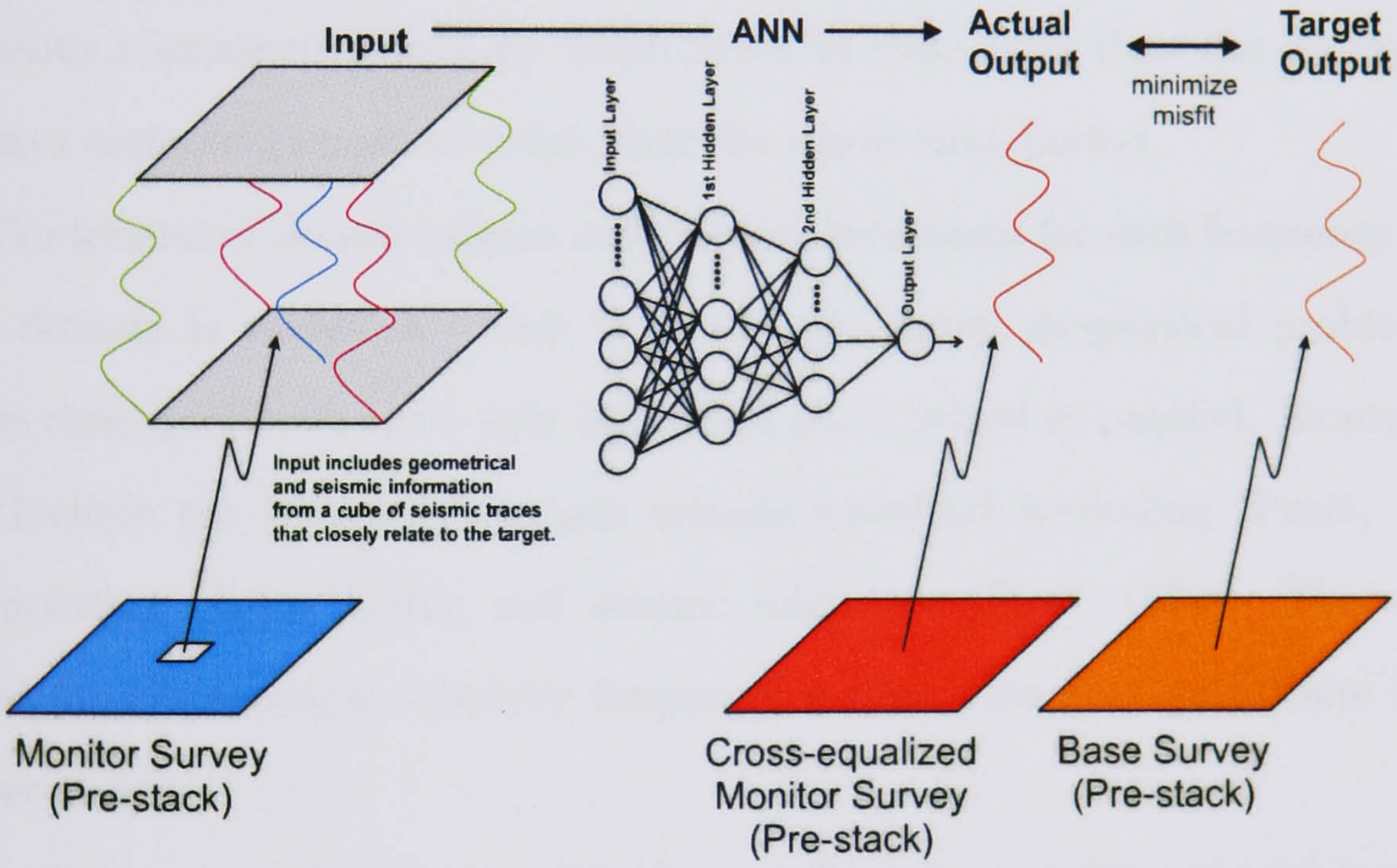


Figure 4.5: Illustration of the input and the output of the network. This figure follows from Figure 4.1 and explains further the type of input and output that are involved. Each parallelogram represents a stacking chart of the survey labelled below it. One ANN is designated to one offset gather and thus all patterns associated with that offset gather are used for that particular ANN. The target output is selected from a certain trace r_b from the base survey. The input is then from the trace r_m that lies at the nearest position in offset and CMP as r_b, but from the monitor survey. The input also includes seismic information from other traces related to (neighbouring) r_m as explained in Figure 4.4 as well as geometric parameters relating the relative position of r_m to r_b. The actual output of the network is the cross-equalized monitor survey that is cross-equalized to match the base survey. During training, the actual output is continuously made more similar to the target output by minimizing the error (misfit) between them.

Time

This is the simplest domain to work in. Data are recorded in this domain, and thus it is straight forward to use and is more intuitive. However, contrary to the other two domains, discussed subsequently, the time domain does not allow an obvious data compression.

Frequency

A time series could be represented as superpositions of sines and cosines (an infinite harmonic series) through the *Fourier transform* (Hatton et al., 1986). This transform represents each frequency within the series as linearly independent from the other. The frequency components have no localization in time, and thus the reverse transform assumes each component to come from the whole time period.

The frequency domain allows independent treatment for each frequency component. This domain is therefore widely used to solve many geophysical problems, since it allows these problems to be split into small parts solved in parallel. Examples of these uses include the frequency domain seismic wavefield modelling (Pratt, 1999), trace interpolation (Spitz, 1991), and seismic migration (Stolt, 1978). This domain also allows for compression, whereby frequency components that contribute little energy are set to zero.

In this study, the frequency domain can allow the matching of each frequency component independently. Therefore, an ANN could be allocated to each frequency component within each offset gather. In addition to the benefit of simplifying the matching by allowing small parallel implementations, this also allows the selective matching of the frequencies that are of interest. Frequencies with low energy (power) can be set to zero with no matching. Furthermore, those frequencies that cannot hold information about production-related changes (e.g. very low frequencies) can be used directly from the base survey. However, the wavelet domain enables similar implementation with added advantages.

Wavelet

A time series could also be represented as a series of *wavelet coefficients*. Whereas the Fourier transform maps the total time–domain signal into a spectrum of frequency components, the *wavelet transform* maps the signal into components that are localized in both time and frequency. The wavelet transform is a form of multi-resolution analysis that not only allows the signal to be analysed for frequency content, but also enables an analysis of how frequencies vary with time. The wavelet transform is frequently compared to the *windowed Fourier transform* — another form of multi-resolution analysis. Whereas the latter uses constant windows for analysis, the former varies the time extent of those windows. In effect, wavelet transform takes into account *reciprocal broadening*, i.e. the lower the frequency analysed the longer the time window required, and vice versa.

Daubechies (1992) and Mallat (1989) offer a theoretical treatment of wavelet analysis. A more introductory approach can be found in Aboufadel and Schlicker (1999), Graps (1995), Kaiser (1994), Mallat (1996), and Vidakovic and Muller (1994).

In order to understand the wavelet-domain, let us take our own understanding of this domain, i.e. through our hearing process. Sound travelling to our ears is a pressure amplitude variation through time, i.e. a time series. Any time series can be a combination of different frequencies. During the hearing process one is not only sensing the amplitude variation through time, i.e. the time domain version, nor just sensing the frequency content. In fact, hearing allows us to sense both time variation and frequency variation, whereby we sense how the amplitude varies with time and what frequency exists in each time span. The domain in which both localization in time and frequency exist is the wavelet domain.

In short, the wavelet transform of a discrete time series works as follows: the series is passed through a smoothing function H and its quadrature mirror G. This is repeated J times, thus producing a wavelet representation of the series on J resolution levels

(scales).

$$J = \log_2(N) - 1, \tag{4.1}$$

where N is the total number of samples in the series. At each stage, H averages or smooths the series and passes forward to the next stage an approximation at a coarser resolution (a smoother half). On the other hand, G collects at each stage the detailed part, which is not passed forward. The algorithm is illustrated in Figure 4.6. The final series contains all those detailed parts collected and the final two coefficients produced by H, which are called the *mother-function coefficients*.

Figure 4.7 illustrates how a time series of 64 samples fits into a frequency-time plot after a wavelet transform. It also illustrates how the coefficients can be denoted. A wavelet coefficient is denoted by C_{ij}, where i represents the resolution level or *scale*, and j represents the relative time position within one scale and starts from one in each scale. The scale increases as resolution increases from the lowest resolution, where scale=1, to f_{nyq}, where scale = J (Equation 4.1). The exception is the mother-function coefficients, which actually represent no frequency or time localization and therefore the scale is set to 0.

In this study, the wavelet domain is used as an alternative to the time domain. Our preference to use the wavelet domain instead of the frequency domain stems from the following facts: firstly, like the frequency domain, the wavelet domain allows both compression and parallel implementation; secondly, and in contrast to the frequency domain, the the wavelet domain enables one to analyse or restrict matching in the time axis; and thirdly, and in contrast to the time domain, the wavelet domain enables one to analyse or restrict matching in the frequency axis. The fact that the frequency domain is more popular does not mean superiority, but more probably familiarity.

There are many types of wavelet transforms, each employing a different pair of G and H quadrature mirror filters. In this study, "Doub4" algorithm, as explained in Press et al. (1997, 2002), is used.

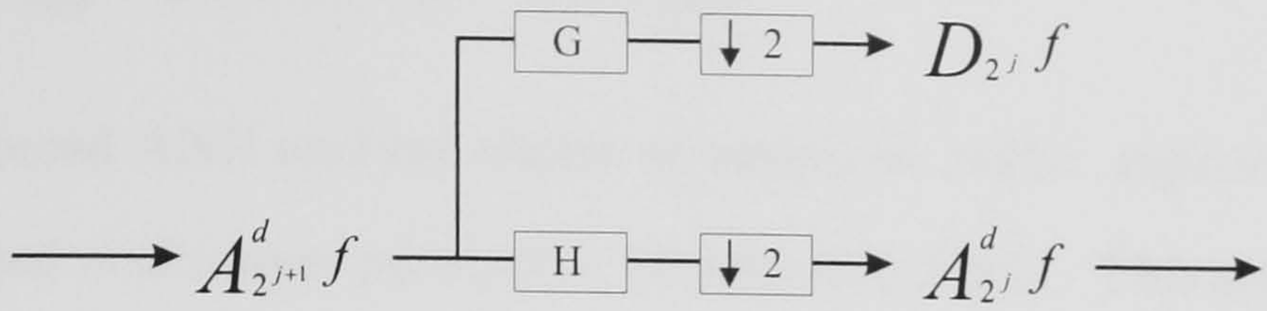


Figure 4.6: Wavelet decomposition of a discrete approximation $A^d_{2^{j+1}} f$ into an approximation at a coarser resolution $A^d_{2^j} f$ and the signal detail $D_{2^j} f$. From Mallat (1989, *p. 681*). By repeating in cascade this algorithm for $J \geq j \geq 0$, we compute the wavelet representation of a signal $A^d_{2^{j+1}} f$ on J resolution levels (scales). J is $\log_2(N) - 1$, where N is the total number of samples in the original series $A^d_{2^{j+1}} f$. The function G and H are quadrature mirror filters, where H is the smoothing filter.

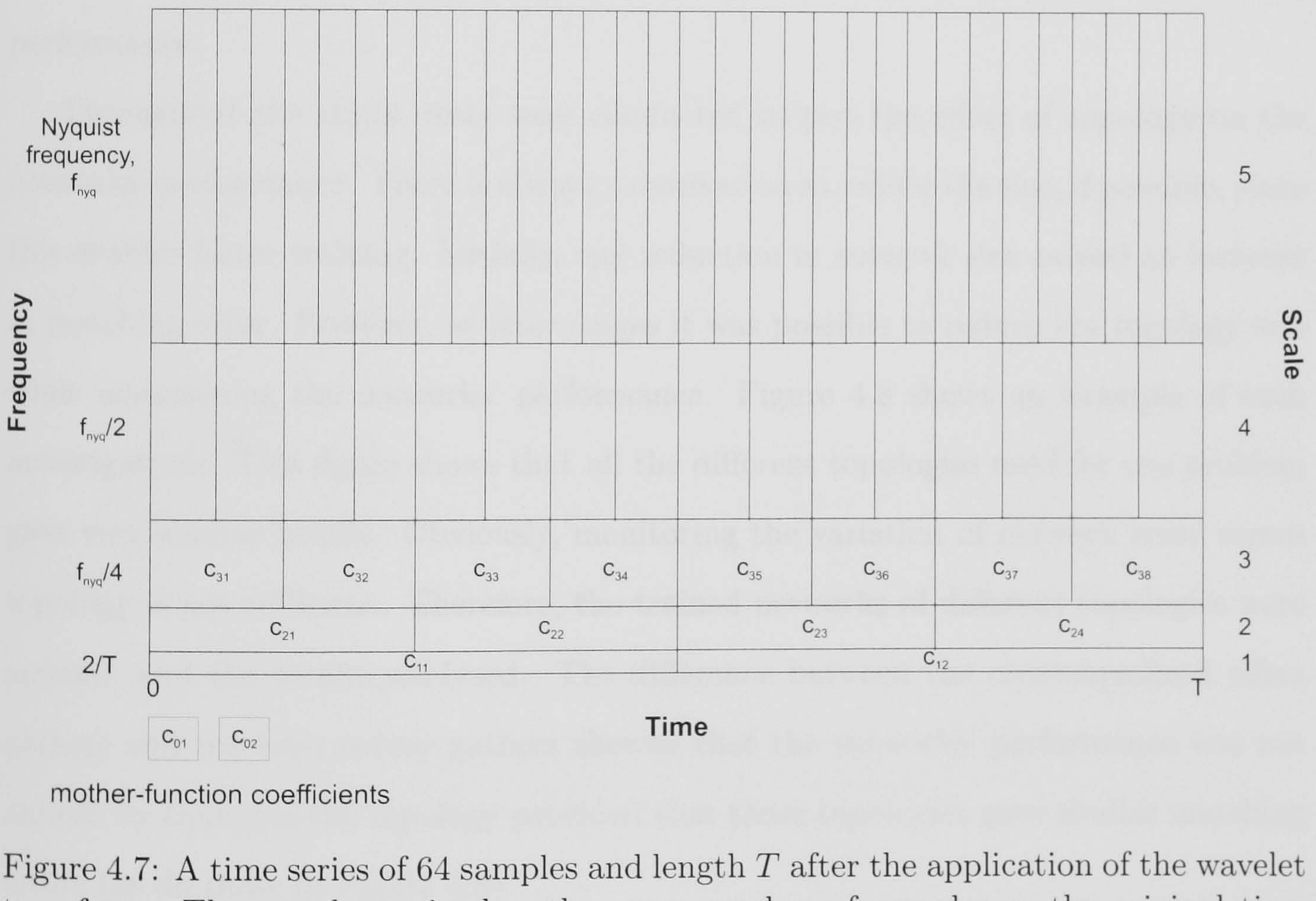


Figure 4.7: A time series of 64 samples and length T after the application of the wavelet transform. The wavelet series has the same number of samples as the original time series, which are called *wavelet coefficients*. These coefficients are represented here by boxes in a time-frequency plot. The first two are the mother-function coefficients, which are independent of both time and frequency. The other coefficients, each corresponds to a location in both frequency and time. The lowest resolution level, or *scale*, has a period of $\frac{T}{2}$. The top-most scale represents the Nyquist frequency f_{nyq} and each scale below it has its frequency progressively halved.

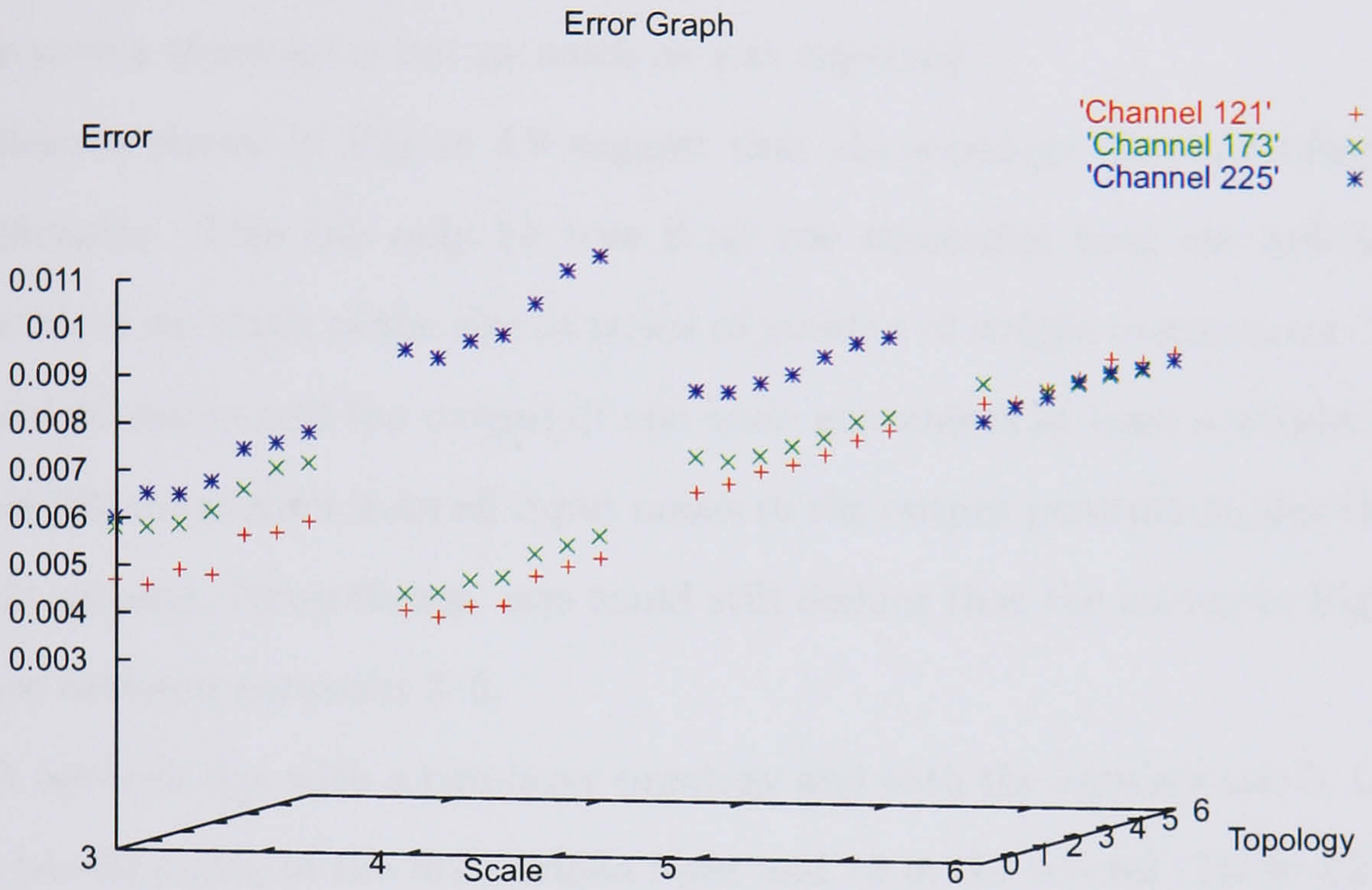
4.3 Topology and ANN design

Chapter 2 introduced ANN and my choice of using the *static*, *supervised*, *feed-forward*, and *fully-connected multi-layer perceptron* (see Section 2.2.2). This section will describe how the topology size is selected.

The choice of the topology initially appeared difficult. The main reason is the dependance of topology on the choice of input and target output parameters. Therefore, initially, topology was selected similar to the ones used by Davies (2000). The topology includes two hidden layers with relatively large number of nodes (more than 60 nodes in the first hidden layer and 30 nodes in the second hidden layer). In addition, initial tests favoured networks of similar size. The RPROP training algorithm used with weight decay enables the use of larger than needed networks while still getting good matching performance.

Throughout the study, tests were conducted to test the effect of topology on the networks' performance. There is always a motivation to reduce the size, if possible, since this enables faster training. Initially, any reduction in network size caused an increase in matching error. However, at later stages it was possible to reduce the topology size while maintaining the networks' performance. Figure 4.8 shows an example of such investigations. This figure shows that all the different topologies used for one problem gave very similar results. Obviously, monitoring the variation of network error versus topology is not sufficient. Therefore, the trained networks of different topologies were applied, and the results analysed. The difference between the cross-equalized offset gathers and the base survey gathers showed that the networks' performance was not altered by changing the topology provided that those topologies gave similar matching errors (as do those in Figure 4.8).

A further experiment was designed to test the effect of topology on network performance when most other parameters, involved in using ANN in cross-equalization, are optimized. The experiment tests networks of both one and two hidden layers, and with varying number of nodes in each layer. Table 4.1 defines four projects where the



(a) A plot of matching error versus different topologies versus different wavelet scales mapped.

Topology number	0	1	2	3	4	5	6
Nodes in 1st hidden layer	20	40	60	80	50	100	150
Nodes in 2nd hidden layer	10	20	30	40	-	-	-

(b) Topology sizes used in (a)

Figure 4.8: Example of topology size investigations. (a) shows the plot of error versus different topologies versus different wavelet scales mapped for three different offset gathers (channel 121, 173, and 225). All the different topologies used to match any gather within one wavelet scale gave very similar matching error.

network's topology size is varied and the final validation error of each network is noted. The number of nodes used in each project is varied according to Table 4.2.

The experiment follows the idea presented by Hush (1989) and which was illustrated in Figure 2.17. The results of this experiment are shown in Figure 4.9. The error curves in Figure 4.9 in general follow the shape suggested by Figure 2.17, but the variation of error versus topology is not as much as was expected.

Results shown in Figure 4.9 suggest that the topology does not affect the results significantly. This can only be true if all the topologies used are sufficient. This is plausible if we think of the size in terms of number of weight connections. The input of around 89 nodes and the output of one node guarantees at least a weight matrix of 90 values (89 connection from all input nodes to the output perceptron plus the on-neuron of the output). Nevertheless, one could still deduce that the curves in Figure 4.9 have a knee between networks 3–5.

A network size with a two-layer topology and with the network size 5, i.e. a network that has 20 nodes in the first hidden layer and 10 in the second (Table 4.2), is selected as the preferred topology to be used in future tests. This network gave a relatively good result and is approximately at the knee of the curves shown in Figure 4.9, and thus not larger nor smaller than necessary. A two-layer topology has been chosen because they are, in theory, capable of mapping arbitrary complex functions (more complex than can be mapped by one-layer topology) and will thus provide a sufficient network even if the problem becomes more complicated.

Table 4.1: The projects conducted on the topology size choice and names given to each project. The actual number of nodes used in each layer is described in Table 4.2.

Number of layers	Gather of channel	Project name
One	121	One_121
One	173	One_173
Two	121	Two_121
Two	173	Two_173

Table 4.2: Actual topology used in the projects defined in Table 4.1.

Network number	One layer	Two layers	
	No. of nodes in 1st hidden layer	No. of nodes in 1st hidden layer	No. of nodes in 2nd hidden layer
1	2	2	2
2	5	5	2
3	10	10	5
4	15	15	5
5	20	20	10
6	25	25	10
7	30	30	15
8	35	35	15
9	40	40	20

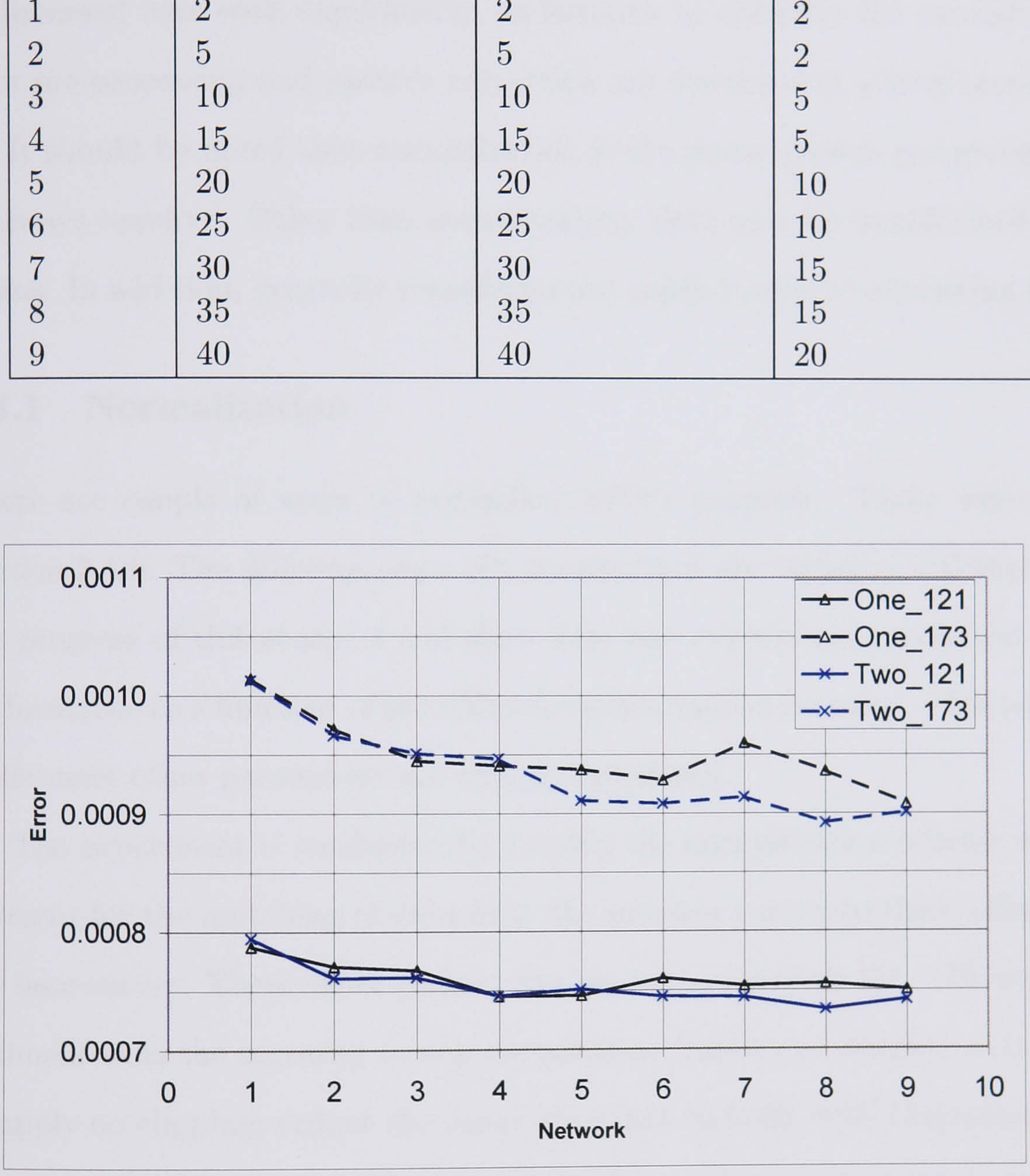


Figure 4.9: A graph showing matching error versus network size. Each curve corresponds to one of the four projects defined in Table 4.1. The horizontal axis is the network size as defined in Table 4.2. The plot of the error with increasing topology size is a practical demonstration of the idea presented in Figure 2.17. The two curves, with the lowest error that correspond to the gather of channel 121, has an approximate knee at Network 4 for both one- and two-layer topologies. On the other hand, One_173 has a knee at Network 3 and Two_173 has a knee at around Network 5.

4.4 Pattern pre-processing

There are three main stages of pre-processing: firstly, seismic data are analysed in the processing package ProMAX, and the appropriate data are prepared for pattern extraction; secondly, patterns are extracted; and thirdly, these are normalized. Normalization is discussed here with experiments undertaken to optimize the procedure. The other data pre-processing and pattern extraction are discussed in a later section.

It should be noted that normalization is the main pattern pre-processing step and is always required. Other than normalization, data may be transformed into other domains. In addition, generally transforms are applied prior to extracting these patterns.

4.4.1 Normalization

There are couple of ways to normalize ANN's patterns. These were introduced in Section 2.4.6. The different ways of normalization are tested at different stages during the progress of this study. I will show here one experiment conducted to test ANN's performance as a function of the different normalization schemes. This test is conducted while most other parameters are already optimized.

The experiment is conducted by varying the normalization scheme when preparing patterns for the matching of data from the monitor survey to three offset gathers from the base survey. These offset gathers correspond to channels 121, 173, and 225. The experiment tests the necessity to clip the patterns (input and output) as in Equation 2.28 or apply no clipping; reduce the range of output to $[0.05, 0.95]$ (Equation 2.25) or leave it as the standard $[0.00, 1.00]$ (Equation 2.24); and normalize the input by the mean and standard deviation (Equation 2.27) or by using Equation 2.26. All the possible eight combinations were tested and are numbered as in Table 4.3.

Figure 4.10 and Figure 4.11 shows the results of the tests conducted. They show that normalization is best if there is no clipping applied, output is limited to the range $[0.05, 0.95]$ (Equation 2.25), and input normalized by Equation 2.26 instead of Equation 2.27. The three conclusions are quoted in decreasing importance.

Table 4.3: Eight tests are conducted to test the effect of normalization on ANN's ability to match. Each test is a combination of three criteria as shown.

Test number	Is clipping applied?	Is the output in the range $[0.0, 1.0]$?	Is the input normalized by Equation 2.27?
1	no	no	no
2	no	no	yes
3	no	yes	no
4	no	yes	yes
5	yes	no	no
6	yes	no	yes
7	yes	yes	no
8	yes	yes	yes

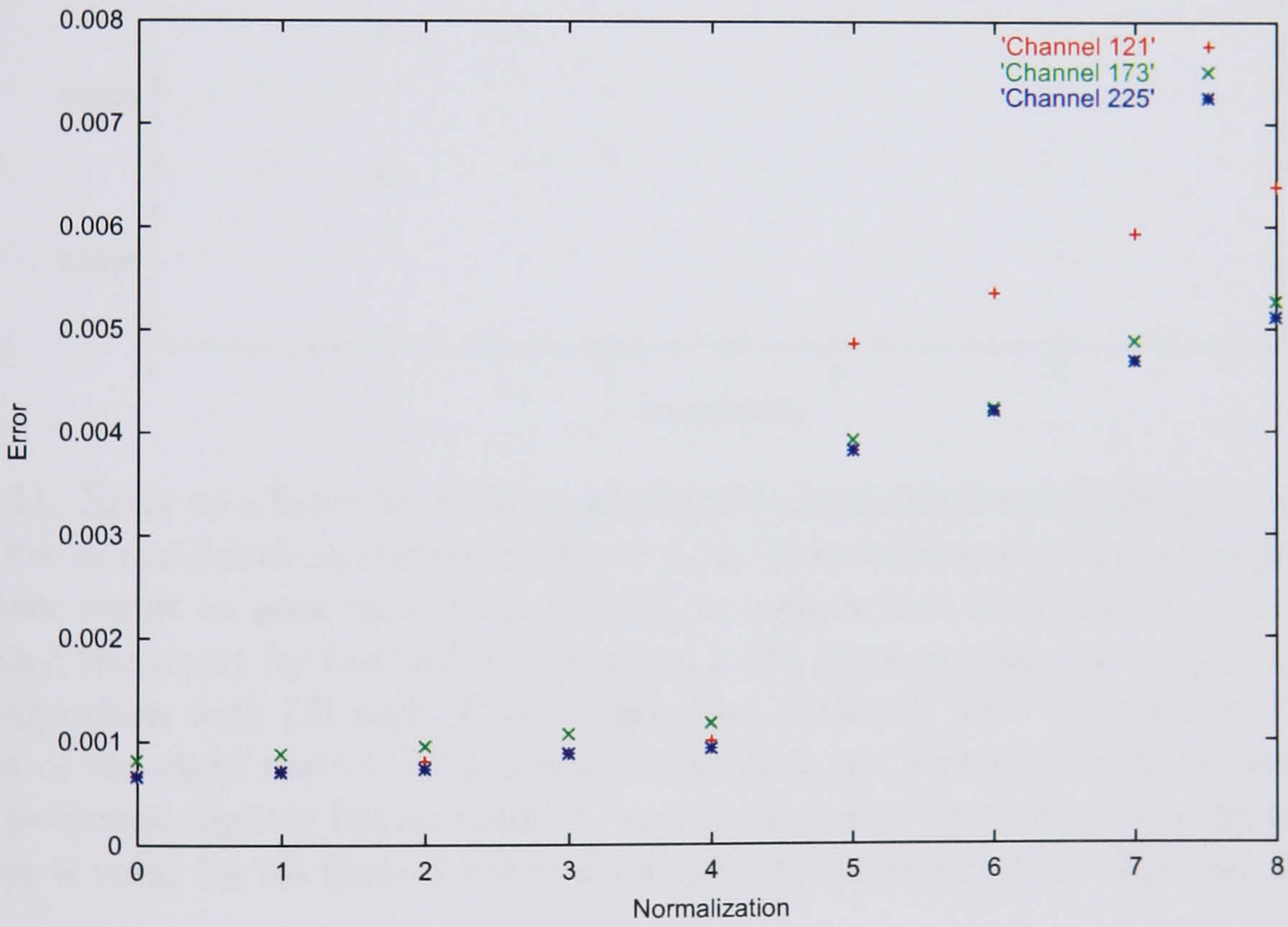


Figure 4.10: Error as a function of the normalization scheme. The error is a measure of the difference between the ANN's actual and target output. The label numbers on the horizontal "normalization" axis indicate the experiment configuration as outlined in Table 4.3. Test number zero is similar to test number (1), but the input is normalized to the range $[-0.5, 0.5]$. Error is a lot lower when no clipping is applied as in tests (0)–(4). This conclusion is correct for all three tested offset gathers (channels 121, 173, and 225).

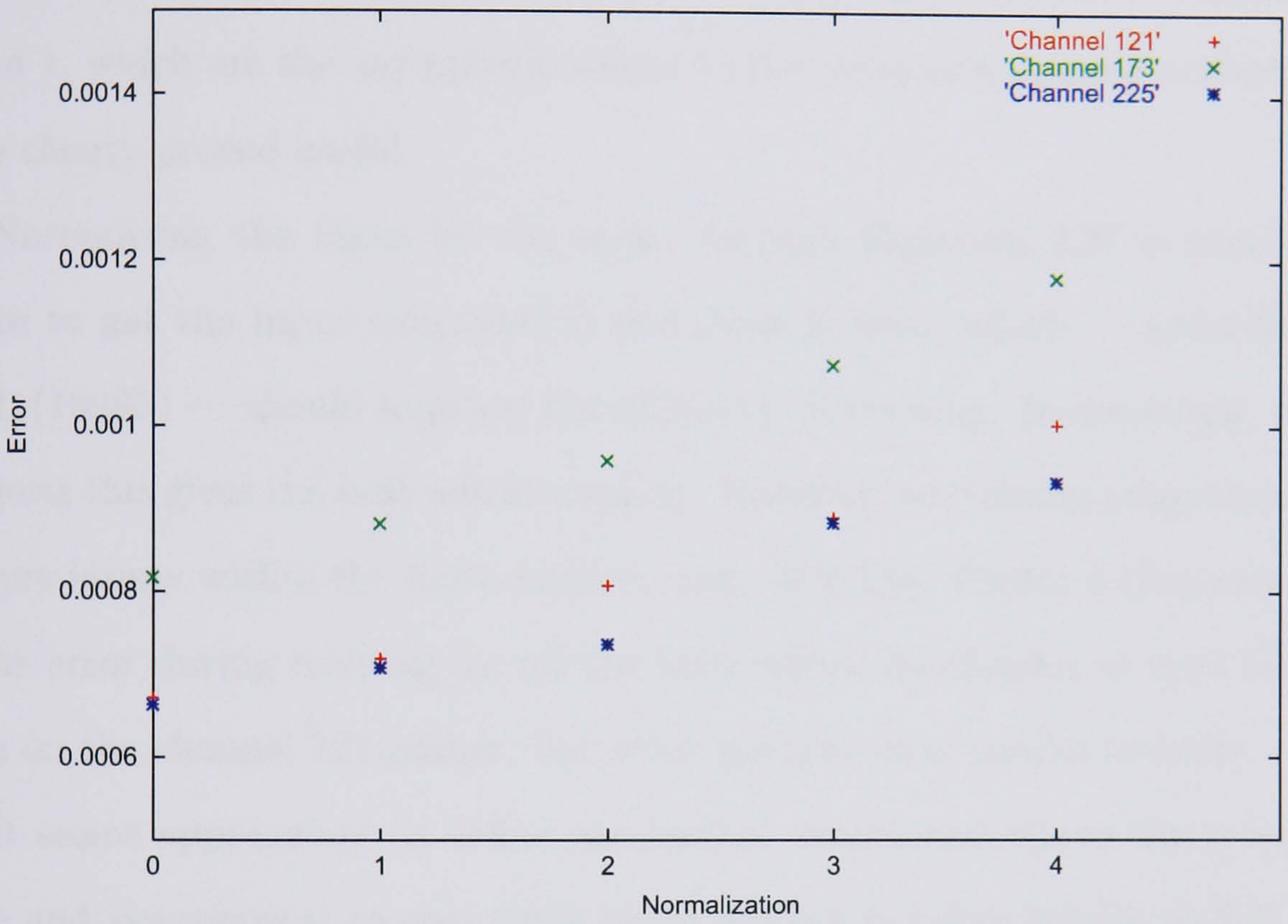


Figure 4.11: Error as a function of the non-clipped normalization schemes, i.e. a zoom at the first five normalizations shown in Figure 4.10. Normalizing the output to $[0.05, 0.95]$ gives better result as seen from runs (1)–(2) in comparison with (3)–(4). In addition, normalizing the input by the mean (Equation 2.27) increases the error as it did to run (4) in comparison with (3) and (2) in comparison with (1). Test (1) produced the best result out of the eight tests (1)–(8) since it combines the best choices of normalization. Test (0) performs slightly better than (1) and is discussed further within the text. The conclusion is valid for all three tested offset gathers (channels 121, 173, and 225).

Although clipping should get rid of problems introduced by outliers (Section 2.4.6), it actually produced an increase in error. This could be because no outlier problem existed and thus clipping only served to change the actual values and their distribution. It should be noted that the data are processed for any spurious spikes and this seems to have been able to get rid of any possible outliers.

Limiting the output to $[0.05, 0.95]$ avoids requiring the network to output the values 0 and 1, which are the asymptotic values to the network's sigmoid activation function. This clearly proved useful.

Normalizing the input by the mean through Equation 2.27 is motivated by the desire to get the input symmetrical and close to zero, which — according to LeCun et al. (1998b) — should improve the efficiency of training. Interestingly, as the theory suggests this gives the best initial training. However, as training progresses the network favours inputs within the finite limited range $[0.0, 1.0]$. Figure 4.12 shows the progress of the error during training for all the tests where no clipping is used (shown for the tests on the channel 121 gather, but other gathers show similar results).

It seems appropriate to define one further experiment where the input is set both close and symmetrical to zero while being limited in range similar to $[0.0, 1.0]$. Therefore, one additional experiment (test number 0) is defined using the best combination as in test number 1, but the input is normalized to the range $[-0.5, 0.5]$. This normalization is achieved by applying a translation of -0.5 to Equation 2.26, thus:

$$\tilde{x}_i = \frac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}} - 0.5. \tag{4.2}$$

This gave the best result as seen in Figure 4.11 and fast training as seen in Figure 4.12. Other experiments widening the range to $[-1.0, 1.0]$ give very similar results.

In conclusion, during normalization, one should apply no clipping to the training patterns and should reduce the output to the range $[-0.05, 0.95]$. In addition, the input should be normalized linearly using the minimum and maximum either to the range $[0.0, 1.0]$ using Equation 2.26, or more preferably to $[-0.5, 0.5]$ using Equation 4.2.

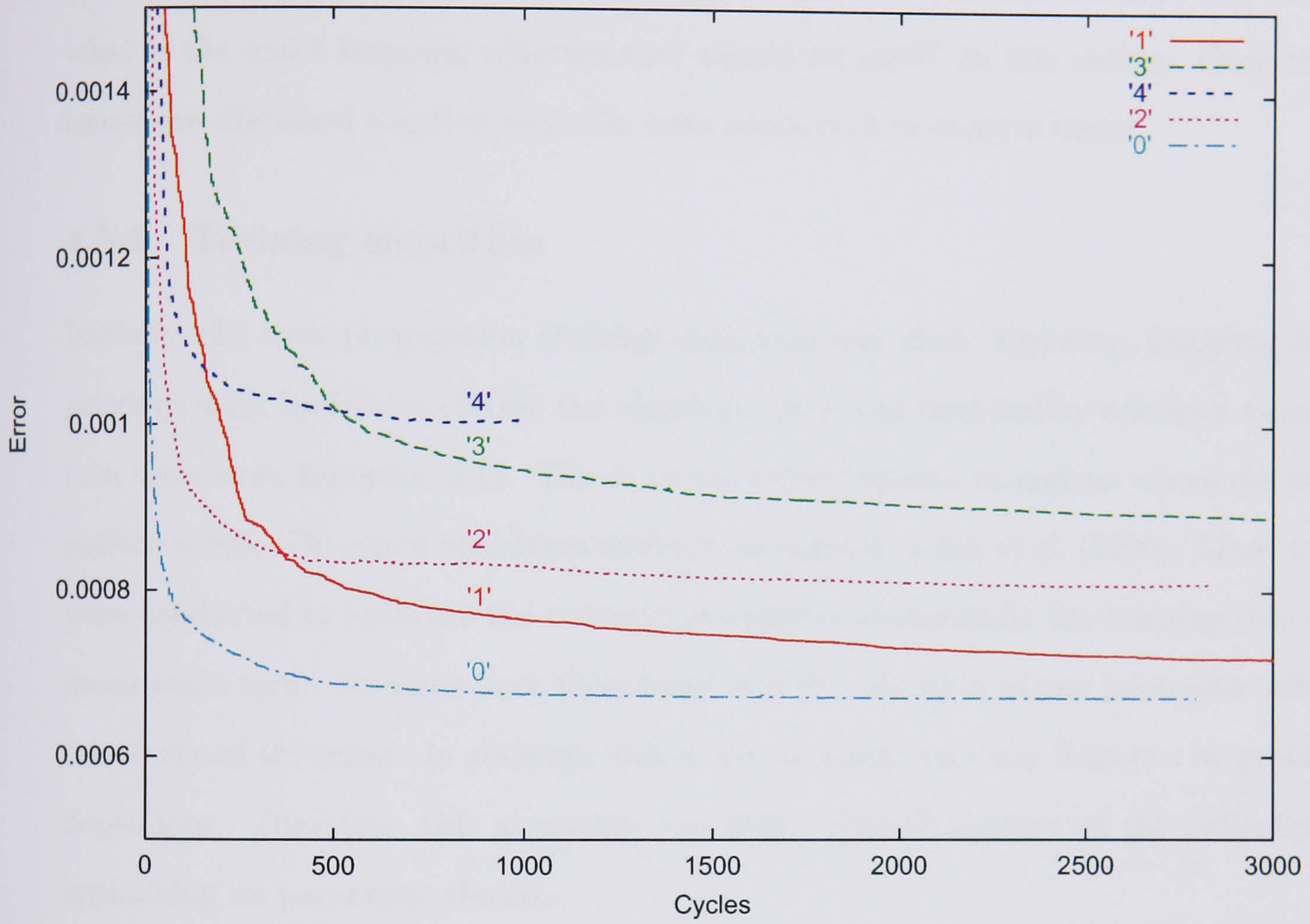


Figure 4.12: Error graph of non-clipped normalization (channel 121). The normalization numbers indicate the experiment configuration as outlined in Table 4.3. Test number zero is similar to test number 1, but the input is normalized to the range $[-0.5, 0.5]$. Those tests with inputs normalized to a range close and symmetrical to zero (2 and 4 as opposed to 1 and 3) shows better initial training, but as training progresses the network favours the other runs limited in range to $[0.0, 1.0]$. Test number 0 is discussed in the text and is both close and symmetrical to zero and limited in range ($[-0.5, 0.5]$). Thus run 0 gives best initial training and remains favourable by the network. In fact, it reaches its minimum more rapidly.

4.5 Training

While training there are several issues that need to be defined: firstly, what training algorithm is employed and with what associated parameters; secondly, should one use algorithms that combat the local minima problem or is this unnecessary; and finally, what is the exact stopping criterion that should be used? In this section, these three issues are discussed together with the tests conducted to analyse them.

4.5.1 Training algorithm

Initially, the back-propagation training algorithm was used. However, following from previous work by Davies (2000), the algorithm used was modified by adding a momentum term as in Equation 2.17. This is to accelerate training in regions where the error surface is flat. The exact algorithm used is that defined in Zell et al. (2000). Many tests were conducted to optimize the training parameters, particularly the learning rate and momentum term. It seems from these tests that the adoption of one parameter affects the choice of the other. In addition, the choice of parameters was found to be problem dependent. Therefore, this algorithm was later dropped because of the difficulty in optimizing its parameter choices.

Resilient propagation (RPROP) was selected as a good alternative to back-propagation with momentum. RPROP is described and its merits are highlighted in Section 2.3.5. The exact algorithm used is defined by Zell et al. (2000), which includes the implementation of the RPROP batch learning as described in Section 2.3.5, and is modified to include weight decay (Equation 2.23). The parameters needed by the algorithm are defined in Section 2.3.5. Two parameters are of interest: the initial update value Δ_0; and the weight decay term (Equation 2.23). Δ_0 is set to 0.1 for all weights, whereas the weight decay term is set to 3. Tests conducted indicate that results achieved using RPROP is comparable with the best results derived by back-propagation with momentum. The choice of parameters is not very crucial since the method is adaptive except for the weight decay term. The choice of weight decay term causes no significant change

in performance, and I found that a weight decay term with a value 2–4 give a good balance between fast training and reasonable weight decay.

4.5.2 Local minima consideration

The local minima problem was described in Section 2.3.4(1). Two solutions were put forward: multiple weight initializations and weight jogging. The two solutions and their benefits are analysed in this section.

Multiple weight initialization

Training might be performed several times each with different weight initialization to ensure starting at a different point on the error surface. The run with the lowest error can then be selected as a solution. This was done at different stages, and in all it was clear that the different initializations give very similar results. Therefore, it can be concluded that the problem — as addressed and with the ANNs design I use — does not suffer from a local minima problem.

Figure 4.13 shows the results of one example of such tests conducted to test the benefit of multi-initializations. Training was conducted three times each with different weight initialization (different runs). The figure shows the error for all the tests that were repeated, where topology was changed and different offset gathers were used. Each individual test produces very similar errors in all the different initializations. Therefore, there is no benefit in repeating the training.

Weight jogging

Weight jogging, whereby weights are changed by a small random amount, can help to push the network out of local minima. An experiment was conducted to test the effect of weight jogging on the network's performance and on error progress with iteration steps. The experiment was conducted while training a network to match the offset gather of channel 200. All other parameters, such as topology and input-output patterns, were

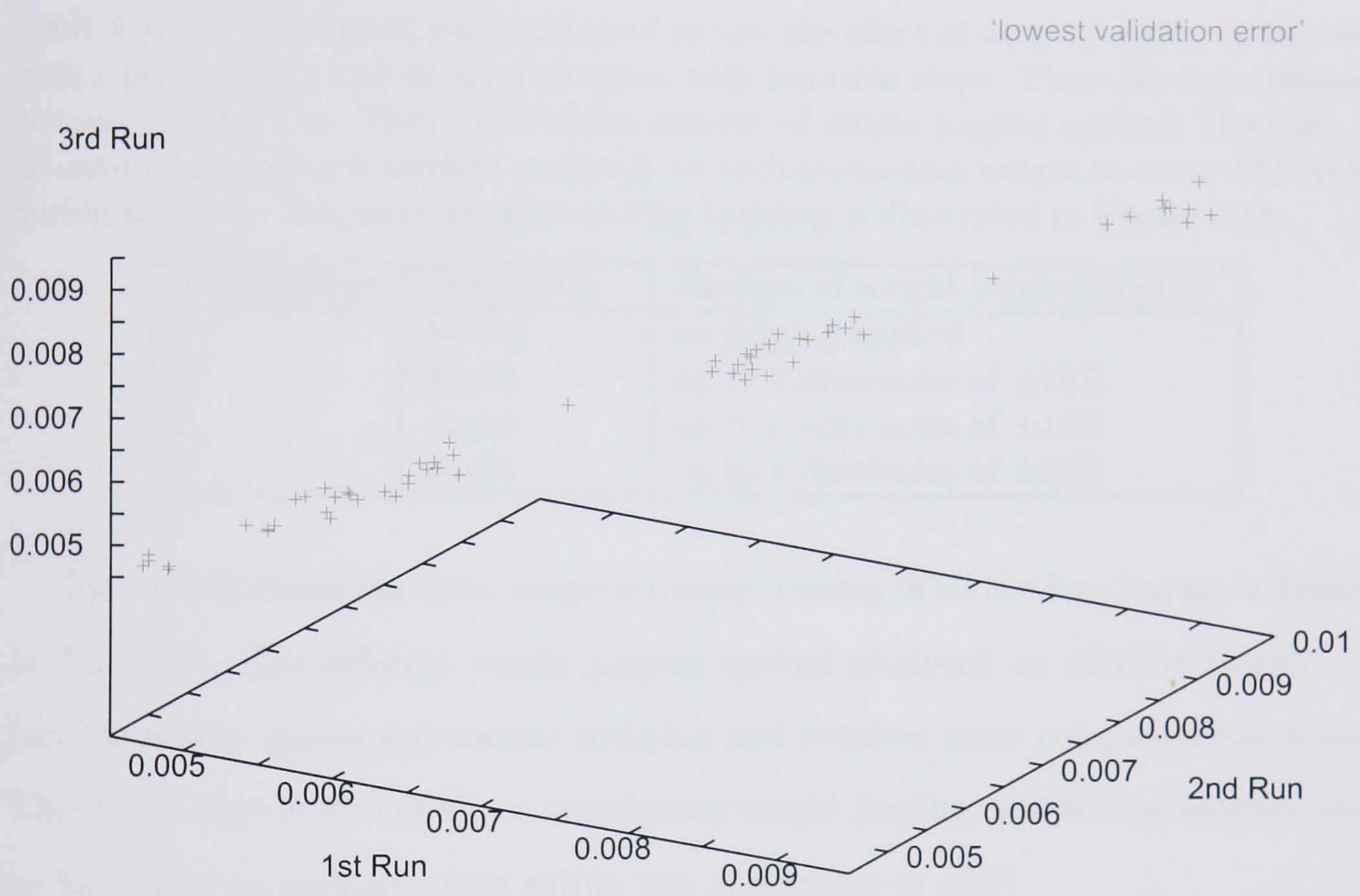


Figure 4.13: Error plot comparing three different weight initialization for many tests. Each axis represents the lowest validation error achievable after one particular weight initialization (or run). Each individual test produces very similar errors in all the different initializations. Therefore, a linear relationship $x = y = z$ with high correlation is observed.

already optimized.

The experiment compares four training instances of the network, where the differences between the four are shown in Table 4.4. Weight jogging was applied every 100 iterations, which was found to be a reasonable choice: too frequent application causes unstable training, whereas too sparse jogging makes the jogging lose its benefit of rescuing training when stuck in local minima.

Table 4.4: An experiment was conducted to test the effect of weight jogging on the network's performance and on error progress with iteration steps. There are four different instances of training. They differ in the amount of weight jogging applied. However, in all cases, the jogging is applied randomly to each connection weight at every 100 cycles during training. The error progress during training is illustrated in Figure 4.14.

Test Number	Test Name	Amount of weight jogging applied
1	NoJog	no jogging applied
2	Jog20	up to a maximum of ±20%
3	Jog10	up to a maximum of ±10%
4	Jog05	up to a maximum of ±05%

Figure 4.14 shows the error progress during training of all the four instances defined in Table 4.4. The different weight jogging applied produced no effective benefit. In fact, it usually causes pre-mature stopping and requires more computational power. Therefore, there is no benefit in introducing weight jogging, which may indicate that no local minima problem exists within this application of ANN.

It should be noted that Figure 4.14 shows that training using weight jogging has lower initial error. The jogging seems to be useful as long as the error is relatively large. It may be worth applying the jogging with a decaying amount, where lower amount of jogging is applied with increasing number of training cycles.

Conclusion

The two experiments conducted to test the benefit of either the multiple weight initialization or weight jogging show no benefit. Therefore, one may conclude that no local minima problem exists within this application of ANN. However, there is a possibility that the problem is made of a very wide local minima and a very narrow global one. In

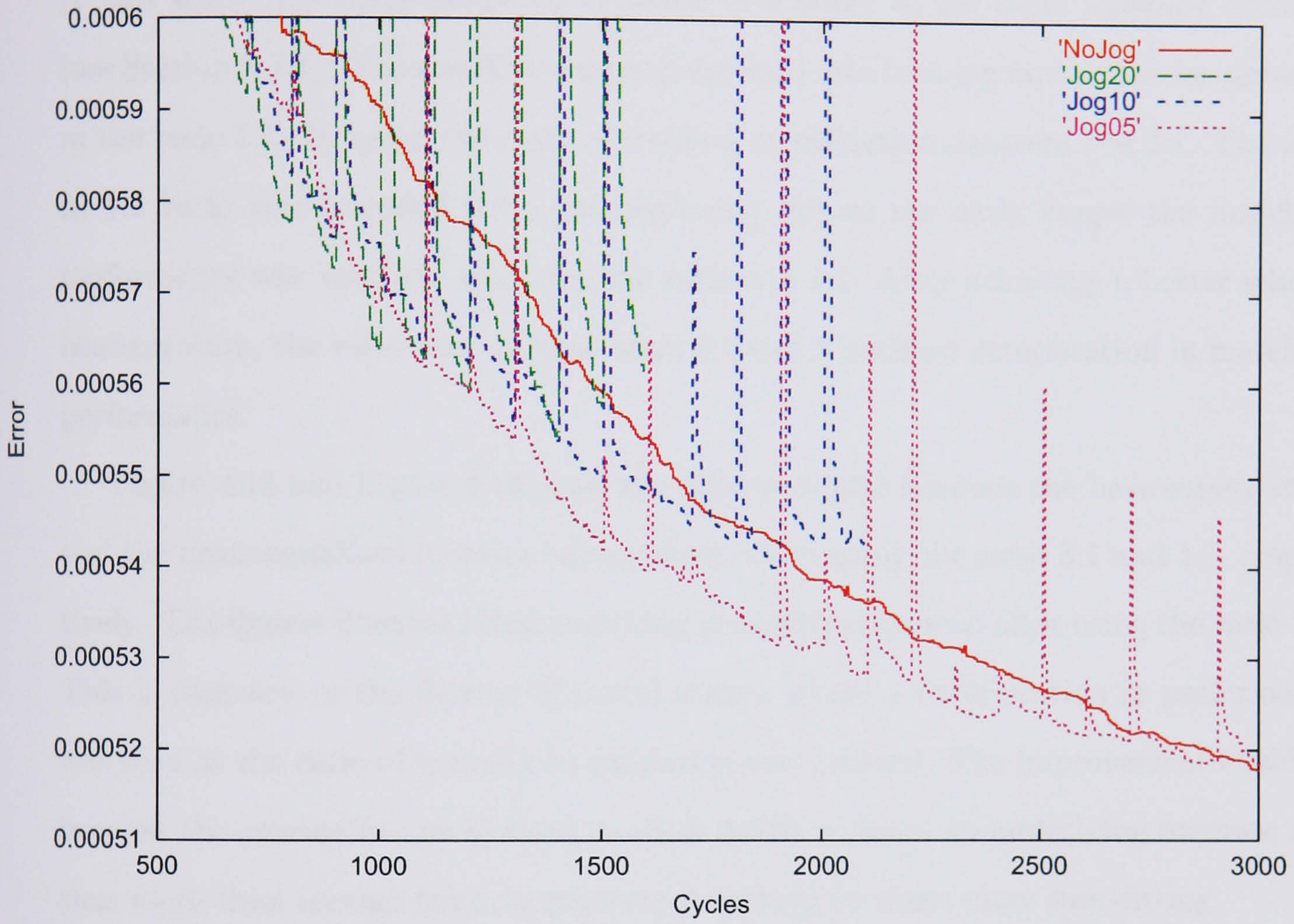


Figure 4.14: Error graph comparing the effect of weight jogging. An ANN is trained at four different instances as shown in Table 4.4. The graph shows the validation error progress during the networks training. Initially, jogging has a positive effect causing fast training. However, jogging slowly becomes a hinderance to error convergence and is thus overtaken by training where no jogging is applied. In addition to slowing the training, jogging might cause pre-mature stopping, i.e. training stops before reaching the lowest possible minimum. The effects explained are higher with higher amount of jogging.

any case, multiple weight initialization or weight jogging cannot easily overcome this. and are thus non-beneficial here.

4.5.3 Stopping criteria

In this study, the cross-validation criterion is selected as the main stopping criterion (see Section 2.3.3). The available patterns are split into training and validation datasets in the ratio 1:1. Initially, the ratio of training to validation datasets was 3:1. The shift to 1:1 ratio was intended from the beginning but at the early stages the matching performance was very low when the the ratio was 1:1. After achieving a better scheme configuration, the ratio was changed from 3:1 to 1:1 with no deterioration in matching performance.

Figure 4.15 and Figure 4.16 show the difference plot between the base survey stack and the cross-equalized monitor survey stack when using the ratio 3:1 and 1:1, respectively. The figures illustrate that matching is slightly enhanced after using the ratio 1:1. This is contrary to the finding of initial stages, where a deterioration in performance was seen as the ratio of training to validation was lowered. The improvement could be because the scheme has progressed to allow ANNs to learn an underlying function and thus more than enough training patterns only help to cause more over-fitting.

Practice showed that one stopping criterion could lead to an unnecessarily long training time. Hence, in addition to cross-validation, two other criteria were introduced to cause stopping: firstly, stopping can occur if the validation error (error when introducing the validation set) reaches a selected minimum, where the minimum is set very low (1.0×10^{-6}); and secondly, stopping can occur if training cycles, *iterations*, reach a certain high value, where the value is set very high (3000). The choice of the two parameters in the added criteria is made to ensure that stopping usually occur through cross-validation, but that training will stop if ANN managed to map very well or if no minimum was reached after a long training time. Those cases that get stopped by other than cross-validation can be distinguished and re-examined after training.

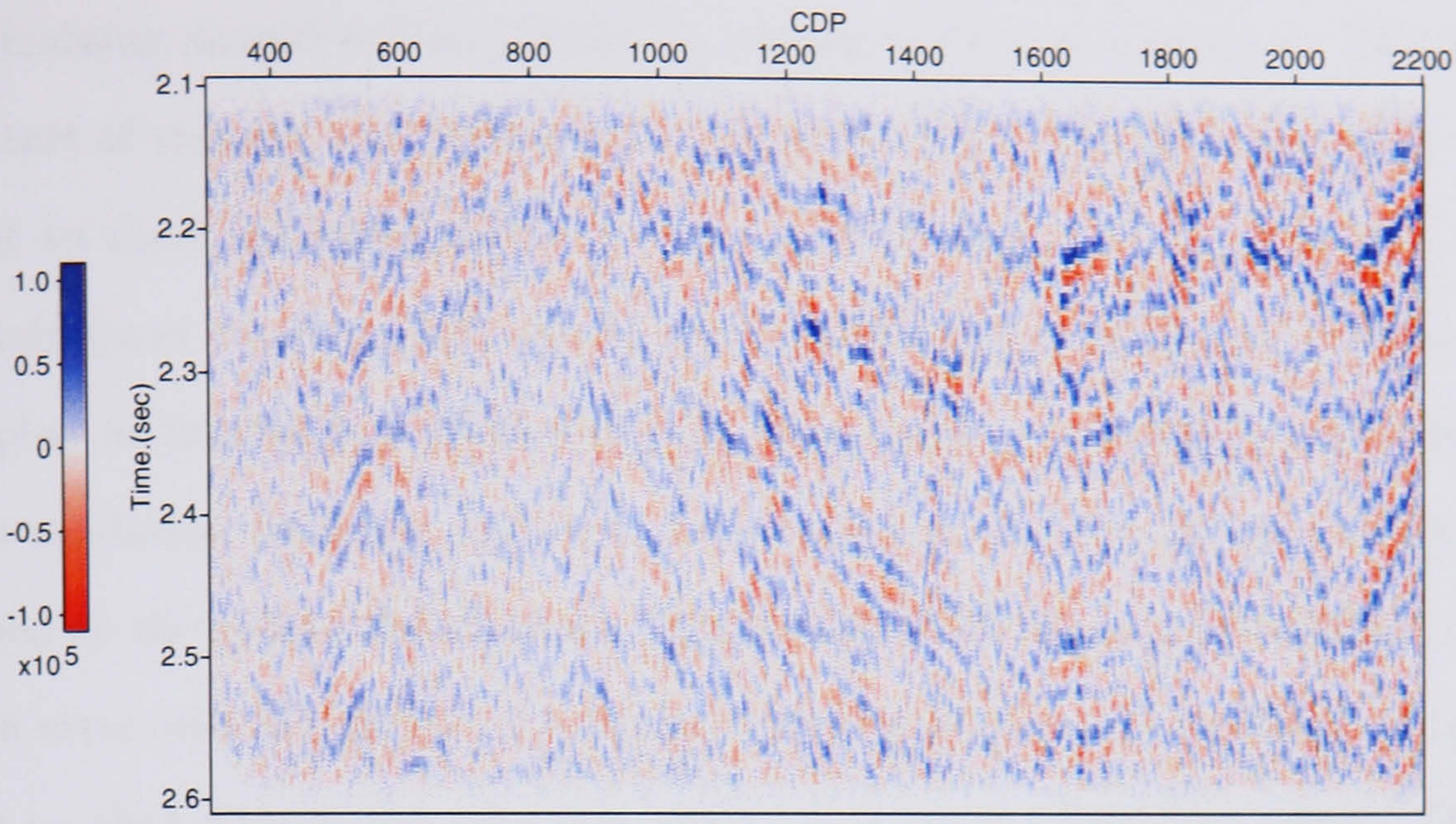


Figure 4.15: Difference plot — between the base survey stack and the stack of the cross-equalized monitor survey — when using a 3:1 ratio of training to validation patterns. In comparison with Figure 4.16, plotted to the same scale, this produces a slightly inferior result of matching.

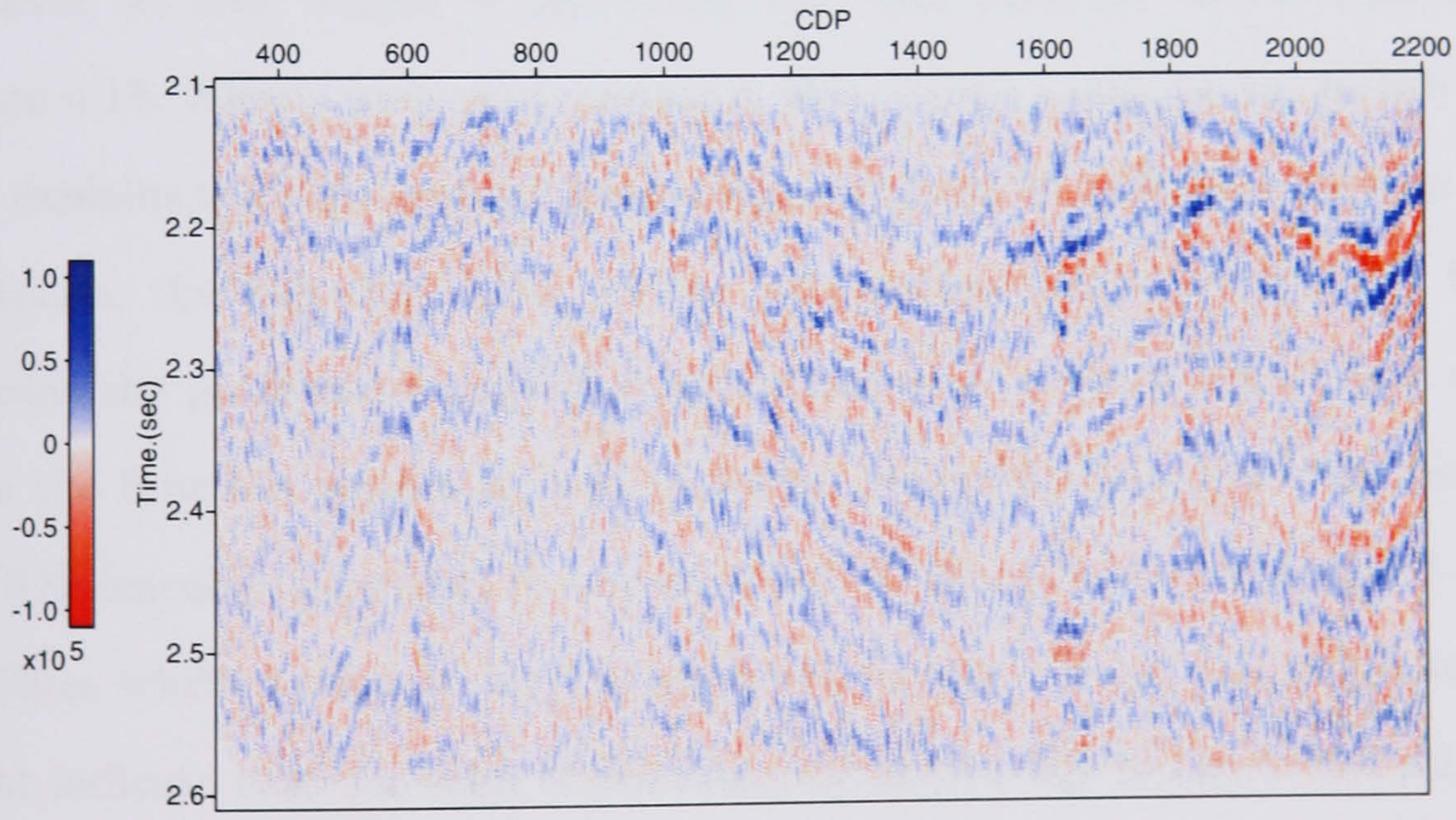


Figure 4.16: Difference plot — between the base survey stack and the stack of the cross-equalized monitor survey — when using a 1:1 ratio of training to validation patterns. In comparison with Figure 4.15, plotted to the same scale, this produces a slightly superior result of matching.

In addition to the above, the stopping criterion defined includes a few other criteria. The training should not stop before a certain minimum iterations. This is because at the start of training the behaviour of the error is not very stable. As the weights come closer to their optimum values a more smooth decline in error is seen. In general, the minimum is set to 300 iterations. In addition to setting a minimum iterations, stopping is defined to continue 200 iterations after reaching the point defined by the cross-validation stopping criterion. The trained network is actually saved at the point defined by the cross-validation stopping criterion, and training continues solely to check if the error will fall again. Therefore, this ensures that the increase in the validation error — that defines the stopping point — is not a temporary event. The sudden or temporary rise in validation error could occur if weight jogging is introduced, and is also seen when using the standard back-propagation training algorithm and its modified version back-propagation with momentum.

Figure 4.17 illustrates a graph of the typical error progress during training. The trained network is saved at the point defined by the cross-validation stopping criterion. However, at later stages of this study the error progress follows that illustrated in Figure 4.18. Firstly and most probably, this change could be due to the same reason that explains the improvement when a lower training-to-validation pattern ratio is used. Therefore, the fact that ANN, at the later stages, learns (acquires) a function also explains the progress of validation and training error which follow each other closely, since the function applies to both datasets. Secondly, this could also be because the RPROP learning algorithm has reached the vicinity of the minimum and subsequently oscillates whilst trying to hit the minimum exactly. Thus, the graph in Figure 4.18 might indicate that the error is behaving asymptotically to the actual minimum. This raises the question: is it useful to train for long time if the error is going to decrease ever more slowly?

The experiment that produced the graph in Figure 4.18 was repeated, but the trained network was saved at different stages of training. In addition, training was

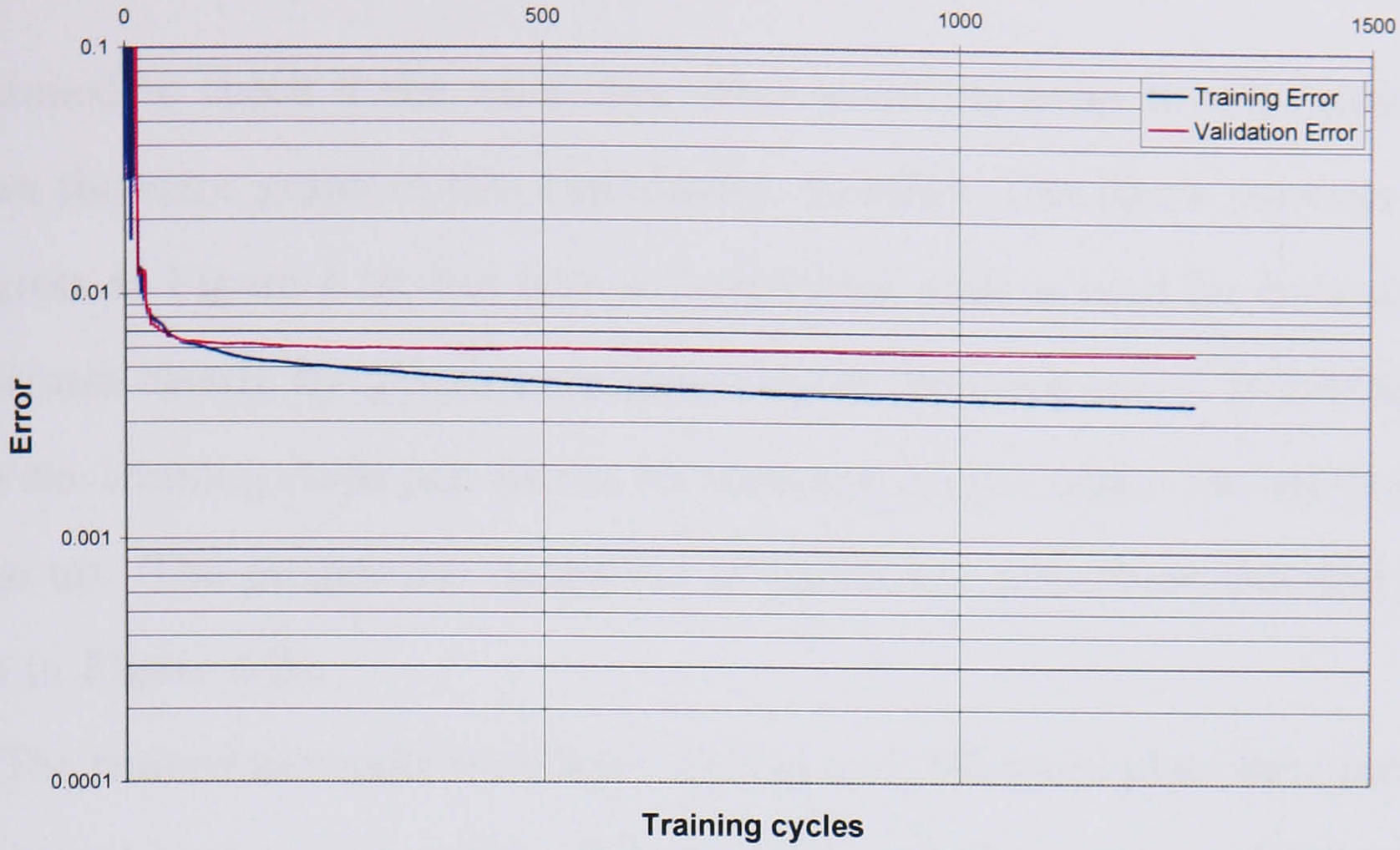


Figure 4.17: A typical graph of error progress during training in the early stages of this study. The error (misfit) is plotted in a logarithmic scale versus training cycles plotted in a linear scale. Both training and validation error drop significantly after the first 100 cycles and then the drop becomes more gentle. The validation error slowly departs from the training error reaching a minimum before starting to rise, and thus indicating that the network is beginning to over-fit the training dataset. Here, the validation reaches the minimum at 1085 training cycles and training is continued for additional 200 cycles to ensure that the increase in the validation error is not temporary.

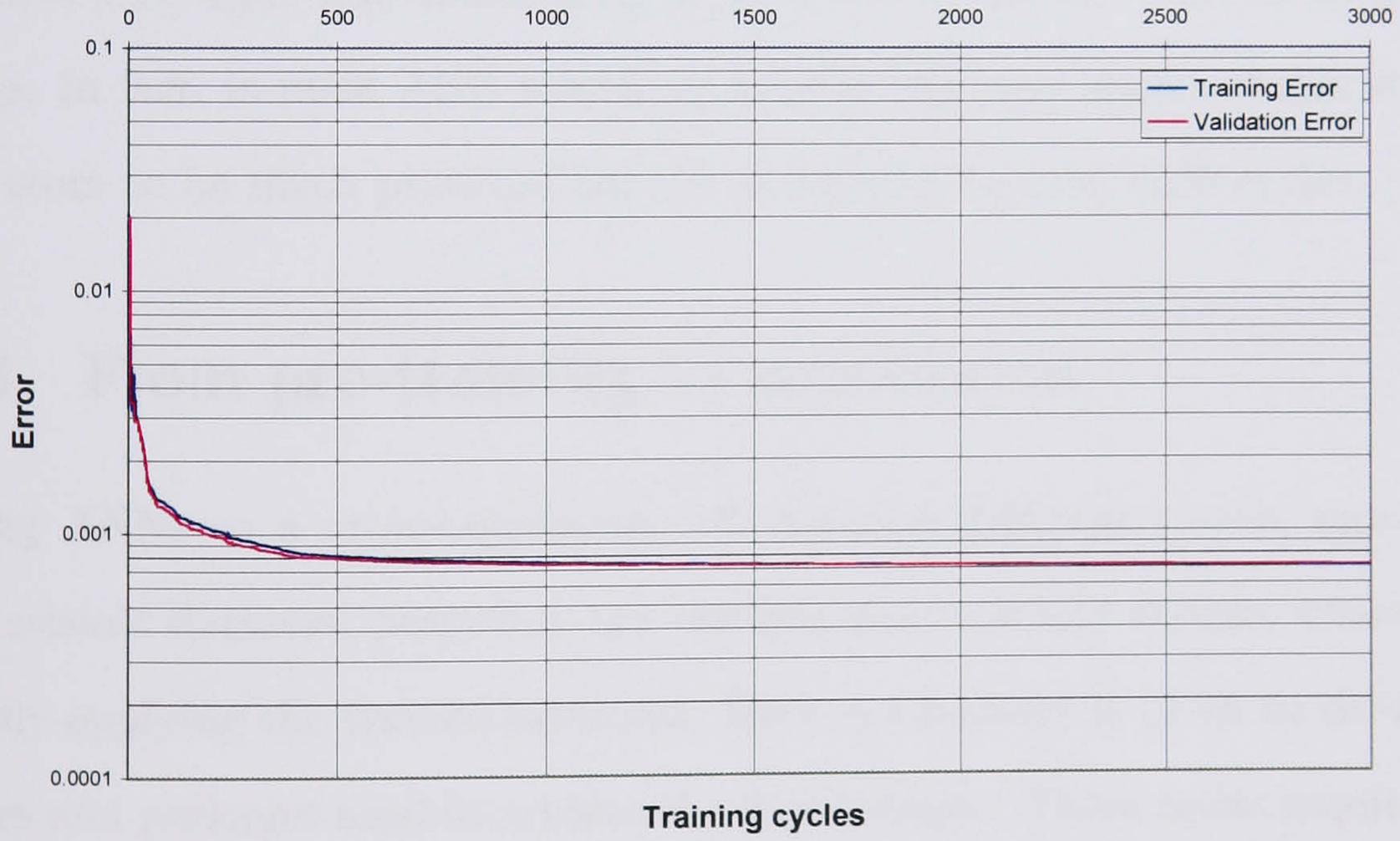


Figure 4.18: A typical graph of error progress during training in the later stages of this study. The error is plotted in a logarithmic scale versus training cycles plotted in a linear scale. Both training and validation error drop significantly at the first 100 cycles and then the drop becomes more gentle and ever more slow. The validation error follows very closely the trend of the training error. The training is stopped after 3000 training cycles, which is the set maximum.

continued to check if the validation error would increase at some point. Figure 4.19 shows the error graph of this experiment. In effect, this figure presents the same error progress as Figure 4.18, but here a logarithmic scale is used for both axes. The figure illustrates clearly the progressive flattening of the error curve. In addition, we observe that the training stops just before 16 thousand cycles, where the validation error starts to go up. The progressive departure of validation error from training is more clearly seen in Figure 4.20.

The trained networks were later applied and difference plots were produced between the actual base survey gather (Figure 4.21) and the cross-equalized gather produced by the trained networks from the monitor survey. These difference plots are arranged to illustrate the progressive error drop in matching as more training cycles were used. The plots are shown in Figures 4.22–4.29.

From Figures 4.22–4.29 one sees little changes after around 500 cycles. This is also apparent from Figure 4.19. However, one cannot predict accurately, before looking at Figures 4.22–4.29, how much drop in the error curve will amount to in the difference plots. In fact, in most ANN matching done in the later stages of this study there does not seem to be much practical benefit of training beyond 1000 cycles.

4.6 From pre-training to application

Using ANNs as a cross-equalizing tool involves different stages, namely: preparing the seismic datasets, preparing the training patterns and scripts, training ANNs, and finally applying the trained networks. Here, a summary is given to show the necessary codes and packages used to achieve the listed stages. These codes required a significant time and effort to establish and are thus considered a major part of this study. The aforementioned stages are followed by a stage of result analysis. Section 4.7 is devoted to the description of this latter stage.

The following sub-sections make use of structural charts to describe the different programs involved. These charts describe sequentially the processes involved and show

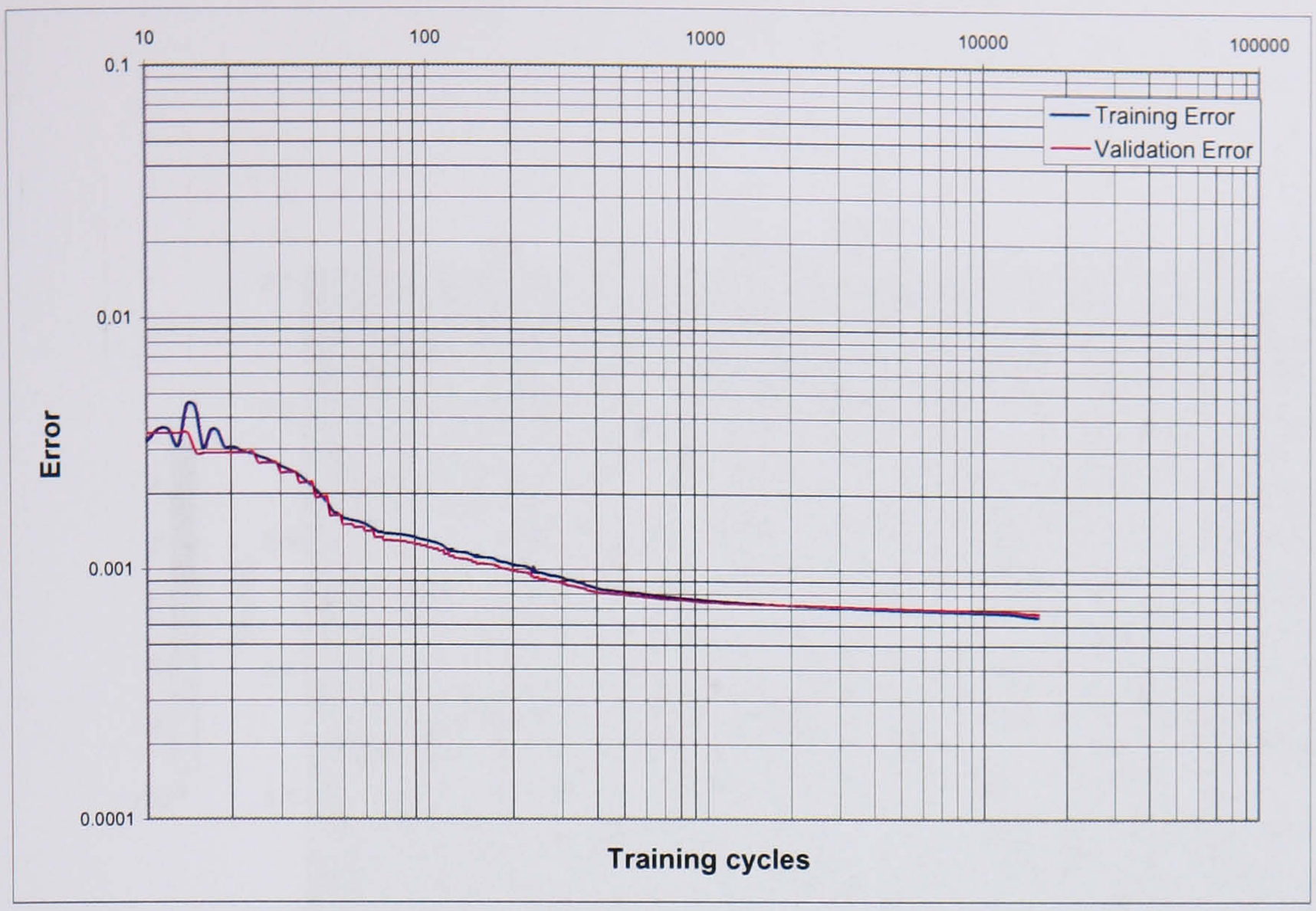


Figure 4.19: Graph of Figure 4.18 re-plotted with a different scale and training is continued further. The error is plotted in a logarithmic scale versus training cycles, which is also plotted in logarithmic scale. The sudden drop of the validation error at every fifth cycle is because validation is only done at every fifth cycle. Even with logarithmic scale for cycles the error slope is still seen to be progressively flattening. After approximately 400 cycles, the error becomes significantly flatter. Validation error starts to rise just before 16 thousand cycles.



