InSAR Phase Unwrapping Using Deep Learning

Eilish Rhiannon O'Grady

Case Partner: Slingshot Simulations

Funded as part of the DEEPVOLC project

Submitted in accordance with the requirements for the degree of Doctor of Philosophy

The University of Leeds

School of Earth and Environment

Institute of Geophysics and Tectonics

November 2025

Declaration

I confirm that the work submitted is my own and that appropriate credit has been given where reference has been made to the work of others.

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

Signed:
Eilish O'Grady

Acknowledgements

And thus, we have reached the end of a (slightly longer than expected) journey. There are many people I need to thank who supported me to reach this point, and without them, I don't think I could have got here.

Firstly, thank you to my supervisors Professor Andy Hooper and Professor David Hogg, who have provided a wealth of support, guidance, teaching and relentless optimism dependent on what I needed at the time. Thank you to everyone within DeepVolc, both past and present, who could be relied upon for encouragement, feedback and to answer any questions I had. I would like to thank everyone in the SENSE CDT, the training and trips have been amazing and the initial learning curve of a geodesy PhD would have been a lot harder without it. The community within SENSE has meant that at any conference I attend, there has always been a friendly face to talk to, and someone to get a confidence boost from. Thanks to all current and past members of IGT, particularly Jess and Rachel, who were there whether I needed a question answered, a chat (distraction), or a parkrun and coffee shop visit.

Thank you to Harry, for being there for a chat or Percy Jackson dependent on the requirement - we've both made it. Nan and Grandad, thank you for always being happy to have me run away to yours for multiple weekends. Thanks to Rosie, Rosa and Mark, for being supportive and a walking, running and tea buddies.

Finally, Mum, Dad, Niamh and Callum (and Ollie), your support over these past years has been immeasurable - I don't think the words exist to express my gratitude and there

is no way I would be where I am without you.

LiCSAR contains modified Copernicus Sentinel data analysed by the Centre for the Observation and Modelling of Earthquakes, Volcanoes and Tectonics (COMET). LiCSAR uses JASMIN, the UK's collaborative data analysis environment (http://jasmin.ac.uk)

I acknowledge the use of ChatGPT-5 'https://chatgpt.com/' to aid code debugging and speed the development of utility code. This has included code to plot discrete values on figures, debugging and the Tversky loss code.

Abstract

Given the importance of monitoring natural hazards, Interferometric Synthetic Aperture Radar (InSAR) has proven to be a powerful tool to monitor, at high spatial and temporal resolution, deformation across the Earth's surface (Lazecký et al. 2020). Wrapped interferograms record phase as modulo 2π, introducing ambiguity in the relative displacement; thus phase unwrapping is required using

$$\phi = \psi + 2\pi k, \tag{0.1}$$

where ϕ is unwrapped phase, ψ is wrapped phase and k is an integer wrap count value. Given the ill posed nature of phase unwrapping (Itoh 1982), generally it is assumed that the absolute difference in phase gradient between neighbouring pixels will not exceed π. However, this is not always true; to identify residues, the summed difference of a 2x2 pixel square is calculated, and where not equal to 0 (Goldstein et al. 1988). During phase unwrapping, often the goal is to identify an optimal integration pathway, which minimises the difference between the wrapped and unwrapped phase gradients through an Lp norm optimisation (Ghiglia et al. 1996):

$$minimise \left\{ \sum_{i,j} w^x_{i,j} |\Delta\phi^x_{i,j} - \Delta\psi^x_{i,j}|^p + \sum_{i,j} w^y_{i,j} |\Delta\phi^y_{i,j} - \Delta\psi^y_{i,j}|^p \right\}. \tag{0.2}$$

Here, $\Delta\phi^x$ and $\Delta\psi^x$ are the horizontal components over all positions i, j of the unwrapped

and wrapped phase gradients respectively, $\Delta\phi^y$ and $\Delta\psi^y$ are the respective components in the vertical direction and $w^x_{i,j}$ and $w^y_{i,j}$ are weightings applied to each direction (Ghiglia et al. 1996). Phase unwrapping is most difficult in regions of high noise (Chen 2001) and where there has been insufficient sampling relative to the amount of deformation (Kester 2009) due to the high degree of uncertainty associated with the wrapped phase gradient and thus, within these regions, phase unwrapping errors are most likely to be introduced.

Phase unwrapping remains an active research field (Mu et al. 2023; Pepin et al. 2024; Zhou et al. 2022a). Recently, the focus has shifted to utilising deep learning models for phase unwrapping, particularly focused upon increasing phase unwrapping robustness in high phase noise regions and, given its importance to optimising the integration pathway, the correct identification of phase gradients (Sica et al. 2022; Wang et al. 2021). A popular approach is for trained models to classify phase gradients into 3 classes.

In this thesis, I present work which explores the use of deep learning to improve the unwrapped phase result by reducing phase unwrapping errors associated with large phase gradients, high phase noise and isolated regions of coherent pixels. This work builds upon previous publications proposals which classify gradients as 0 change, +1 or more and -1 or less wrap count difference. This does not allow the differentiation of absolute gradients exceeding 1 wrap count.

I begin by presenting a complete unwrapping approach called Multi-Class Greedy Phase Integrator (MC-GI). A semantic segmentation U-Net model is trained to include at least one additional phase gradient class to allow the explicit identification of larger gradients and a novel Greedy Phase Integrator is used to integrate the gradient maps, prioritising those gradients where the model has a high degree of certainty in its prediction. Though the MC-GI has good performance on synthetic interferograms, and interferograms with high fringe rates but low noise, the robustness of unwrapping in the presence of high noise was poor. Furthermore, the isolation of coherent regions led to local accuracy in the unwrapped phase result but global unwrapping errors overall due to an offset in wrap count between isolated regions.

I then present a Multi-Class Cost Guided Phase Integrator approach (MC-CGI) with the aim of improving robustness of unwrapping in high phase noise regions whilst retaining the advantage of identifying large phase gradients. I train a semantic segmentation model using a training data set which includes circular Gaussian noise. Gradient maps are integrated using the cost-guided optimisation method SNAPHU (Chen et al. 2001), where the wrap count gradient and model prediction certainty are used to inform the offset and variance respectively a priori. Results indicate an improved unwrapping performance for synthetic and real Sentinel-1 data, compared to applying SNAPHU alone. Whereas for synthetic data, a MC[4]-CGI (4 class) approach performed best, for real data, the success of the approach was not reliant on the number of classes chosen, with a 3 class, 4 class and 5 class model performing equally well.

To minimise the unwrapping error associated with isolation of coherent regions, I explore the impact of using a Brownian Bridge Diffusion Model (Bo et al. 2023) to gap fill masked regions with a realistic data distribution. Qualitatively, results for mainly atmospheric gaps are promising where few signals are associated with deformation, however either model fine-tuning or retraining using a data set containing signal examples of ringed deformation, often associated with earthquakes, mountainous regions and volcanic deformation, is required to increase the generalisability of the BBDM. With respect to phase unwrapping, the gap filling does reduce the phase error associated with isolated regions, however it success is dependent on the signal type of signal and mask size, meaning whilst initial results are promising, further work is required to increase the methods reliability and generalisability.

Contents

1 Introduction **1**

1.1 Synthetic Aperture Radar 2

1.2 Interferometric Synthetic Aperture Radar 5

1.2.1 Coherence 7

1.2.2 Unwrapping Error Sources 9

1.3 Classical Phase Unwrapping Techniques 11

1.3.1 SNAPHU 12

1.4 Deep Learning 16

1.5 Deep Learning Phase Unwrapping Techniques 19

1.5.1 One-step Phase Unwrapping 19

1.5.2 Multi-step Phase Unwrapping 22

1.6 Motivation for work and the aims 33

2 Multi-Class Greedy Phase Integrator **39**

2.1 Introduction 39

2.2 Method: Multi-Class Gradient 42
2.2.1 Model Architecture 43
2.2.2 Loss Function 47
2.2.3 Generation of Training Dataset 49
2.2.4 Noise Generation 52
2.3 Method: Greedy Integration 53
2.4 Results and Discussion 58
2.4.1 Qualitative Assessment of the MC[4]-GI Gradient Estimation . . . 58
2.4.2 Quantitive Assessment of MC-GI for Wrap Count Gradient Prediction on Synthetic Data 61
2.4.3 Assessment of the Full MC-GI Process 66
2.4.4 Further Assessment of the Complete MC[4]-GI approach on Synthetic Interferograms 73
2.4.5 Loop Closure Error Analysis Explanation 78
2.4.6 Assessment of MC-GI unwrapping performance on Sentinel-1 Interferograms 79
2.5 Conclusion 87

3 A Noisy Problem Part 1: Gradient Classification 91

3.1 Introduction 91
3.2 Methods 93

101 3.2.1 Noise Generation 93

102 3.2.2 Model Training 97

103 3.2.3 Loss Functions 99

104 3.2.4 Datasets 100

105 3.3 Results and Discussion 101

106 3.3.1 MC[4]-CGI Gradient Classification Performance 101

107 3.4 Conclusion 106

108 **4 A Noisy Problem Part 2: The Integration Step** **109**

109 4.1 Introduction 110

110 4.1.1 SNAPHU 111

111 4.2 Methods 112

112 4.2.1 Unwrapping Process 112

113 4.2.2 Cost file preparation using wrap count gradient estimations 113

114 4.3 Results and Discussion 119

115 4.3.1 Testing Unwrapping using Synthetic Interferograms 119

116 4.3.2 Testing Unwrapping using Sentinel-1 Interferograms 134

117 4.4 Conclusions 139

118 **5 Please Mind the Gap** **141**

119 5.1 Introduction 141

120 5.2 Method 146

121 5.2.1 Dataset Preparation 146

122 5.2.2 Model Architecture and Training 148

123 5.3 Results and Discussion 150

124 5.3.1 Testing on a Test Data Set with a Groundtruth 150

125 5.3.2 Pixel-Wise Analysis of the Confidence in BBDM Prediction 156

126 5.3.3 Gap-filling larger interferograms 156

127 5.3.4 The Impact of Gap Filling on Phase Unwrapping 163

128 5.4 Conclusion 176

129 **6 Conclusions** **178**

130 6.1 Summary of Research Outcomes 178

131 6.2 Broader Summary 181

132 6.3 Future Avenues 183

List of Figures

1.1 How SAR satellites work 3

1.2 Illustration of the impact of surface roughness on the microwave reflection 4

1.3 Illustration of the side-looking nature of satellites. 4

1.5 Examples of wrapped and unwrapped interferograms, where phase unwrapping has occured. 10

1.6 Diagram illustrating aliasing. 11

1.7 Reproduction of a schematic diagram showing normalised costs against offset from Chen (2001). 13

1.8 Reproduced diagram from Chen (2001) showing an example cost function for a phase gradient. 15

1.9 Schematic of a neural network. 17

1.10 Schematic of convolutional filtering. 18

1.11 Flowchart of the link between thesis Chapters. 36

2.1 Diagram of the model architecture. 43

148 2.2 Examples of synthetic wrapped and unwrapped sill, dyke, mogi and quake
149 deformation patterns. 50

150 2.3 Example synthetic training data. 51

151 2.4 Wrap count return algorithm flowchart. 53

152 2.5 Multiple stages of the wrap count return algorithm. 57

153 2.6 Examples of the noiseless synthetic wrapped interferogram model input
154 and the corresponding model prediction accuracy. 59

155 2.7 Examples of synthetic wrapped interferogram model input with varying
156 degrees of noise and the corresponding model prediction accuracy. 60

157 2.8 Confusion matrixes for MC[4]-GI and SNAPHU when models were tested
158 on the entire synthetic test set. 64

159 2.9 Mean Interception over Union per coherence for models MC[4]-GI and
160 SNAPHU. 65

161 2.10 Root mean squared error against coherence for MC[3]-GI, MC[4]-GI, MC[5]-
162 GI and SNAPHU. 68

163 2.11 Root mean square error plotted against coherence for interferograms where
164 larger jumps occur for MC[3]-GI, MC[4]-GI, MC[5]-GI and SNAPHU. . . 68

165 2.12 Examples of synthetic wrapped interferograms where larger jumps (exceed-
166 ing ± 1) are present. 69

167 2.13 Root mean square error against coherence for examples of synthetic wrapped
168 interferograms where larger jumps (exceeding ± 1) are present. 70

2.14 Example of wrapped and unwrapped synthetic interferogram, with similar unwrapping performance for SNAPHU, MC[3]-GI, MC[4]-GI, MC[5]-GI approaches. 71

2.15 Example of a synthetic unwrapped interferogram with large noise regions, returned wrap count and accuracy using the MC[4]-GI and SNAPHU approaches. 75

2.16 Example of a synthetic unwrapped interferogram with dense fringes, returned wrap count and accuracy using the MC[4]-GI and SNAPHU approaches. 76

2.17 Example of a synthetic unwrapped interferogram with isolated coherent regions, returned wrap count and accuracy using the MC[4]-GI and SNAPHU approaches. 77

2.18 Wrap count results and loop closure error for Frame 172A 04842 091010, acquisition dates 2024/01/19, 2024/02/12 and 2024/02/24, returned by MC[4]-GI and SNAPHU. 81

2.19 Wrap count results and loop closure error for Frame 128A 05172 131313, acquisition dates 05/01/2022, 17/01/2022 and 29/01/2022, returned by MC[4]-GI and SNAPHU. 82

2.20 Wrap count results and loop closure error for Frame 128A 05172 131313, acquisition dates 05/01/2022, 17/01/2022 and 29/01/2022, returned by MC[4]-GI and SNAPHU, over the deformation area. 83

2.21 The horizontal and vertical MC[4]-GI gradient output over the deformation region of Frame 128A 05172 131313. 84

192 2.22 MC[4]-GI and SNAPHU unwrapped interferograms and loop closure error
193 for Frame 140D SM FGbr s4, acqusitions dated 2015/06/12, 2015/09/16
194 and 2016/03/02. 85

195 2.23 Horizontal and vertical gradient output returned by the MC[4]-GI model
196 over the deformation area of Frame 140D SM FGbr s4 test 86

197 3.1 Example of filtering impacting the MC[4]-GI unwrapped result. 92

198 3.2 Diagram of circular gaussian noise added to phase. 94

199 3.3 Example of circular gaussian noise added to a wrapped interferogram. . . 95

200 3.4 Distribution of data for Chapter 2, Chapters' 3 and 4 and real data. . . . 97

201 3.5 Schematic diagram of the U-Net model architecture. 98

202 3.6 Diagram of the attention-gate. 98

203 3.7 Mean interception over union for various 4 class models when predicting
204 vertical and horizontal gradients. 104

205 3.8 Confusion matrices of the best performing 4 class model and SNAPHU. . 105

206 3.9 Examples of wrap count gradient predictions made by MC[4]-CGI in a high
207 noise and low noise scenario. 108

208 4.1 Simple example case illustrating the requirement for using the suggested
209 approach to return unwrapped phase. 116

210 4.2 Example of wrapped and unwrapped interferogram using SNAPHU and
211 MC-CGI approach. 117

212 4.3 Root mean square error at varying coherence levels for MC[3]-CGI, MC[4]-
213 CGI and MC[5]-CGI. 121

214 4.4 Example of wrapped and unwrapped synthetic interferogram pairs using
215 SNAPHU, MC[3]-CGI, MC[4]-CGI and MC[5]-CGI to unwrap. 122

216 4.5 Examples of unwrapped synthetic interferograms using SNAPHU, MC[3]-
217 CGI, MC[4]-CGI and MC[5]-CGI. 123

218 4.6 Example of wrapped and unwrapped phase, where using a 4 class model
219 has improved the unwrapped result. 125

220 4.7 Gradient maps used in the cost files to unwrap interferograms in Figure 4.6.126

221 4.8 Unwrapped phase using the greedy algorithm and MC[4]-CGI applying
222 Chapter 2 model (high noise). 128

223 4.9 RMSE of MC[4]-GI original setup and SNAPHU at varying coherence levels.129

224 4.10 Unwrapped phase using the greedy algorithm and MC[4]-CGI applying
225 Chapter 2 model (low noise). 130

226 4.11 RMSE of MC[4]-GI and MC[4]-CGI unwrapping performance across vary-
227 ing coherence when the best Chapter 3 model is used 131

228 4.12 Unwrapped phase using the greedy algorithm and MC[4]-CGI 132

229 4.13 Unwrapped phase using the greedy algorithm and MC[4]-CGI 133

230 4.14 Comparison of unwrapping results between SNAPHU and SNAPHU cost
231 guided by the classification map from the 3,4 and 5-class models for LiC-
232 SAR Frame 123D 05292 242525 test. 135

233 4.15 Figure showing the vertical and horizontal wrap count gradients for MC[3]-
234 CGI, MC[4]-CGI and MC[5]-CGI for the boxed regions of Figure 4.14. . . 136

235 4.16 Example where the MC-CGI approach has led to improved phase un-
236 wrapped result compared with using SNAPHU alone in noisy regions. . . 138

237 5.1 Examples of incorrectly unwrapped interferograms caused by the isolation
238 of coherent regions. 144

239 5.2 Examples of good inpainting results. 151

240 5.3 Violin plots of Mean Squared Error, Local SSIM, Mean Absolute Error
241 and Peak Signal to Noise Ratio. 154

242 5.4 Examples of inpainted results with low PSNR scores or low SSIM scores. 155

243 5.5 Example of good inpainting. 157

244 5.6 Example where high masking has led to poor inpainted results. 158

245 5.7 Example where poor inpainting is likely resultant from dissimilarity be-
246 tween the test and training data. 159

247 5.8 Example of inpainting result for Frame 062A 13470 13131, dated 2022/03/25
248 - 2022/04/06. 159

249 5.9 Example of inpainted result for LiCSAR Frame 062A 13470 131313 reduc-
250 ing phase unwrapping errors due to isolation of coherent regions. 160

251 5.10 A comparison of the unwrapped result when using the inpainted result
252 of Frame 128D 09016 110500, and when not using the inpainted wrapped
253 interferogram. 162

254 5.11 Example of inpainting used to reduce the loop closure error caused by
255 isolated coherent regions for LiCSAR Frame 006D 07728 131313. 164

256 5.12 Inpainted wrapped phase. 165

257 5.13 Figure demonstrating the improvement in unwrapping when using inpaint-
258 ing applied to LicSAR Frame 140D SM FGbr s4 test. 166

259 5.14 Example of gap filling being applied to reduce loop closure error. 167

5.15 Example of poor MC[4]-CGI result when unwrapping the inpainted wrapped interferogram. 170

5.16 Boxplots of the RMSE distribution for 46 unwrapped interferograms by the nearest neighbour (NN) and BBDM (Inpaint) approach. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap. 171

5.17 Inpainting using nearest neighbour and BBDM approach 172

5.18 Inpainting using nearest neighbour and BBDM approach 173

5.19 Inpainting using nearest neighbour and BBDM approach 174

5.20 Inpainting using nearest neighbour and BBDM approach 175

List of Tables

2.1 Table of model setup details 45

2.2 Details of unchanged parameters used to train the model 46

2.3 Table showing the classes and their meanings 46

2.4 Table containing the Macro-MIoU score for SNAPHU and MC[4]-GI. . . 63

2.5 Table containing the statistical scores Macro-Precision, Macro-Recall, and Macro-F1 SNAPHU and MC[4]-GI. 63

2.6 Table of frame id and time example 79

3.1 Table showing the MC[4]-CGI model results 102

3.2 Table showing the MC[3]-CGI model results - included for completeness . 102

3.3 Table showing the MC[5]-CGI model results - included for completeness . 102

4.1 MC[4 127

5.1 Table of parameters used to train the BBDM. 148

5.2 Mean values of Mean Squared Error, Local SSIM, Mean Absolute Error, and Peak Signal to Noise Ratio over the testing dataset. 153

Abbreviations

BBDM Brownian Bridge Diffusion Model

CNN Convolutional Neural Network

DDPM Denoising Diffusion Probabilistic Model

FN/FP False Negative/ False Positive

TN/TP True Negative/True Positive

GI Greedy Phase Integrator

GANS Generative Adversarial Network

InSAR Interferometric Synthetic Aperture Radar

MAE Mean Absolute Error

MC-CGI Multi-Class Cost-Guided Phase Integrator

MC-GI Multi-Class Greedy Phase Integrator

MCF Minimum Cost Flow

MIoU mean intersection over Union

MSE Mean Square Error

MST Minimum Spanning Tree

PSNR Peak Signal to Noise Ratio

RMSE Root Mean Square Error

SAR Synthetic Aperture Radar

SNAPHU Statistical-cost network-flow algorithms for Phase Unwrapping

SNR Signal to Noise Ratio

SSIM Structured Similarity Index Measure

rad Radians

Chapter 1

Introduction

At approximately 510 million km^2, the Earth's surface area, defined as the boundary where solid earth and oceans meet the atmosphere, is vast. On Earth, there are at least 1500 active volcanoes (British Geological Survey 2024), 50 - 70 of which erupt per year. Today, an estimated 800 million people live within 100 km of an active volcano (Calvari et al. 2022) despite the associated hazards. This may be due to the possible benefits, such as nutrient rich soil for growing crops, volcano tourism and the potential exploitation of geothermal energy, which all provide possible income sources for locals to capitalise upon. The monitoring of volcanoes to anticipate, assess and reduce the potential consequences of an eruption is therefore essential (Calvari et al. 2022; Bountos et al. 2022). Detection and monitoring of such geohazards, combined with the need to monitor environmental changes associated with anthropogenic behaviours, such as deforestation, mining (Wu et al. 2021b) or water extraction (Payne et al. 2025), means any regular, large-scale monitoring is an incredibly daunting task.

The advent of remote sensing has revolutionised our ability to collect vast amounts of data at spatial and temporal resolutions that were unimaginable a century ago. A key resource in the European Union's Earth Observation geodesy capabilities is the Sentinel-1 radar satellites, which are used for Interferometric Synthetic Aperture Radar (InSAR) monitoring (Lazecký et al. 2020). Sentinel-1a was launched in April 2014, with Sentinel-

1B and 1C launched in April 2016 and December 2024 respectively, meaning there is now over a decade of Synthetic Aperture Radar (SAR) data freely accessible (Copernicus SentiWiki 2026). The Sentinel-1 constellation originally consisted of Sentinel-1A and 1B offset at 180°, with acquisitions over the same footprint area taken on average every 6 days and 175 orbits undertaken per cycle. The successful launch of Sentinel-1C following a malfunction of Sentinel-1B in December 2021 ensures that the Earth's surface is still monitored, with SAR data being regularly acquired and made available.

1.1 Synthetic Aperture Radar

Onboard each Sentinel-1 satellite is a C-band synthetic aperture radar (SAR) instrument (Copernicus SentiWiki 2026). Unlike passive sensors such as the optical sensor onboard Sentinel 2, radar does not rely on reflected wavelengths within the visible region of the electromagnetic spectrum. Instead, SAR are active instruments; electromagnetic wave pulses in the microwave region of the spectrum are emitted (Figure 1.1), and the phase and amplitude of the reflected backscatter are recorded, corresponding to the strength and fraction of one complete phase cycle (Balmer et al. 1998). This means the technique has the benefit that radar can be used in cloudy regions and at night. The recorded phase and amplitude of the backscatter are related to the surface properties of the objects that radar waves are reflected from, as illustrated in Figure 1.2. A rougher surface will cause a greater degree of scattering, with a larger amount of signal returning to the satellite than for a smooth surface, that may reflect the signal away from the satellite. Human made structures and multi-layered structures will normally result in double bounce and multiple interactions respectively, leading to a bright return signals for both (Copernicus SentiWiki 2026).

Sentinel-1 satellites are side-looking which is essential for generating SAR imagery. Consider the case in which a radar pulse is transmitted. If the radar beam is directed in the nadir-geometry (straight downwards), two targets located equidistant from the satellite, but positioned at different points within the swath, will reflect signals that return simultaneously to the satellite. As the two signals are returned simultaneously, the two targets are indistinguishable, as shown by Figure 1.3(a). In contrast, side-looking radar introduces a delay in the return time for the same two targets which are now positioned at different distances within the swath relative to the satellite. The delay is proportional to the across-track distance across the swath, and consequently means that the two points are now distinguishable.

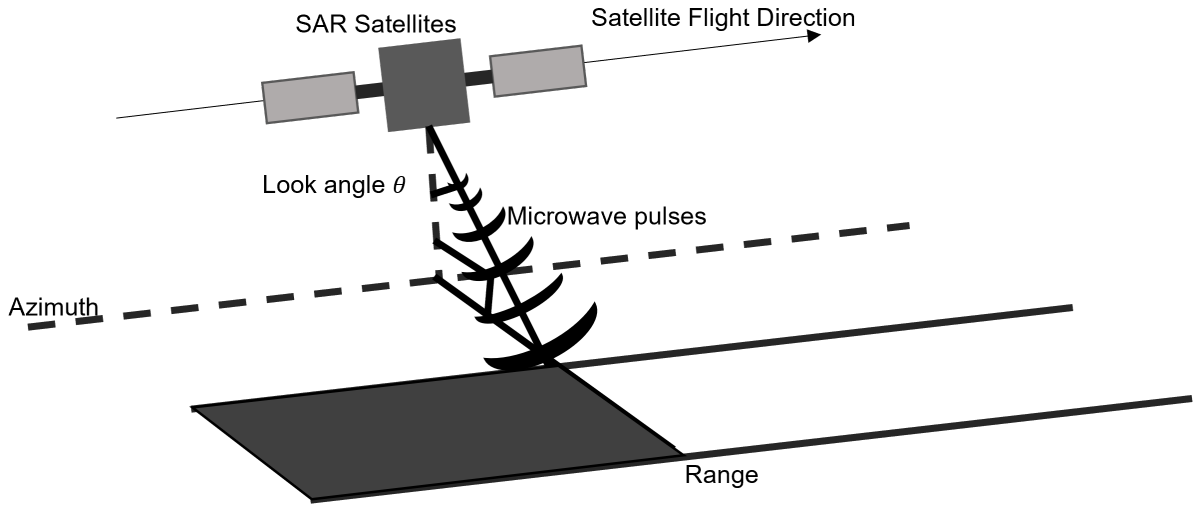


Figure 1.1 – Diagram illustrating how SAR satellites work. The side-looking satellite travels in the flight direction, emitting microwave pulses at a pulse repetition frequency (PRF) to cover a ground footprint. The microwave pulses are reflected from surfaces, and the echoes are recorded by the satellite. Multiple acquisitions are combined to form SAR images. The look angle is the angle at which the microwave pulses are sent relative to nadir (the vertical direction underneath the satellite).

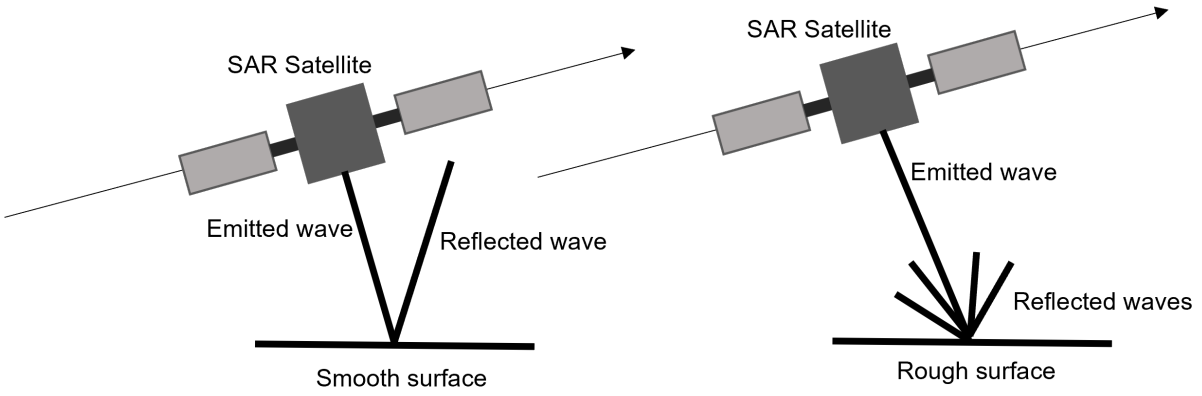


Figure 1.2 – Diagram illustrating the impact of surface roughness on the microwave reflection. (Left) When the surface is smooth, specular reflection occurs. The wave is reflected at the same angle from the normal as the angle of incidence, which means that the satellite will not be in a suitable position to record the reflection. (Right) When the surface is rough, diffuse reflection leads to scattering of the returned signal, with a portion of energy scattered in the direction of the satellite.

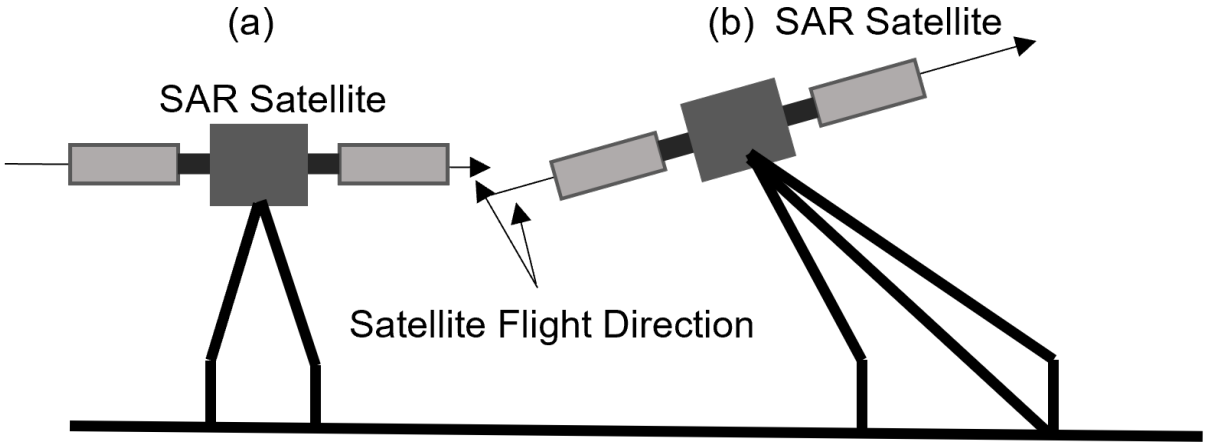


Figure 1.3 – Illustration of how side-looking geometry provides range resolution compared to nadir-looking geometry. (a) For a satellite emitting radar in the nadir direction, signals from two objects at equal range distance within the swath return simultaneously, making the objects indistinguishable. (b) For a side-looking satellite, signals from laterally separated objects return at different times, allowing them to be differentiated.

1.2 Interferometric Synthetic Aperture Radar

SAR data is useful for monitoring ground deformation using the technique Interferometric Synthetic Aperture Radar (InSAR). Since InSAR was first used by Gabriel et al. 1989 to detect small temporal changes in surface elevation, it has been used to measure ground motion across the planet, from subsidence related to water extraction (Payne et al. 2025) to analysing earthquake events (Zhou et al. 2020), monitoring mining activities (Gao et al. 2024) and volcanoes (Lazecký et al. 2020). This is possible because pixels are temporally correlated, though the phase of an individual pixel is statistically independent from adjacent pixels (Chen 2001). The returned signal's phase and amplitude remains temporally stable provided that the scattering mechanisms, such as surface scattering properties, remain consistent between acquisitions; for example, a hard, man-made surface such as concrete buildings generally remains in the same place, and retains the same scattering properties between the two acquisitions taken 6 days apart and will thus have a stable phase shift in the return signal overtime, which can allow the detection of surface displacements on a millimetre scale. Where temporal decorrelation occurs, caused by factors such as vegetation growth, changes in soil moisture or snow cover, the reliability of the interferometric measurements may be reduced (Ma et al. 2022).