Figure 4.20: Zoom on the final error progress shown in Figure 4.19 plotted with a linear scale. The validation error progressively departs from the training error until finally (after 15560 training cycles) it starts to increase and thus training is stopped. There is a fairly sudden dip in error after 13 thousand cycles and may be related to the stair-step nature of the error surface described in Section 2.4.1.

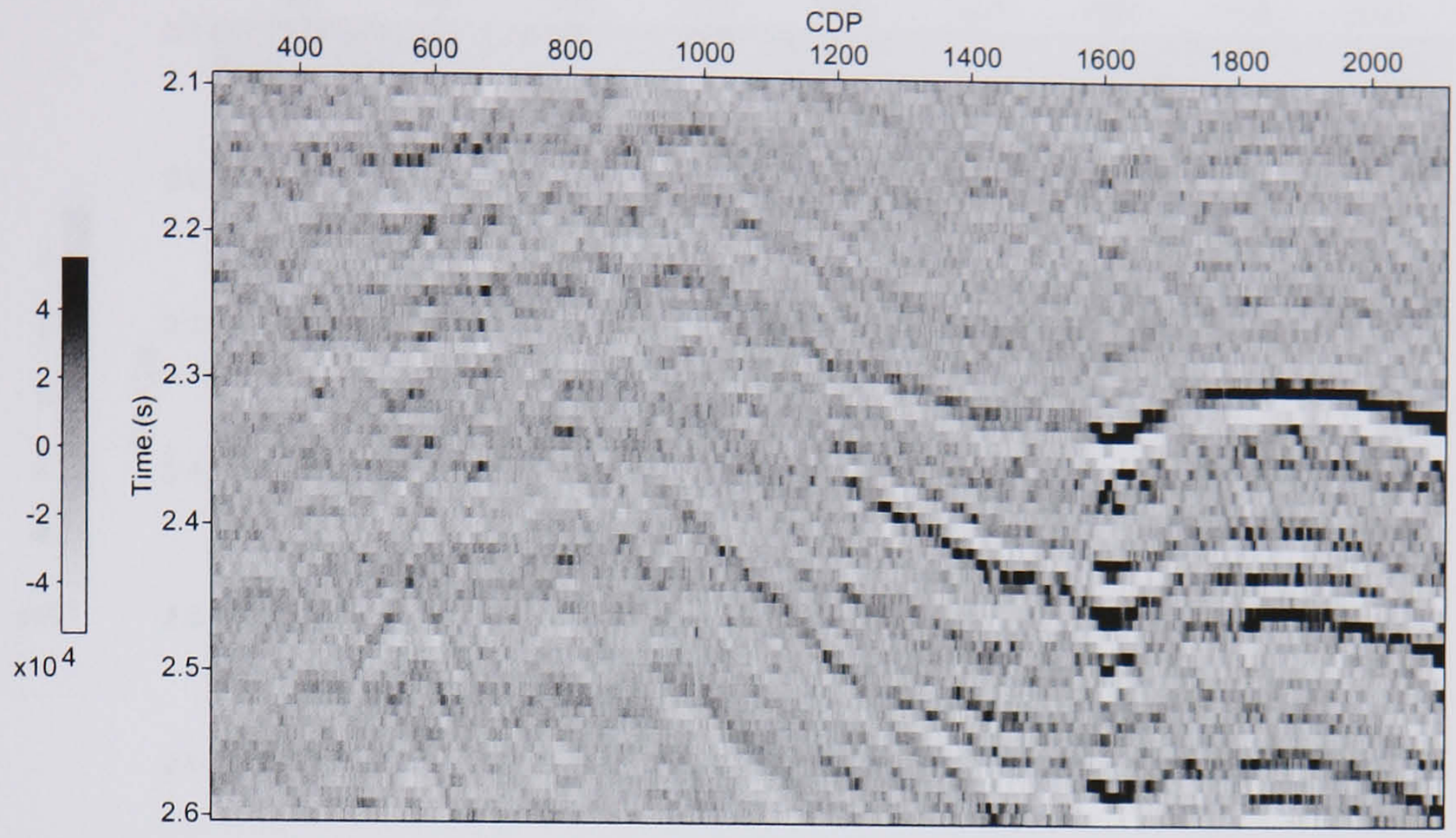


Figure 4.21: Offset gather (channel 121) of the base survey for comparison with Figures 4.22–4.28. Plotted to the same scale as Figures 4.22–4.28.

Figure 4.22: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 20 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. Here, the network has hardly learnt much and the error is large.

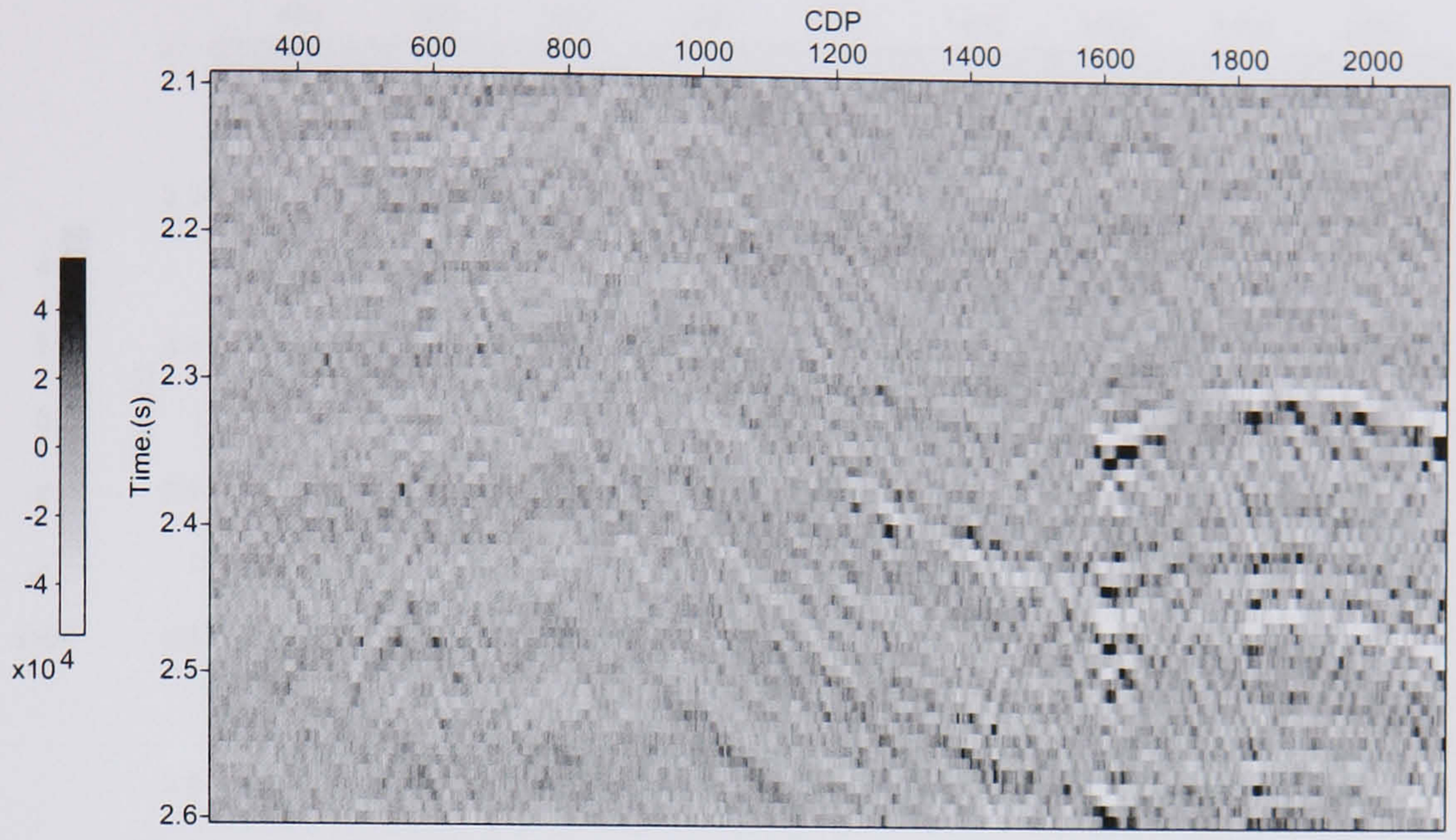


Figure 4.23: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 50 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is a significant drop in error when compared with Figure 4.22.

Figure 4.24: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 100 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is a significant drop in error when compared with Figure 4.23, but the drop is less than that observed between Figure 4.22 and Figure 4.23.

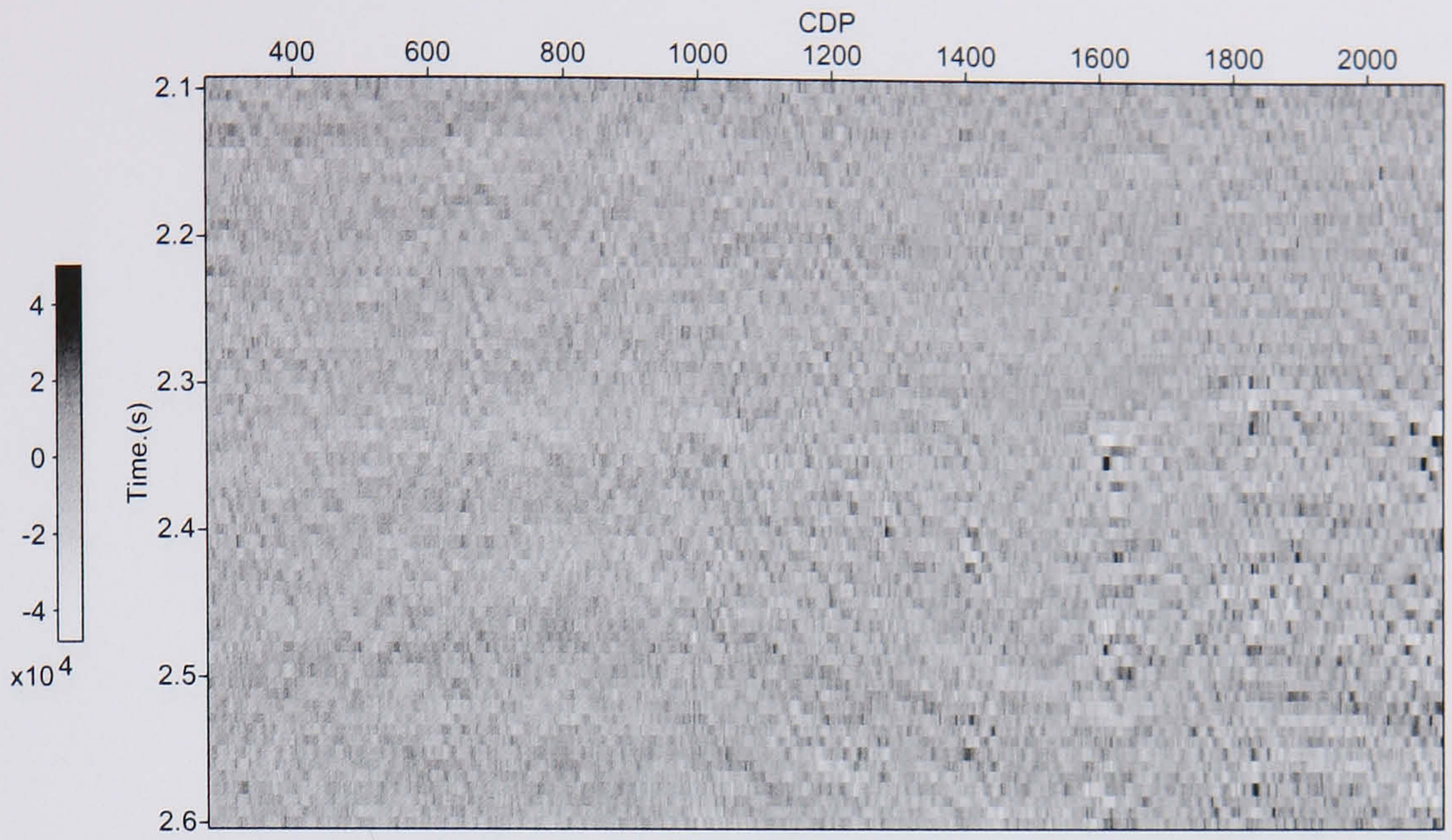


Figure 4.25: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 500 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is a significant drop in error when compared with Figure 4.24, but many more training cycles were needed to achieve this drop in comparison with that between Figure 4.23 and Figure 4.24.

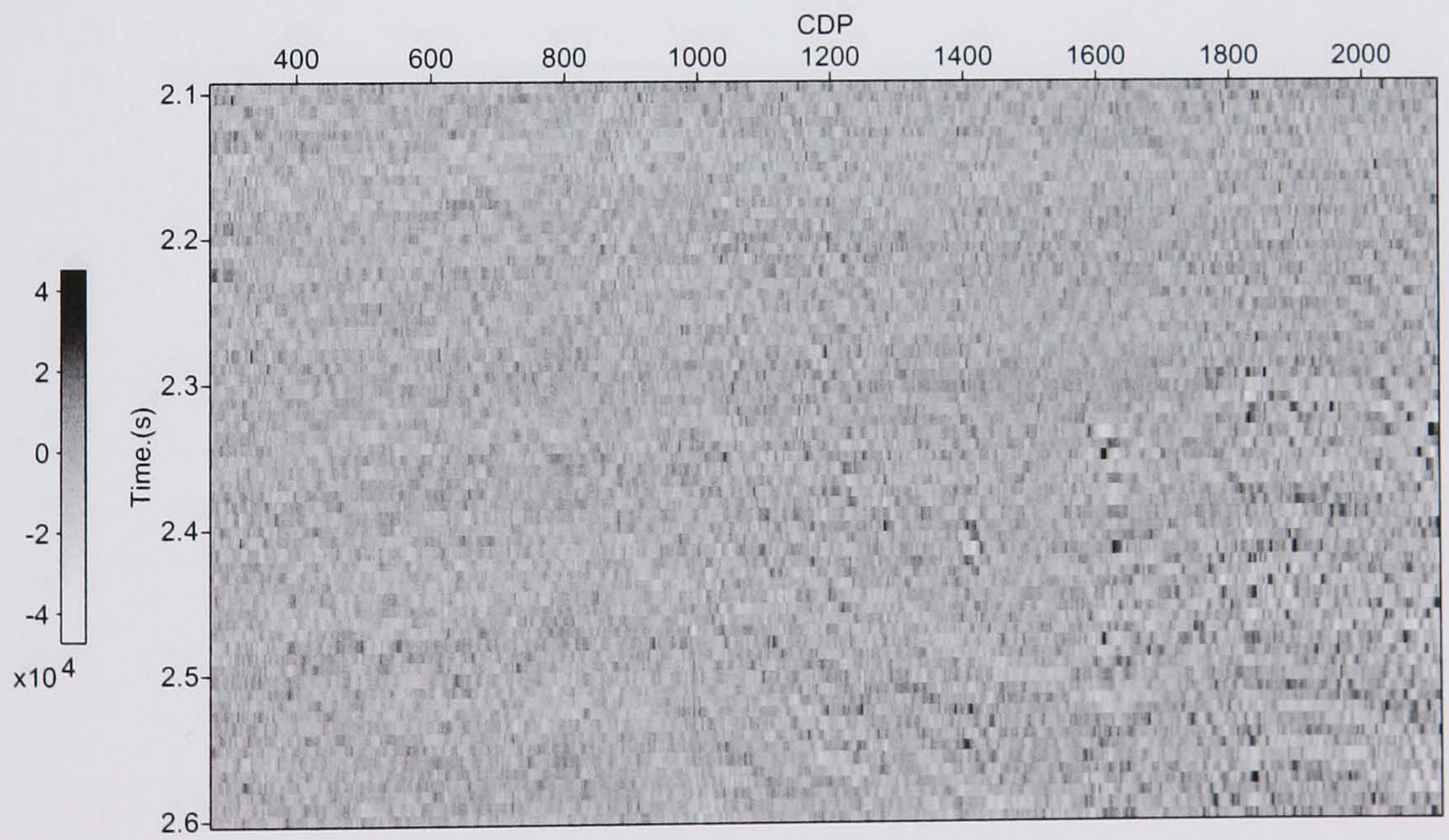


Figure 4.26: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 1000 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is hardly much drop in error when comparing with the last plot in Figure 4.25.

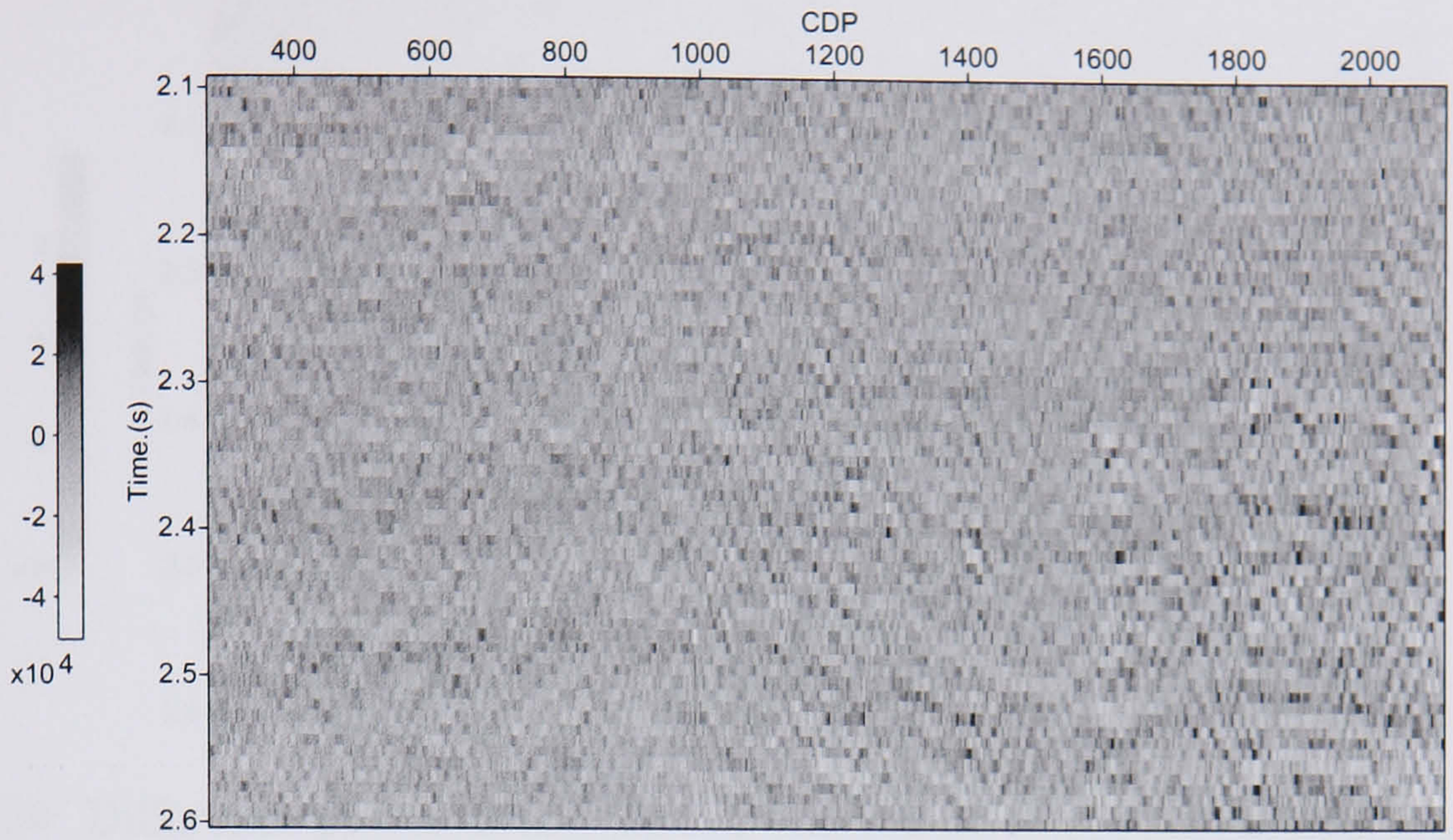


Figure 4.27: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 2000 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. Insignificant changes in error observed when compared with the last plot in Figure 4.26.

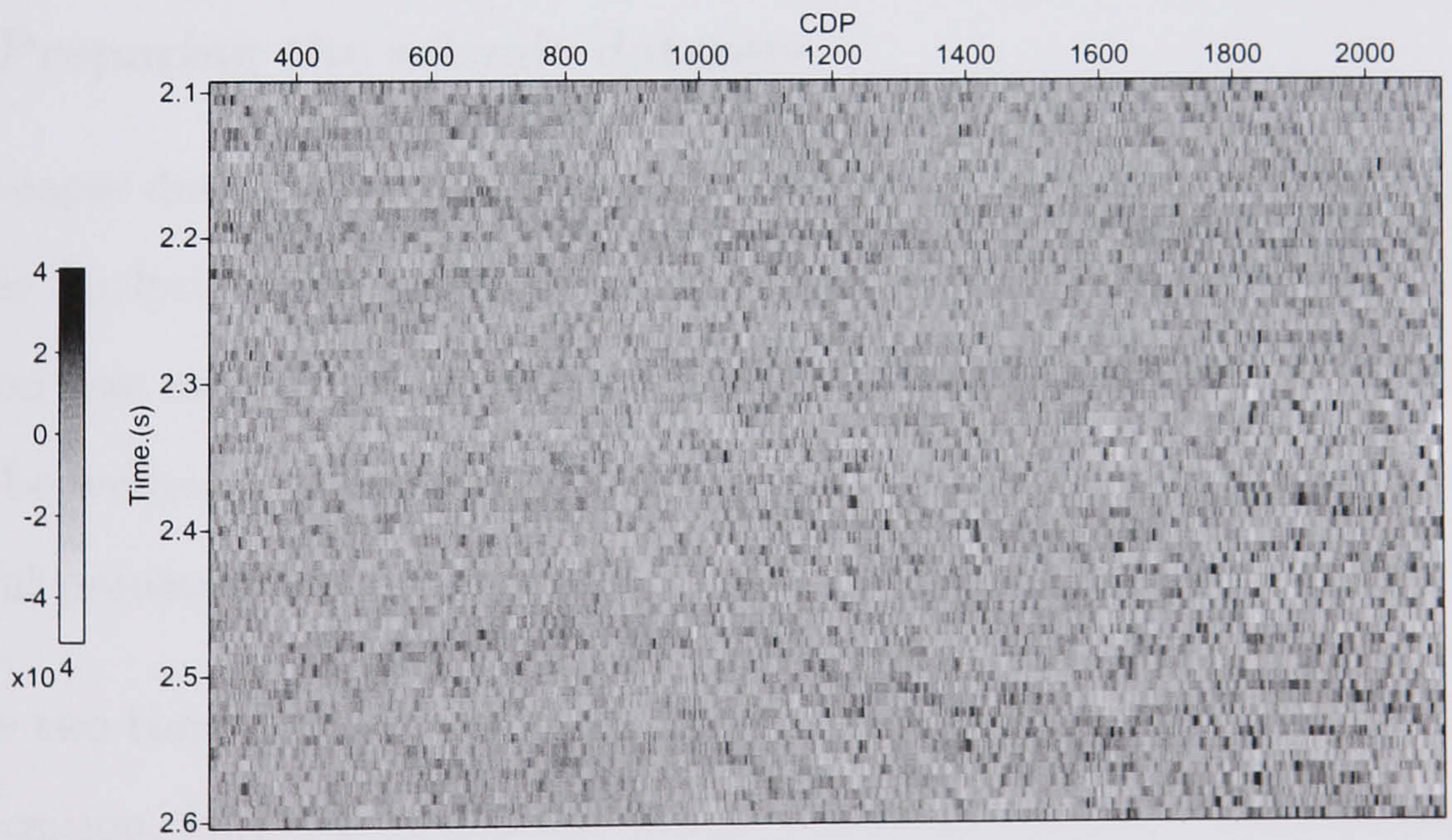


Figure 4.28: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather gather from the monitor survey after 3000 training cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is no visual changes in error observed when compared with the last plot in Figure 4.27.

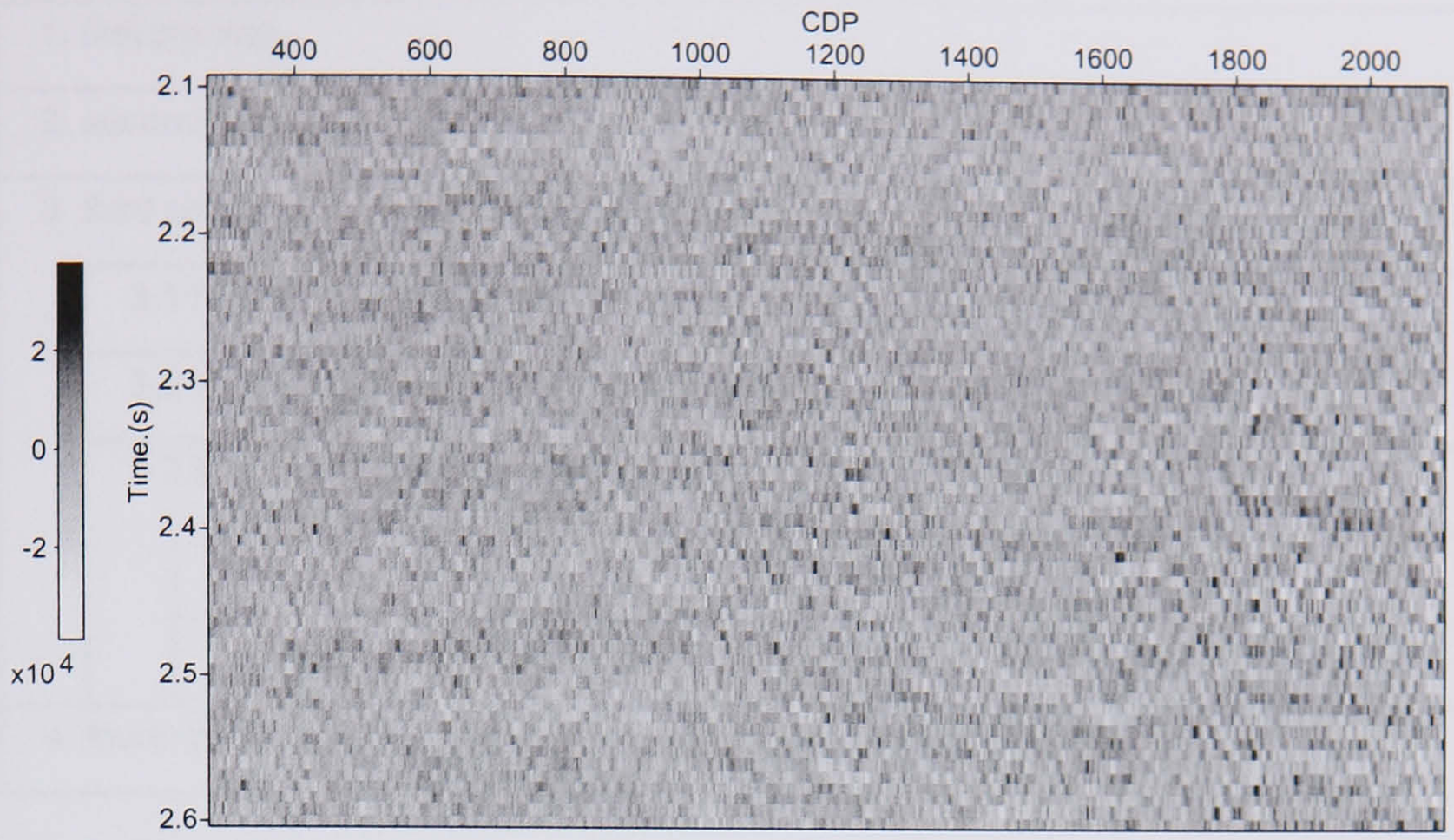


Figure 4.29: Difference plot between the base survey gather (channel 121) (Figure 4.21) and the cross-equalized gather from the monitor survey after reaching the lowest validation error after 15560 cycles. Plotted to the same scale as Figure 4.21 and all Figures 4.22–4.28 for easier comparison. There is no visual drop in error observed when compared with the last plot in Figure 4.28.

the relationship between sub-processes and their parent process. Figure 4.30 explains these charts and is provided for reference.

4.6.1 Preparing the seismic datasets

The time-lapse datasets are loaded into the processing package ProMAX. This package allows the analysis of the geometry and character of the two datasets. In turn, a description was established as given in Section 3.2.1. This description allows suitable codes to be written and the cross-equalization to be addressed in a suitable manner.

Several processes are undertaken to prepare the two time-lapse datasets:

1. The two time-lapse datasets are sorted in shot x-position, followed by the in-line y-position, followed by channel 1–240 (where 240 is the nearest offset channel). CMP (or CDP) is defined with the same value at the same x-position in both datasets. Only the common part (in x-position) of the two datasets is considered in this study.

2. A time delay correction of 51 ms is applied to the base survey. This follows

<table>
<tr><td colspan="2">1. first process</td></tr>
<tr><td colspan="2">2. second process</td></tr>
<tr><td colspan="2">3. third process (e.g. FOR EACH offset in the SEG-Y file)
<table>
<tr><td colspan="2">3.1 first sub-process of the third process</td></tr>
<tr><td colspan="2">3.2 second sub-process of the third process</td></tr>
<tr><td colspan="2">3.3 third sub-process of the third process (e.g. IF trace is r_m)
<table>
<tr><td>true</td><td>false</td></tr>
<tr><td>sub-process if 3.3 is true</td><td>sub-process if 3.3 is false</td></tr>
</table></td></tr>
</table></td></tr>
<tr><td colspan="2">4. fourth process (e.g. ReadData: reads data from the input file)</td></tr>
</table>

Figure 4.30: Description of structural charts. Here, each process is given a number to define their relative position, but in practice no such numbers are given. Processes are given sequentially from top to bottom. A parent process (like the third process) has its cell extended vertically so that all its sub-processes are indented to the right of the cell. When a process is an *if*-statement (like process 3.3), then the *true* and *false* sub-processes are defined as shown. When a process defines a function or subroutine within the main illustrated program (like process 4), then the process is given a name written in italic. The job of the function is explained after a colon. In addition, in some instances the function is described in full by showing all its sub-processes.

from comparison between several traces of the same CMP and offset from both time-lapse datasets.

3. A simple amplitude recovery is applied to the data. This makes sure the data are not biased by geometric spreading, causing some to seem less important to the ANNs because of their low energy.

4. A frequency band-pass filter is applied to the two datasets. The spectrum of the two datasets was determined to be very similar (Section 3.2.1). However, the band-pass filter ensures that the two time-lapse datasets have the same band-width, and no frequency aliasing is present.

5. The datasets are processed for spurious spikes. If these exist, then they are removed. In effect, this processing ensures that the datasets have no significant outlier data samples that can reduce the effectiveness of pattern normalization.

6. The time window 2100–2612 ms is selected for matching and analysis. This window contains the reservoir zone and sufficient data for matching and analysis. However, the selected time window is enlarged to twice the length, where the section of interest 2100–2612 ms is the middle part, i.e. the time section used is 1844–2868 ms. This choice allows for neighbouring input data, which consists of data from higher or lower times than the matched samples. The selection ensures also that the number of samples used N is a multiple of two ($N = 2^n$, where $n = 1, 2, 3, \ldots$), since this is necessary for fast discrete transforms to the wavelet or the frequency domain.

7. The selected data are re-sampled in time from 4 ms to 8 ms and are in some instances also re-sampled in space. This is to reduce the amount of data that has to be matched. In addition, the Nyquist frequency at 8 ms sampling is above the bandwidth that contains high energy in both datasets. An appropriate frequency band-pass is applied before this re-sampling.

8. The data required for pattern preparation are output from ProMAX as standard SEG-Y files. This output contains a file from the base survey that will provide the output patterns to ANNs and corresponds to the data from 1844–2868 ms from the in-line provided by the second streamer (higher in y-position). Another file contains data from the monitor survey that will provide input patterns to ANNs. This file include the time window 1844–2868 ms from the three in-lines surveyed by the three streamers.

4.6.2 Preparing the training patterns and scripts

Preparing training patterns requires efficient and flexible access to the seismic files and the data within. In this study, this is achieved through the use of the standard SEG-Y format (Barry et al., 1975) as a format to write the seismic files and the use of ANSI C++ codes ("IC" codes), written by Alberts (2002), to access them. The codes define C++ object classes (Stroustrup, 2000) for the SEG-Y file and traces within.

They enable easy access to all the headers and data samples through functions defined and associated with the classes. A few modifications were introduced to Alberts (2002) original class definitions. In particular, the classes are modified not to allow the reading to memory of the whole SEG-Y file since this study deals with relatively large files. The modifications enable a SEG-Y class to be initialized by a SEG-Y file, from which it reads the line header and the first trace, and thus extracts sufficient information about the SEG-Y file. When a trace is required, the SEG-Y class can retrieve it from the file using the knowledge of its structure.

A code was written to prepare the training patterns. The code is split into two parts. Figure 4.31 shows the structure of the program used to create the required training patterns. On the other hand, Figure 4.32 shows the structure of the program used to write those patterns after normalizing them and splitting them into validation and training patterns; write the necessary network (or networks with different topologies if one is interested in testing the effect of topology); and write the batchman training script.

4.6.3 Training ANNs

Training of the ANNs is done within a cluster of 16 computers. Computer clusters are employed in many similar studies to enable processing of many tasks in a shorter time. In general, clusters enable both independent fully parallel processes and those which require some information sharing. In this study, each ANN training is fully independent of the others. Therefore, each network is trained as a stand-alone process in parallel to other networks' training.

A script is prepared to run network training from the master (server) computer in parallel on the 16 computers. The script passes sequentially all the training jobs to the client computers as soon as one of them becomes available and is not processing a previous job. Each of the training jobs is designed to train one ANN and runs training as defined on the *batchman training script*.

initialize the project name: this name is a prefix to all related files

read the input to the program and initialize the variables,
e.g. Line89 and Line92 are the SEG-Y files of the base and monitor survey respectively

FOR EACH shot in Line89

FOR EACH offset (channel) to be matched

calculate the position of trace r_b to be matched in Line89 (aIndex89)

read trace r_b = Line89[aIndex89],
i.e. the trace number "aIndex89" in SEG-Y of the base survey "Line89"

save the variable aIndex89 for later use in a file holding all traces matched within this offset

calculate position of trace r_m, in Line92 – this is an approximation

FindNearestInput: find exact position of r_m= aIndex92

read trace r_m to be trace aIndex92 in Line92

FOR EACH sample in trace r_b (or wavelet coefficient)

define the output pattern to be the sample in the output trace r_b

define an intput pattern container (aInputPattern)

add geometry information to aInputPattern

ChooseInputPatterns: select the input to the network

FOR EACH trace defined as input traces

calculate the position of the trace
(given the streamer, gather type, and position within gather)

read the trace

IF trace is r_m

true	**false**
read as input 13 samples and add to aInputPattern	read as input 5 samples and add to aInputPattern

write the output and input patterns in a file allocated
for this offset, its sub-group (e.g. scale in wavelet domain), and sample

Figure 4.31: Structural chart of the program used to prepare training. *FindNearestInput* function works by finding a trace r_m within the monitor survey with the nearest shot-x position followed by the nearest offset position to trace r_b from the base survey.

initialize the project name: this name is a prefix to all related files

read all the offsets required to be matched

FOR EACH offset

MakeFileChoice: for each trace in the offset gather allocate a choice of where the patterns from it should be written (validation or training)

FOR EACH offset sub-group (e.g. scale in wavelet)

define aPatternSet: a container for all patterns related to the sub-group.

FOR EACH sample

read all the patterns related to this sample and append to aPatternSet

WritePatterns: write all the patterns in aPatternSet after normalizing

calculate statistics of the patterns (to enable normalization)

prepare two output files: one for validation patterns and another for training patterns

FOR EACH pattern

normalize and write to either of the two files according to the choice made in *MakeFileChoice*

FOR EACH topology analysed

MakeNet: write a network file according to the specified topology

MakeRpropBatchmanFile: write a batchman file for training the network specifying the training parameters and required files

write scripts for running the training specifying all required files

Figure 4.32: Structural chart of the program used to write all necessary files for training. The files include the training and validation pattern files, the network to be trained, and the batchman training script.

The batchman training script is written in the simple coding language provided by SNNS (see Section 2.4.7). The batchman script I wrote defines not only the training algorithm and stopping criteria, but also outputs sufficient information to analyse the training and log the error progress. Figure 4.33 shows a structural chart of how the training script is defined.

4.6.4 Applying the trained ANNs

The trained ANNs are applied to the whole input patterns to produce the cross-equalized dataset. Figure 4.34 shows the structural chart of the program written for the application of the trained ANNs.

4.7 Quality control (QC): analysing the results

The trained networks and their cross-equalization result can be analysed in several ways. These include an initial qualitative and mainly visual analysis and a further quantitative analysis. This section describes the two.

4.7.1 Initial qualitative analysis

The mean square error (MSE) between the actual and target outputs of the network is an initial indicator of its performance. The validation error or $\text{MSE}_{\text{valid}}$ (MSE of all the validation dataset passed through the trained network) is particularly analysed. The result is further analysed by applying the networks and inspecting the seismic offset gathers and the difference between these and the gathers of the base survey.

Usually, the analyses mentioned above are sufficient to filter out the least appropriate designs. For a more thorough analysis, ANNs are trained and applied to a range of offset to enable processing and stacking. After stacking using the flow shown in Table 3.2, the stacks are analysed, and the difference between those and the equivalent stacks from the base survey are analysed and compared. The stacks and difference plots enable a good visual estimate and comparison of the matching performance.

initialize the data: file names

initialize the training parameters

load initial network and pattern files

initialize the network and the SNNS simulator

REPEAT

train the network on training patterns

FOR every validation interval (e.g 5 cycles)

test the network on the validation patterns

IF validation error is lower than lowest recorded: **then** save the network

assess the stopping criteria

print the training and validation error

UNTIL stopping criterion is satisfied

print the lowest validation error, number of cycles required, and training time

Figure 4.33: Structural chart of the network training — programmed in SNNS's batchman language.

4.7.2 Further quantitative analysis

In addition to the previous visual and mainly qualitative analysis, it is also appropriate to provide a few quantitative measures of seismic matching. Three quantitative measures are defined: predictability, normalized root mean square (NRMS), and residual root mean square (RRMS). Predictability and NRMS are discussed in length by Kragh and Christie (2001, 2002) as a measure of repeatability between two seismic time-lapse datasets. The third measure (RRMS) is a more intuitive difference between two time series in terms of root mean square (RMS).

NRMS and predictability

Given two traces or discrete time series, r_1 and r_2, then the normalized root mean square (NRMS) is defined as:

$$\text{NRMS} = \frac{200 \times \text{RMS}(r_1 - r_2)}{\text{RMS}(r_1) + \text{RMS}(r_2)}. \tag{4.3}$$

initialize the project name: this name is a prefix to all related files

read all the offsets to be matched

FOR EACH offset

read the index numbers, within the base survey SEG-Y file,
of all the traces to be matched (usually the entire offset gather)

FOR EACH topology analysed

create a map object to store for each associated trace its
cross-equalized samples

FOR EACH offset subgroup (scale in wavelet domain)

initialize variables to hold normalization parameters

create a function of trained network (using snns2c) and link it to the
main program

FOR EACH sample in offset subgroup

ApplyNetwork Function

FOR EACH trace within the offset

read associated patterns

pass patterns through the linked network function

denormalized the output

save the output in the trace-collection map

WriteTraces Function

FOR EACH trace in the trace-colection map

create an empty trace

copy the trace header from the base survey SEG-Y file

fill the trace with the cross-equalized samples

append the trace onto a file that will hold all the traces
associated with the offset matched and the topology

Figure 4.34: Structural chart of the application of the networks after training.

The root mean square (RMS) operator is defined as:

$$\mathrm{RMS}\big(x(t)\big) = \sqrt{\frac{\sum_{t_1}^{t_2}\big(x(t)\big)^2}{N}}, \tag{4.4}$$

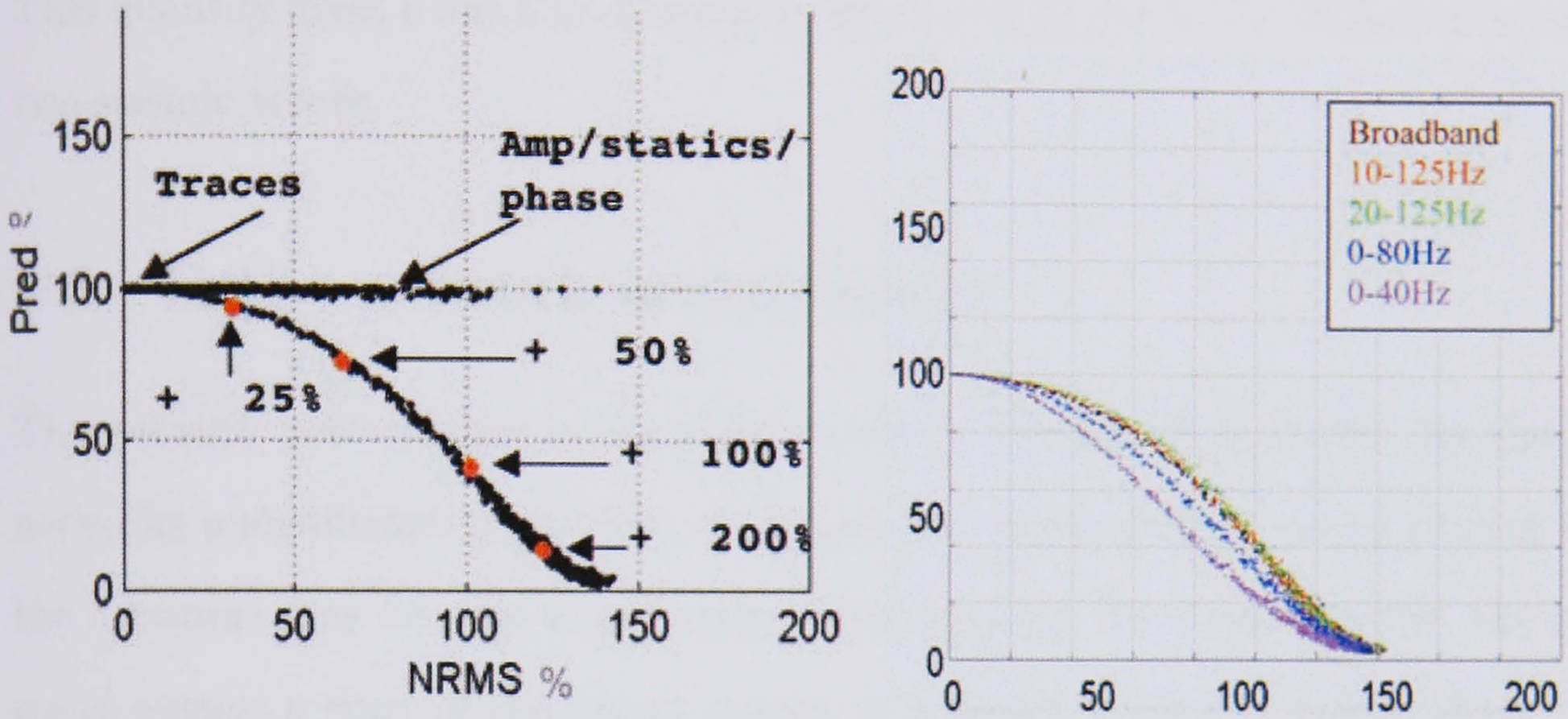
where N is the number of samples in the time interval t_1–t_2. The values of NRMS are not intuitive and are not limited to the range 0-100% (Kragh and Christie, 2002). For example, if both traces contain purely random noise, the NRMS value is 141%. If both traces anti-correlate (i.e., 180° out of phase, or if one trace contains only zeros) the NRMS error is 200%, the theoretical maximum. If one trace is half the amplitude of the other, the NRMS error is 66.7%.

The predictability is a measure of how much a series can be predicted linearly from another.

$$\text{Predictability} = 100 \times \frac{|\psi_{12}(\tau)\cos(\theta) + \psi_{12}^{*}(\tau)\sin(\theta)|}{\sqrt{\psi_{11}(0)\psi_{22}(0)}}, \tag{4.5}$$

where ψ_{12} is the cross correlation between the two time series r_1 and r_2, and ψ_{12}^{*} is the Hilbert transform of ψ_{12}. ψ_{11} and ψ_{22} are the auto-correlation of r_1 and r_2 respectively. In addition, τ is the optimum time shift, and similarly θ is the optimum phase shift (see Section 1.2.2). The definition in Equation 4.5 is similar to both Thomas (1999) and Kragh and Christie (2002). However, I followed the definition given by Al-Mazroey (2002) primarily because the phase misfit θ terms, which remove the phase dependency of predictability, are explicitly shown in the equation.

Predictability is sensitive to random changes between the two time series and noise, but insensitive to linear misfits (amplitude, time, and phase). Conversely, NRMS is extremely sensitive to the smallest changes in the data. Figure 4.35 illustrates how predictability and NRMS vary with both noise and linear misfits. Plotting predictability versus NRMS, as in Figure 4.35, has the advantage of being able to pick areas that depart from the ideal curve due to linear misfits. It is also informative in that the plot moves rightward due to linear misfits and both downward and rightward due to non-linear misfits.



(a) (b)

Figure 4.35: Effect of variation between two seismic sections on predictability and NRMS. Both plots show predictability versus NRMS between two seismic plots. (a) shows the limits of the plot and causes of shifts. Two identical traces plot at 100% predictability and zero NRMS, i.e. at point labelled "Traces". The position of the plot after adding random noise to the traces is shown by the points labelled with the noise percentage, e.g. "25%" and "50%". The position of the plot after introducing a linear amplitude, time, or phase misfit is labelled "Amp/statics/phase". Therefore, addition of a certain percentage of random noise to otherwise identical traces moves the plot down along a bell-shaped curve. On the other hand, amplitude, time and phase misfits (linear misfits) cause no change in predictability but increases NRMS causing a rightward shift. In summary, predictability versus NRMS plot moves rightward due to linear misfits and both downward and rightward due to non-linear misfits. The position of the leftward limiting curve is bandwidth dependent as shown by the synthetic modelled data in (b). Figures from Kragh and Christie (2002)

Rrms

The final quantitative measure of matching is the residual root mean square (RRMS). This is defined as:

$$\text{RRMS} = 100 \times \frac{|\text{RMS}(r_2) - \text{RMS}(r_1)|}{\text{RMS}(r_1)} \tag{4.6}$$

This quantity gives a more intuitive quantitative measure to the difference between any two seismic traces.

4.8 Is the network memorizing?

The network performance in matching should be looked at carefully. By showing the networks a significant proportion of the seismic target output values during training, the networks may be able to memorize these outputs. In another words, the networks could output a copy of the target output it is shown during training, where the copy is only hindered by how much information the topology could hold. The network is not desired to replicate the base survey, but to learn how to normalize the acquisition difference.

In theory, the chance of the network only memorizing the output is lowered significantly by using the cross-validation stopping criterion. In addition, the topology selected is approximately the smallest that can perform the mapping. However, to ascertain that the network is not memorizing and is actually learning a function, an experiment is conducted. This experiment makes one important alteration to the method of selecting input to the network described in Figure 4.5 — the input here is selected from a trace chosen at random from anywhere within the monitor survey. The experiment is conducted to match the offset gather of channel 121 of the base survey shown in Figure 4.21 from input selected from the monitor survey. The idea of this experiment is as follows: if the network memorizes the target output, then it should be able to memorize it regardless of the input, where the input is only used as a key to knowing what output is required. Therefore, if the network memorizes, we will get a matched

gather similar to Figure 4.21, which is the case when we use the method described in Figure 4.5.

The matched output of the network involved in this experiment is shown in Figure 4.36. This output is not at all similar to Figure 4.21 and is in contrast to the result when a realistic input is given (Figure 4.37). Therefore, the network is not memorizing.

The ability of the network to learn a function, other than relying on memorizing the output patterns shown during training, is further demonstrated in Chapter 6.

4.9 Record: list of conducted experiments

This section lists the main experiments that were conducted during the progress of this research and prior to the writing of this thesis. Some of these experiments had interesting and useful results and were therefore elaborated upon in this chapter or in the following few chapters. On the other hand, many proved only useful to justify a progress in one direction and not the other, but their usefulness is limited and are thus not described in much detail:

1. Test ANNs' ability to map a wavelet coefficient.

2. Test the effect of varying the training parameters of back-propagation with momentum and topology on the matching performance of ANNs.

3. Test ANNs' ability to map an offset gather (in the wavelet domain).

4. Test ANNs' ability to map a range of offsets (in the wavelet domain).

5. As in Item 4, but exclude CMP 1000–1350 from training to see the effect on the network's matching capability.

6. Design one ANN per scale (resolution level) instead of one ANN per coefficient. This enabled a box of data representing the reservoir to be excluded from training.

7. Perform a few linear cross-equalizations.

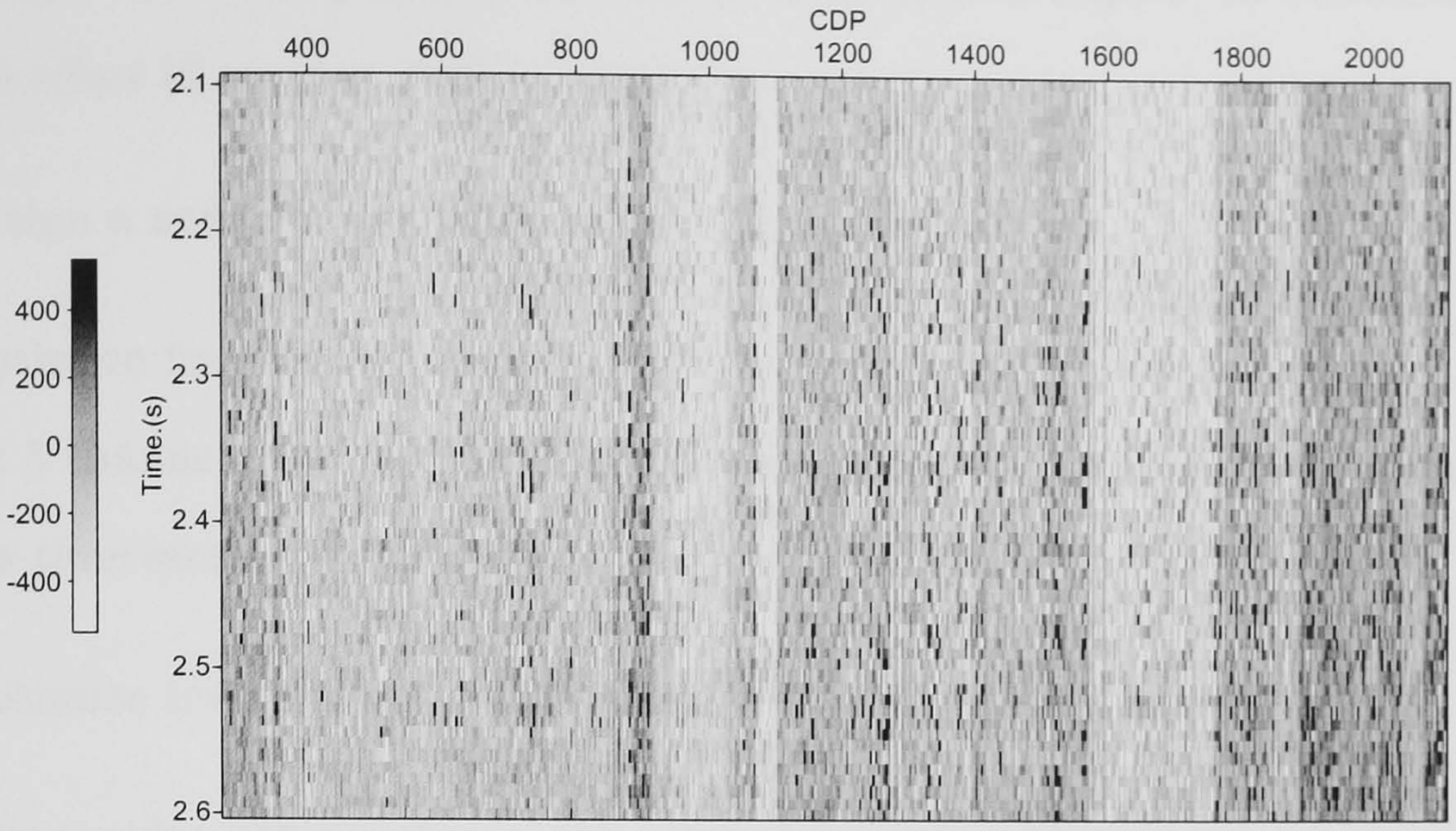


Figure 4.36: The output of a network trained to match the base survey gather of channel 121 (Figure 4.21) using random traces from the monitor survey as input. The network was not able to match the gather at all. Its output is close to zero induced by the network's tendency to reduce the error during training given that no deterministic relationship exists between the input and target output. This shows that the network is not able to memorize the target output shown during training, otherwise it should have produced an output similar to Figure 4.21. Compare the result here to that in Figure 4.37.

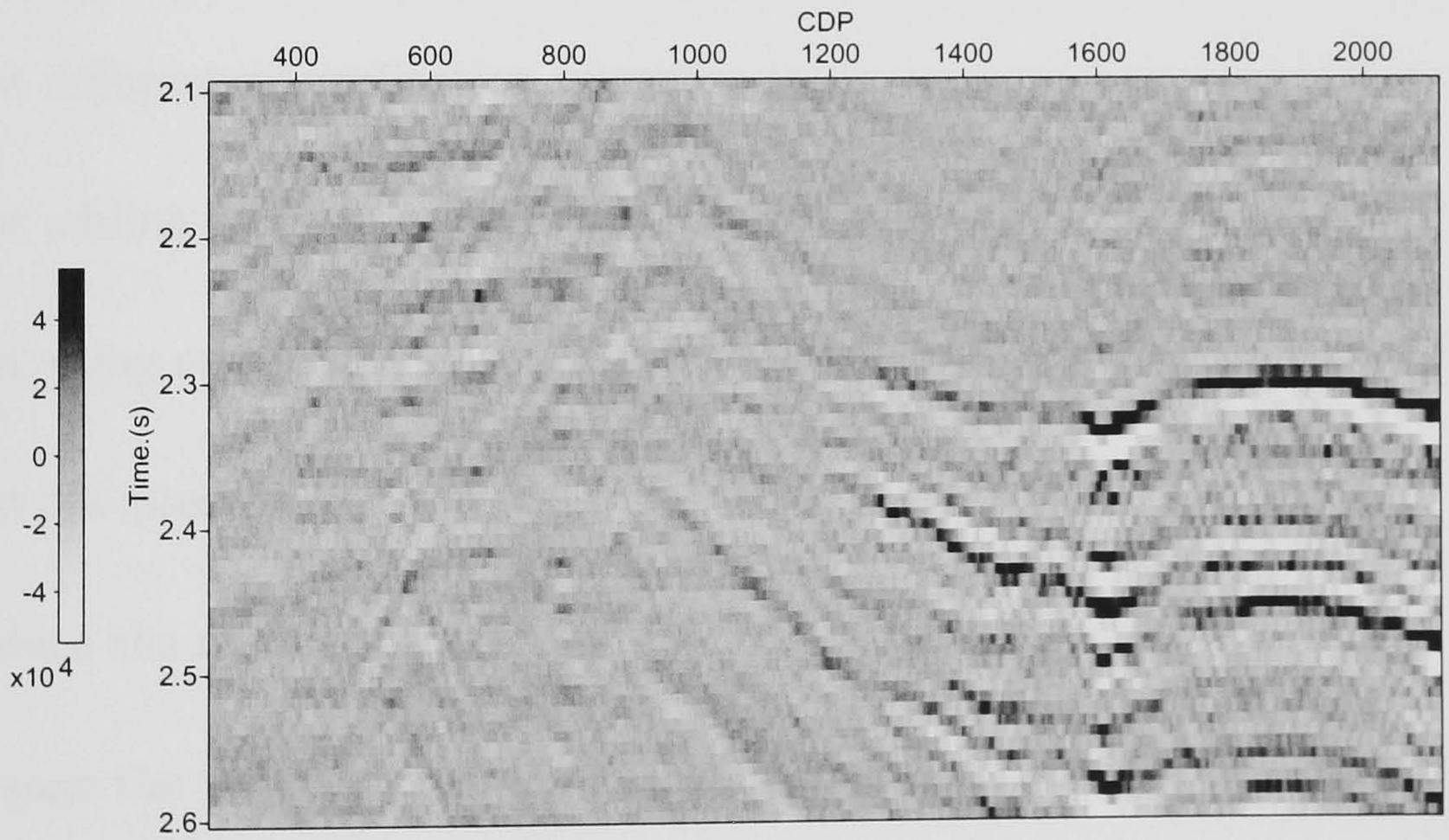


Figure 4.37: The cross-equalized gather (channel 121) from the monitor survey after training the network to match the base survey gather (Figure 4.21) using input as defined in Figure 4.5. Contrary to Figure 4.36, the cross-equalized gather here is very similar to Figure 4.21. There is evidently a deterministic relationship between the input and output data, albeit an unknown one.

8. Design one ANN per scale, but exclude no spatial region. In addition, compare the effect of varying ANN's topology.

9. Design a network per offset instead of per channel.

10. Optimize training parameters while matching in the wavelet domain. Test this for 3 channel gathers, 7 topologies, and 3 different training initializations (along the time section 2250–2762 ms).

11. Optimize training parameters as in Item 10 but in the time domain.

12. Test varying the number of the network's input parameters.

13. Compare cross-equalization in the time domain to that in the wavelet domain on a wider range of offsets to allow stacking and better comparison.

14. Repeat Item 13 but for the time window 2100–2612 ms.

15. Introduce a 24% anomaly to see the effect on the matching (in the time domain, while splitting the 512 ms matched section into four regions each with one ANN).

16. Test different normalization schemes.

17. Test adding geometrical information to the input.

18. Test using one ANN per 512 ms instead of four. This was not successful.

19. Test the preservation of a 10% anomaly.

20. Reduce the number of input parameters and test performance.

21. Change the way the input is defined to include a window of samples from several traces.

22. Analyse the connection weights of the trained network of Item 21. Input is successfully reduced by only including those inputs that the networks consider important in mapping to the target output.

23. Test different topologies with the new input definition, which was deduced in Item 22. The topology is successfully reduced without compromising the performance.

24. Test using two or one ANN per 512 ms instead of four. This is successful and performance is maintained.

25. Test splitting patterns equally between training and validation instead of the ratio of 3:1 respectively.

26. Test cross-equalizing with no introduced anomaly to try to find the best section with no ProMAX induced errors. The time window of interest is changed to 2096–2608 ms instead of 2100–2612 ms. This gets rid of problems associated with fractional statics not applied.

27. Test applying ANNs to match seismic on post-stack.

28. Test applying ANNs to match on offset-stacks.

29. Re-investigate and compare different normalization schemes.

30. Assess introducing weight jogging in the training algorithm.

31. Assess the difference between different runs of ANN cross-equalization. Results show that each run reach the same optimum matching result. The difference between the matching results is insignificant in comparison with the matching error of any one run, and is in the form of small amount of random noise introduced.

32. Assess splitting the inline into three sections, each matched by a different ANN. This gives similar result to that before splitting.

In addition to the aforementioned experiments, many major tests were re-conducted and improved upon to ensure that the effect of the different optimizations are incorporated. The tests include all those that will be shown in Section 5.4.1 and onward.

4.10 Summary

The cross-equalization of the two time-lapse datasets is addressed as producing a cross-equalized match to one in-line of the base survey from three in-lines of the monitor survey. Each offset gather is matched independently from the others to allow fast parallel implementations and more effective matching. An ANN is required, in each instance, to match one time sample or wavelet coefficient from a trace r_b from the base survey. The input to the network is a window of time samples or wavelet coefficients from a trace r_m from the monitor survey that is the nearest to r_b in CMP and offset. In addition, other seismic data are used as input from traces that relate to r_m within the different gathers. This input allows the ANN scheme to assume no direct correspondence of one trace to another on the two time-lapse datasets.

The topology favoured is a two–hidden–layer topology with 20 nodes in the first hidden layer and 10 nodes in the second. The patterns should be normalized linearly, where the output is normalized to the range $[-0.05, 0.95]$ using Equation 2.25 and the input is normalized to $[-0.5, 0.5]$ using Equation 4.2.

Our application of ANN was tested to determine if it suffers from a local minima problem (Section 2.3.4(1)). Both multiple weight initialization and weight jogging do not result in any benefit, indicating that there is no significant local minima problem. The cross-validation stopping criterion is used as the main criterion. However, a few more criteria are added to make stopping more practical. The additional criteria include stopping after a set maximum iterations (3000) and avoiding stopping before a set minimum iterations (300).

Using ANNs as a cross-equalizing tool involves: preparing the seismic datasets, preparing the training patterns and scripts, training ANNs, and applying the trained networks. Therefore, these aforementioned steps were described together with the procedures undertaken to achieve them. In addition, methods were described to quality control the results. Finally, a list is given of the different tests conducted during the progress of this research.

Chapter 5

Matching the Time-lapse Datasets

This chapter describes the progress and results of the ANN cross-equalization scheme.

The chapter builds on the matching procedure described in Section 4.1. In addition, this chapter emphasizes the matching capability, and reserves most of the discussion regarding the reservoir to the next chapter.

The discussion of the two time-lapse datasets in Chapter 3 describes the expected production-related anomaly to be very small. Therefore, one could judge the cross-equalization scheme by its ability to match without initially being too concerned to preserve the production-related anomaly. The next chapter will provide a way of distinguishing if ANNs are able to preserve production-related anomalies while achieving an otherwise good match.

The tests discussed in this chapter are dealt with in their historical order, starting with early tests that showed both problems and promises, to final tests in which most of the early problems were overcome. In practice, this progression was not via a single linear path, but involved many false trails and much back-tracking. The majority of the false trails are not discussed here, and many more tests were performed than have been described in detail.

5.1 Matching in the wavelet domain

The matching by ANNs at the initial stages of this study was done in the wavelet domain. This was motivated by the following considerations:

1. There is a strong incentive for reducing the computational requirement of the method. This is because the process is performed pre-stack, and the initial network topology required to produce a good performance was large. The wavelet domain allows compression.

2. The wavelet domain allows — by allocating a network per wavelet scale — an ANN to be trained above and around a reservoir area, and be applied at the reservoir itself.

3. The wavelet domain allows — by allocating a network per wavelet scale — averaging within a given time section. This means that a network trained to normalize for acquisition differences at a certain time position could be applied to a position that is within the reservoir.

This section summarizes the tests conducted and the results achieved while using the wavelet domain.

5.1.1 Initial tests

Matching one wavelet coefficient at a particular offset in the base survey, using input from the monitor survey and a single network, was first tried and eventually gave a satisfyingly low error. These early tests were taken a step further using multiple networks to try to match all coefficients at one offset. This permitted the matching, and thus the visual analysis of a whole offset gather back in the time domain. The results were promising, but to be able to judge the performance of the network, one needs to train for all offsets, process the outcome and stack the traces. This will give a better judgement as to whether the error in mapping is just a random noise that can be suppressed by normal processing algorithms or not.

Judging from the application to one offset gather, around 240 days are required to train and apply ANNs to produce a cross-equalized match to the full in-line of the base survey (using the same method and with the hardware available). This is of course impractical during this research when many alternatives must be explored. Until the scheme is developed further to be more efficient, the data are reduced in such a way as to allow both reasonable analysis and fast application. To establish this, the following steps were undertaken and are illustrated in Figure 5.1:

1. Only half the possible offsets are considered and hence the estimated training time is reduced to 50%. It was decided to consider the near offset half, and thus channels 117 to 236 are considered. In addition, a network that is dedicated to one offset would in fact be trained for two adjacent channels, thus further reducing training.

2. Instead of matching all offsets in the chosen half, every other network is omitted, see Figure 5.1. This will further reduce training time by 50%.

3. Instead of considering the full trace, only samples in the time interval 2100–2612 ms are considered. In addition, data are re-sampled to 8 ms instead of the original 4 ms. Hence, instead of having 1024 samples as before now there are only 64 time samples and hence 64 wavelet coefficients. This will further reduce training time by a factor of 16.

Before training, the re-sampled data were prepared and stacked. This was to determine whether they are sufficient to give a good stack. The stack of the base survey, shown in Figure 5.2, seems satisfactory, hence training commenced on the re-sampled data.

Configuration

One ANN is used per wavelet coefficient within each offset gather. The networks have the topology shown in Figure 5.3.

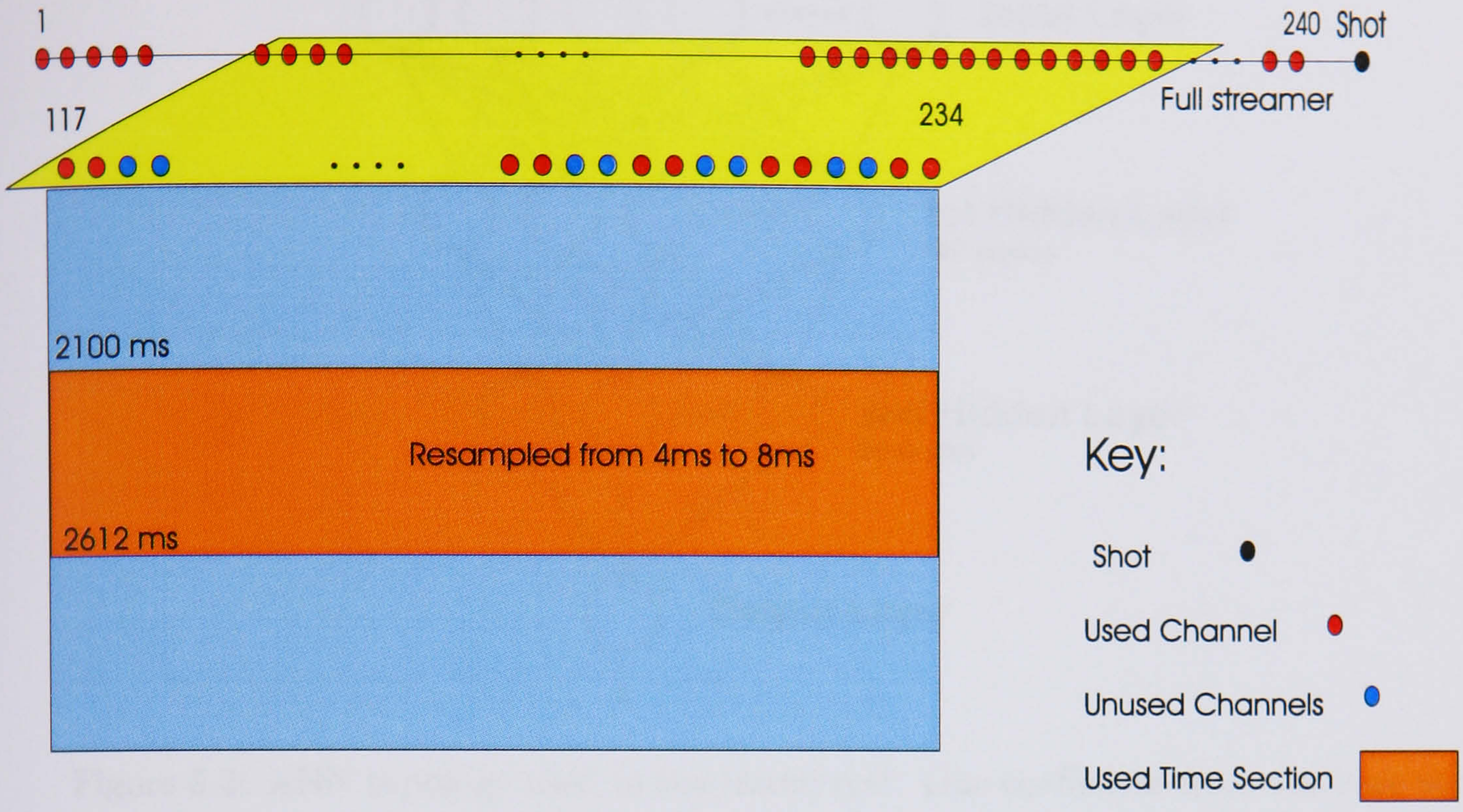


Figure 5.1: Data considered in the analysis in Section 5.1.1. The original streamer has 240 channels from which every other two channels are used from channels 117–236. In addition, only the time section 2100–2612 ms is used after re-sampling to 8 ms.

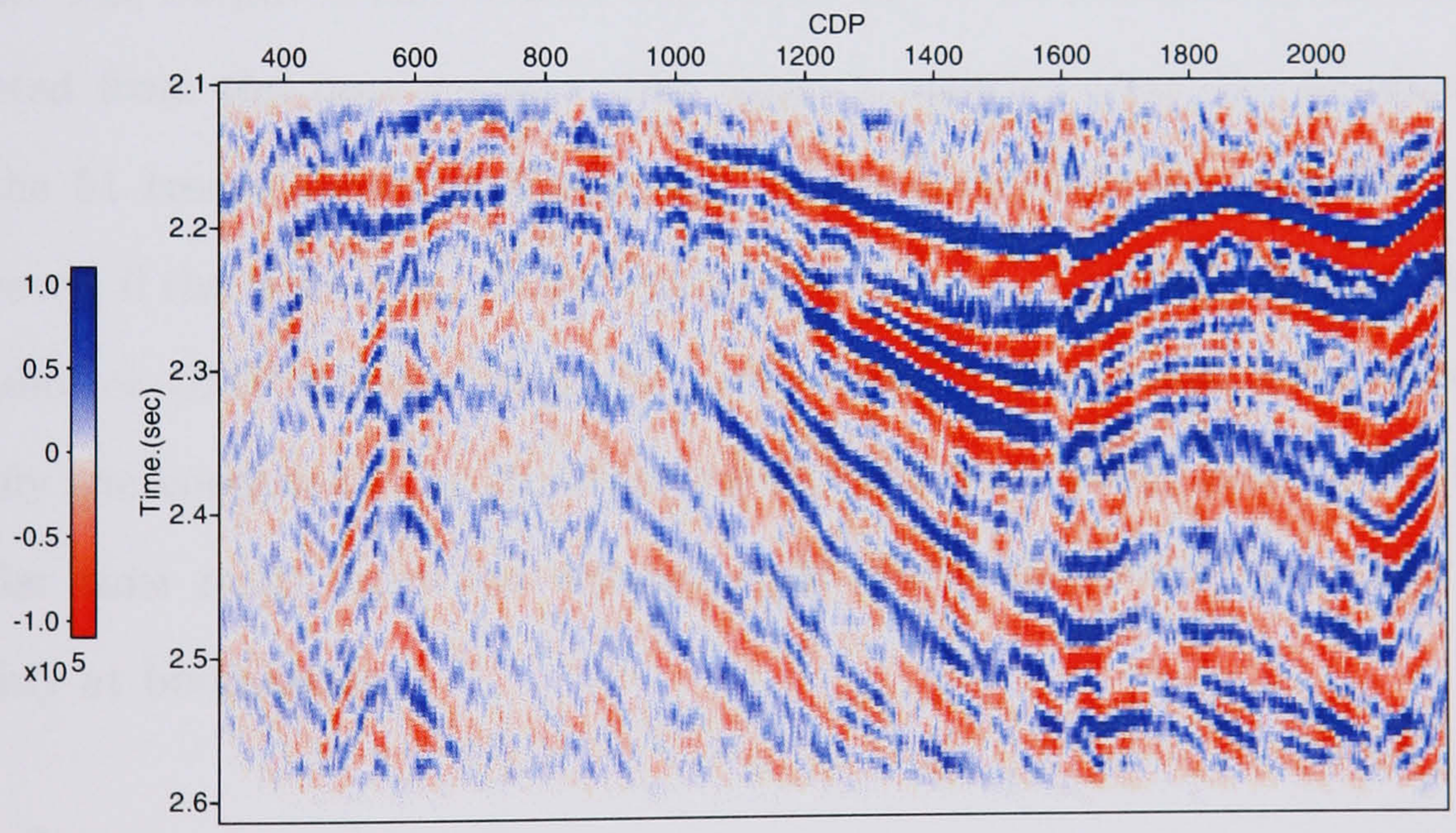


Figure 5.2: Two time-lapse seismic lines NH8906 and NH9201 were acquired over the Oseberg field in 1989 and 1992 respectively. The figure shows a seismic stack plot of data from the base survey, NH8906. Only a re-sampled version of the data is used as illustrated in Figure 5.1. Compare this stack with that of line NH9201 in Figure 5.5. Although the two lines are base and monitor surveys over the same area, there are clear differences.

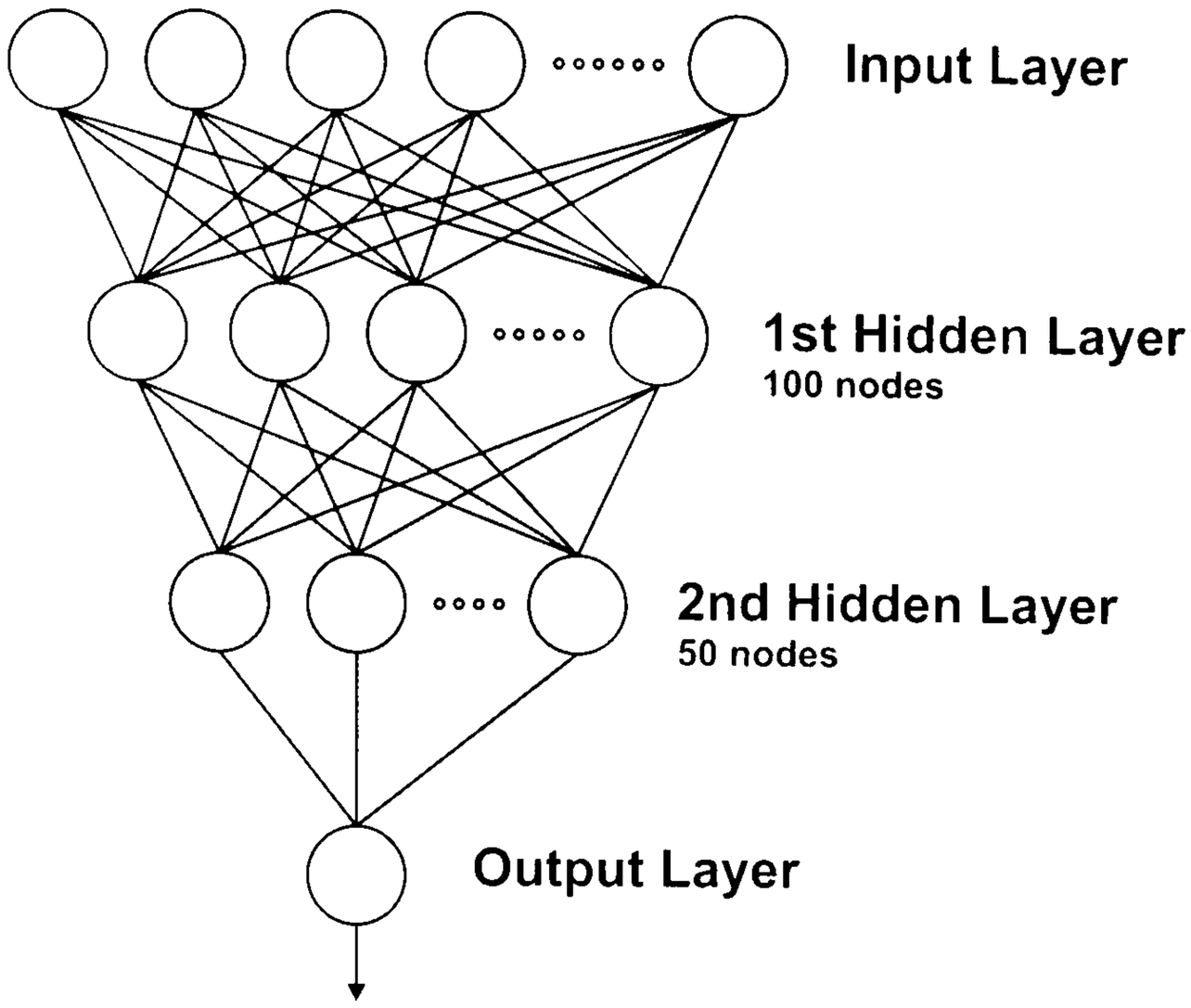


Figure 5.3: ANN topology used in the initial test. One coefficient from the base survey is used as a target output. Input patterns are generated from the monitor survey and are typically 150 coefficients from 51 traces that relate to the position of the output trace used to generate the target coefficient.

Each training pattern contains a target output and around 150 parameters as an input. The output is the wavelet coefficient C_{ij} to be matched by the ANN concerned selected from the base survey. The input is selected from the monitor survey from all the 51 traces shown in Figure 4.4, where each trace supplies the coefficient C_{ij}. However, if the trace is the direct coinciding trace r_m or immediately adjacent in either the shot or CMP domain, then the trace supplies more coefficients. These five traces supply the coefficient C_{ij} plus four coefficients from either side of the time axis and similar time range from the wavelet scale above and below C_{ij}. The choice differs slightly at boundaries.

Results

The training and application required around two weeks within a 16 computer cluster, i.e. around 220 CPU days.

The mean square error of the validation dataset ($\text{MSE}_{\text{valid}}$) of the different trained networks is shown in Figure 5.4. All offsets performed similarly. There is a certain

trend of error versus time seen on the higher three scales in Figure 5.4. The trend could be explained by two main reasons:

1. Boundary coefficients are harder to match since the scheme has to work with less input data. This happens because to match a certain coefficient in the base survey, inputs include the closest coinciding coefficient in the monitor survey and some neighbouring coefficients. At boundaries some of these neighbouring coefficients do not exist.

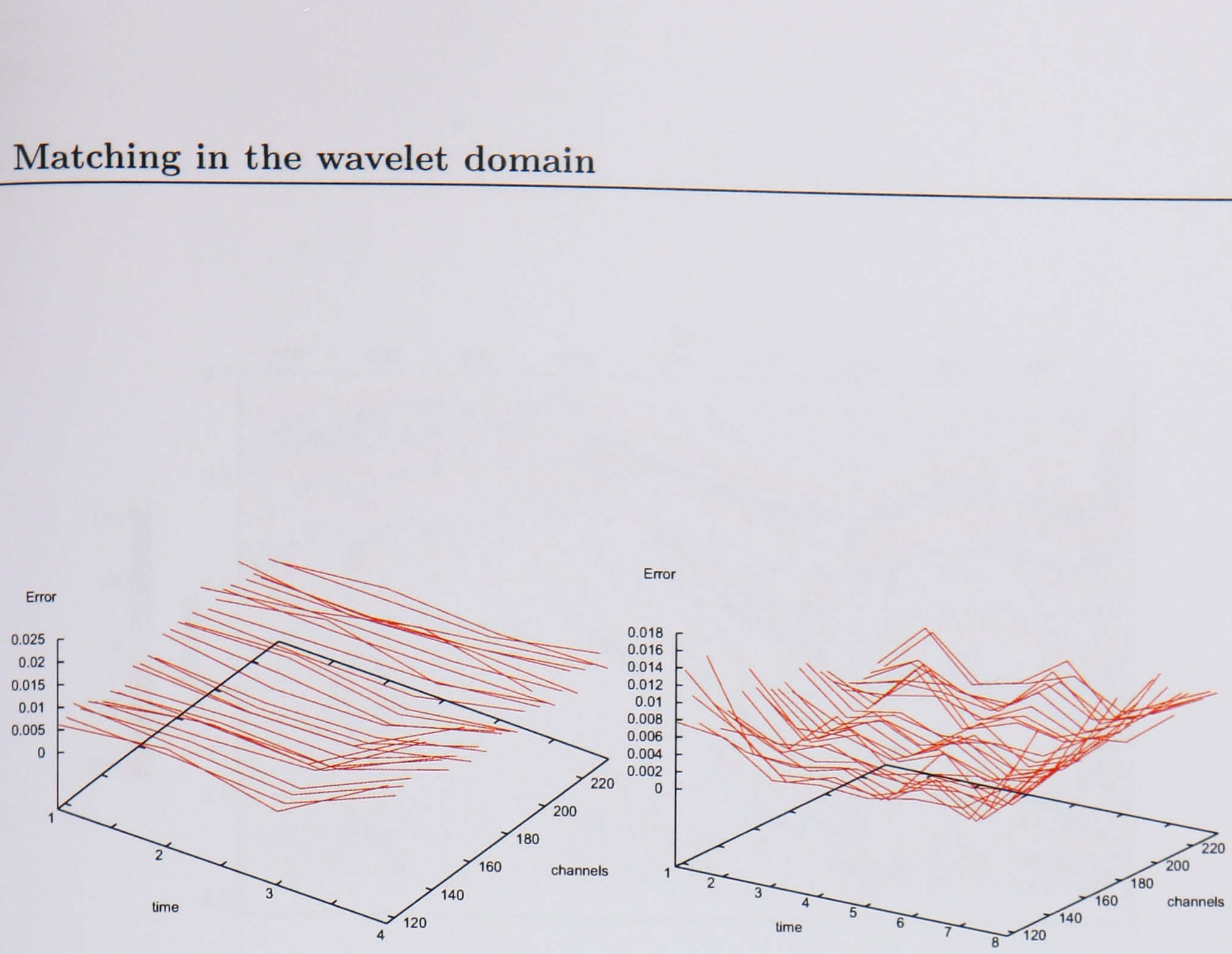
2. Outside boundaries, areas of higher amplitude reflections are matched better due to higher signal-to-noise ratio and hence the error in matching is lower.

The trained ANNs were applied to the whole pattern sets to produce a cross-equalized dataset from the monitor survey that matches that of the base survey. The cross-equalized data and that of the base survey were processed and stacked using the processing flow shown in Table 3.2.

Figure 5.2 shows the stack plot of the base survey. A similar stack of the monitor survey is shown in Figure 5.5. The two plots illustrate the difference between the two surveys.

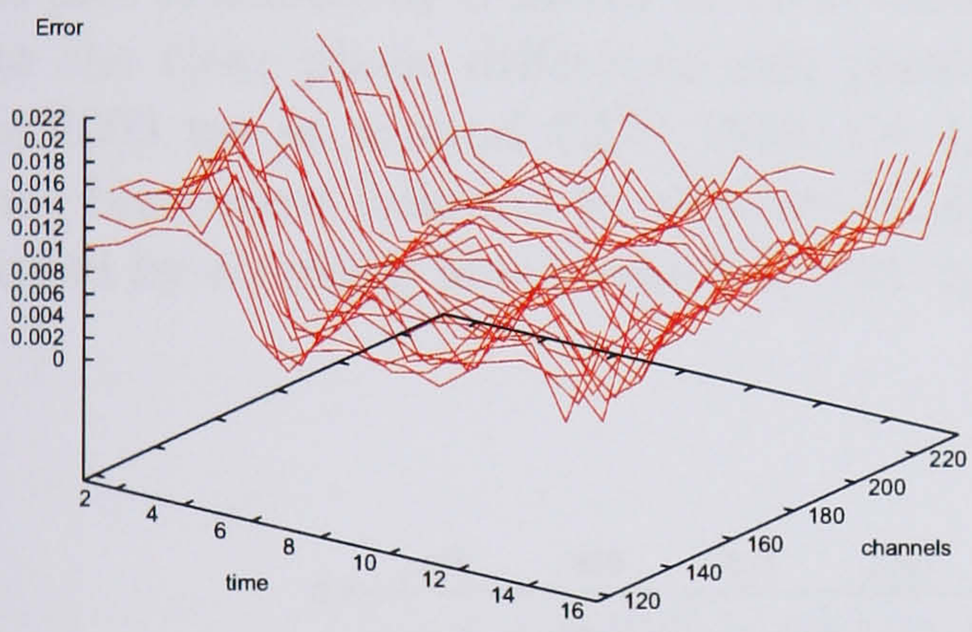
A linear post-stack cross-equalization was applied to the monitor survey stack in Figure 5.5 to match that of the base survey in Figure 5.2. This cross-equalization was conducted in ProMax, and was achieved by applying a phase rotation followed by a least-square matching filter. However, when the matched stack (Figure 5.6) is compared to that it should match, i.e. the base survey (Figure 5.2), there is still a significant difference as shown in Figure 5.7.

The trained ANNs, after application, mapped data from the pre-stack monitor survey and produced a match to the pre-stack data from the base survey. This represents a non-linear pre-stack cross-equalization of the monitor survey to match the base survey. Figure 5.8 shows ANN matched data after stacking. This stack is more similar to the base survey stack (Figure 5.2) than the stack after linear cross-equalization (Figure 5.6). There is a slight lateral smoothing within the cross-equalized data (Figure 5.8). The

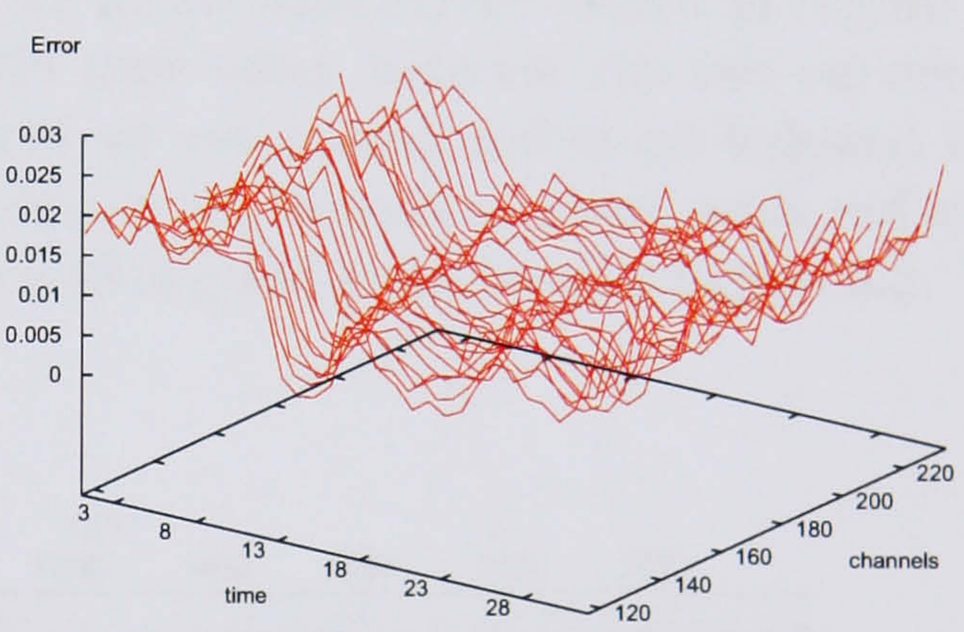


(a) 2nd wavelet scale

(b) 3rd wavelet scale



(c) 4th wavelet scale



(d) 5th wavelet scale

Figure 5.4: The plots show the variation of the mean square error after the trained networks are applied on the validation dataset (MSE_{valid}). The error is plotted against the channel gather matched and versus time. The time axis represents the full time section 2100–2612 ms, but the label is the coefficient number within the wavelet scale (see Figure 4.7). a) error plot for the 2nd wavelet scale with error averaging to 0.005. b) error plot for the 3rd wavelet scale with error averaging to 0.004. c) error plot for the 4th wavelet scale with error averaging to 0.005. d) error plot for the 5th wavelet scale (the Nyquist frequency) with error averaging to 0.0075. There is a certain trend of error versus time seen on the higher three scales, where the error starts a bit high then goes to a minimum, then rises and drops twice before it finally rises again.

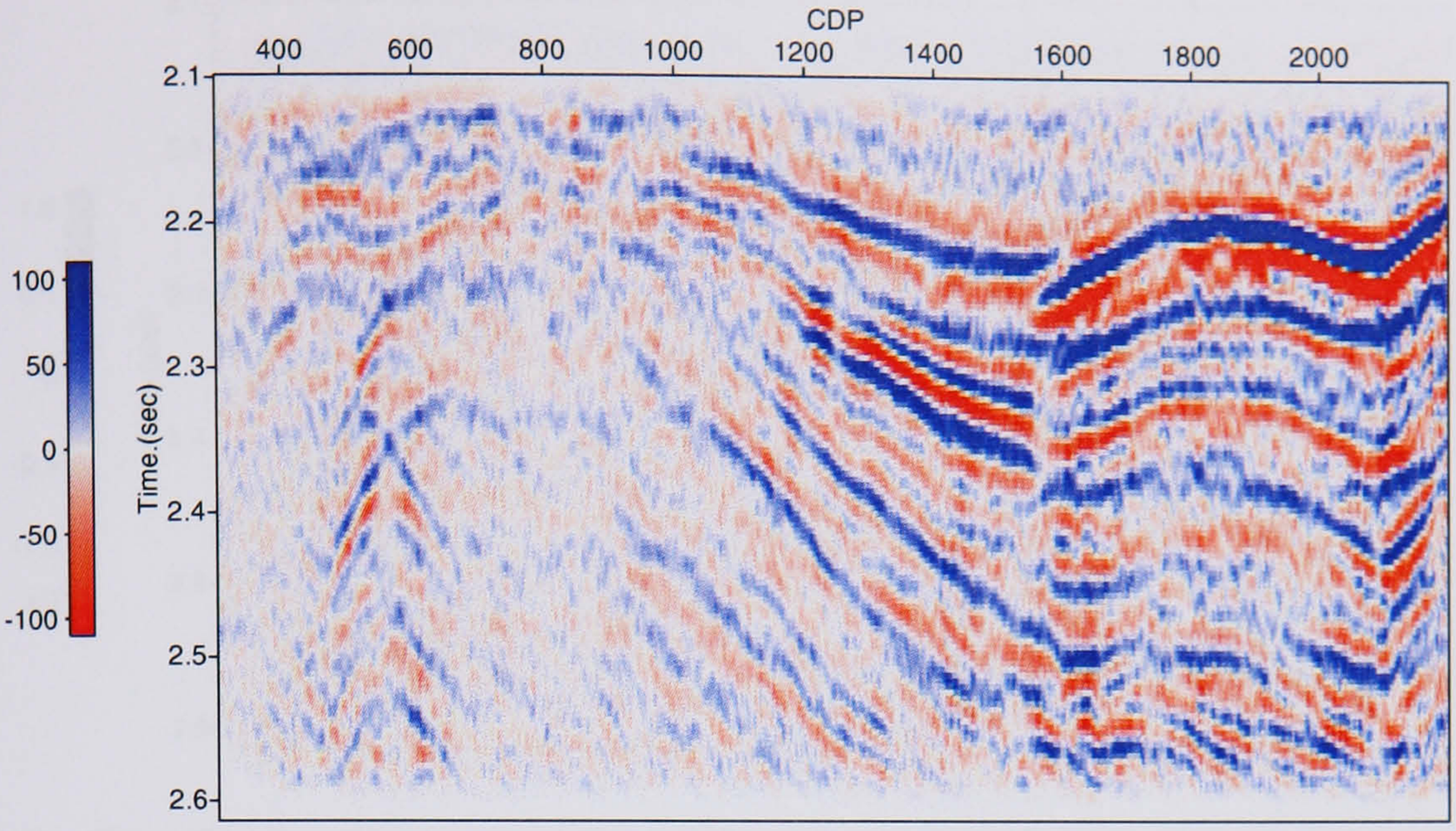


Figure 5.5: The figure shows a seismic stack plot of data from the monitor survey, NH9201. Only the re-sampled version of the data is used as illustrated in Figure 5.1. This plot is scaled by a factor of 1000 relative to the base survey shown in Figure 5.2. Note the clear phase difference and possible time delay between the two surveys at time 2200 ms at around CDP 1850, i.e. here we see a weak red event followed by a strong blue event followed by another weak red event, instead of a very weak red event followed by a strong blue event followed by a strong red event seen in Figure 5.2.

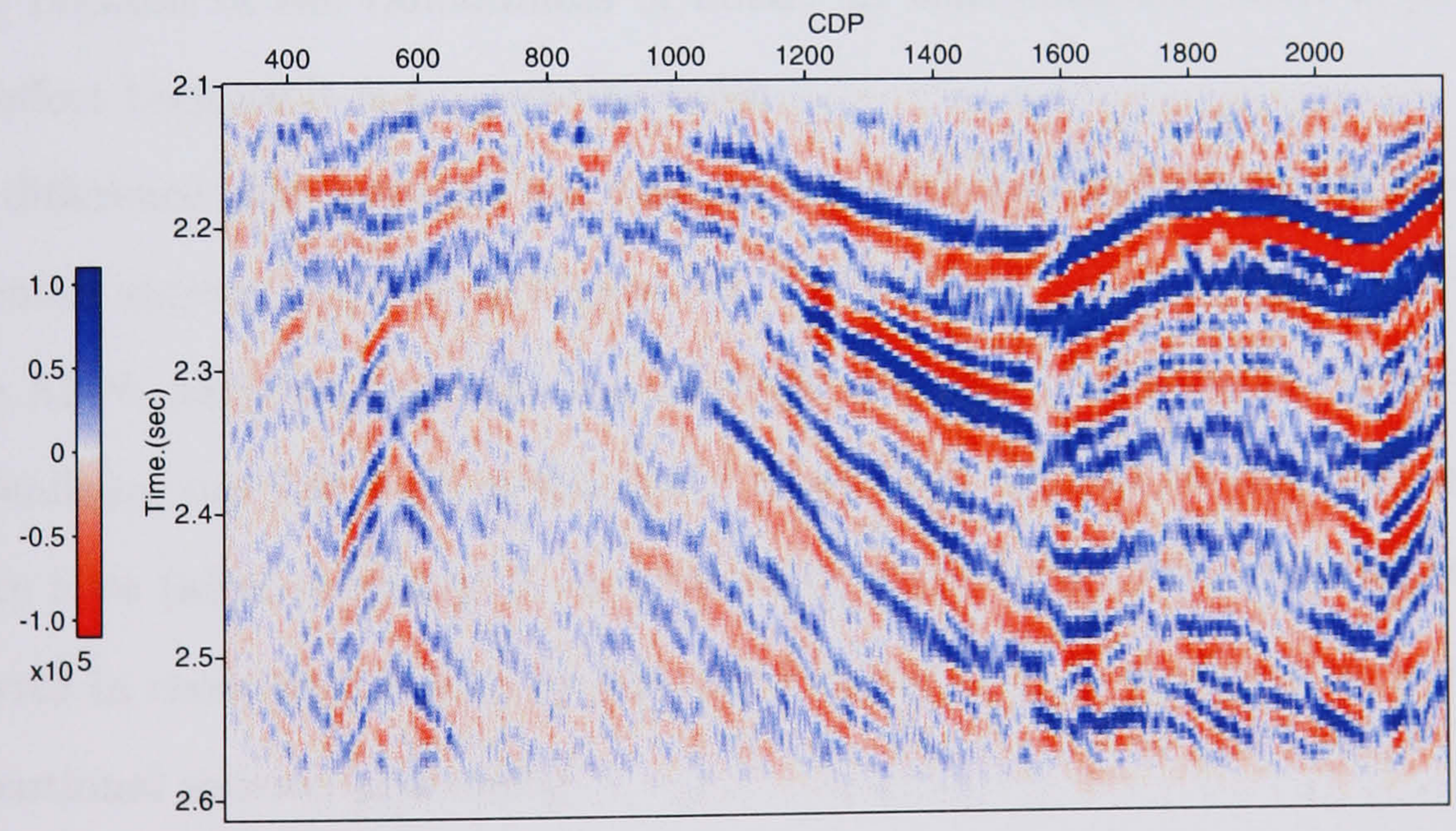


Figure 5.6: The figure illustrates the monitor survey stack (Figure 5.5) after post-stack linear matching with the base survey stack (Figure 5.2). The plot uses the same scale as that in Figure 5.2. Compare this figure to Figure 5.8, which shows non linear pre-stack matching done using ANNs. A difference plot between this stack and that of the base survey (Figure 5.2) is illustrated in Figure 5.7.

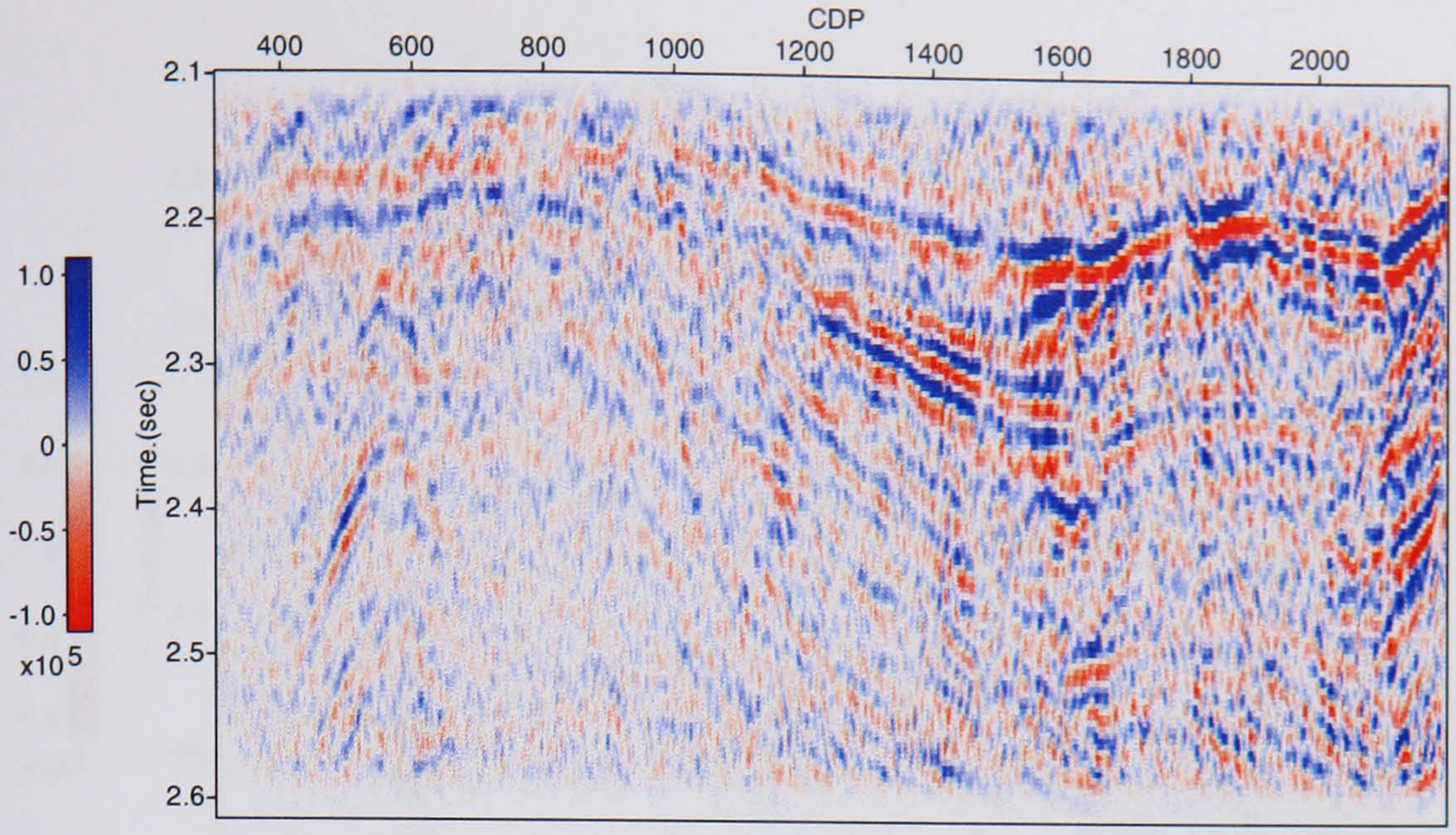


Figure 5.7: The figure illustrates the difference between the base survey stack (Figure 5.2) and the linearly matched monitor survey stack (Figure 5.6). Compare this to Figure 5.9, where ANNs non-linear cross-equalization shows lower error in matching. The plot uses the same scale as that in Figure 5.2.

lateral smoothness exists because the same networks are used for all shot gathers and hence high frequency lateral variation is hard to capture. If this high frequency lateral variation is mainly random noise then failure of the ANNs to map this lateral variation is partly because of the randomness of noise. In that case, the ANN is producing a desired effect because it is suppressing noise.

The difference plot, between the base survey stack and that of the ANN cross-equalized monitor survey, is shown in Figure 5.9. The difference plot in Figure 5.9 illustrates that the ANNs have matched the two surveys very well. Whereas the difference after cross-equalizing one offset is of the same magnitude as that of the base survey, the difference here (after stacking) is at a far lower magnitude. This indicates that much of the error in cross-equalizing any one offset gather is in fact noise that is suppressed by conventional seismic processing.

Comparison between the linear cross-equalization and the non-linear cross-equalization can be made by looking at the difference plots in Figure 5.7 and Figure 5.9. Those two plots are to the same scale as that in Figure 5.2. Both methods of cross-equalization had problems matching abrupt lateral changes, such as that around CDP 1600

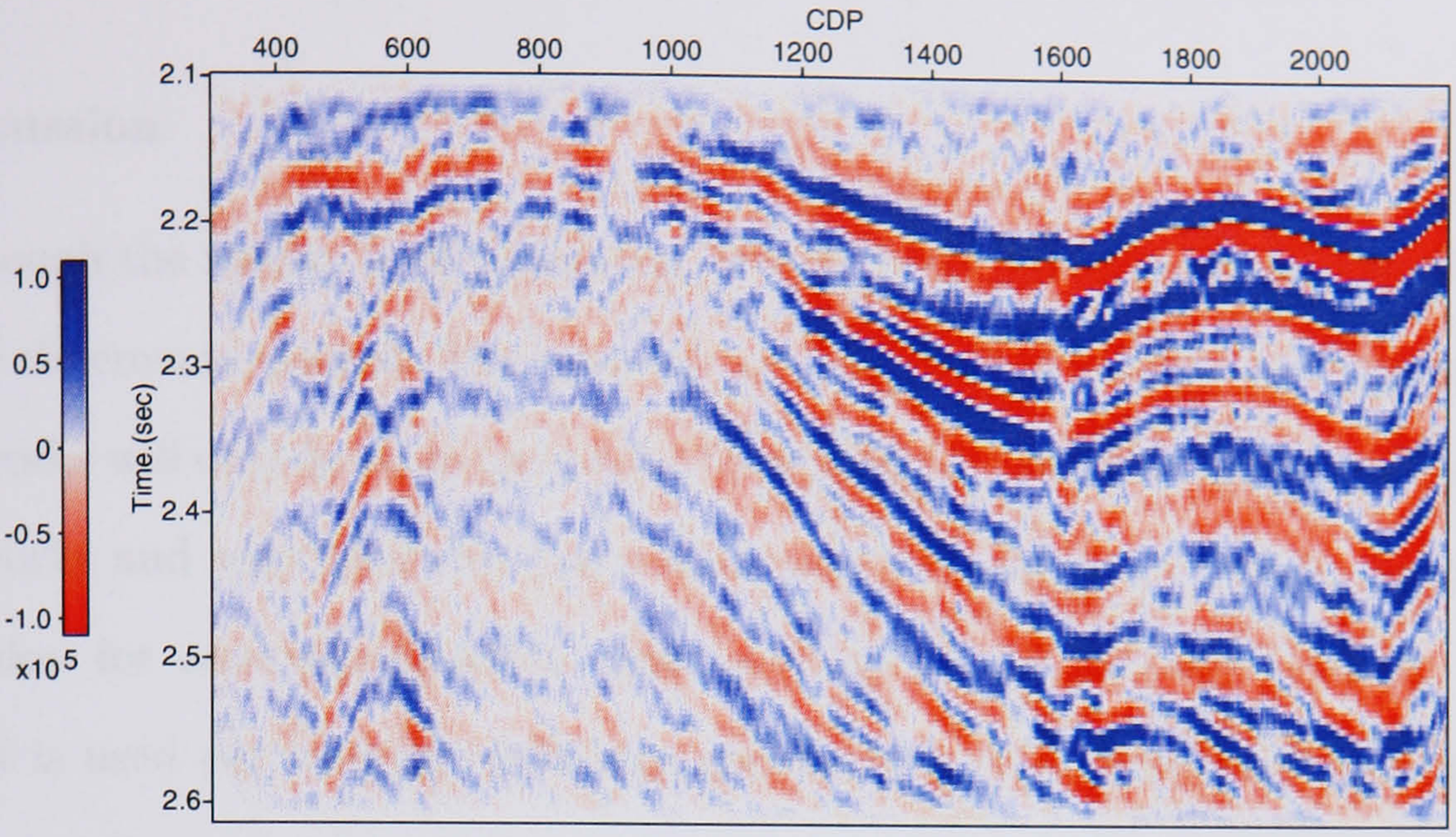


Figure 5.8: The figure illustrates a seismic stack of the monitor survey after pre-stack cross-equalization using ANNs with base survey. The plot uses the same scale as that in Figure 5.2. Compare this to Figure 5.6 that shows linear matching done using ProMAX processing package. Note how the discontinuity at around CDP 1600 is matched better after ANNs cross-equalization. A difference plot between this stack and that of the base survey (Figure 5.2) is illustrated in Figure 5.9.

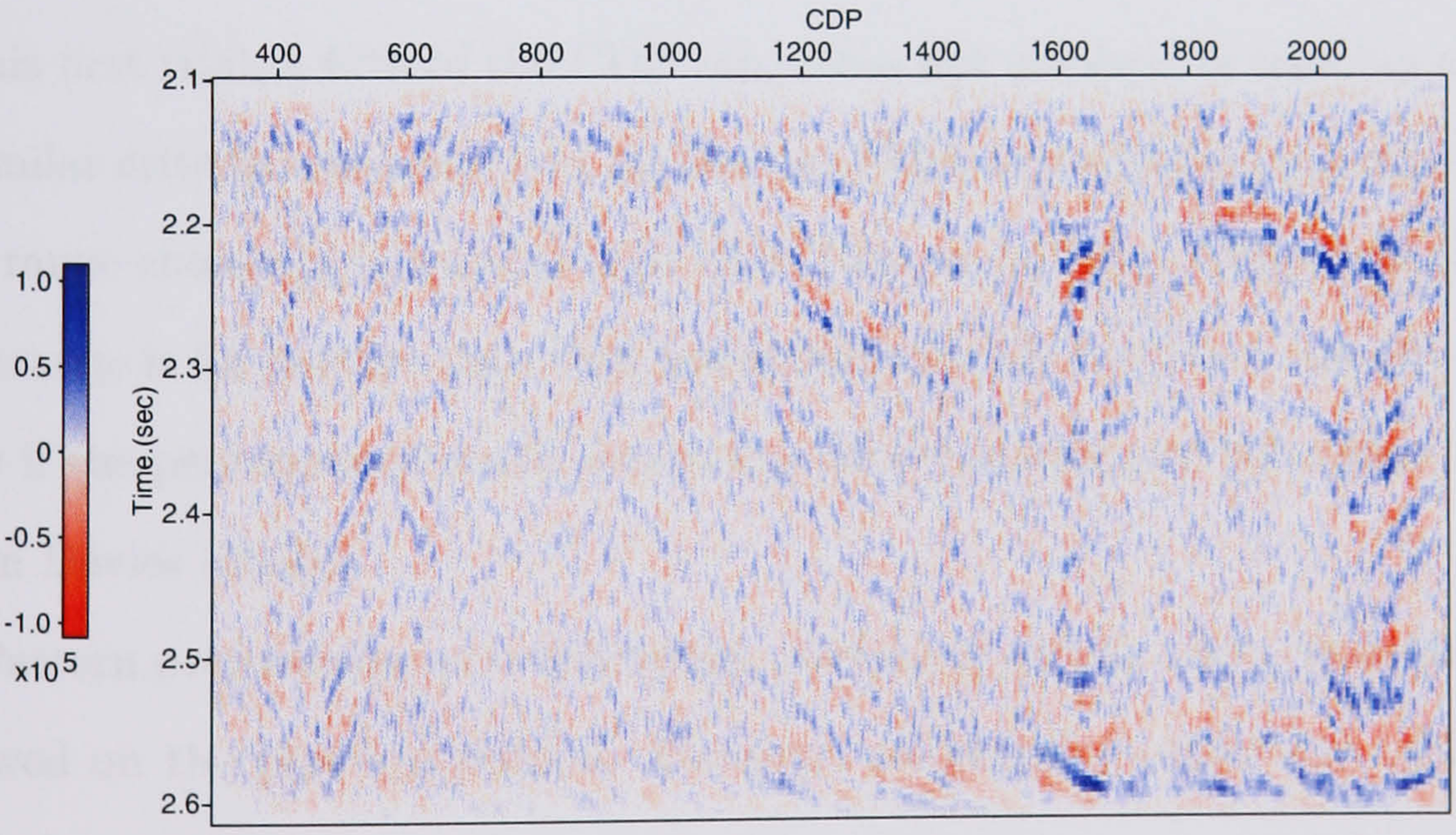


Figure 5.9: The figure illustrates the difference between the base survey stack (Figure 5.2) and the non-linearly cross-equalized monitor survey stack (Figure 5.8). Compare this to Figure 5.7, where linear cross-equalization shows higher error in matching. The plot uses the same scale as that in Figure 5.2.

and difficulty near boundaries. However, ANN matching is superior.

Discussion

Although the results were promising, it was determined that there are still major steps and refinements needed. The reservoir area should be excluded from training so that the networks will only learn to normalize for acquisition related differences. Secondly, many networks and very long run times were used to achieve the result. To get a practical solution for cross-equalization, these need to be significantly reduced. Thirdly, one ANN is used per wavelet coefficient and hence a trained network can only be applied to a very limited time and frequency range.

In the following two sections, I will show work done to try to investigate the possibility of excluding an area from training. This work was intended to allow training outside a reservoir area, while being able to apply the trained networks within the reservoir. Later, I will discuss the desirability or otherwise of this approach.

5.1.2 Excluding a CDP range from training

In this first trial, a fifth of the CDP range was not involved in training nor validation. A similar criteria was employed by Davies (2000) to avoid training over the reservoir. The range chosen here was between CDP 1000 and 1350 as illustrated in Figure 5.10. This range is far smaller than that associated with the reservoir, but the trial serves to show if the performance would deteriorate at the section excluded from training (as it did in Davies (2000)).

Pattern preparation, network design, and input-output choice was identical to that followed on the previous section. However, all patterns associated with the selected CDP range were designated as as a testing dataset[1].

The trained ANNs were applied to the re-sampled monitor survey data (Figure 5.1). The data were processed and stacked. Figure 5.11 shows the difference between the

[1]A testing dataset is not involved in training as the training set nor involved in cross-validating the training as the validation set, see Section 2.3.1.

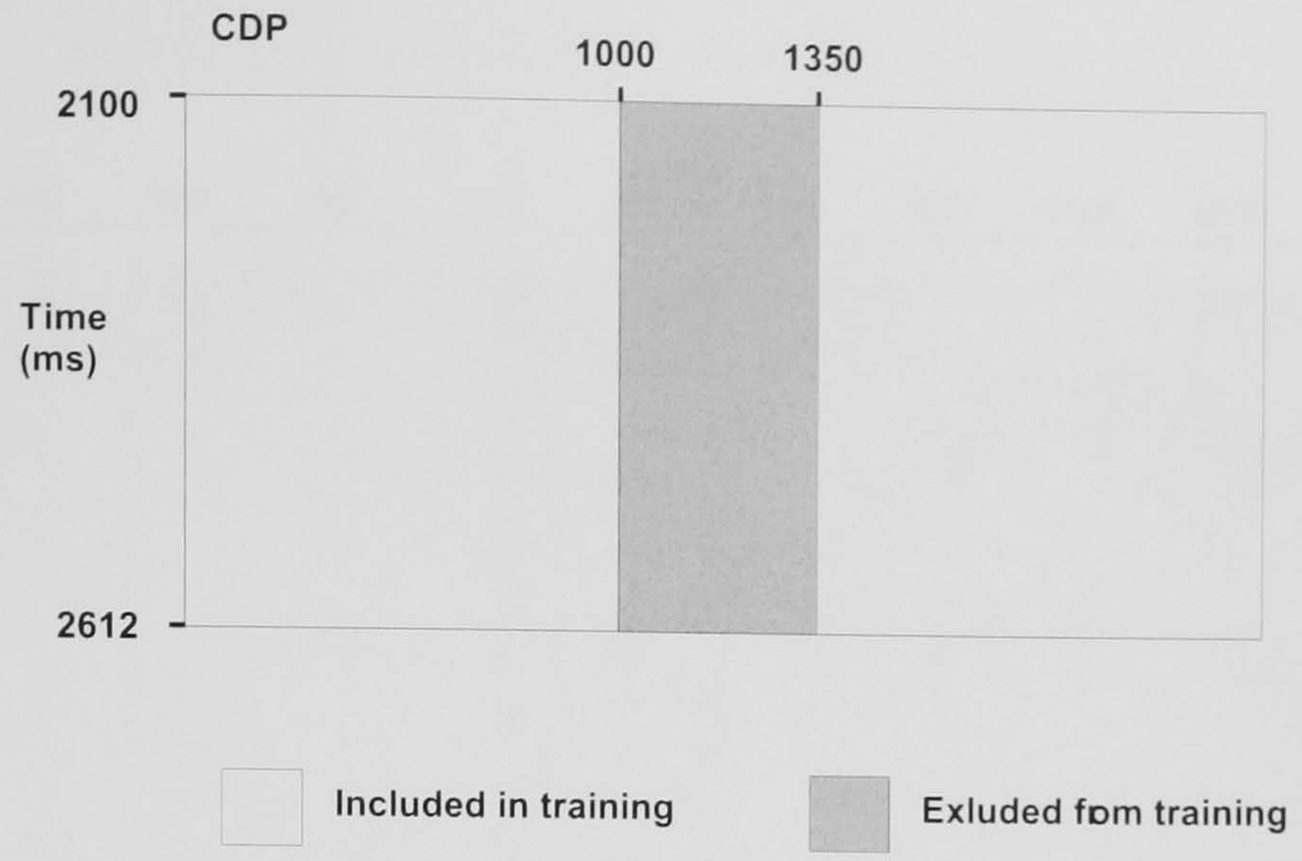


Figure 5.10: A schematic diagram showing the section excluded from training while preparing ANNs to interpolate over the CDP range 1000–1350.

stack plot of the ANN cross-equalized monitor survey and the base survey (Figure 5.2). The difference plot shown in Figure 5.11 indicates similar performance to that shown earlier in Figure 5.9, but also reveals a large matching deterioration at the area excluded from training, i.e. at CDP 1000–1350. Therefore, it was concluded that a different approach is required to allow ANNs to generalize well at the reservoir region.

5.1.3 Excluding a box area from training

Tests were conducted to assess the possibility of using one network to map all coefficients within one wavelet scale. Within each wavelet scale, there are a number of coefficients each relating to a certain position in time. This number depends on the scale (resolution), see Section 4.2.3. A coefficient within a certain scale may require similar matching mechanism as another coefficient within the same scale. In addition, it may be possible for a network to learn how the matching mechanism changes as a function of time. This, in turn, would allow a network trained on coefficients localized at a certain time position to be applied to coefficients of the same scale localized at a different time position. This method will allow using seismic data above the reservoir in training and may therefore perform better than excluding a whole CDP range. Figure 5.12 shows how a box area could be excluded from training.

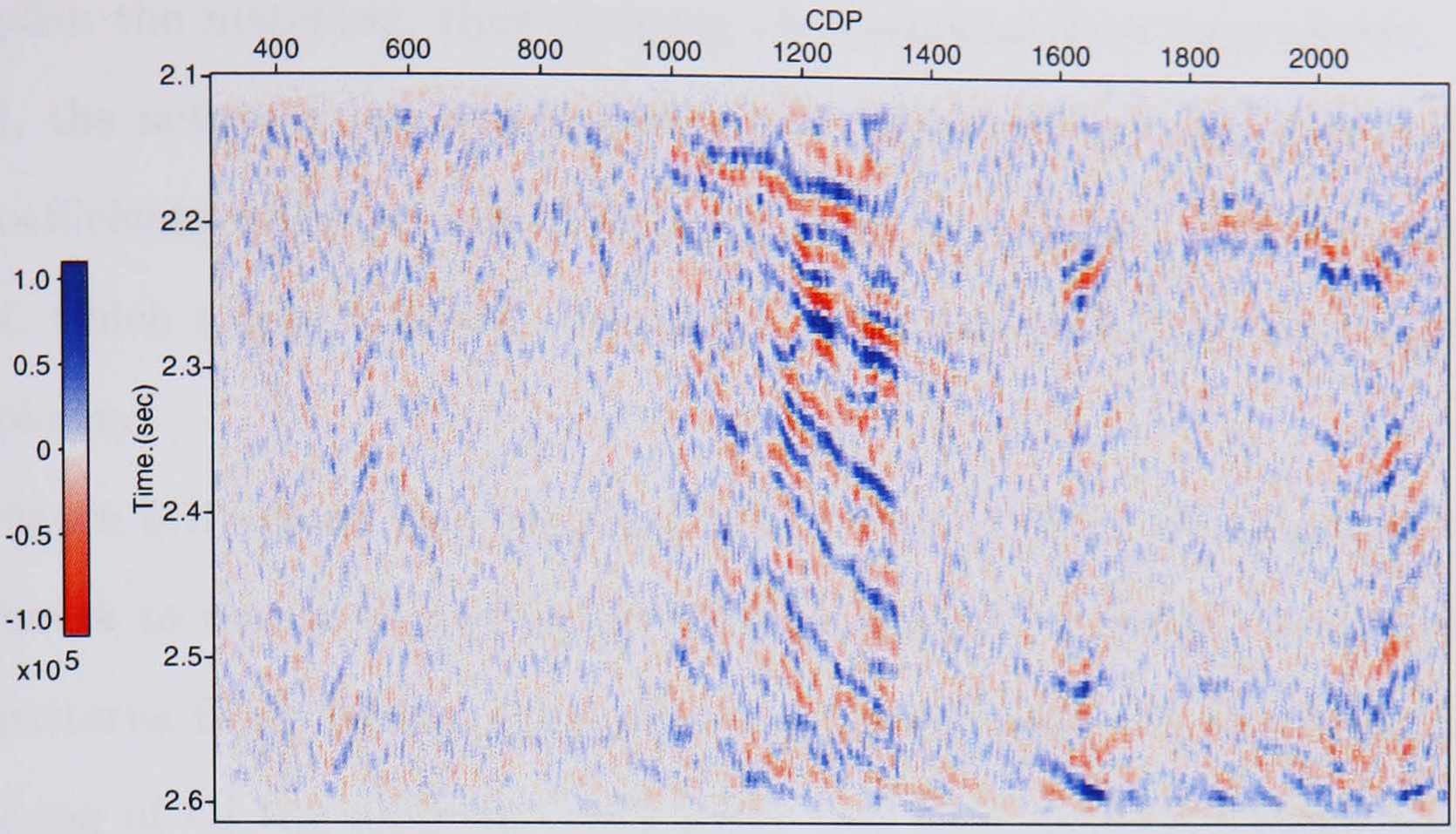


Figure 5.11: Difference stack plot between the base survey (Figure 5.2) and the ANN cross-equalized monitor survey after excluding a CDP range. The plot is similar to that in Figure 5.9, but shows a deterioration in matching between CDP 1000 and 1350 when compared to it.

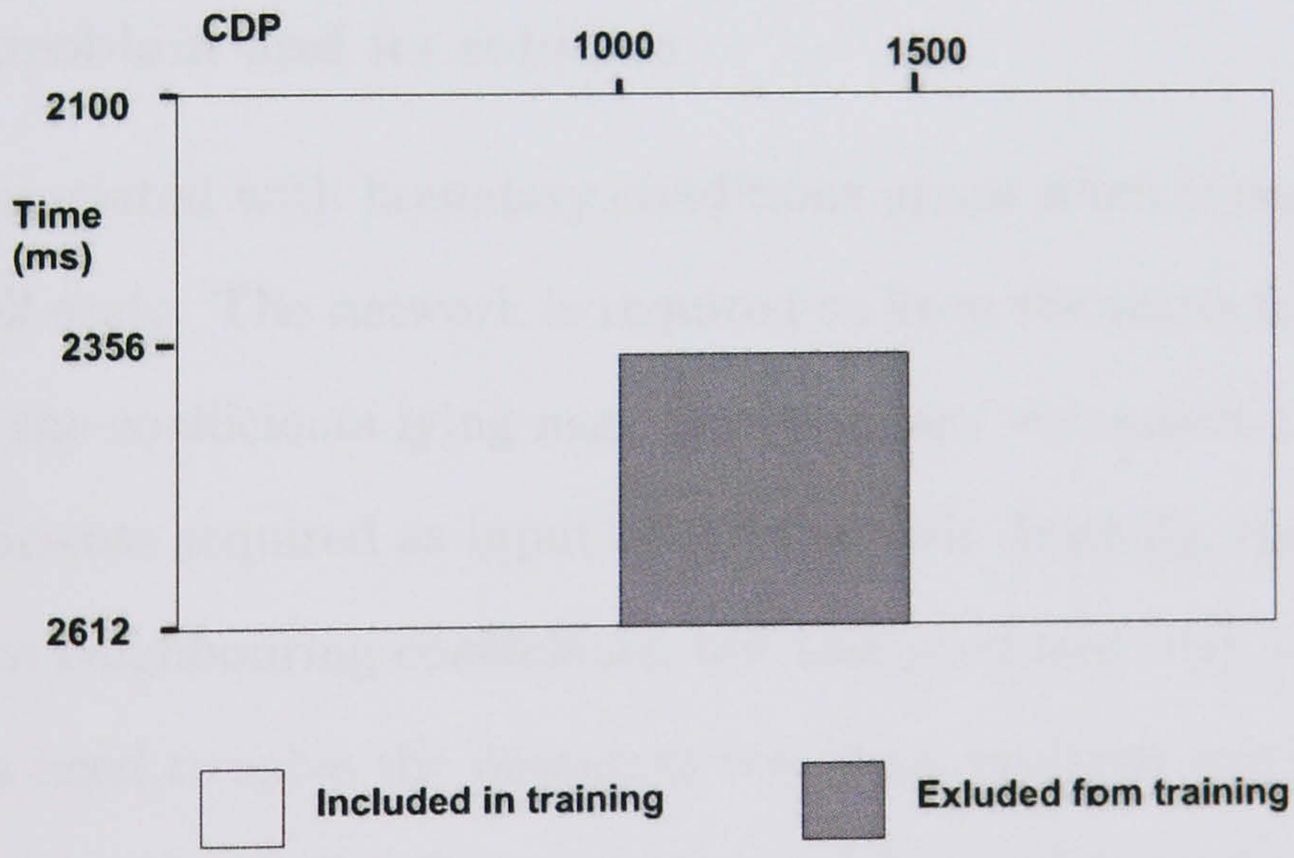


Figure 5.12: A schematic diagram showing the section intended to be excluded from training. The trained networks would be required to interpolate over a box area in the CDP range 1000–1500 at the lower half of the time section 2100–2612 ms. A network, if trained for a full wavelet scale, can be trained on coefficients that lie above the region excluded from training.

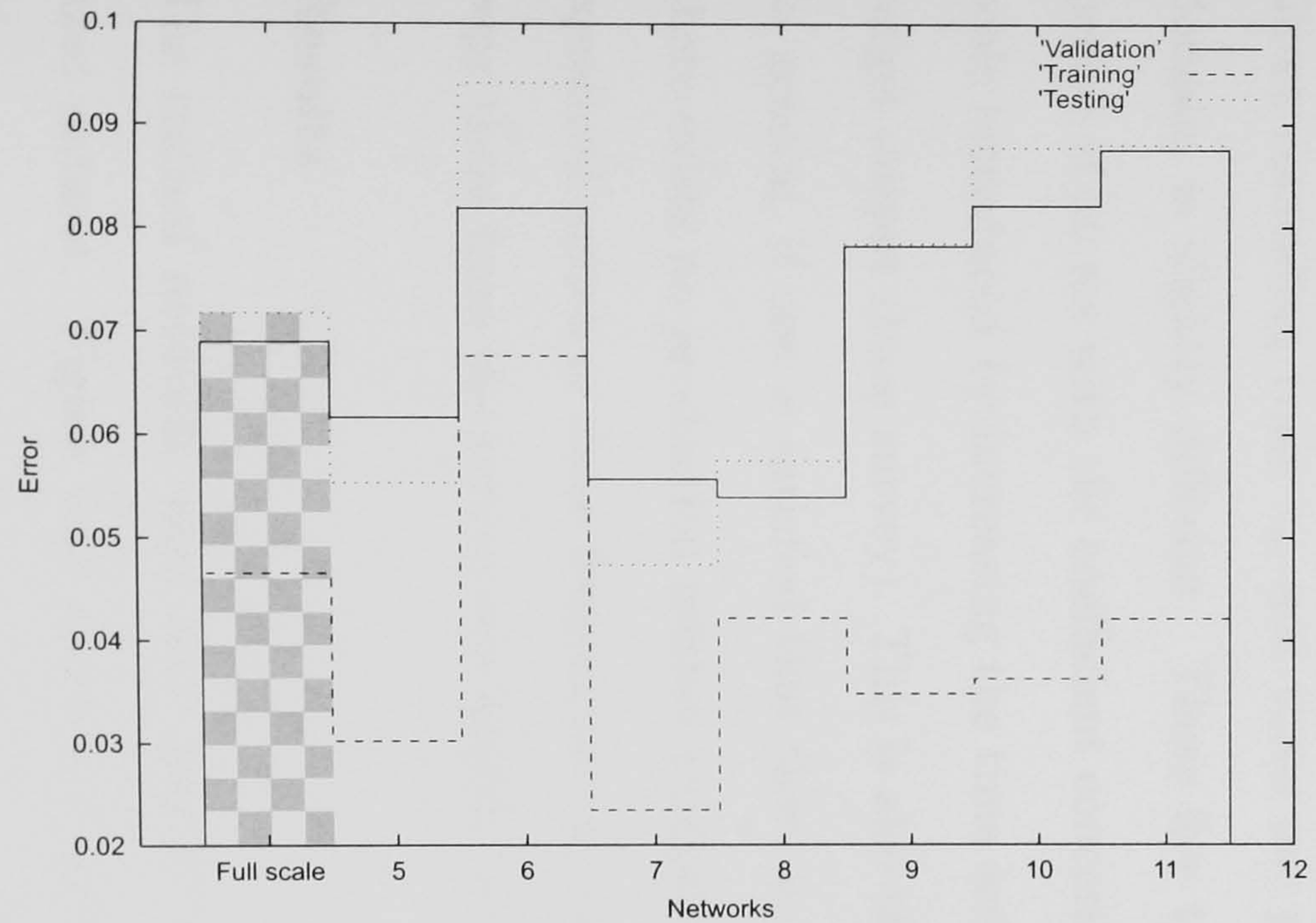
Assigning a network per scale reduces the total number of ANNs required to accomplish the matching, thus reducing the computational requirement. On the other hand, the networks are required to solve a more difficult problem (having to match all coefficients within a scale). However, more patterns are available to an individual ANN, which might help the networks to learn and overcome the increase in problem complexity.

Figure 5.13 shows the results of initial tests carried out to assess performance of a network to match all coefficients within a scale. Two scales were tested with training patterns from Section 5.1.1. In each scale, a network was designated to do the matching of all the different coefficients, and then its performance was compared to the performance of networks designed to match each of the individual coefficient. The error in the matching displayed by the network allocated for the whole scale is similar on average and in some cases lower than the error produced by networks allocated for individual coefficients. In addition, from Figure 5.13(a), one might argue that the network allocated to the whole scale achieved a better generalization.

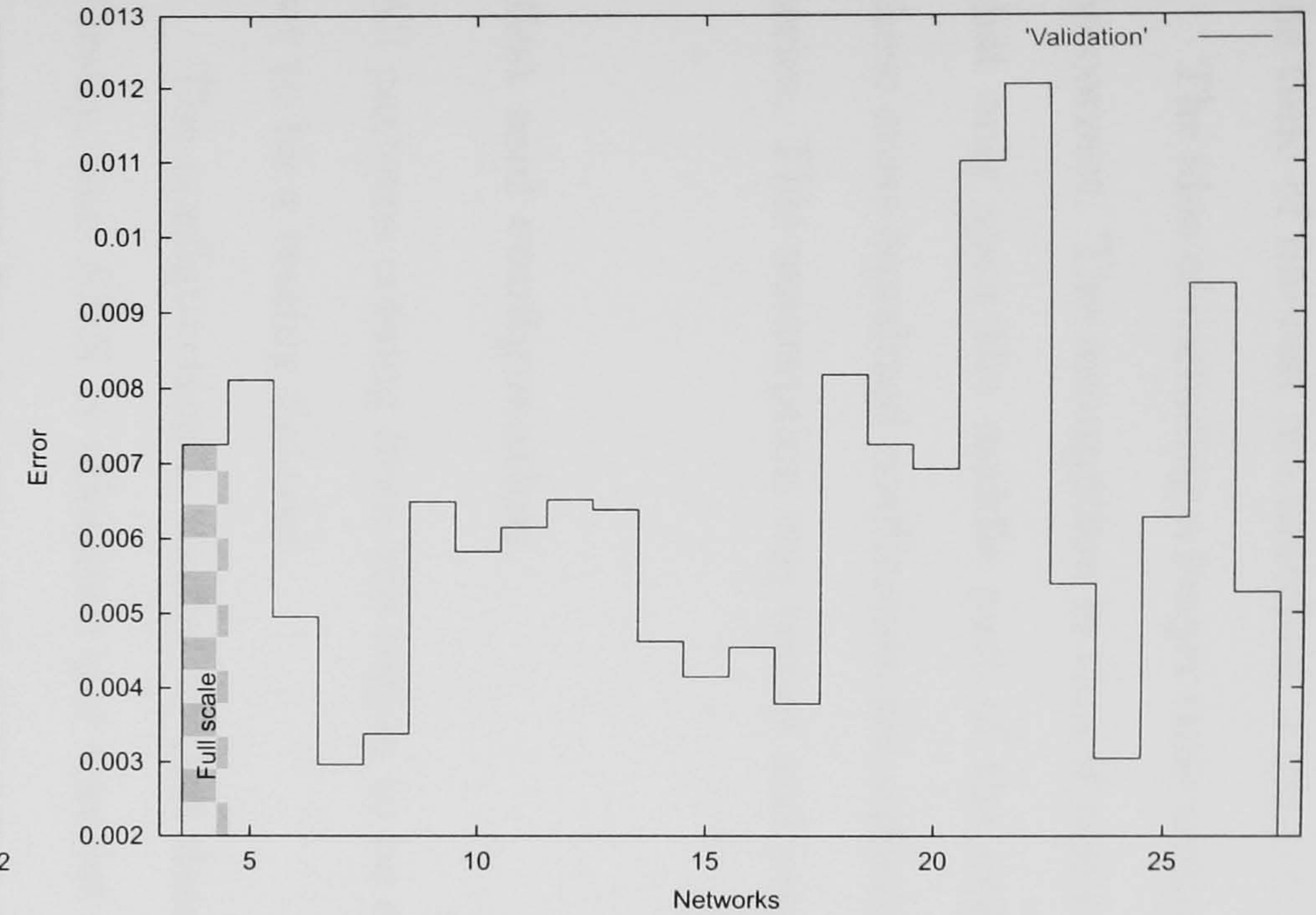
Boundary problem and its solution

A problem associated with boundary conditions arises when trying to designate a network for a full scale. The network is required to keep the same topology design. However, to map the coefficients lying near the boundary we cannot provide for the neighbouring coefficients required as input to the network. Initially, the problem was solved by allowing no neighbouring coefficients, but this produced very unsatisfactory results.

There is a need to solve the boundary condition problem and include neighbouring coefficients in the time axis. This is because a time delay might easily exist between traces from the two time-lapse surveys. Including a time interval in the input allows an ANN to find the most appropriate input to a certain output. To remedy the problem, the time section used is doubled. Instead of considering the time-section 2100–2612 ms (512 ms), a longer time-section 1840–2864 ms (1024 ms) can be considered. The added



(a)



(b)

Figure 5.13: Error graph comparing performance of a network allocated to match a full scale to the performance of networks allocated to match individual coefficients within the scale. a) Error plot for the different networks within the 4th scale corresponding to 32Hz, where the error is plotted for the three different pattern datasets. b) Error plot for the different networks within the 5th scale corresponding to the Nyquist 64Hz. In each graph, the first network is used to match all the coefficients while the others were used to match individual coefficients. Except for the first network the number on the "Networks" axis signifies the relative time position of the coefficient mapped. To avoid boundary problems, coefficients at time boundary are not included in the test. In both graphs the network matching all the coefficients within the scale performed similar to those matching individual coefficients. In addition, on plot (a) we see that the performance on both validation and testing datasets is closer to that on the training dataset for the network allocated to the whole scale. This observation could mean a better generalization achieved by these networks when compared to networks designated for individual coefficients.

time is only used to provide the neighbouring coefficients. Only those coefficients in the time of interest are matched. Figure 5.14 illustrates the proposed scheme.

The idea of including a longer time section when training, requires one assumption to be correct. The assumption is that if one produces cross-equalized wavelet coefficients that only span the middle part of the entire time the wavelet series represent, then these cross-equalized coefficients are sufficient to reproduce the middle part of the time series. This assumption was tested and proven to be true as illustrated in Figure 5.15.

Test and configuration

All patterns coming from the region to be excluded from training (see Figure 5.14) are set to be a testing dataset.

The configuration is similar to that described in Section 5.1.1 with few alterations. Firstly, one ANN is allocated per wavelet scale per offset. Secondly, input from the direct coinciding trace r_m and those one position away from it in the shot or CMP domain, is slightly different. These five traces supply coefficients that span a time period of 96 ms with the coefficient concerned (C_{ij}) in the middle. Thirdly, the lowest scale introduced by increasing the time series is not matched, but is derived from the target output (base survey). This is also the case for the mother-function coefficients. In general, if one is satisfied that there is a certain frequency threshold below which there exists no production related anomaly, then there is no need to produce a cross-equalized match of these frequencies, and it is sufficient and more appropriate to just copy these from the survey one intends to match (the base survey in our case).

Results

The trained networks were later applied to the whole area producing a cross-equalized dataset. Figure 5.16 shows the difference plot between the base survey stack (Figure 5.2) and the cross-equalized monitor survey using the scheme mentioned here.

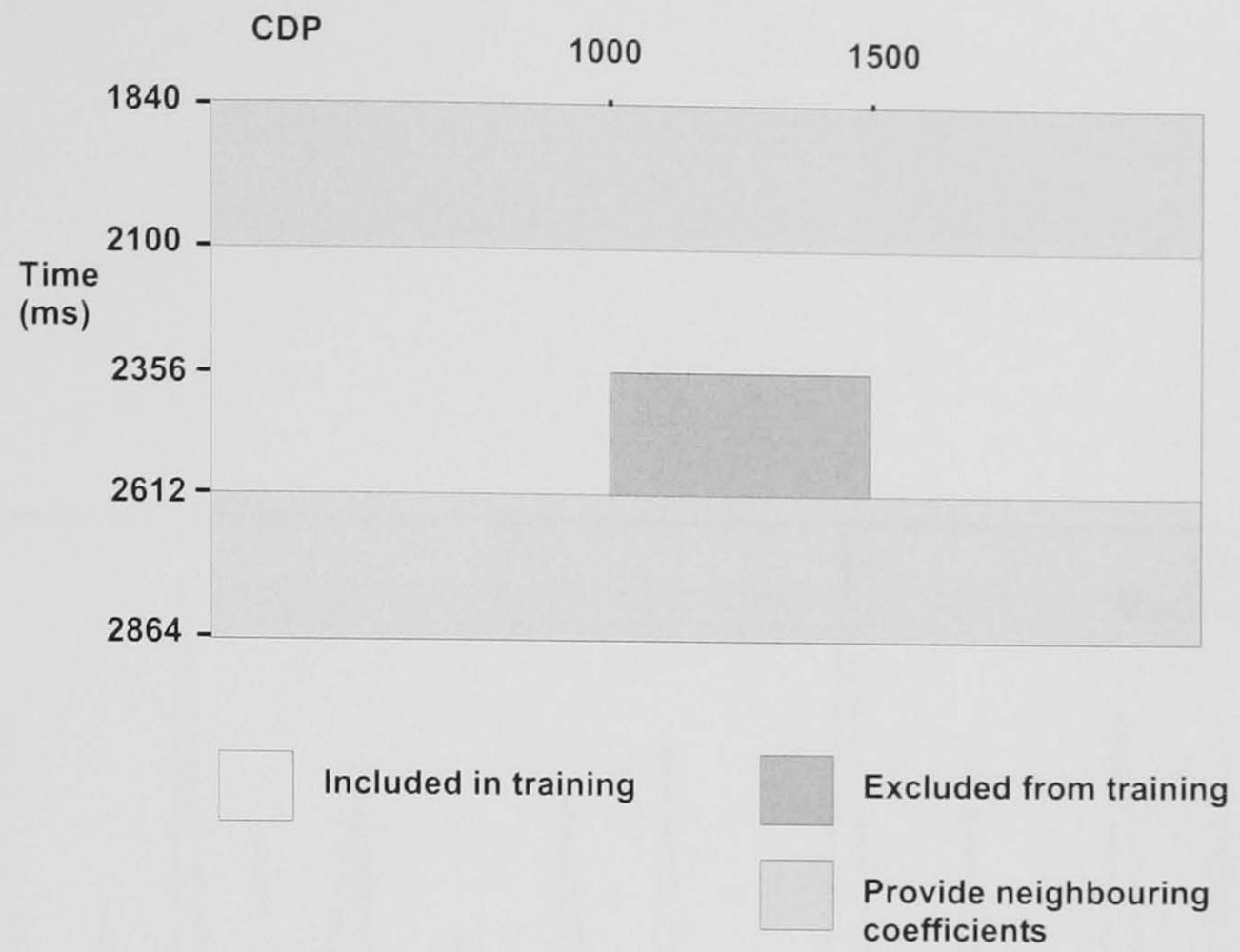


Figure 5.14: A schematic diagram showing the section intended to be excluded from training while providing for neighbouring inputs. The trained networks would be required to interpolate over a box area in the CDP range 1000–1500 at the lower half of the time section 2100–2612 ms. The network could utilize the first and last quarter of the shown time section (1840–2864 ms) to provide for all the input data required to match the section 2100–2612 ms.

This figure shows that the matching has deteriorated due to utilizing one ANN per scale instead of one per coefficient.

5.1.4 Discussion

The last section showed initial tests and progress of the research. Although some of the results look promising, improvement is required on three main aspects. Firstly, the general matching capability should be improved. Secondly, the computational load should be reduced. This was achieved by designing one ANN per scale, but further work is required to maintain the matching performance seen earlier and to then improve on it. Thirdly, a better way is required to allow the ANNs to learn outside the reservoir while being applicable to the reservoir area.

Excluding a significant area from training, to avoid making the network obliterate the production related anomaly within the reservoir, suffers from a few problems. The reservoir is usually hard to pin-point exactly causing one to exclude a large area from training. In addition, both schemes put forward to exclude an area from training do not

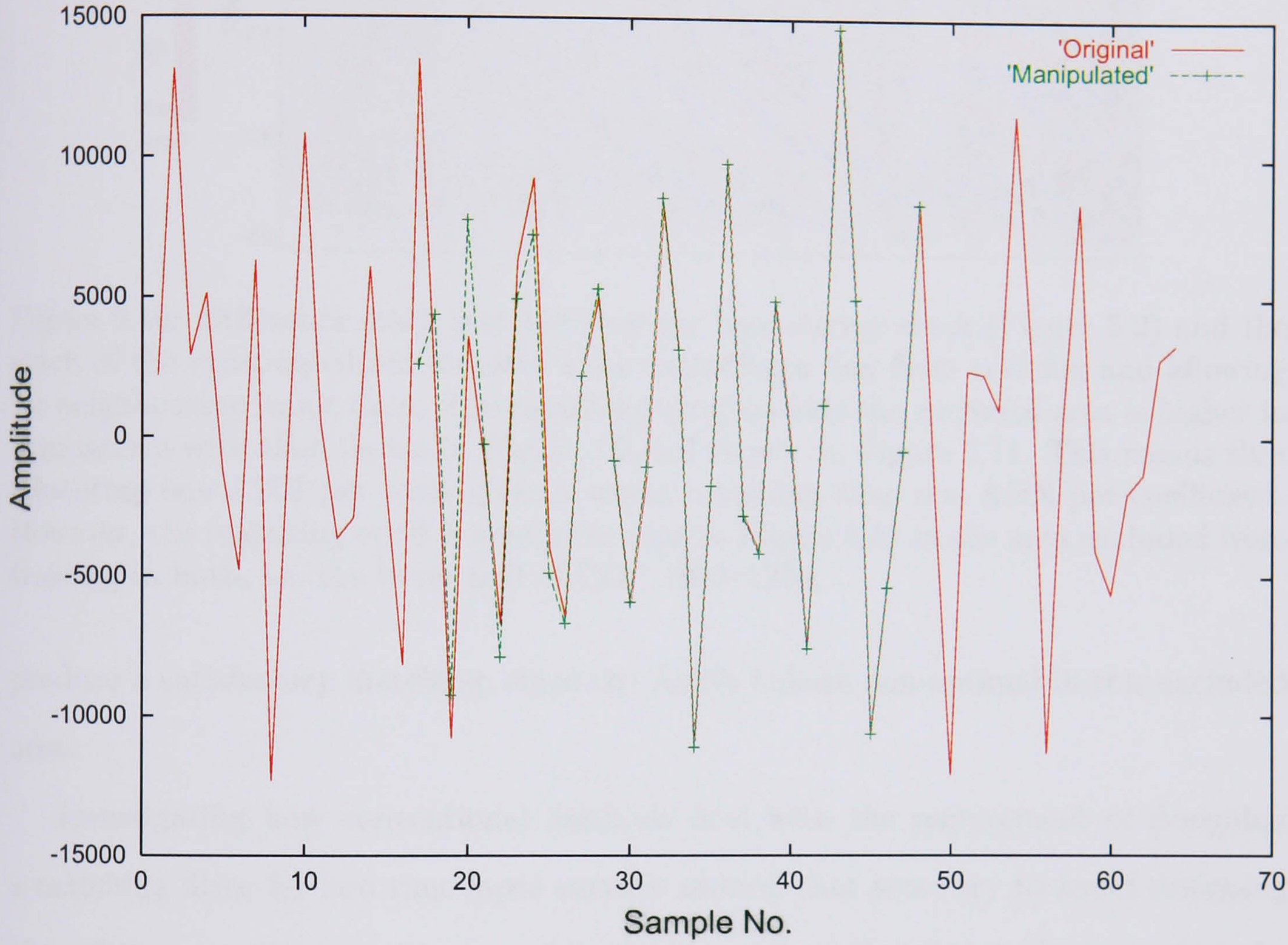


Figure 5.15: This figure shows the result of an experiment conducted to test the ability of reproducing part of a time series from a sub-set of a wavelet series. A time series of 64 samples were transformed to the wavelet domain. The wavelet series has its coefficients set to zero at $\frac{1}{4}$ of the way from either end of the time axis. The manipulated series is reverse transformed and the non-zero part is plotted together with the original time series. Both series are identical with a slight edge effect seen on the manipulated series. This proves that only those wavelet coefficients that span the time of interest are required to reproduce the time series at that period of interest.

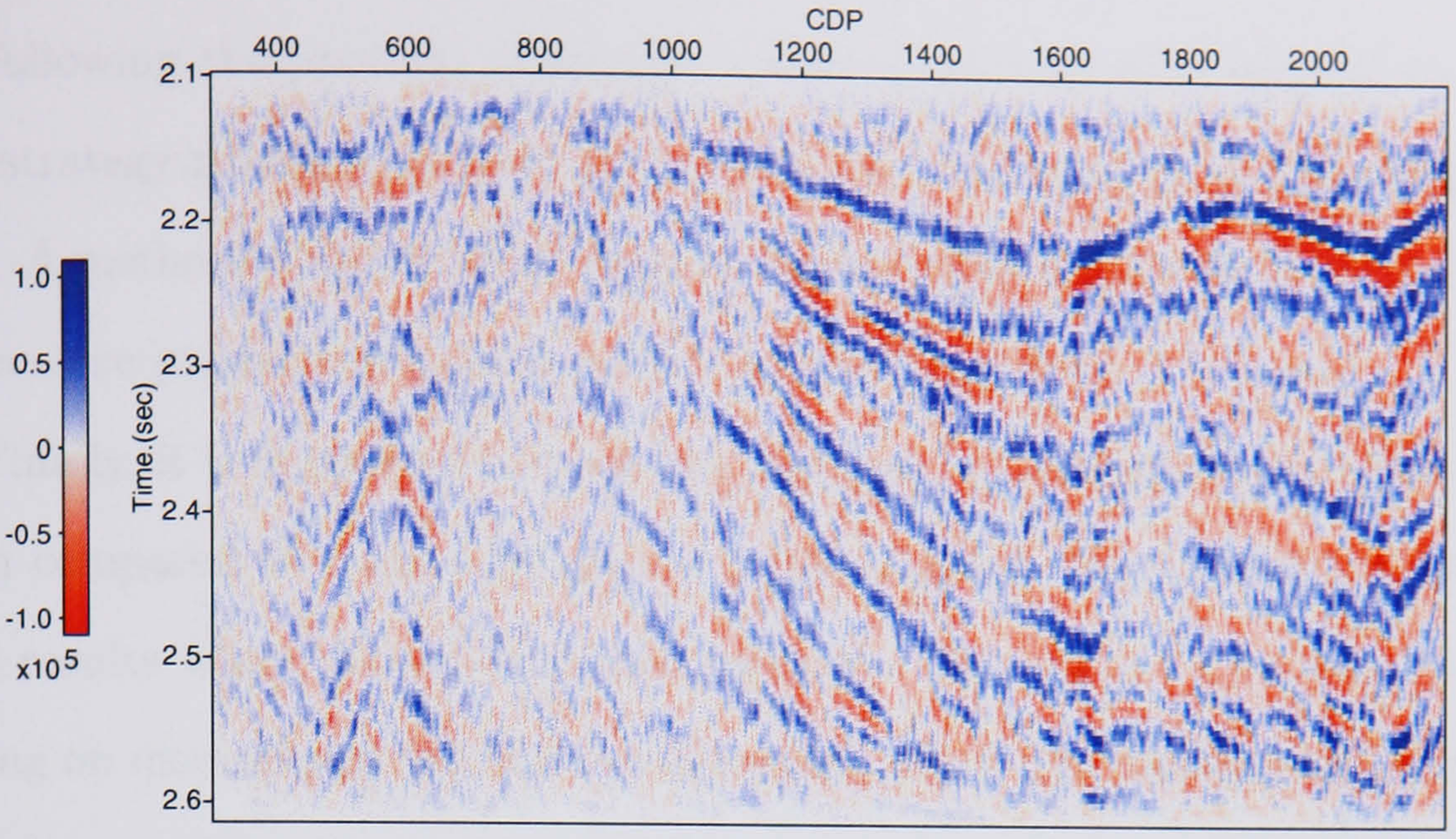


Figure 5.16: Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor after excluding a box from training and allowing for neighbouring input data. The matching error outside the excluded area is higher in comparison with that shown in Figure 5.9 and shown on Figure 5.11. This means that allocating one ANN per scale gives a worse matching than one ANN per coefficient. However, the matching error is similar to that in Figure 5.11 in the area excluded from training in both, i.e. the lower half of CDP 1000–1350.

produce a satisfactory matching, since the ANNs behave non-optimal in this excluded area.

Investigating how conventional methods deal with the requirement of designing a matching filter for two time-lapse surveys showed that some try to avoid designing those filters over the reservoir area while there are others that do not. It seems that the second approach is a widely used strategy relying on the fact that the reservoir–induced anomalies occupy only a small area of the whole seismic, and are low in amplitude, and are thus only able to bias the matching insignificantly. On the other hand, excluding the reservoir requires excluding a large area, thus deteriorating the matching capability very significantly. John Brittan (pers. com. 2001), a senior research geophysicist at the Petroleum Geo-Services (PGS) company, says that the matching capability of linear filters is enhanced significantly by including even those areas the reservoir occupies, while only biasing the matching toward obliterating production-related anomaly by an insignificant amount.

Following the previous argument, I shifted strategy with regards to the reservoir. The strategy now is to include all the data in training without excluding the reservoir area. A method is described in the next chapter that assesses the ability of the networks to preserve production-related anomalies while matching otherwise.

Finally, it is important to note that the previous examples used ANNs of large size when compared to what is suggested in Section 4.3. However, initial work gave very poor results when the topology was reduced. This could be because the ANNs were relying on memorizing the training examples, which also explains the favour for a large size and possibly the matching deterioration when an area is excluded from training. To avoid making the ANNs rely on memorizing, the cross-equalization scheme will be designed and improved to force the networks to learn a matching function. This is done by both improving on the input-output choice and trying to reduce the size of the ANN while maintaining performance. At the end, a test is conducted to make sure that the ANNs are not memorizing (see Section 4.8).

5.1.5 Final test

The last test was repeated without excluding an area from training. The new test tries to carry forward the idea of setting one ANN per scale. The intention is to produce a difference plot that can be compared with results from future tests that would not exclude any area from training.

Configuration

One ANN is allocated per wavelet scale per offset. The topology used is similar to that in Section 5.1.1 (see Figure 5.3). In addition, another smaller topology is tested with 60 nodes in the first hidden layer and 30 nodes in the second.

The target output is a wavelet coefficient C_{ij} within the scale i to be matched by the ANN concerned and is selected from the base survey. The input is selected from the monitor survey from all the 51 traces shown in Figure 4.4, where each trace supplies

the coefficient C_{ij}. However, if the trace is the direct coinciding trace r_m, or is one position away in the shot or CMP domain, then the trace supplies more coefficients. These five traces supply coefficients that span a time period of 96 ms with the coefficient concerned (C_{ij}) in the middle.

Figure 5.17 illustrates how the matching scheme is implemented on the different wavelet coefficients. The figure shows how the ideas, raised in Section 5.1.3, are implemented.

Results

The results from both topologies tested are identical. Therefore, an improvement is achieved by being able to match using a smaller topology.

Figure 5.18 shows the difference stack plot of the new test with the smaller topology. In comparison with Figure 5.16 there is an improvement in matching.

The training and application for one ANN topology design required around eight days within a 16 computer cluster, or around 120 CPU days. This is a significant improvement to the 220 CPU days of the initial test reported in Section 5.1.1.

5.2 Matching in the time domain

The use of the wavelet domain was motivated by three facts. Firstly, it simplifies the matching by allowing small parallel implementations. Secondly, it allows the selective matching of the frequencies that are of interest, whereby frequencies with low energy can be set to zero with no matching and frequencies that cannot hold information about production related changes (e.g. very low frequencies) can be copied directly from the base survey (see Figure 5.17). Thirdly, it was the plan from the start to use one ANN per scale and thus enable a network to learn an averaged (more global) matching function. That is to say the networks will not be global with respect to trace locality only but also along the time domain. This enables a network to be applicable to a reservoir when trained around it, and to be less prone to obliterating a production-related anomaly if

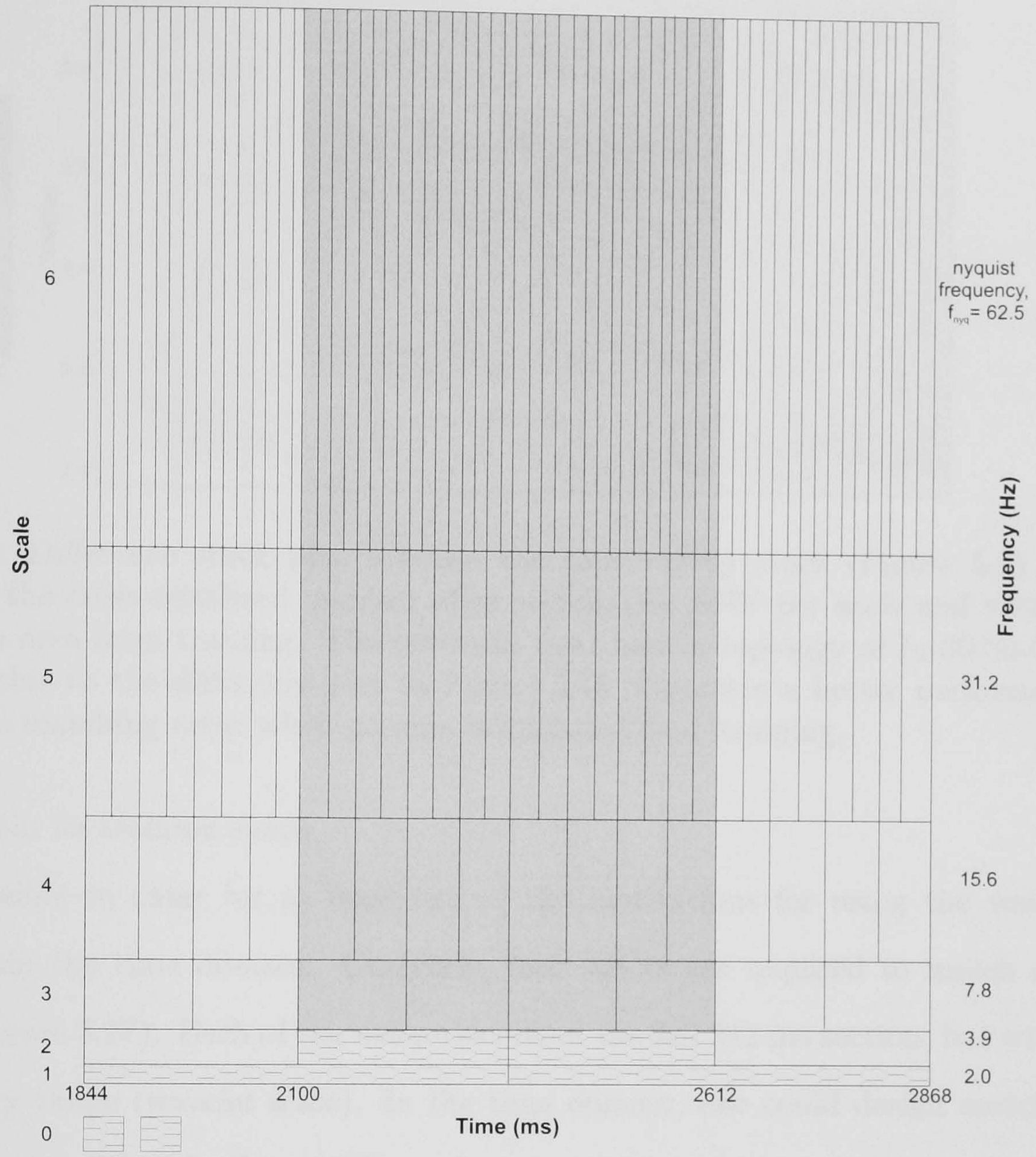


Figure 5.17: Plot of the wavelet coefficients illustrating those that are matched by ANNs. The plot follows the convention used in Figure 4.7, where each wavelet coefficient is represented by a box that spans a time and a frequency range. It is intended to cross-equalize the time section 2100–2612 ms, but all the data from 1844–2868 ms is wavelet transformed to solve boundary conditions, see Section 5.1.3. The plot is a section transformed from the base survey, where it is intended to produce a cross-equalized match to it from the monitor survey data. Since 8 ms sampling-rate and a time interval of 1024 ms is used, six wavelet scales are produced as the plot shows. Only the shaded coefficients are matched. Therefore, four ANNs are required, since there are four scales to match. The coefficients with a mesh pattern, representing the mother-function coefficients and the two lowest scales, are not matched but are directly copied from the base survey (see configuration description on Page 194). The final cross-equalized section is a reverse wavelet transform of the cross-equalized coefficients, the copied coefficients, and zeros representing the non-shaded coefficients.

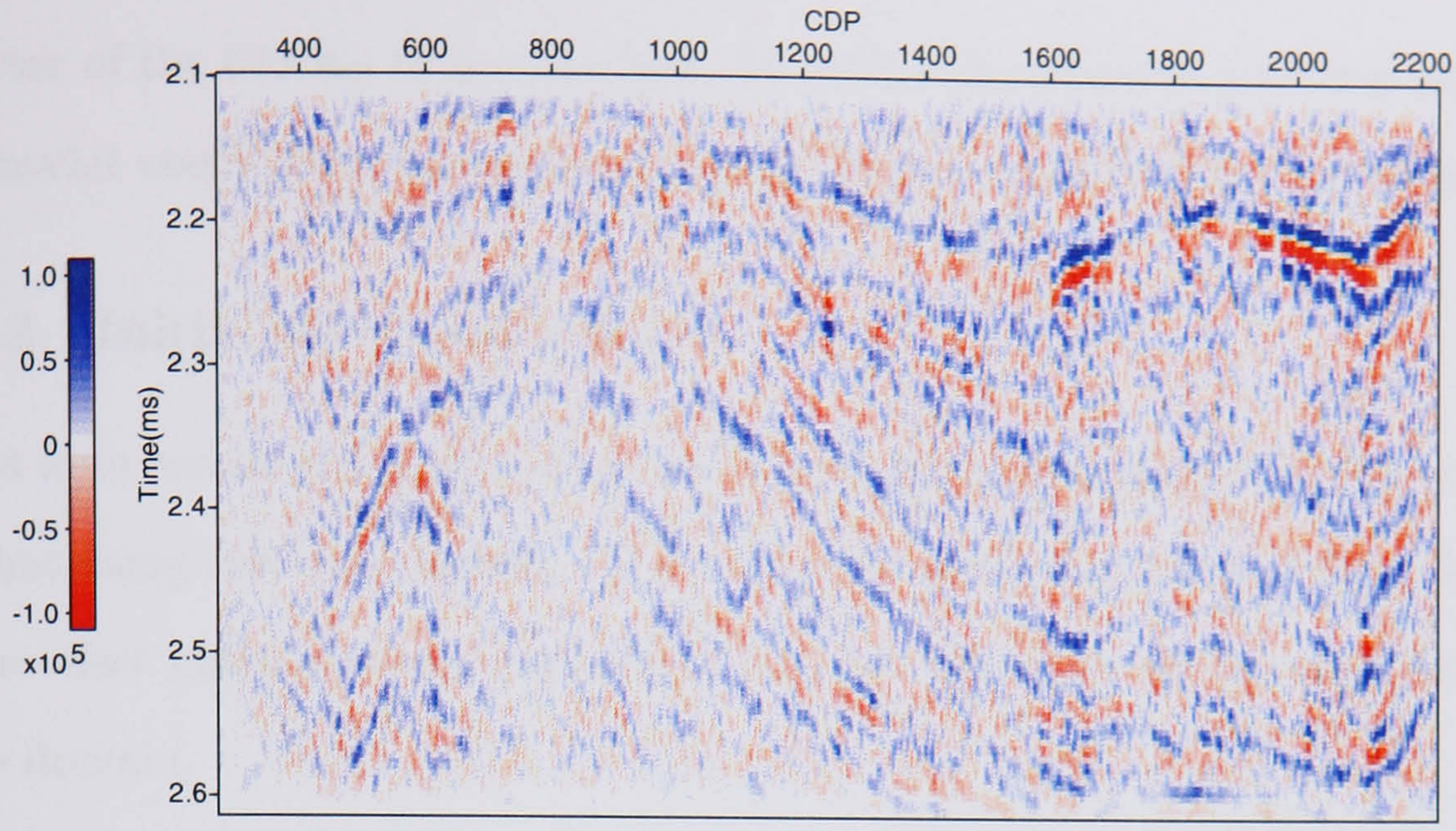


Figure 5.18: Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor after setting one ANN per scale and without excluding an area from training. The networks used have a topology of In-60-30-Out. Comparing this to the difference plot in Figure 5.16 illustrates a better performance and thus less matching error when no area is excluded from training.

it exists within its training space.