To prepare an interferogram, the SAR single look complex (SLC) images are co-registered to a sub-pixel level to ensure each pair of SAR image pixels represents the same area (Marinkovic et al. 2004). Following co-registration, the complex valued pixels in the reference image are multiplied by the complex conjugate values of the secondary image, giving the differences in phase and amplitude over time

$$s_1 s_2^* = |s_1||s_2| \exp(j(\phi_1 - \phi_2)). \tag{1.1}$$

Here, s_1 and s_2 are the complex values of corresponding pixels from the reference and secondary images, * denotes the complex conjugation, j = $\sqrt{-1}$ and ϕ_1 and ϕ_2 are the

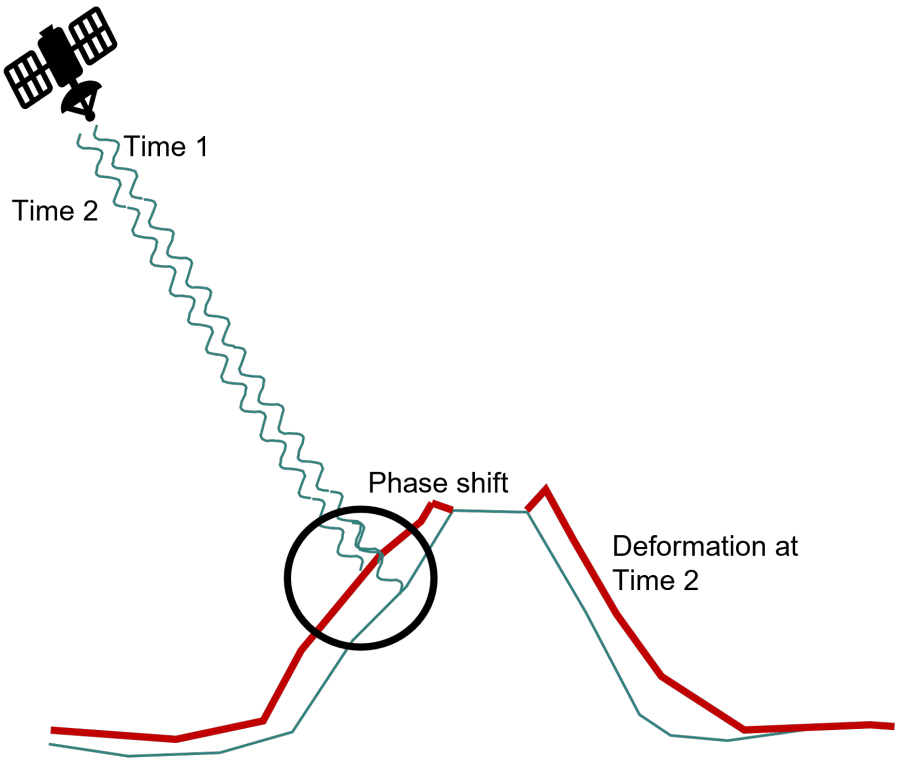


Figure 1.4 – I
llustration of how a phase shift occurs.Diagram illustrating a change in phase of a SAR wave. When a wave is emitted, a given number (n) of wavelengths are required for the wave to reach the ground from the satellite and return. The phase does not record the integer number of wavelengths required to reach the ground, but instead the fractional difference in the final wavelength (phase). If an integer number of wavelengths is required to reach the Earth and return, phase = 0. Otherwise phase $\neq 0$, $-\pi \leq phase \leq \pi$

phases of s_1 and s_2 (Pepe et al. 2017).

The measured phase is wrapped as modulo 2π, which introduces ambiguity in the relative phase cycle or wrap count, between neighbouring pixels (Figure 1.4). Phase unwrapping is used to return a continuous scale of relative displacement across each interferogram, resolving the ambiguity. It is important to note that the observed phase differences are influenced not only by surface displacement, representing the motion of the ground between acquisitions, but also by topographic variations arising from static differences in terrain elevation, which in turn impact the returning phase pathway leading to phase shifts. Additionally, atmospheric delays result in spatial and temporal phase variation (Wu et al. 2022).

The unwrapped phase is returned from wrapped phase using

$$\Phi_{i,j} = \Psi_{i,j} + 2\pi k_{i,j}, \tag{1.2}$$

where $\Phi_{i,j}$ is unwrapped phase, $\Psi_{i,j}$ is wrapped phase and $k_{i,j}$ is the wrap count number at position i, j.

The presence of two unknowns within Equation 1.2 means phase unwrapping is an ill-posed problem. Given the ill-posed nature of phase unwrapping, assumptions are made during the integration step. The commonly applied Itoh condition (Itoh 1982) uses the assumption that the difference in wrap count between neighbouring pixels does not exceed one for most gradients. However, this condition does not always hold, leading to an erroneous phase unwrapping step which subsequently propagates across the remaining interferogram. Unwrapping is one of the likeliest sources of error in the InSAR processing.

1.2.1 Coherence

In a SAR image, each pixel represents the sum of all the scatterers within the 2D area footprint rather than a single point, where each scatterer contributes an amplitude (strength of the returned signal) and phase (related to the path length travelled) (Hooper 2008). A reflection surface termed a "strong and stable scatterer" or "persistent scatterer" is a surface that remains stable in its scattering properties over time. These are often smooth, anthropogenic structures though natural surfaces such as bare rocks can also act as persistent scatters.

The coherence quantifies the similarity of the scatterers between acquisitions and is defined as

$$\gamma = \frac{\left|\langle s_1 s_2^* \rangle\right|}{\sqrt{\langle s_1 s_1^* \rangle \langle s_2 s_2^* \rangle}}. \tag{1.3}$$

Here, γ is the coherence magnitude for a given pixel, s_1 and s_2 are the complex values of SLC images 1 and 2, * indicates the complex conjugation, and $\langle\rangle$ is the average over a spatial window (Ferretti et al. 2007).

Coherence values close to 1 indicate a high proportion of persistent scatterers, whereas coherence values of 0 indicate complete decorrelation with few, if any, persistent scatterers. Coherence is influenced by multiple factors such as temporal decorrelation, caused by changes in the scatterers between acquisitions (e.g: vegetation movement, snow melt or water surface movement) (Hooper et al. 2004), stochastic phase noise and surface reflection properties, such as surface roughness or material type (Ferretti et al. 2007).

As previously stated, pixels in SAR images are composed of multiple signal returns within a given footprint. In areas where the majority of signal returns originate from strong, persistent scatterers with high temporal and spatial stability, the noise is suppressed and the pixels tend to have a high signal to noise ratio, and high coherence. Conversely, where there are fewer persistent scatterers, or the persistent scatterers have weak signal returns relative to the contribution from noisier scatterers, the noisier scatterers may dominate. This can lead to a lower signal-to-noise ratio for the pixels in the area, and is typically observed in vegetated or water-covered regions where there is high variability in the scattering properties.

Incoherence occurs when the scattering properties have high temporal variation between acquisitions. This can arise from vegetation growth and movement, snow melt and human activities such as construction work. Changes in viewing geometry between satellite passes can also contribute towards decorrelation. As a result, coherence often exhibits spatial patterns related to land cover (e.g high coherence in urban areas where the scatterers are temporally stable and good signal reflectors, and low coherence for vegetated areas), and can exhibit temporal patterns such as seasonal changes.

1.2.2 Unwrapping Error Sources

In InSAR, the measured phase is wrapped within a range of $-\pi - \pi$. Unwrapping errors occur when noise or low coherence leads to uncertainty in the phase measurements, meaning the unwrapping algorithm has a high degree of uncertainty in the number of 2π cycles between neighbouring pixels. Phase unwrapping error is most likely to occur in low coherence regions, and the error can propagate throughout the unwrapping process and lead to large scale unwrapping errors. To reduce the risks, pixels below a certain coherence threshold are often classified as noise, and are generally either excluded or down-weighted during unwrapping.

Unwrapping errors can be introduced from multiple sources. High degrees of random phase noise introduces uncertainty in the relationship between neighbouring pixels which can mean that these regions are difficult to unwrap, due to containing little to no "useful" pixel information. Regions of coherent pixels can be isolated due to separation by noise fields, such as in Figure 1.5c, leading to difficulties during unwrapping as the integration step often attempts to connect the isolated regions or coherent regions can remain wrapped. When unwrapping does occur through the noise field, assumptions are once again applied including replacing values via interpolation from the nearest coherent pixel values. An alternative approach to restrict the potential for error propagation during the integration step is to unwrap regions of low coherence as late as possible in the integration process.

Another possible way unwrapping errors may be introduced is in scenarios where insufficient sampling has occured relative to the amount of change and is known as aliasing (Figure 1.6)(Kester 2009). This is likely the cause of the unwrapping error shown in Figure 1.5a.

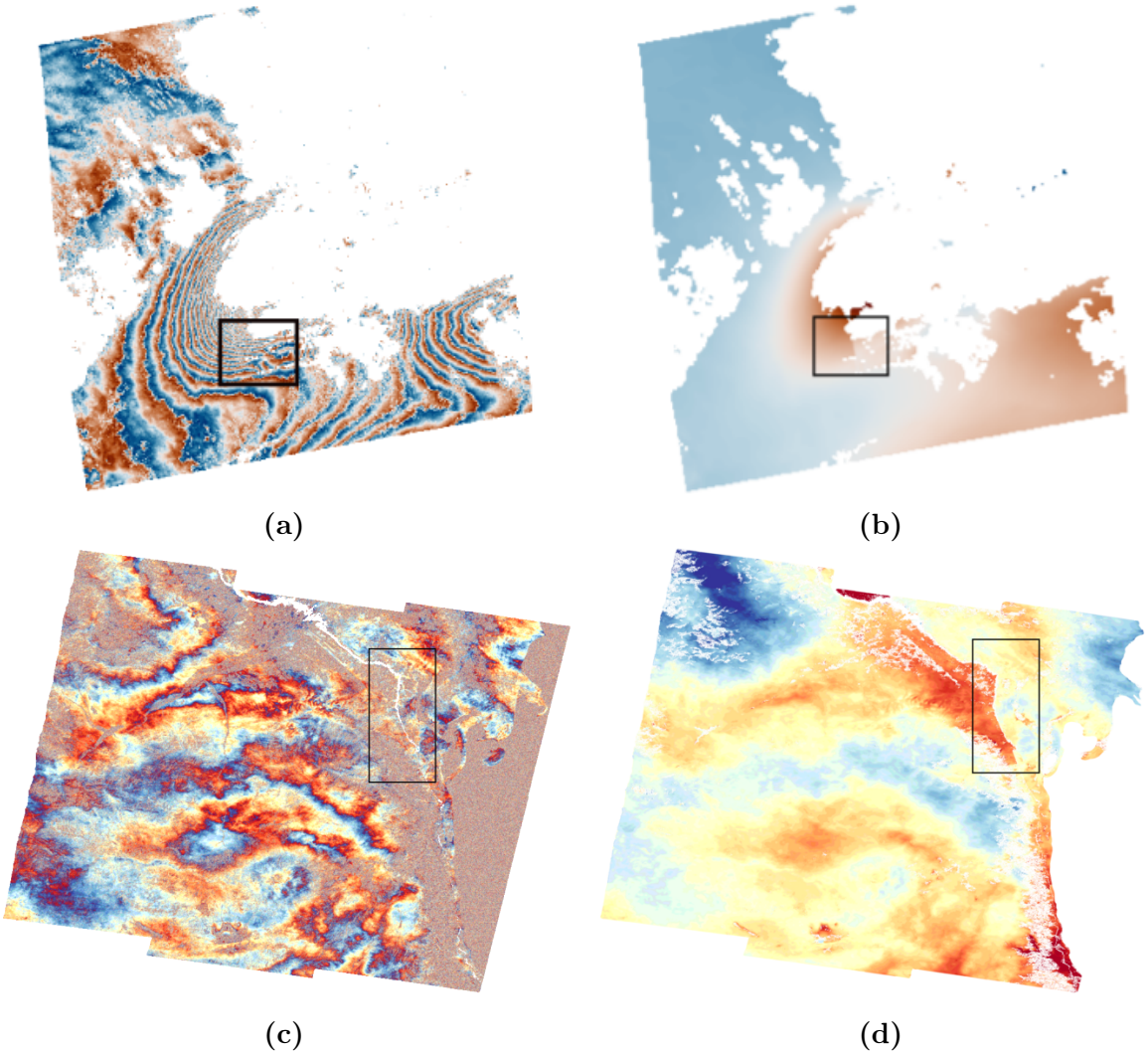


Figure 1.5 – Figure showing (a) Wrapped Interferogram (LicSAR Frame ID: 124A 06996 091406), (b) Unwrapped Interferogram (LicSAR Frame ID: 124A 06996 091406), (c) Wrapped Interferogram (LicSAR Frame ID: 006D 05111 131313), (d) Unwrapped Interferogram (LicSAR Frame ID: 006D 05111 131313). For (b) and (d), the unwrapping error is shown within the boxed region. In (b), the sharp colour change, rather than a smooth transition, is indicative of a phase unwrapping error. In (d), each side of the river has a different colour, indicating incorrect unwrapping between two isolated, coherent regions.

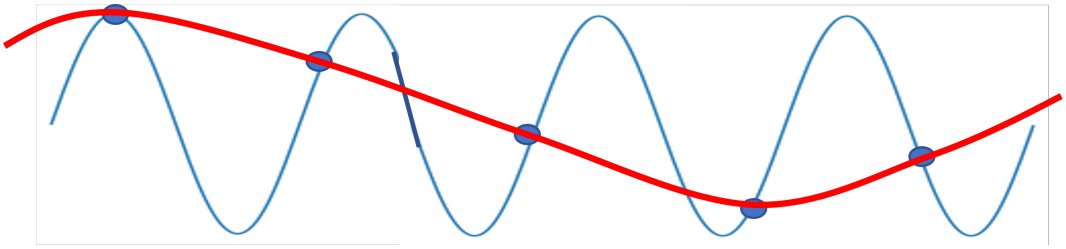


Figure 1.6 – Diagram reproduced from Kester 2009 illustrating aliasing. If undersampling occurs (aliasing - red line), multiple phase cycles are missed. This can lead to the introduction of unwrapping errors if the larger phase jumps are not considered. Where sufficient sampling occurs, phase cycles are recorded fully (blue line), and the true unwrapped phase values are returned.

1.3 Classical Phase Unwrapping Techniques

The objective for classical phase unwrapping techniques is to minimise the difference between the wrapped and unwrapped phase gradients to some penalisation power, using the Lp norm approach defined generally as

$$minimise \left\{ \sum_{i,j} w^x_{i,j} |\Delta\phi^x_{i,j} - \Delta\psi^x_{i,j}|^p + \sum_{i,j} w^y_{i,j} |\Delta\phi^y_{i,j} - \Delta\psi^y_{i,j}|^p \right\}. \tag{1.4}$$

Here $\Delta\phi^x$ and $\Delta\psi^x$ are the horizontal direction components over all positions i, j of the unwrapped and wrapped phase differences respectively, and $\Delta\phi^y$ and $\Delta\psi^y$ are the respective components in the vertical direction and $w^x_{i,j}$ and $w^y_{i,j}$ are weightings applied in the horizontal and vertical direction respectively (Ghiglia et al. 1996).

Pixels with unreliable phase measurements can be identified either by low coherence or through the detection of residues, where the sum of the difference between a square of 2x2 neighbouring pixels is not equal to 0 (Goldstein et al. 1988). Residues indicate the presence of discontinuities and once identified, the phase can be integrated to avoid these gradients. If a region is bounded by gradients identified to be discontinuities, the pixels within the isolated region may remain unwrapped if the branch-cut approach is used. The branch-cut approach can be viewed as an L_0 approach, where the objective is to minimise the total length of non-zero gradients; positive and negative residues are either

connected to oppositely charged residues or to an edge by the shortest route using branch cuts (Goldstein et al. 1988). Unlike L1 and L2 methods, the magnitude of the gradients between neighbouring pixels along the branch-cuts are not considered. As a result, this can lead to a situation where a branch cut with a wrap count change of +4 is favoured over a cut with a wrap count change of +2 due to the length of the cut being shorter, which may lead to sharp discontinuities across the branch cut. Integration can occur up until the branch cuts, but is not allowed across the branch cuts, hence areas remaining wrapped if completely isolated by the branch cuts.

Minimum spanning trees (MST) are an alternate L0 method for phase unwrapping, where the purpose is to connect all residues but reduce the sum of the weights of the edges within the tree, where an edge is the connection between neighbouring pixels. The MST is complete once all residues are connected; integration is then performed (Xin et al. 2010; Yang et al. 2006).

The minimum cost flow (MCF) approach is a commonly used L_1 norm global optimisation method, and seeks the minimum-cost configuration that balances all positive and negative residues (Chen et al. 2001). Positive and negative residues are connected via pathways, with each pathway having an associated cost influenced by factors such as pathway length, or information from quality maps such as the relative coherence of a region. In a lower coherence region, it is more likely a larger phase jump will occur, but there is also a higher degree of uncertainty in the gradient than if the same situation were to occur in a region of high coherence, and thus the lower coherence region will have a lower cost for this case. Integration is then performed (Goldstein et al. 1988).

1.3.1 SNAPHU

Statistical-cost network-flow algorithms for phase unwrapping (SNAPHU) (Chen et al. 2001) is probably the most commonly applied phase unwrapping approach for InSAR data currently. The objective of many phase unwrapping techniques described above is

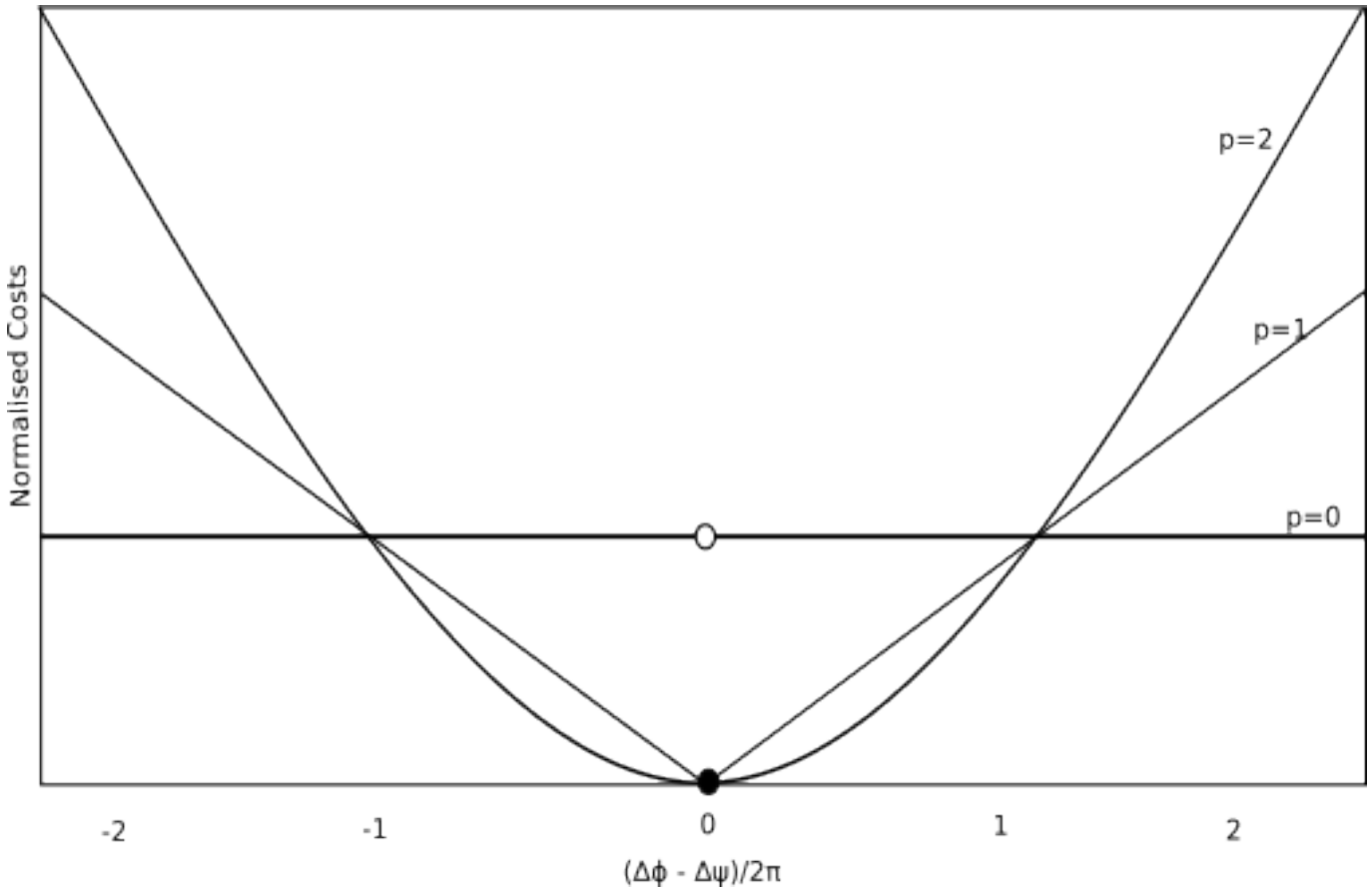


Figure 1.7 – Figure reproduced from Chen (2001). The figure shows the normalised costs against the offset between the unwrapped phase gradients and wrapped phase gradients for the optimisation cost function where the Lp norms are p = 0,1 and 2.

to minimise the difference between the wrapped and unwrapped phase gradients to some penalisation power (Equation 1.4).

Prior to Chen (2001), phase unwrapping applied a fixed p value during optimisation, which meant that the cost function applied across an interferogram inherently had a fixed shape during phase unwrapping (Figure 1.7). SNAPHU instead allows the cost function shape to vary spatially across the interferogram; Chen (2001) developed this approach to better minimise the negative logarithm of the unwrapped phase gradient probability density function, which is defined as

$$g_k(\Delta\phi_k, \Delta\psi_k) = -log(f(\Delta\phi_k|\Delta\psi_k),) \tag{1.5}$$

where

$$f(\Delta\phi \mid \Delta\psi) = \begin{cases} \dfrac{f_{\Delta\phi}(\Delta\phi)}{\sum_{m=-\infty}^{\infty} f_{\Delta\phi}(\Delta\psi + m2\pi)}, & \text{if } \Delta\phi = \psi + n2\pi. \\ 0, & \text{otherwise} \end{cases} \tag{1.6}$$

Here m and n are integer wrap count values (Chen 2001).

Chen (2001) simplifies the cost function Equation 1.5 to

$$g_k(\Delta\phi_k, \Delta\psi_k) = -log(f(\Delta\phi_k)) \equiv \Delta\psi_k + n2\pi. \tag{1.7}$$

by assuming that the unwrapped phase gradients are integer offsets of the wrapped phase gradients, and the numerator for the probability mass function (Equation 1.6) is dependent upon the unwrapped phase gradient $\Delta\phi$. The cost function is the inverse of the probability density function. SNAPHU characterises the cost function using four components: offset, variance, DzMax and Laycost g_d. Figure 1.8 is reproduced from Chen 2001 and illustrates each of these components.

Variance and offset ($\Delta\phi$) determine the width of the cost function, and the integer number of phase cycles difference between $\Delta\psi$ and $\Delta\phi$ respectively. Where there is high confidence in the unwrapped phase gradient, normally determined by setting a threshold coherence value, the probability density function is simplified to a Gaussian distribution (shown by the checked line in Figure 1.8). Where there is low confidence in the unwrapped phase

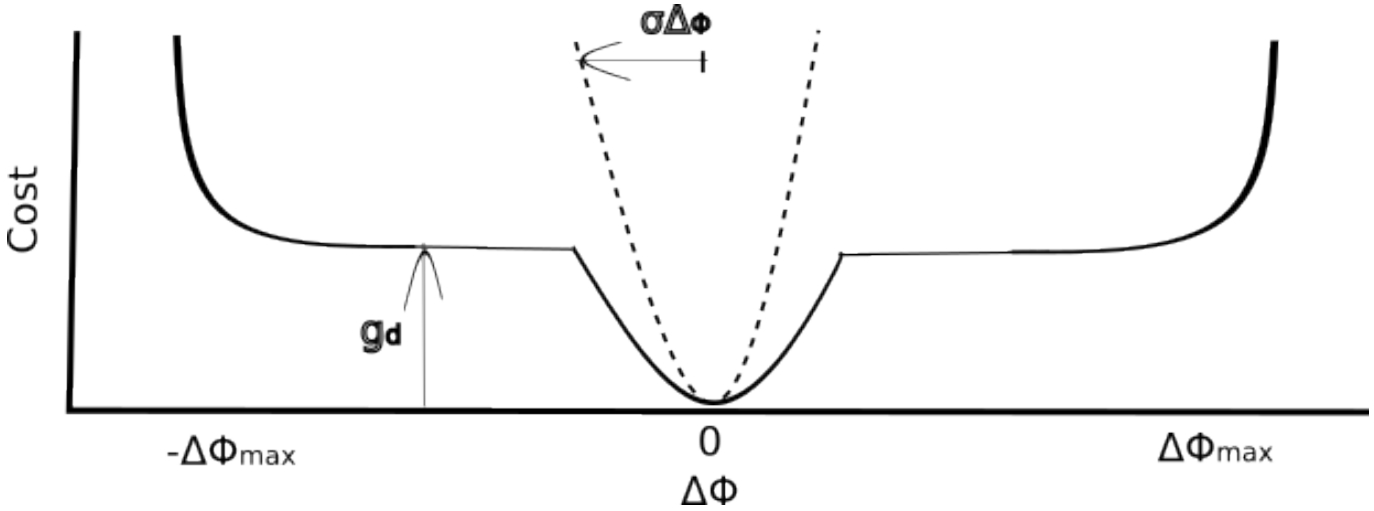


Figure 1.8 – Figure reproduced from Chen (2001). The figure shows an example cost function for deformation phase gradients. Where coherence is below a threshold, the cost function has a shelf-like region (solid line). Otherwise it has a parabola shape (dashed line). g_d is the conditional discontinuity probability, $\sigma\Delta_\phi$ is the coherence, and $\Delta\phi_{max}$ is the assumed maximum discontinuity.

gradient, a plateau is added using DzMax and Laycost. DzMax ($\Delta\phi_{max}$) is the value of $\Delta\phi$ after which the cost rapidly increases; a wider shelf indicates a greater degree of uncertainty in the unwrapped phase gradient. The height of the shelf, g_d, is representative of the probability of a phase discontinuity; a high value signifies a high associated cost, and vice versa.

Reliability of the unwrapped result can be improved by including additional information. For example, in-cooperation of coherence into the conditional probability function enables the model to weight each observation according to its phase reliability, reducing the probability of introducing an unwrapping error and supporting more accurate phase unwrapping (Chen 2001). For a full description of SNAPHU, please refer to Chen et al. 2001.

SNAPHU works well to unwrap the majority of wrapped interferograms; the flexibility of the cost function supports phase unwrapping as the unwrapped phase gradients increase in magnitude, and as noise increases. However, SNAPHU is dependent on the accurate calculation of the phase gradient and relies on the assumption that the majority of absolute phase gradients are less than π Mu et al. 2023. As noise levels increase, and where

deformation has occurred too rapidly relative to the level of sampling, the reliability of the phase unwrapping result declines. SNAPHU is also dependent on the quality index used to determine the quality of pixel gradients Wu et al. 2022 and the choice of reference pixel.

Deep learning models do not rely on the Itoh assumption, and have shown great promise in improving phase unwrapping. Recent research has introduced models that improve key components of the phase unwrapping process, including wrap count identification (Spoorthi et al. 2019), phase gradient estimation (Pu et al. 2021), optimal branch cut placement (Zhou et al. 2022a) and identification of discontinuities (Wu et al. 2022). These advances aim to increase robustness to noise, and improve the accuracy of phase gradients - an important factor given the reliance of optimisation techniques on an accurate prediction.

In the following sections, I will a broad overview of deep learning, followed by a literature review exploring how these models have been applied to reduce errors in unwrapped interferograms.

1.4 Deep Learning

Artificial intelligence, machine learning and deep learning have become increasingly popular approaches in recent years within the scientific research community (LeCun et al. 2015). With the release of popular tools such as ChatGPT making AI more accessible, combined with media enthusiasm for publishing both success and scare stories related to AI, it is unsurprising that the field continues to grow in size, popularity and prominence.

Neural networks are inspired by the brain's neural pathways (Oveis et al. 2021), with artificial neural networks made up of hundreds or thousands of artificial neurons or processing units (Montesinos López et al. 2022). Like a brain's neural pathway, an artificial neural network consists of interconnected neurons, though there is great diversity in the connection matrix of the neurons, which affects a model's ability to learn and perform

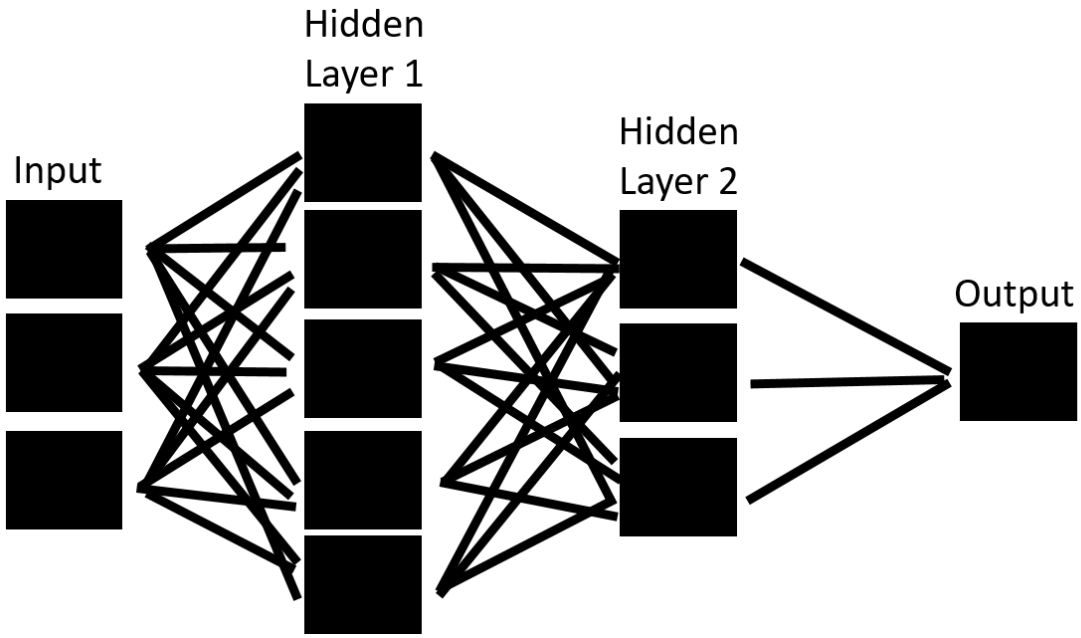


Figure 1.9 – Diagram of a fully connected neural network where each of the neurons within each layer are connected to all neurons in subsequent and previous layers.

tasks.

The basic neural network is constructed from an input layer, some number of hidden layers and an output layer (Figure 1.9). Every connection between neurons has an associated weight, which influences the level of impact one neuron has upon another, and a bias, which allows flexibility in when a neuron is activated, increasing the model's adaptability to more complex tasks. Neurons consist of a sum of linear and non-linear activation functions. Non-linear activation functions allow the capturing of non-linear relationships, rather than being restricted to purely solving linear regression problems (Oveis et al. 2021). Examples include the sigmoid, tanh or relu functions (Montesinos López et al. 2022).

Convolutional layers consist of a series of filters or kernels applied to a local region of the input which extract features. Kernels are small matrices of numbers which will be centered on each pixel, and element-wise multiplication performed and the returned values summed (Figure 1.10) (Oveis et al. 2021). The kernel is repeatedly applied to an image, resulting in a feature map; the feature map number corresponds to the number of filters, with each filter containing a different distribution of weightings so that the

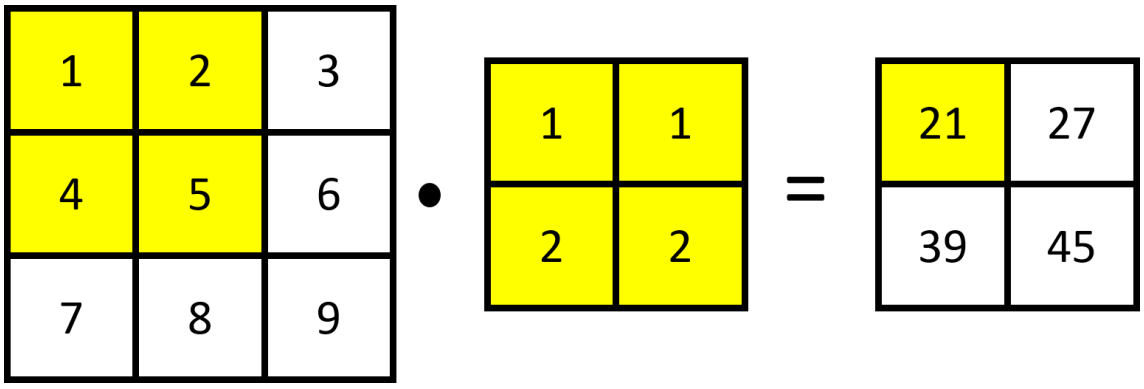


Figure 1.10 – Diagram illustrating how the convolutional filter works. The image values (left matrix) are multiplied by the convolutional filter using the dot product (middle matrix) to give the summed output (right matrix). The output in this case is smaller in size than the input.

model learns different features such as edges and corners. Stride size and padding are pre-set by the user, with padding allowing data preservation at map borders, and stride length influencing whether more global or local, fine scale features are captured. Both also influence whether the output and inputs are the same size.

Pooling layers are used to reduce the size of convolutional layers through the use of a summary operation, such as a maximum, minimum or average over a number of pixels, which reduces the dimensionality in only the width and height direction (Oveis et al. 2021). Dropout layers randomly deactivate a fraction of neurons during training to prevent over-fitting.

During the training process, the weights on the connections between neurons are adjusted and updated to contribute with different intensity levels to the final model output, with the purpose of improving the model's ability to perform a specific task. The model is trained on a set of seen data, which has a similar distribution to the data the trained model will be applied. The theory is that the model will learn from these past experiences and use this learnt knowledge to inform its output for a new, unseen scenario.

1.5 Deep Learning Phase Unwrapping Techniques

There are several ways deep learning models can be trained including using an unsupervised, semi-supervised and supervised approach. Given the success of applying deep learning across multiple other fields (LeCun et al. 2015), it is perhaps unsurprising that researchers have looked to use deep learning methods to improve phase unwrapping (Wang et al. 2021; Pu et al. 2021). When deep learning is used for phase unwrapping 2D single-baseline interferograms, supervised learning tends to be used (Spoorthi et al. 2020; Wang et al. 2021). A broad definition of supervised learning is that the model is trained using input-output pairs (Montesinos López et al. 2022). In cases where the output has categorical labels, such as "cat" or "dog", an activation function such as softmax is applied, whereas if the model is trained to target continuous output data, a regression function such as a sigmoid function is used. Semantic segmentation is a supervised learning approach where each pixel is classified into one of multiple pre-determined classes using the surrounding pixels to inform the classification choice (Zhou et al. 2022a).