It is possible to cater for at least two of the motivations for using the wavelet domain within the time domain. Currently, four ANNs are required to match each offset (see Figure 5.17). Each of the networks match the full 512 ms section, but within one frequency range (wavelet scale). In the time domain, one could design matching to be also accomplished by four ANNs each assigned to a quarter of the time section 2100–2612 ms. Four ANNs would allow the networks to work with the same kind of topology and matching load while allowing each network to be global. If the network's performance is similar or better in the time domain, then the time domain is favourable as it is more intuitive and simpler to use. Later, the use of one ANN for the 512 ms time-section would be desired. This is to allow a better global matching and, if topology is maintained, a smaller computational load.

5.2.1 Configuration

While using the time domain, the configuration is similar to that of the wavelet domain (Section 5.1.5) with few differences. Firstly, an ANN is allocated per offset gather per

quarter of the 512 ms to be matched. Secondly, the patterns are time samples instead of wavelet coefficients, but otherwise their choice is identical to that in Section 5.1.5.

5.2.2 Initial tests and results

Tests were conducted to compare results from ANN matching using the wavelet domain to that using the time domain. The tests were conducted by matching the three time-lapse offset gathers of channels 121, 173, and 225 using the wavelet domain and the time domain.

The results indicate that the time domain offers a better matching. This is clear from the comparison of the offset gather difference plots of the wavelet domain and the time domain, shown in Figure 5.19 and Figure 5.20 respectively. The plots correspond to the offset gather of channel 225, but similar results were seen when comparing the other two offset gather differences.

5.2.3 Further test and analysis

The experiment was repeated at all offsets that were used previously in Section 5.1.5. Figure 5.21 shows the difference plot between the base survey stack and the cross-equalized stack of the monitor. The figure confirms that the time domain gives a better matching.

Figure 5.22 compares predictability and normalized root mean square (NRMS) error of the matching done here with that in Section 5.1.5. This figure also illustrates the superiority of time domain matching. Matching in the wavelet domain might be inferior because each error within one scale is propagated and added to the error of other scales when the cross-equalized series is reverse transformed.

5.2.4 Conclusion

It is therefore more desirable to use the time domain for future tests. However, it should be noted that both domains can be treated similarly in many regards (e.g. designing

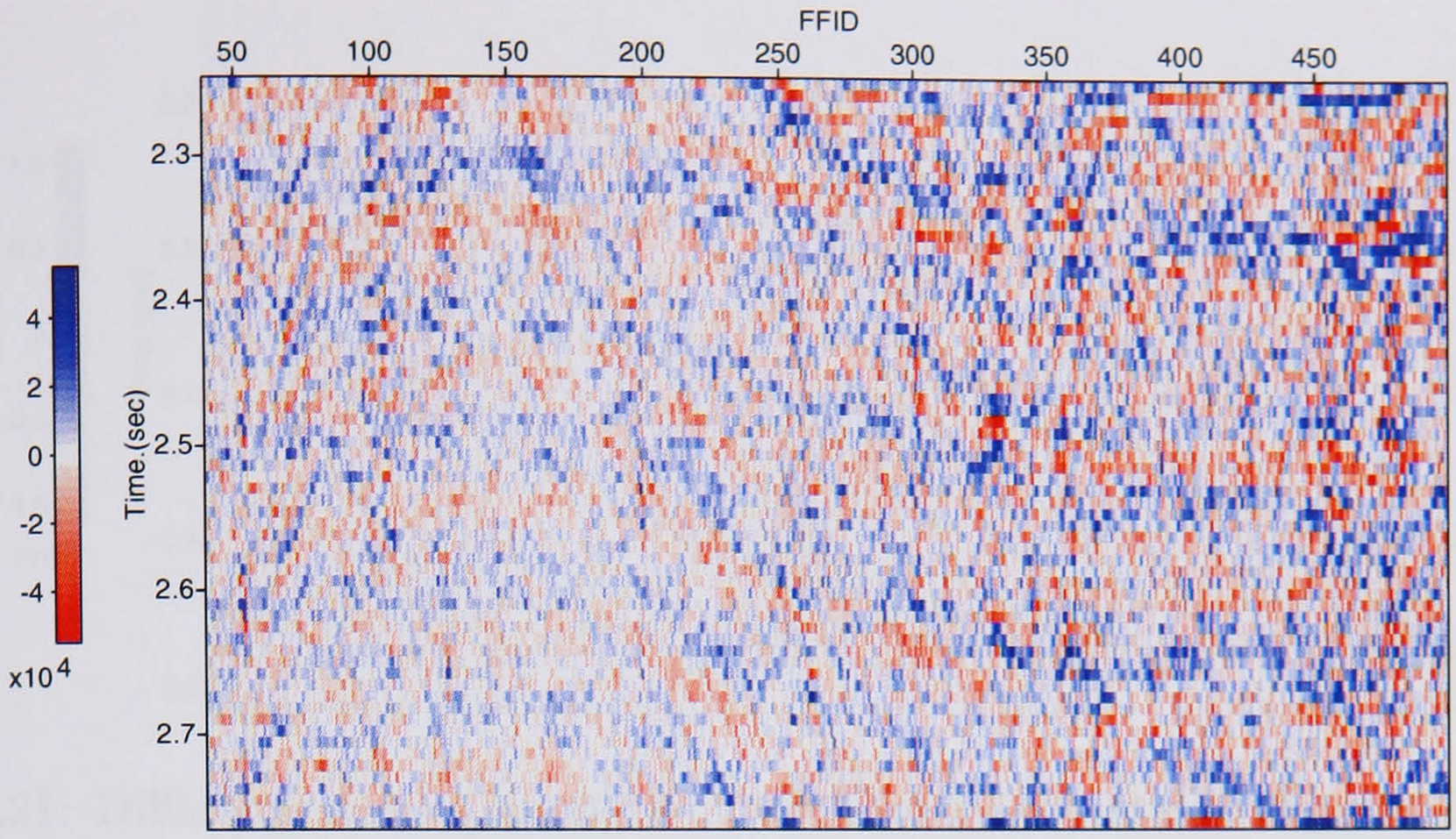


Figure 5.19: Difference plot of the offset gather corresponding to channel 225 between the base survey and the cross-equalized monitor survey matched using the wavelet domain. The matching produced a higher error here in comparison with that shown in Figure 5.20 where the time domain was used.

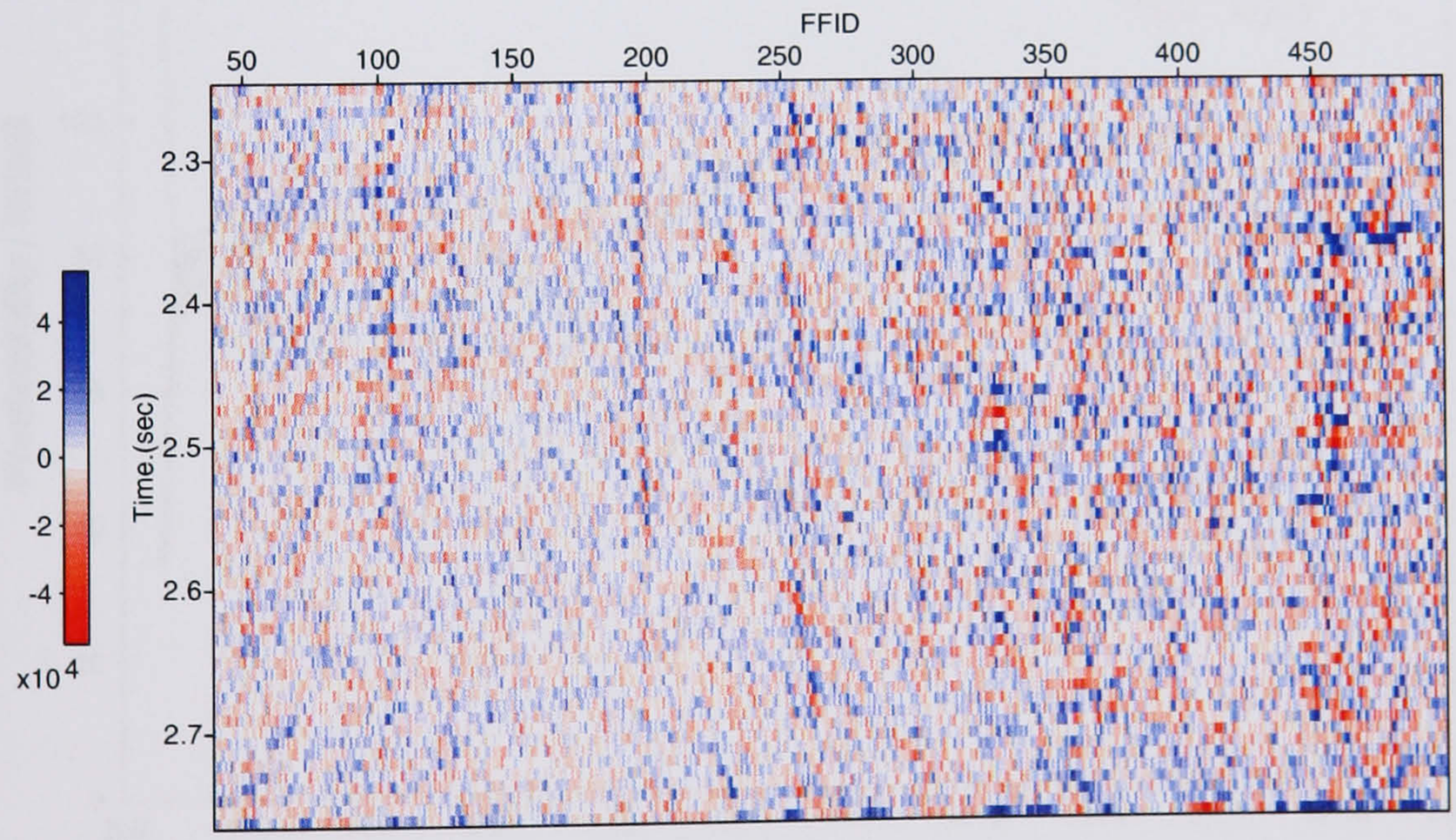


Figure 5.20: Difference plot of the offset gather corresponding to channel 225 between the base survey and the cross-equalized monitor survey matched using the time domain. The matching produced a lower error here in comparison with that shown in Figure 5.19 where the wavelet domain was used.

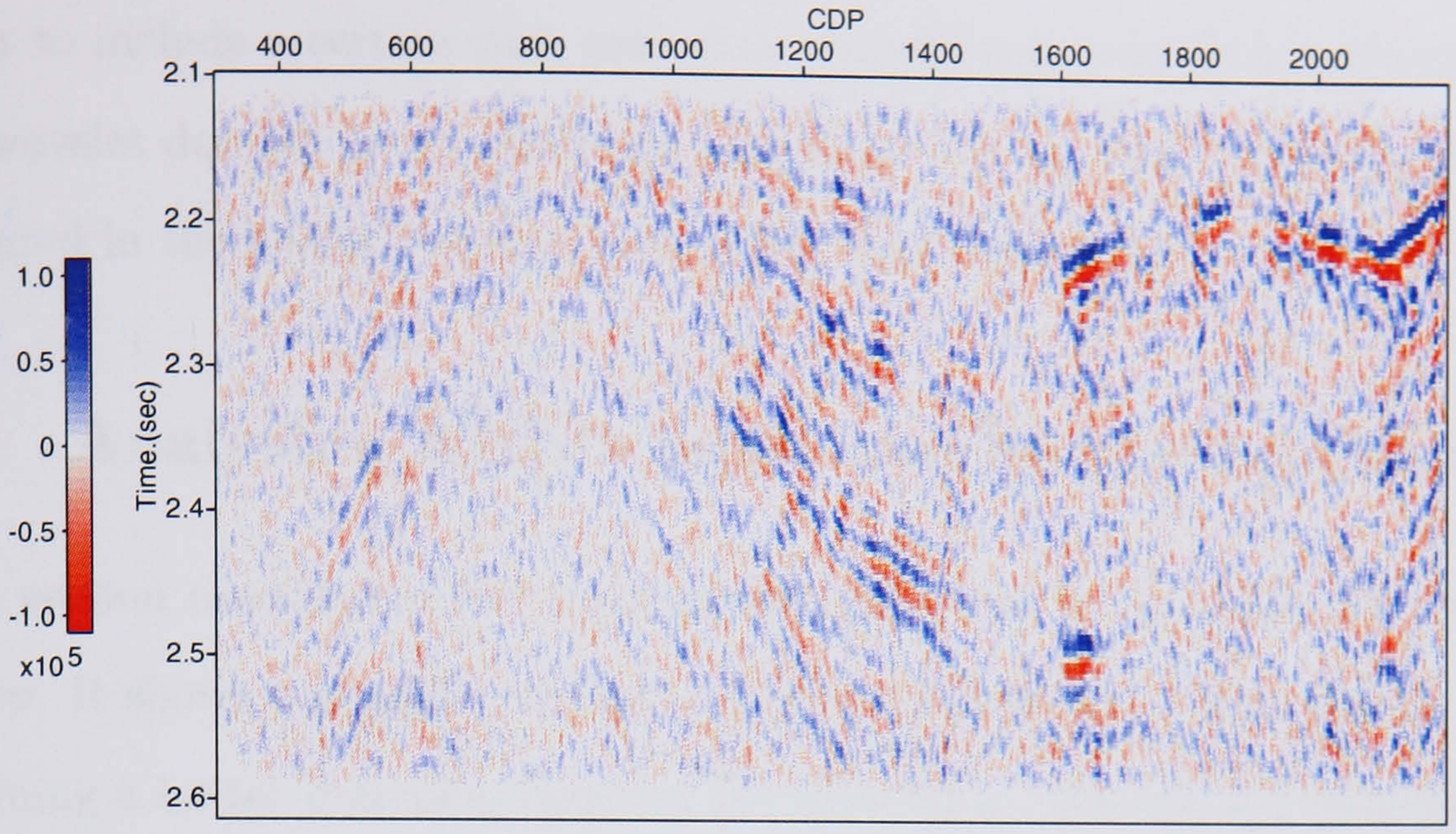


Figure 5.21: Difference stack plot between the base survey stack (Figure 5.2) and the stack of the cross-equalized monitor using the time domain. This can be compared to Figure 5.18, where matching was done using the same configuration but in the wavelet domain. The anomaly seen at time 2300–2500 ms CDP 1000-1400 should be ignored, as I reserve its discussion to the next chapter (Section 6.1.3). The plot here shows a better performance of cross-equalization than that shown in Figure 5.18. Therefore, the time domain gives a better performance.

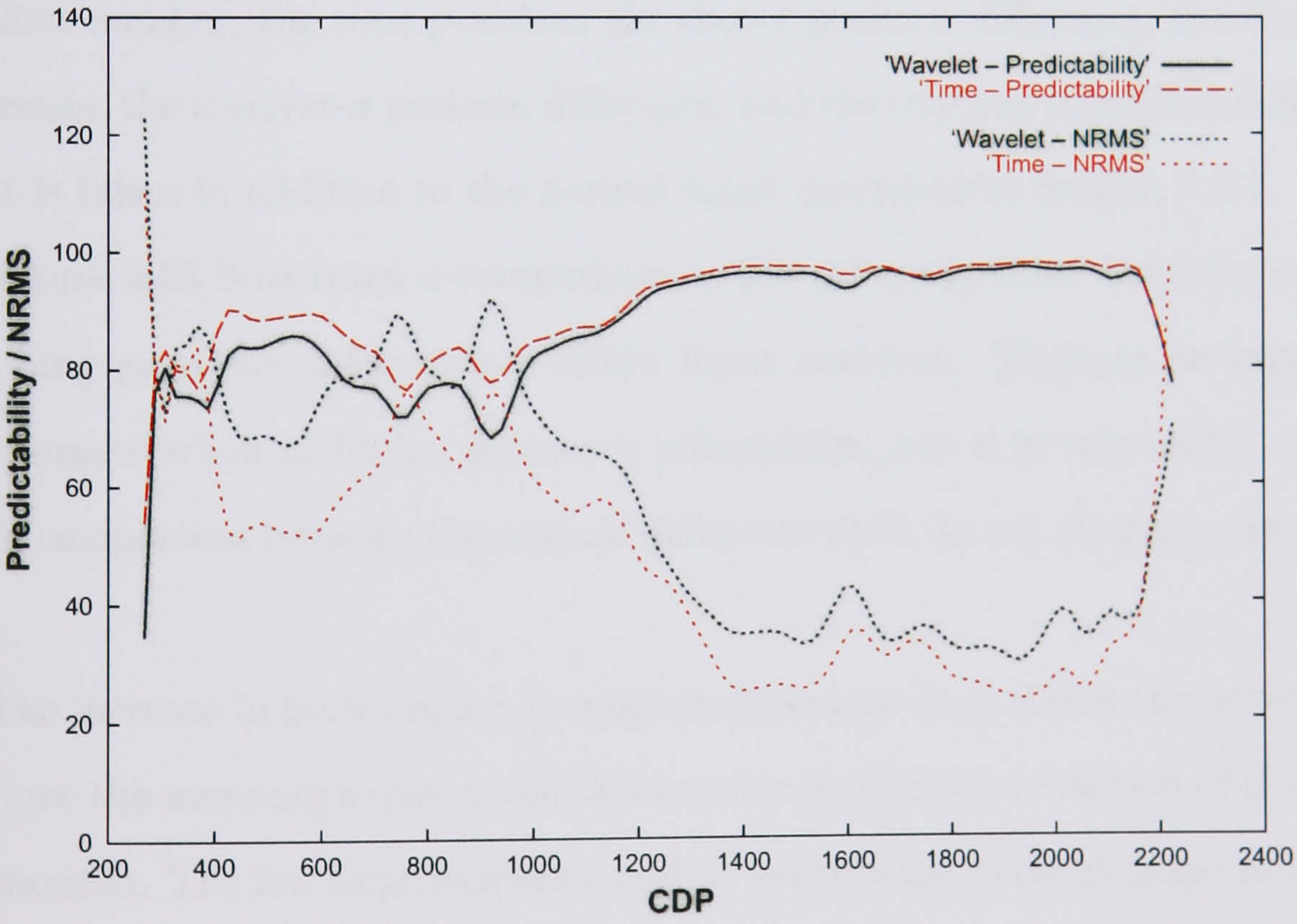


Figure 5.22: Plot comparing predictability and NRMS of the two matching schemes: time domain scheme versus wavelet domain scheme. Predictability and NRMS are calculated between the base survey stack and the cross-equalized stack for the two schemes. Predictability is higher and NRMS is lower when the time domain was used. Therefore, the time domain is favoured.

input to include a certain time period or excluding an area from training). Therefore, the wavelet domain is not ruled out as a viable option, but rather the time domain is favoured in the analysis of this novel ANN implementation.

5.3 Analysing ANN's input and increasing performance

This section describes a few experiments conducted to optimize the network's input choice. It shows a progress towards a better ANN design, where "better" means both attaining a better matching between the time-lapse datasets and reducing the computational time required.

5.3.1 Geometrical information

A project is defined to test the effect of including, as an input, information about the relative position of the output trace r_b to the input trace r_m. The information includes: the shot number, the time position, the shot-x position difference, the shot-y position difference, the receiver-x position difference, and the receiver-y position difference. This input is taken in addition to the normal input described in Section 5.2.1.

Figure 5.23 illustrates a comparison in the matching error between including and excluding geometry information within input patterns. There is an improvement in performance when including geometry information, but it is very small. On the other hand, comparison between the seismic difference plots do not show much improvement.

The increase in performance is somewhat obvious as it allows the network to recognize how the mapping between the two surveys changes as a function of the geometrical information. The low improvement could be because the network is gathering the information from the other input data, where it is realizing that a certain pattern indicates that a certain mapping is required. However, since there is an improvement, and since the geometrical information constitute a small number of input parameters and will not therefore increase training time significantly, it is beneficial to include them. In

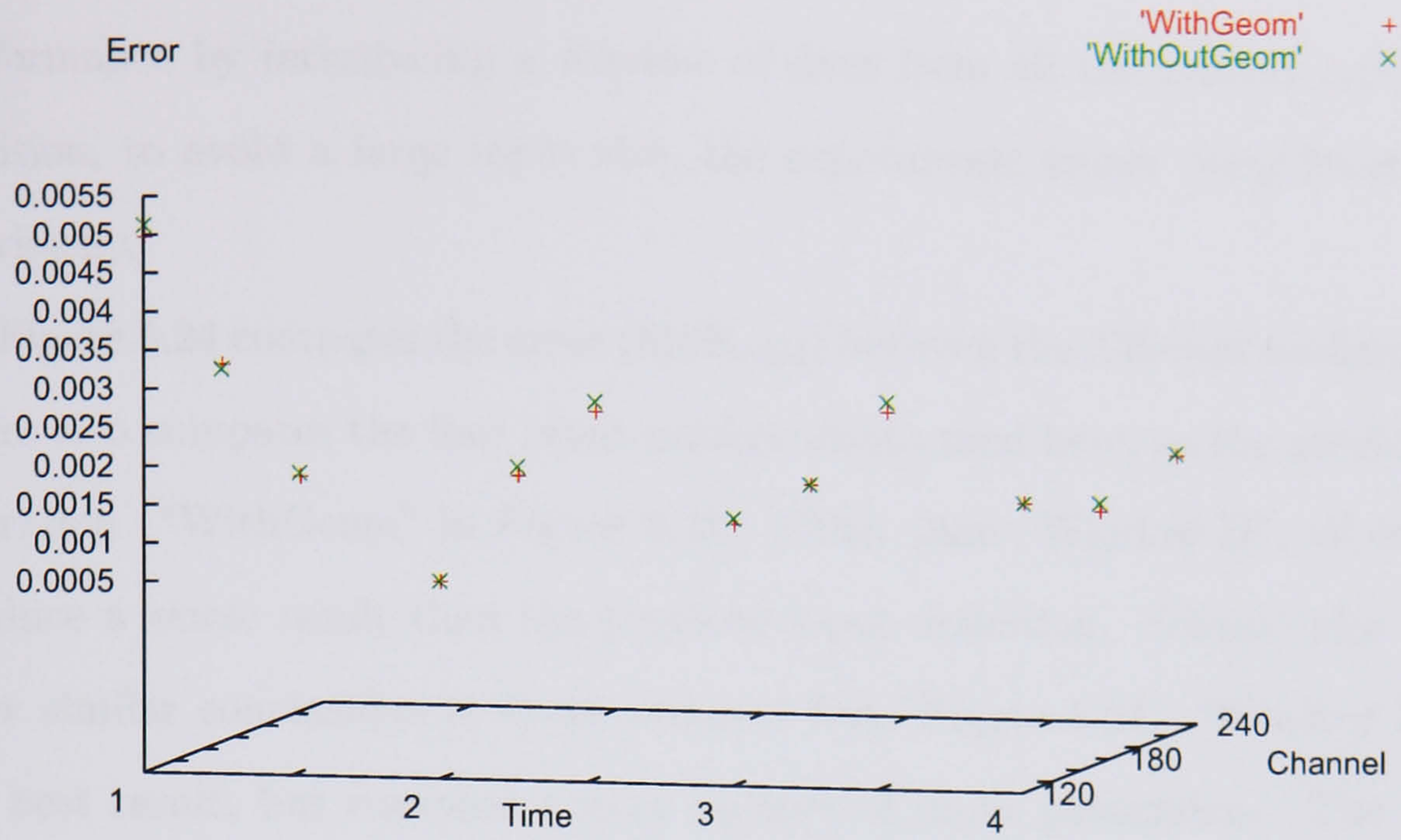


Figure 5.23: Error (MSE_{valid}) plot comparison between including and excluding geometry information within input patterns. The error is plotted against offset gather matched (channel) and time period. The axis labelled "Time" signifies position in time, where the section 2100–2612 ms is split into four quarters (labelled 1–4) each with one network allocated to it. Geometry information reduces the error, but only by a very small amount.

addition, the next section shows that this information is indeed important to the network.

5.3.2 Comparing different input-configurations

Previously, the input was defined to be one sample from each of the neighbouring traces to r_m (Figure 4.4) and a window of samples from the trace r_m itself and a few other traces. In addition, the previous section introduced geometry as an additional input. Here, an experiment is designed to compare four alternative ways of defining input.

In this experiment, the four configurations contain six input parameters representing the geometry information as defined in the last section. In addition, given a time position n of a target output, the input has a 13 sample-window centred at n from trace r_m and a five sample-window from the neighbouring traces (Figure 4.4). The difference is how many traces are involved. Table 5.1 defines the neighbouring traces involved in each of the four configurations. The idea in this experiment is to increase

performance by introducing a window of data from all the related input traces. In addition, to avoid a large input size, the experiments assess using fewer traces than previously.

Figure 5.24 compares the error (MSE_{valid}) between the different configurations. The figure also compares the four input configurations used here, to the previous best configuration ("WithGeom" in Figure 5.23). Other than "Window D", all configurations produce a worse result than the previous input definition. Seismic plot comparisons show similar conclusions to those attained from Figure 5.24. "Window D" produces the best result, but contains a large number of input parameters. The next section explains how the performance of "Window D" is maintained while also reducing the number of input parameters.

5.3.3 Connection weight analysis

The discussion, given in Section 2.4.5, shows that by analysing the magnitude of the connection weights of a trained network, given the use of weight decay during training as in this study, allows one to identify the important inputs from the unimportant. This analysis was performed at different stages. This section describes one instance where this proved to be very useful.

The input configuration of "Window D" shown in the previous section gave the best result. However, its large input size of 269 parameters as opposed to around 100, which was used previously in projects that give a comparable performance, makes it

Table 5.1: Traces involved in the four input-configurations tested in the Window project. The gathers are abbreviated as follows: CS is common-shot; CMP is common-midpoint; CO is common-offset; and CR is common-receiver.

Project name	Gathers involved	Gather positions relative to trace r_m
Window A	CS, CMP, CO, and CR	-1 to +1
Window B	CS, and CMP	-1 to +1
Window C	CS, and CMP	-2 to +2
Window D	CS, CMP, CO, and CR	-2 to +2

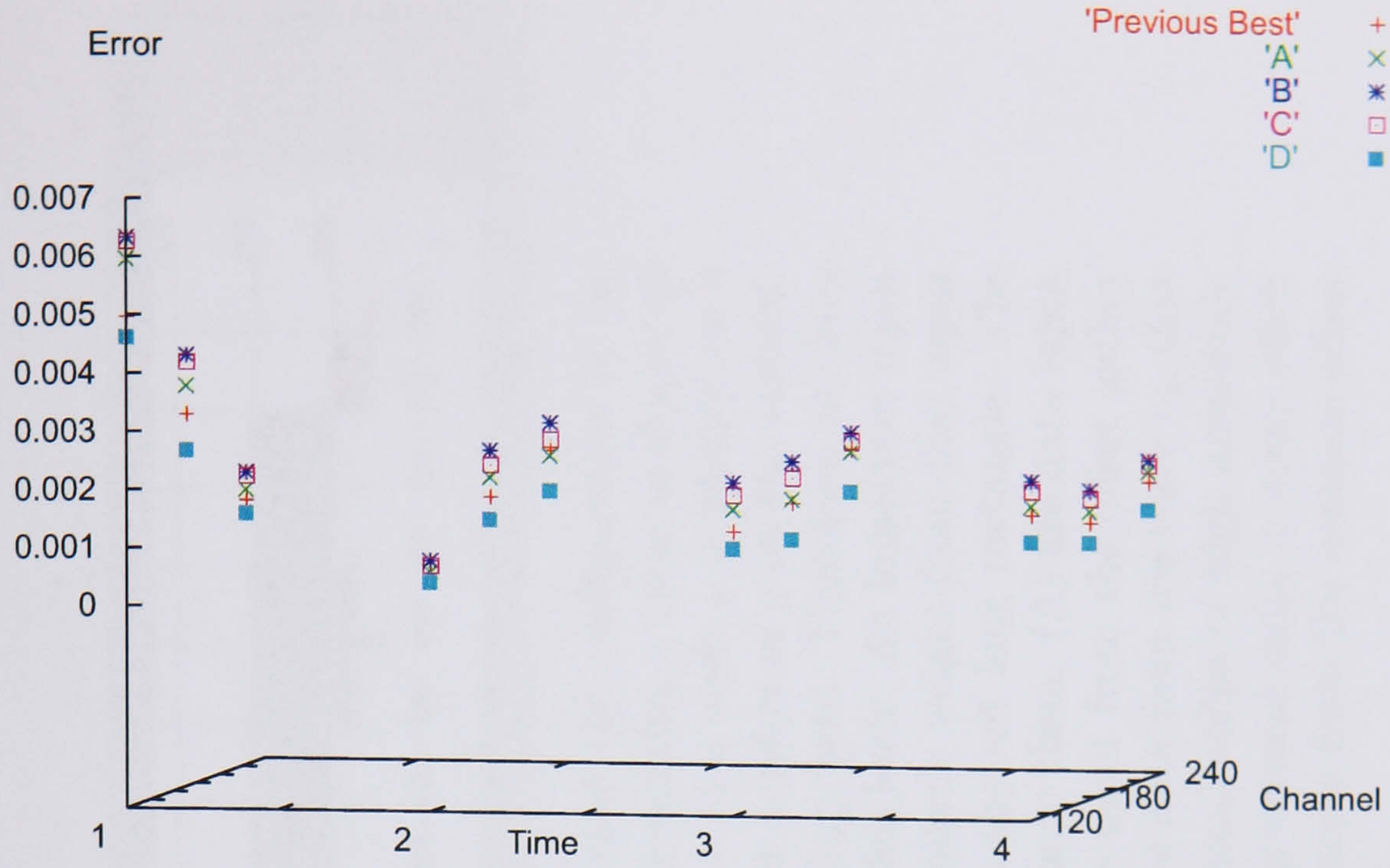


Figure 5.24: Error (MSE_{valid}) plot comparison between the different input configurations shown in Table 5.1 and the best configuration prior to this experiment "Previous Best". The error is plotted against offset gather matched (channel) and time period. The axis labelled "Time" signifies position in time, where the section 2100–2612 ms is split into four quarters (labelled 1–4) each with one network allocated to it. Clearly, "B" is the worst in error and "D" is the best.

unfavourable to use. This is because training time and memory space required to store pattern files are increased largely (more than double). An attempt is made at analysing the weights of trained ANNs that used the input configuration of "Window D" in the hope that important inputs can be distinguished from unimportant ones.

The weight analysis is conducted on many ANNs, which were trained for matching different offset gathers and different time periods within the matched section 2100–2612 ms. Figure 5.25 shows one such analysis. In all analyses it is found that the inputs favoured by the network come from the common-offset gather traces in addition to the geometry information.

5.3.4 Assessing the results of the weight analysis

Taking the results of the weight analysis, the input to the networks is designed to comprise only the ones favoured by the network: geometry information and window-samples from traces coming from the common-offset gathers. Thus, the configuration is

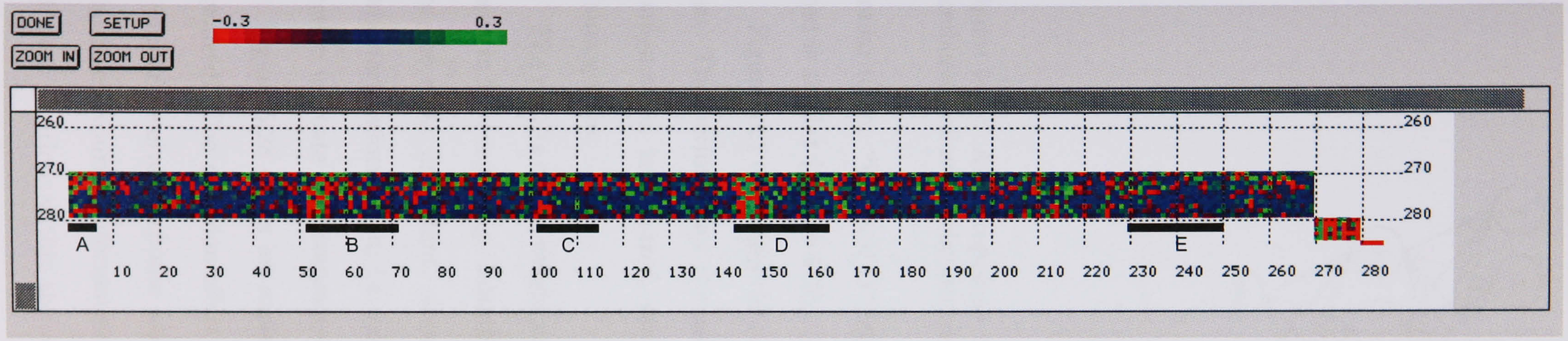


Figure 5.25: Analysis of the connection weights of a trained ANN that used the configuration of the "Window D" experiment. Each connection weight is shown as a colour representing a value as shown on the legend given on the top. The weight of the connection from a node x to a node y is plotted on a grid where its position is the node number of x on the horizontal and node number of y on the vertical. The nodes are numbered as depicted in Figure 5.26, which shows the topology used. Therefore, the long rectangle contains all the connections from the input layer to the first hidden layer. An important input to the problem the network is solving will appear as a high absolute magnitude weight from that input to nearly all the first hidden layer, i.e. it will show as a vertical strip on the left long rectangle. The dark black boxes are introduced as labels highlighting the vertical strips above them. (A) are data input corresponding to the geometrical information. (B), (D), and (E) are data input from the offset gather traces from the lower-y, middle and higher-y streamers respectively. (C) are data from the trace r_m that closely matchs the position of the target output trace r_b. There are vertical strips of high amplitude weights above (A), (B), (C), and (D). Although (E) does not show a strong vertical strip, in many other trained networks it does. Therefore, the inputs favoured by the network come from the common-offset gather traces (including from r_m) in addition to the geometry information.

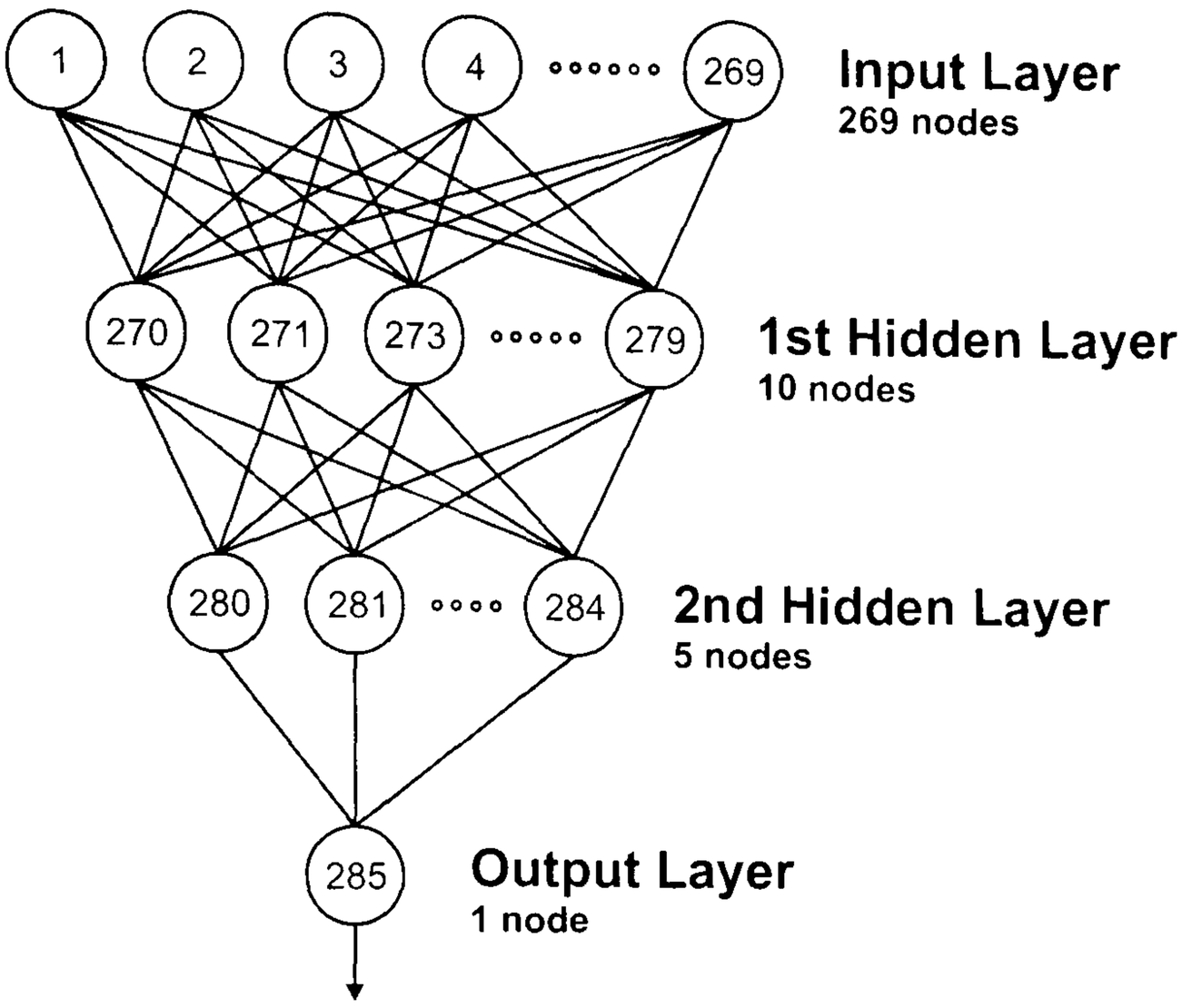


Figure 5.26: ANN topology used in the "Window D" experiment. The nodes are numbered following the convention used in Figure 5.25.

similar to the "Window D" experiment except for the input. In addition, the topology used is similar (Figure 5.26), but with an input size of 89.

Figure 5.27 compares the result of this experiment to that before reducing the input size. The performance is similar to the best in the last experiment. Therefore, this configuration is favoured, since it has smaller input size, and thus requires shorter training time.

The results of the weight analysis and the experiment that followed it (Figure 5.27) seem to be surprising. Initially, it was thought that CMP and common-shot gathers are the most important, since those two provide what was thought to be the most important information, i.e. geology and source respectively. However, results in this section indicate that common-offset gathers are favoured by the network.

Figure 5.28 offers an explanation to why input from the common-offset traces is favoured. The common-offset gather traces not only share a common offset but also a similar receiver response since they all come from the same channel. The network might be favouring information that have most in common. In addition, input from the different positions within the common-offset gather enables scanning an area of geology

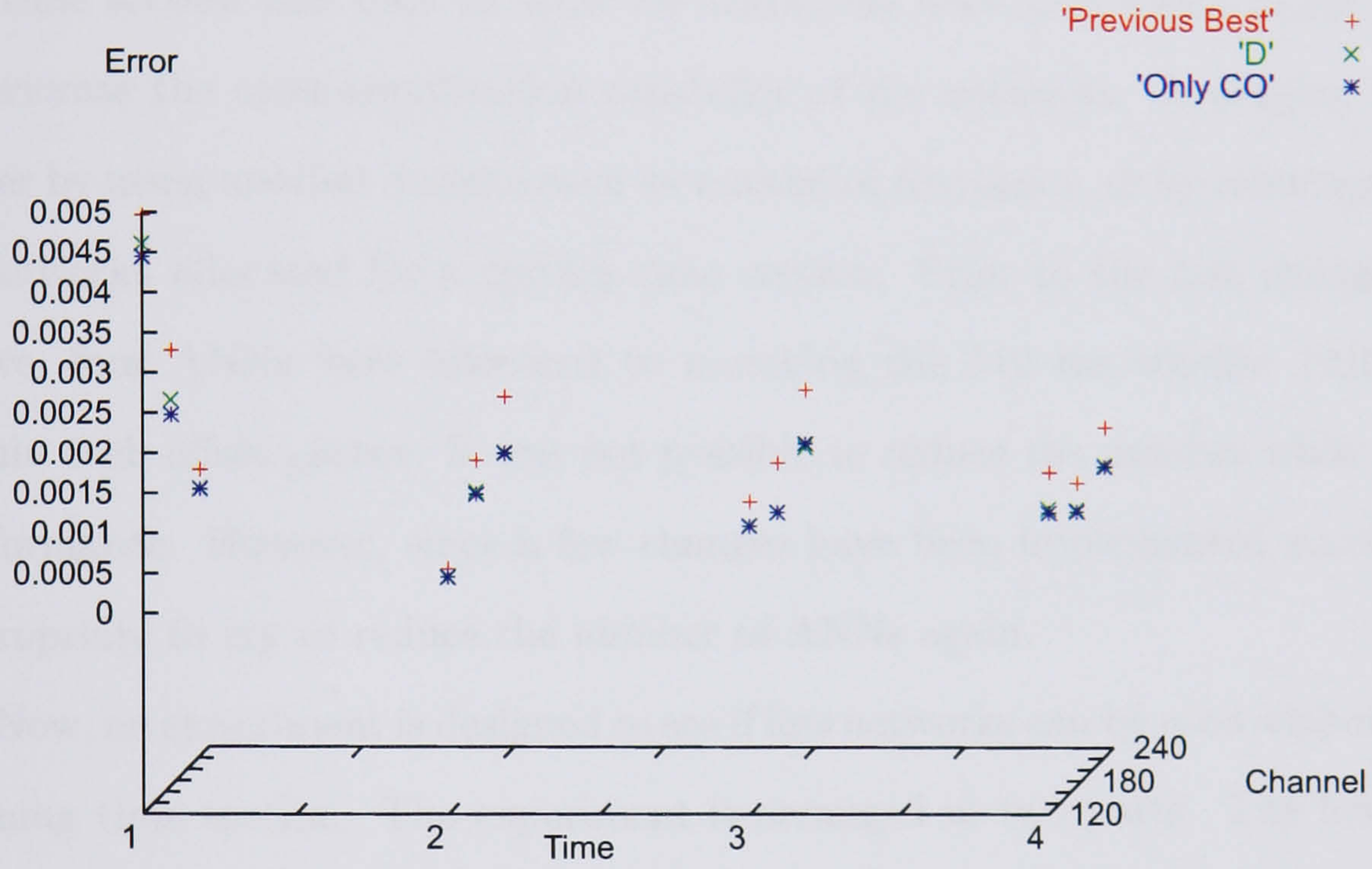


Figure 5.27: Error (MSE_{valid}) plot comparison between different input configurations before and after using only common-offset traces for input. The error is plotted against offset gather matched (channel) and time period. The axis labelled "Time" signifies time, where the time section 2100–2612 ms is split into four quarters (labelled 1–4) each with a network allocated to it. The experiments shown are that before the window experiment "Previous Best", the last with the most traces used in the Window experiment "D", and the experiment with only common offset gather input "Only CO". The matching in "Only CO" has similar error to "D".

and hence gives the network enough information to enable choosing the information most suitable and relevant to the output.

5.3.5 Reducing the number of ANNs per offset

One of the aims when assessing different input configurations is to reduce the number of ANNs required. This is to reduce the computational load and to enable averaging over the time section and thus to allow for horizontal reservoirs which might significantly deteriorate the cross-equalization capability of the networks. Averaging can be done either by using another domain such as wavelet or frequency, or by reducing the number of networks allocated for a certain time section. Prior to the last changes described above, four ANNs were allocated to matching the 512 ms window (2100–2612 ms) within each offset gather. It was not possible to reduce the number while maintaining performance. However, since a few changes have been implemented successfully, it is appropriate to try to reduce the number of ANNs again.

Now, an experiment is designed to see if less networks can be used within the selected training time section. The experiment is arranged in two parts. The first introduces two networks within the 512 ms and the second only one network. The configuration is otherwise similar to that in the last experiment.

Figure 5.29 compares the results of this experiment with previous results. Both cases of using one and two ANNs per offset behave similarly to previous results, and in fact better in some instances. Since one ANN per offset reduces computation significantly, it is chosen for future designs.

5.3.6 Final test

The results shown on the last few sections are promising. The conclusion of good performance was supported by analysing the error of the ANN matching to three offset gathers. In addition, the ANNs were applied to produce matched offset gathers. Analysis on these gathers show similar results attained through the analysis of the error

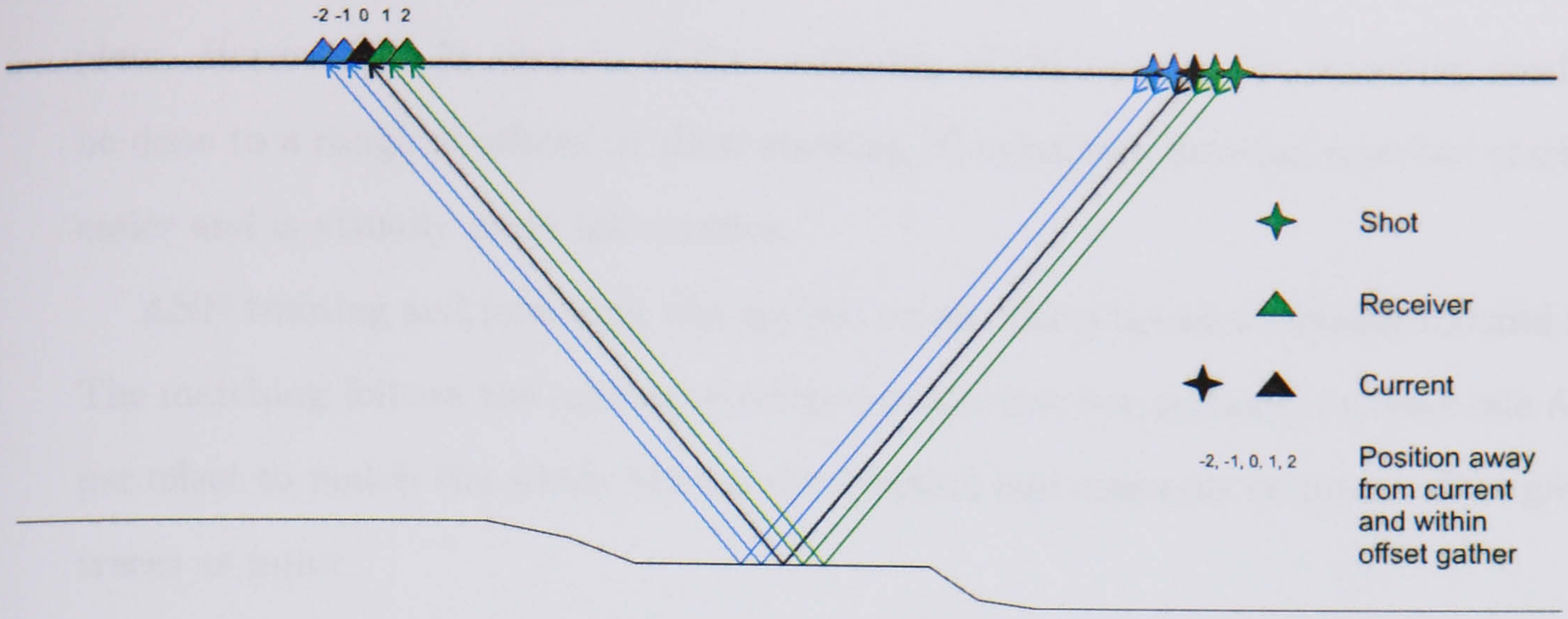


Figure 5.28: Diagram shows how neighbouring traces from the same offset gather scan a strip of geology. Furthermore, by also including offset gather neighbouring traces from the two other streamers the geology scanned is within a 3D area. If these are included as input to a network, the most relevant information can be chosen that closely matches the target. In addition, those traces come from the same channel within the gather and hence have identical receiver response.

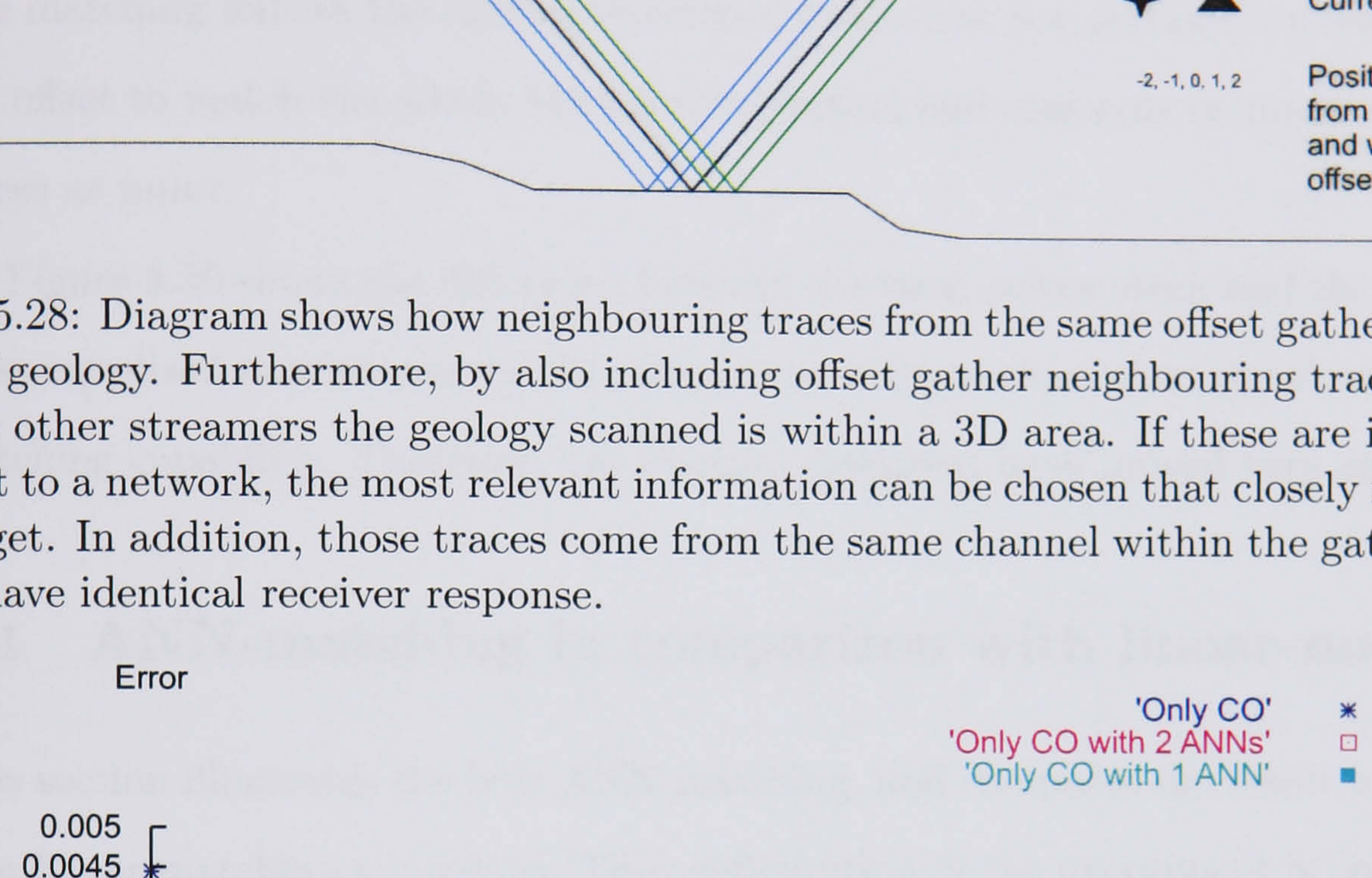


Figure 5.29: Error (MSE_{valid}) plot comparison between different input configurations before and after using only common-offset traces for input and one ANN within 512 ms. The error is plotted against offset gather matched (channel) and time period. The experiments shown are "Only CO" with only common offset used for input but otherwise similar configuration to "Window D", and this experiment with only common offset gather input with two networks "Only CO with 2 ANNs", and with one network "Only CO with 1 ANN". The axis labelled "Time" signifies position in time, where the section 2100–2612 ms is split into four quarters (labelled 1–4) each with one network allocated to it in "Only CO". The position of the plot on the "Time" axis for the two latest experiments "Only CO with 1 ANN" and "Only CO with 1 ANN" are arbitrary. This experiment has similar error to "Only CO".

plots. However, to be certain of the credibility of the results, the matching needs to be done to a range of offsets to allow stacking. Comparison between matched stacks is easier and is visually more informative.

ANN training and matching was applied on all offsets between channel 100 and 230. The matching follows the scheme developed in the last few sections, i.e. uses one ANN per offset to match the whole 512 ms time-section and uses only common-offset gather traces as input.

Figure 5.30 shows the difference between the base survey stack and the stack of the cross-equalized monitor survey. In comparison with previous plots this shows a superior matching capability. Therefore, the changes discussed have proved very useful.

5.4 ANN-matching in comparison with linear-matching

This section illustrates the best ANN-matching, and compares the result to those done by a linear matching procedure. The configuration of the experiment is chosen so that it incorporates all the optimizations discussed in this chapter and Chapter 4. This allows a better judgement of the results than through previous experiments where a few aspects were not optimal. In addition, this section serves to consolidate all that has been discussed so far in terms of optimal configuration.

5.4.1 Matching using ANNs

Configuration

The matching is done in the time domain where one ANN is allocated per offset for the whole 512 ms (the time section 2100–2612 ms). In addition, normalization of patterns is done as suggested in Section 4.4.1, topology as suggested in Section 4.3, and training as described in Section 4.5. The ANN topology used is illustrated in Figure 5.31.

The training patterns are created as follows: for each sample at time t, offset m, and CDP x, a possible training pattern is created. The target output is the sample at time t from a trace r_b at CDP x and offset m from the base survey. The input consists

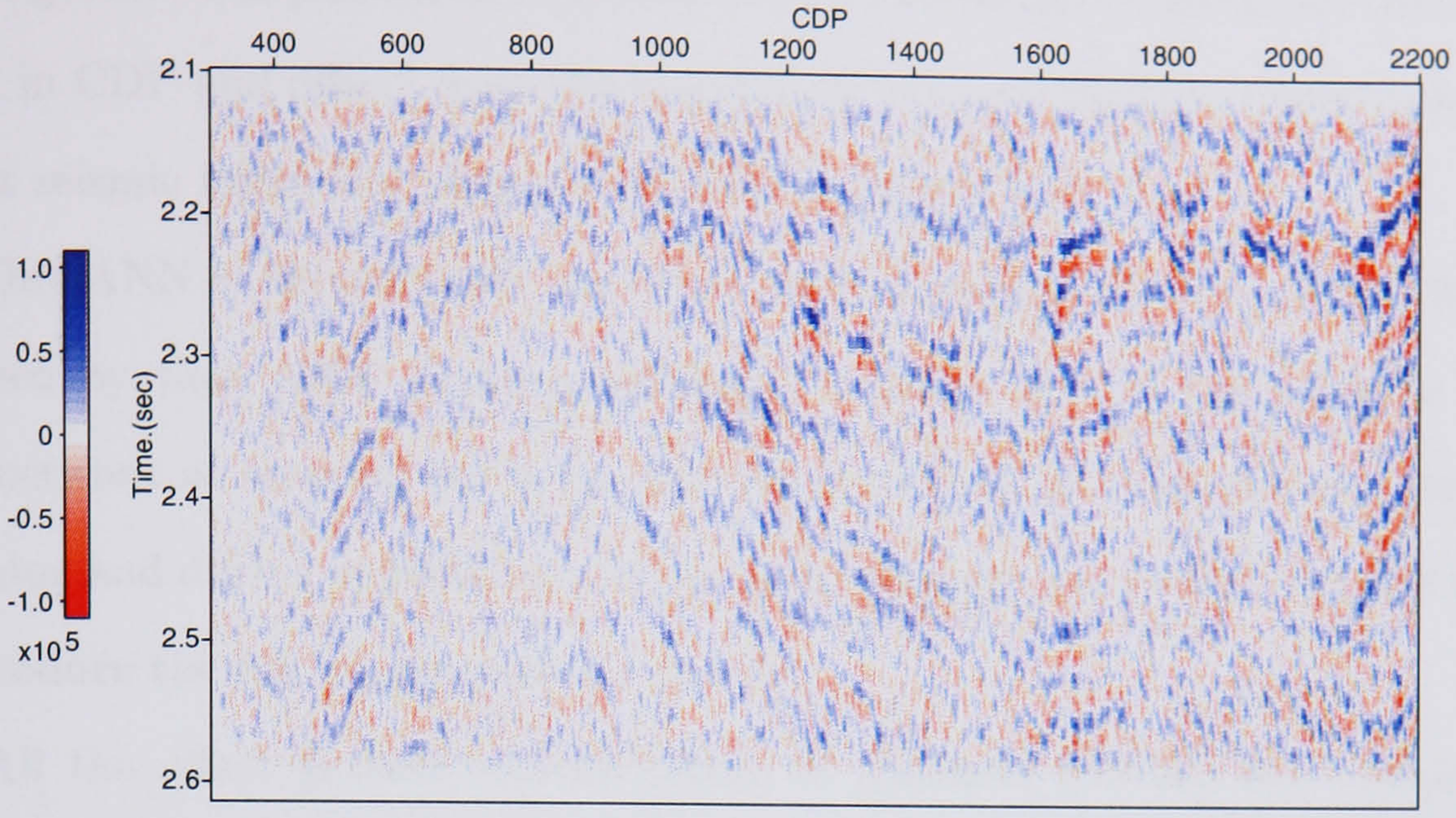


Figure 5.30: Difference between the base survey stack and the stack of the cross-equalized monitor survey when only common-offset traces are used for input and one ANN within 512 ms. The anomaly seen at time 2300–2500 ms CDP 1000-1400 should be ignored, as I reserve its discussion to the next chapter (Section 6.1.5). The network configuration here produces a better matching to that illustrated in Figure 5.21

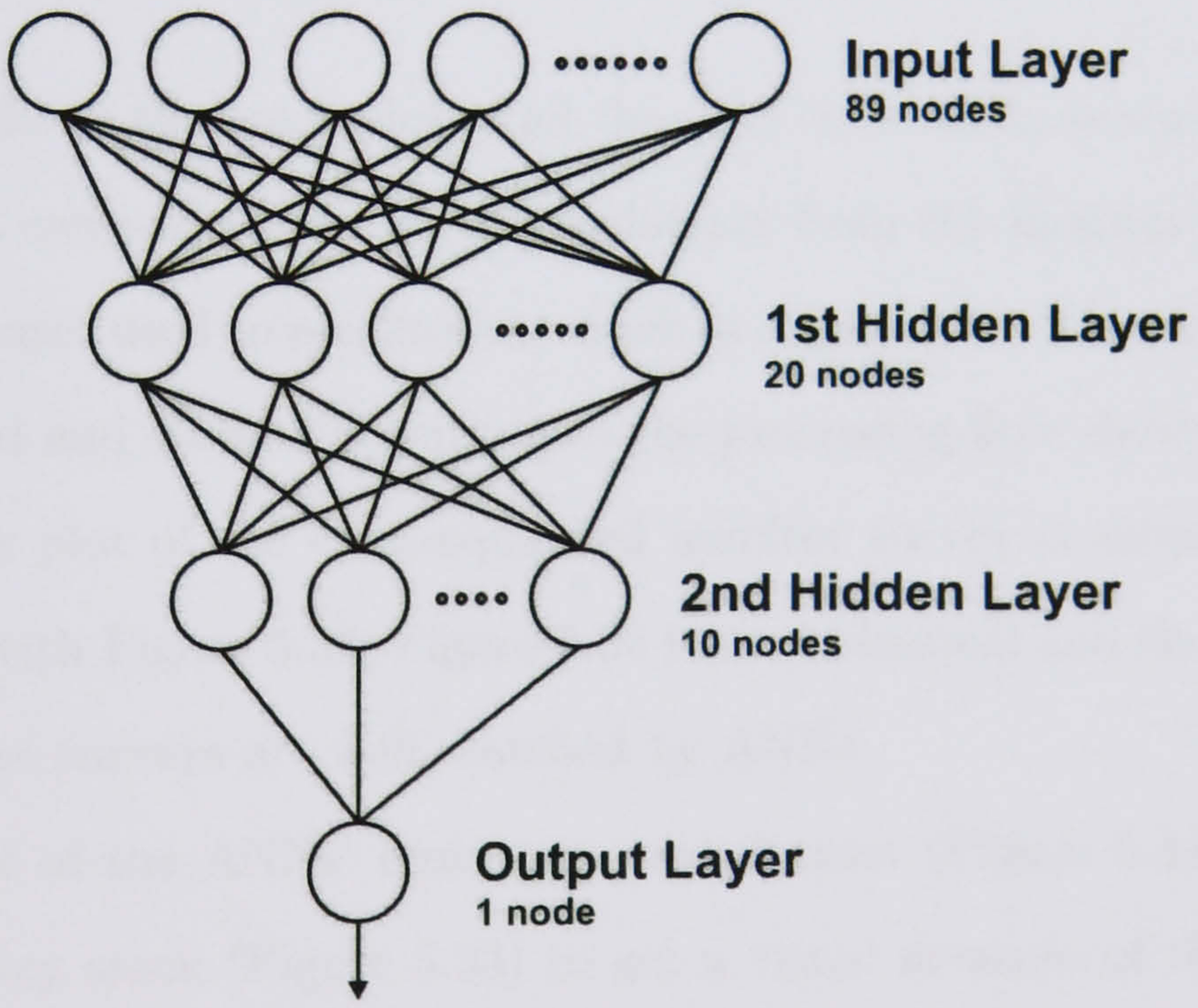


Figure 5.31: ANN topology used in the final matching test. One time sample from the base survey is used as a target output. Input patterns are six geometrical parameters (Section 5.3.1), a window of samples spanning 96 ms from the monitor (13 samples) at the nearest coinciding trace r_m from the monitor survey, and windows of 32 ms (5 samples) from the offset gather neighbouring traces, see Figure 5.32.

of six geometrical parameters (Section 5.3.1), and seismic input from trace r_m (nearest to r_b in CDP and offset) from the monitor survey and 14 other traces close to r_m. The exact seismic input and output is defined in Figure 5.32.

One ANN is allocated for one offset, and hence any pattern created for that offset is used by that ANN. During training, a random half of the patterns is shown to the network as training patterns, and the other half to cross-validate training. After training and during application, all the input patterns are passed through the networks to produce the full cross-equalized dataset.

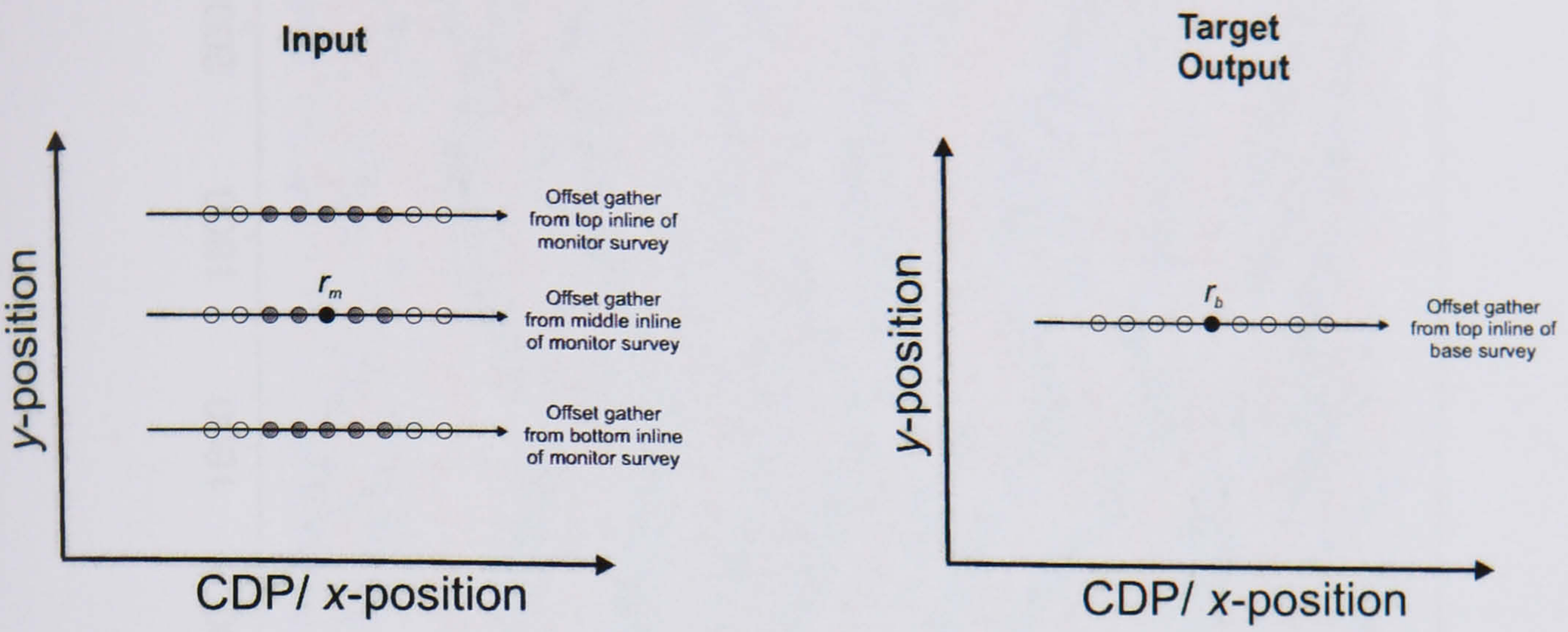
All the offset gathers corresponding to channels 100–230 are matched. This is possible because the computational time is much faster. The experiment took one day on a cluster of 16 computers or approximately 16 CPU-days, which is significantly less than the 220 CPU-days reported in the first test in Section 5.1.1 while actually matching more offset gathers.

Results

Figure 5.33 shows the stack plot of all the data from the base survey. ANNs were used to produce a cross-equalized pre-stack dataset from the monitor survey to match the pre-stack dataset used to produce the stack in Figure 5.33. This cross-equalized dataset was processed and stacked according to the processing-flow shown in Table 3.2.

The stack plot of the cross-equalized monitor survey is shown in Figure 5.34. In comparison with Figure 5.33, Figure 5.34 is near identical and thus illustrates that the two time-lapse surveys are well matched by ANNs.

The stack of the ANNs' cross-equalized dataset (Figure 5.34) is subtracted from the base survey stack (Figure 5.33) to get a visual measure of the error in matching. Figure 5.35 shows the difference plot, which illustrates a very good matching capability. The result of this matching exceeds results attained by the early scheme configurations (shown earlier in this chapter) and by linear methods that are shown in the next section.



(a) Traces involved: each trace is represented by a circle, where time is perpendicular to the page

(b) Samples involved

Figure 5.32: The seismic input-output to the network during training to perform pre-stack cross-equalization. Given that a certain sample — at time t, offset m, and CDP x — is to be matched, (a) shows the traces involved in training the network, and (b) shows the exact seismic input and target output. Thus, the target output is the sample at time t from a trace r_b at CDP x and offset m from the base survey, and the input is from trace r_m from the monitor survey (nearest to r_b in offset, and CDP or shot x- and y-position) and from 14 neighbouring traces to r_m. The neighbouring traces are those shaded in (a). The exact seismic input is a window of 96 ms (13 samples) from trace r_m and windows of 32 ms (5 samples) from the 14 neighbouring traces, where all these windows are centred at time t.

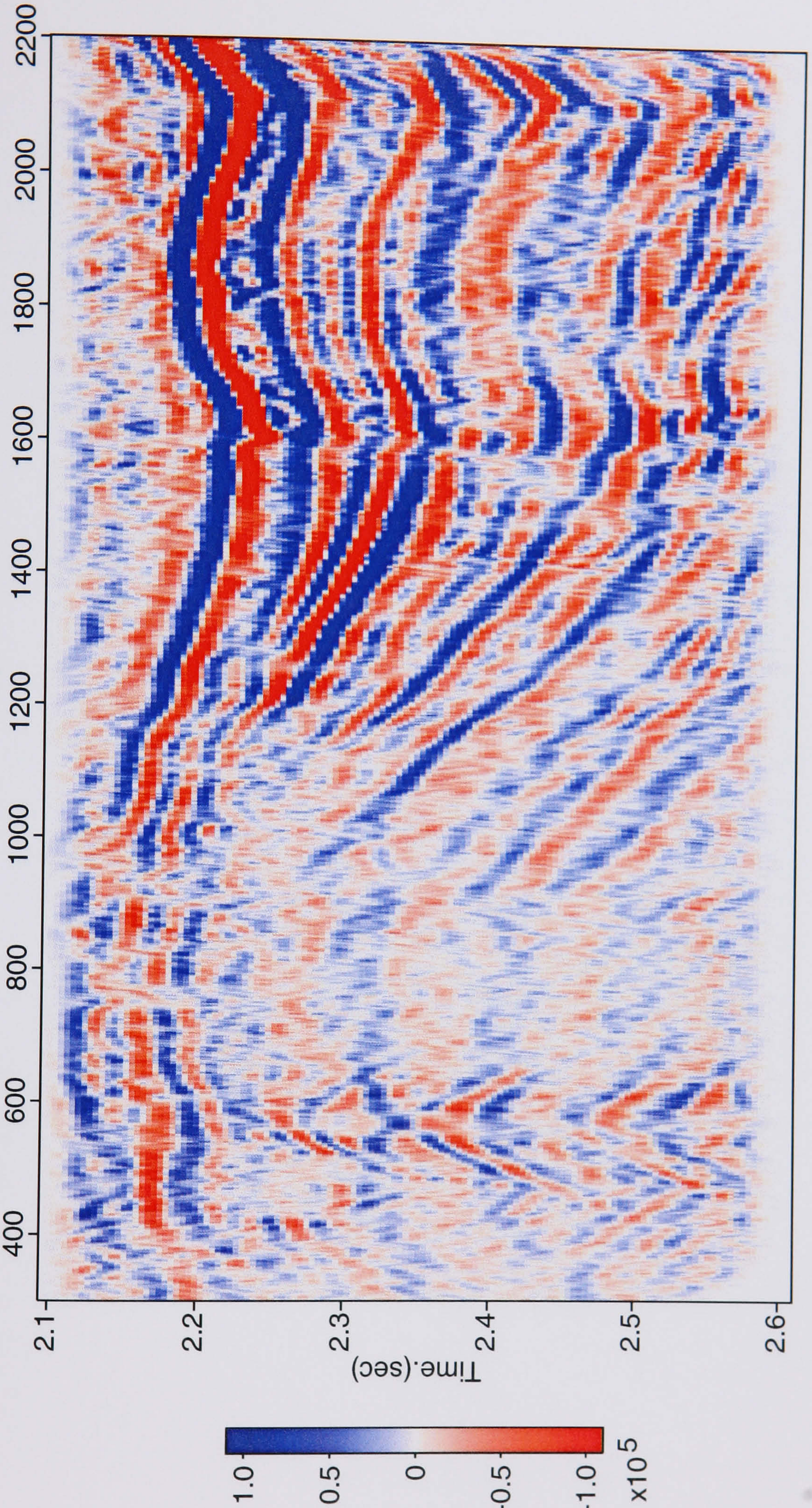


Figure 5.33: Seismic stack plot of all the offset gathers of channels 100–230 of the base survey. ANNs were used to produced a cross-equalized pre-stack dataset from the monitor survey to match those used to produce this stack. This plot is thus used for comparison with results achieved after cross-equalization.

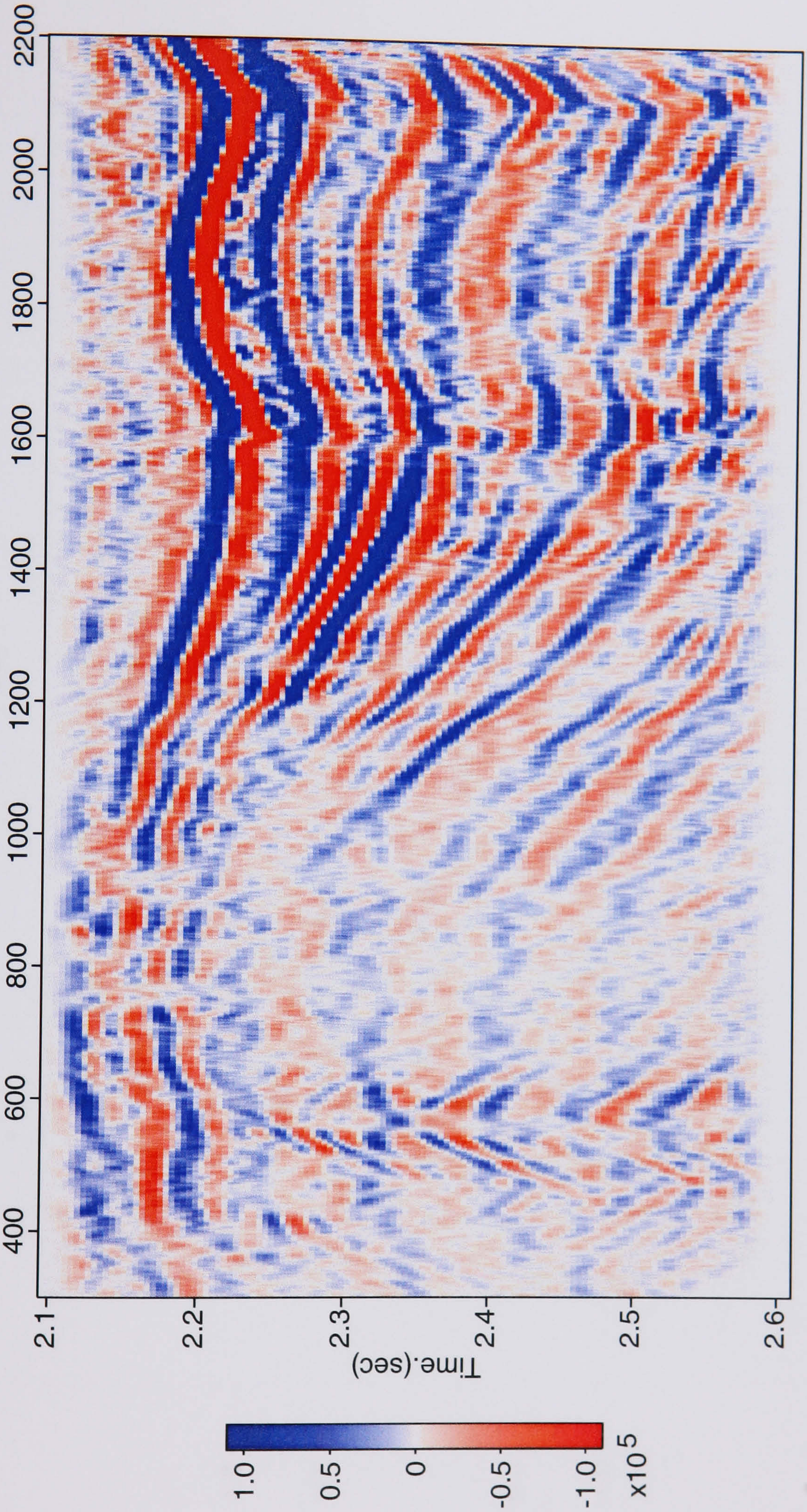


Figure 5.34: Seismic stack plot of the cross-equalized monitor survey using the best configuration. This can be compared to the stack plot of the base survey (Figure 5.33) plotted to the same scale. The plots are near identical and thus the matching is very good.

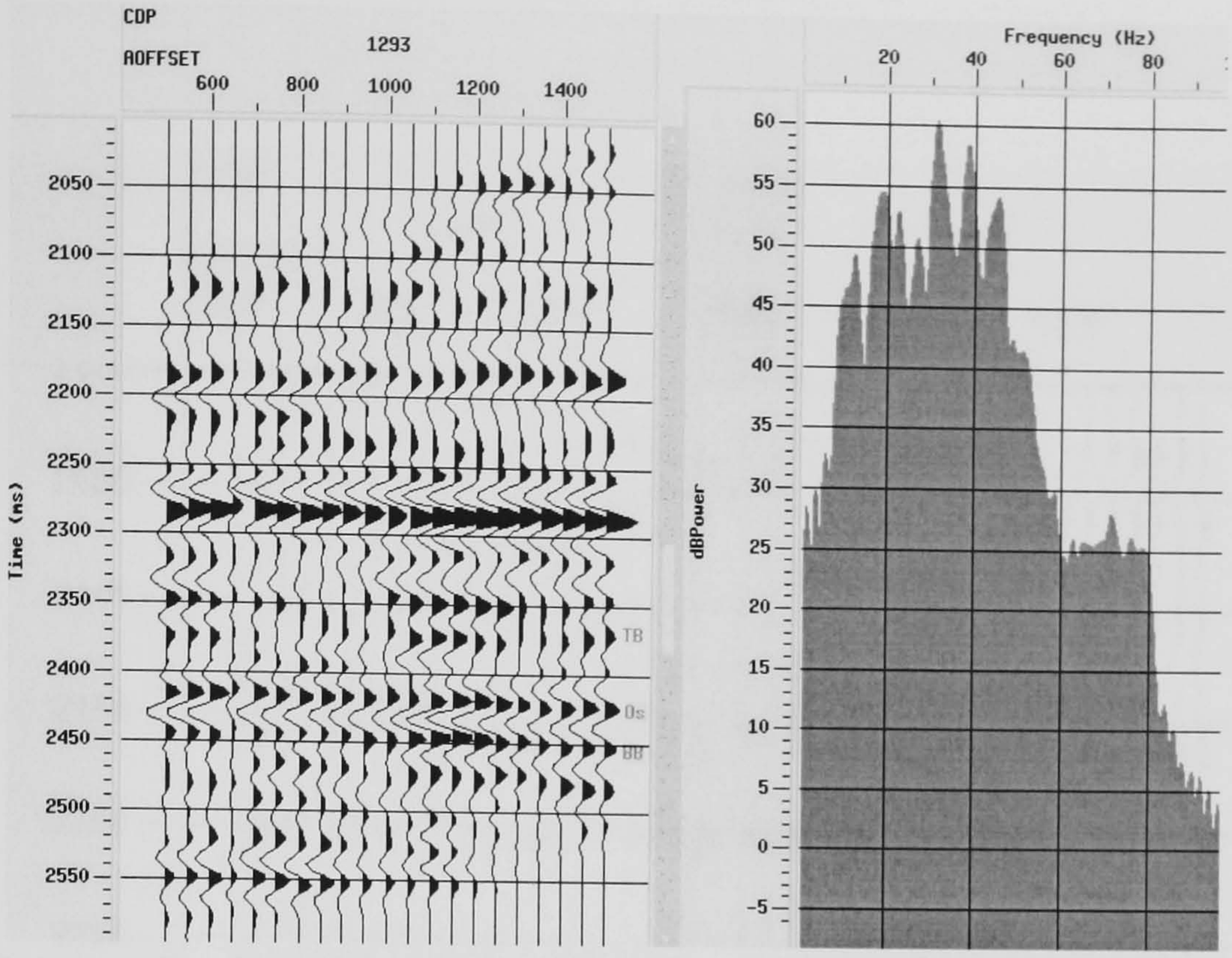
Figure 5.35: Difference between the base survey stack (Figure 5.33) and the stack of the cross-equalized monitor survey using the best configuration (Figure 5.34). This shows a very low error in matching when compared to the difference plots achieved after cross-equalization using linear methods (*cf.* Figure 5.38, Figure 5.39, and Figure 5.40).

Figure 5.36 shows the seismic and amplitude spectrum at pre-stack CDP 1293 of the ANN cross-equalized monitor survey in comparison with that of the base survey. The two spectra are near identical. The seismic data indicate good matching achieved by ANNs. This seismic is further compared in Figure 5.37. which shows a close-up of the seismic. The figure show minor differences between the base survey and the ANN cross-equalized CDP gathers. The most significant difference occur in the reservoir area, and may thus be production-related.

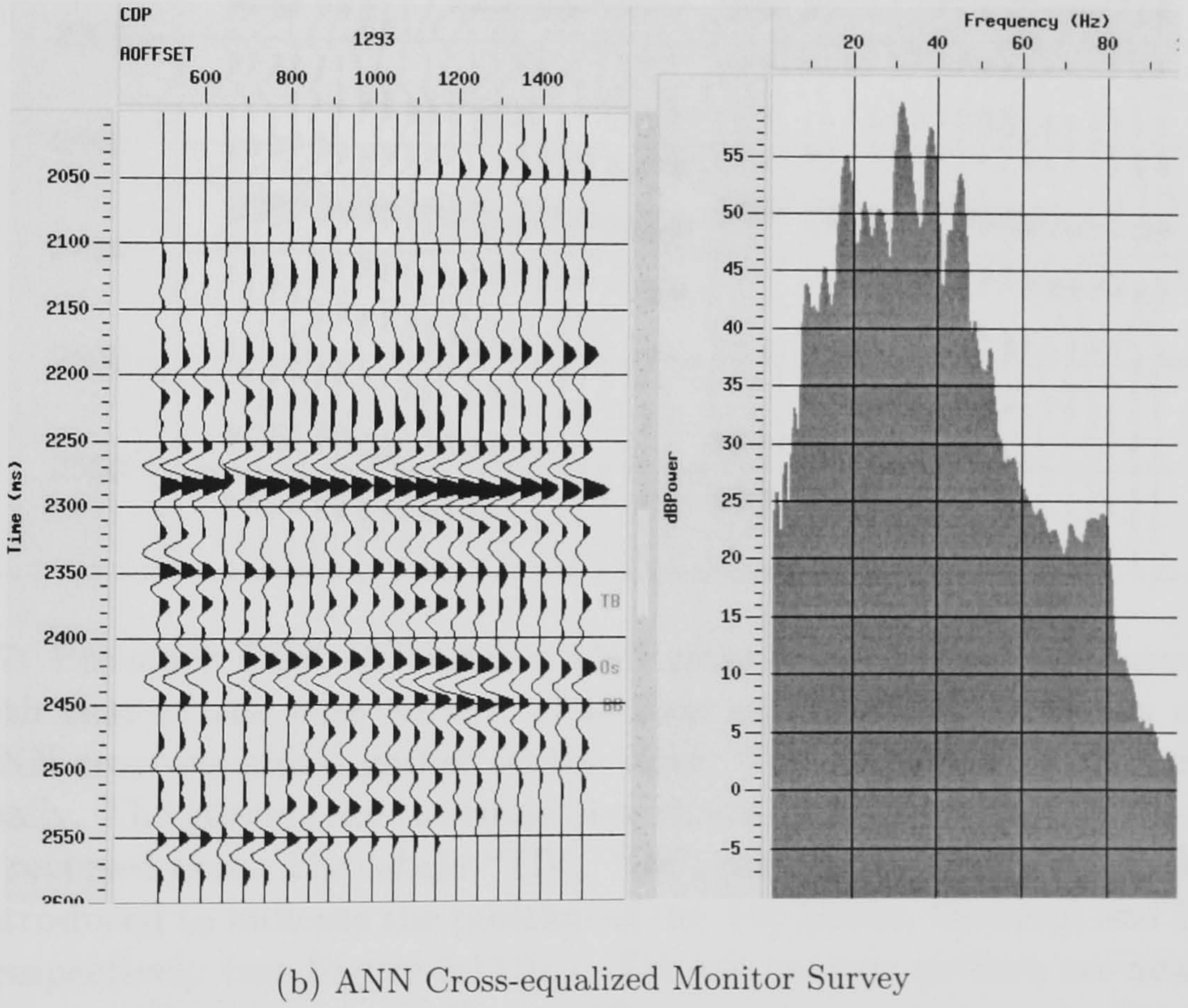
5.4.2 Matching using linear filters

To compare ANN cross-equalization results, cross-equalization was also done using the linear method that was described in Section 1.2.2. This method estimates three scalar misfit parameters (amplitude, time, and phase) between two traces (or time series) and then corrects for those misfit parameters. The aim is to show a fair comparison with the matching achieved by ANNs. Therefore, the matching is done pre-stack on the same data matched by ANNs. The complication is how to use the linear method to do some form of global matching.

One ANN is allocated for each offset gather and thus it seems appropriate to compare ANN-matching to the matching of one linear filter averaged for the whole offset. However, ANNs have the ability to incorporate many more free parameters than the linear filter which only contains three parameters. In addition, an ANN is capable of varying its matching across the offset gather. If one chooses to estimate a filter for each two time-lapse traces, then a lot more free parameters are involved and the linear matching is able to vary matching across the offset gather. However, this linear method would be prone to obliterating production-related anomalies. Therefore, it is appropriate to estimate a semi-global filter that averages its parameters across a certain range of traces.



(a) Base Survey



(b) ANN Cross-equalized Monitor Survey

Figure 5.36: Spectrum of the ANN cross-equalized monitor survey in comparison with that of the base survey at pre-stack CDP 1293. Each plot shows the spectrum on the right and the seismic section from which the spectrum is calculated on the left. The spectrum is plotted in dB log scale with the reference power being 1. The two spectra are near identical and peak around 40Hz. The seismic is NMO corrected and plotted to the same scale. The two seismic gathers are near identical. The most significant difference could be at the trough between 2375 ms and 2420 ms. A more detailed comparison is shown in Figure 5.37, where the polarity is reversed to pick out the troughs shown here.

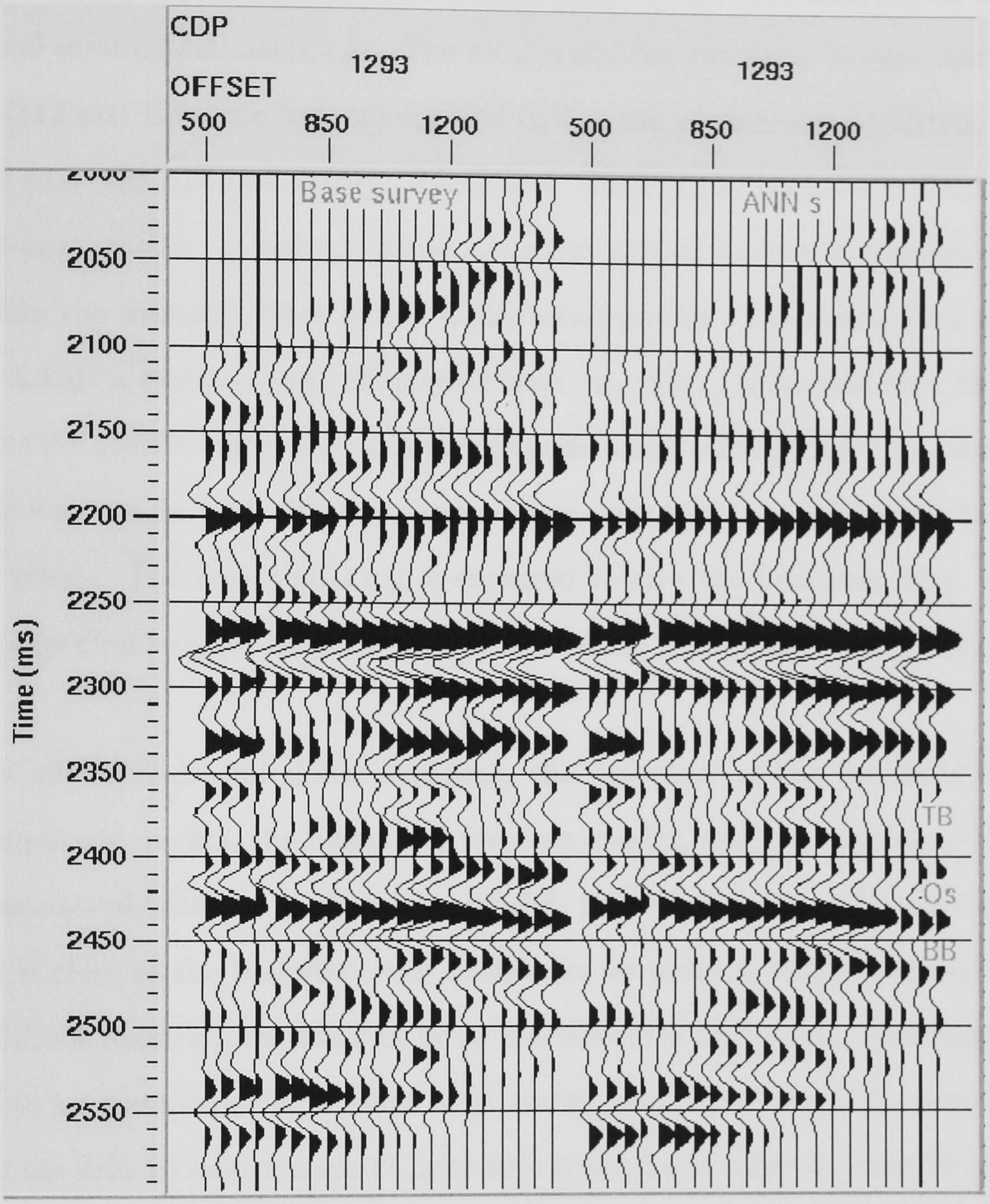


Figure 5.37: Pre-stack CDP 1293 of the ANN cross-equalized monitor survey in comparison with that of the base survey. The base survey gather is shown on the left and the ANN cross-equalized gather on the right. The seismic is NMO corrected and scaled globally. The gathers are the same ones shown in Figure 5.36, but the amplitude polarity is reversed here. The labels "TB", "Os", and "BB", shown on the lower right side, are introduced to indicate the position of the Top Brent, Oseberg, and Base Brent Horizons, respectively (see Figure 3.11(b)). The two seismic gathers are near identical with only minor differences. A significant difference occurs between the Top Brent and Oseberg horizons, which may be production-related.

Configuration

The linear matching was conducted in several ways: local matching, global matching, and several semi-global matchings. The local matching considers in each instance one trace r_b (512 ms) from the base survey and match the nearest corresponding trace r_m (512 ms) from the monitor survey. The global matching estimates the filter for each two time-lapse traces, r_b and r_m, along the offset gather, and then averages the filter and applies the averaged filter to the whole offset gather. The semi-global estimates its filter across a certain range of traces within an offset gather and then applies the average to the middle trace, and does so along the whole offset gather. The semi-global matching was run in four different modes where averaging is done across 11, 21, 51, and 101 traces. The local matching is expected to give the best matching, while the global is expected to give the worst matching, and the semi-global a result between the two.

In the global and semi-global matching, the "average" of the misfits is the mean of the amplitude misfits, the median of the time misfits, and the median of the phase misfits associated with the median time misfit[2]. In semi-global modes, if a trace to be matched is close to the boundary, the misfits are an average of the specified range of traces without requiring the trace to be at the middle of the range. Therefore, if trace no. 11 is to be matched using a range of 51 traces, then the averaging is over 51 traces involving the first 10 traces, trace 11, and 40 traces after it; after the first 20 traces and before the last 20 traces, the trace concerned is not at a boundary and thus averaging is over 51 traces that involve the trace itself and 20 traces from either side.

Results

The linearly cross-equalized monitor survey datasets were processed and stacked in a similar fashion to the ANN cross-equalized dataset. The stacks were subtracted from

[2]As explained in Section 1.2.2, for every time misfit there is a corresponding phase misfit. Therefore, once a time misfit value $\bar{\tau}$ (the median time misfit) is selected, then there are a number of phase misfits associated with every time misfit that has the value $\bar{\tau}$, and thus it is possible to calculate the median phase misfit from those.

the base survey stack (Figure 5.33) to produce plots that act as a visual representation of the error in matching. In addition, these plots can be compared to the difference plot in Figure 5.35. Three plots are shown here: the difference plot after local matching (Figure 5.38), after global matching (Figure 5.39), and after a semi-global matching using a range of 51 traces (Figure 5.40).

The local matching is the best and the global is the worst as expected. ANN matching is superior as it even produces a better matching than the local linear matching. In addition, linear matching deteriorates significantly in areas with relatively strong event dip. This is even more significant with more global matching.

Figure 5.41 shows a close comparison on pre-stack data between the ANN and linear (local) cross-equalizations. The figure illustrates further the superiority of ANN matching. Although ANN cross-equalization does not produce an identical match to the base survey, all the main events are matched. Most of the differences are minor and not consistent within the different offsets, and may therefore disappear during processing to stack. On the other hand, the linear cross-equalization found optimum linear misfits and corrected for those but, in doing so, it introduced spurious events in areas where the misfits are not optimal at that local interval. Spurious events, such as the peak at 2310 ms and the trough at 2380 ms, are significant and consistent within the different offsets and may therefore appear as large differences after stacking.

5.4.3 Quantitative comparison

In this sub-section, a few quantitative measures of the matching performance are shown. This gives a better measure of the matching capability, and provides a better comparative measure. The three quantitative measures are predictability, normalized root mean square (NRMS), and residual root mean square (RRMS). These were defined in Section 4.7.

Figure 5.42 shows the predictability graph. The figure illustrates that an ANN produces better matching through its ability not only to correct linear misfits but the

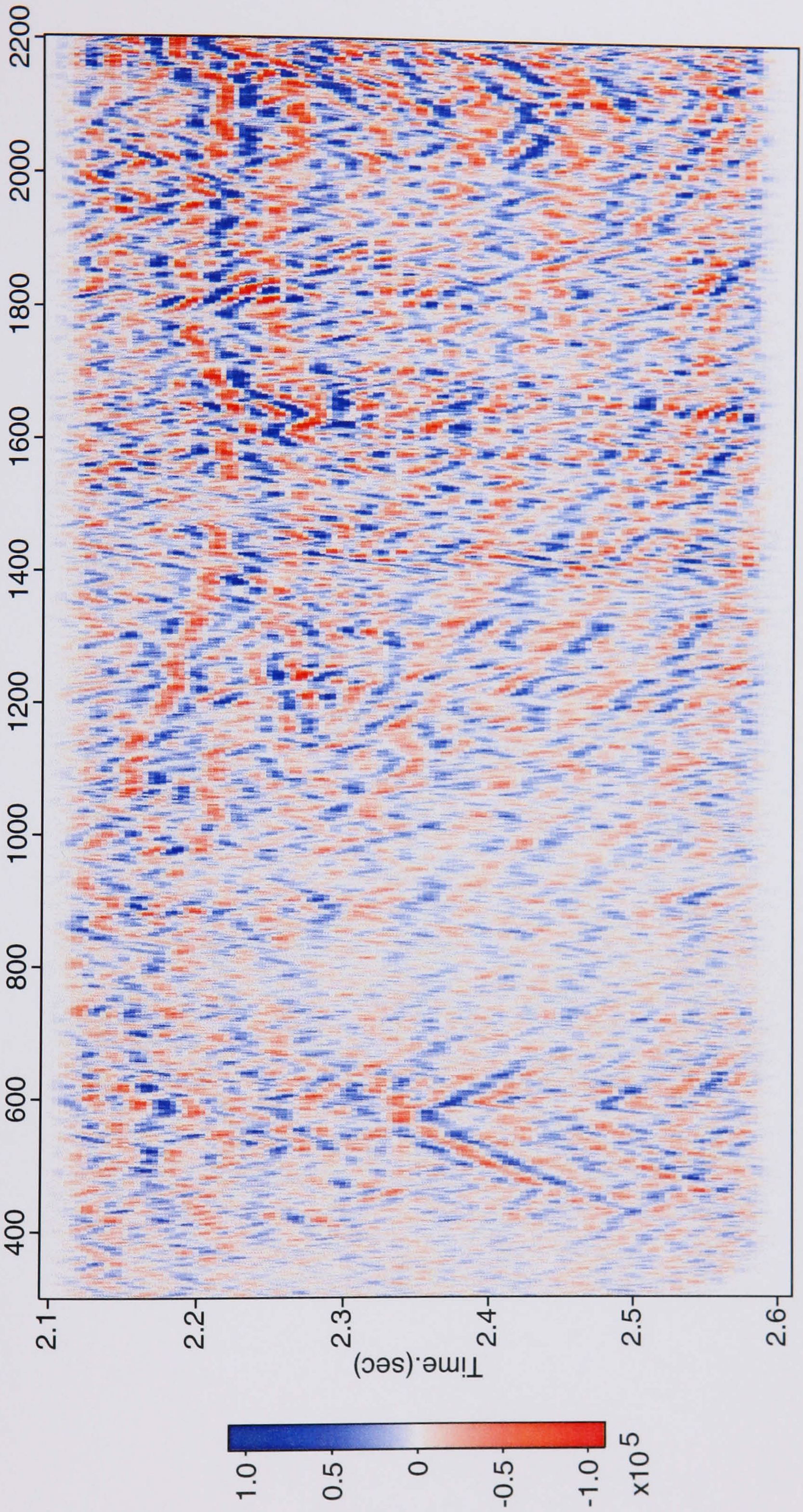


Figure 5.38: Difference between the base survey stack and the stack of the cross-equalized monitor survey using linear cross-equalization method with local mode. The local mode estimates a filter for each two time-lapse traces. The plot represents the error in matching, and represents the best matching in comparison with the more global matching with the same linear method (cf. Figure 5.39 and Figure 5.40). However, the result is worse than that seen when ANNs were used (Figure 5.35).

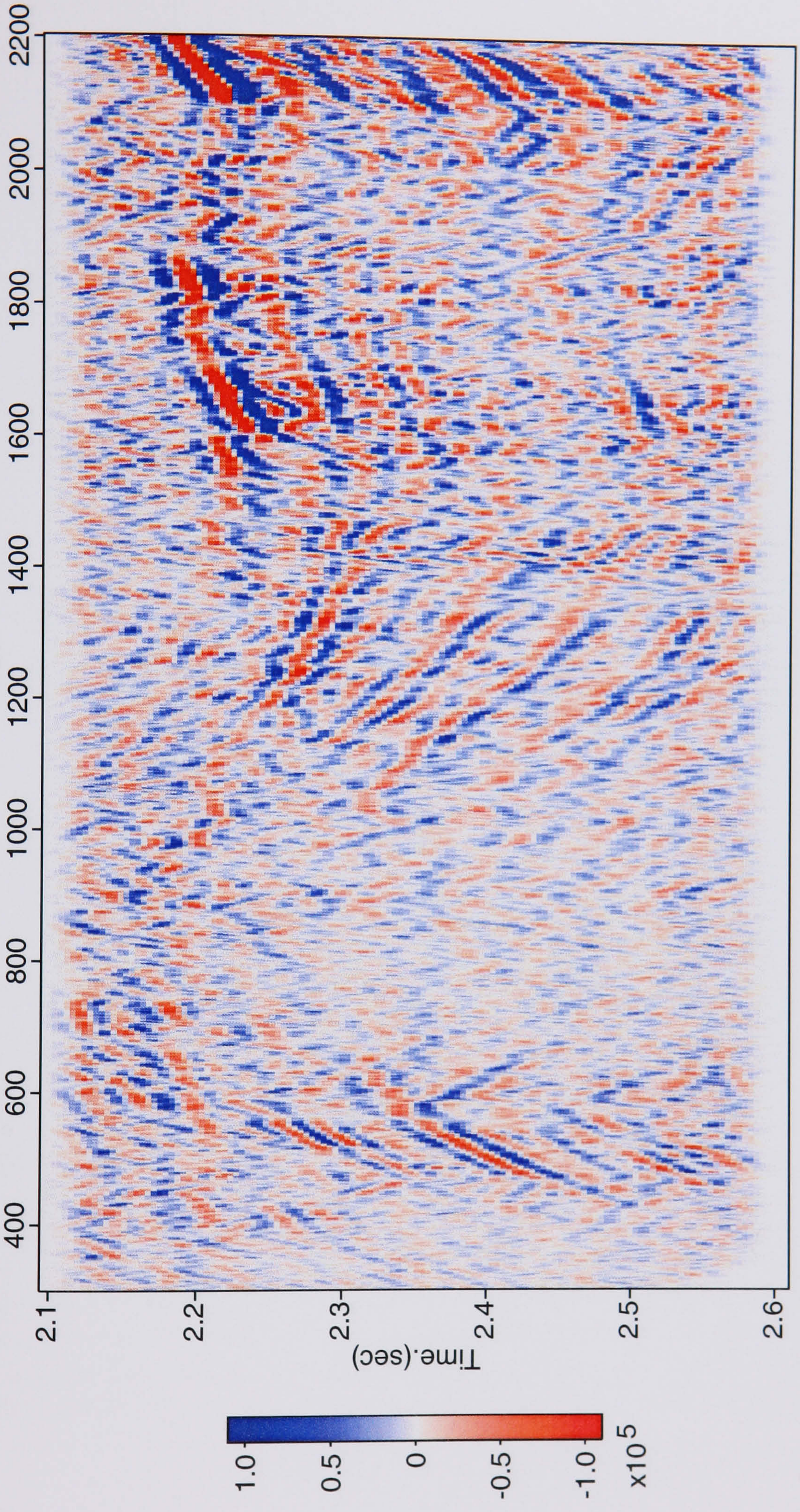


Figure 5.39: Difference between the base survey stack and the stack of the cross-equalized monitor survey using the linear cross-equalization method with global mode. The global mode estimates a filter for all two time-lapse traces within an offset gather and then averages the filter and applies it to all the traces within the gather. The plot represents the error in matching, and represents the worst matching in comparison with less global matching with the same linear method (cf. Figure 5.38 and Figure 5.40). The matching error shown here increases in areas with strong dip (at CDP 1600–1800 and less so at CDP 1200–1400).

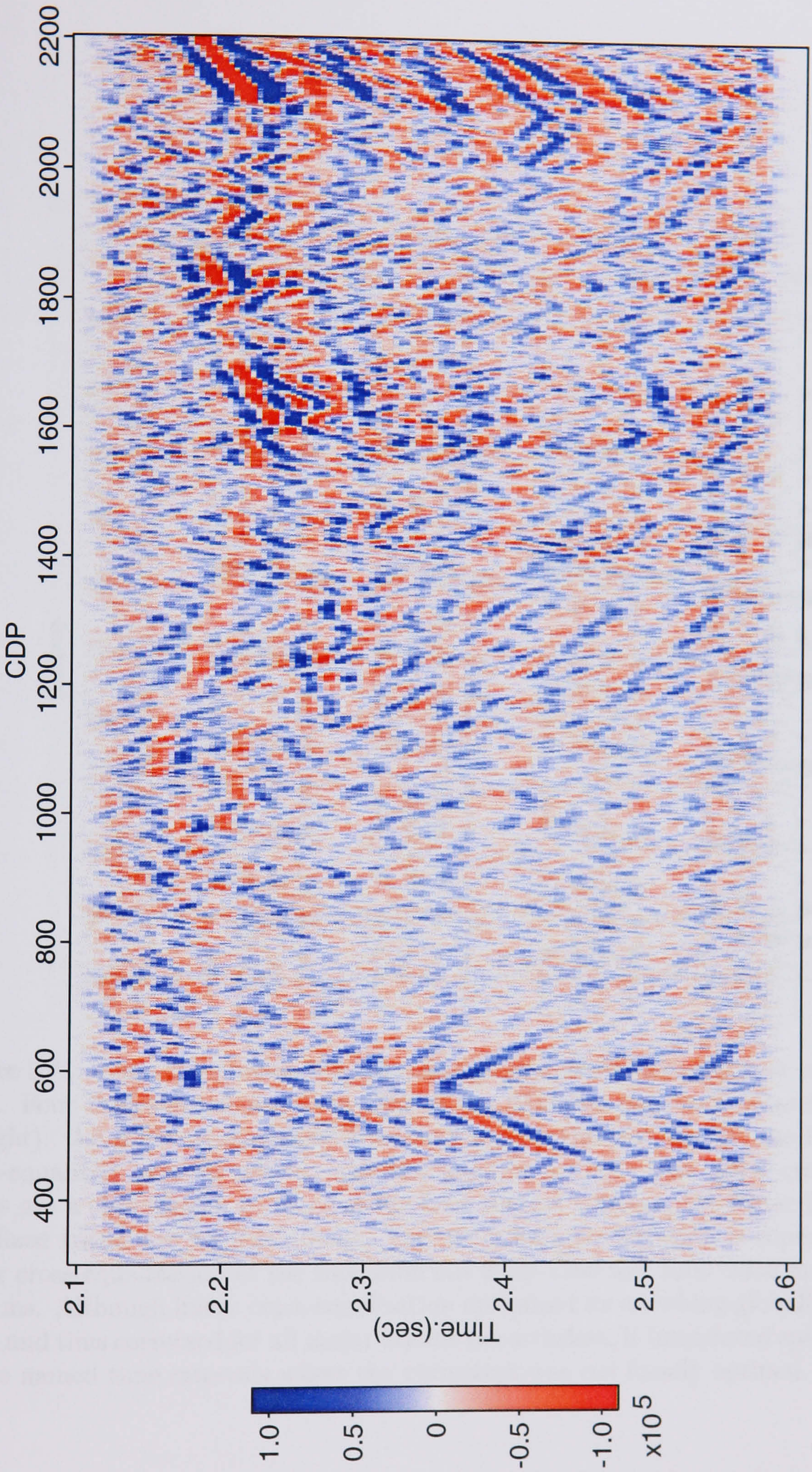


Figure 5.40: Difference between the base survey stack and the stack of the cross-equalized monitor survey using the linear cross-equalization method with semi-global mode involving averaging over 51 traces. This mode estimates a filter for all two time-lapse traces within an offset gather and then averages the filter for each 51 traces within the gather, and applies the filter to the middle trace and does so for all the traces within the gather. The plot represents the error in matching, and shows a performance midway between local matching and global matching (cf. Figure 5.38 and Figure 5.40).

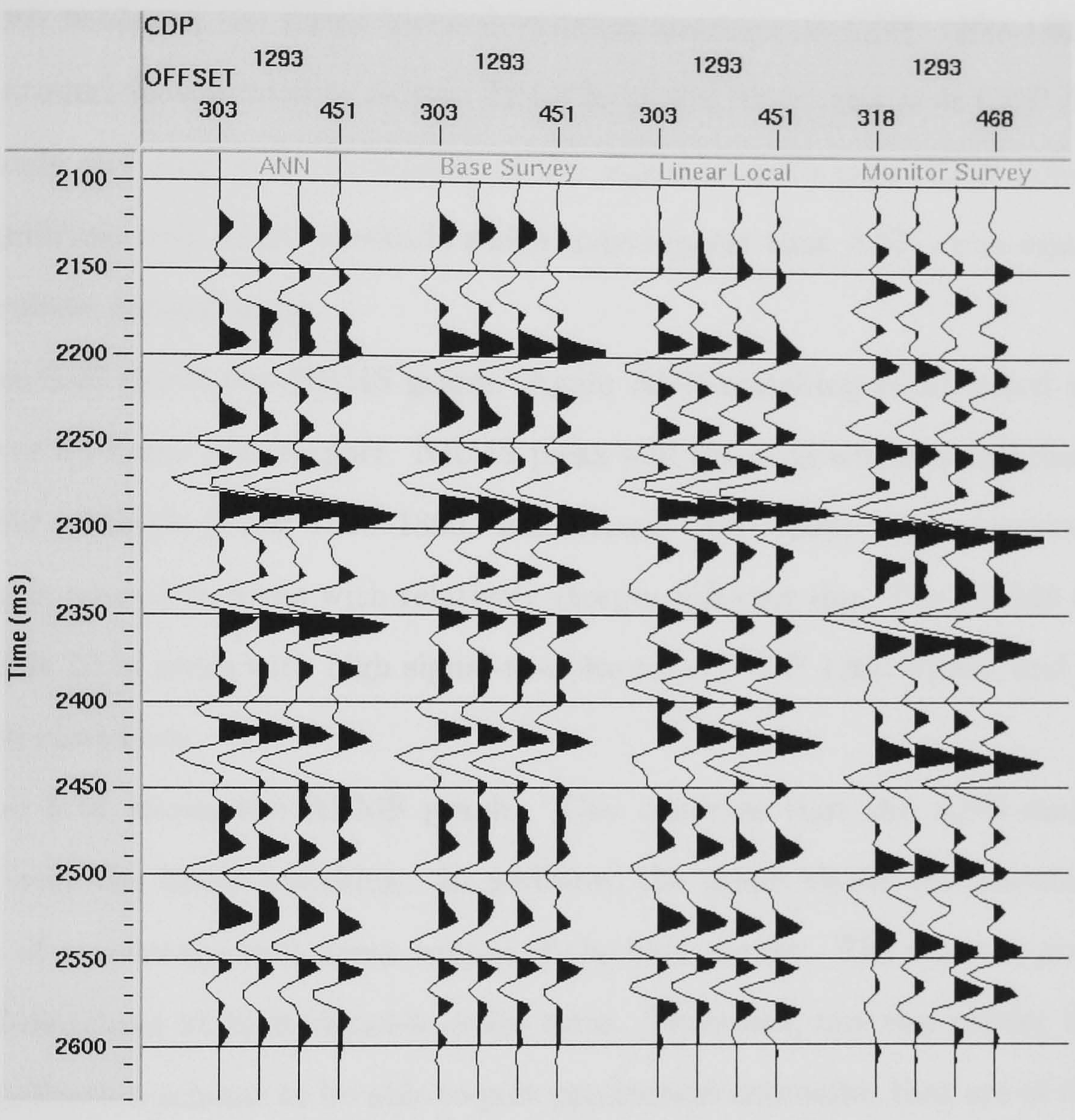


Figure 5.41: Close comparison between the different datasets at the near offset of CDP 1293. Four traces are shown from each of the following pre-stack datasets (from left to right): the ANN cross-equalized monitor survey, the base survey, the linear (local) cross-equalized monitor survey, and the monitor survey. The ANN cross-equalized traces are a close match to those of the base survey; whereas, the linear (local) cross-equalized traces are far less similar. The ANN cross-equalization is superior over the linear cross-equalization at the time intervals 2220–2250 ms, 2300–2330 ms, and 2370–2410 ms. Although linear cross-equalization optimized its matching globally along each trace and thus corrected for all major events; nevertheless, it introduced spurious events at the named time intervals where the correction was not locally optimal.

non-linear misfits. Predictability is highest in areas with relatively high signal-to-noise ratio (CDP 1200–2200). ANN matching produces a very high predictability value of about 98 in high signal-to-noise ratio areas and at around 92 elsewhere. On the other hand, both modes of the linear method behaved similarly at CDP 1200–1600 with a value of around 93 and drop to around 72 for local and 68 for global at CDP 300-1200. Interestingly the drop in predictability at low signal-to-noise area (CDP 300–1200) is more significant with linear methods which may suggest that ANN cross-equalization is more robust against noise.

Figure 5.43 shows the NRMS graph. Again ANN-matching is indicated to be superior over its linear counterpart. NRMS picks well the area which is matched poorly with linear methods (CDP 1600–1800, and around CDP 1200). This deterioration in matching is related to areas with relatively steeper reflector dip. The NRMS for ANN matching is 20 in areas with high signal-to-noise ratio (CDP 1200–2200), and between 40 and 60 elsewhere.

Figure 5.44 shows the RRMS graph. This confirms that the ANN-matching is superior over the linear-matching. In addition, the graph shows the matching error in terms of percentage root mean square of the base survey. The value is around 4% for ANN-matching at high signal-to-noise ratio. Therefore, one can expect the ANN cross-equalization scheme to be able to pick production anomalies that are of this order of magnitude or higher with respect to the surrounding seismic, but not much lower. This conclusion is tested in Chapter 6. However, it is likely that this percentage is very dataset dependent.

Figure. 5.45 shows a cross-plot of predictability versus NRMS of the ANN cross-equalized monitor stack traces and the base survey stack traces. Similar plots of the local linear matching and the global linear matching are shown in Figure 5.46 and Figure 5.47, respectively.

To interpret these plots, one can consult the synthetic plot given in Figure 4.35, where it was shown that predictability versus NRMS plot moves rightward due to linear

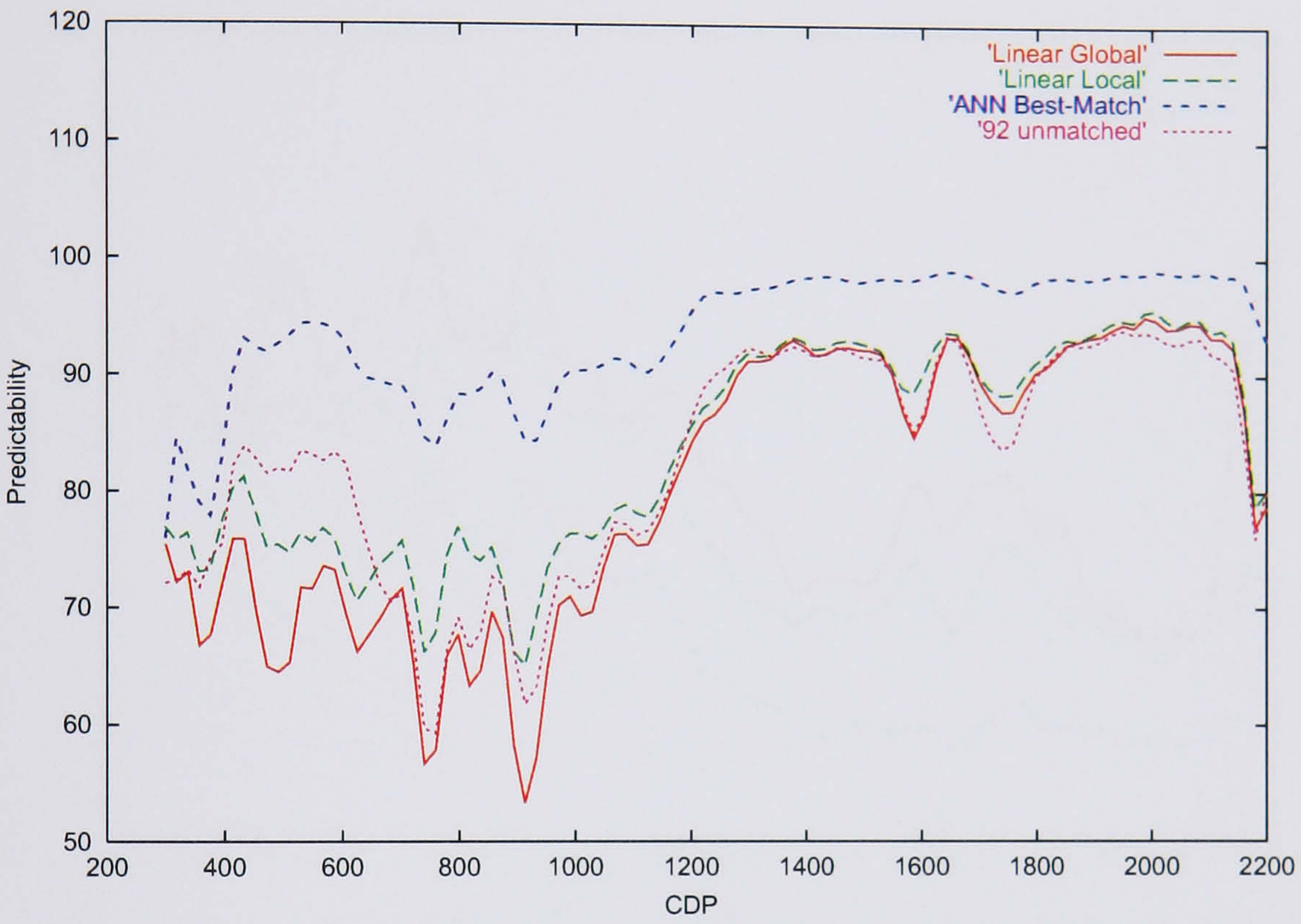


Figure 5.42: Graph of predictability between the cross-equalized monitor stack traces and the base survey stack traces. Predictability is plotted against the CDP as a smooth curve. The graph illustrates and compares results of ANN matching, both global and local linear matching, and predictability between the base and the unmatched monitor stack '92 unmatched'. The linear matching predictability follows closely that of the unmatched monitor. This is because linear corrections do not alter predictability, whereas ANN through non-linear transforms can. This can be envisaged through ANN's ability to take into account alteration in trace position and correct non-linearly for it, and through its ability to suppress noise by not being able to match it.

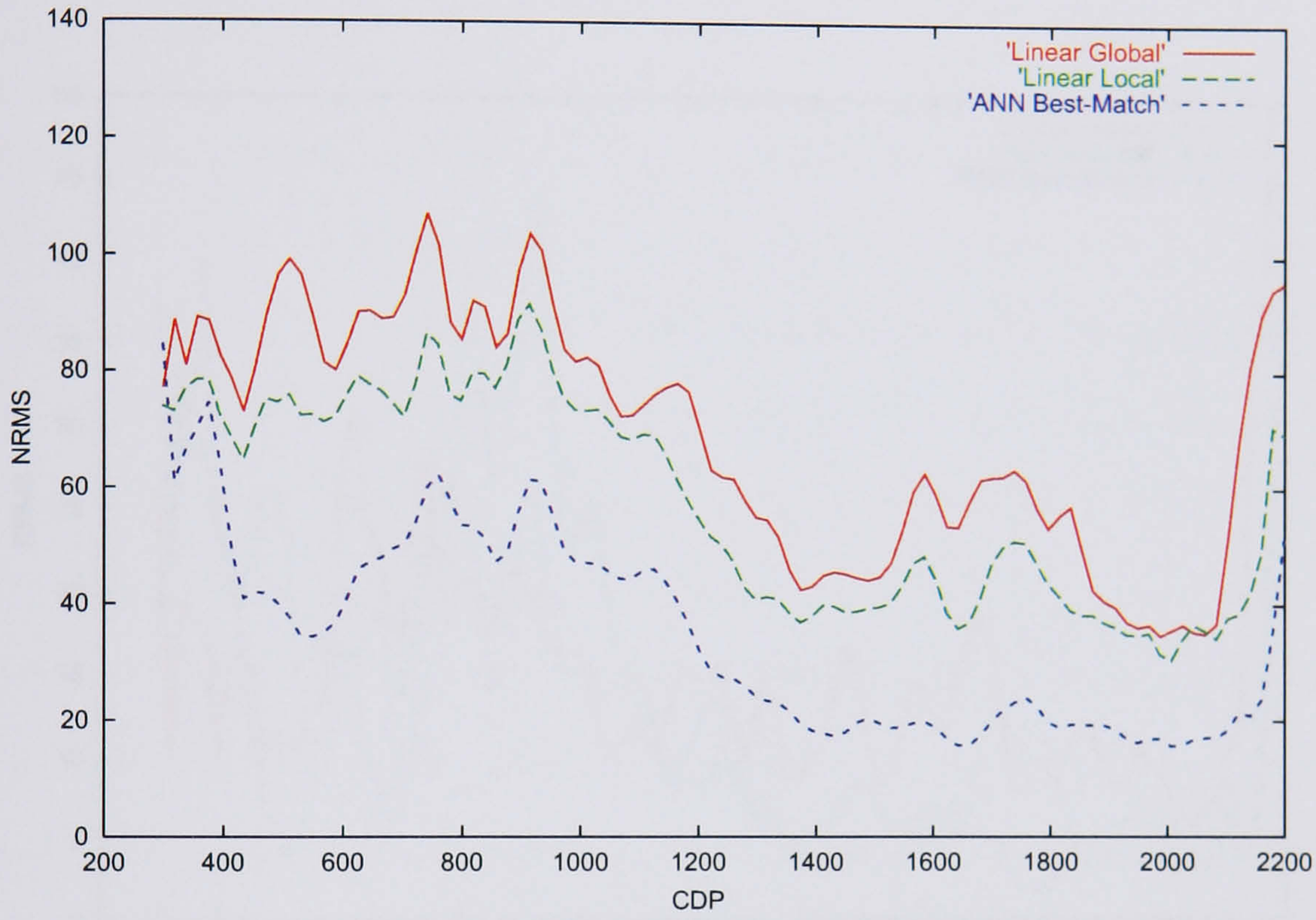


Figure 5.43: Graph of normalized root mean square (NRMS) between the cross-equalized monitor stack traces and the base survey stack traces. NRMS is plotted against the CDP as a smooth curve fit. The graph illustrates and compares results by ANN matching and both global and local linear matching. As in Figure 5.42, the performance of all the matching methods drops in areas of lower signal-to-noise ratio (CDP 300–1200). The graph illustrates the superiority of ANN matching with a value of around 16 in high signal-to-noise areas (CDP 1200–2200). Interestingly, the linear methods seem to struggle at CDP 1600–1800. This could be because of the relatively steeper dip angle.

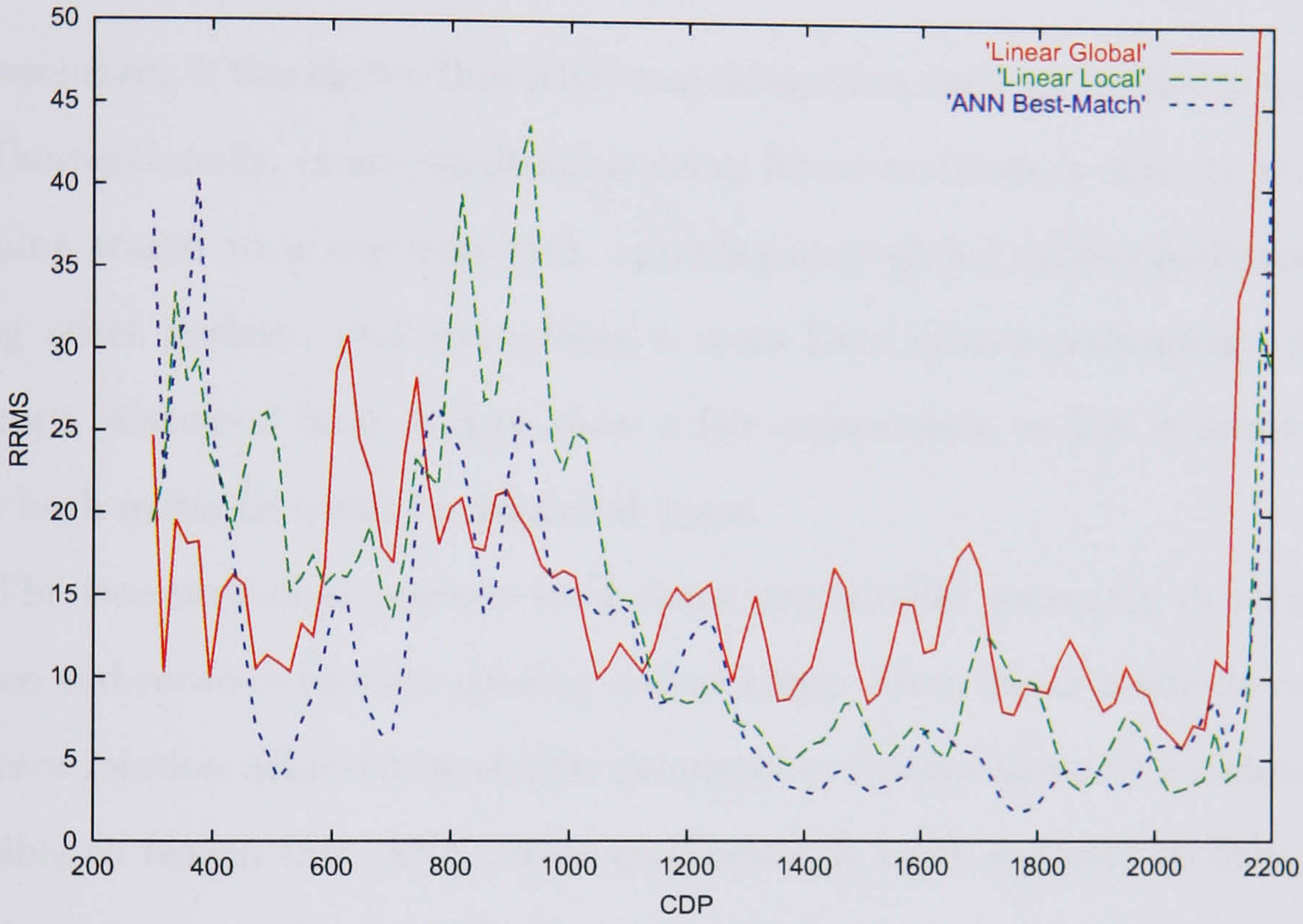


Figure 5.44: Graph of residual root mean square (RRMS) between the cross-equalized monitor stack traces and the base survey stack traces. RRMS is calculated according to Equation 4.6 and is plotted against CDP as a smooth curve fit. The graph represents the error in matching in terms of RMS. This again shows the general superiority of ANN matching over its linear counterpart. The RRMS is at around 4% in the area with better signal-to-noise ratio (CDP 1200–2200).

misfits and both downward and rightward due to non-linear misfits. Here, the plots illustrate the ANN's superiority in matching due to its non-linear transform ability. In addition, it shows how the global matching suffers due to the global averaging. Although the ANNs perform a form of global averaging, they still manage to optimize their matching.

5.4.4 Conclusion

In conclusion, it was shown that ANN-matching offers superior results to linear-matching.

Conventionally, cross-equalization using linear methods is done in stages: e.g. re-aligning traces to a common grid, applying semi-global cross-equalization pre-stack along offset gathers, and later doing a more local cross-equalization. However, the attempt, discussed here, was to show a fair comparison, and it is important to note that both methods could be improved upon.

The two time-lapse surveys used share very similar geometry in terms of source-source and receiver-receiver spacing and azimuth. Since linear methods requires trace-to-trace relation achieved by similar geometry or interpolation to a common grid, it is possible to reason that ANN cross-equalization is more superior to linear cross-equalization in cases with very different geometries.

5.5 Summary

This chapter aimed to investigate the ability of ANNs to match two time-lapse datasets. The wavelet domain was initially chosen for training patterns in order to reduce the computational cost, and to ensure applicability to a reservoir area when training is done elsewhere. Initial tests gave promising results and highlighted several issues to be addressed, thus enabled the advanced scheme to be built. The application on initial test took 220 CPU days to cross-equalize 512 ms of 56 offset gathers.

The time domain was analysed for suitability, and proved both suitable and competitive to the wavelet domain. The time domain was carried forward as the domain

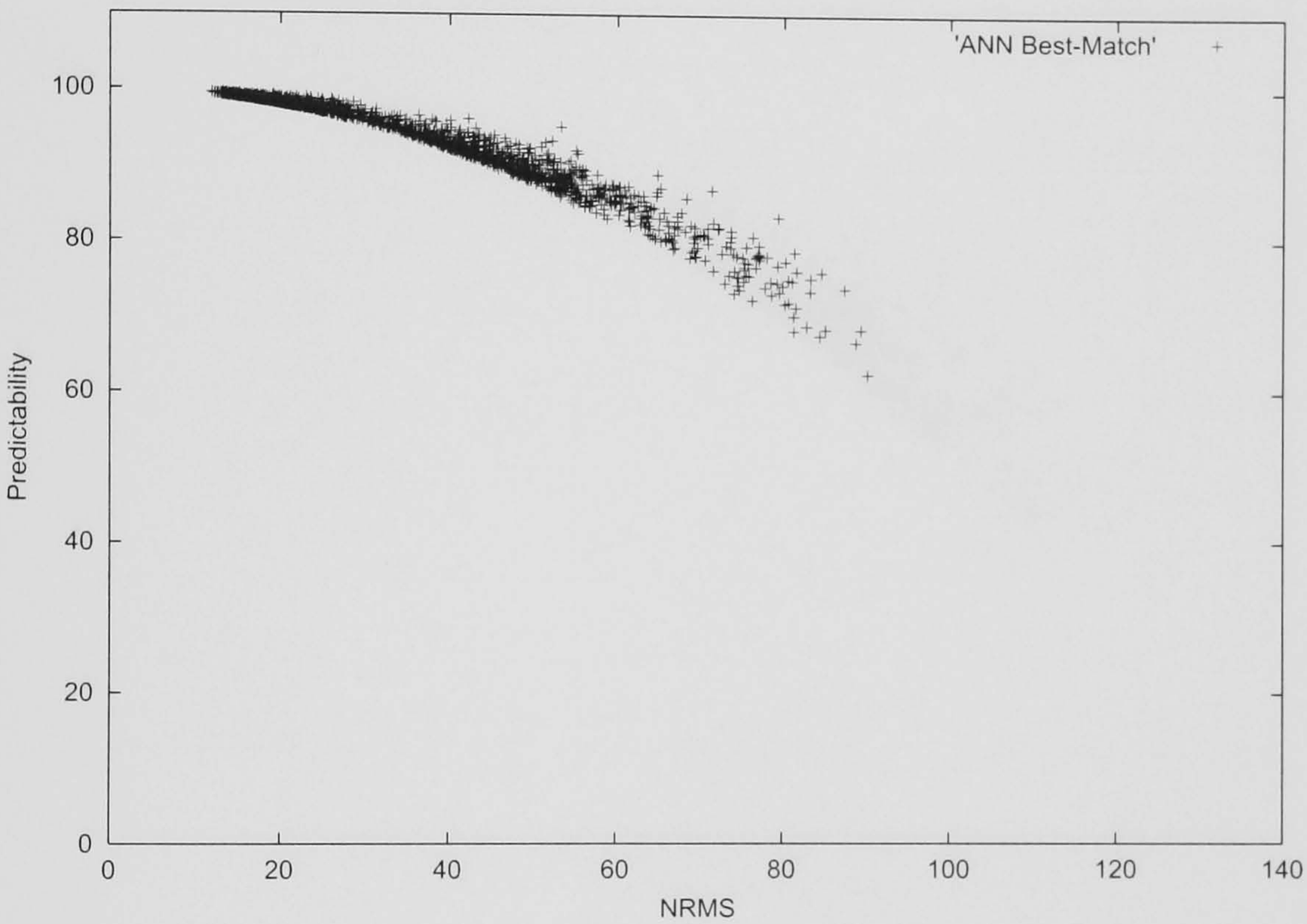


Figure 5.45: Cross-plot of predictability versus NRMS between the ANN cross-equalized monitor stack traces and the base survey stack traces. The left edge of the plot is higher than the plots for the linear (Figure 5.46 and Figure 5.47). This is because the linear cannot change predictability while ANNs, through non-linear transforms, have the ability to increase it. The plot is also very narrow suggesting that linear misfits are well corrected.

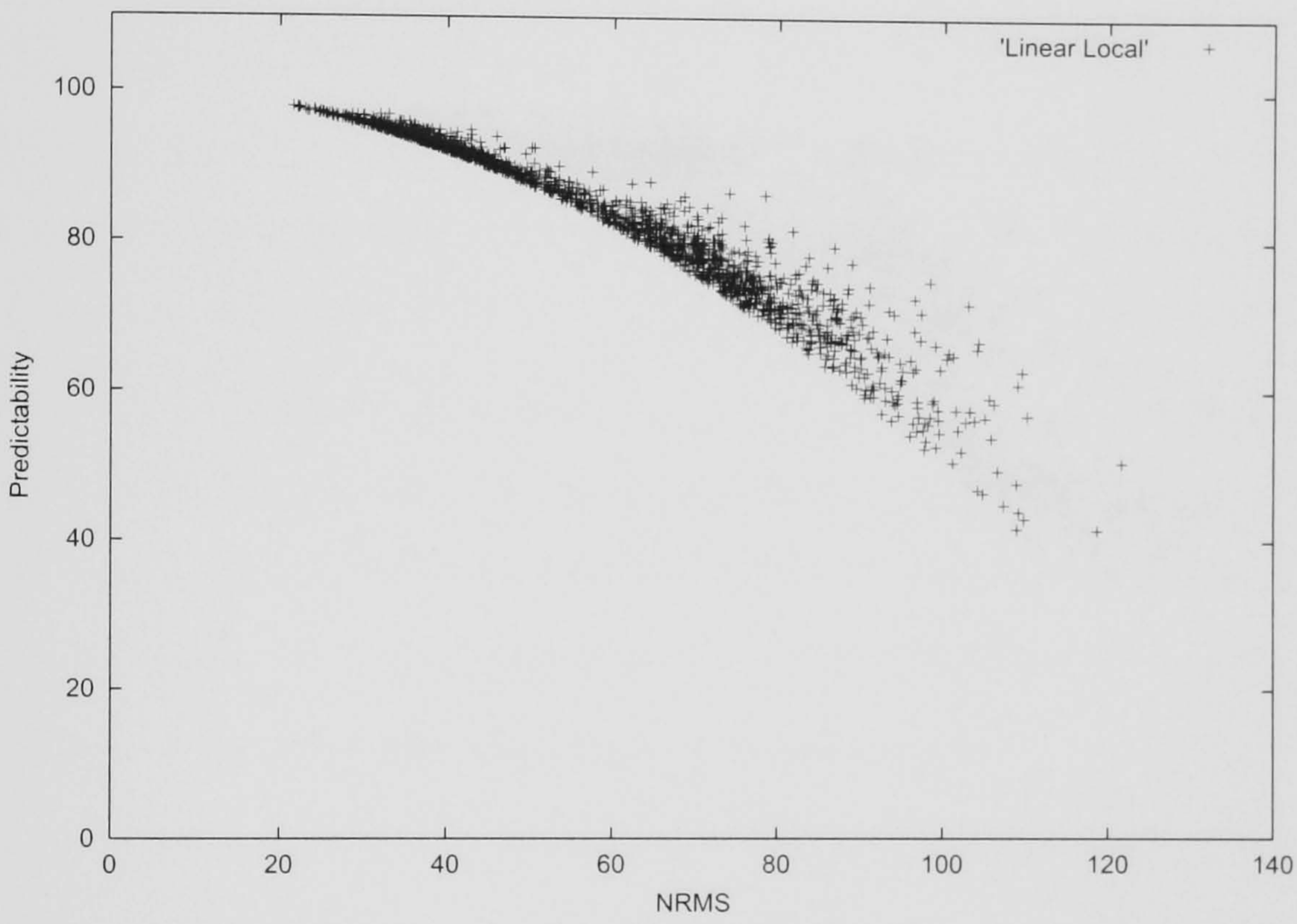


Figure 5.46: Cross-plot of predictability versus NRMS between the local linear cross-equalized monitor stack traces and the base survey stack traces. The left edge of the plot is lower than in the ANN plot (Figure 5.45) and similar to the global linear plot (Figure 5.47) since predictability is not enhanced by linear correction, but enhanced by the non-linear correction achieved by the ANN. The plot shows less scatter than the global plot, since here each trace is corrected optimally with no averaging.

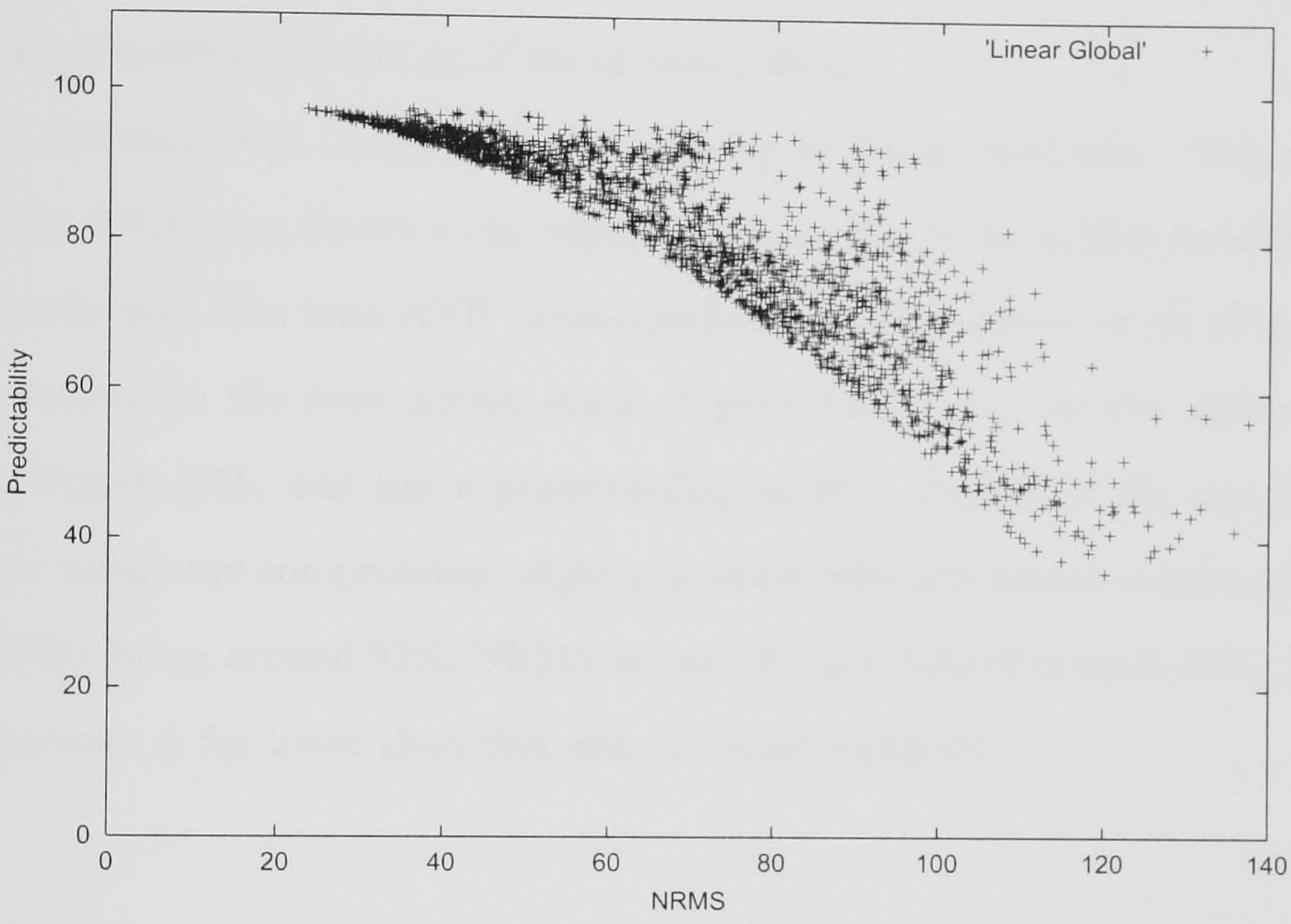


Figure 5.47: Cross-plot of predictability versus NRMS between the global linear cross-equalized monitor stack traces and the base survey stack traces. The left edge of the plot is lower than in the ANN plot (Figure 5.45) and similar to the local linear plot (Figure 5.46), since predictability is not enhanced by linear correction, but enhanced by the non-linear correction achieved by the ANN. The plot shows more scatter than the local plot since here each trace is not corrected optimally due to averaging.

to work on. The scheme initially required four ANNs per offset. ANN topology was relatively large with around 60 nodes in the first hidden layer and 30 nodes in the second.

Tests were carried out to optimize the input choice and reduce the topology. This was successful leading to a better configuration. Now, one ANN is allocated per offset with a small topology and more optimal pattern choice (Figure 5.31). The final test with optimal configuration took 16 CPU days to cross-equalize 512 ms of 130 offset gathers, representing a speed-up of about thirty times.

ANN matching was analysed and compared to linear methods. The matching achieved by ANNs was shown to be superior, partly due to its ability to achieve non-linear transforms. The final ANN cross-equalized monitor survey stack (Figure 5.34) when compared to the base survey stack (Figure 5.33) produces the difference plot shown in Figure 5.35, and has a predictability of 98%, NRMS of 20, and RRMS of 4%. These values are compromised slightly in areas with low signal-to-noise ratio with predictability being around 92%, NRMS around 50, and RRMS around 10%. However, the compromise is far lower than that seen in linear methods.

Chapter 6

Further Investigation on ANN cross-equalization

Any successful cross-equalization needs to be able to preserve genuine production-related anomalies, while performing an otherwise good matching. Therefore, it is necessary to assess ANN ability to preserve desired anomalies while matching otherwise.

The two time-lapse datasets, used in this work, do not have a clear genuine production-related time-lapse anomaly nor are they predicted to have a strong anomaly (Section 3.2.2). Hence, to assess better the ability of desired-anomaly preservation, a synthetic anomaly is introduced and its preservation is tested. Practically, introducing a synthetic anomaly has an added advantage in that both its location and magnitude are known. Therefore, testing the ability to preserve genuine time-lapse anomalies can be performed with more confidence, and not only qualitatively but also quantitatively. This chapter will describe work conducted and results achieved while assessing ANN as a cross-equalizing tool that should preserve genuine time-lapse anomalies.

The chapter also contains a section that analysis an alternative cross-equalization procedure. It investigates post-stack cross-equalization, and compares the results achieved to those of pre-stack cross-equalization.

6.1 Preserving desired anomalies

A synthetic anomaly is introduced pre-stack between the Top Brent and the Base Brent horizons (Figure 3.11) in the base survey. The anomaly is between these two horizons, and in the CDP range 1050–1450 and the time interval 2280–2512 ms. The new pre-stack dataset will form our new base survey. Figure 6.1 shows the location of the introduced anomaly. In addition, Table 6.1 defines and sets abbreviations for the different datasets used. The abbreviations are introduced for simplification and in order to avoid confusion.

ANN will be used to produce a cross-equalized monitor survey dataset (*NewXEqMonitorDataset*) to the new base survey dataset (*NewBaseDataset*). Cross-equalization will be conducted as described in the previous chapters (Section 5.4.1). However, instead of the original base survey (*BaseDataset*) the new base survey with a synthetic anomaly (*NewBaseDataset*) will be used.

The introduced anomaly is different from the acquisition differences and can be treated as a genuine production-related time-lapse anomaly. This anomaly should be preserved by any successful cross-equalization. It follows that the *NewXEqMonitorDataset* should be a match to the original base survey (*BaseDataset*), which contains no anomaly. The *NewDifferenceDataset* (the difference dataset between the *NewXEqMonitorDataset* stack and the *NewBaseDataset* stack) should be similar to the *DifferenceDataset* (the difference dataset between the *XEqMonitorDataset* stack and the *BaseDataset* stack, e.g. Figure 5.35), but with the introduced anomaly preserved. Thus, whereas it was desirable to see the *DifferenceDataset* near zero, it is desirable to see the *NewDifferenceDataset* showing the introduced anomaly as in the *AnomalyDataset*, where *AnomalyDataset* is the difference between the *NewBaseDataset* stack and the *BaseDataset* stack.

Table 6.1: Definition of several important datasets used in Chapter 6. The main datasets used are defined with an abbreviation and a description. If the ANN cross-equalization scheme works perfectly, then the *DifferenceDataset* should be zero, and the *NewDifferenceDataset* should equal the *AnomalyDataset*.

Abbreviation	**Description**
BaseDataset	The original base survey acquired in 1989.
MonitorDataset	The original monitor survey acquired in 1992.
XEqMonitorDataset	The monitor survey after cross-equalization to match the base survey. Therefore, during ANNs training, the input is the *MonitorDataset* and the target output is the *BaseDataset*. The actual output is this dataset.
DifferenceDataset	The difference between the *BaseDataset* stack and the *XEqMonitorDataset* stack.
RawDifferenceDataset	The raw difference before any cross-equalization between the *BaseDataset* stack and the *MonitorDataset* stack.
NewBaseDataset	The base survey after introducing a pre-stack synthetic anomaly. This anomaly acts as a time-lapse anomaly between the *NewBaseDataset* and the *MonitorDataset*. Note that the anomaly size is different in the different sections within this chapter.
AnomalyDataset	The difference between the *NewBaseDataset* and the *BaseDataset*. Thus, this shows the actual anomaly introduced.
NewXEqMonitorDataset	The monitor survey after cross-equalization to match the *NewBaseDataset*. Therefore, during ANNs training, the input is the *MonitorDataset* and the target output is the *NewBaseDataset*. The actual output is this dataset.
NewDifferenceDataset	The difference between the *NewBaseDataset* stack and the *NewXEqMonitorDataset* stack.
NewRawDifferenceDataset	The raw difference before any cross-equalization between the *NewBaseDataset* stack and the *MonitorDataset* stack.

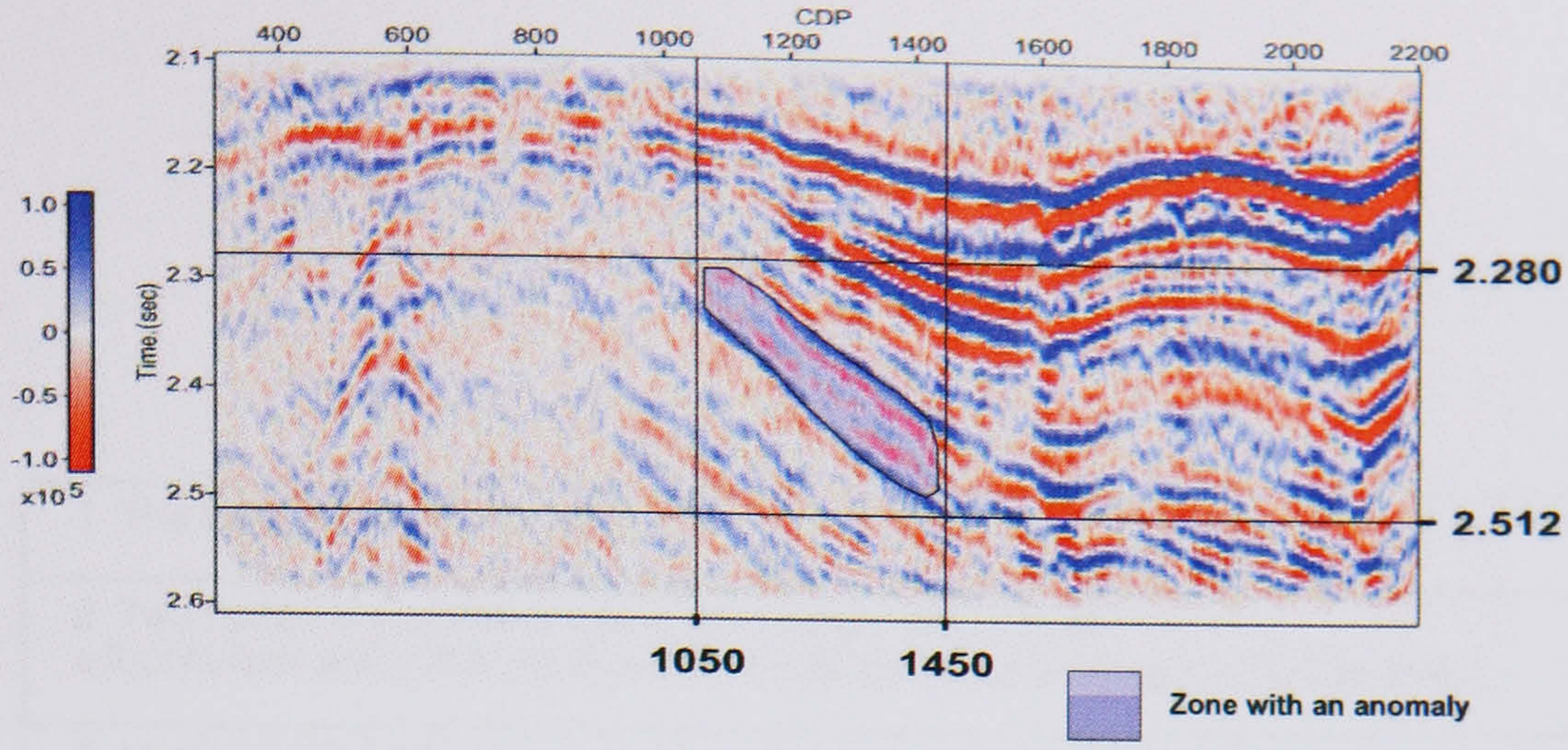


Figure 6.1: Location of the introduced synthetic anomaly on the base survey (*BaseDataset*) stack. The synthetic anomaly is introduced *pre-stack* between the Top Brent and the Base Brent horizons (Figure 3.11) in the *BaseDataset*. The anomaly is between these two horizons and in the CDP range 1050–1450 and the time interval 2280–2512 ms. Note that although the location of the anomaly is shown here after stacking, the anomaly is actually introduced pre-stack. The entire pre-stack dataset with the added anomaly forms the new base survey (*NewBaseDataset*). A cross-equalized dataset (*NewXEqMonitorDataset*) is to be produced from the monitor survey (*MonitorDataset*) that matches the *NewBaseDataset*, but preserves the introduced anomaly.

6.1.1 Designing the anomaly

The choice was made to set the introduced anomaly as a percentage gain in amplitude at the desired location. It is also desired that this anomaly is not abrupt in nature and should have smooth edges. This ensures that the data are not severely changed in character by altering the bandwidth through introducing sharp edges.

The synthetic anomaly is designed in the seismic processing package ProMAX. The anomaly is designed by: firstly, making a copy of the pre-stack *BaseDataset* and setting every sample value to one; secondly, applying a fractional anomaly between the Top and the Base Brent horizons as picked in the *BaseDataset* stack section (e.g. set the sample values to 1.10 to get a 10% gain); thirdly, a reverse move-out is applied to the *BaseDataset* copy to make sure the anomaly is at the right pre-stack position; and finally, multiplying the edited *BaseDataset* copy with the original pre-stack *BaseDataset* to generate a new base survey dataset with an anomaly (*NewBaseDataset*). Figure 6.2 shows all the major steps involved.

1. The base survey is stacked.

2. Two horizons are picked in the CDP range 1050--1450:
aTopHorizon along the Top Brent; and *aBottomHorizon* along the Bottom Brent.

3. Make a copy of the whole pre-stack base survey. Save it as *aBaseCopy*.

4. In *aBaseCopy*, set all samples equal to one.

5. Make a repeat of each trace in *aBaseCopy*, i.e. each trace has an equivalent copy.

6. **For Each** second copy of traces in *aBaseCopy:*

   6.1. Apply a top mute along *aTopHorizon* with a 15ms ramp.

   6.2. Apply a bottom mute along *aBottomHorizon with a 15ms ramp*.

   6.3. Multiply each sample by a set fractional value (*aValue*) between +1 and -1.

   6.4. Apply a trace weighting, where the value of a sample is ramped from zero to 100% from CDP 1050 to 1100 and from 100% to zero from CDP 1400 to 1450.

7. Add the two copies of each trace in *aBaseCopy*.

8. Apply a reverse normal move-out: This moves the introduced anomaly to the correct time position for pre-stack data.

9. Save the dataset as *aBaseAnomaly*. This dataset holds the required fractional anomaly to be applied on pre-stack data of the base survey.

10. Multiply each sample in the pre-stack base survey with that in *aBaseAnomaly*.

11. Save the new dataset as the new base survey with added synthetic anomaly.

Figure 6.2: Structural chart showing how the synthetic anomaly is designed and applied to the base survey. The anomaly applied on the base survey copy, *aBaseCopy*, in (4)–(7) is positioned according to the horizon picking on the base survey stack in step (2). Therefore, step (8) makes sure that the fractional anomaly is reverse moved-out to be appropriate to be applied to a pre-stack dataset. The mute ramping in (6.1) and (6.2) and the trace weighting in (6.4), ensures that the anomaly applied has smooth boundaries, and is limited in CDP and time range.

6.1.2 Quantifying the anomaly

The anomaly was designed as a certain percentage gain in amplitude. This percentage anomaly (or fractional anomaly) was set at a location defined by horizons picked on the *BaseDataset* stack section. Since the velocity model is not absolutely accurate, the anomaly introduced is not maintained in exact location and magnitude after reverse move-out to produce a pre-stack dataset with anomaly and after normal move-out when processing the new dataset to stack. Therefore, another measure is defined to quantify the anomaly.

The anomaly introduced is localized in the CDP range 1050–1450 and the time interval 2280–2512 ms between the Top Brent and the Base Brent horizons. Therefore, to compare a dataset (e.g. the *BaseDataset*, or a *NewXEqMonitorDataset*) to the *NewBaseDataset*, a measure is defined to compare the total RMS amplitude of each dataset at the anomaly location. Thus, if the *NewBaseDataset* stack is x and the dataset compared to it is y, then the anomaly is measured as follows:

$$\text{Anomaly} = 100 \times \frac{\text{RMS}_{\text{loc}}(x) - \text{RMS}_{\text{loc}}(y)}{\text{RMS}_{\text{loc}}(x)}. \tag{6.1}$$

The root mean square operator at the anomaly location (RMS_{loc}) is defined as:

$$\text{RMS}_{\text{loc}}\big(x(t)\big) = \sqrt{\frac{\sum_{\text{CDP}_1}^{\text{CDP}_2} \sum_{t_1}^{t_2} \big(x(t)\big)^2}{N}}, \tag{6.2}$$

where N is the number of samples in the CDP interval CDP_2–CDP_1 (1050–1450) and the time interval t_1–t_2. The time interval is defined as the interval between the Top Brent and the Base Brent horizons, at the CDP range 1050–1450 and within the time interval 2280–2512 ms.

Clearly, Equation 6.1 allows a measure of both the original synthetic anomaly introduced and the preserved anomaly in the cross-equalized datasets. The original anomaly introduced is quantified by setting y to be the *BaseDataset* stack. On the other hand, the preserved anomaly in the cross-equalized dataset is measured by setting y to be the

NewXEqMonitorDataset stack.

It is important to note that the synthetic anomaly is introduced in a region of the dataset where amplitudes are approximately a factor of four lower than the much stronger events immediately above. Since detecting week anomalies below stronger reflectors is difficult, it may have been more appropriate to use the amplitude of the strong events $\mathrm{RMS}_{\mathrm{strong}}(x)$ to normalize the difference in Equation 6.1. In addition, usually an anomaly is quoted by summing the RMS over the entire trace and normalizing likewise, i.e. the time interval for $\mathrm{RMS}_{\mathrm{loc}}$ becomes 2100–2612 ms. If the later is done, then the % introduced anomalies discussed in Sections 6.1.3, 6.1.4 and 6.1.5 become 5.5%, 3.0% and 1.9% instead of the quoted 23.9%, 14.5%, and 9.6%, respectively.

6.1.3 Assessing preservation of a desired 23.9% anomaly

The first test conducted introduced a relatively large anomaly to the pre-stack dataset of the original base survey (offset gathers of channels 101–230). The stack of the original base survey (*BaseDataset*) is shown in Figure 6.3, and the stack of the new base survey with the introduced anomaly (*NewBaseDataset*) is shown in Figure 6.4. The anomaly introduced has a value of 23.9%. This anomaly is clear in Figure 6.5, which displays the *AnomalyDataset*, i.e. the difference between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.4).

ANNs were used to cross-equalize the *MonitorDataset* to match the *NewBaseDataset*. The configuration is identical to that reported in Section 5.4.1. Figure 6.6 shows the stack plot of the cross-equalized dataset (*NewXEqMonitorDataset*). The plot shows ANN ability to match in such a way that normalizes acquisition differences and retains the synthetic. This ability is well illustrated in the *NewDifferenceDataset* shown in Figure 6.7, which is similar to Figure 6.5 as it should be.

Figure 6.8 shows a zoom at the anomaly location and compares the *NewDifferenceDataset* to the *AnomalyDataset*, and thus allows a better assessment of the preservation of the anomaly or otherwise. This figure shows a further assurance that the anomaly

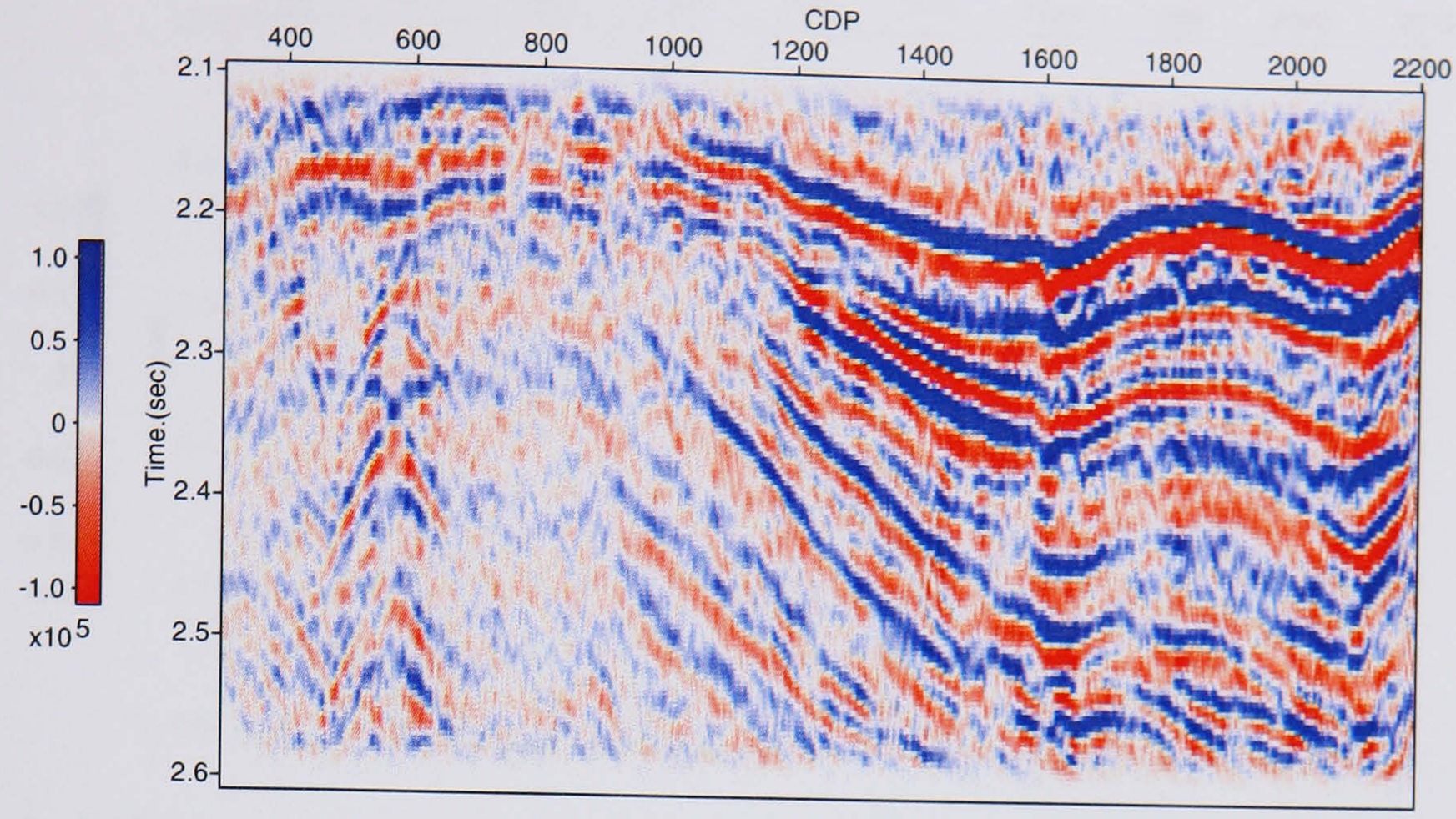


Figure 6.3: Seismic stack plot of all the offset gathers of channels 100–230 of the original base survey (*BaseDataset*).