Given that the supervised model accuracy is reliant upon the unwrapped groundtruth accuracy, a training data set consisting of synthetically generated interferograms are generally favoured, meaning the deep learning model accuracy is not reliant upon the success of another phase unwrapping approach. The proposed deep learning phase unwrapping approaches can be broadly split into two categories. One-step processes go directly from the wrapped to the unwrapped phase whereas multi-step approaches create intermediate products which are then used during phase unwrapping in combination with a classical phase unwrapping approach.

1.5.1 One-step Phase Unwrapping

One-step approaches have an input of wrapped phase to directly target unwrapped phase. Spoorthi et al. (Spoorthi et al. 2019) proposed the first deep learning method for phase

unwrapping, and targeted a wrap count map. Given that the approach did not specifically target phase unwrapping within the InSAR field, the training data lacked the random phase noise, atmospheric turbulence and topographically correlated noise which make unwrapping InSAR interferograms such a complex problem. The model was limited to an accurate prediction range of -36 to 36 radians, despite the training data set containing examples of interferograms with a range of -90 to 90 radians (31 different wrap counts) due to a class imbalance in the representation of the more extreme wrap count values. This restricts PhaseNets' application to deformation events where both a high range of wrap count values and a complex noise distribution are expected. Spoorthi et al. 2019 did demonstrate that deep learning has a comparable unwrapping performance at a reduced computational time when compared to classical unwrapping methods (in this case, quality guided phase unwrapping and a Matlab unwrap module).

Since PhaseNet, few papers have proposed to target wrap count explicitly, likely due to the restriction in model generalisation as they would be limited by the number of wraps the model was trained upon. In addition, class imbalance also presents an additional challenge as there are proportionally more examples of no change in wrap count change between neighbouring pixels and of low wrap count values. Instead, researchers have investigated alternative deep learning target outputs, focusing upon improved phase unwrapping and good generalisation. This included using wrapped phase as input to directly identify unwrapped phase.

A commonly favoured architecture for going directly from wrapped to unwrapped phase is the Generative Adversarial Network (GAN). GANS consist of two component models, the generative model and the discriminator model, trained in parallel in an adversarial competition (Zhou et al. 2022b). The discriminator model is trained to distinguish between the "real" groundtruth unwrapped data and the "augmented" generator unwrapped phase produced by the generator model, classifying each as either "real" or "fake". The generator model is trained to produce unwrapped phase outputs which are indistinguishable to the discriminator from the real unwrapped phase. Training proceeds

until the discriminator is tricked about 50% of the time which indicates the discriminator is doing no better than guessing whether the unwrapped phase is real or fake. The methodology was advanced through the development of a conditional GAN, whereby the introduction of extra information during model training allows the developers to provide some guidance as to what the correct target output image should be (Zhou et al. 2022b).

PU-GAN (Zhou et al. 2022b) and pu-pix2pix (Zhang et al. 2023) both use a GAN architecture to unwrap InSAR interferograms. PU-GAN attempted to remedy issues where the use of an L2 loss led to blurring in the unwrapped phase by using a cGAN generator and PatchGAN discriminator. PatchGAN discriminates between real and fake unwrapped phases at a NxN patch scale rather than at the image level. Pix2pix continued to use PatchGAN as the discriminator, but added a quality map as input to the generator structure to add conditional restrictions on the input features.

Whereas PU-GAN was trained using only real interferograms, pu-pix2pix was trained on a combination of real and synthetic interferograms. This was presumably to increase the model generalisation, which was highlighted to be weak in PU-GAN. PU-GAN used 158 training pairs from two Tandem-X interferograms and 1036 interferograms from 4 interferograms, neither set of which had an average coherence less than 0.71 whereas pu-pix2pix had 1000 training pairs of both simulated and real InSAR data, which likely covered a wider range of examples given that the data did not just originate from 6 interferograms. Both struggled to unwrap tight fringes in regions of very high noise and at fringe edges, though showed good performance at on data with simulated noise and phase filtering at coherence levels above 0.5. Neither showed examples where the models had been tested on deformation events related to earthquakes or volcanoes.

Whilst GANs are successful across a range of tasks, including text-to-image synthesis and image-to-image translation, they are notoriously fickle during training, with a tendency for non-convergence, mode collapse (generator collapses producing limited varieties of samples) and diminished gradient (the discriminator becomes too good at distinguishing between the generated and real images leading to the generator gradient vanishing and

poor model performance) with strong dependencies on chosen hyper-parameters (Dhariwal et al. 2021). Alongside a lack of generalisation and blurring, it is perhaps unsurprising that more recent phase unwrapping research has generally favoured alternative approaches.

1.5.2 Multi-step Phase Unwrapping

Multi-step phase unwrapping approaches use deep learning to target outputs which are often subsequently combined with a classical phase unwrapping technique. These approaches can be further sub-categorised as gradient estimators and 'residue based' approaches.

Gradient Estimators

Between 2020 and 2022, there was a keen interest in investigating the potential application of training deep learning models to target both phase gradient and wrap count gradients. Across most proposals, the assumption is made that the classes will cover "most" situations. This meant for the majority of cases using a 3 class approach $(\pm 1, 0)$ which was equivalent to assuming an absolute phase change of no more than one wrap count value. The CNN proposed by (Sica et al. 2022) has 5 classes $(\pm\frac{1}{2}, \pm 1, 0)$ which corresponds to the range covered by $\pm 1, 0$.

D-LinkNet also applies a 3 class assumption when classifying the wrap count gradients, but combines the output with an Unscented Kalman Filter (UKF). The rationale of applying a UKF is that it increases the robustness to noise and regions of high gradient changes increases by applying a quality-guided phase unwrapping approach which leaves the lowest quality gradients, which are often in regions of high gradient changes, until last. This should reduce the potential for phase unwrapping error propagation during the integration step and whilst the model itself showed good performance in the examples where the coherence levels were above 0.7, no examples were including of unwrapping

interferograms at a lower coherence.

Methods generally moved from initially exploring the viability of using deep learning models to learn edge information (Sica et al. 2022; Zhou et al. 2020), to considering issues such as class imbalance (Wang et al. 2021), the requirement for two networks to be trained (Li et al. 2022; Gao et al. 2022b) and the impact of noise on post gradient identification unwrapping steps (Gao et al. 2022b; Sica et al. 2022). The lowest coherence tested was 0.4, despite Sica et al. 2022 containing training data with coherence level 0.25, with worsening model performance for an average coherence below 0.5.

Residue Based Classifiers

Gradient estimators are still limited by the assumption that the 3 classes $\pm 1, 0$ will cover most, but not all, situations likely to occur in interferograms. Whilst generally leading to increased robustness in unwrapping results in regions of high noise, the limited classes can still lead to increased unwrapping errors in situations of high phase gradients or high noise. Therefore, researchers have since explored the potential of other intermediate products which could support optimisation-based, classical phase unwrapping techniques.

BCNet (Zhou et al. 2022a), PG-BCNet(Zhou et al. 2022c), DENet(Wu et al. 2022) and MCNet(Yang et al. 2023a) are all semantic segmentation models, trained to use at least a residue map as input. DENet gives a confidence in a discontinuity being present on a pixelwise basis, where discontinuity is defined as locations where the wrapped difference of wrapped phase differs from the true phase difference between adjacent pixels. DENet is given the residue, wrapped phase, vertical and horizontal gradients as model input. Instead of using a binary classification, a threshold of 0.3 is applied to model certainty in the prediction in each horizontal and vertical direction, where a model certainty below 0.3 is given class 0 indicating no discontinuity, whilst above 0.3 indicates a discontinuity and is classified as 1. The discontinuity map is used to guide the integration pathway of a MCF solver to improve the unwrapped solution. Wu et al. 2022 highlighted the need

for continued development to improve model performance in the presence of severely decorrelated regions.

BCNet and the subsequently developed PG-BCNet targeted optimal branch cut placements. BCNet continued to show results which had issues of leaving areas as wrapped phase through isolation (Yang et al. 2023a) of the region, and struggled with dense fringes even when little noise was present.

MCNet generates a map which labels pixels as either masked or not masked (Yang et al. 2023a). It showed good performance, but was primarily tasked with unwrapping TanDEM-X data, and did not provide any examples where it had been applied to Sentinel-1 data. In the paper, it was highlighted that improvement was required for successful application to more complicated interferograms, such as those with high noise or those which contain examples of large phase gradient changes.

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
PhaseNet (Spoorthi et al. 2019)	Wrapped Phase	Wrap Count	U-Net	First DLPU	Limited generalisation, not InSAR specific
PhaseNet 2.0 (Spoorthi et al. 2020)	Wrapped Phase	Wrap Count	Fully Convolutional DenseNet	Improved generalisation, generalisable to range -25 to 25	Not InSAR specific, instability in high noise (DENet paper shows this)
Phaseformer (Fan et al. 2025)	Wrapped Phase	Wrap Count	transformer backbone, atrous spatial pyramid pooling, semantic attention module	improved robustness to noise and complex environments	performs well over noise, less well than SNAPHU when fringes have good coherence on real data. Wrap count generalisability (up to 53cm deformation).
CNN (Calvanese et al. 2020)	Wrapped Phase	Wrap Count gradient estimation	U-Net	Alongside PGNet, first to target gradients	High noise, didn't test on deformation, training noise set as 0.25 lowest coherence, testing data only as far as 0.65 coherence

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
PGNet (Zhou et al. 2020)	Wrapped Phase	Phase Gradient	2 subnetworks - feature extraction and pixel classification	no dependency on Itoh condition, one of first target wrap count gradient	assumes 3 classes $\pm 1, 0$, limited testing on real InSAR, poor results in large gradient regions, PGE needs improvement, 0.4 coherence used for training and testing dataset
CNN (Zhang et al. 2021)	Wrapped Phase	Horizontal and Vertical Phase Gradient	multi-scale feature map extraction subnetwork, squeeze and excitation subnetwork	Improve stability of fast fourier transform least squares unwrapping algorithm	Limited testing shown in paper, not testing on deformation, -2dB SNR 0.39 coherence
PGENet-LS (Pu et al. 2021)	Wrapped Phase	Phase gradients	U-Net	Improvement on Least squares phase unwrapping	0.5 lowest coherence testing, did not test on cases of rapid deformation, smoothing, training and testing coherence lowest 0.5

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
GAU-Net (Wang et al. 2021)	Wrapped Phase	Wrap Count Gradient Estimations	U-Net with global attention upsampling	Explicitly target class imbalance, generalisation of performance across multi-source dataset, first analysis on InSAR deformation	mis-classification at fringe edges, lowest training coherence 0.4, drop in performance from below 0.6 coherence
Fusion Wrap Count Gradient Estimator(Li et al. 2022)	Wrapped Phase	Wrap Count Gradient	Encoder, Decoder with Residual Modules	One network for gradient estimation, improved performance for multiple residues	Residue proportion $> 25\%$ = performance drop, doesn't look at impact of noise
D-LinkNet PGE (Gao et al. 2022b)	Wrapped Phase	Wrap Count Gradient Estimation using continuous scale rather than classification	LinkNet, ResNet34 as encoder	First to combine deep learning and Unscented Kalman Filter, UKF circumnavigates large phase jumps, removes noise at higher coherence (0.7)	terrain trained, no deformation events, coherence used for training is 0.8

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
CCN (Sica et al. 2022)	Wrapped Phase, Coherence	Horizontal and Vertical Phase Gradient	U-Net	Improved handling of phase inconsistencies, coherence map helps model identify and manage noise regions	Worsening of performance for low coherence areas of sudden jumps of 2π and separated region consistency, testing to 0.65 coherence level
PU-GAN (Zhou et al. 2022b)	Wrapped Phase	Unwrapped Phase	Generative adversarial network	no post-processing	limited generalisability-trained on terrain phase, struggles with tight fringes
FCNet-CA (fully convolutional neural network with channel attention) (Yang et al. 2023b)	Wrapped Phase	Unwrapped Phase	Fully Convolutional Neural Network - Model initially trained on deformation, transfer learning to train on mining deformation	severe noise and large ground surface displacement associated with mining	worsening performance where deformation gradient per pixel > 3 radians for noisier interferograms, not Sentinel-1 friendly, limited generalisability

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
pu-pix2pix (Zhang et al. 2023)	Wrapped Phase, quality map	Unwrapped Phase	conditional GAN, PatchGan discriminator, U-Net generator		error at edges of unwrapped phase, noise tested on is synthetically generated noise, not teste on difficult phase and terrain INSAR situations
CycleGAN (Mu et al. 2023)	Wrapped Phase (real InSAR)	Unwrapped Phase (using SNAPHU, coherence maps)	CycleGAN, encoder-decoder generator, PatchGAN discriminator	pixel-level mapping, L1 loss and coherence inclusion to improve original CycleGAN performance, use RMS value to improve quality evaluation	
TLPU - HRU-Net, combined with a second step (2025) (Yan et al. 2025)	Wrapped Phase	Unwrapped Phase	3 part model; U-Net without skip connections (part 2), down-sampler (part 1), (part 3) residual modules, deconvolution model, convolution module	low coherence, large gradient changes	Haven't looked at deformation, specifically targeting bistatic LT-1 dataset

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
PU-Net (Liu et al. 2023)	Wrapped Phase	Unwrapped Phase	U-Net	synchronously denoising and phase unwrapping	limited showing of noise range for denoising, when was noisier, shows smoothing effect on fringe edges, struggles with noise
DLPU (Wang et al. 2019)	Wrapped Phase	Unwrapped Phase	U-Net	First Wrapped to Unwrapped Phase directly using Deep Learning	Not InSAR based, smoothing, instability in high noise (DENet paper shows this)
DMP-PUNet (Chen et al. 2025)	Wrapped Phase	Unwrapped Phase	U-Net ++ inspired model	improved robustness in complex terrains and noisy conditions	Limited comparison to real InSAR data, restrictions on interferogram size.
MoDL-PU (Zhou et al. 2025)	Wrapped Phase	Unwrapped Phase	2 step training, Res-UNet-Inception, supervised and semi-supervised approach	Separate models for single baseline and multi-baseline, multi-step to capture low frequency and high frequency phase information separately	requires further work to cover more scenarios, still needs some improvement over the high fringe density regions.

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
MCNet-PU (Yang et al. 2023a)	Residue locations	Binary classification of pixels as non-masked/masked	Convolutional neural network	Mask generation not reliant on quality map	Really complex interferograms, tested on Tandem-X - Sentinel-1c will be trickier
BCNet (Zhou et al. 2022a)	Residue Map	optimal branch cut map	deep convolutional neural network, Downsampling, Upsampling Network, Branch Cut Classification Layer	Shift from expert dependence, new PU target	Thick branch cuts in areas of higher residues, no guarantee of balancing residues. performs worse than MCNet (shown in MCNet). Can cause island effect - isolated regions remaining wrapped. not good at dense fringes (shown in MCNet) even when clean

Model	**Input**	**Target**	**Description**	**Contribution**	**Problems**
DENet (Wu et al. 2022)	Wrapped Phase, horizontal and vertical gradients, residue map	Phase Discontinuity maps in horizontal and vertical direction	main network and auxiliary network, discontinuity map as weights for cost flow tools (e.g: SNAPHU)	alternate to coherence for guiding MCF unwrapping	Limited testing of noise and large scale deformation events, severely decorrelated regions, aliasing regions
PG-BCNet (Zhou et al. 2022c)	Two-stage - wrapped phase for PG part, gradients for BCNet	gradients for PG part, optimal branch-cut map for BCNet	3 dilated convolution layers, batch normalisation layer, ReLU layer, standard convolutional layer, softmax layer	trained BCNet via transfer learning from PGNet	struggles at sharp edges, leaves areas unwrapped still. No testing on real InSAR.

1.6 Motivation for work and the aims

As stated, the deep learning models proposed to classify pixels according to wrap count gradients have tended to assume either that the wrap count gradient is 0, +1 or -1, arguing that this covers most situations, with any gradients exceeding these values grouped into ±1. The wrap count gradients $\Delta_k(s, s-1)$, are given by

$$\Delta_k(s, s-1) = \begin{cases} 1 & \text{if } k(s) - k(s-1) \geq 1, \\ 0 & \text{if } |k(s) - k(s-1)| < 1, \\ -1 & \text{if } k(s) - k(s-1) \leq -1. \end{cases} \tag{1.8}$$

The inherent misclassification of large wrap count changes which exceed 1 wrap count change as class ±1 remains a problem and can lead to subsequent erroneous phase unwrapping. This is particularly problematic in rapidly deforming regions, such as when a volcano is rapidly deforming or when an earthquake has occurred. Therefore, the logical next step is to consider instances where large-scale deformation events have occurred, using additional model classes to consider cases where the absolute wrap count gradient exceeds 1 during model training.