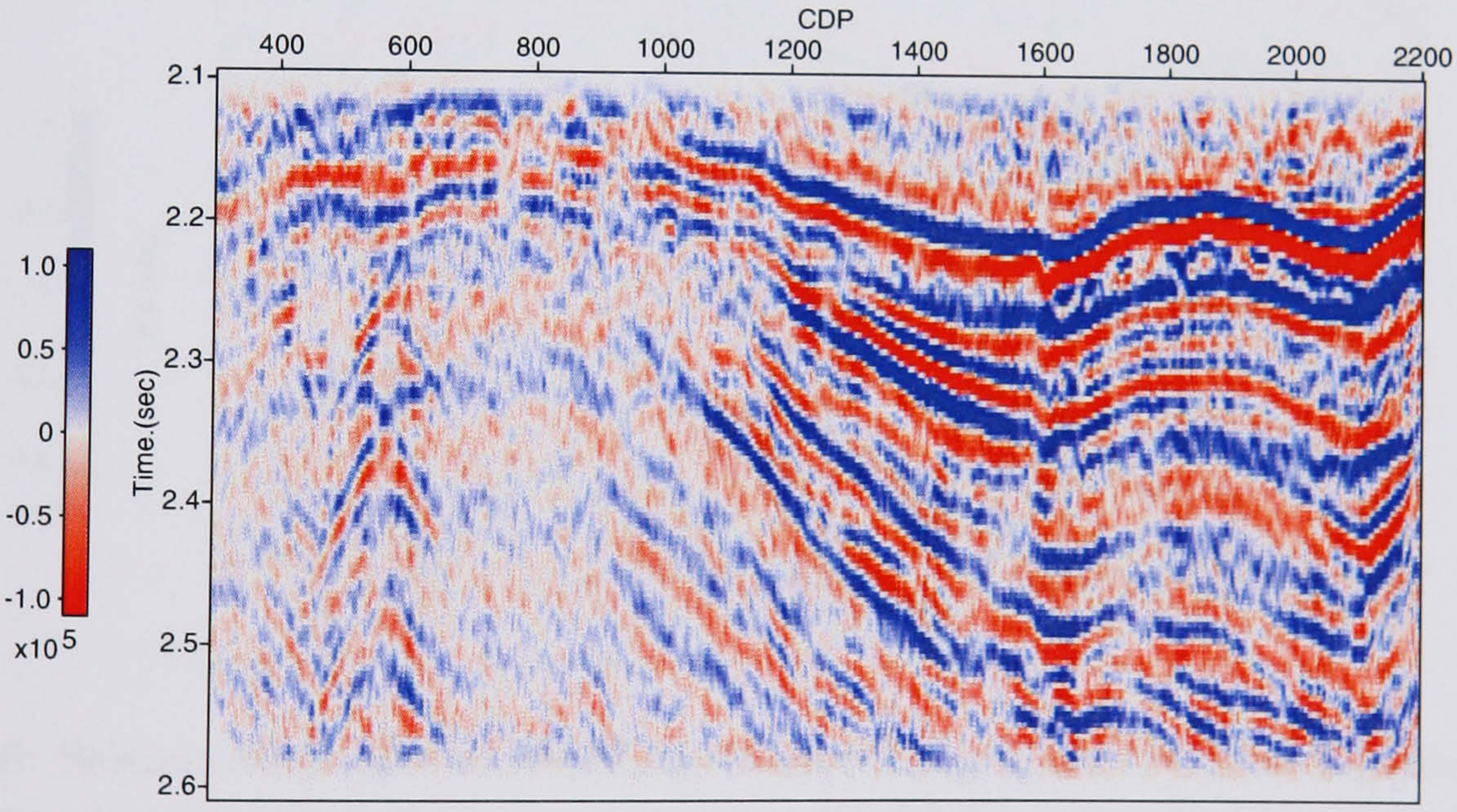


Figure 6.4: Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with a 23.9% anomaly (*NewBaseDataset*). The anomaly is introduced at the location defined in Figure 6.1. ANNs were used to produced a cross-equalized pre-stack dataset (*NewXEqMonitorDataset*) from the monitor survey (*MonitorDataset*) to match those used to produce this stack.

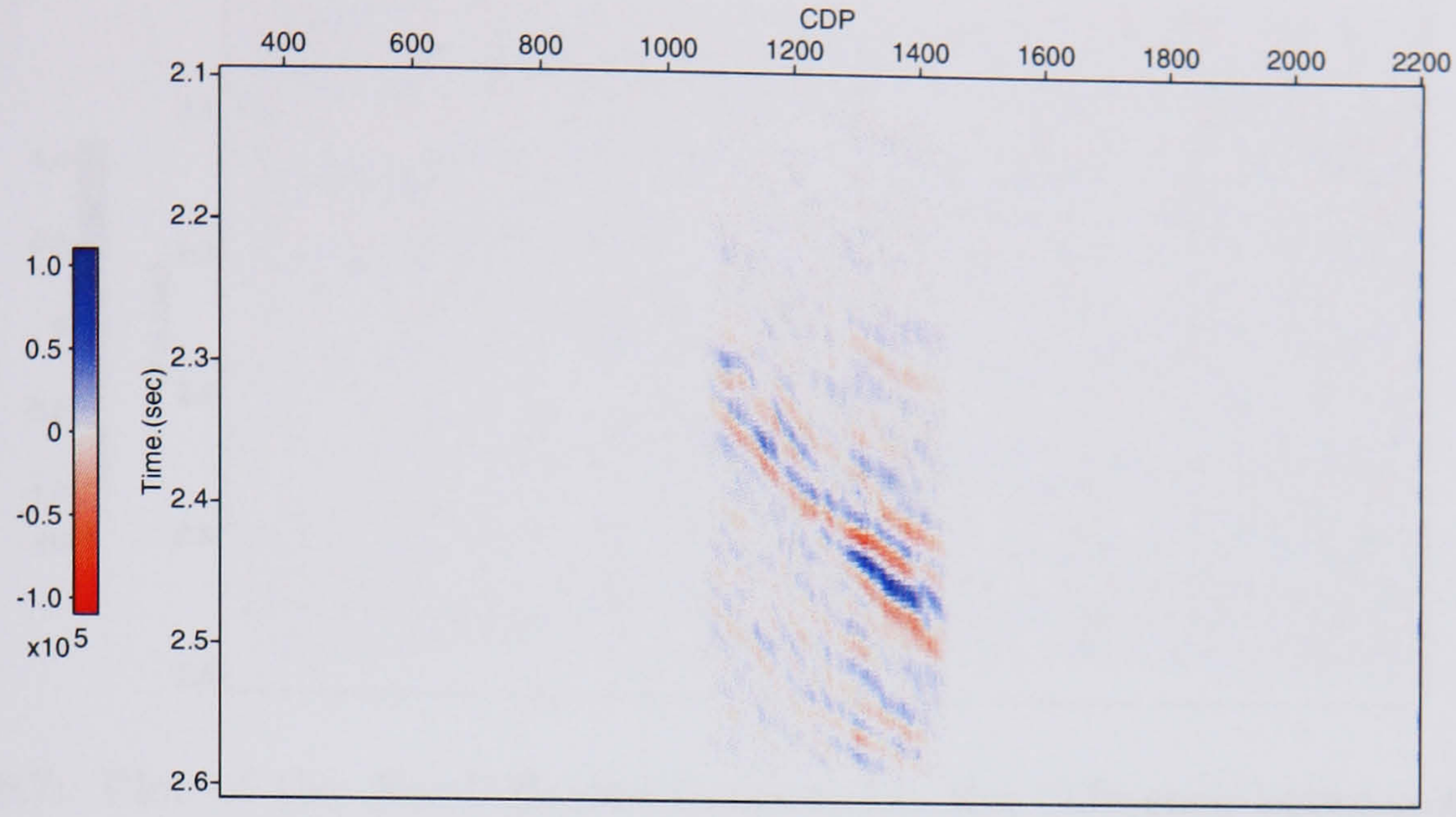


Figure 6.5: Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *New-BaseDataset* stack (Figure 6.4), showing the introduced 23.9% anomaly.

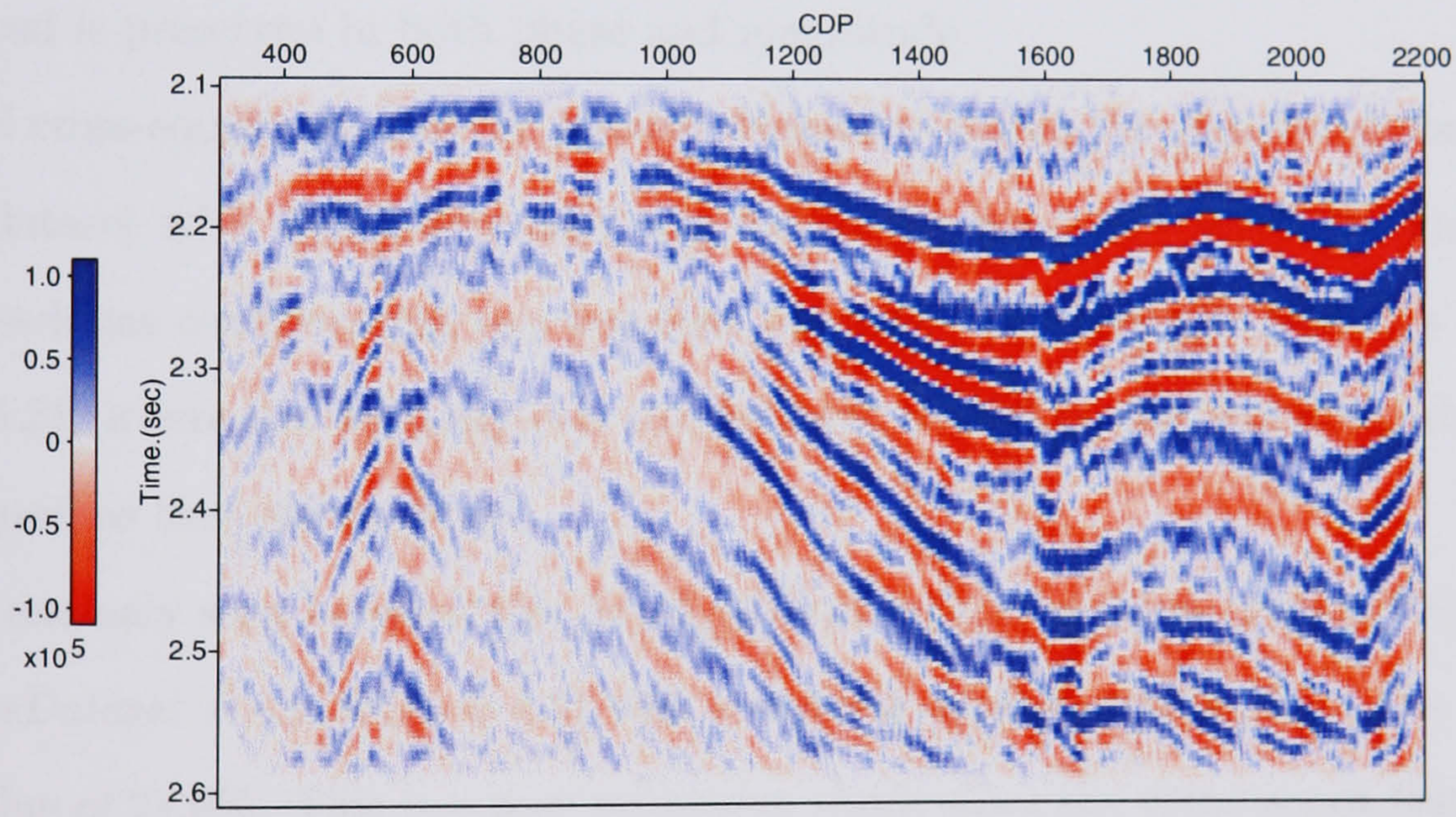


Figure 6.6: Seismic stack plot of *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 23.9% anomaly (*NewBaseDataset*). This can be compared to the *NewBaseDataset* stack (Figure 6.4) and the *BaseDataset* stack (Figure 6.3), plotted to the same scale. The plots are similar and thus the matching is very good. The matching also shows that the *NewXEqMonitorDataset* stack is more similar and near identical to the *BaseDataset* stack (compare at CDP 1050–1450 and time 2280–2512 ms). Therefore, ANN preserved the anomaly introduced in the *NewBaseDataset* (Figure 6.4) by not replicating it here in the *NewXEqMonitorDataset*.

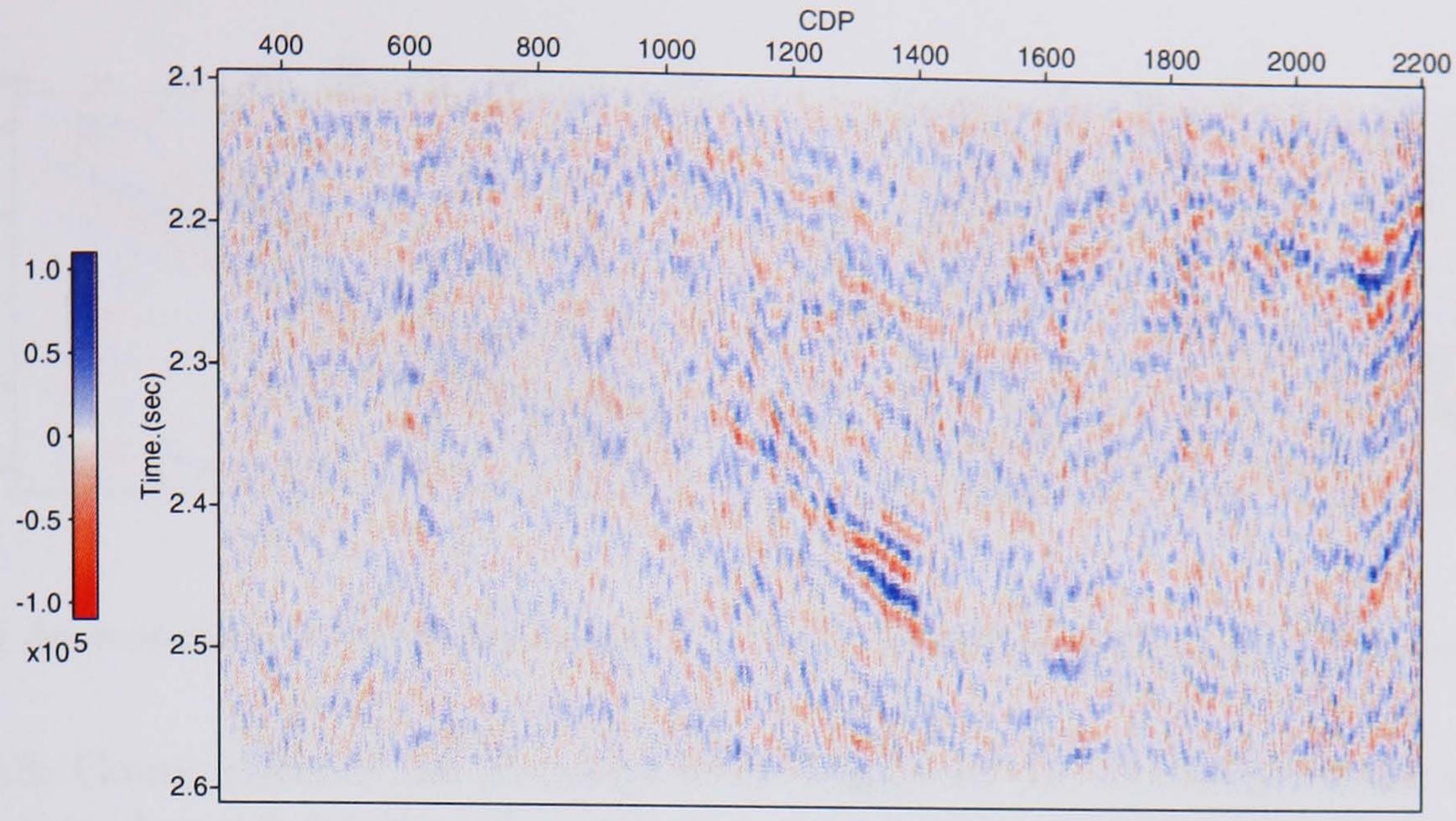


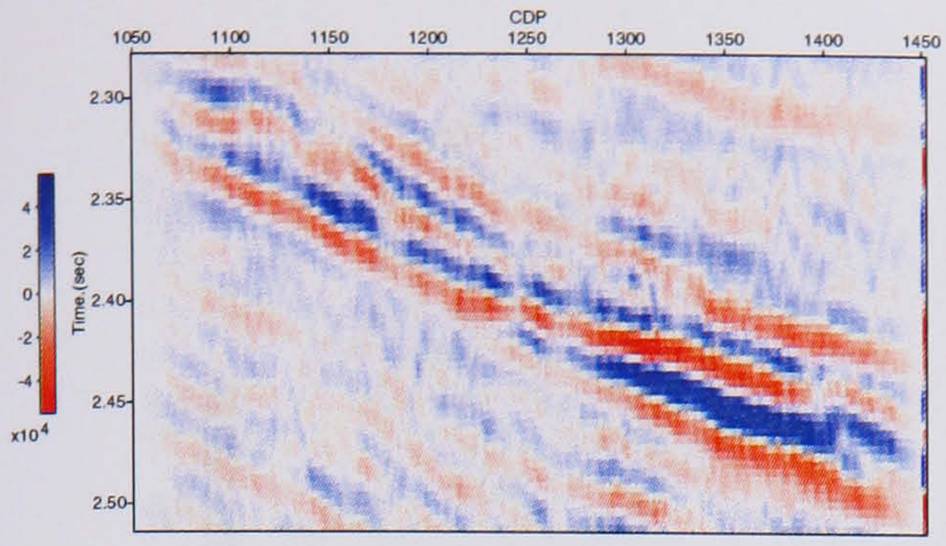
Figure 6.7: Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.4) and the *NewXEqMonitorDataset* stack (Figure 6.6). This shows a very low error in matching. In addition, the cross-equalization has preserved the introduced 23.9% anomaly, which is clear in this difference plot and is not masked by matching error unlike in the linear cross-equalization difference plots (*cf.* Figure 6.9–6.13).

introduced is preserved in both phase and magnitude.

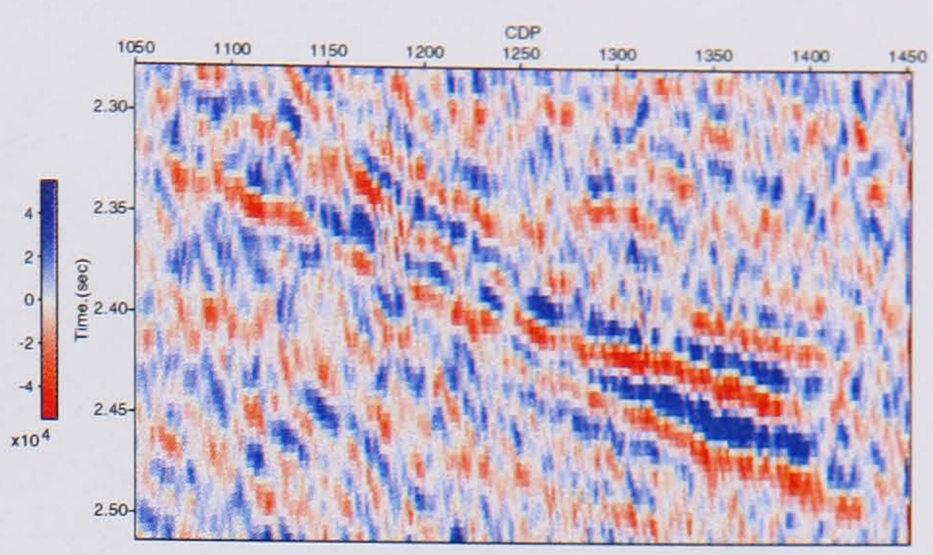
ANN cross-equalization of the *MonitorDataset* to match the *NewBaseDataset* (base survey dataset with 23.9% anomaly) was conducted previously when the cross-equalization–scheme configuration was less than optimal. The difference plot was shown in Figure 5.21, where the anomaly can also be seen. However, the matching ability was not as good as that shown here.

The anomaly seen between the *NewXEqMonitorDataset* stack (Figure 6.6) and the *NewBaseDataset* stack (Figure 6.4) was measured using Equation 6.1. The anomaly has a value of 21.0%. This is a very promising result since the value is not just positive like the introduced one, but is also very close to the actual introduced value (23.9%).

The reason for the discrepancy, between the original introduced anomaly value (23.9%) and that preserved by ANN cross-equalization (21.0%), is that the networks are actually asked to match all the *NewBaseDataset* including that portion of data where the anomaly was introduced. It is expected, because the anomaly location is small in comparison with the total matched area, that ANNs would manage to preserve the



(a) *AnomalyDataset* (23.9% anomaly)



(b) *NewDifferenceDataset* (23.9% anomaly)

Figure 6.8: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 23.9%. The plots are plotted with a higher scale than previously used to enable better comparison. The plot re-emphasizes that the anomaly introduced is well preserved in phase and magnitude. At the righthand-side area, where a stronger anomaly is introduced, the phase and magnitude are near identical. However, in areas with lower introduced anomaly there is some overprinting by noise, but nevertheless the anomaly character is reasonably maintained.

introduced anomaly. Nevertheless, it should be expected that ANN cross-equalization, as defined, will cause some reduction in the amplitude of the anomaly. Furthermore, ANN matching has a degree of error in matching and thus the anomaly value might be affected also by introduced noise.

Linear cross-equalization

The cross-equalization is repeated using the scalar-misfit linear method (Section 1.2.2), which was used in Section 5.4.2. The linear cross-equalization was performed using different modes (see Section 5.4.2): local mode, a set of semi-global modes, and a global mode. The difference plots between the *NewBaseDataset* stack and the stacks of the linearly cross-equalized datasets are shown in Figure 6.9–6.13, from local to increasingly more global mode. The local mode should have some ability to preserve the introduced anomaly since it globalizes in the time domain. However, it was fair to test more global modes to see if preservation of the anomaly is improved.

It is clear, from the difference plots (Figure 6.9–6.13), that more global modes suffer from worse matching, which overweighs their possible ability to preserve more

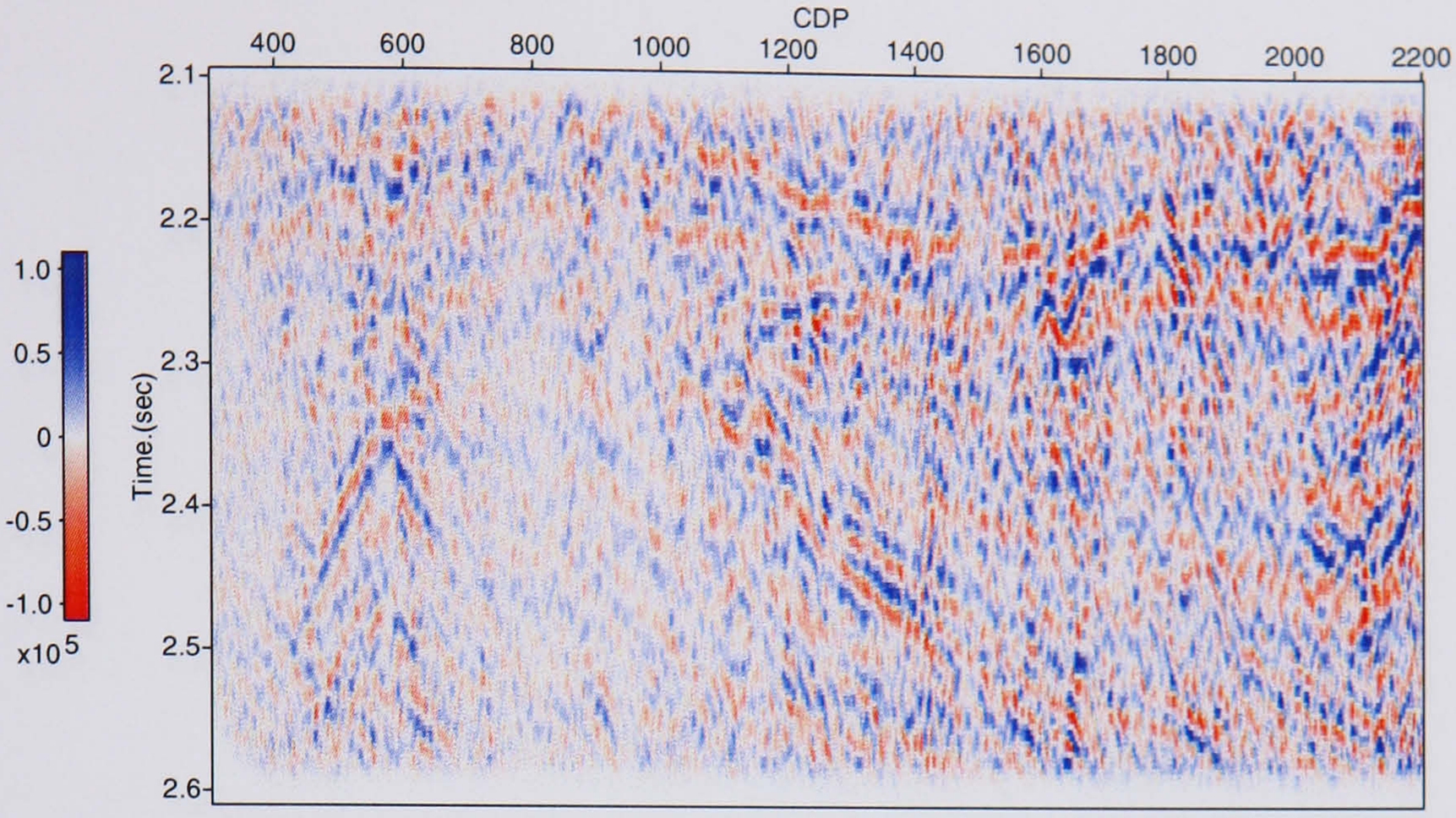


Figure 6.9: Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with local mode. The result shows worse matching than that achieved by ANN (*cf.* Figure 6.7). In addition, the introduced anomaly (Figure 6.5), although reasonably preserved, is hard to pick from amongst other matching error.

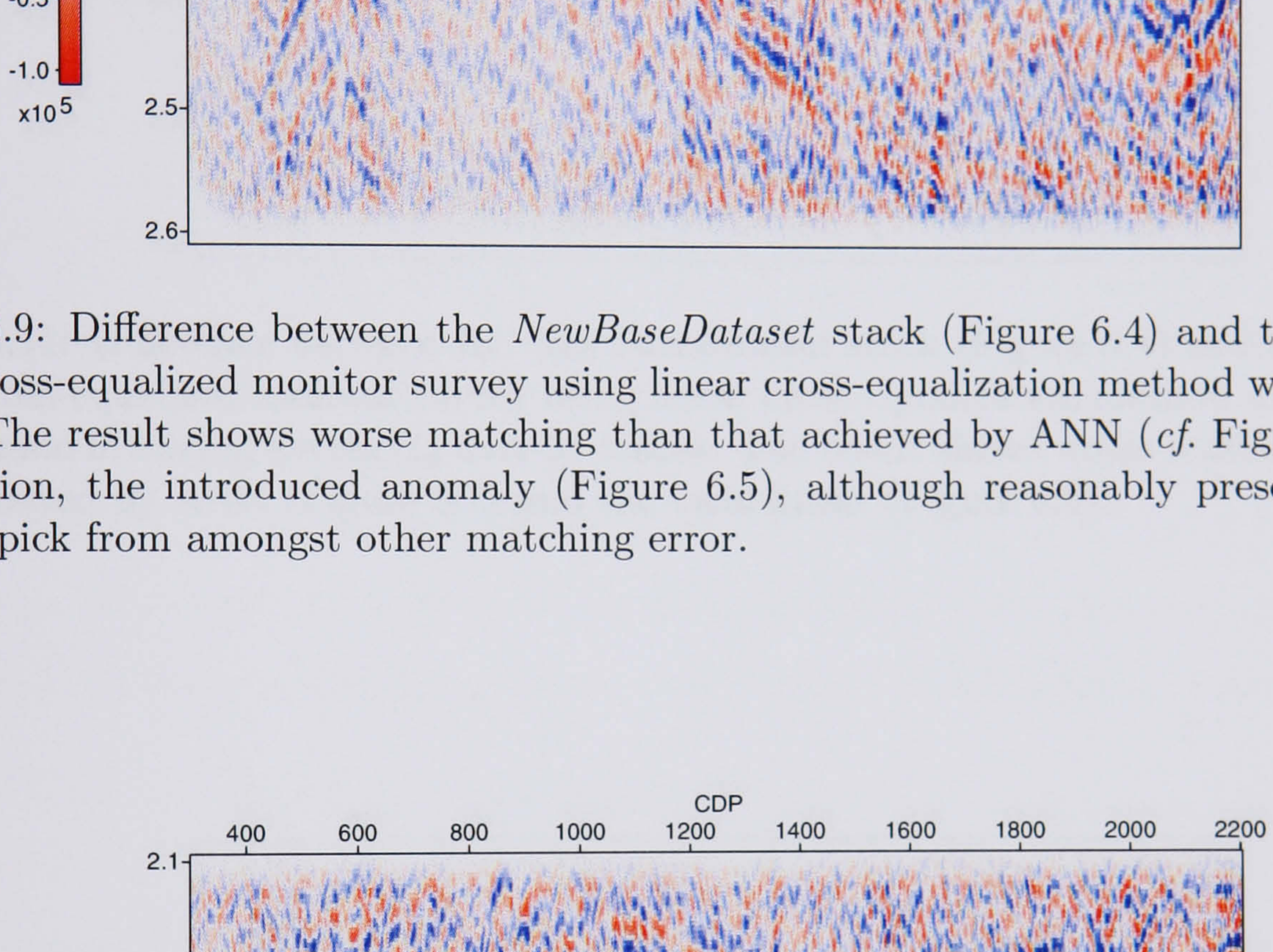


Figure 6.10: Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 11 traces. The result shows worse matching than that achieved by ANN (Figure 6.7), but very similar to the local linear (Figure 6.9).

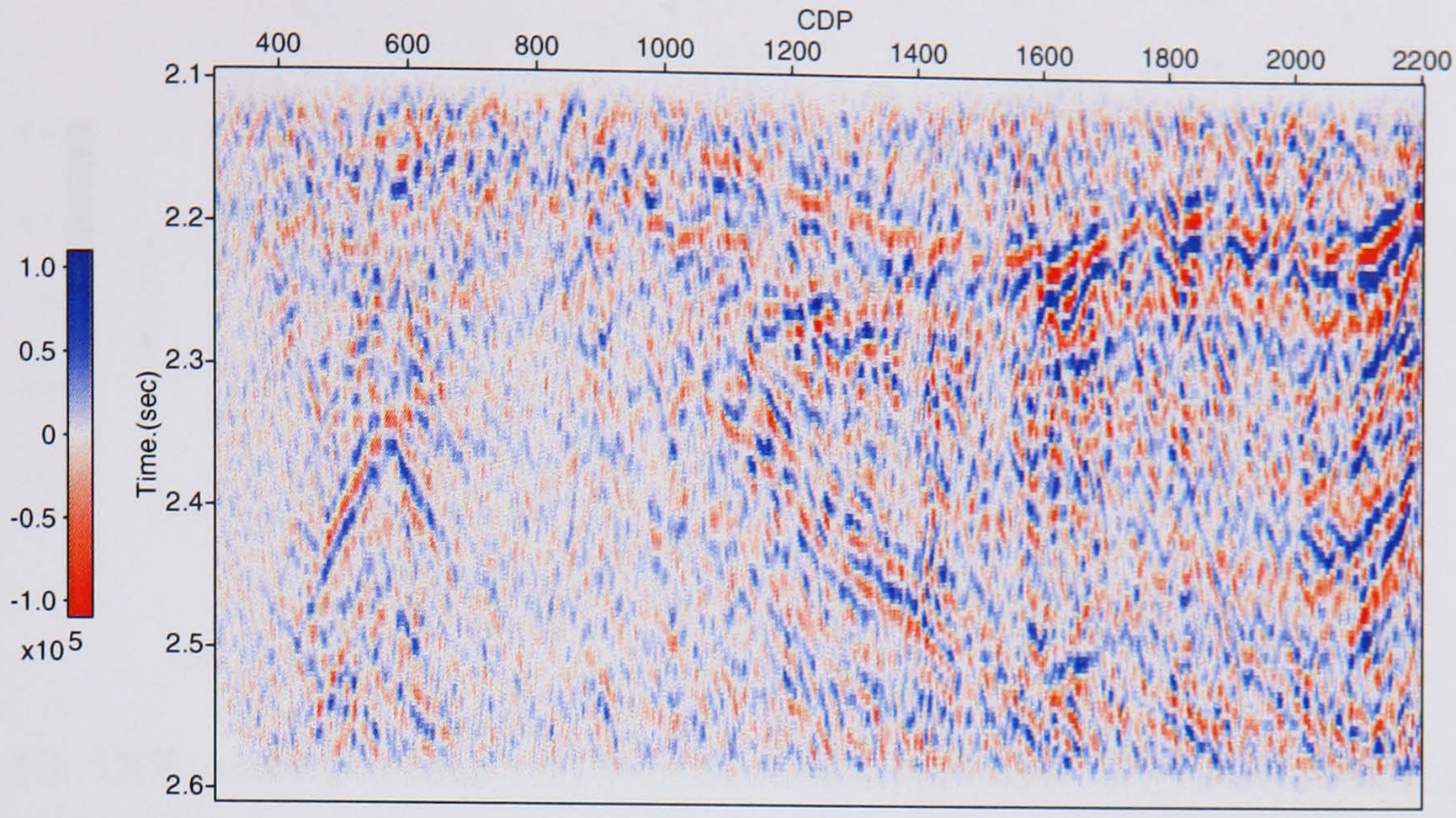


Figure 6.11: Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 21 traces. The result shows worse matching than that achieved by ANN (Figure 6.7) and the local linear (Figure 6.9).

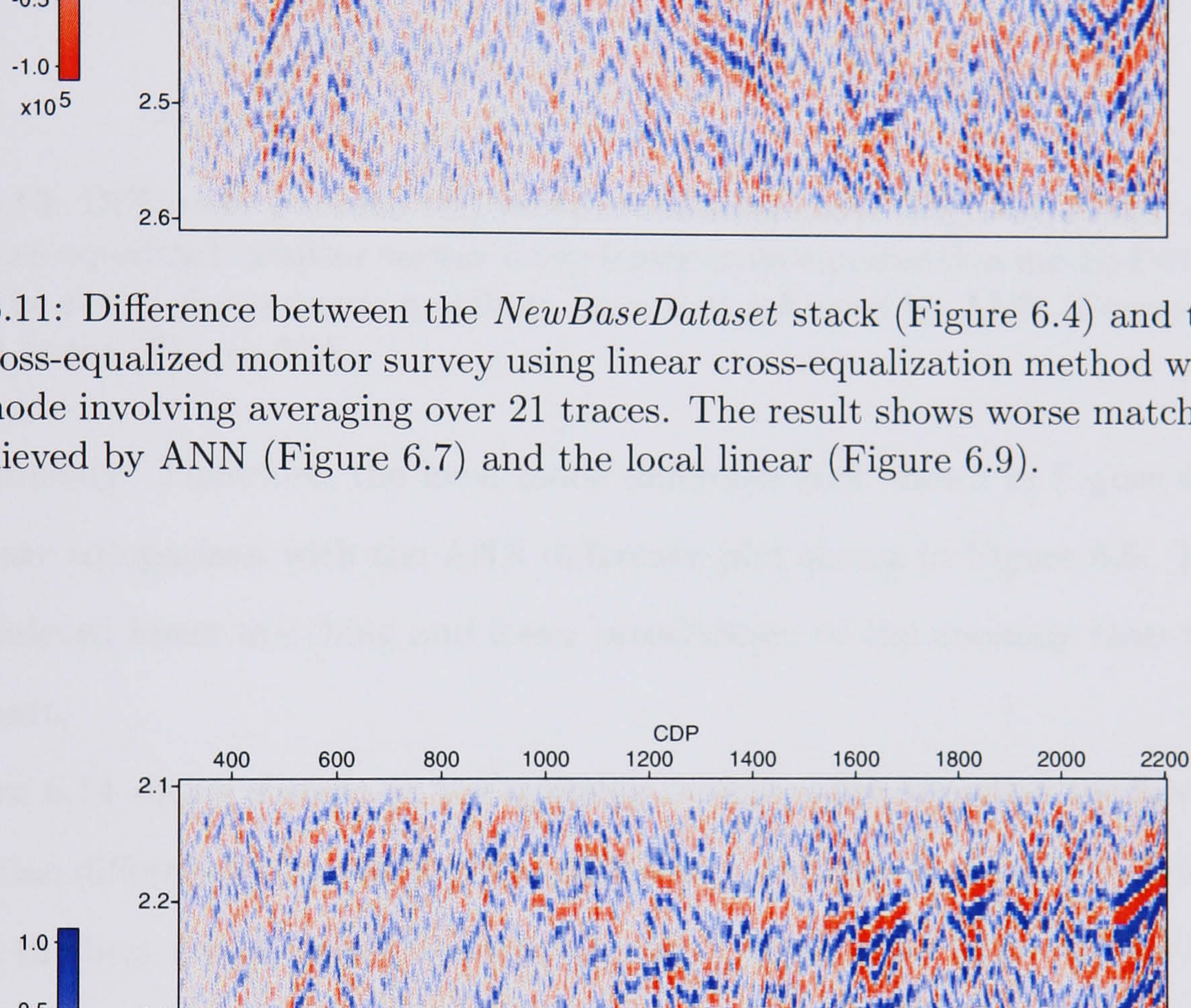


Figure 6.12: Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with semi-global mode involving averaging over 51 traces. The result shows worse matching than that achieved by ANN (Figure 6.7) and the local linear (Figure 6.9).

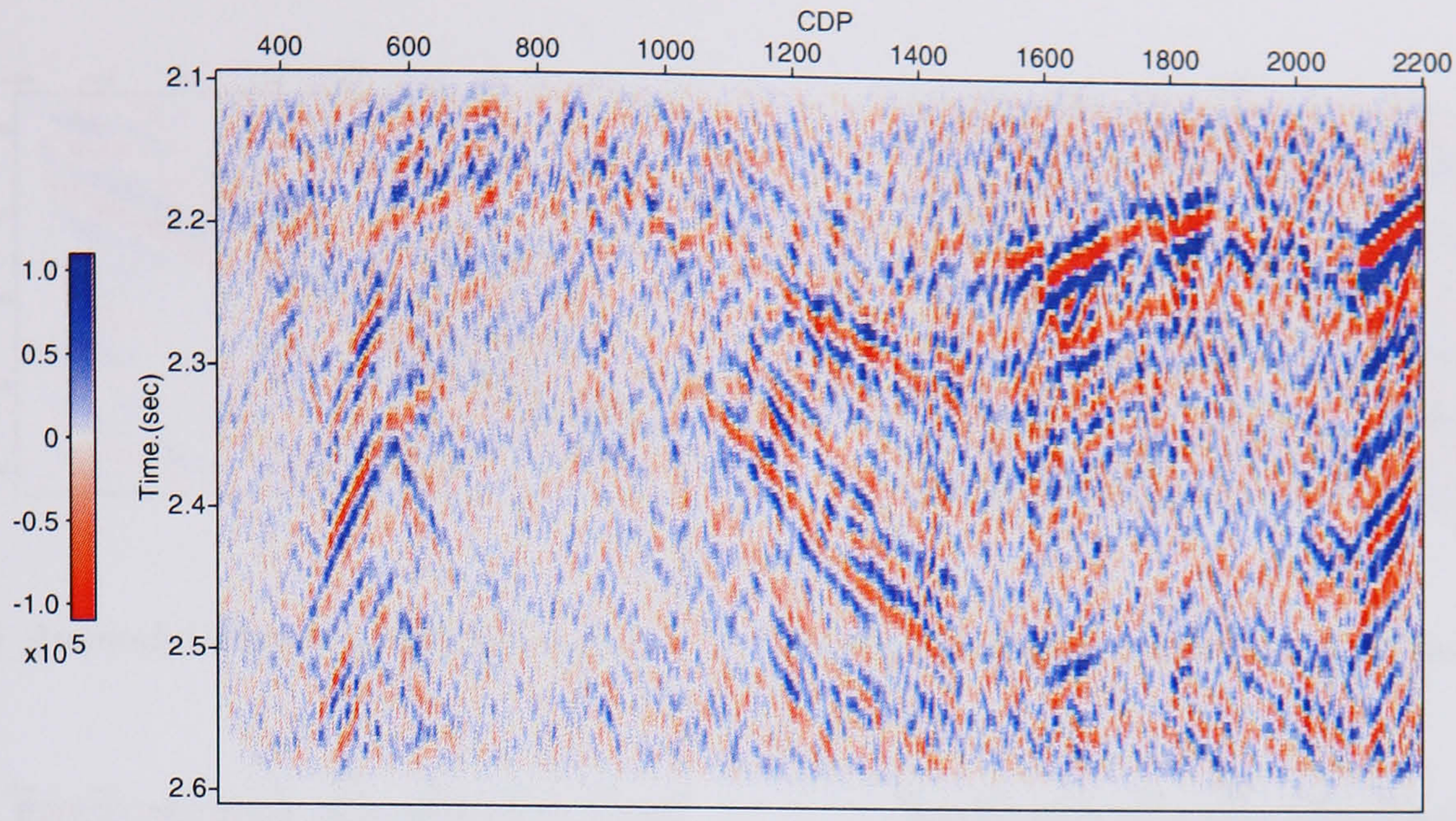
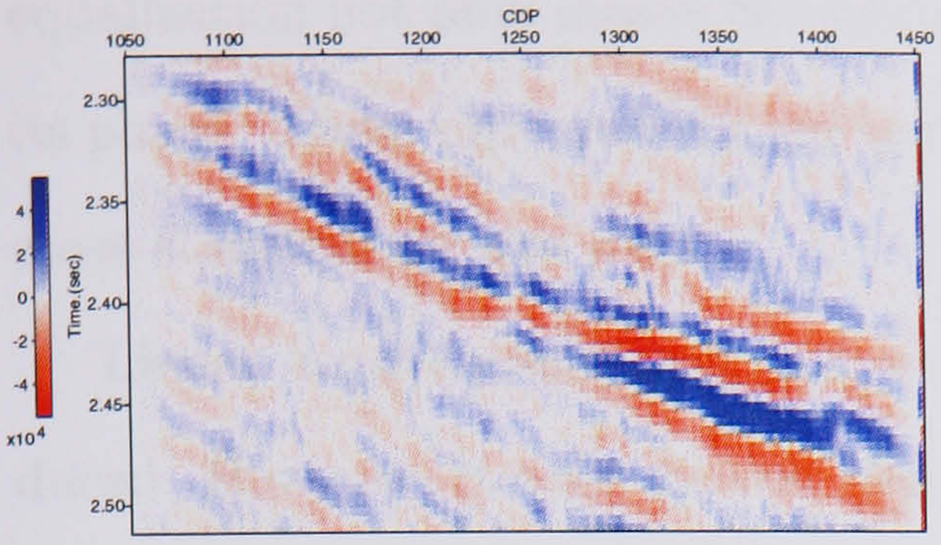


Figure 6.13: Difference between the *NewBaseDataset* stack (Figure 6.4) and the stack of the cross-equalized monitor survey using linear cross-equalization method with global mode. The result shows worse matching than that achieved by ANN (Figure 6.7) and the local linear (Figure 6.9).

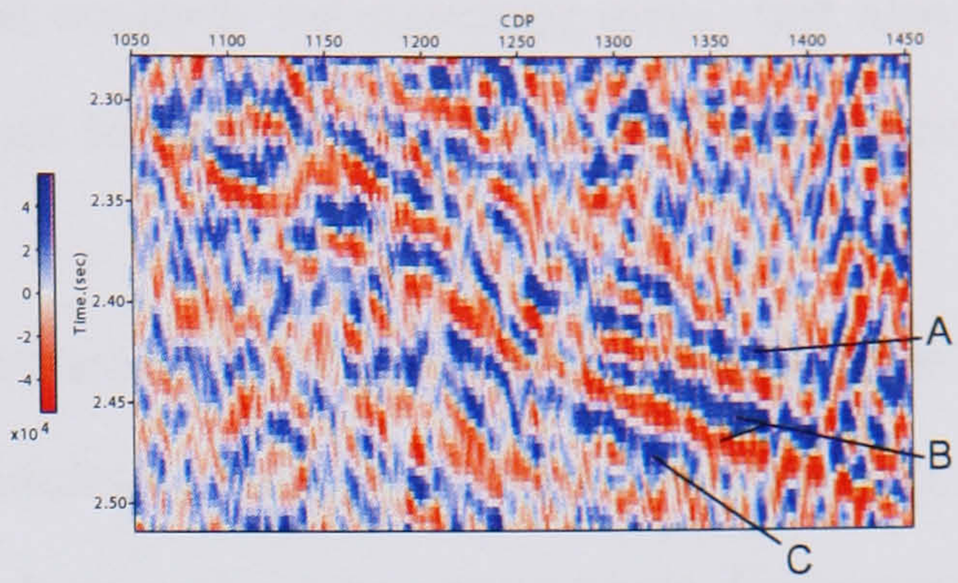
of the anomaly. Therefore, the local mode difference plot shown in Figure 6.9 is the best Linear comparison with the ANN difference plot shown in Figure 6.6. This local linear achieved lower matching and lower preservation of the anomaly than the ANN counterpart.

Figure 6.14 shows a zoom at the anomaly location and compares the linear cross-equalization difference plots to the *AnomalyDataset*. It shows that the local linear mode achieved the best result among the other modes. However, the figure further indicates that the preservations seen by ANN cross-equalization is not matched here by any of the modes of linear cross-equalization, where matching error masks the introduced anomaly.

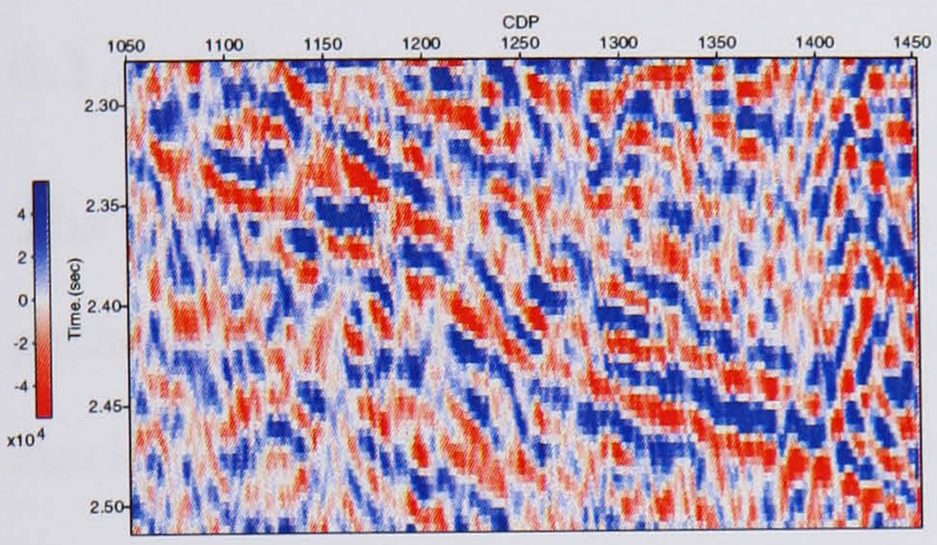
Looking closely at Figure 6.14(b), we see that the local linear cross-equalization misses the red phase before "A", gets the relative strength of the blue and red events reversed at "B", and introduces a spurious blue phase at "C". The alterations may suggest that the phase is wrong after linear cross-equalization. The reason for this could simply be because during linear matching the optimum phase rotation required for the full trace is not the optimum at the anomaly location. Therefore, linear cross-



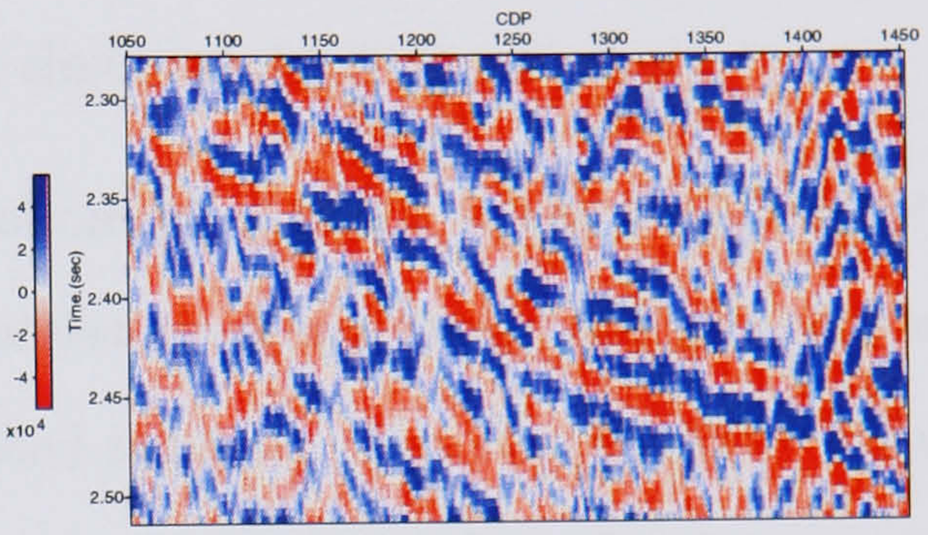
(a) *AnomalyDataset* (23.9% anomaly)



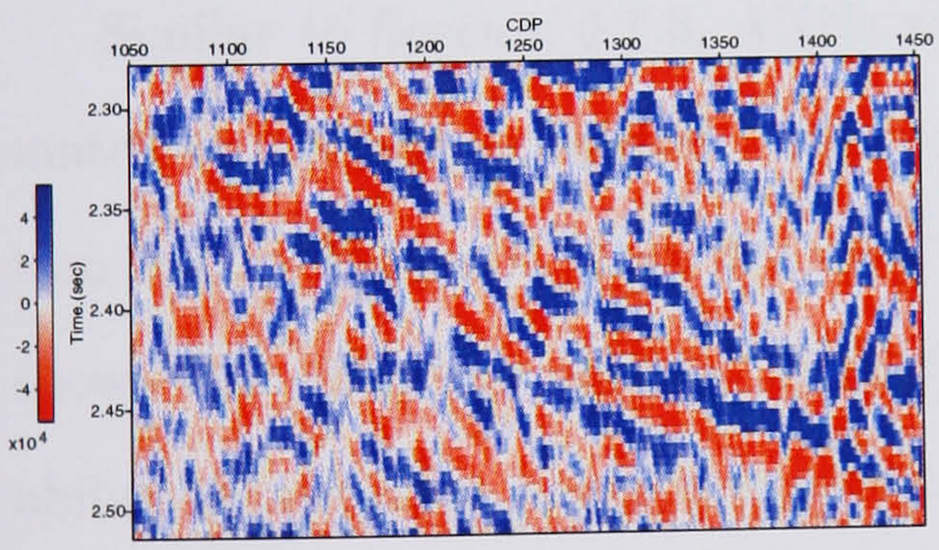
(b) *NewDifferenceDataset* local linear



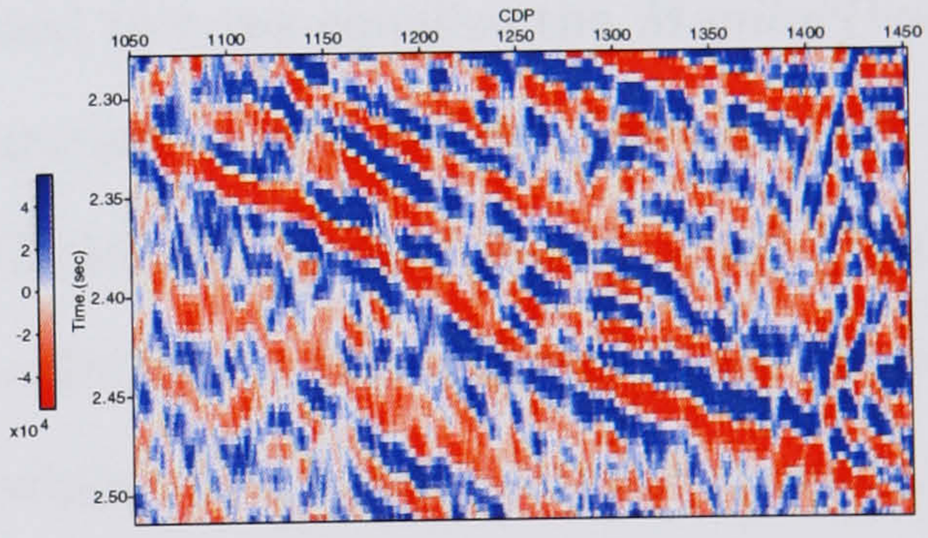
(c) *NewDifferenceDataset* semi-global(11) linear



(d) *NewDifferenceDataset* semi-global(21) linear



(e) *NewDifferenceDataset* semi-global(51) linear



(f) *NewDifferenceDataset* global linear

Figure 6.14: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when linear cross-equalization is used and an anomaly of 23.9% is introduced. The plots are plotted with a higher scale than previously used to enable better comparison. The local mode is the most similar to (a), and more structural leakage (matching error) is seen with more global modes. In addition, the local mode shows a reasonable anomaly preservation, since it approximately maintains the character of the anomaly especially at area with the strongest introduced anomaly (around CDP 1350 and time 2450 ms). However, this preservation is inferior to that seen by ANN cross-equalization (*cf.* Figure 6.8). Labels "A", "B", and "C" shown on plot (b) point at a few positions where the anomaly character is altered. These are further discussed in text.

equalization not only masks the introduced anomaly by matching error, but also alters its phase. These observations are in contrast to that seen in Figure 6.8(b), where ANN cross-equalization was used.

Linear cross-equalization was also conducted when smaller anomalies were introduced. However, it only showed similar results to those seen here. Therefore, it seems sufficient to show linear cross-equalization here and not to repeat their discussion when smaller anomalies are introduced in the following two sections.

6.1.4 Assessing preservation of a desired 14.5% anomaly

The second test conducted introduced an anomaly of 14.5% to the *BaseDataset* (offset gathers of channels 101–230). The *BaseDataset* stack was shown in Figure 6.3, and the stack of the new base survey with the added anomaly (*NewBaseDataset*) is shown in Figure 6.15. The anomaly is clear in Figure 6.16, which displays the *AnomalyDataset*, i.e. the difference between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.15).

Similar to Section 6.1.3, ANNs were used to cross-equalize the *MonitorDataset* to match the *NewBaseDataset*. The configuration is identical to that reported in Section 5.4.1. Figure 6.17 shows the *NewXEqMonitorDataset* stack, while Figure 6.18 shows the difference section *NewDifferenceDataset*. The difference plot shows ANN ability to normalize acquisition differences while retaining the synthetic anomaly. Therefore, the synthetic time-lapse anomaly is clear on the *NewDifferenceDataset* and is not masked by matching error.

Figure 6.19 shows a zoom at the anomaly location and compares the *NewDifference-Dataset* to the *AnomalyDataset*, and thus allows a better assessment of the preservation of the anomaly or otherwise. This figure shows a further assurance that the anomaly is preserved in both phase and magnitude, although some obliteration by noise is seen. However, contrary to Figure 6.8, here the matching error has a visual masking effect to parts of the introduced anomaly.

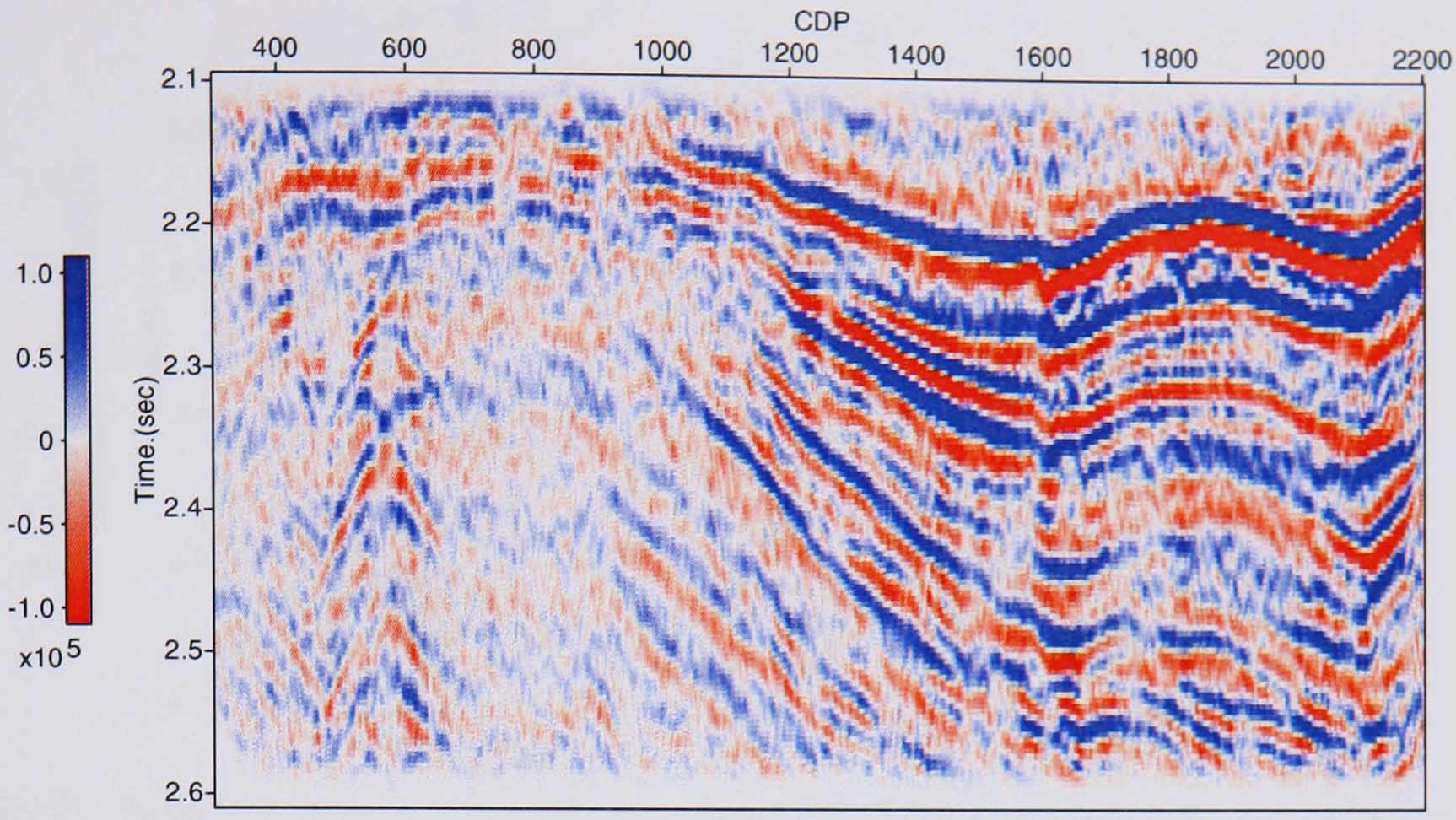


Figure 6.15: Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with 14.5% anomaly (*NewBaseDataset*). The anomaly is introduced at the location defined in Figure 6.1. ANNs were used to produced a cross-equalized pre-stack dataset (*NewXEqMonitorDataset*) from the monitor survey (*MonitorDataset*) to match those used to produce this stack.

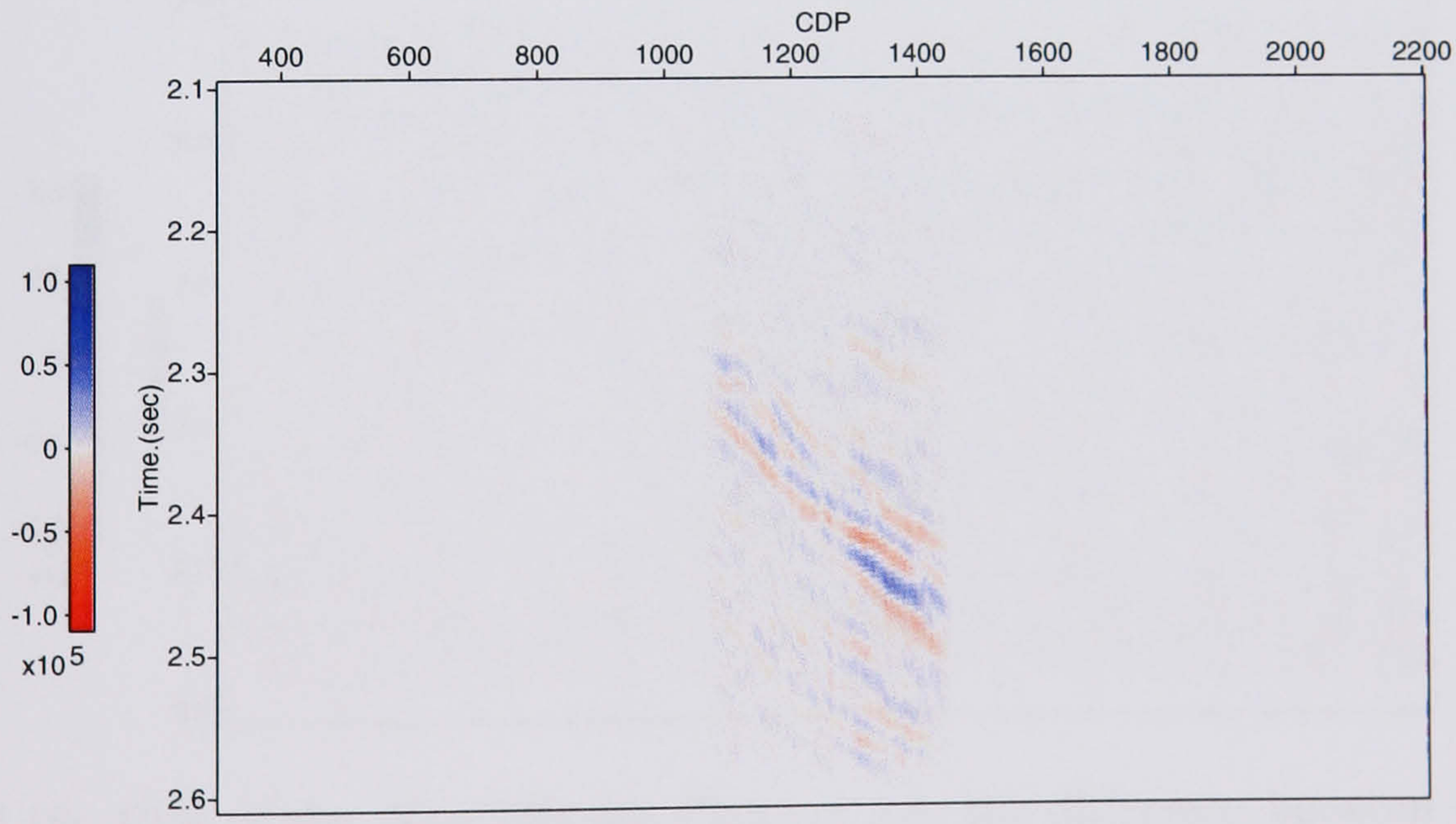


Figure 6.16: Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.15), showing the introduced 14.5% anomaly.

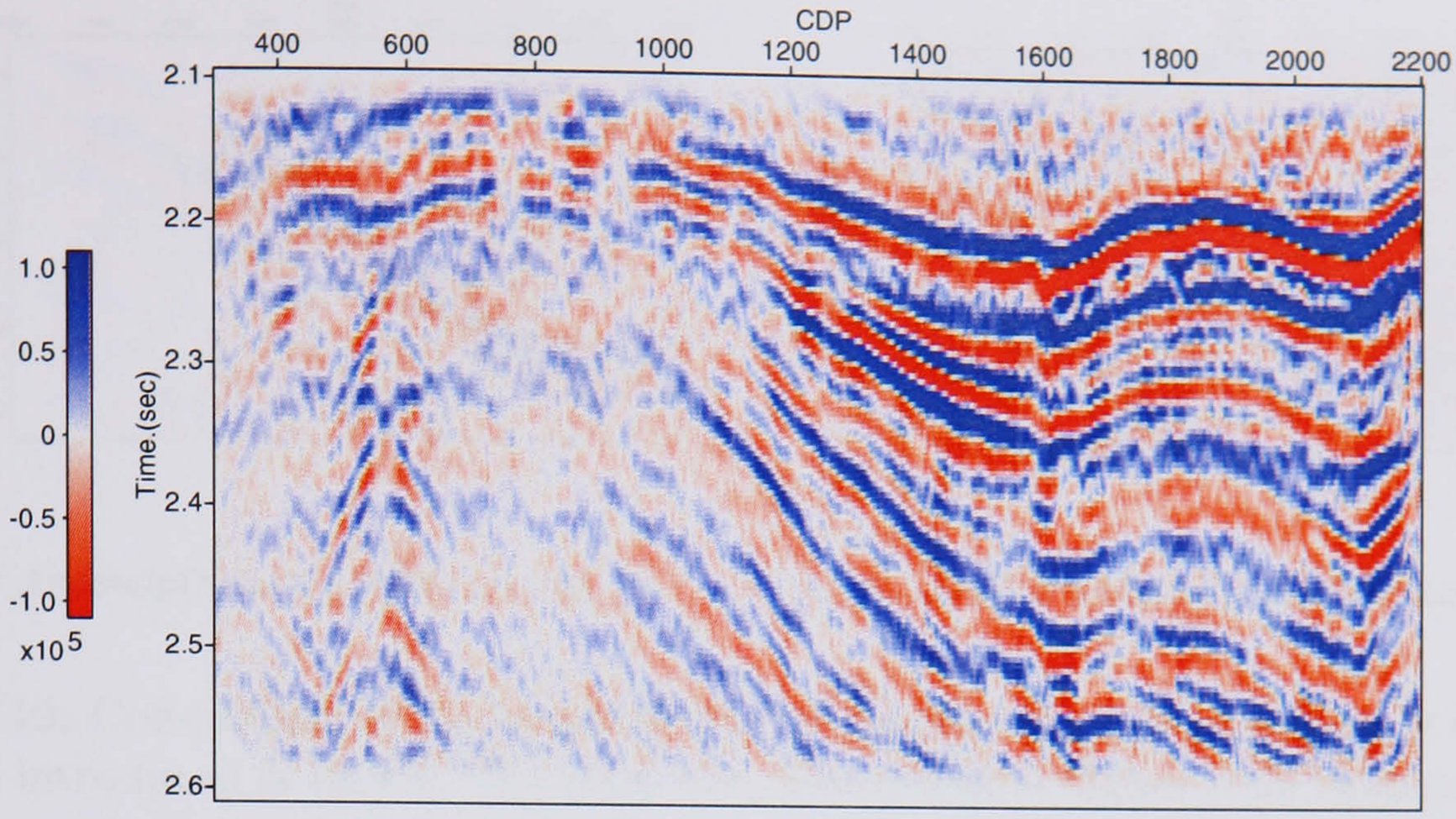


Figure 6.17: Seismic stack plot of the *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 14.5% anomaly (*NewBaseDataset*). This can be compared to the *NewBaseDataset* stack (Figure 6.3) plotted to the same scale. The plots are similar and thus the matching is very good. The matching also show that the *NewXEqMonitorDataset* stack is more similar and near identical to the *BaseDataset* stack (compare at CDP 1050–1450 and time 2280–2512 ms). Therefore, ANN preserved the anomaly introduced in the *NewBaseDataset* survey (Figure 6.15) by not replicating it on the *NewXEqMonitorDataset*.

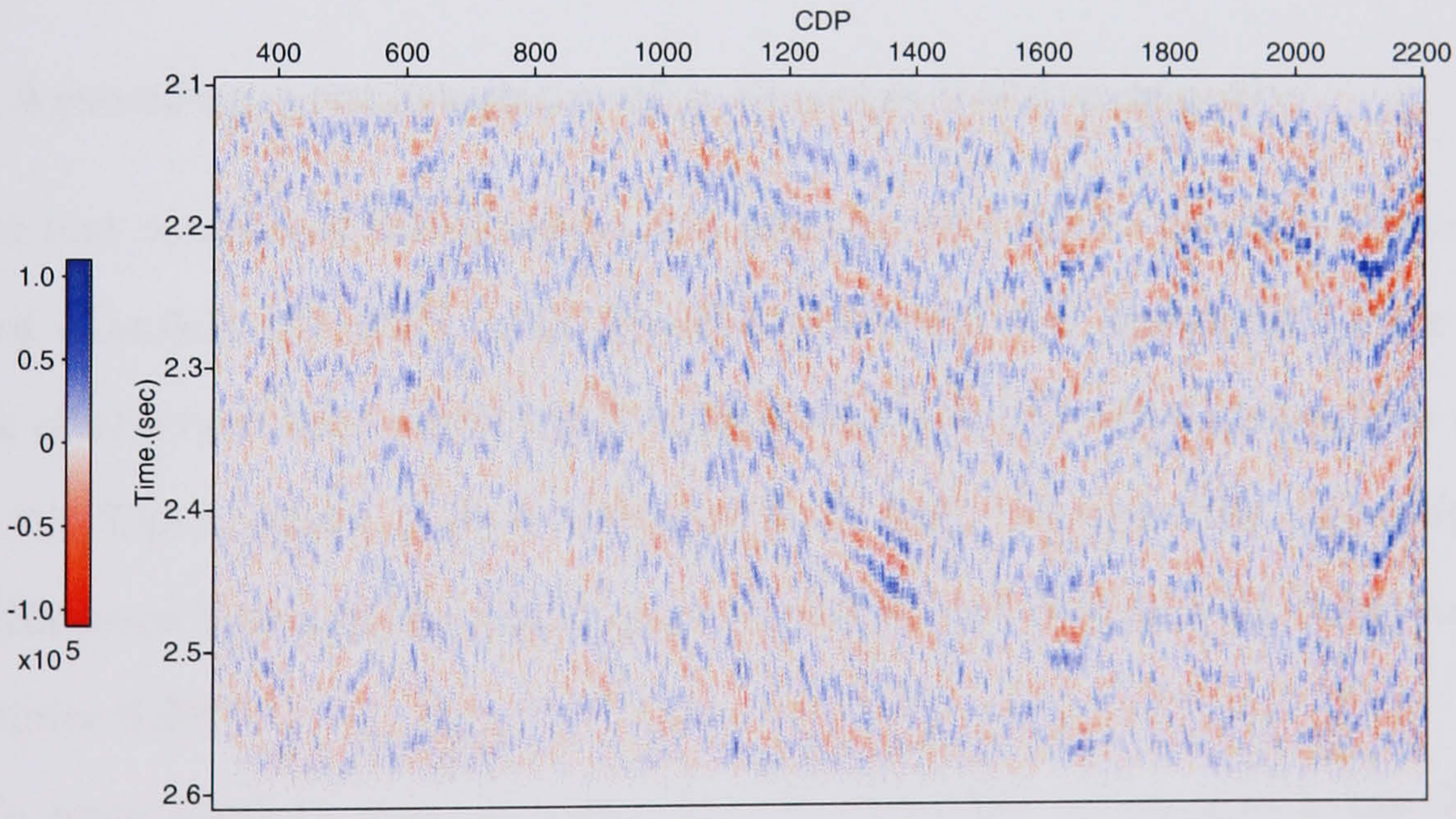
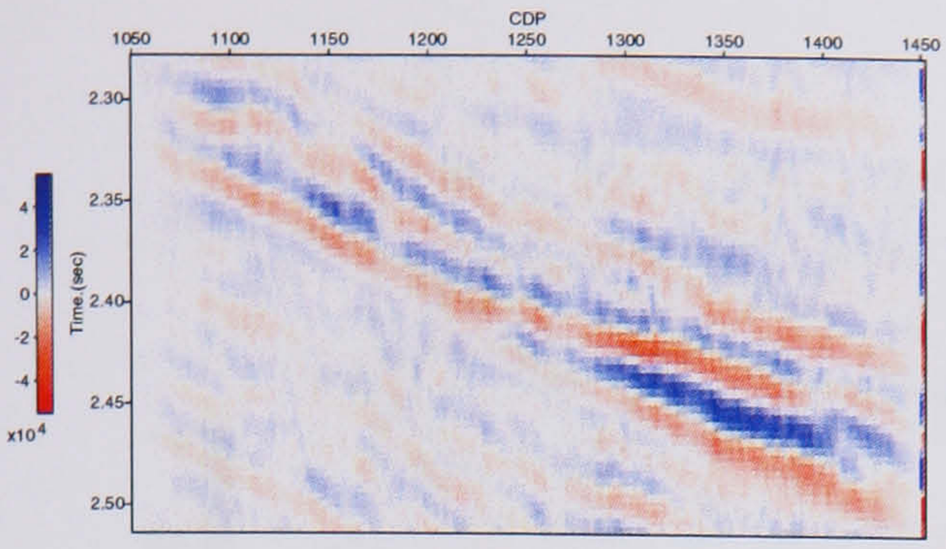
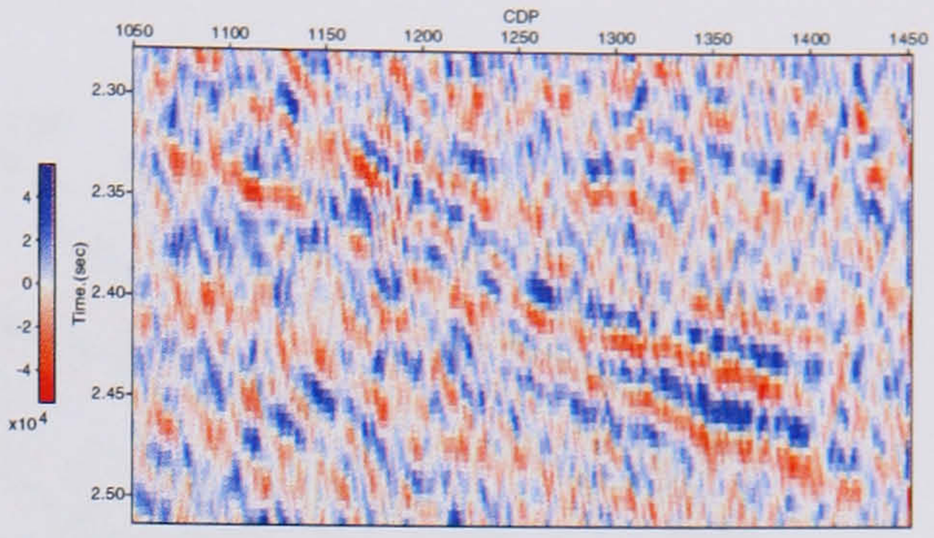


Figure 6.18: Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.15) and the *NewXEqMonitorDataset* stack (Figure 6.17). The cross-equalization achieved have preserved the introduced 14.5% anomaly, which is clear in this difference plot. Furthermore, since the general matching error is very low, the synthetic time-lapse anomaly is not masked by this error.



(a) *AnomalyDataset* (14.5% anomaly)



(b) *NewDifferenceDataset* (14.5% anomaly)

Figure 6.19: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 14.5%. The plots are plotted with a higher scale than previously used to enable better comparison. The plot gives assurance that the anomaly seen in Figure 6.18 is the anomaly introduced, since it is reasonably preserved in phase and magnitude. At the righthand-side area, where a strong anomaly is introduced, the phase and magnitude are very similar. However, in areas with lower introduced anomaly there is a clear obliteration and masking by matching error.

The anomaly seen between the *NewXEqMonitorDataset* stack (Figure 6.6) and the *NewBaseDataset* stack (Figure 6.4) was measured using Equation 6.1. The anomaly has a value of 14.1%. This is in very close agreement to the introduced anomaly (14.5%).

6.1.5 Assessing preservation of a desired 9.6% anomaly

The third test conducted introduced a low anomaly of 9.6% to the *BaseDataset* (offset gathers of channels 101–230). The *BaseDataset* stack was shown in Figure 6.3, and the stack of the new base survey with added anomaly (*NewBaseDataset*) is shown in Figure 6.20. The anomaly is clear in Figure 6.21, which displays the *AnomalyDataset*, i.e. the difference between the *BaseDataset* stack (Figure 6.3) and the *NewBaseDataset* stack (Figure 6.20).

ANNs were used to cross-equalize the *MonitorDataset* to match the *NewBaseDataset*. The configuration is identical to that reported in Section 5.4.1. Figure 6.22 shows the *NewXEqMonitorDataset* stack, while Figure 6.23 shows the difference section *NewDifferenceDataset*. The difference plot shows ANN ability to normalize acquisition differences while retaining the synthetic anomaly. However, it is clear that this low

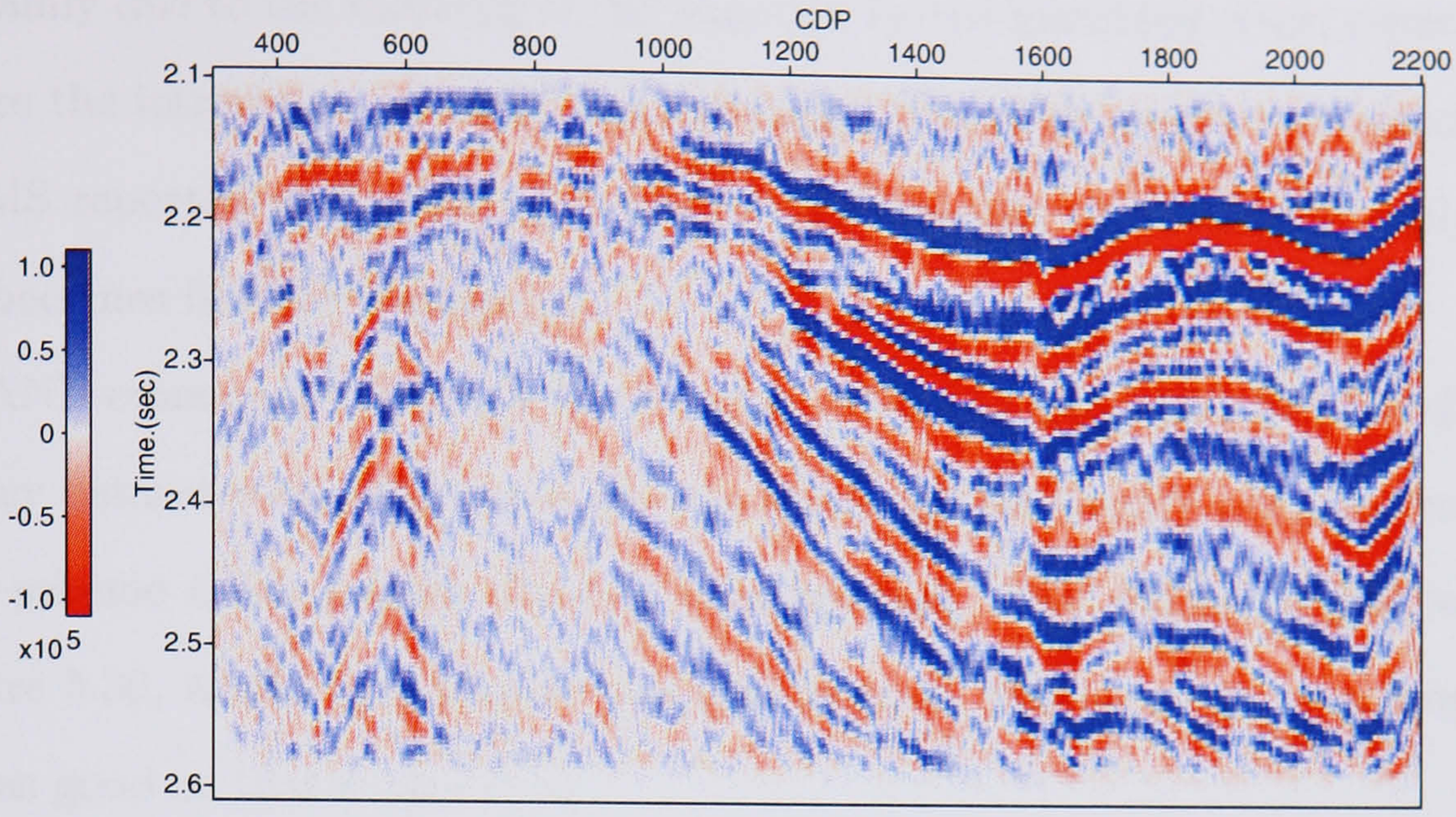


Figure 6.20: Seismic stack plot of all the offset gathers of channels 100–230 of the base survey with 9.6% anomaly (*NewBaseDataset*). The anomaly is introduced at the location defined in Figure 6.1. ANNs were used to produced a cross-equalized pre-stack dataset (*NewXEqMonitorDataset*) from the monitor survey (*MonitorDataset*) to match those used to produce this stack.

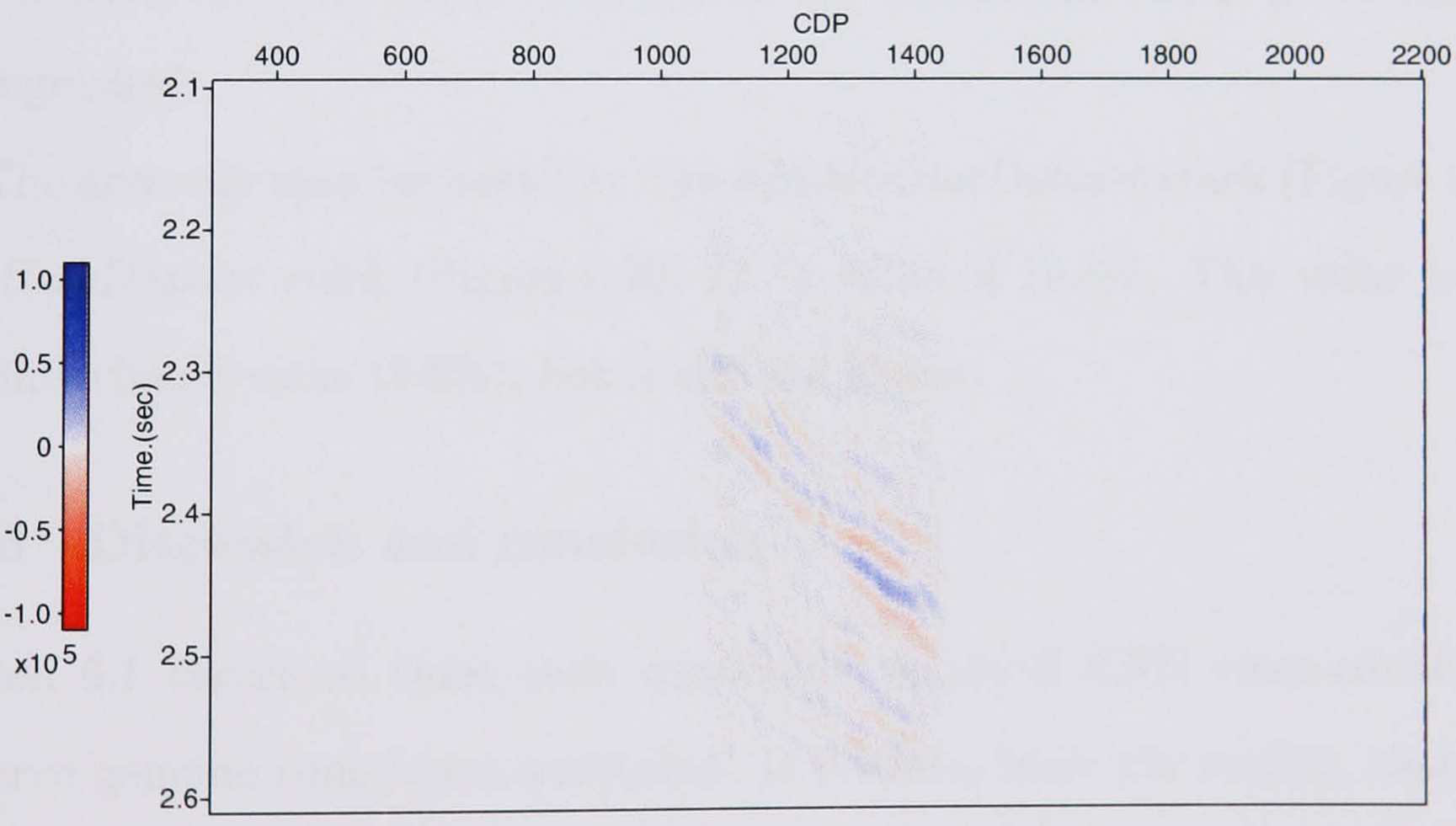


Figure 6.21: Difference plot, between the *BaseDataset* stack (Figure 6.3) and the *New-BaseDataset* stack (Figure 6.20), showing the introduced 9.6% anomaly.

synthetic anomaly of 9.6% is just about distinguishable from the matching error. This is mainly due to the masking of the anomaly by the matching error, especially in areas where the introduced anomaly has low magnitude. This might be suspected since the RRMS reported in Section 5.4.3 is 4%, and thus any time-lapse anomaly lower than 4% becomes hard to distinguish.

ANN cross-equalization of the *MonitorDataset* to match the *NewBaseDataset* (base survey dataset with 9.6% anomaly) was conducted previously when the cross-equalization–scheme configuration was less than optimal. The difference plot was shown in Figure 5.30, where the anomaly can also be seen. However, the matching ability was not as good as that shown here.

Figure 6.24 shows a zoom at the anomaly location and compares the *NewDifferenceDataset* to the *AnomalyDataset*, and thus allows a better assessment of the preservation of the anomaly or otherwise. This figure gives assurance that the anomaly seen in Figure 6.23 is the anomaly introduced, since it is sufficiently preserved in phase and magnitude. However, contrary to the previously introduced anomalies, this anomaly (being very small) is close to being fully masked by the matching error. Nevertheless, it seems that an anomaly as low as this introduced one is preserved and can be distinguished.

The anomaly seen between the *NewXEqMonitorDataset* stack (Figure 6.22) and the *NewBaseDataset* stack (Figure 6.20) has a value of 10.8%. This value is close to the real introduced value (9.6%), but is slightly higher.

6.1.6 Discussion and conclusion

Section 6.1 described three tests conducted to see if ANN cross-equalization could preserve genuine time-lapse anomalies. It is clear, from the results, that ANN cross-equalization has the ability to preserve genuine time-lapse anomalies. Furthermore, we could quantitatively say that an anomaly as low as approximately 10%, is preserved and can be distinguished from other matching error on this dataset. This is about a

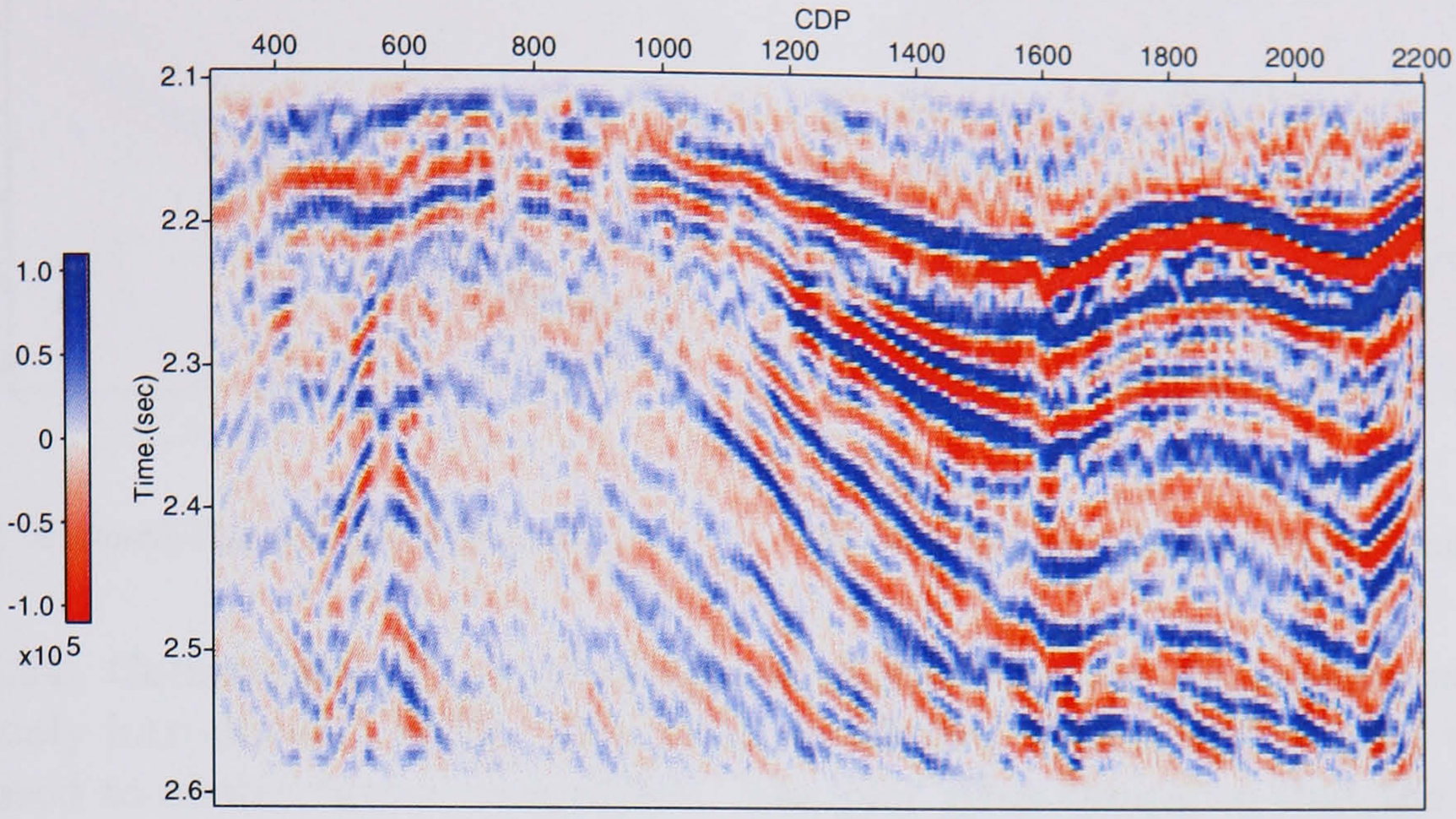


Figure 6.22: Seismic stack plot of the *NewXEqMonitorDataset*, i.e. the cross-equalized monitor survey when matched to the base survey with 9.6% anomaly. This can be compared to the *NewBaseDataset* stack (Figure 6.20) and the *BaseDataset* stack (Figure 6.3) plotted to the same scale. The plots are similar and thus the matching is very good.

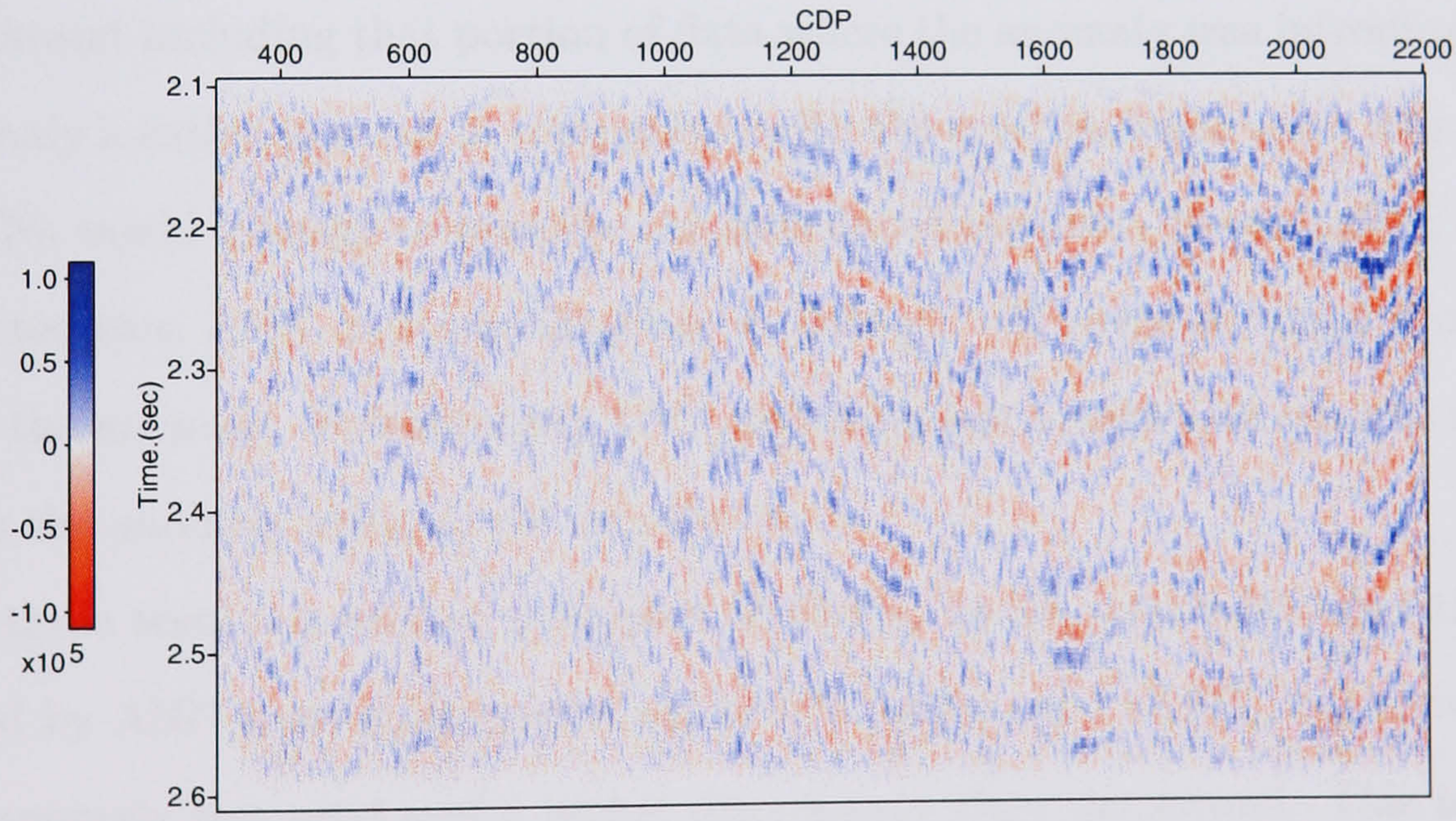
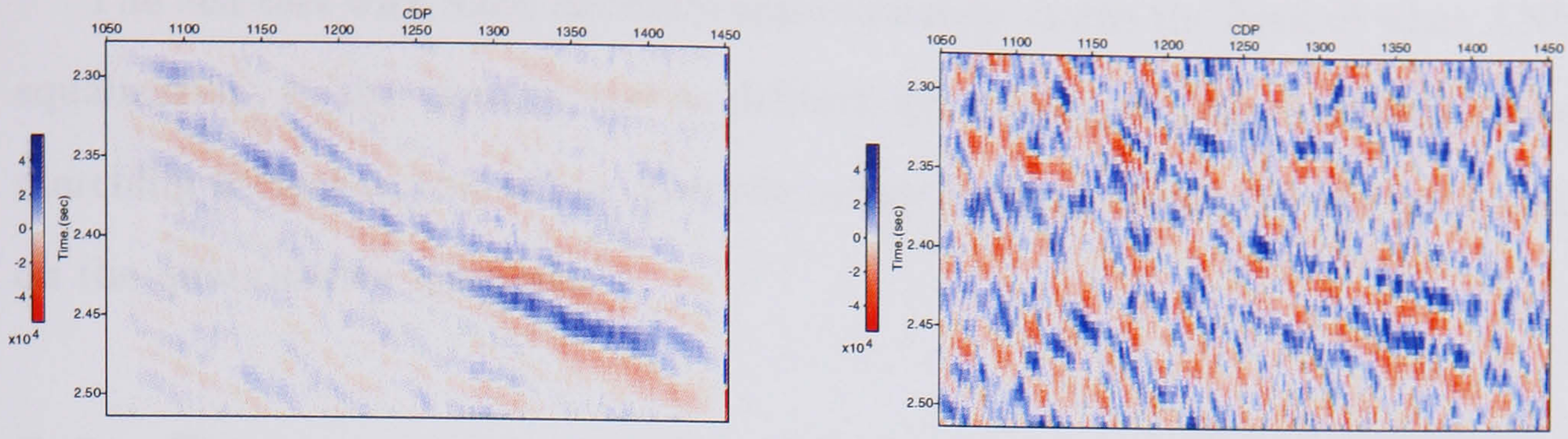


Figure 6.23: Plot of the *NewDifferenceDataset*, i.e. the difference between the *NewBaseDataset* stack (Figure 6.20) and the *NewXEqMonitorDataset* stack (Figure 6.22). This shows a very low error in matching. Furthermore, the cross-equalization achieved have preserved the introduced 9.6% anomaly, which is clear in this difference plot, although nearly masked by matching error.



(a) *AnomalyDataset* (9.6% anomaly) (b) *NewDifferenceDataset* (9.6% anomaly)

Figure 6.24: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when the anomaly introduced is 9.6%. The plots are plotted with a higher scale than previously used to enable better comparison. The plot gives assurance that the anomaly seen in Figure 6.23 is the anomaly introduced, since it is sufficiently preserved in phase and magnitude. At the righthand-side area, where a strong anomaly is introduced, the introduced anomaly is clear with the correct phase and slightly obliterated magnitude. However, in areas with lower introduced anomaly the matching error masks the introduced anomaly.

factor of two better than linear methods can achieve.

It was explained that the networks are actually asked to match all the new base survey dataset including that portion of data where the anomaly was introduced. Since the anomaly location is small in comparison with the total matched area, it is expected that ANNs would manage to preserve the introduced anomaly. Nevertheless, it should be expected that ANN cross-equalization, as defined, will cause an amount of obliteration to the anomaly. Furthermore, ANN matching has a degree of error in matching and thus the anomaly value might be affected.

The three tests introduced anomalies of 23.9%, 14.5%, and 9.6%, and the values preserved by ANN cross-equalization are 21.0%, 14.1%, and 10.8%, respectively. The highest anomaly test produced a higher discrepancy than the second. This is because the higher anomaly makes the region where it is introduced more crucial for ANN to match during training since it causes a higher error cost function. Therefore, ANNs will be more prone to causing an obliteration to a larger time-lapse anomaly than to a smaller one.

The last test with 9.6% anomaly approximately marks the limit of what ANN cross-equalization, to this dataset and as defined, can preserve. This is because the error in matching is close to this value. However, more tests are needed to put more confidence on the quantitative measures.

6.2 Post-stack cross-equalization

There are other possible alternative ways to performing cross-equalization using ANNs than the one chosen in this research. This thesis described the motivation for the choices made. However, I did investigate few other options. In particular, I investigated performing time-lapse cross-equalization post-stack or even on several offset stacks, where a limited range of offsets are stacked.

Applying cross-equalization post-stack has the advantage of being faster and working with higher signal-to-noise ratio data. The disadvantage is that pre-stack time-lapse analysis (such as AVO) becomes not possible. In addition, less data will be available to train ANNs, and simple time-lapse misfits might become more complicated to solve. Nevertheless, one might desire to perform post-stack cross-equalization. Therefore, this section will shows results of experiments done to assess ANN cross-equalization performance if it is done post-stack. It will also illustrate some findings that were also seen in pre-stack cross-equalization, but which are easier to illustrate in post-stack cross-equalization.

In this section, the full in-line corresponding to the second streamer of the base survey is stacked. In addition, the three in-lines corresponding to the three streamers of the monitor survey are stacked. During post-stack cross-equalization, a close match to the base survey stack at time section 2100–2612 ms is to be produced from the monitor survey's stacks. This section will analyse post-stack cross-equalization and compare it to pre-stack cross-equalization. Furthermore, it will compare linear versus ANN post-stack cross-equalization, both in terms of matching and desired anomaly preservation.

6.2.1 Matching post-stack using ANNs

Configuration

ANN post-stack cross-equalization was configured in a similar way to cross-equalization of one particular offset gather. Thus, training to match a specific sample at time t and CDP x, requires the sample at time t from a trace r_b at CDP x from the base survey stack as a target output. In addition, the input consists of the CDP and time position as geometrical data, and seismic input from the monitor survey stacks as defined in Figure 6.25.

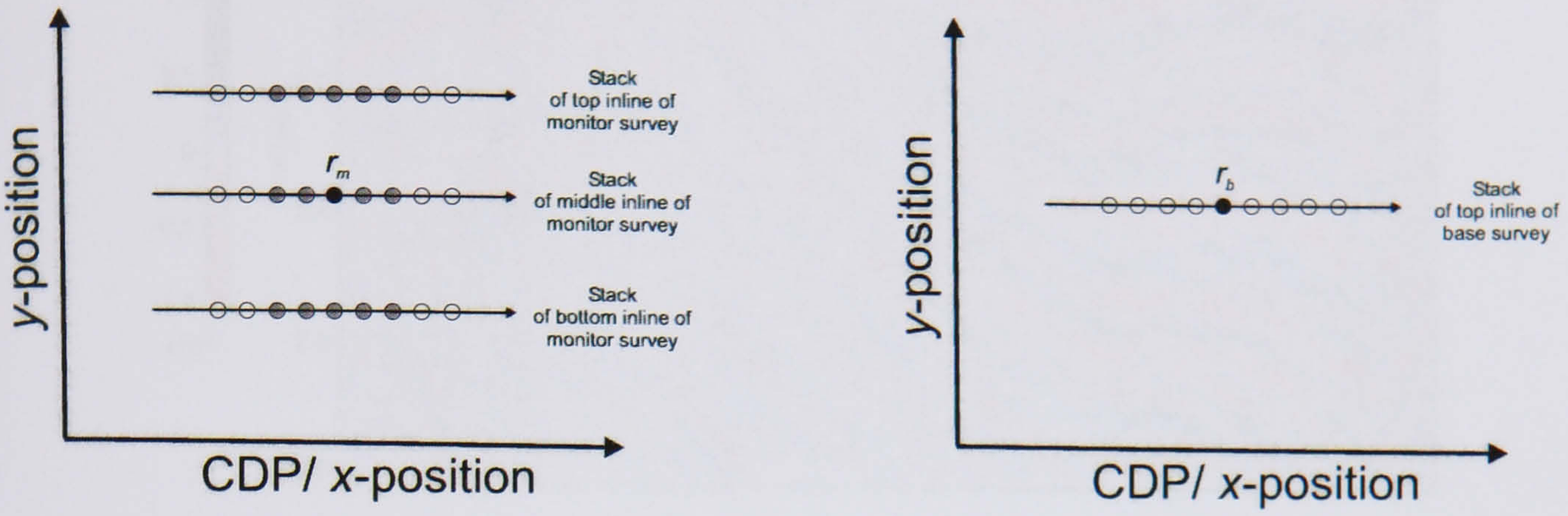
The other configurations — such as normalization, and training — are identical to that in ANN pre-stack cross-equalization shown in Section 5.4.1. One ANN is used with the same topology used in Section 5.4.1 (Figure 5.31), but the input layer has 85 nodes instead of 89 since only two geometrical parameters are used here.

Results

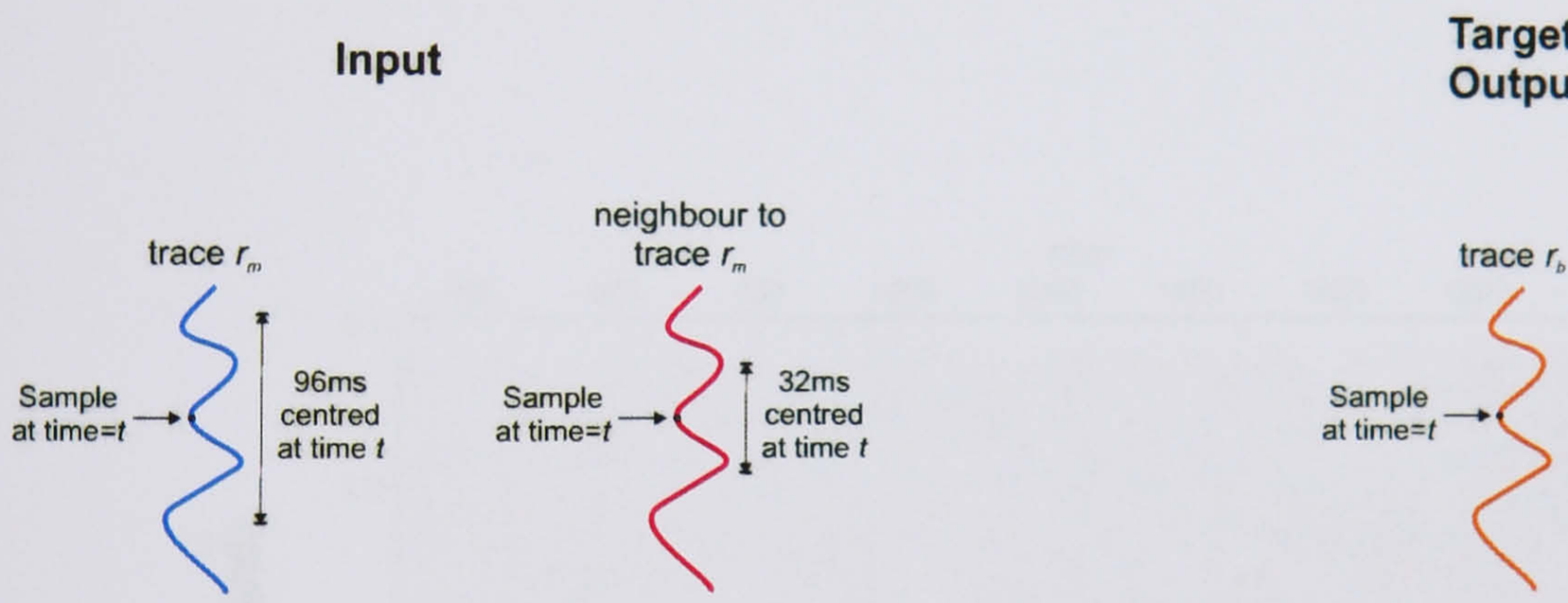
Figure 6.26 shows the stack plot of the base survey. While, Figure 6.27 shows the difference plot between Figure 6.26 and the stack of the ANN post-stack cross-equalized monitor survey. The difference plot shows a higher error in matching than that seen in pre-stack matching (*cf.* Figure 6.28).

The ANN post-stack cross-equalization was initially performed using only one in-line of the monitor survey to provide input for the ANNs. Basically, only the middle in-line of the monitor survey is used, and thus only trace r_m and four neighbouring traces are used from Figure 6.25. This is more similar to the case when performing linear post-stack cross-equalization, where no neighbouring traces are incorporated. The difference plot between the base survey stack (Figure 6.26) and the ANN cross-equalized monitor using this configuration is shown in Figure 6.29. It is evident that while using the three in-lines (Figure 6.27) a better result was achieved.

The result seen, when comparing results of including or not including neighbouring information from neighbouring in-lines, was previously seen when matching offset



(a) Traces involved: each trace is represented by a circle, where time is perpendicular to the page



(b) Samples involved

Figure 6.25: The seismic input-output to the network during training to perform post-stack cross-equalization. Given that a certain sample at time t and CDP x is to be matched, (a) shows the traces involved in training the network, and (b) shows the exact seismic input and target output. Thus, the target output is the sample at time t from a trace r_b at CDP x from the base survey, and the input is from trace r_m from the monitor survey (nearest to r_b in CDP and y-position) and from 14 neighbouring traces to r_m. The neighbouring traces are those shaded in (a). The exact input is a window of 96 ms (13 samples) from trace r_m and windows of 32 ms (5 samples) from the 14 neighbouring traces, where all these windows are centred at time t.

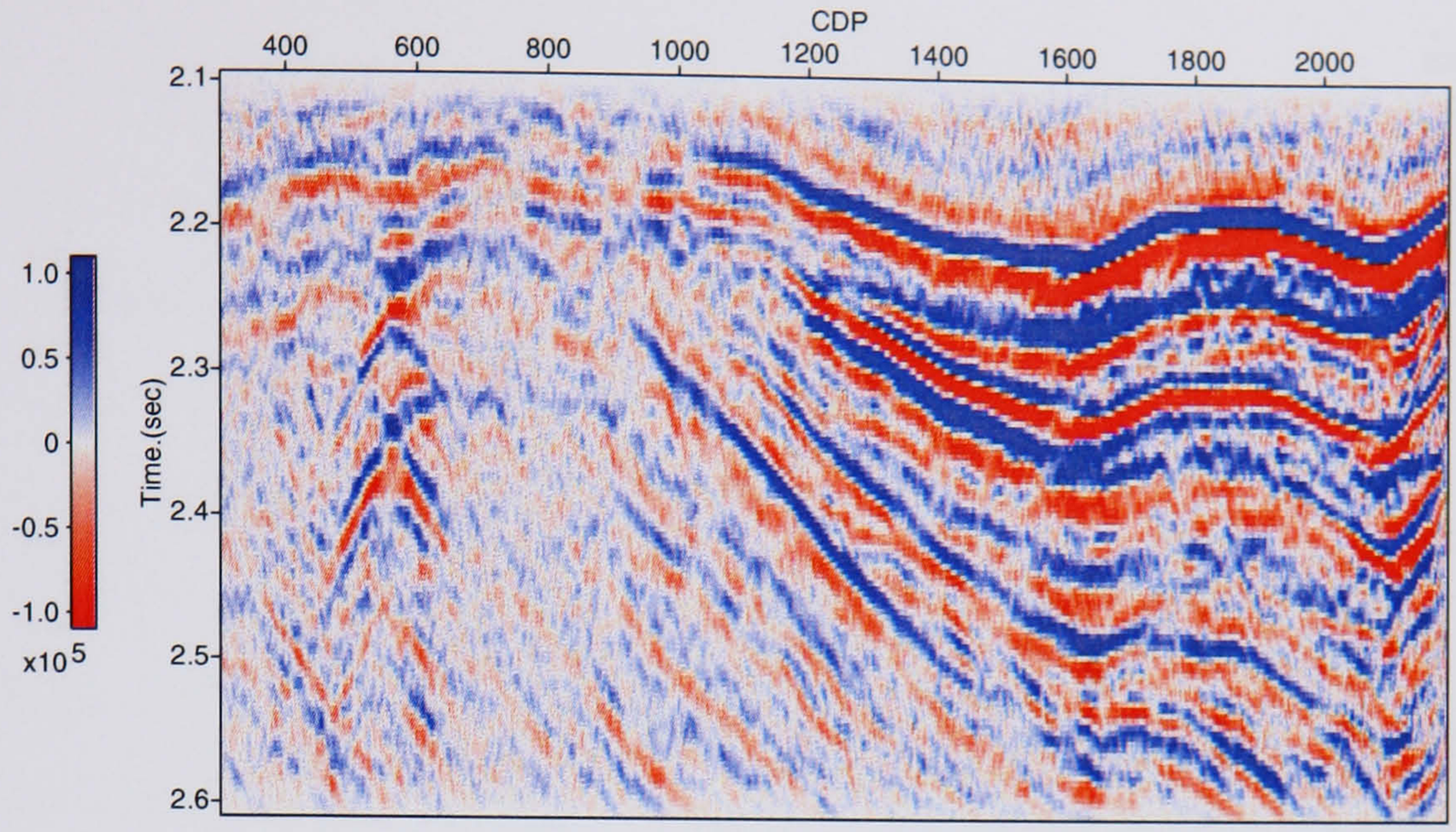


Figure 6.26: Seismic stack plot of all the offset gathers of channels 1–230 of the base survey.

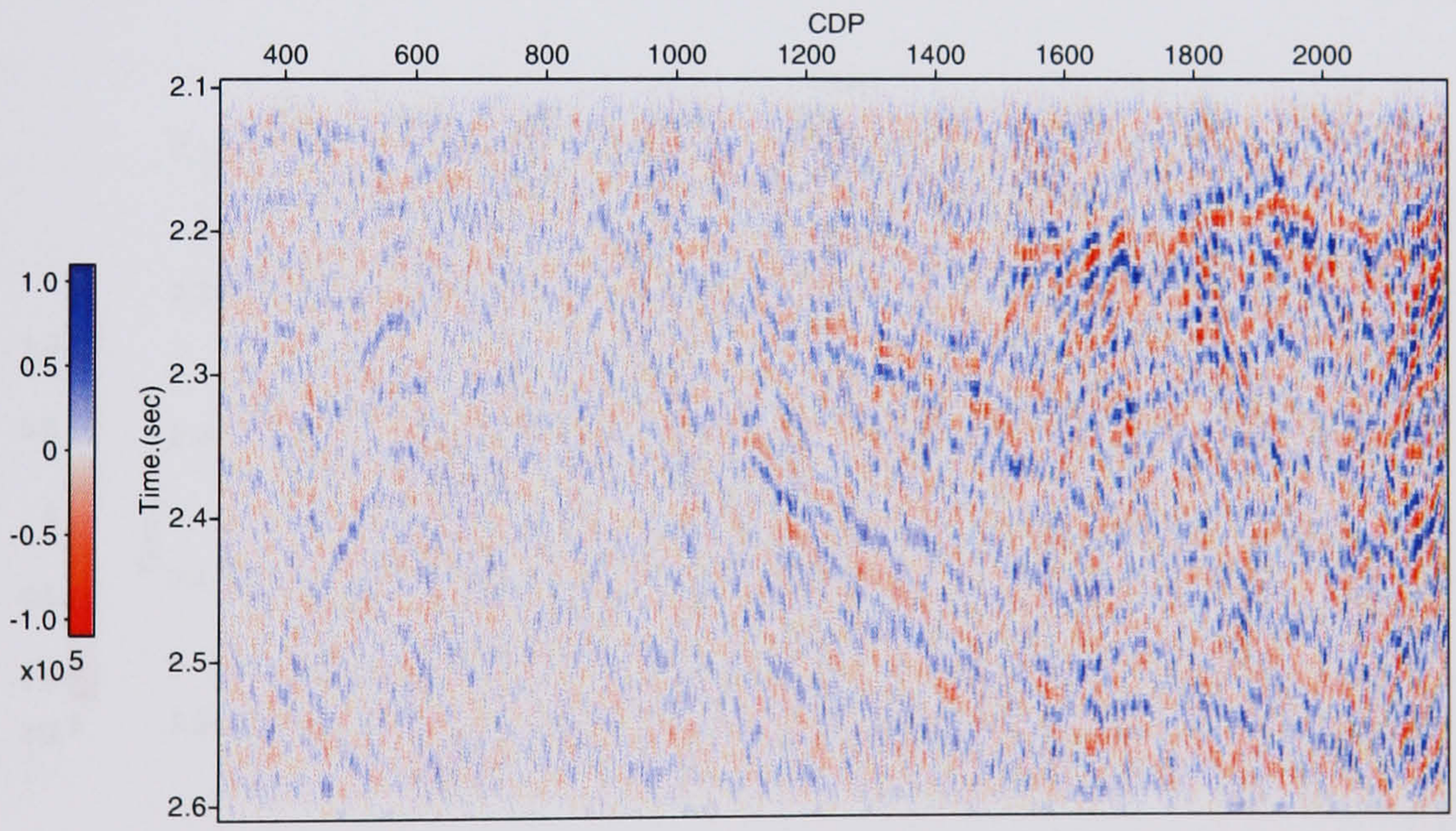


Figure 6.27: Difference between the base survey stack (Figure 6.26) and the stack of the ANN post-stack cross-equalized monitor survey. This shows a higher error in matching than that seen in (*cf.* Figure 6.28). However, ANN post-stack post-stack cross-equalization is still superior over the its linear counterpart (*cf.* the difference linear post-stack cross-equalization, Figure 6.32).

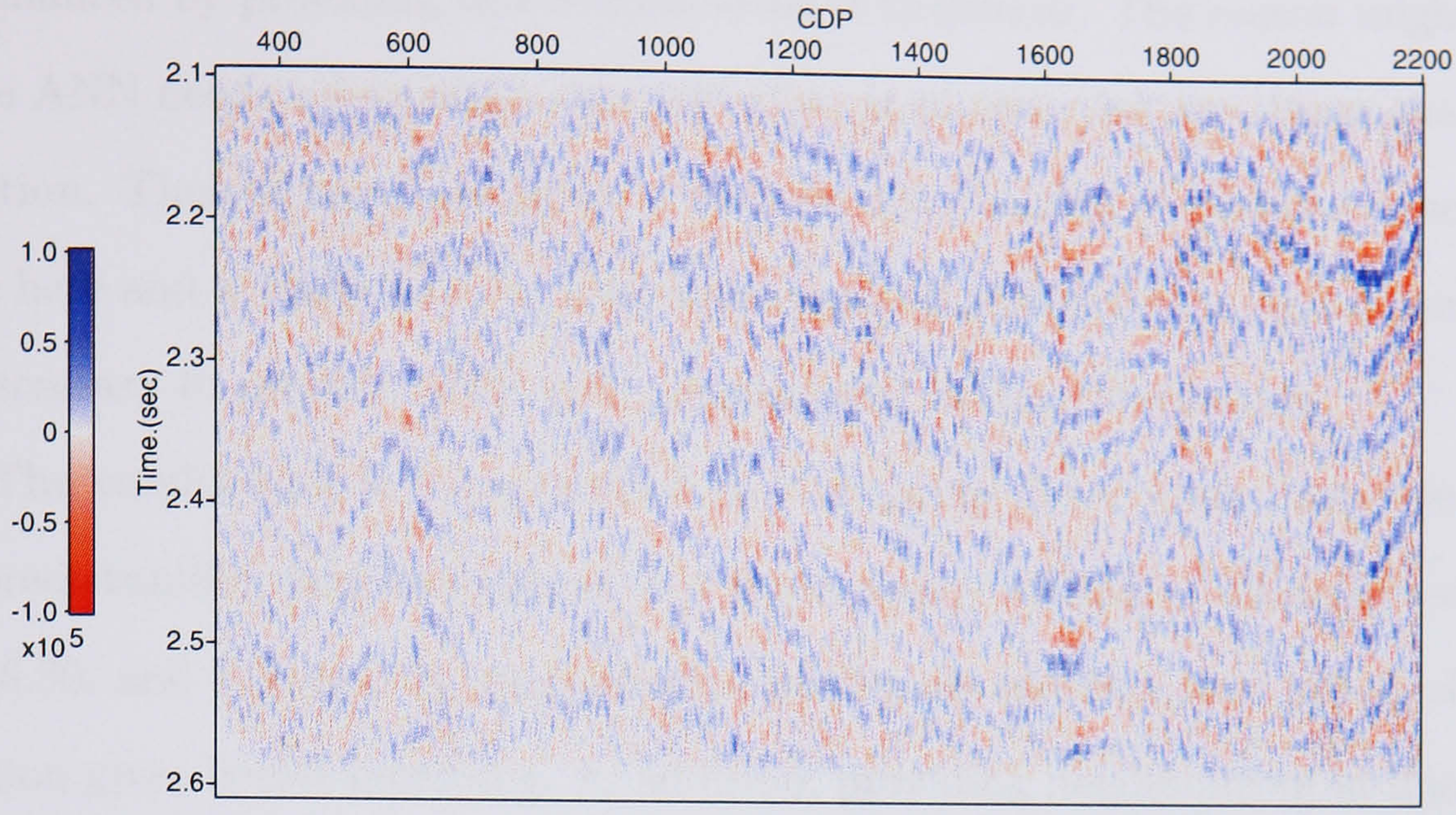


Figure 6.28: Figure 5.35 re-plotted to the same scale as Figure 6.27 for comparison. This difference plot of ANN pre-stack cross-equalization shows superior matching to that shown on the post-stack counterpart (Figure 6.27).

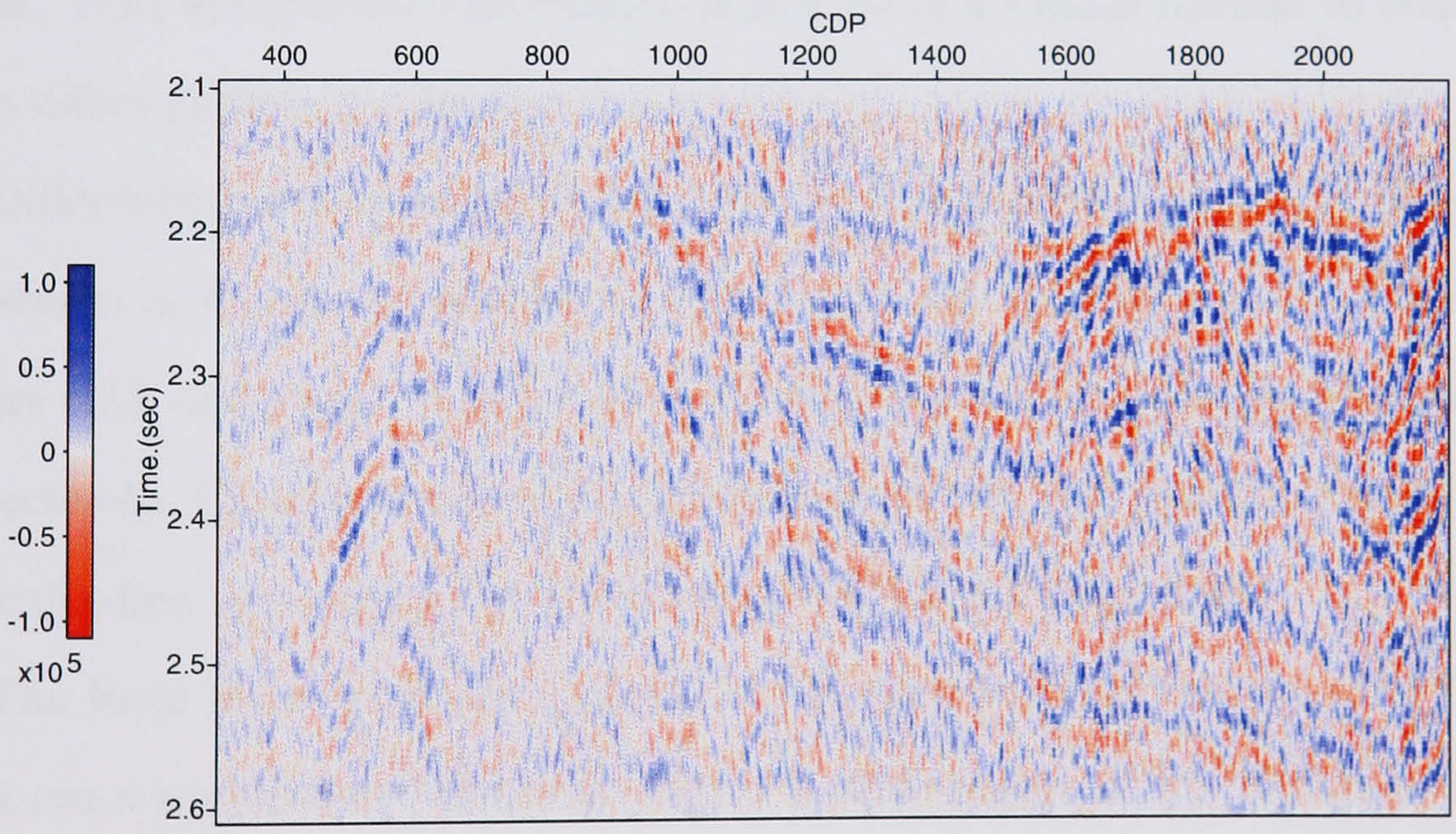


Figure 6.29: Difference between the base survey stack (Figure 6.26) and the stack of the ANN post-stack cross-equalized monitor cross-equalized monitor survey when one in-line of the monitor was used as input to the result was achieved while using the three in-lines of the monitor survey as input (*cf.* Figure 6.27).

gathers pre-stack. The result is similar, and indicates that ANN matching capability is enhanced by providing this neighbouring information. The reason might be obvious, since ANN needs information that will allow it to cater for time-lapse changes in trace position. Thus, if two time-lapse in-lines do not coincide in position exactly, as is the case here and is likely in most time-lapse analysis that relies on legacy surveys, then it is necessary to provide ANN with neighbouring in-line information.

The results seen are emphasized by comparing other quantitative measures such as predictability, and normalized root mean square (NRMS). These are shown in Figure 6.30, and Figure 6.31, respectively. Both plots indicate that pre-stack cross-equalization gives better matching. In addition, providing neighbouring in-line information enhances matching.

6.2.2 Linear post-stack cross-equalization

Post-stack cross-equalization was also performed using the linear method discussed in Section 5.4.2 with three different modes: local, global, and semi-global that involves 51 traces. Post-stack cross-equalization is treated in a similar fashion to cross-equalization of an offset gather, see Section 5.4.2.

Difference plots were produced between the base survey stack (Figure 6.26) and the stacks of the linear post-stack cross-equalized monitor survey. Thus, Figure 6.32, Figure 6.33, and Figure 6.34 are the difference plots of the local, semi-global, and global, respectively. Clearly, the local linear mode gives the best matching results. This is also evident when comparing the NRMS and RRMS in Figure 6.35.

The local linear post-stack cross-equalization is slightly inferior to the ANN post-stack cross-equalization (compare Figure 6.32 to Figure 6.27). However, the magnitude of the difference between the two is far lower than that seen earlier during pre-stack comparison (Section 5.4.1). It is also interesting to see that the local linear post-stack cross-equalization shows similar or better result than its pre-stack counterpart (compare Figure 6.32 to Figure 6.36). Both these two findings can be confirmed or otherwise

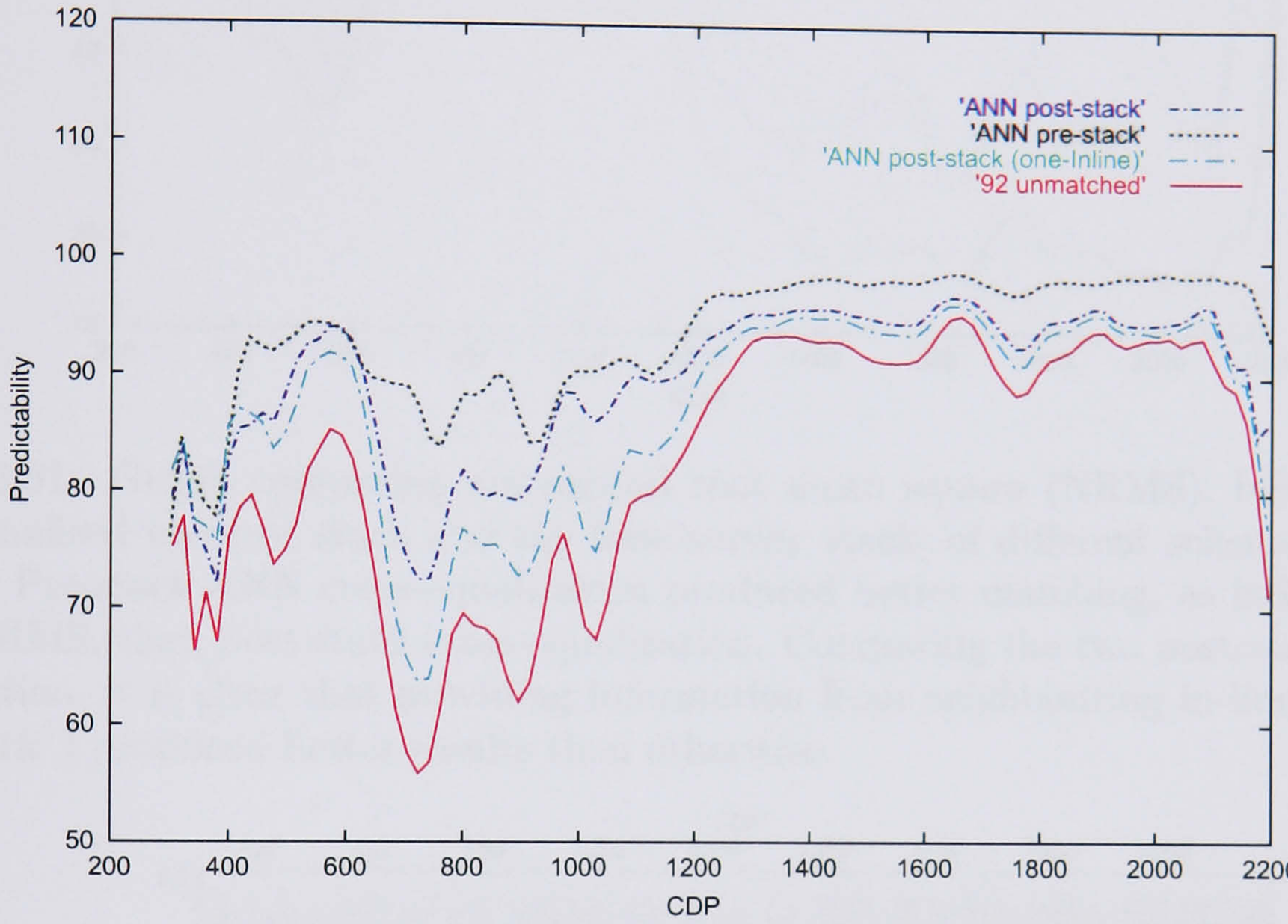


Figure 6.30: Graph comparing predictability, between the cross-equalized monitor stack and the base survey stack, of different scheme configurations. The graph shows that predictability has increased after ANN cross-equalization, when compared to the predictability before matching "92 unmatched". This is contrary to linear cross-equalization, which does not alter predictability. Pre-stack ANN cross-equalization produced better matching, as indicated by higher predictability, than post-stack cross-equalization. Comparing the two post-stack cross-equalization, it is clear that providing information from neighbouring in-lines ("ANN post-stack") produced better results than otherwise.

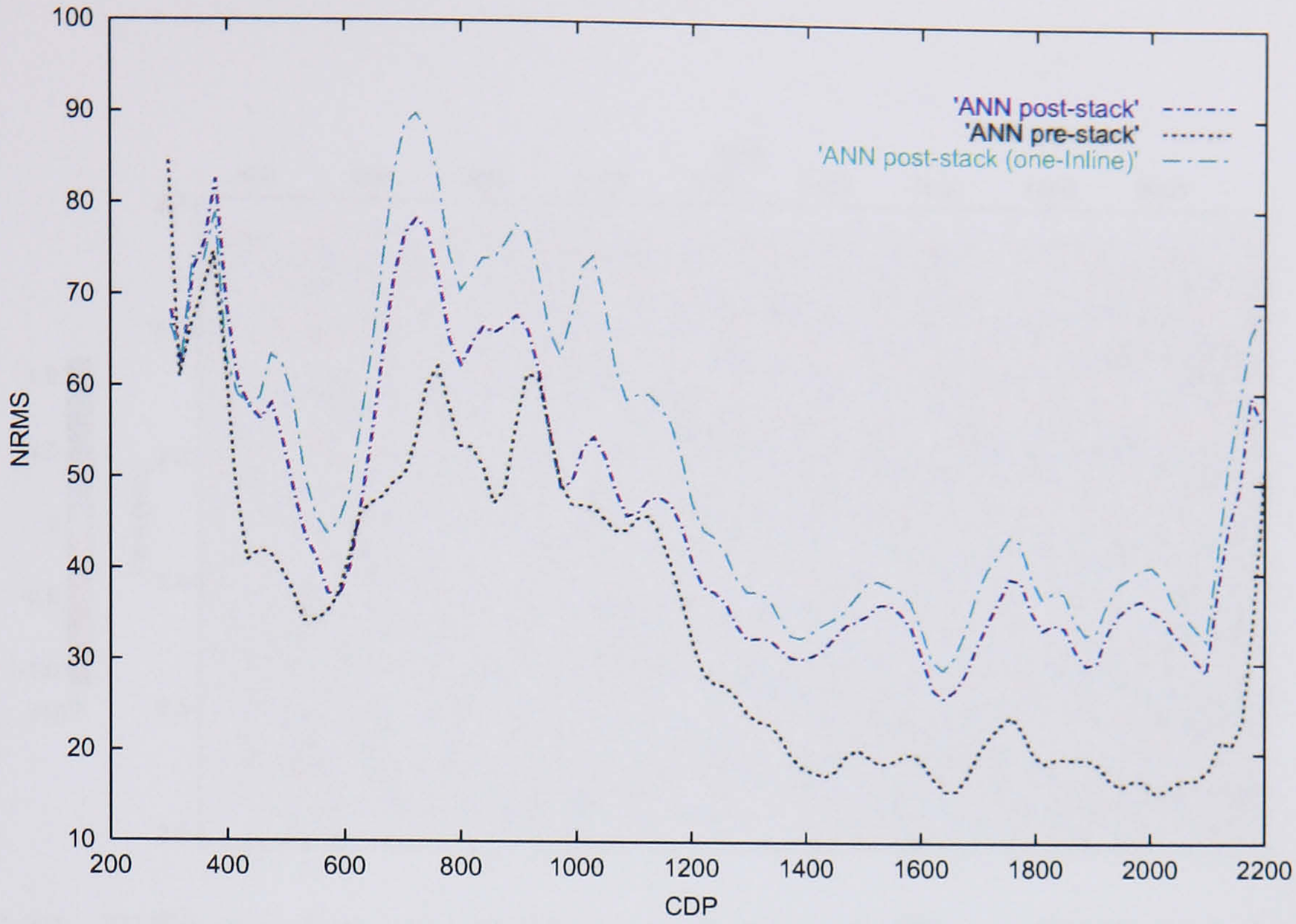


Figure 6.31: Graph comparing normalized root mean square (NRMS), between the cross-equalized monitor stack and the base survey stack, of different scheme configurations. Pre-stack ANN cross-equalization produced better matching, as indicated by lower NRMS, than post-stack cross-equalization. Comparing the two post-stack cross-equalization, it is clear that providing information from neighbouring in-lines ("ANN post-stack") produced better results than otherwise.

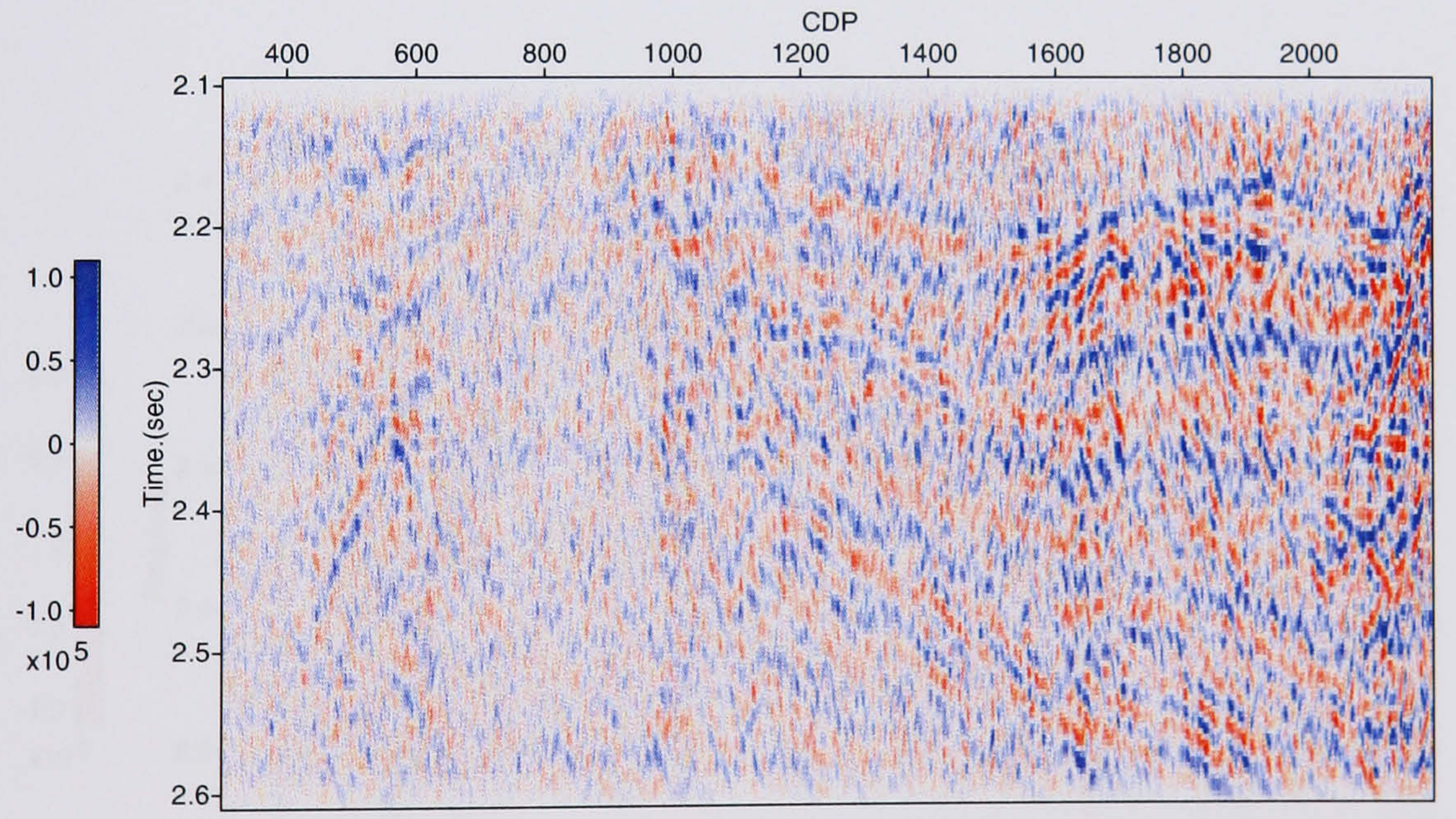


Figure 6.32: Difference between the base survey stack (Figure 6.26) and the stack of the local linear post-stack cross-equalized monitor survey. This shows better matching than the semi-global and global modes shown in Figure 6.33, and Figure 6.34, respectively. Comparing this to the difference plot of the local pre-stack cross-equalization (replotted in Figure 6.36), we find that the linear post-stack cross-equalization produces similar or better result than the pre-stack. However, it seems that ANN post-stack cross-equalization achieved a slightly better matching result (*cf.* Figure 6.27).

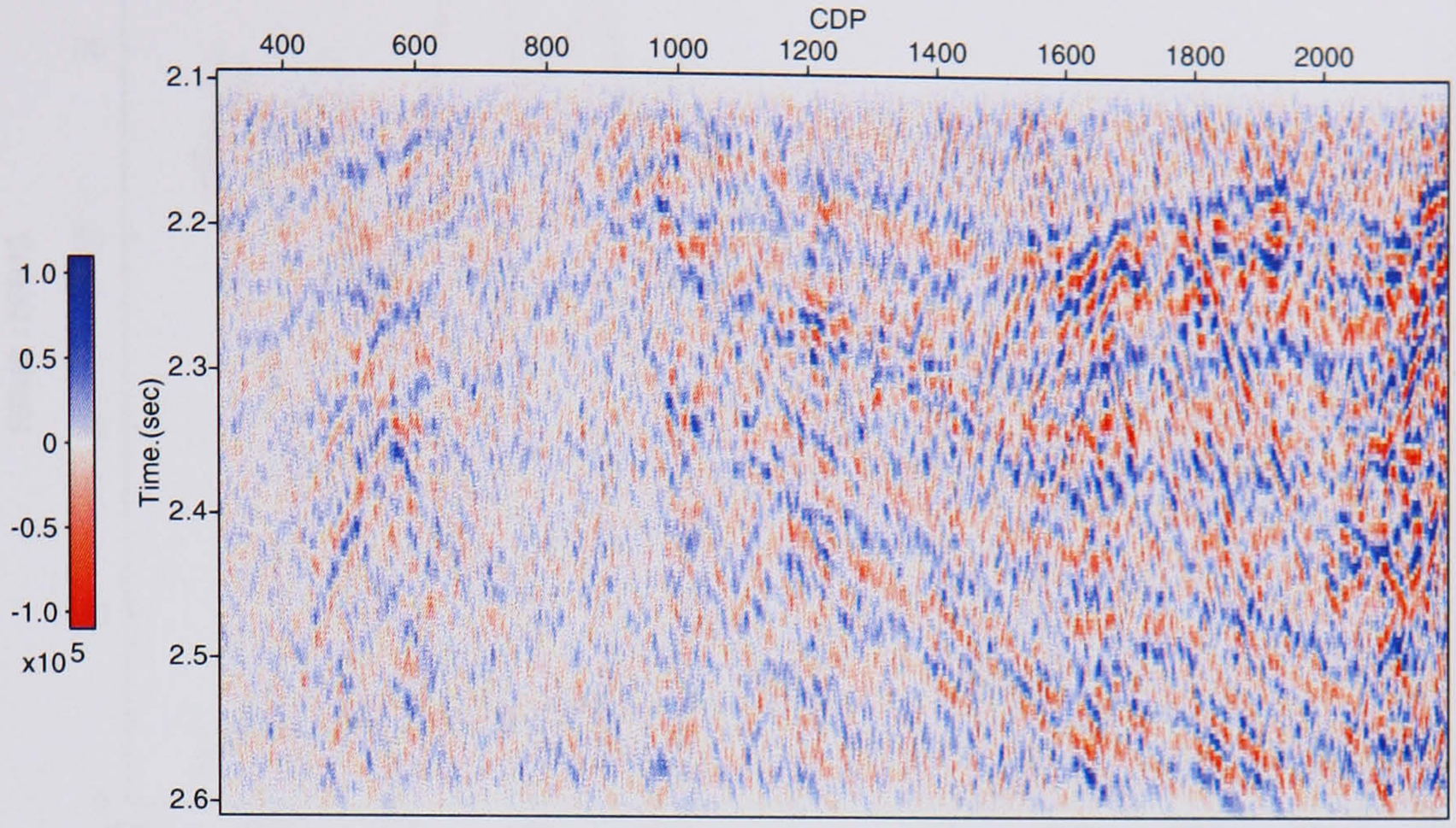


Figure 6.33: Difference between the base survey stack (Figure 6.26) and the stack of (Figure 6.26) and the stack of the semi-global (51 traces) linear post-stack cross-equalized monitor survey. This shows a slightly worse matching than that shown by the local mode (Figure 6.32).

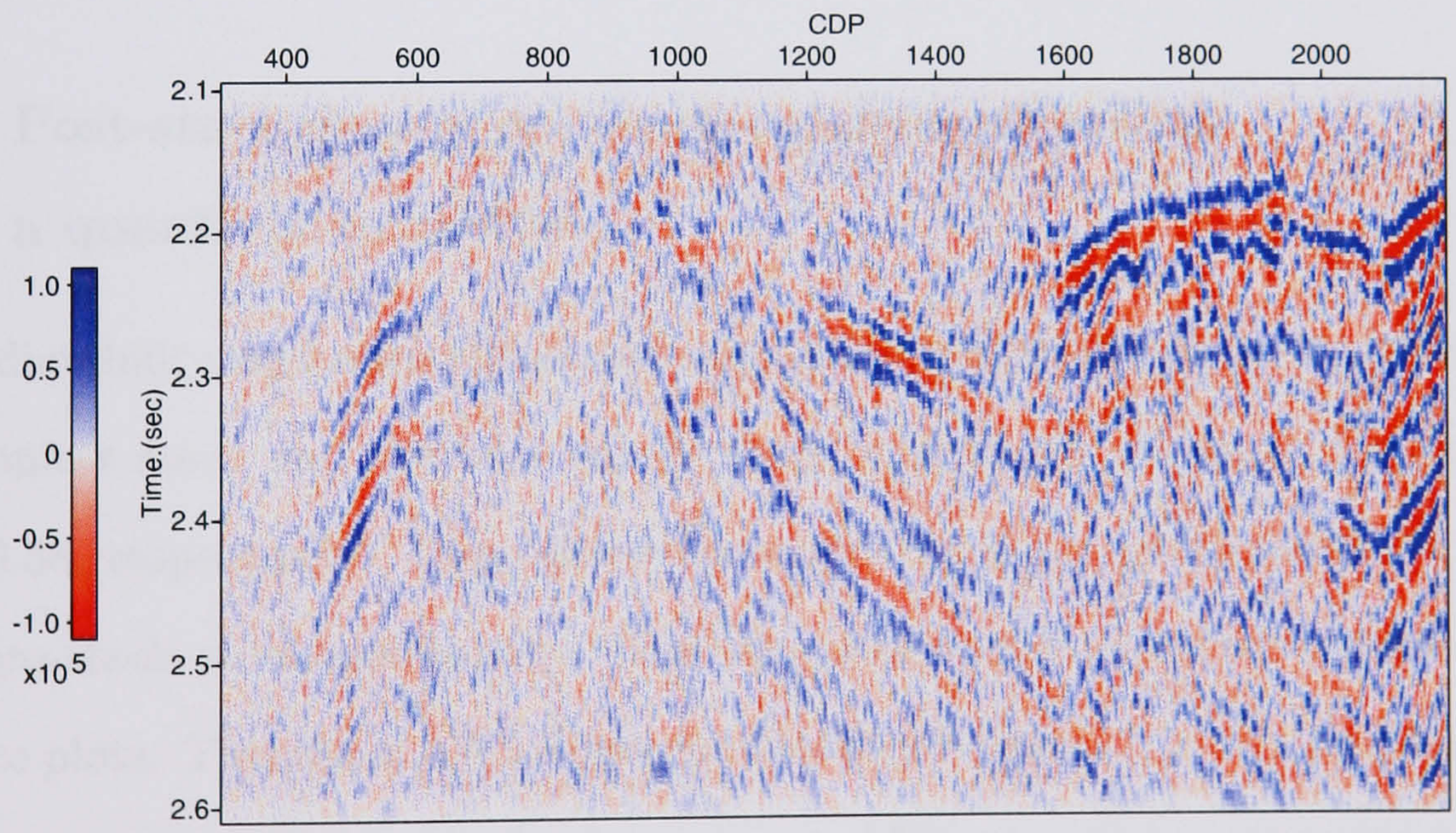


Figure 6.34: Difference between the base survey stack (Figure 6.26) and the stack of the global linear global linear post-stack cross-equalized monitor survey. This shows worse matching than that shown by the local mode (Figure 6.32).

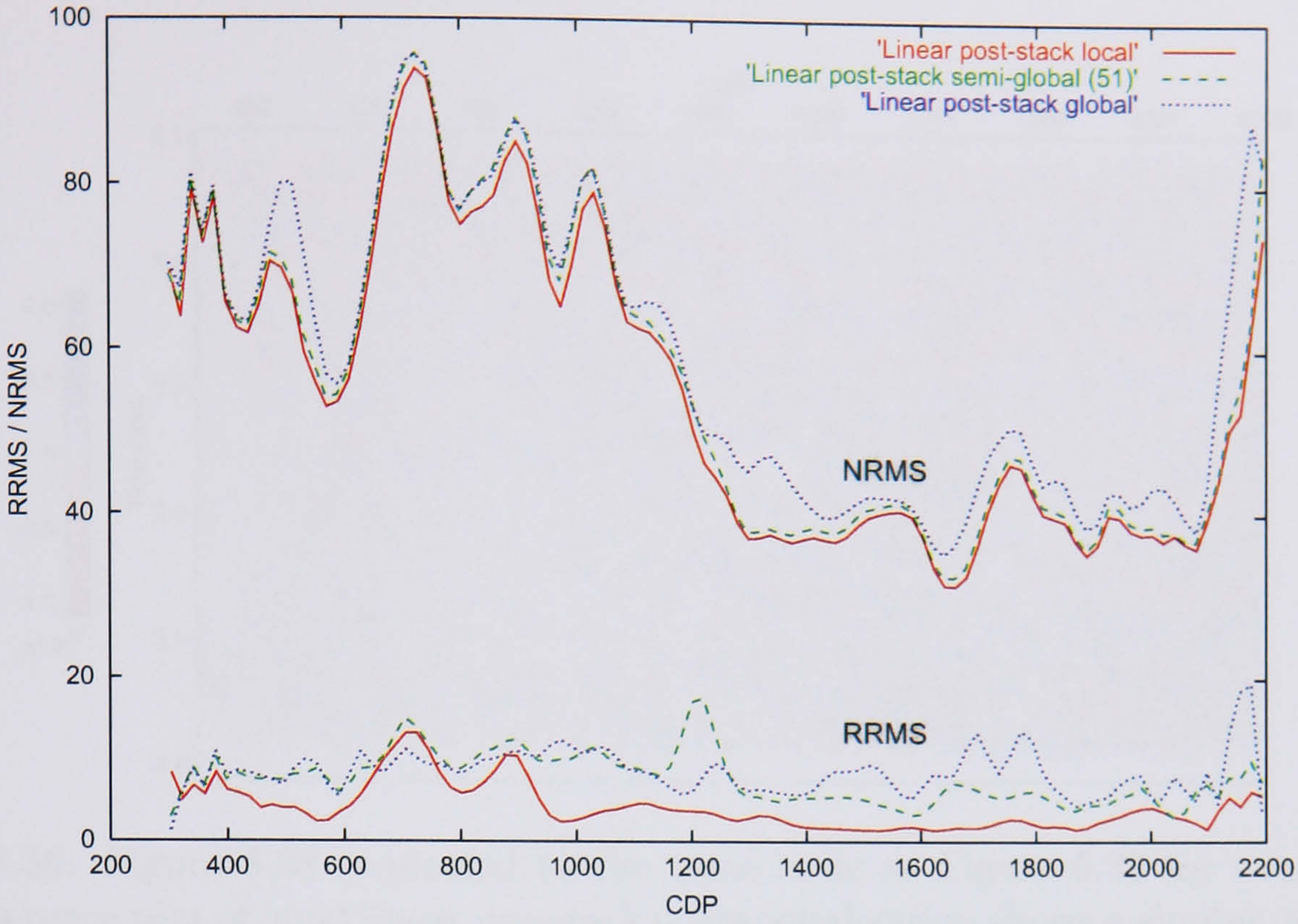


Figure 6.35: Comparison of normalized and residual root mean square (NRMS and RRMS, respectively) between the different modes of the linear post-stack cross-equalization. As expected, the local mode is superior in matching as indicated by the lower values of NRMS and RRMS.

through analysis of predictability, NRMS, and RRMS, which give an additional insight.

6.2.3 Post-stack versus pre-stack cross-equalization: a quantitative analysis

The predictability, and normalized root mean square (NRMS), between the cross-equalized monitor stack and the base survey stack, graphs are shown in Figure 6.37 and Figure 6.38, respectively. These graphs compare matching performance of post-stack versus pre-stack cross-equalization. The two plots confirms the results seen from the difference plots. Therefore, ANN cross-equalization is superior to the linear counterpart in both pre-stack and post-stack. In addition, ANN pre-stack is superior to its post-stack counterpart, but the two linear cross-equalizations are similar.

The residual root mean square (RRMS), between the cross-equalized monitor stack and the base survey stack, is shown in Figure 6.39. Although the difference plots,

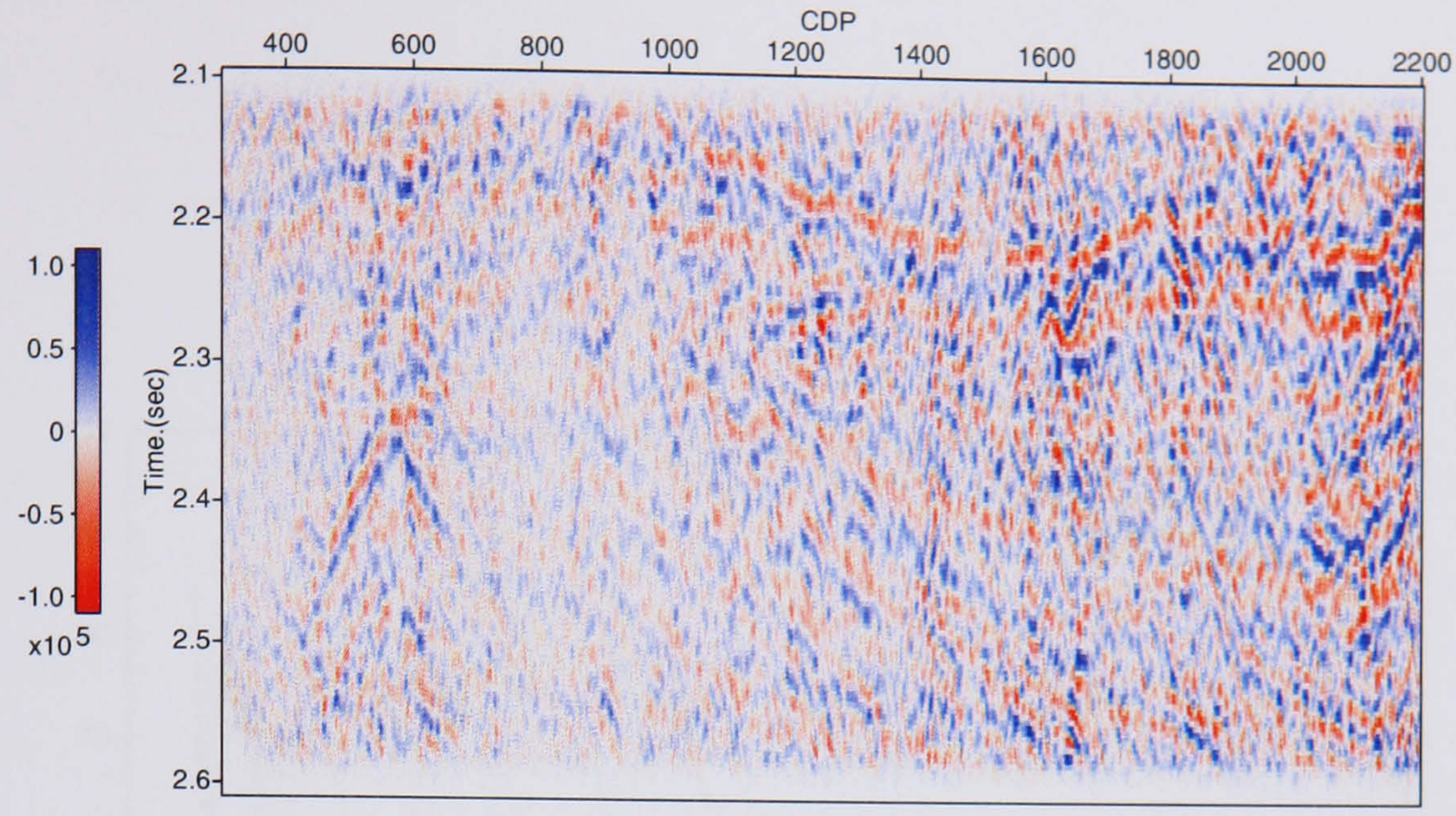


Figure 6.36: Figure 5.38 re-plotted to the same scale as Figure 6.32 for comparison. This difference plot of local linear pre-stack cross-equalization shows a similar or slightly inferior matching to that shown on the post-stack counterpart (*cf.* Figure 6.32).

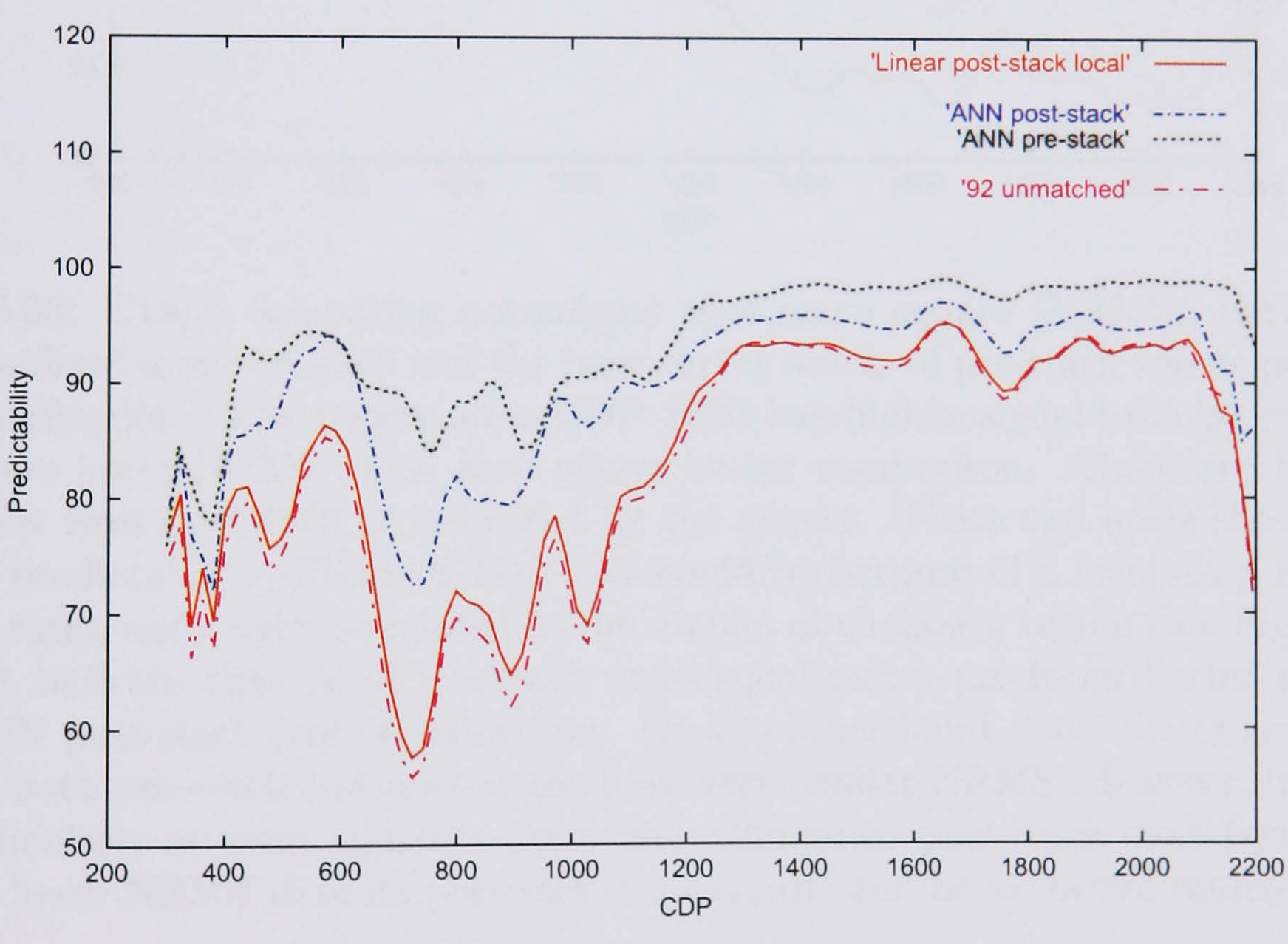


Figure 6.37: Graph comparing predictability, between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization. This plot shows, as seen earlier, that ANN pre-stack cross-equalization produced higher increase in predictability than its post-stack counterpart. In addition, the plot shows that the linear post-stack cross-equalization does not alter predictability, in accordance with what was seen previously with pre-stack linear cross-equalization.