Chapter 2 contains two innovations. Building upon approaches discussed previously which target wrap count gradients, I expand the wrap count gradient model classes to include a 4th class, allowing the explicit identification of wrap count gradients exceeding the generally assumed 1 wrap count change between neighbouring pixels.

$$\Delta_k(s, s-1) = \begin{cases} \mathrm{C}|2| & |k(s) - k(s-1)| \geq 2, \\ C1 & k(s) - k(s-1) = 1, \\ C0 & k(s) - k(s-1) = 0, \\ C\bar{1} & k(s) - k(s-1) = -1. \end{cases} \tag{1.9}$$

This explicitly allows the identification of large wrap count gradient changes which exceed the "covers most situations" scenario, such as those at the centre of the deformation in Figure 1.5a, and allows the development of an alternate method to reduce the impact of aliasing on unwrapped phase.

Once gradient maps have been identified, previously published deep learning approaches have tended to combine the gradient maps with a classical phase unwrapping technique. Often the estimated gradients replace the wrapped phase gradients in the optimisation step. L2 norm approaches treat the phase unwrapping as a least-squares problem, and though the efficient data processing methods making it highly efficient, it will generally underestimate the phase gradient estimates in fringe density areas (Gao et al. 2024) and over-smooth the unwrapped phase result (Zhong et al. 2024).

Instead, I propose a novel wrap count assigning algorithm, known as the Greedy Phase Integrator (GI). The GI is based upon the intuition that where the multi-class gradient classifier model has the highest degree of certainty in its estimation, the predictions are most likely to be correct and therefore are least likely to cause phase unwrapping errors, with the inverse true for those with the lowest degree of certainty. Greedy approximation algorithms seek the global optimal solution through a series of steps, where each step is solved based on the best available choice at the moment, without considering the potential future consequences (García 2025). The GI searches for the optimal wrap count map solution, assuming this will have the lowest unwrapping error, by assigning wrap counts based upon the model prediction certainty, from highest to lowest. Once a pixel pair are connected, there is no possibility to back-track and change the wrap count gradient, as the assumption is that the already connected points are the optimal solution.

Wrap counts are returned across the interferogram without requiring previous connections, leading to patches of pixels where the wrap count has been returned in isolation, which are subsequently connected at a later stage. The GI step has enough flexibility to allow generalisation to any number of possible classes from the multi-class model. For the sake of ease, the following section will describe the case where the model is a 4 class

classification model (C0 , C1 , $C\bar{1}$, $C|2|$).

Additionally, whilst many papers have specifically targeted InSAR data, few have demonstrated and quantified model capabilities when applied specifically to Sentinel-1 data containing large-scale deformations, such as those synonymous with earthquakes or volcanic activity. Chapter 2 will quantify the performance the multi-class GI (MC-GI) when applied to both synthetic and Sentinel-1 data, and compare the results to SNAPHU.

Chapters 3,4 and 5 build upon the conclusions I draw in Chapter 2 (see Figure 1.11). Two prominent limitations are highlighted with MC-GI in Chapter 2. The GI algorithm is a single-loop optimisation process that relies upon the assumption that by taking the highest probability solution based on the model's certainty, the returned wrap count map will be optimal. However, it is known that phase noise will increasingly destroy the phase gradients as the level of noise increases, which then significantly impacts the success of the wrap count return algorithm.

The work I present in Chapters 3 and 4 aims to increase the model robustness to noise. I have presented a model which performs better than the MC-GI model in the presence of circular gaussian noise and also explored a more robust approach to integrate the wrap count gradient maps with varying levels of phase noise. Chapter 3 includes information about the training of the model. The work has built upon Chapter 2 as rather than include uniform gaussian noise, the noise has instead been added to the complex domain as circular Gaussian noise to better replicate the impact of noise at the fringe edges where there is a transition in the wrapped phase from π to $-\pi$ and vice versa. Like the innovation introduced in Chapter 2, this work investigated the inclusion of additional wrap count classes to allow the explicit identification of large gradient changes.

Chapter 4 is a continuation of Chapter 3. The work presented in Chapter 3 and Chapter 4 was separated for clarity. In Chapter 4, I present my exploration of an alternative

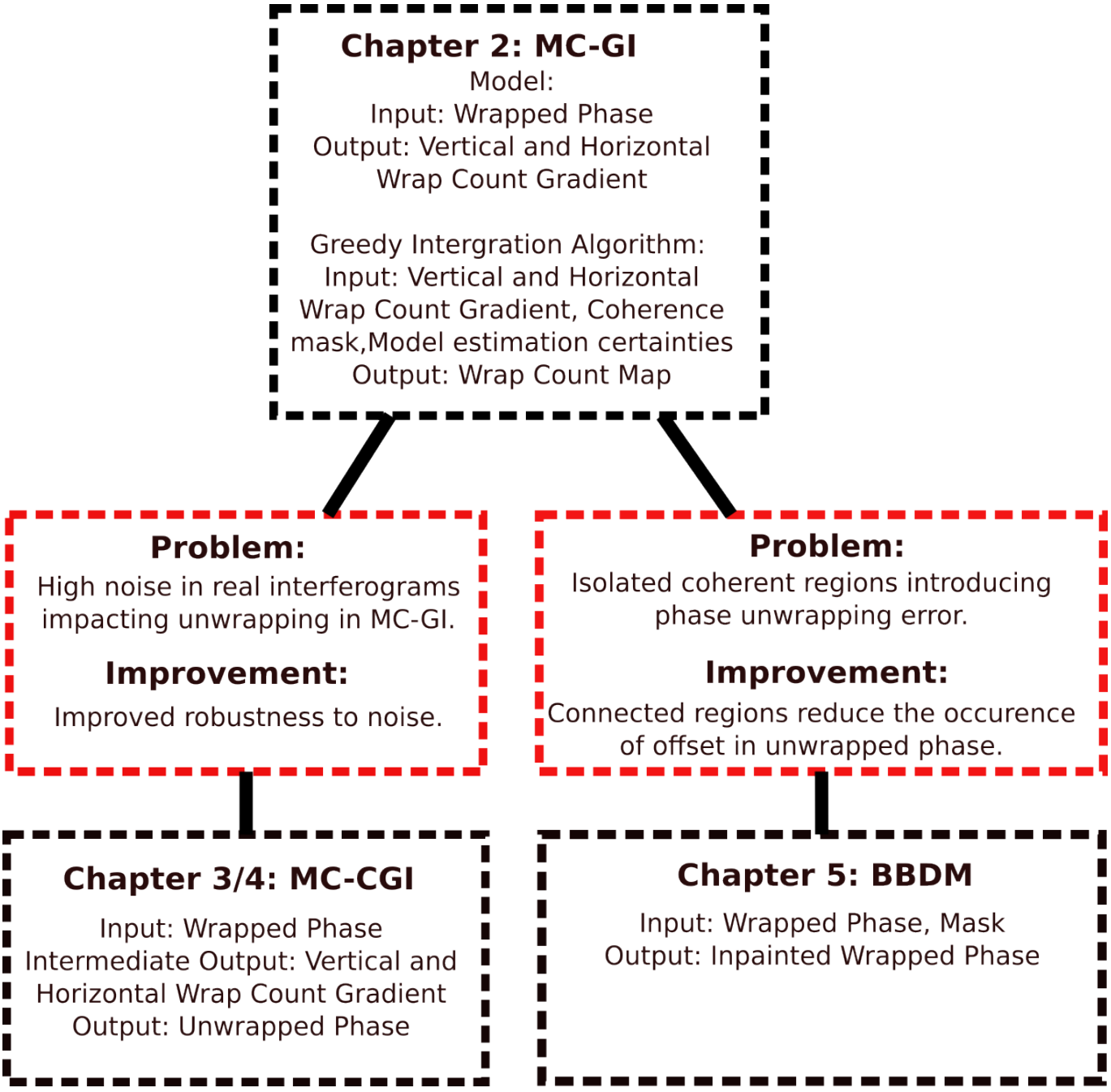


Figure 1.11 – Flowchart showing the link between Chapter 2, Chapter 3, Chapter 4 and Chapter 5 work. Chapter 5 work can be applied at each stage.

approach to integrate the wrap count gradient map. Whilst the multi-class model and Greedy Integration Algorithm in Chapter 2 performs well for situations where little phase noise is present, the algorithm is reliant upon high certainty for correctly classified wrap count gradients, low certainty for incorrectly classified pixels and is limited in the level of phase noise it can accommodate before the gradient classification accuracy declines. Instead, within Chapter 4, I propose an approach where wrap count gradients and model

certainty are used in combination with SNAPHU for cost guided integration, termed MC-CGI. Whilst Wu et al. 2022 has used the discontinuity confidence estimation as costs for MCF in combination with SNAPHU, the direct implementation of wrap count gradients as offsets and model certainty as variance in the cost file prior to phase unwrapping has not yet been investigated.

The second limitation of MC-GI is introduced due to the existence of isolated coherent regions. On such cases, when unwrapping using MC-GI, whilst pixels that were connected within a coherent region were correctly unwrapped relative to each other, unwrapped, isolated regions were offset relative to each other. This phenomena is known to impact phase unwrapping generally, including when unwrapping using SNAPHU, not just when applying MC-GI. In Chapter 5, a Brownian Bridge Diffusion Model (BBDM) is proposed as a solution; the BBDM is a generative model and can be used to gap fill the masked noise regions. In this case, masked regions may isolate coherent pixel regions, and the BBDM is proposed as a technique to inpaint the gap, providing a reasonable guess of possible return signals. I explored whether the gap filling would actually work through qualitative analysis, given that the use of a Brownian Bridge Diffusion Model for the purpose of gap filling wrapped interferograms has not been published before. I also investigated the optimal way to apply the BBDM to large scale interferograms exceeding a patch size of $256x256$ because memory limitations meant the BBDM could not be applied to interferograms larger than this. Finally, I investigated whether gap filling impacted upon the overall accuracy of the phase unwrapped result of SNAPHU and the MC-GI approach.

Each technique proposed in Chapter's 2,4 and 5 can be used in isolation. However, each technique proposed can also be used in parallel during phase unwrapping dependent on the wrapped interferogram, as each has a relative strength. MC-GI (proposed in Chapter 2) is particularly good for unwrapping interferograms with relatively little noise, but high fringe rates. MC-CGI (proposed in Chapter 4) is better than MC-GI and SNAPHU for interferograms with higher phase noise. The MC-GI model output can also be used as a

"first guess" input to SNAPHU, meaning both MC-GI and MC-CGI can be used together. The inpainting technique proposed in Chapter 5 can be used in parallel with any phase unwrapping approach, but currently has limitations in placing fringe edges.

Chapter 2

Multi-Class Greedy Phase Integrator

In this Chapter, I propose a complete method for phase unwrapping using a two-step approach known as Multi-Class Greedy Phase Integrator (MC-GI). A semantic segmentation model is trained to take wrapped phase as input, and target the wrap count gradients in both the horizontal and vertical direction. A novel Greedy Phase Integrator, which is based upon the intuition that the least uncertain and therefore highest probability predictions will likely lead to the best unwrapped solution, is then used to assign wrap count based upon the wrap count gradient maps.

2.1 Introduction

In interferometric synthetic aperture radar (InSAR), the wrapped phase has ambiguity due to being wrapped modulo 2π. Where an interferogram has been generated using Sentinel-1 data, one phase cycle of the interferogram corresponds to 2.8cm of deformation in the direction of the satellite's line-of-sight, and is visibly represented by one cycle of the colour spectrum. Phase unwrapping is the process of returning the relative displacement from the wrapped phase. In cases where little noise is present and sufficient sampling has occured relative to the amount of displacement, it is reasonable to assume that there

is no greater than one phase cycle difference between neighbouring pixels, which means that the unwrapped phase can be considered independent of the integration pathway chosen. In reality, unwrapping is often more challenging due to the presence of phase noise, which can be topographic, turbulent, thermal or speckle. Large phase gradients do occur between neighbouring pixels as a result of rapid deformation and in areas of steep topography. Therefore, the unwrapped result can no longer be considered independent of the integration pathway, meaning optimisation of the integration pathway is required to reduce the potential number of unwrapping errors and to give the overall best unwrapped result.

When classical phase unwrapping techniques are used to unwrap wrapped phase, it is assumed that the wrapped phase varies smoothly and the difference between neighbouring pixels will be less than π. However, in regions of significant phase noise or steep topography, phase jumps are often unavoidable. Whilst classical techniques are capable of placing the larger phase changes, they try to avoid doing so wherever possible, using techniques such as the placement of branch cuts to stop the integration pathway crossing the larger phase jump (Goldstein et al. 1988). The l1 norm, minimum cost flow approach penalises jumps dependent on their location and magnitude. The objective is to optimise the cost function to minimise the total cost of these jumps by identifying the optimal set of jumps from a given residue and ending at either an oppositely charged residue or an interferogram edge. This method aims to achieve the lowest possible total cost. Unlike the branch cut method, where jumps across the branches are excluded, the jumps are incorporated into the integration method (Flynn 1998).

L2 norm based approaches treat unwrapping as a global optimisation problem, seeking an unwrapped solution which has the minimal difference between the wrapped and unwrapped phase gradients. As the penalisation power squares the difference in gradients, large differences are heavily penalised, meaning that the algorithm tends to avoid concentrating errors in one area, which minimises the risk of introducing large jumps but does result in smoothing through high noise regions and errors which are more globally spread.

Weighted L2 methods, where each pixel pair is weighted based on external information, reduce the impact of unreliable data and reduce the level of smoothing across the global solution (Ghiglia et al. 1994).

Given that the objective of the Lp norm approaches is to return the best unwrapped solution by minimising the difference between the wrapped and unwrapped phase gradients to some penalisation power (Equation 1.4), it is a reasonable conclusion that a more accurate phase gradient estimation will lead to an improved unwrapping result (Pu et al. 2021). The application of deep learning techniques to improve phase gradient estimation has been explored (Sica et al. 2022; Li et al. 2022).

In existing studies, a series of semantic segmentation models have been proposed which are trained to classify neighbouring pixel gradients based on wrap count gradients into 3 wrap count gradient classes (C0 (+0), $C\bar{1}^*$ (-1 or less) and $C1^*$ (+1 or more)), making the assumption that 3 classes will cover most situations [1.8](Li et al. 2022). The output gradient maps are subsequently used in combination with a classical optimisation to improve the integration step of phase unwrapping.

In the existing publication, there is limited testing on cases where the average coherence of an interferogram is below 0.4. Wang et al. 2021 were the first to include an example of applying a deep learning model in combination with a traditional phase unwrapping optimisation technique to unwrap a differential interferogram, giving the example of an earthquake event in Hawaii in 2018. Though the training dataset did not contain examples of deformation caused by volcanoes and earthquakes, GAU-Net was able to identify fringe edges successfully when tested on such examples (Wang et al. 2021), though issues still existed in regions of high fringe rate and at pixels located at edges between the high coherence and low coherence areas. A later publication highlights these problems at fringe edges (Chen et al. 2023b), and proposed a loss function and model architecture adaptations. Whilst this adjustment in training the model did lead to improved wrap count reconstruction, examples of testing included in the publication were limited to synthetic samples with an average coherence across the interferogram above 0.5.

Two persistent challenges are repeatedly highlighted with using deep learning gradient estimation models for phase unwrapping: the accuracy of the unwrapped result in regions of high noise and in regions where there are large phase gradients. Moreover, while some studies have suggested that incorporating additional gradient classes beyond the three-class approach prevalent in the literature could improve performance (Zhou et al. 2020), no published work has yet demonstrated the impact of including additional classes on model performance and unwrapping accuracy.

In this work, I build upon existing publications which identify wrap count gradients in the horizontal and vertical direction. I trained a model which increases the number of wrap count gradient classes from 3 to 4, with a comparison to a 5 class model. The additional class in the 4 class model represents cases where the absolute wrap count change exceeds 1, whilst the 5 class model uses two additional classes compared to the 3 class model to differentiate between those wrap count gradients greater than +1 and less than -1. This will allow for the explicit identification of large gradients to improve unwrapping in regions of where rapid deformation has led to undersampling relative to the level of change. The second contribution is the development of a novel Greedy Phase Integrator to assign wrap counts based upon the multi-class model gradient maps which is equivalent to an L0 norm approach. Finally, whilst many publications have specifically targeted InSAR data, few have demonstrated and quantified the impact of applying model results to large scale deformation events shown in Sentinel-1 data. Within this work, I will quantify the impact of applying the Multi-Class GI approach in comparison with the popular SNAPHU technique.