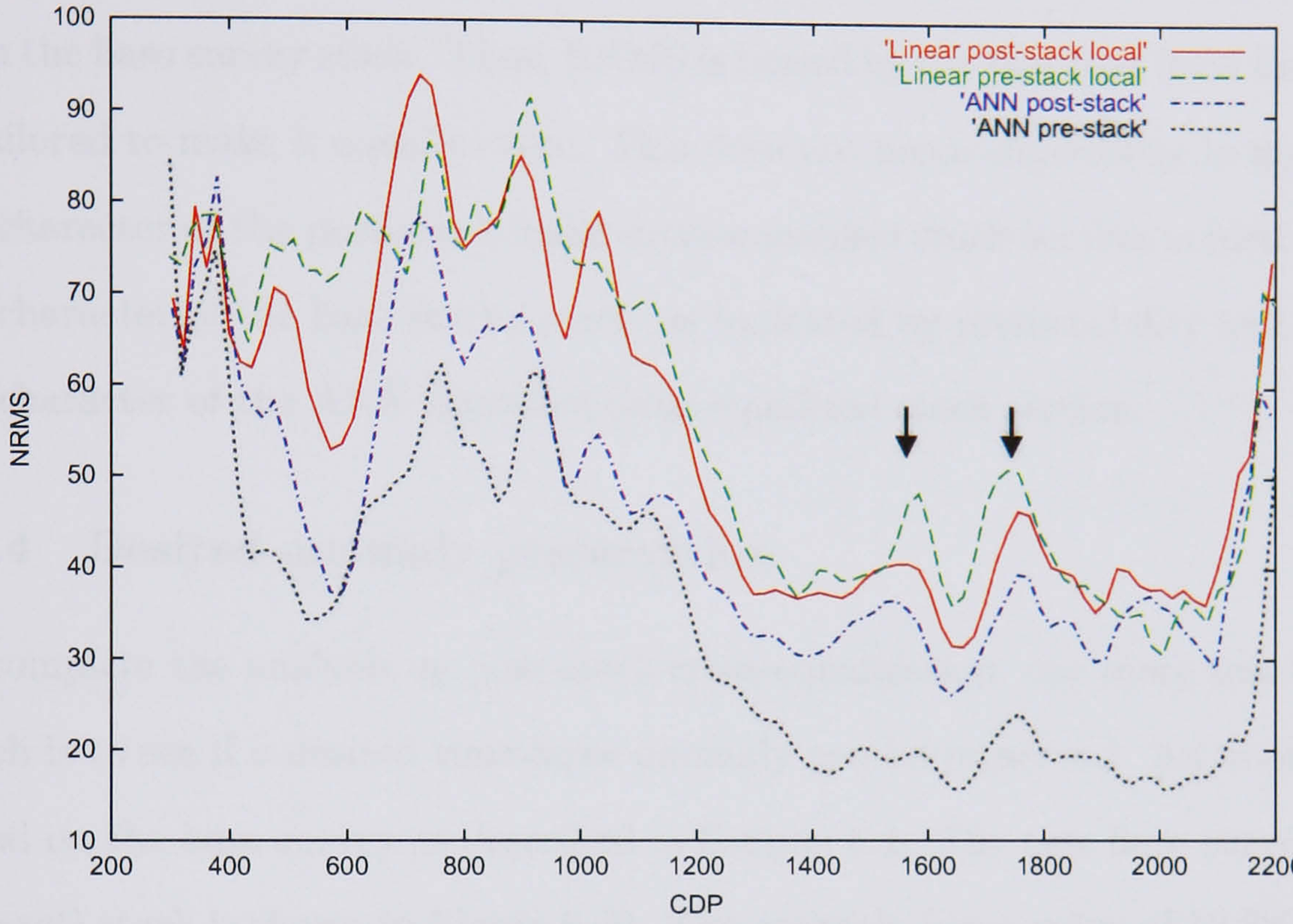


Figure 6.38: Graph comparing normalized root mean square (NRMS), between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization. The region after CDP 1200 has higher signal-to-noise ratio, and thus shows lower NRMS. This area allows better comparison. There are two clear anomalous rises in NRMS as indicated by the arrows. These two areas show also as a dip in predictability (Figure 6.37). This could be because of a local drop in signal-to-noise ratio, and might be related to the diapirs at shallower region (see Figure 3.9). The plot indicate that ANN pre-stack cross-equalization produced better matching than ANN post-stack cross-equalization. On the other hand, local linear cross-equalization, both pre-stack and post-stack, have very similar NRMS. However, in general and particularly at lower signal-to-noise ratio, the linear post-stack cross-equalization achieves lower NRMS than its pre-stack counterpart, and hence better matching.

predictability, and NRMS, all show ANN pre-stack cross-equalization superior to the other cross-equalization schemes, this figure shows local linear post-stack cross-equalization more superior, with a RRMS value close to zero. This is only because, during linear post-stack cross-equalization, a trace r_m from the monitor survey stack is multiplied by a value that makes it equal in root mean square (RMS) to the corresponding trace r_b from the base survey stack. Thus, RRMS is biased by the fact that local linear matching is tailored to make it equal to zero. This does not mean superiority in matching, since the character of the post-stack linear cross-equalized stack section is further away from the character of the base survey stack, as indicated by predictability and NRMS, than the character of the ANN pre-stack cross-equalized stack section.

6.2.4 Desired anomaly preservation

To complete the analysis on post-stack cross-equalization, one more test is performed, which is to see if a desired time-lapse anomaly can be preserved. An anomaly is introduced on the base survey as described in Section 6.1. The new base survey (*NewBaseDataset*) stack is shown in Figure 6.40. The anomaly has a value of 12.8% as measured by Equation 6.1. This anomaly is shown in Figure 6.41 as the difference between the *NewBaseDataset* stack (Figure 6.40) and the original *BaseDataset* stack (Figure 6.26).

ANN cross-equalization

ANN post-stack cross-equalization was performed to cross-equalize the monitor survey stacks to match the *NewBaseDataset* stack. The difference plot, between the *NewBaseDataset* stack (Figure 6.40) and the stack of the ANN post-stack cross-equalized monitor survey, is shown in Figure 6.42. The anomaly is hard to distinguish in the difference plot from other matching error.

Figure 6.24 shows a zoom at the anomaly location and compares the *NewDifferenceDataset* to the *AnomalyDataset*. This latter figure further indicates that the anomaly

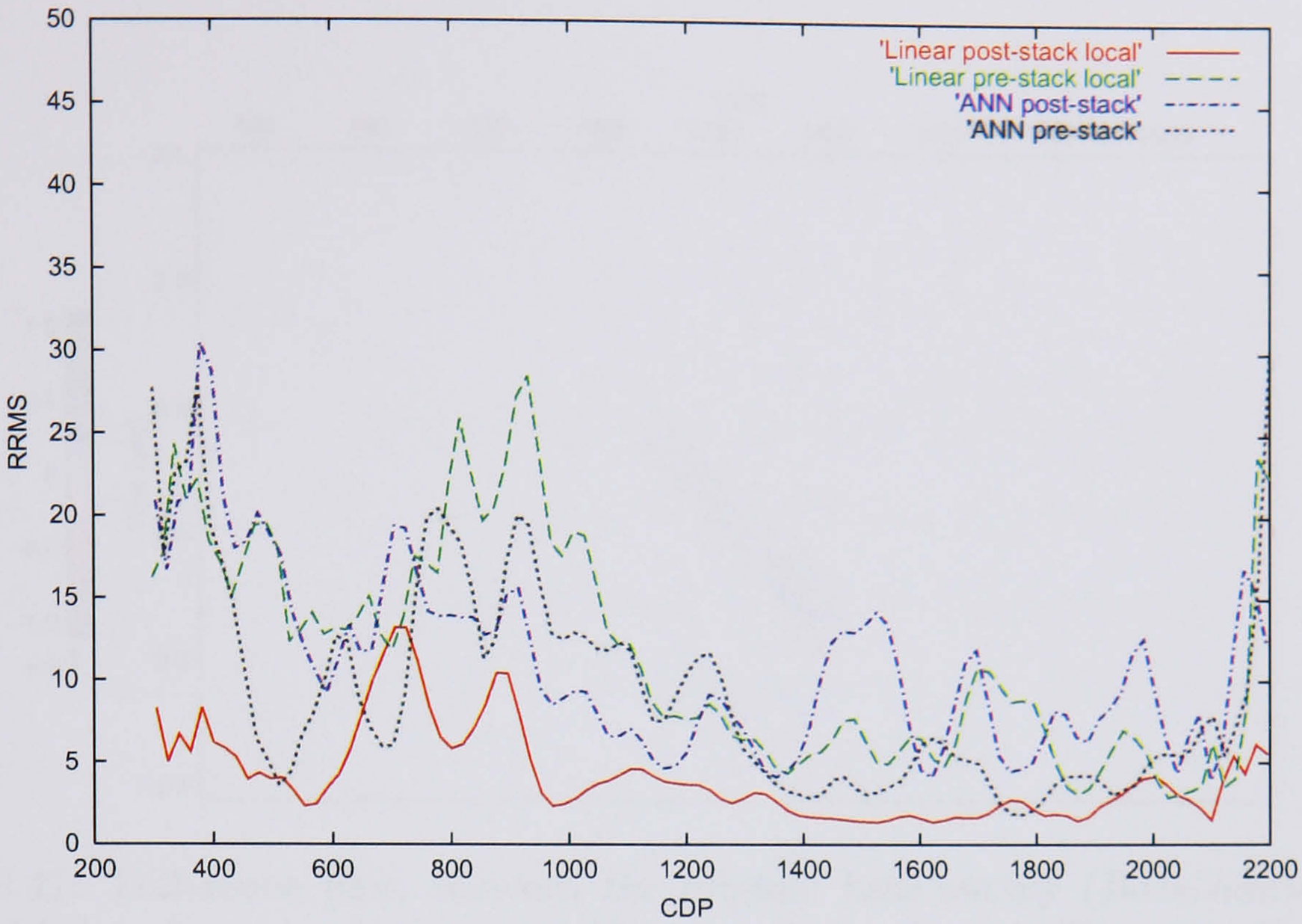


Figure 6.39: Graph comparing residual root mean square (RRMS), between the cross-equalized monitor stack and the base survey stack, of pre-stack versus post-stack cross-equalization. This plot confirms the superiority of ANN pre-stack cross-equalization over ANN post-stack cross-equalization. However, it shows that the local linear post-stack cross-equalization not only has a lower RRMS than its post-stack counterpart, but even lower than ANN pre-stack cross-equalization, see text for discussion.

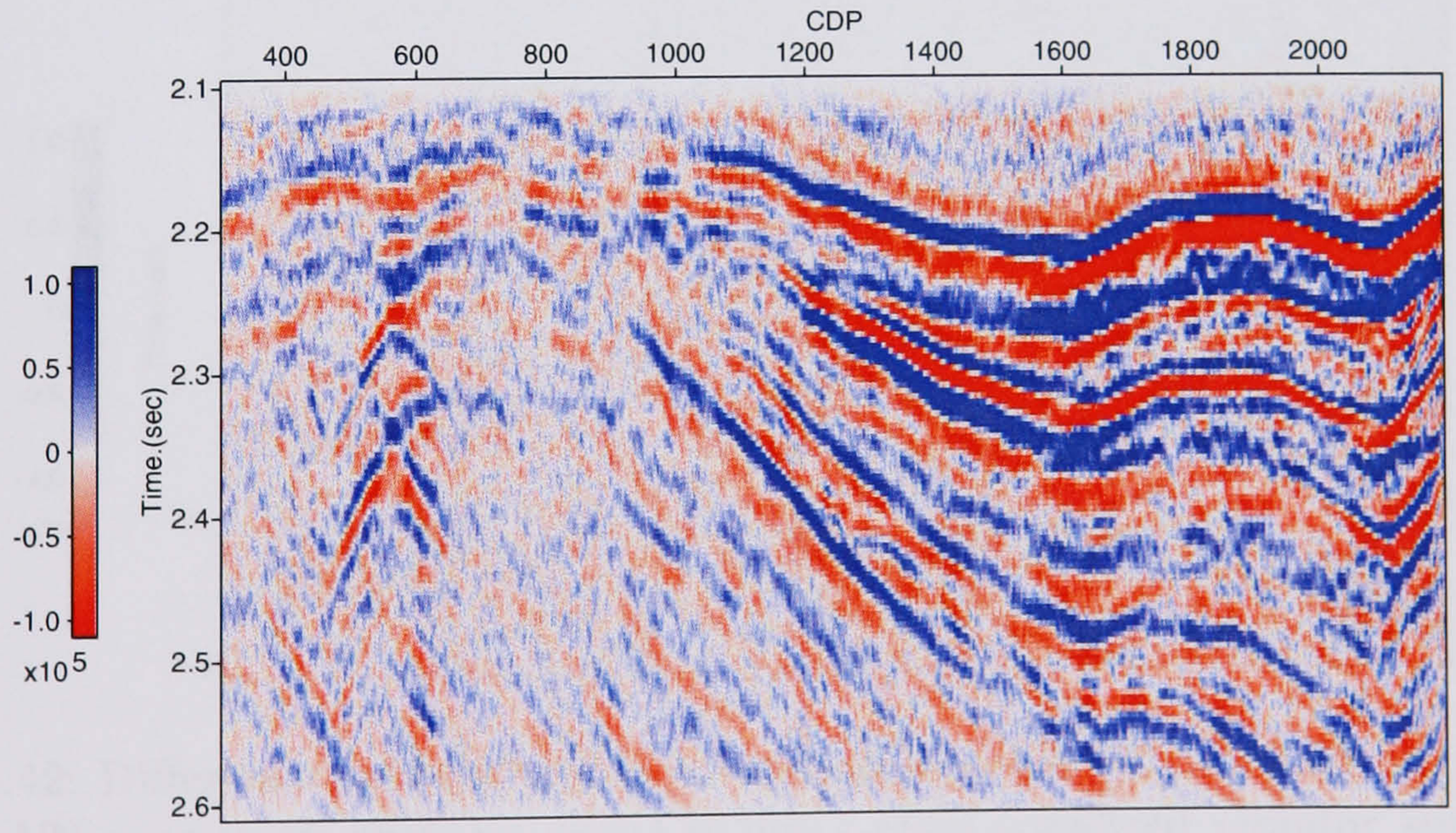


Figure 6.40: Seismic stack plot of all the offset gathers of channels 1–230 of the base survey with added 12.8% anomaly (*NewBaseDataset*).

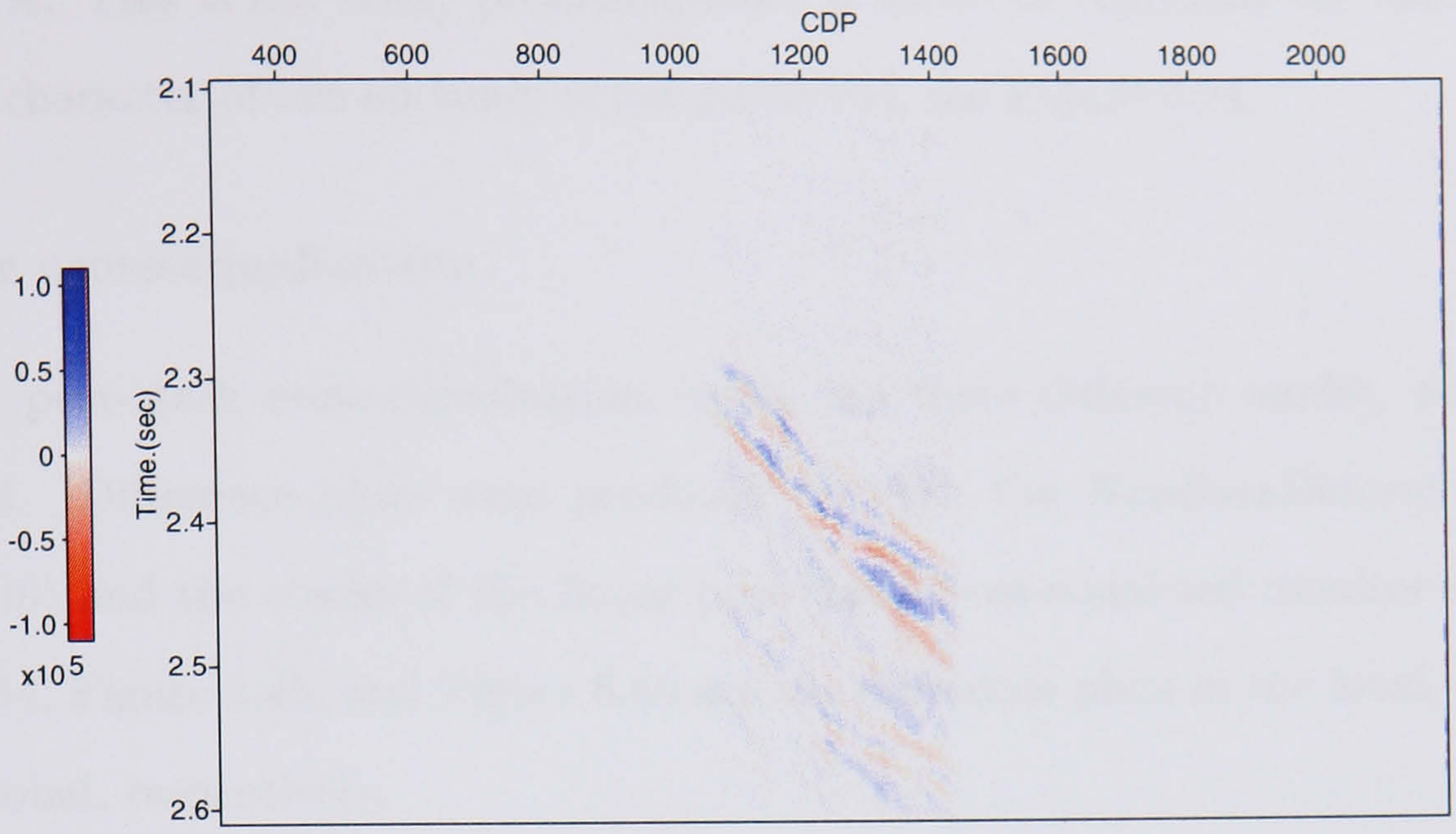


Figure 6.41: Difference plot, between the original base survey (*BaseDataset*) stack (Figure 6.26) and the new base survey (*NewBaseDataset*) stack (Figure 6.40), showing the introduced 12.8% anomaly.

Figure 6.42: Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the ANN post-stack cross-equalized monitor cross-equalized monitor survey. The introduced time-lapse anomaly is hard to distinguish from other matching errors.

is hard to distinguish. The anomaly measured on the cross-equalized stack has a value of 16.1%. This is not really promising since it seems to represent the matching error, as the character of the anomaly is not preserved, see Figure 6.24.

Linear cross-equalization

Linear post-stack cross-equalization, using the three different modes, was also performed. Difference plots were produced between the *NewBaseDataset* stack (Figure 6.40) and the stacks of the linear post-stack cross-equalized monitor survey. Figure 6.44, Figure 6.45, and Figure 6.46 are the difference plots of the local, semi-global, and global, respectively.

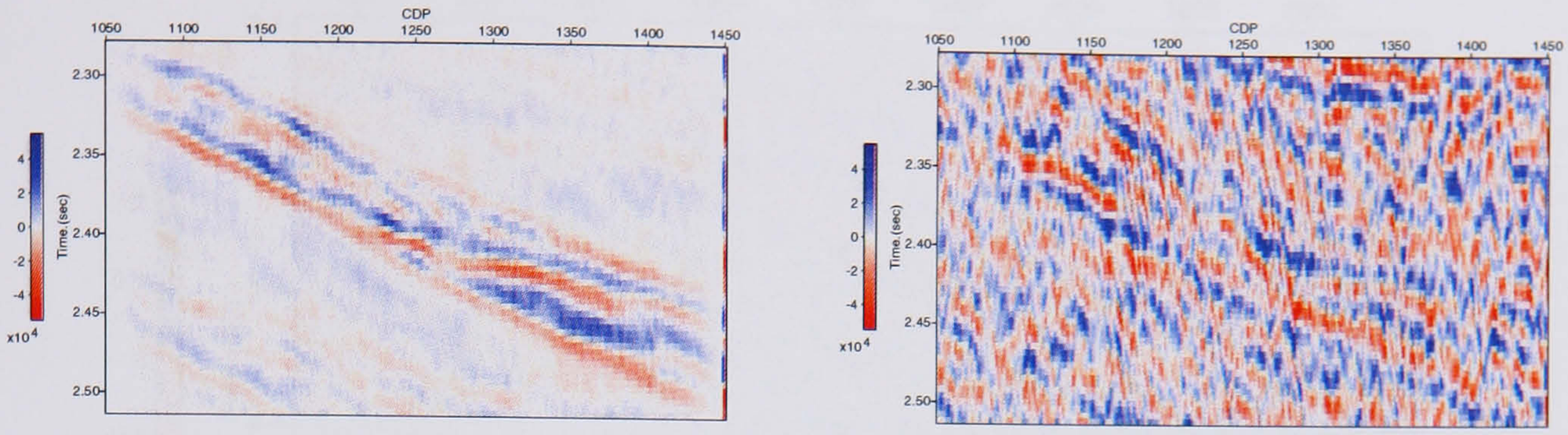
The difference plots in all the three different modes of linear cross-equalization, do not allow one to pick the location of the introduced anomaly. Figure 6.47 shows a zoom at the anomaly location and compares the *NewDifferenceDataset* plots to the *AnomalyDataset*. This latter figure further indicates that the anomaly is not preserved in character, it is both masked by the matching error and obliterated.

6.2.5 Discussion and conclusions

The results seen in this section can be summarized as follows: firstly, the ANN pre-stack cross-equalization produces better matching than the ANN post-stack cross-equalization; secondly, ANN post-stack cross-equalization gives slightly better matching than the linear post-stack cross-equalization; and finally, the linear post-stack cross-equalization gives slightly better or similar result to the linear pre-stack cross-equalization.

The results might seem puzzling: why is ANN pre-stack cross-equalization much better than ANN post-stack cross-equalization, why are the two linear cross-equalization similar, and why is the difference between the ANN and the linear cross-equalization much greater in the pre-stack than in the post-stack cross-equalization?

The results can be explained by understanding that there are two main factors at play when comparing pre-stack to post-stack cross-equalization, and that these two



(a) *AnomalyDataset* (12.8% anomaly) (b) *NewDifferenceDataset* (12.8% anomaly)

Figure 6.43: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when post-stack cross-equalization is used and an anomaly of 12.8% is introduced. The plots are plotted with a higher scale than previously used to enable better comparison. These two plots indicate that the introduced anomaly, shown in (a), cannot easily be distinguished in (b) from other matching error.

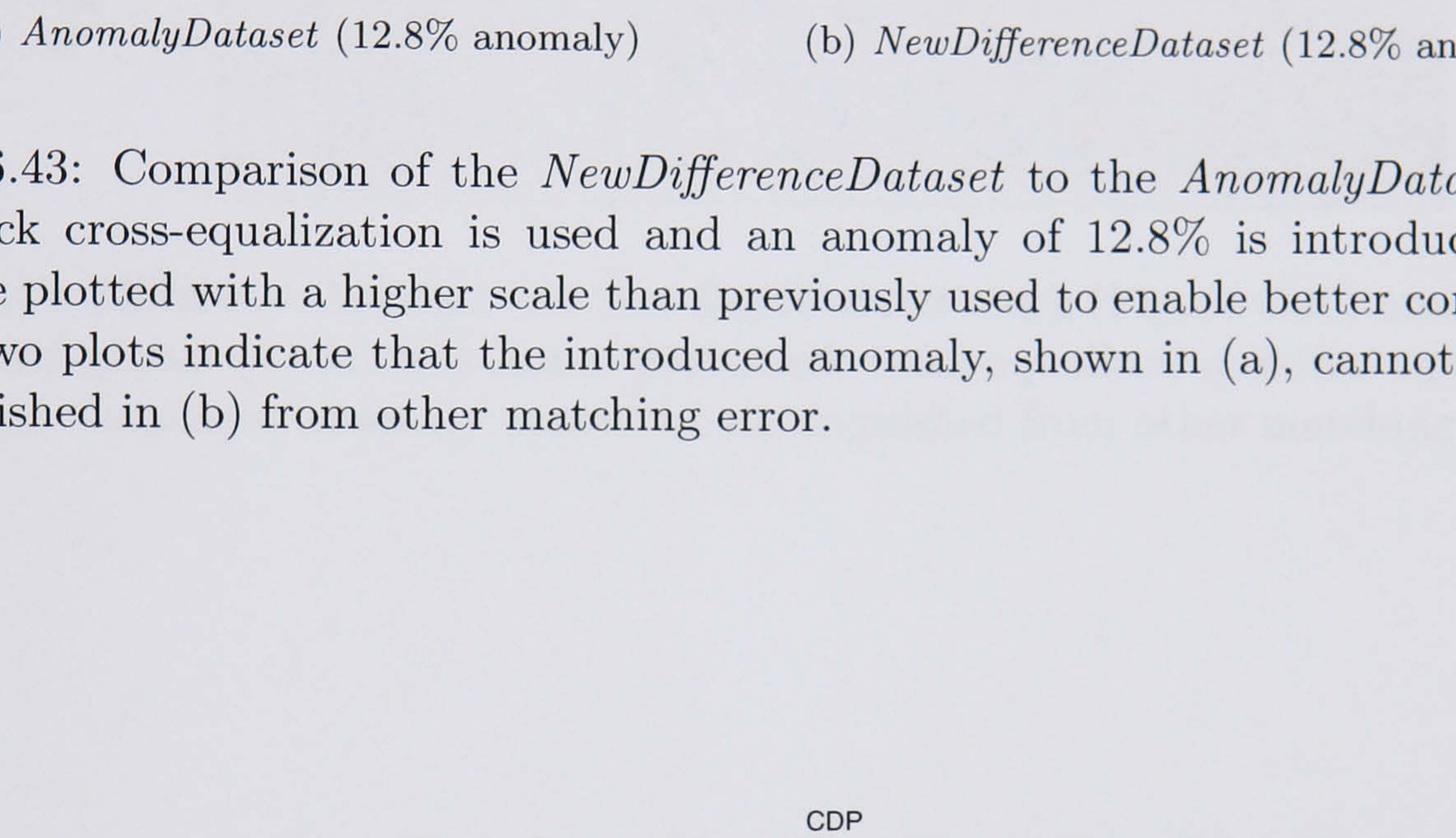


Figure 6.44: Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the local linear post-stack cross-equalized monitor survey. The introduced time-lapse anomaly cannot be distinguished from other matching error.

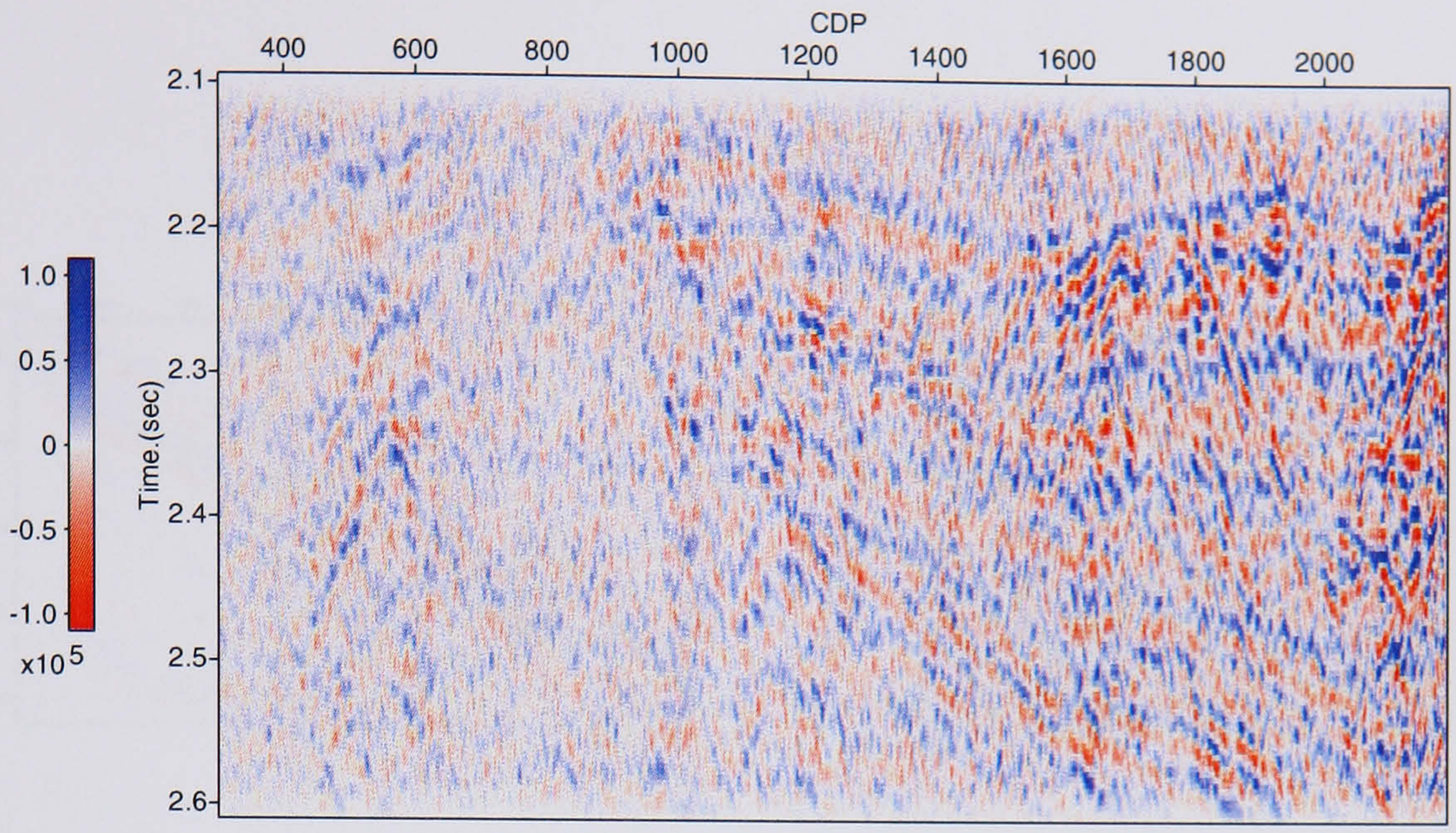


Figure 6.45: Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the semi-global (51 traces) linear post-stack cross-equalized monitor survey. The introduced time-lapse anomaly cannot be distinguished from other matching error.

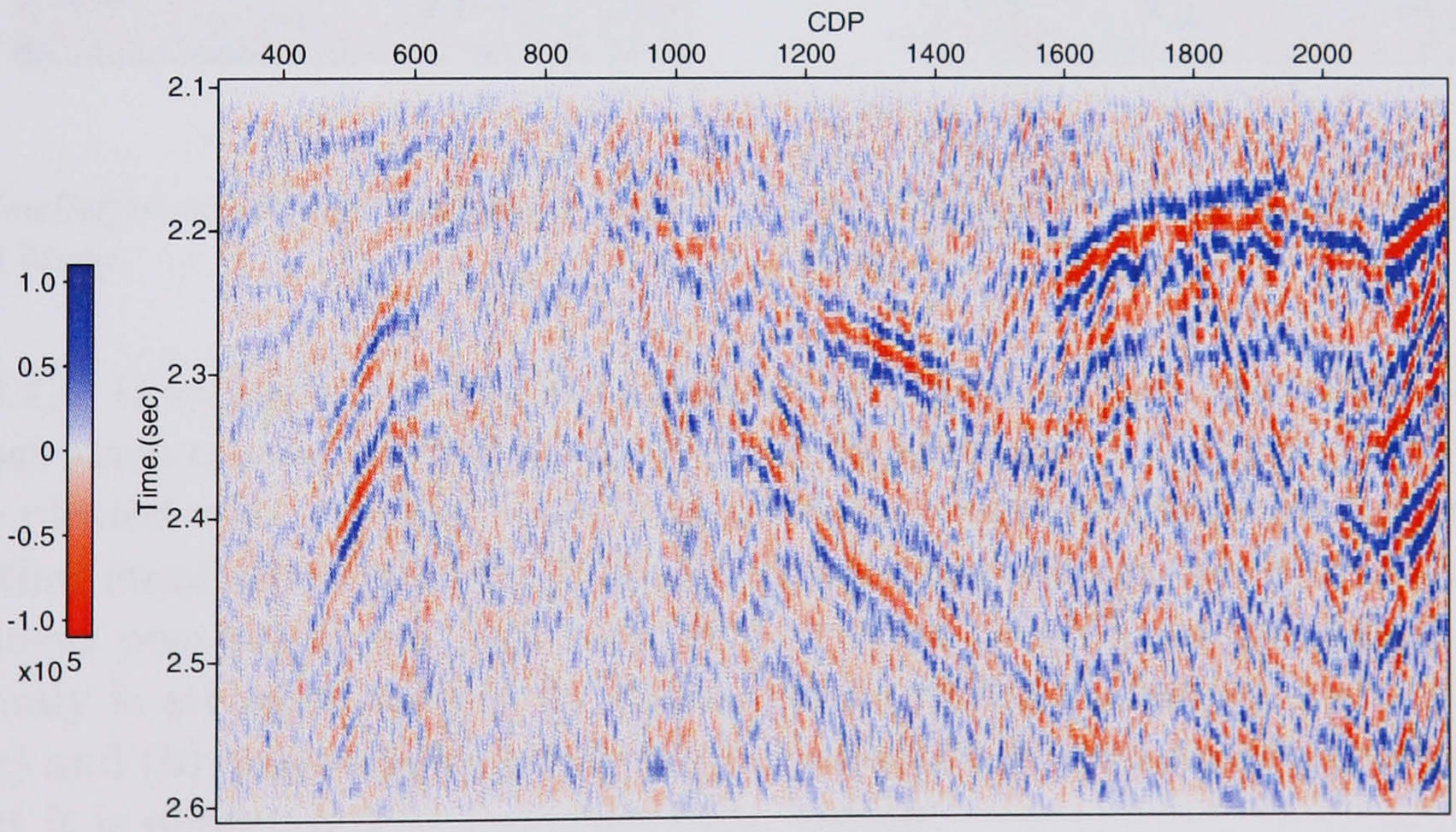
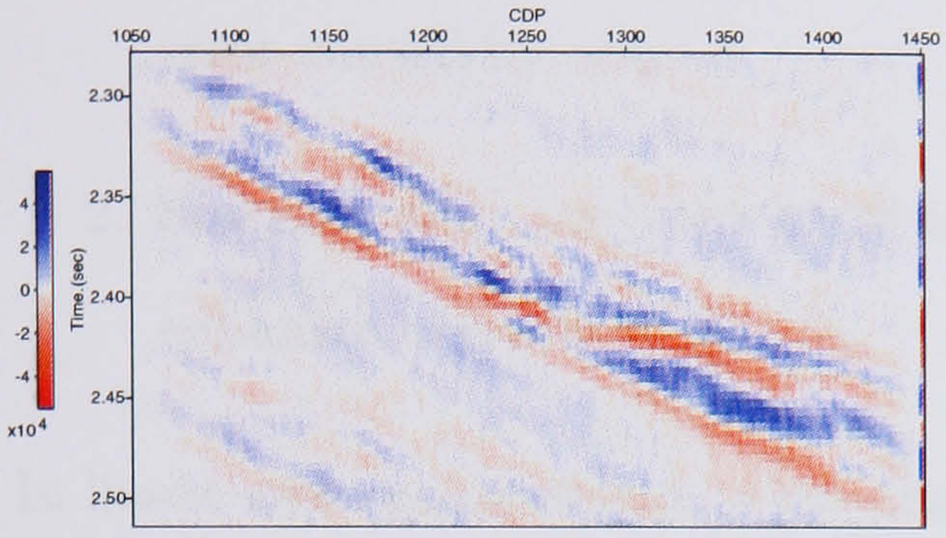
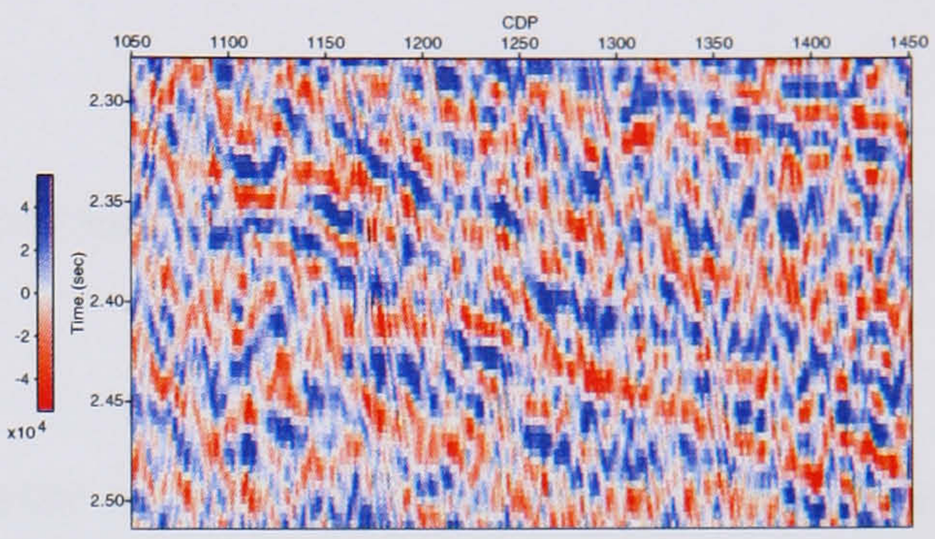


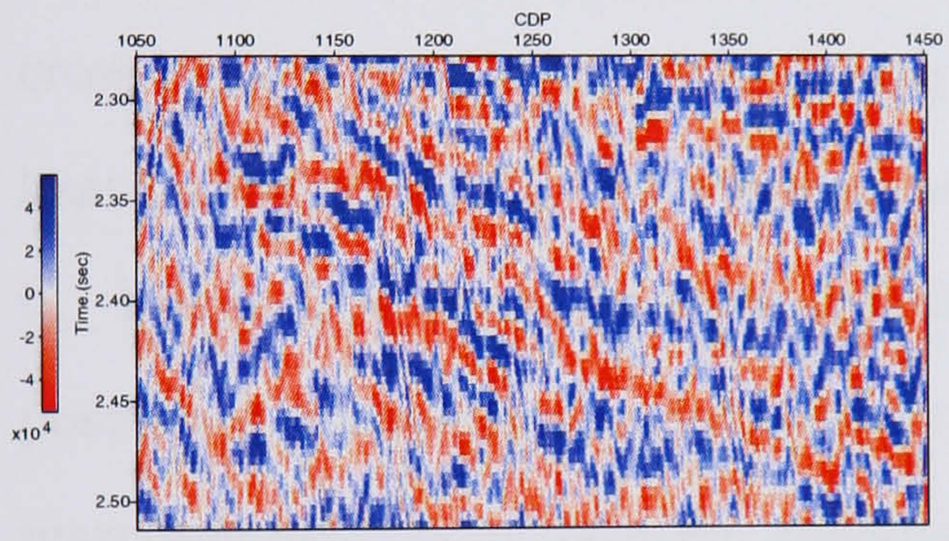
Figure 6.46: Difference between the *NewBaseDataset* stack (Figure 6.40) and the stack of the global linear post-stack cross-equalized monitor survey. The introduced time-lapse anomaly cannot be distinguished from other matching error.



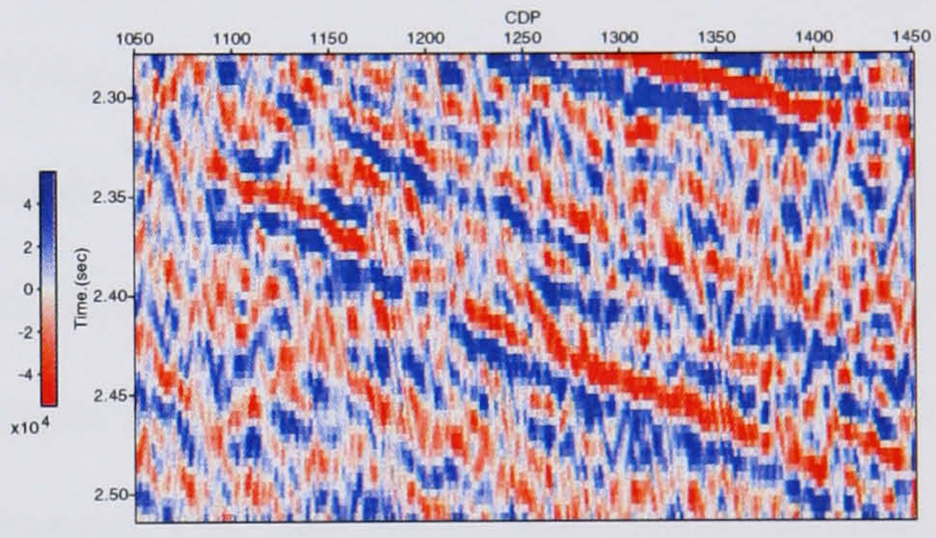
(a) *AnomalyDataset* (12.8% anomaly)



(b) *NewDifferenceDataset* post-stack local linear



(c) *NewDifferenceDataset* post-stack semi-global linear



(d) *NewDifferenceDataset* post-stack global linear

Figure 6.47: Comparison of the *NewDifferenceDataset* to the *AnomalyDataset* when linear post-stack cross-equalization is used and an anomaly of 12.8% is introduced. The plots are plotted with a higher scale than previously used to enable better comparison. This further emphasizes that the introduced anomaly seen in (a) is not preserved on all the linear post-stack *NewDifferenceDataset* plots (b)–(d). In the global mode (d), the anomaly is strongly masked by the matching error. In the semi-global and local mode ((c) and (b), respectively), not only the anomaly is seen masked by the matching error, but it is obliterated by the cross-equalization.

factors affect differently the two cross-equalization schemes (linear and ANN cross-equalization). The two factors are listed below.

1. Stacking increases signal-to-noise ratio. Therefore, if stacking is performed before cross-equalization, then it should allow better misfit prediction and less matching error caused by noise.

2. Stacking after pre-stack cross-equalization destructs random matching error and enhances correlative signals.

In linear cross-equalization, noise causes error in predicting the misfits. Therefore, pre-stack linear cross-equalization corrects for misfits that are jeopardized by noise, and are thus non-optimum for the real signal. Stacking does not solve the problem, since the horizons do not stack optimally, although, some introduced noise in linear pre-stack cross-equalization is removed during stacking. On the other hand, ANN cross-equalization has the ability to suppress noise since it cannot and therefore does not match it. Hence, pre-stack cross-equalization within one offset produce a result similar to post-stack cross-equalization. However, since noise is suppressed and mostly random matching error is introduced, stacking enhances the pre-stack matching by destructing noise and enhancing the signal.

This section also illustrated the fact that ANN cross-equalization is enhanced by including, as input, neighbouring information from neighbouring in-lines. A reason was put forward that ANN needs input information that will allow it to cater for the time-lapse change in trace position.

Finally, this section investigated desired–anomaly preservation in the post-stack cross-equalization. In the linear cross-equalization, there is clear obliteration to the introduced anomaly (12.8%) and masking by the matching error. In the ANN it was also difficult to notice anomaly preservation, since the matching error was larger than in pre-stack cross-equalization.

6.3 Summary

In order to define ANN cross-equalization as successful, it is necessary to be able not only to normalize acquisition differences but to also be able to preserve genuine production-related time-lapse anomalies. Therefore, the first part of this chapter (Section 6.1), assessed ANN cross-equalization for its ability to preserve genuine (non-acquisition) time-lapse anomalies.

A synthetic anomaly was introduced on the base survey dataset to act as a known non-acquisition time-lapse anomaly. The monitor survey was cross-equalized to match the new base survey dataset that contains the synthetic anomaly. The conducted tests required ANN cross-equalization to match the two datasets but not the introduced anomaly. The tests show that ANN cross-equalization has the ability to preserve non-acquisition time-lapse anomalies, which can be distinguished from any matching error.

The first part also attempted to measure quantitatively ANN cross-equalization ability to preserve genuine time-lapse anomalies. Results indicate that an anomaly as low as approximately 10% difference in root mean square is preserved and can be distinguished from other matching error. The quantitative measure needs further analysis and tests to be conclusive, and it only applies to ANN cross-equalization as performed on the two time-lapse dataset involved in this study. Furthermore, the % quoted is dependent on the quantification criterion as explained in Section 6.1.2. The linear cross-equalization was also tested and showed that its results are inferior to that of ANN cross-equalization and they suffer from more matching error that masks the introduced anomaly.

In the second part of this chapter (Section 6.2), post-stack cross-equalization was investigated as an alternative to pre-stack cross-equalization. It was clear that ANN cross-equalization is superior to the linear cross-equalization in both pre- and post-stack. In addition, the ANN pre-stack cross-equalization produces better matching than the ANN post-stack cross-equalization. However, it was found that the linear post-stack cross-equalization gives slightly better or similar result to the linear pre-

stack cross-equalization. The latter two and seemingly puzzling results were explained by the fact that stacking before cross-equalization affect differently the two cross-equalization schemes (linear and ANN cross-equalization).

Chapter 7

Summary and Conclusions

The study this thesis presents, developed and investigated a novel technique where artificial neural networks are used to cross-equalize time-lapse seismic surveys. The study was conducted on two time-lapse datasets acquired over the Oseberg field, North Sea. This chapter summarizes the results, draws conclusions and provides an outlook for further possible work.

7.1 Summary

The first chapter introduces time-lapse seismic as an invaluable tool to reservoir management. This tool allows the monitoring of hydrocarbon production through the visualization and possible quantification of changes introduced by such production. However, this tool is largely hindered by the lack of repeatability between the different time-lapse surveys, especially those involving legacy surveys or those acquired with different shooting geometry. This lack of repeatability can cause ambiguity in time-lapse seismic interpretation.

Cross-equalization, a method that normalizes for non-production related time-lapse differences, was shown necessary to solve the aforementioned problem. Cross-equalization was described and divided into two main categories: deterministic, where time-lapse surveys are matched by correcting for known sources of mismatch; and statistical,

where a matching filter is designed to minimize the misfits between the two time-lapse datasets in a least square sense. Furthermore, the conventional cross-equalizing procedures and the main cross-equalization algorithms were summarized.

There is no one standard cross-equalization technique currently available. However, in most cases, cross-equalization involves: firstly, interpolating the two time-lapse surveys to a common grid to align the traces especially when the two time-lapse survey geometries are different; and secondly, applying a statistical cross-matching or a mixture of statistical and deterministic cross-matching, since the deterministic is usually not sufficient by itself. Chapter 1 concludes by describing a desire to find a cross-equalization tool that should not require exact knowledge of the reasons behind the lack of repeatability; should correct not only for the four basic misfit elements (time, amplitude, phase, and bandwidth), but also for other misfits such as trace misalignments and noise; and needs to be able to handle interactions between the misfit elements and should not be hindered by the possible non-linearity of the relationship. This motivated me to investigate artificial neural networks as a possible tool for time-lapse seismic cross-equalization with all the aforementioned merits. Therefore, Chapter 1 puts forward the study objective so as to develop a scheme to cross-equalize two time-lapse seismic datasets using artificial neural networks.

The background to artificial neural networks (ANNs), with an introduction to their main concepts and basic theories, is provided in Chapter 2. The different parameterizations associated with ANNs are outlined. The standard back-propagation training algorithm is explained. In addition, an alternative training algorithm, the resilient propagation (RPROP), is described and subsequently used in this study. The choice to use RPROP was taken primarily because of its ability to update automatically its learning parameters. Weight decay was introduced in the training algorithm to enhance generalization and to allow the interpretation of the relative importance of the different inputs through ANN's connection–weight analysis.

The background to the time-lapse datasets used in this study is provided in Chap-

ter 3. The datasets were acquired over the Oseberg field as part of a 4D seismic project. The study was conducted on a time-lapse swath from the base survey (NH8906) and the monitor survey (NH9201) acquired during 1989 and 1992 respectively. Chapter 3 analyses and compares the two aforementioned datasets in terms of acquisition parameters and seismic character. The two datasets display differences, even outside the reservoir area, that appear as misfits in time, phase and amplitude. These differences are due to three main reasons: firstly, there is no exact equivalent trace with the same offset and common-mid-point in the monitor survey for every trace in the base survey; secondly, there is difference between the two time-lapse datasets in the seismic source wavelet and receiver response; and thirdly, there exists different noise in each of the two time-lapse datasets.

A summary of the previous studies conducted on the time-lapse datasets used is also provided in Chapter 3. The summary indicates difficulty in interpreting genuine production-related time-lapse anomalies from others due to the lack of repeatability problem between the time-lapse surveys. In addition, the summary shows that any production-related anomaly that exists between the surveys is very subtle.

The implementation of ANNs in time-lapse cross-equalization is presented and discussed in Chapter 4. Cross-equalization of the two time-lapse datasets was addressed as producing a cross-equalized match to one in-line of the base survey from three in-lines of the monitor survey. Each offset gather was matched independently from the others to allow fast parallel implementations and more effective matching. The different parameterizations required were assessed and, with support of experimental results, a choice was put forward.

The codes and procedures implemented, while using ANNs as a cross-equalization tool, are also presented in Chapter 4. In addition, a description is given to the methods used to quality control (QC) the results of cross-equalization. These include both visual analysis and quantitative measures. Finally, Chapter 4 presents a test that determined that, while training to perform cross-equalization, the ANNs are not memorizing a

copy of the dataset required to be matched, but in fact are learning a function that normalizes the acquisition differences.

The historic development of the cross-equalization scheme is described in Chapter 5. Initially, the ANNs' training patterns were normalized seismic data transformed to the wavelet domain. Section 5.1.5 demonstrates the final test performed in this domain, and illustrates how the wavelet domain was used to reduce the size of the problem into small parallel problems and concentrate the effort of matching on the required time and frequency. The time domain was later selected to be the domain to represent the seismic data in training patterns. This shift was because the results achieved using the time domain were slightly better and the cross-equalization procedure was made slightly simpler. However, the wavelet domain is not ruled out as a viable domain that offers its own advantages.

The input and target output parameters of the ANNs are considered to be the prime factor in deciding the matching capability of ANNs. Therefore, these were optimized through many trial and error experiments and through ANN's connection–weight analysis. The latter, aided by the use of the weight decay in the training algorithm, allowed the interpretation of the relative importance of the different inputs to the networks.

The final optimized input and target output parameter choice, together with all the other optimized parameterizations described in Chapter 4, were used in the final matching test in Chapter 5 in Section 5.4.1. The results of this final test were analysed and compared with results from a more conventional linear cross-equalization method.

ANN cross-equalized monitor survey stack when compared to the base survey stack has a predictability of 98%, normalized root mean square (NRMS) of 20%, and residual root mean square (RRMS) of 4%. These values are compromised slightly in areas with low signal-to-noise ratio with predictability being around 92%, NRMS around 50, and RRMS around 10%. However, the compromise is far lower than that seen in linear methods.

The matching achieved by ANN proved to be superior to the more conventional

linear cross-equalization methods analysed. This superiority in matching was shown to be partly due to ANNs ability to achieve non-linear transforms. In addition, ANN cross-equalization scheme is able to cater for any time-lapse change in trace position. without relying on direct trace-to-trace correspondence or initial interpolation to a common grid. This latter ability of the ANN cross-equalization scheme, with reasons behind such ability, is demonstrated more clearly later in Section 6.2.1 (Chapter 6).

The application on the test in Section 5.4.1, with the optimal configuration, took 16 CPU days in 500 MHz EV5 21164 Alpha Workstations to cross-equalize 512 ms of 130 offset gathers. However, the cross-equalization can easily be conducted as 130 parallel processes, since each offset is treated independently.

In order to define ANN cross-equalization as successful, it is necessary to be able not only to normalize acquisition differences but to also be able to preserve genuine production-related time-lapse anomalies. Therefore, the final chapter, Chapter 6, describes a methodology to assess the ability of ANNs to preserve production-related anomalies while matching elsewhere.

A synthetic anomaly was introduced on the base survey dataset to act as a known non-acquisition time-lapse anomaly. The monitor survey was cross-equalized to match the new base survey dataset that contains the synthetic anomaly. The tests conclude that ANN cross-equalization has the ability to both normalize for acquisition differences and preserve genuine time-lapse anomalies. The conclusion is valid for time-lapse anomalies that exceed in magnitude the general matching error, which as seen in Section 5.4.1 is very low. Furthermore, the linear cross-equalization was also tested for its ability to preserve a genuine time-lapse anomaly and showed that its results are inferior to that of ANN cross-equalization primarily because its matching error masks the introduced anomaly.

Results also indicate that a production-related anomaly as low as approximately 10% in root-mean-square amplitude difference (Equation 6.1) is preserved and can be distinguished from other matching error. Equation 6.1 may need improvement to give

a % that is independent of the anomaly location. In addition, the quantitative measure needs further analysis and tests to be conclusive, and it only applies to ANN cross-equalization as performed on the two time-lapse dataset involved in this study. Furthermore, it is likely to be highly dataset dependent, and ANN cross-equalization may actually be more successful within full 3D time-lapse datasets as will be outlined in Section 7.3.

The second part of Chapter 6 assesses post-stack cross-equalization as an alternative to pre-stack cross-equalization, and compares the two. It is clear that ANN cross-equalization is superior to the linear cross-equalization in both pre- and post-stack. In addition, the ANN pre-stack cross-equalization produces better matching than the ANN post-stack cross-equalization. However, it was found that the linear post-stack cross-equalization gives slightly better or similar results to the linear pre-stack cross-equalization. The latter two, seemingly puzzling results, were explained by the fact that stacking before cross-equalization affects differently the two cross-equalization schemes, i.e. the linear and the ANN cross-equalization.

7.2 Conclusions

The goal of developing a scheme to cross-equalize two time-lapse seismic datasets using artificial neural networks (ANNs) has been achieved. In addition, the investigation and optimization of the different criteria and parameters and the development of the codes necessary were fulfilled. Furthermore, the ability of the cross-equalization scheme to preserve a production-related time-lapse anomaly within the two time-lapse seismic datasets while matching acquisition anomaly was tested. This latter test proved that the scheme has the ability to distinguish between the two aforementioned anomalies and perform good matching to the acquisition anomaly only.

ANNs, as a cross-equalizing tool for time-lapse seismic, was compared with other more conventional methods and proved superior. Its superiority stems from ANN's ability to perform non-linear transfers and the ability of the ANN cross-equalization scheme to cater for the time-lapse change in trace position.

7.3 Outlook

The work presented in this thesis could be carried forward in an application or further research. Here, I outline several aspects that are worth considering.

It is beneficial to carry out all the possible deterministic linear cross-equalization before applying ANN cross-equalization. Therefore, if there exists a known time-delay or phase shift between the time-lapse datasets, then these should be corrected for. Another example, if different de-ghostings need to be applied because of difference in streamer depths, then these should be applied prior to ANN cross-equalization. All these deterministic linear cross-equalizations will ensure that ANNs are not overwhelmed by having to solve problems for which a simple and known procedure exists. Section 4.6.1 (Item 2) showed an example of this where a known time-delay was corrected for before applying ANN cross-equalization.

ANNs learn best from training patterns that exhibit similar features. Therefore, instead of training ANNs across one long in-line, it is more useful to train ANNs across several shorter and parallel in-lines. In other words, application of ANN cross-equalization may be more successful within full 3D time-lapse datasets (4D seismic) than within 2D time-lapse datasets, since ANNs can be trained and applied within small horizontally equi-dimensional time-lapse areas. This is because the geology, and therefore the character of the training patterns, varies more along one long line than across an area with shorter sides. In the application shown in this thesis, approximately 462 shots from one long 2D time-lapse line are used to train ANNs, which are then applied on that long in-line. It may be more useful to use an equi-dimensional 3D time-lapse area, i.e. approximately 44 shots from 11 parallel in-lines.

The cross-equalization shown was conducted on data that were re-sampled to 8 ms sampling interval. The intention was to reduce the computational cost. After achieving a scheme that allows faster computation, work continued on 8-ms-resampled data to be consistent with procedures in previous tests. However, cross-equalization results may improve by applying cross-equalization on 4 ms sampled data.

A methodology was described in Chapter 6 to assess the ability of ANNs to preserve production-related anomalies while matching otherwise. The methodology involves introducing a synthetic time-lapse anomaly at a known location and with a known magnitude. This method could be employed to QC ANN cross-equalization on different time-lapse datasets. Therefore, if one intends to use ANN cross-equalization it could be beneficial to introduce a similar synthetic anomaly outside the reservoir area. By training the ANNs on the datasets and applying at and around the location where the synthetic anomaly was introduced, one can determine the quality and likelihood of ANN cross-equalization producing good results, i.e. not obliterating the introduced anomaly and matching very well otherwise. In fact, this method can be used as a QC tool for other cross-equalization methods.

A test was presented in Chapter 4 that ensures that, while training to perform cross-equalization, the ANNs are not memorizing a copy of the dataset required to be matched, but in fact are learning a function that normalizes acquisition difference. This also seems to be an essential QC required before performing ANN cross-equalization. In fact, this QC tool is recommended for any ANNs application, where ANNs are required to learn a mapping function and not to act as a database.

Quantitative analyses on ANNs ability to preserve desired time-lapse anomalies was presented in Chapter 6. However, many more tests are required to better quantify this ability. In addition, tests are required on many different datasets. These tests will enable relating the success of ANN cross-equalization to issues such as signal-to-noise ratio, magnitude and type of production-related time-lapse anomalies that can be preserved, type of acquisition differences that can be successfully cross-equalized, size of the datasets required to train the ANNs, and the relative size of the seismic area occupied by production-related time-lapse anomaly compared with that of the whole dataset.

The use of ANNs involves many parameters which need optimization. I presented the relevant concepts and algorithms in Chapter 2, and investigated some of those in

Chapter 4. However, there are two aspects that are worth further investigations:

1. It can be useful to investigate algorithms that can find an optimal network size. Growing algorithms, such as Cascade Correlation (Fahlman and Lebiere. 1990), or pruning algorithms, such as Optimal Brain Damage (LeCun et al., 1990b), can lead to a network that generalizes better and has the smallest required size, and thus requires less computational time when trained or applied.

2. It can also be useful to investigate other algorithms that perform a fast search for an optimal weight matrix during training and attempt at solving the many problems linked to this search. Since the search for global minima in multidimensional space is not confined to ANNs applications and is found in many other fields, many such algorithms exist.

I argued that the use of RPROP training algorithm complemented by the use of *weight decay* during training and by a systematic investigation of different topologies yields good results, where a network that is optimal in size and performance is found. Indeed the criteria employed proved to be robust in finding a network that trains relatively fast, allows for good generalization, and performs well. In addition, the criteria enabled an effective connection–weight analysis that led to optimizing the input parameter choice. Furthermore, the criteria allowed me to concentrate on other issues related to the project without being overwhelmed by ANN parameter optimization. However, research on the two aforementioned aspects may yield better results.

An example of an algorithm to solve the local minima problem is *weight jogging*. This was investigated in Section 4.5.2, but showed no effective benefit. However, there was evidence to suggest that applying the jogging with a decaying amount, where lower amount of jogging is applied with increasing number of training cycles, may produce a more optimal weight matrix, i.e. lead to lower levels of error during the ANN application.

Section 5.1.4 discusses the desirability or otherwise of excluding the reservoir area from training. I concluded that such exclusion, which is intended to ensure that ANN

cross-equalization will not try to minimize production-related time-lapse anomalies. leads to the introduction of relatively large anomalies caused by ANNs behaving in a non-optimal way in the excluded area. I further concluded that no exclusion is required to preserve production-related time-lapse anomalies, and this was proved to be accurate in Section 6.1. However, Section 6.1 also showed that ANNs can slightly normalize genuine time-lapse anomalies, as was expected.

The conclusion that favours non-exclusion of the reservoir area from training was reached before finding the final optimal parameter choices. Therefore, it may be interesting to test the effect of excluding an area from training, while using the final scheme configuration, on the cross-equalization results. In addition, an investigation could be conducted to see the effect of including a lower number of training patterns from within the reservoir area than from outside it. By excluding an area from training or treating it differently, there is of course the danger that an anomaly arises within this area which can be misinterpreted as a production-related time-lapse anomaly.

Many of the implementation codes and parametrization choices presented in this thesis may be applicable to other similar uses of ANNs in seismic. These include the different codes written to handle seismic data; to create and process training patterns; to apply trained ANNs; to QC and compare seismic traces using predictability, NRMS, and scalar misfit estimates; and to perform linear cross-equalization in different modes. These include also C++ class definitions that allow the transfer of time series to the wavelet domain and enable different functionalities related to this domain. The parametrization includes pattern pre-processing, and the choice of training algorithm and stopping criteria. In addition, the criteria of how a task can be split into small independent tasks, along the offset axis, and the criteria on how input–output was selected may be useful as an implementation choice for other similar problems.

An example of the use of the implementation design of this project to other similar projects is their use in a project where ANNs are used to interpolate missing shots. I am involved in this new project, and it is currently undertaken here at Imperial College.

In short, the task involves applying trained ANNs to interpolate missing shots on an in-line which is located between two in-lines where ANNs are trained and which are shot more densely. In this new project, one ANN was allocated per offset and trained to interpolate a trace given input from four traces closest to it in the offset and CMP gather. The ANN's output is one seismic sample and the input is a window of seismic data selected from each of the four input traces with a size that accounts for structural dips. A quick test showed that a 100 m shot spacing could be interpolated to 50 m spacing with promising quality.

Bibliography

Aboufadel, E., and Schlicker, S., 1999, Discovering Wavelets: John Wiley and Sons, Inc.

Al-Mazroey, M., 2002, Identifying non-linear time-lapse wavefield perturbations using predictability and spectral coherency: 72nd Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts.

Al-Mazroey, M. A., 2003, Fluid characterization using 4-D seismic amplitude and coherence analysis: Application to Oseberg gas/oil field: Ph.D. thesis, Imperial College.

Alberts, P., Warner, M., and Lister, D., 2000, Artificial neural networks for simultaneous multihorizon tracking across discontinuities: 70th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session: INT 5.4.

Alberts, E. P., 2002, Horizon Mapping in Exploration Seismology Using Artificial Intelligence: Ph.D. thesis, Imperial College.

Barry, K. M., Cavers, D. A., and Kneale, C. W., 1975, Report on recommended standards for digital tape formats: Geophysics, **40**, No. 02, 344–352.

Batzle, M., and Wang, Z., 1992, Seismic properties of pore fluids:, *in* Soc. of Expl. Geophys., Geophysics, 1396–1408.

Baum, E. B., and Haussler, D., 1989, What size net gives valid generalization?: Neural Computation, , No. 1, 151–160.

Becker, W. M., Kleinsmith, L. J., and Hardin, J., 2000. The world of the cell: The Benjamin/Cummings Publishing Company.

Becker, W. M., 1986, The world of the cell: The Benjamin/Cummings Publishing Company.

Berryman, J. G., 1999, Origin of Gassmann's equations:, *in* Soc. of Expl. Geophys., Geophysics, 1627–1629.

Biondi, B., Mavko, G., Mukerji, T., Rickett, J., Lumley, D., Deutsch, C., Gundeso, R., and Thiele, M., 1998, Reservoir monitoring: A multidisciplinary feasibility study: The Leading Edge, **17**, No. 10, 1404–1414.

Bishop, T. N., and Nunns, A. G., 1994, Correcting amplitude, time, and phase mis-ties in seismic data: Geophysics, **59**, No. 06, 946–953.

Bishop, C. M., 1995, Neural Networks for Pattern Recognition: Oxford University Press.

Bunch, A. W. H., and White, R. E., 1985a, Cross-Equalisation Of Seismic Traces: Geoexploration, , No. 23, 239–256.

——— 1985b, Least-squares filters without transient errors - An examination of the errors in least-squares filter design: Geophys. Prosp., **33**, No. 05, 657–673.

Burkhart, T., Hoover, A. R., and Flemings, P. B., 4 2000, Time-Lapse (4-D) Seismic Monitoring of Primary Production of Turbidite Reservoirs at South Timbalier Block 295, Offshore Louisiana, Gulf of Mexico: Geophysics, vol 65, NO. 2.

Burns, C. S., 1997, Integrating 3D seismic into the reservoir model, and its impact on reservoir management: Society of Petroleum Engineers, SPE, , No. 38996.

Burns, C. S., 1998, 4D Seismic Interpretive Processing: Optimizing differential imaging for reservoir monitoring: Society of Petroleum Engineers, SPE, , No. 50668, 329–343.

Campbell, C. J., 1997, The Coming Oil Crisis: Mult-Science Publishing CO.

Castagna, J. P., 2001, Recent advances in seismic lithologic analysis:, *in* Soc. of Expl. Geophys., Geophysics, 42–46.

Christie, P. A. F., 2000, Research and development for reservoir monitoring: A contractor's viewpoint: The Leading Edge, **19**, No. 03, 308–310.

Daubechies, I., 1992, Ten Lectures on Wavelets: Philadelphia: SIAM.

Davies, P., and Warner, M., 6 1999, Prediction of Multi-Channel Seismic Data Using Parallel Arrays of Neural Networks: EAGE Conference 61st.

Davies, P., 2000, Neural Network Prediction and Interpolation of Multi-channel seismic data: Ph.D. thesis, Imperial College.

Druzhinin, A., and MacBeth, C., 2001, A Cross-Matching Scheme for 3D Seismic Data: 63rd Conference and Technical Exhibition, EAGE, Amsterdam, Session: P202.

Duranti, L., 2001, Time-lapse, multicomponent seismic analysis of reservoir dynamics: Ph.D. thesis, Colorado School of Mines.

Ebrom, D., Krail, P., Ridyard, D., and Scott, L., 1998, 4-C/4-D at Teal South: The Leading Edge, **17**, No. 10, 1450–1453.

Elde, R., Haaland, M. A., Zachariassen, E., and Sandoe, I., 2000, Troll West–Time lapse processing and analysis: 70th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session: RC 2.5.

Erlandsen, S. M., 2000, Production Experience From Smart Wells in the Oseberg Field: Society of Petroleum Engineers, SPE, , No. 62953.

Essenreiter, R., Karrenbach, M., and Treitel, S., July 1998, Multiple reflection attenuation in seismic data using backpropagation: IEEE Transactions on signal processing, **46**, No. 7, 2001–2011.

Fahlman, S. E., and Lebiere, C., 1990, The Cascade-Correlation Learning Architecture *in* Touretzky, D. S., and Morgan-Kaufmann, Eds., Advances in Neural Information Processing Systems: Los Altos CA, 524–532.

Fahlmann, S. E., 1989, Faster-learning variations on back-propagation: an empirical study, *in* Los Altos CA, Proceedings of the 1988 connectionist models summer school. 3851.

Graps, A., 1995, An Introduction to Wavelets: IEEE Compu. Sci. Eng., **2**, No. 2.

Greaves, R. J., and Fulp, T. J., 1987, Three-dimensional seismic monitoring of an enhanced oil recovery process: Geophysics, **52**, No. 09. 1175 -1187.

Gringarten, A. C., 1998, Evolution of Reservoir Management Techniques: From Independent Methods to an Integrated Methodology. Impact on Petroleum Engineering Curriculum, Graduate Teaching and Competitive Advantage of Oil Companies: Society of Petroleum Engineers, SPE, , No. 39713.

Gurney, K., 1997, An Introduction to neural networks: UCL press.

Harris, P. E., and Henry, B., 1998, Time lapse processing: A North Sea case study: *in* 68th Ann. Internat. Mtg: Soc. of Expl. Geophys., 1–4.

Hatton, L., Worthington, M. H., and Makin, J., 1986, Seismic data processing: theory and practice: Blackwell Science.

Hecht-Nielsen, R., 1990, Neurocomputing: Addison-Wesley.

Hobbs, R., and Jakubowicz, H., 2000, Marine source signature measurement using a reference seismic source: *in* 62nd Mtg.: Eur. Assn. Geosci. Eng., Session:L0013.

Hornik, K., 1991, Approximation capabilities of multilayer feedforward networks: Neural Networks, **4**, 251257.

Huang, G.-B., and Babri, H. A., Jan 1998, Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions: IEEE Transactions on neural networks, **9**, No. 1, 224–229.

Huang, S.-C., and Huang, Y.-F., Jan 1991, Bounds on the number of hidden neurons in multilayer perceptrons: IEEE Transactions on neural networks, **2**, No. 1, 47–55.

Huang, X., Berni, A., and Jones, T., 1999, Reprocessing of time-lapse seismic data improves seismic history matching: 69th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1703–1706.

Hush, D. R., and Horne, B. G., January 1993, Progress in supervised neural networks: IEEE Signal Processing Magazine, **10**, No. 1, 8–39.

Hush, D. R., Slas, J. M., and Horne, B., 1991, Error surfaces for multi-layer perceptrons: IJCNN-91-Seattle International Joint Conference on Neural Networks, **1**, 759–764.

Hush, D. R., Horne, B., and Slas, J. M., 1992, Error surfaces for multi-layer perceptrons: IEEE Transactions on Systems, Man, and Cybernetics, **22**, No. 5, 1152–1161.

Hush, D. R., 1989, Classification with neural networks: a performance analysis: IEEE International Conference on Systems Engineering, pages 227–280.

Irie, B., and Miyake, S., 1988, Capabilities of three-layered Perceptrons: International Conference on Neural Nerworks, 641–648.

Jack, I. G., 1997, Time-Lapse Seismic in Reservoir Management: Soc. Expl. Geophys.

Jenkins, S. D., Waite, M. W., and Bee, M. F., 1997, Time-lapse monitoring of the Duri steamflood: A pilot and case study: The Leading Edge, **16**, No. 09, 1267–1273.

Johnsen, J. R., Rutledal, H., and Nilsen, D. E., 1995, Jurassic reservoirs: field examples from the Oseberg and Troll fields: Horda Platform area *in* Hanslien, S., Ed., Petroleum exploration and exploitation in Norway: NPF Special Publication: Norwegian Petroleum Society (NPF), 199–234.

Johnstad, S. E., and Ahmed, H., 1991, Application of the VSP technique in the central area of the Oseberg field: First Break, **9**, No. 08, 361–373.

Johnstad, S. E., Uden, R. C., and Dunlop, K. N. B., 1993, Seismic reservoir monitoring over the Oseberg field: First Break, **11**, No. 05, 177–185.

Johnstad, S. E., Seymour, R. H., and Smith, P. J., 1995, Seismic reservoir monitoring over the Oseberg field during the period 1989-1992: First Break, **13**, No. 05, 169–183.

Johnston, D., McKenny, R., Verbeek, J., and Almond, J., 1998, Time-lapse seismic analysis of Fulmar Field: The Leading Edge, **17**, No. 10, 1420, 1422–1426, 1428.

Johnston, D. H., Eastwood, J. E., Shyeh, J. J., Vauthrin, R., Khan, M., and Stanley, L. R., 2000, Using legacy seismic data in an integrated time-lapse study: Lena Field, Gulf of Mexico: The Leading Edge, **19**, No. 03, 294–302.

Johnston, D. H., 5 1997, A Tutorial on Time-Lapse Seismic Reservoir Monitoring: Offshore Technology Conference, OTC, , No. 8289.

Kaiser, G., 1994, A Friendly Guide to Wavelets: Birkhauser-Boston.

Knight, R. J., Dvorkin, J., and Nur, A., 1998, Acoustic signatures of partial saturation: Geophysics, **63**, No. 01, 132–138.

Koster, K., Gabriels, P., Hartung, M., Verbeek, J., Deinum, G., and Staples, R., 2000, Time-lapse seismic surveys in the North Sea and their business impact: The Leading Edge, **19**, No. 03, 286–293.

Kragh, E., and Christie, P., 2001, Seismic repeatability, normalized RMS and predictability: *in* 71st Ann. Internat. Mtg: Soc. of Expl. Geophys., 1656–1659.

Kragh, E., and Christie, P., 2002, Seismic repeatability, normalised rms, and predictability: The Leading Edge, **21**, No. 07, 640–647.

Landmark. **ProMAX 3D Reference Guide**. Landmark Graphic Corporation: Landmark - A Halliburton Company, 1997.

LeCun, Y., Matan, O., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D., and Baird, H. S., 1990a, Handwritten Zip Code Recognition with Multilayer Networks, *in* IEEE, Proc. of the International Conference on Pattern Recognition, 35–40.

—— 1990b, Optimal Brain Damage, *in* Morgan Kaufman, Advances in Neural Information Processing Systems 2 (NIPS*89).

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., November 1998, Gradient-Based Learning Applied to Document Recognition: Proceedings of the IEEE, **86**, No. 11, 2278–2324.

—— 1998b, Efficient BackProp, *in* Springer, Neural Networks: Tricks of the trade.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S., 1993, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function: Neural Networks, **6**, 861867,.

Lippmann, R. P., April 1987, An Introduction to Computing with Neural Nets: IEEE, ASSP Magazine, **4**, 422.

Liu, Z., and Castagna, J., 2000, Seismic wavelet estimation constrained by well data using artificial neural networks: 70th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session: ST P1.12.

Liu, Z., and Liu, J., 1998, Seismic-controlled nonlinear extrapolation of well parameters using neural networks: Geophysics, **63**, No. 06, 2035–2041.

Looney, C. G., 1997, Pattern Recognition Using Neural Networks: Theory and Algorithms for Engineers and Scientists: Oxford University Press.

Lumley, D. E., Behrens, R. A., and Wang, Z., 1997, Assessing the technical risk of a 4-D seismic project: The Leading Edge, **16**, No. 09, 1287–1291.

Lumley, D., Cole, S., Meadows, M., Tura, A., Hottman, W., Cornish, B., Curtis, M., and Maerefat, N., 2000, A risk analysis spreadsheet for both time-lapse VSP and 4-D seismic reservoir monitoring: 70th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session: RC 8.8.

Lumley, D. E., 1995, Seismic time-lapse monitoring of subsurface fluid flow: Ph.D. thesis, Stanford University.

Lumley, D. E., 2001, Time-lapse seismic reservoir monitoring: Geophysics, **66**, No. 01, 50–53.

Mallat, S. G., 1989, A Theory for Multiresolution Signal Decomposition: The Wavelet Representation: IEEE Transactions on Pattern Analysis and Machine Intelligence, **11**, No. 07, 674–693.

Mallat, S. G., 1996, Wavelets for a vision: Proceedings of the IEEE, **84**, No. 04, 604 –614.

Mavko, G., and Mukerji, T., 1998, Bounds on low-frequency seismic velocities in partially saturated rocks:, *in* Soc. of Expl. Geophys., Geophysics, 918–924.

Mavko, G., Mukerji, T., and Dvorkin, J., 1998, The rock physics handbook: Cambridge University Press.

Monk, D. J., 1993, Wave-equation multiple suppression using constrained gross-equalization: Geophysical Prospecting, **41**, 725–736.

Nur, A., 1989, Four-dimensional seismology and (true) direct detection of hydrocarbons: The petrophysical basis: The Leading Edge, **08**, No. 09, 30–36.

Pennington, W. D., 1997, Seismic petrophysics: An applied science for reservoir geophysics: The Leading Edge, **16**, No. 03, 241–244.

Pennington, W., 2001, Reservoir geophysics:, *in* Soc. of Expl. Geophys., Geophysics, 25–30.

Poulton, M. M., Sternberg, B. K., and Glass, C. E., 1992, Location of subsurface targets in geophysical data using neural networks: Geophysics. **57**, No. 12. 1534–1544.

Pratt, R. G., 1999, Seismic waveform inversion in the frequency domain, Part 1: Theory and verification in a physical scale model:, *in* Soc. of Expl. Geophys., Geophysics, 888–901.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., 1997, Numerical Recipes in C: The Art of Scientific Computing: Cambridge University Press.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., 2002, Numerical Recipes in C++: The Art of Scientific Computing: Cambridge University Press.

Rickett, J., and Lumley, D. E., 1998, A cross-equalization processing flow for off-the-shelf 4-D seismic data: *in* 68th Ann. Internat. Mtg: Soc. of Expl. Geophys., 16–19.

Rickett, J., and Lumley, D. E., 2001, Cross-equalization data processing for time-lapse seismic reservoir monitoring: A case study from the Gulf of Mexico:, *in* Soc. of Expl. Geophys., Geophysics, 1015–1025.

Riedmiller, M., and Braun, H., 1992, RPROP — A fast adaptive learning algorithm: Proceedings of 7th Int. Symp. on Comp. and Inf. Science (ISCIS VII).

Riedmiller, M., and Braun, H., 1993, A Direct Adaptive Method For Faster Backpropagation Learning: The Rprop Algorithm: IEEE, pages 586–591.

Robertson, J. D., 1989, Reservoir management using 3-D seismic data: The Leading Edge, **08**, No. 02, 25–31.

Ross, C. P., and Altan, M. S., 1997, Time-lapse seismic monitoring: Some shortcomings in nonuniform processing: The Leading Edge. **16**, No. 06, 931–937.

Ross, C. P., Cunningham, G. B., and Weber, D. P., 1996, Inside the crossequalization black box: The Leading Edge, **15**, No. 11, 1233–1240.

Roy-Chowdhury, K., Devilee, R., and Curtis, A., 1997, Use of neural networks for non-linear non-unique inversion in geophysics: 67th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1620–1623.

Ryseth, A., Fjellbirkeland, H., Osmundsen, I., Skalnes, A., and Zachariassen, E., 1998, High Resolution Stratigraphy and Seismic Attribute Mapping of a Fluvial Reservoir: Middle Jurassic Ness Formation, Oseberg Field: AAPG Bulletin, **82**, No. 9, 1627–1651.

Satter, A., and Thakur, G. C., 1994, Integrated Petroleum Reservoir Manegement. A Team Approach: PennWell Books, Tulsa, Oklahoma.

Schiffmann, W., Joost, M., and Werner, R., Optimization of the Backpropagation Algorithm for Training Multilayer Perceptrons: Technical report, University of Koblenz, 1992.

Schiffmann, W., Joost, M., and Werner, R., Optimization of the Backpropagation Algorithm for Training Multilayer Perceptrons: Technical report, University of Koblenz, Sept. 1994.

Sheriff, R. E., 1997, Encyclopedic dictionary of exploration geophysics: Soc. Expl. Geophys.

Sognesand, S., 1997, Reservoir management of the Oseberg Field during eight years' production: Society of Petroleum Engineers, SPE, , No. 38555.

Sonneland, L., Veire, H. H., Raymond, B., Signer, C., Pedersen, L., Ryan, S., and Sayers, C., 1997, Seismic reservoir monitoring on Gullfaks: The Leading Edge, **16**, No. 09, 1247–1252.

Spitz, S., 1991, Seismic trace interpolation in the F-X domain: Geophysics, **56**, No. 06, 785–794.

Stolt, R. H., 1978, Migration by Fourier transform: Geophysics, **43**, No. 01, 23–48.

Stroustrup, B., 2000, The C++ Programming Language (Special Edition): Addison-Wesley.

Taner, M. T., Koehler, F., and Sheriff, R. E., 1979, Complex seismic trace analysis: Geophysics, **44**, No. 06, 1041–1063.

Tanja Oldenziel, P. d. G., and Kvamme, L. B., 2 2000, Statfjord Study Demonstrates Use of Neural Network to Predict Porosity and Water Saturation from Time Lapse Seismic: first break.

Thomas, R., 1999, Quantitative controls on the quality of pre-stack seismic data attribute analysis: Ph.D. thesis, Birkbeck College, University of London.

Tollenaere, T., 1990, SuperSAB: fast adaptive back propagation with good scaling properties: Neural Networks, **3**, No. 5, 561–573.

van der Baan, M., and Jutten, C., 2000, Neural networks in geophysical applications:. *in* Soc. of Expl. Geophys., Geophysics, 1032–1047.

Vapnik, V. N., and Chervonenkis, A. Y., 1971, On the Uniform Convergence of Relative Frequencies of Events to their Probabilities: Theory of Probability and Applications, **16**, No. 2, 264–280.

Vidakovic, B., and Muller, P. Wavelets for Kids: A Tutorial Introduction:. Duke University: available at ftp, 1994.

Walden, A. T., and White, R. E., 1984, On Errors of fit and accuracy in matching synthetic seismograms and seismic traces: Geophysical Prospecting, **32**, 871–891.

Walden, A. T., and White, R. E., 1998, Seismic wavelet estimation: a frequency domain solution to a geophysical noisy input-output problem: IEEE Transactions on geoscience and remote sensing, **36**, No. 1, 287–297.

Wang, L. X., and Mendel, J. M., 1992, Adaptive minimum prediction-error deconvolution and source wavelet estimation, using Hopfield neural networks:, *in* Soc. of Expl. Geophys., Geophysics, 670–679.

Wang, Z., Hirsche, W. K., and Sedgwick, G., 1991, Seismic monitoring of water floods? A petrophysical study: Geophysics, **56**, No. 10, 1614–1623.

Wang, Z., Cates, M. E., and Langan, R. T., 1998, Seismic monitoring of a CO2 flood in a carbonate reservoir: A rock physics study:, *in* Soc. of Expl. Geophys., Geophysics, 1604–1617.

Wang, Z., 1997, Feasibility of time-lapse seismic reservoir monitoring: The physical basis: The Leading Edge, **16**, No. 09, 1327–1329.

Wang, Z., 2001, Fundamentals of seismic rock physics:, *in* Soc. of Expl. Geophys., Geophysics, 398–412.

Werbos, P. J., 1988, Backpropagation: Past and future:, *in* IEEE, Proceedings of the IEEE International Conference on Neural Networks (ICNN), 343–353.

White, R. E., 1980, Partial coherence matching of synthetic seismograms with seismic traces: Geophysical Prospecting, **28**, 333–358.

Wolberg, G., 1996, Recent advances in image morphing: IEEE Computer Graphics International, pages 64–71.

Yilmaz, O., 1987, Seismic Data Processing: *in* Doherty, S. M., Ed., Seismic Data Processing: Soc. of Expl. Geophys., 526.

Yilmaz, O., 2000, Seismic Data Analysis: *in* CooperM. R.; Doherty, S. M., Ed., Seismic Data Analysis Vol. 1: Soc. of Expl. Geophys., 01, 1000.

Zell, A., Mamier, G., Vogt, M., Mache, N., Hübner, R., Döring, S., Herrmann, K.-U., Soyez, T., Schmalzl, M., Sommer, T., Hatzigeorgiou, A., Posselt, D., Schreiner, T.,

Kett, B., Clemente, G., Wieland, J., and Gatter, J. **SNNS, Stuttgart Neural Network Simulator**. University Of Stuttgart, Germany, 2000.

LIBR. LONDON. UNIV.