2.2 Method: Multi-Class Gradient

A U-Net model is trained to classify the wrap count gradients in the horizontal and vertical directions into 1 of 4 classes. Classes C1 and $C\bar{1}$ represent a wrap count gradient of +1 and -1, respectively, class C0 represents a wrap count gradient of 0 or pure random

Gaussian noise, and Class $C|2|$ is a bin class for all wrap count gradients where the absolute gradient is greater than $+1$.

2.2.1 Model Architecture

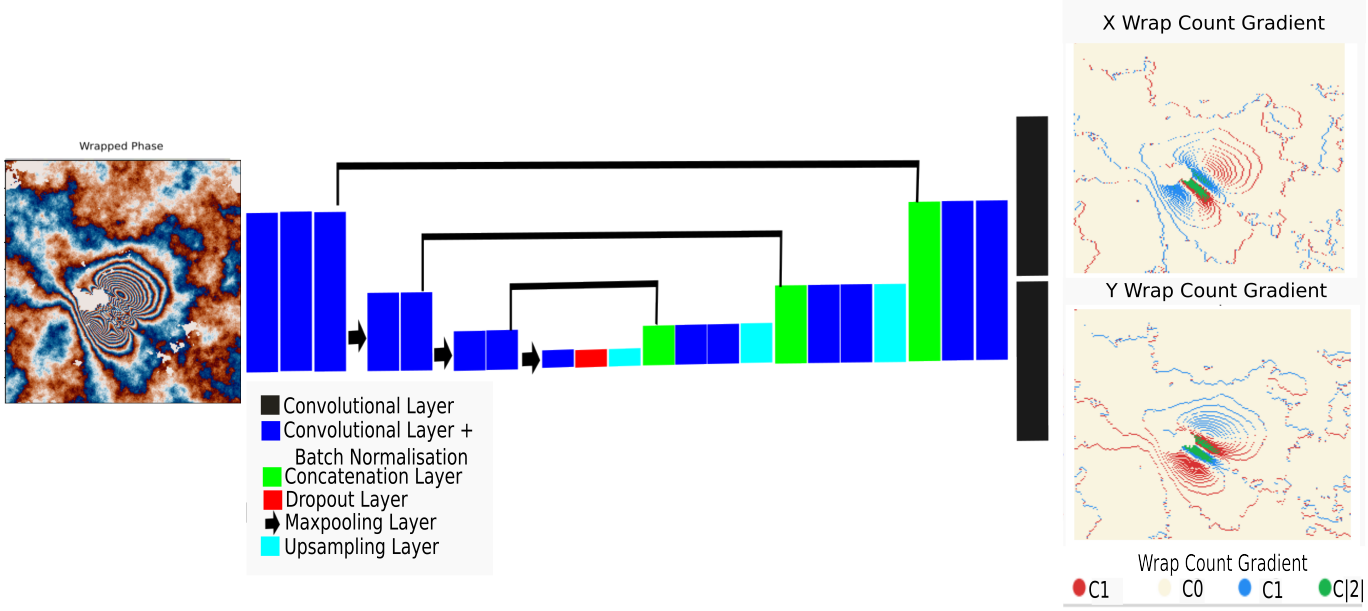


Figure 2.1 – Figure of model architecture, consisting of convolutional layers, batch normalisation and max-pooling layers in the encoder, convolutional layers, batch normalisation and concatenation layers in the decoder with skip connections. The wrapped phase input and corresponding vertical and horizontal wrap count gradient maps are shown, where there is a 4 class target output (C0 , C1 , $C\bar{1}$ and $C|2|$). For illustration purposes, the input and output are represented as if they are images, however model input and output each are greyscale, i.e one channel.

The U-Net architecture is based upon Ronneberger's U-Net model which was originally proposed for biomedical segmentation (Ronneberger et al. 2015), and chosen due to the consistently high performance shown in other studies to correctly classify the wrap count gradients (Wang et al. 2021).

The model input is the greyscale, wrapped phase. As shown in Figure 2.1 and Table 2.1, the model consists of three encoder blocks which decrease the spatial resolution of the input. Each block consists of a number of 2D convolutional layers and batch

normalisation pairings. Including batch normalisation increases the convergence speed and training stability and reduces the potential for overfitting (Chen et al. 2023a). Each block ends with max-pooling layer which downsamples the input layer and decreases the spatial information, helping to support fine-scale feature learning.

The decoder section of the U-Net model reintroduces spatial information via skip connections. Each decoder block contains an upsampling layer and two pairs of a 2D convolutional layer with a batch normalisation layer. An issue often highlighted in previous publications is the requirement for training two separate models - one network each for the horizontal and vertical wrap count gradients. Given that information from the horizontal gradients likely aids the prediction of the vertical gradient and vice versa, the proposed U-Net shares the same backbone architecture for the horizontal and vertical direction and uses two output heads. Each head consists of a convolutional layer followed by a softmax activation function for the classification of pixels on a pixel-wise basis per each head (horizontal and vertical).

The parameters used to train the model are in Table 2.2.

The output gives a probability of a pixel being in each of the 4 classes, each of which describe a different relationship between neighbouring pixels (see Table 2.3). For the purposes of this Thesis, the term minority classes refers to $C|2|$, $C2^*$, and $C\bar{2}^*$, whilst the term majority classes refers to all remaining classes. This is to make discussion of the model performance in later sections more succinct.

Block	**Layer Information**	**Filter Maps**
Encoder Block 1	3 x (Conv layer, batch normalisation), 1 x Max-Pooling	64
Encoder Block 2	2 x (Conv layer, batch normalisation), 1 x Max-Pooling	128
Encoder Block 3	2 x (Conv layer, batch normalisation), 1 x Max-Pooling	256
Block 5	1 x (Conv layer, batch normalisation)	512
Decoder 1	1 x Upsampling, Concatenation, 2 x (Conv layer, batch normalisation)	256
Decoder 3	1 x Upsampling, Concatenation, 2 x (Conv layer, batch normalisation)	128
Decoder 4	1 x Upsampling, Concatenation, 2 x (Conv layer, batch normalisation)	64
Final Block Xa Final Block Xb	Conv layer Conv layer	32 Number of Classes (4)
Final Block Ya Final Block Yb	Conv layer Conv layer	32 Number of Classes (4)

Table 2.1 – Table of model setup details

Parameter	Information
Learning Rate	1e-5
Optimizer	Adam
Dropout Rate	0.2
Filter Size	3 x 3

Table 2.2 – Details of unchanged parameters used to train the model

Meaning	Class
$C0$	0 change in the wrap count gradient. Also noise regions
$C1$	+1 change used for MC[4]-GI and MC[5]-GI
$C\bar{1}$	-1 change used for MC[4]-GI and MC[5]-GI
$C1^*$	+1 or more change used for MC[3]-GI
$C\bar{1}^*$	-1 or more less change used for MC[3]-GI
$C2^*$	+2 or more change used for the MC[5]-GI
$C\bar{2}^*$	-2 or less change used for the MC[5]-GI
$C\|2\|$	$\|+2\|$ or more change used for MC[4]-GI

Table 2.3 – Table showing the classes and their meanings

2.2.2 Loss Function

To avoid the class imbalance between the majority C0 background class and the minority classes C1 , $C\bar{1}$ and $C|2|$ negatively impacting the model classification accuracy, a Weighted Sparse-Categorical Cross-Entropy (WCCE) and Dice Loss were used to train the model.

Dice Loss is commonly used for imbalanced semantic segmentation classification tasks. It calculates the similarity between the model predictions and the ground truth (Wu et al. 2022).

Dice loss was calculated per class using

$$\text{Dice Loss} = 1 - \frac{2|X \cap Y|}{|X| + |Y|} = 1 - \frac{2TP}{2TP + FN + FP}, \tag{2.1}$$

where X and Y represent the ground truth and predicted ambiguity gradient maps respectively. TP, FP and FN represent true positive, false positive and false negative pixels.

The Dice loss was calculated per class and averaged across the classes to give the final overall loss. Each class equally contributes to the final loss. For each class, the "true positive" result meant the correction classification of that specific class to a pixel, e.g C1 to a C1 pixel, with a negative result meaning a different class had been assigned (C0 , $C\bar{2}$ or $C\bar{1}$ in this case). The Dice loss for each class contributed equally to the final average dice loss. By allowing each class to equally contribute towards the final dice loss, it ensures that C0 , which is most represented in the training dataset, does not disproportionality influence the model training as an "easy" classification option. I.e: if 90% of the training data set is C0 , the model may classify all pixels as C0 as it will result in a 90 % accuracy.

Dice Loss was chosen because it has been shown to improve model robustness in situations where the classes are imbalanced and also penalises the false positives and false

negatives. When used alone, Dice Loss can lead to an unstable training process due to slow convergence (Wang et al. 2021). To counteract this, I combined Dice Loss with Weighted Sparse Categorical Cross-Entropy (WCCE) during training.

With a data set where the classes are imbalanced, there is a danger that the model will classify all pixels into the majority class as this will generally lead to a low loss. The Weighted Sparse Categorical Cross-Entropy loss is used to more heavily penalise the incorrect classification of minority classes, forcing the model to learn correctly cases it is likely to see fewer of. The loss is defined as

$$\text{Weighted Sparse Categorical Cross-entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} w_j y_{i,j} \cdot \log(\hat{p}_{i,j}), \quad (2.2)$$

where N is the total number of samples, C is the number of classes, w_j is the weight for class j, $y_{i,j}$ indicates whether the sample i belongs to class j, $\hat{p}_{i,j}$ is the predicted probability that sample i belongs to class j where 1 and 0 means it does and does not belong to class j respectively (Zhang et al. 2022). The outer sum averages the loss over all samples in the batch, whilst the inner sum applies the weighted cross-entropy over all classes.

Weighting allowed the misclassification of the minority class to be more heavily penalised compared with the majority class, forcing the model to learn to correctly classify the minority class. Weights were proportional to the class sample size within the training dataset using

$$w_i = \text{total pixels}/(\text{C} * count_i), \quad (2.3)$$

where w_i is the computed weight for class i and $count_i$ is the number of samples in class i, and C is the total number of classes. Each loss applied alone to train the model resulted

in poorer model performance than when applying the combined loss.

The dice loss and weighted cateogorical cross-entropy contributed equally to the overall loss calculated during training.

2.2.3 Generation of Training Dataset

Synthetic wrapped and corresponding unwrapped interferograms of size $224x224x1$ pixels were generated using the software SyInterferopy (Gaddes et al. 2019). SyInterferopy was developed by Gaddes et al. 2019 to generate synthetic interferograms similar to those from Sentinel-1. Syinterferopy uses allows the user to select any region covered by the SRTM3 digital elevation model. A topographically correlated atmospheric phase screen and a turbulent phase screen are combined. Syinterferopy also provides the option to replicate spatial patterns consistent with deformation events in each interferogram; the choice of deformation events are dyke, sill, mogi, earthquake and no deformation event. The code was modified so that Gaussian noise was added to the final wrapped phase value, and the output is produced as a single channel.

To ensure larger wrap count gradients were represented in the data set, each variable contributing to the generation of the deformation source was randomly sampled within a range; this included varying both the depth and opening size of the source. The variable range was chosen through iterative analysis, where I visually inspected results to ensure spatial patterns remained consistent with those I had seen within the LiCSAR portal, and also through numerical analysis to ensure larger wrap count gradients were being introduced to the training data set. I also limited the wrap count gradient difference. This threshold was set at 15, based on an observed decrease in occurrence of higher gradient values within the training dataset. As most wrap count gradients were below 6, this threshold allowed the inclusion of more extreme deformation events. Examples of each of the deformation types are provided in Figure 2.2.

To prepare the target wrap count step change between neighbouring pixels data in the

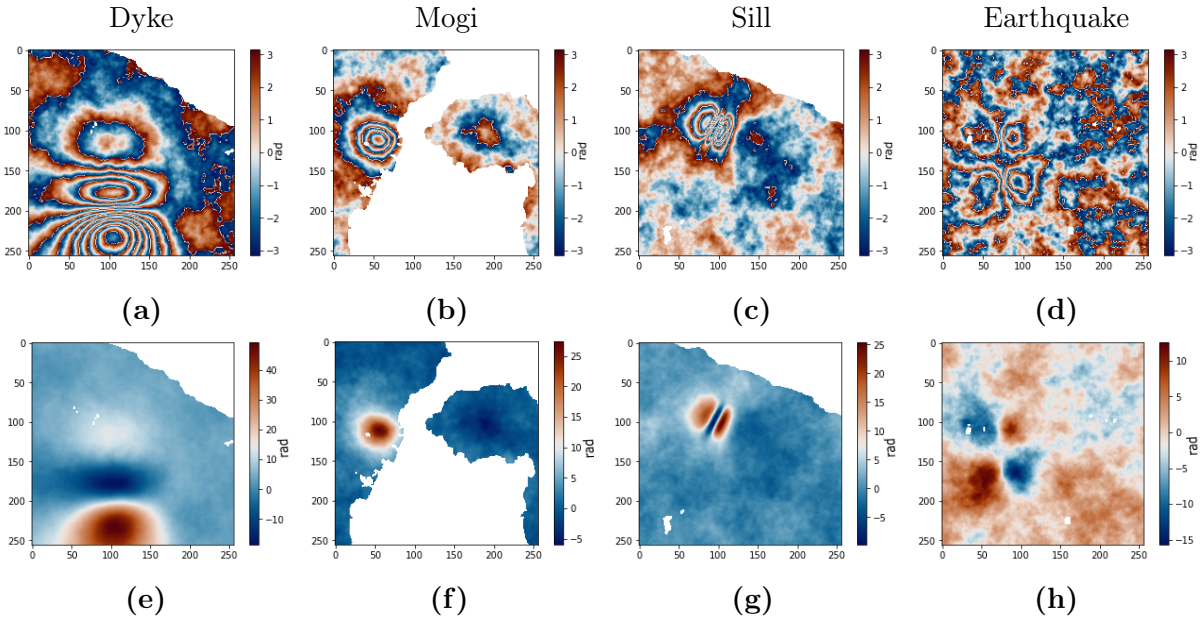


Figure 2.2 – Example of (a) wrapped dyke deformation, (b) wrapped mogi deformation, (c) wrapped sill deformation, (d) wrapped earthquake deformation, (e) unwrapped dyke deformation, (f) unwrapped mogi deformation, (g) wrapped sill deformation and (h) unwrapped quake deformation.

horizontal and vertical direction, a rearrangement of Equation 1.2 is used:

$$k_{i,j} = round(\frac{(\phi_{i,j} - \psi_{i,j})}{2\pi}), \tag{2.4}$$

where the horizontal wrap count gradient is given by

$$\nabla_h(k_{i,j}) = k_{i+1,j} - k_{i,j}, \tag{2.5}$$

and the vertical wrap count gradient is given by

$$\nabla_v(k_{i,j}) = k_{i,j+1} - k_{i,j}. \tag{2.6}$$

Pixels which are at the edge of the interferogram are assumed to have an equivalent wrapped phase value to the edge pixel meaning the class C0 is present in the target gradient map. The training dataset is prepared by patching interferograms of size 224 ×

1063 224×1 into non-overlapping $32 \times 32 \times 1$ patches. Patches which contain a greater than
1064 80% proportion of the minority class (larger jump classes) are augmented by rotation of
1065 90^{oc}, 180^{oc} and 270^{oc} to increase the proportion of this data within the training dataset.

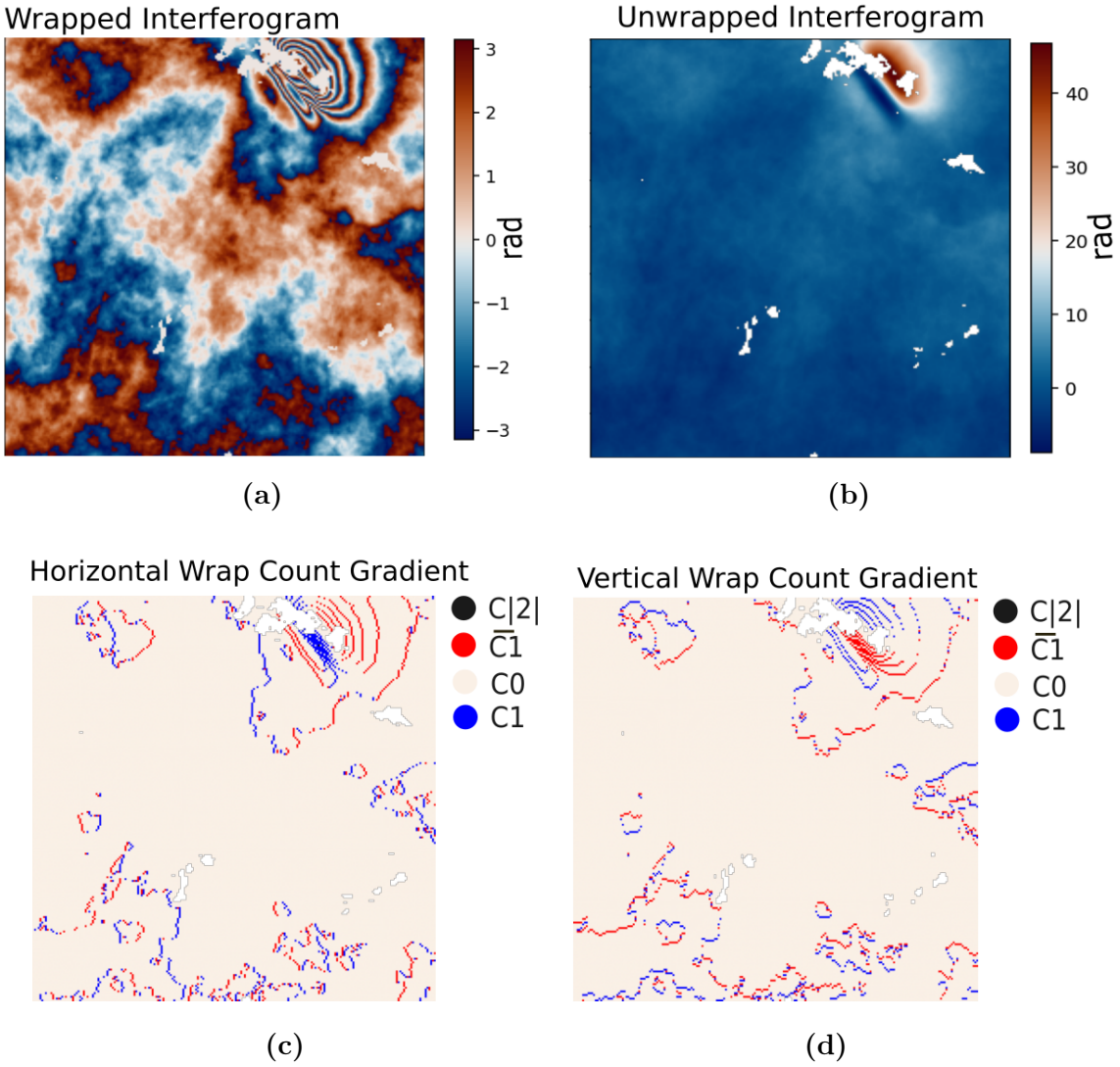


Figure 2.3 – Figure showing an example of the data used to train MC[4]-GI, where: (a) Wrapped interferogram, (b) unwrapped interferogram of (a), (c) wrap count horizontal gradient, (d) wrap count vertical gradient. The wrap count gradient classes are C0 , C1 , $C\bar{1}$, and $C|2|$.

1066 *NaN* regions were produced as part of SyInterferopy to replicate completely incoherent
1067 regions such as water bodies, or snow covered areas; *NaN* regions were filled with random
1068 values between $-\pi$ and *pi* to replicate the random noise seen over completely incoherent
1069 areas.

1070 The data set size is $112 \times 112 \times 1 \times 55498$ where 80% is used for training and 20% used for

validation. A separate testing data set is prepared using the same method as described above.

2.2.4 Noise Generation

Each interferogram has uniform Gaussian noise added and is clipped to a -π to π range during the training. This approach not only augments the data, increasing model robustness and reducing overfitting risk, but also ensures the model learns to accurately classify pixel gradients, even when noise was present. The standard deviation and mean of the uniform Gaussian noise added is uniform across each interferogram.

2.3 Method: Greedy Integration

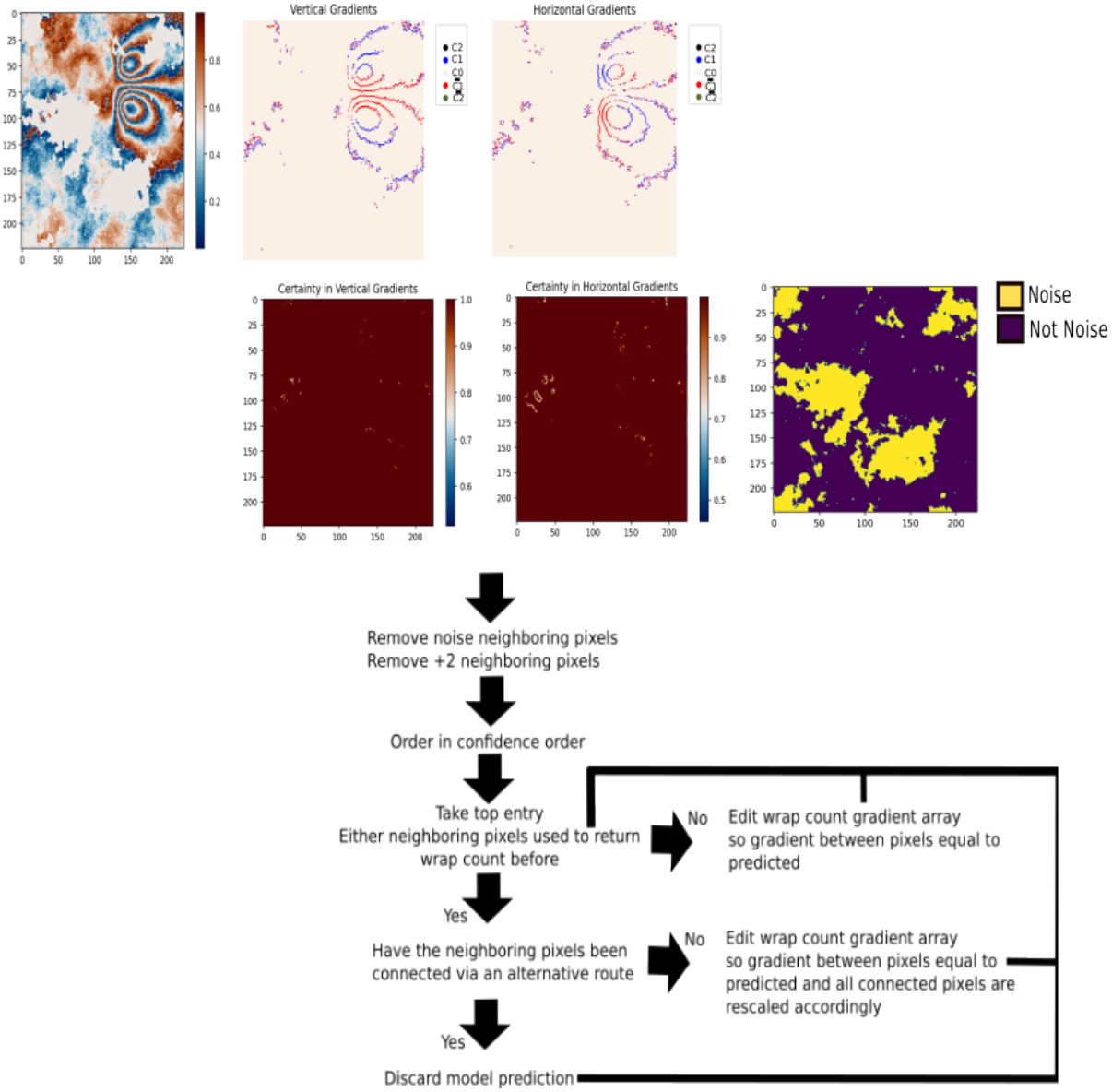


Figure 2.4 – Flowchart demonstrating the logic used during the wrap count return from the X and Y model prediction values

I have developed a Greedy Phase Integrator algorithm to return wrap count directly from the horizontal and vertical wrap count gradient estimations, rather than combining the model output with classical phase unwrapping techniques. The Greedy Phase Integrator algorithm uses the intuition that by using certainty to guide the integration order, it is likely to lead to the optimal global wrap count return configuration with the highest

summed model prediction certainty.

The algorithm uses the horizontal wrap count gradient estimation, the vertical wrap count gradient estimation, coherence, model certainty in the horizontal wrap count gradient estimation and model certainty in the vertical wrap count gradient estimation as input. In the algorithm, pixels below a manually set, minimum coherence threshold level or labelled as $C|2|$ are excluded from the integration process. The minimum threshold value can be set to balance requirements for high unwrapping accuracy with the need to unwrap as many pixels as possible. Whilst the method shares characteristics with the quality guided phase unwrapping approaches, the proposed method does not rely on the highest quality pixels being neighbours, but instead allows sequential integration to occur anywhere across the interferogram. This is advantageous as it allows wrap count to be returned in isolated regions of high coherence pixels without any risk of being impacted by incorrect unwrapping across the noise field. An overview of the integration process is given in Figure 2.4.

Step 1: Pixel gradient estimation preparation

Consider the example of an interferogram of size $224 \times 224 \times 1$. The wrapped interferogram is fed into the multi-class gradient prediction model which generates a corresponding map of wrap count gradient predictions along with the model's certainty in these predictions for both the horizontal and vertical directions. The wrap count predictions are then ordered by the model's certainty, from highest to lowest.

Each wrap count step change prediction has an associated information feature set:

- Location of pixel in wrap count map to be updated at pixel position i,j in the wrap count map
- Model softmax layer output - probability associated with the most probable class
- Pixelwise categorisation where pixels are either 'noise' or 'non-noise'. Based upon a mask generated from the coherence map where below a threshold, mask = 0,

otherwise mask = 1 at each pixel location

- Direction of gradient (horizontal or vertical)

Wrap count gradient predictions identified between pixels which are below the minimum coherence threshold, neighbouring to such pixels at position $i+1$ or $j+1$ in the horizontal and vertical directions respectively, and those classified as $C|2|$ are removed from the list.

Step 2: Iterative wrap count return

The Greedy Phase Integrator is then applied to the remaining pixels.

The most confident gradient assignment is taken. Using the location of the neighbouring pixel pairing (Pixel A and Pixel B), whether either pixel has already been assigned a wrap count is checked. In cases where no wrap count value has been assigned, the wrap counts of Pixel A and Pixel B are updated to ensure the gradient between the pixels matches the model estimated gradient. The connection array is updated to track that Pixel A and Pixel B are now connected.

Where only one pixel in the pairing has been previously involved in the Greedy Phase Integrator process, and is therefore linked to other pixels already (say Pixel A has been involved previously), Pixel B would be updated to ensure the wrap count gradient between Pixel A and B matches the model estimation, and the connection array is updated to show Pixel B is now linked to Pixel A and all the pixels already connected to Pixel A.

Where both Pixel A and B have been assigned a wrap count value already, the connection array is checked to identify if Pixel A and B have been previously been connected via an alternative route. In cases where this is a negative, Pixel A and connected pixels would be updated to ensure the gradient between Pixel A and B matches the model estimate for the wrap count gradient. The connection array is updated to demonstrate that Pixel A and connected pixels are now connected to Pixel B and its connected pixels.

Where Pixel A and B have already been connected via an alternative route, the greedy nature of the Greedy Phase Integrator prevents further adjustment. This occurs even

when the wrap count gradient in the resulting wrap count map does not align with the model's estimated gradient. The algorithm operates on the assumption that the previously established connection route has a higher global probability than any potential adjustment. As a result, the information package is removed, and no updates are made to the wrap count.

The process is iterated until all gradient estimates have been used or discarded. The benefit of the approach is that there is no requirement for any pair of pixels to have been connected before which ensures all regions of highly coherent pixels are assigned wrap counts without the introduction of error through the need to connect to other pixels through a noise region. This also means that a reference point does not need to be chosen at the start of the process. Instead, the wrap counts are scaled relative to the most probable grouping of connected wrap counts at each step of the process. This is advantageous as it means isolated coherent regions do not need to be connected through regions where all pixels are considered below a coherence threshold. However, it does result in isolated regions being correctly unwrapped relative to the connected points within the region, but wrongly offset relative to other groupings of coherent pixels.

At the end of the process, a wrap count map of matching size to the wrapped interferogram is returned. Figure 2.5 demonstrates the wrap count at multiple stages of the wrap count return algorithm process from start of the process, to the completion of the process.

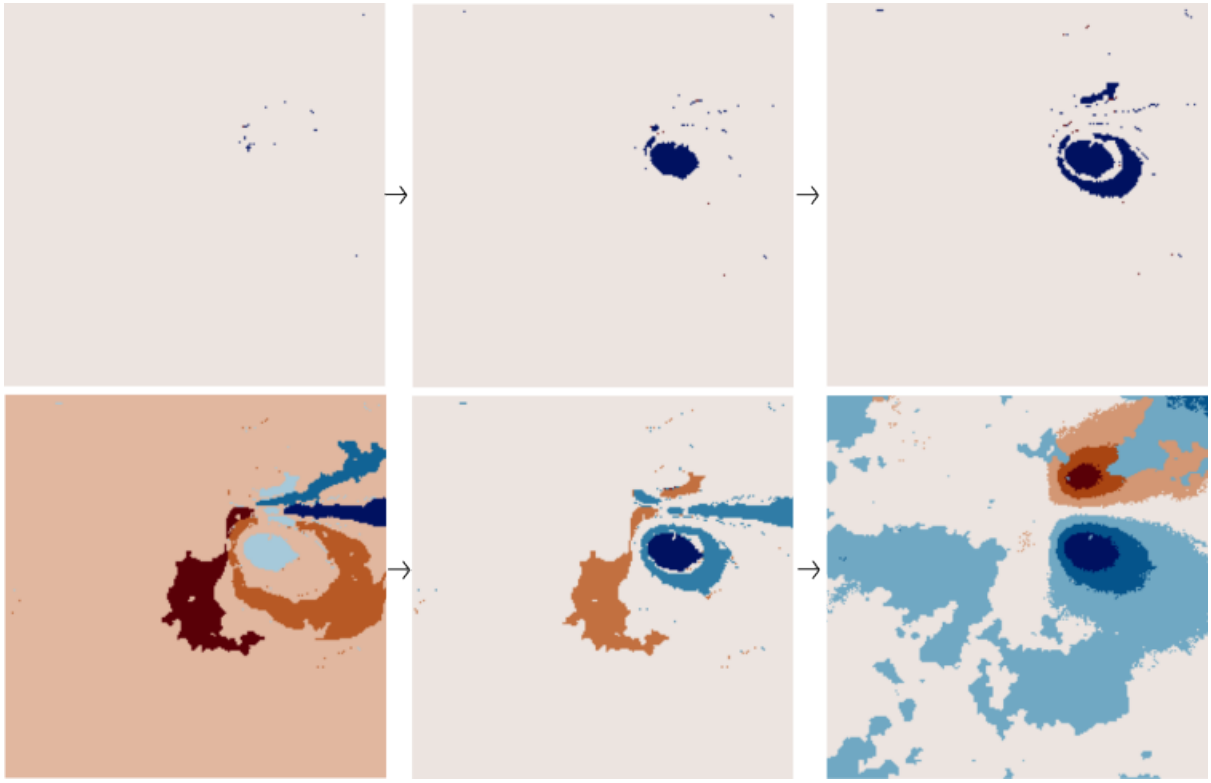


Figure 2.5 – Figure demonstrating multiple stages of the wrap count return process from start (top, far left) to finish (bottom, far right).

2.4 Results and Discussion

2.4.1 Qualitative Assessment of the MC[4]-GI Gradient Estimation

I began by qualitatively evaluating the model's performance solely on the it's ability to correctly classify the horizontal and vertical gradients when varying levels of uniform Gaussian noise are added to synthetic, wrapped interferograms. The test data set consisted of 400 unseen synthetic interferograms of size $256 \times 256 \times 1$ pixels.

Figure 2.6d-f illustrate when no additional noise has been added to a synthetic interferogram, MC[4]-GI is capable of differentiating between the C1 $C\bar{1}$, $C|2|$ and C0 gradients. A particularly impressive example is shown in Figure 2.6d, where the MC[4]-GI has correctly differentiated between C1 , $C\bar{1}$ and $C|2|$ gradients. The multi-class prediction model still does make some errors, as shown in Figure 2.6d and in Figure 2.6f, indicated by it misclassifying pixel gradients at the transition border between $C|2|$ and either C1 or $C\bar{1}$, however it is likely during the normal unwrapping process these pixels would exist in regions which would likely be masked due to having a low coherence because of the rapidly changing signal.

Comparing the results shown in Figure 2.6 with the results shown in Figure 2.7, where additional Gaussian noise has been added to the wrapped interferograms, the MC[4]-GI model does continue generally to correctly classify gradients. However, there are increased occurrences of gradient misclassification, particularly as C0 , which are likely caused by the model's limited denoising capabilities. As noise increases, the ability to differentiate between patterns related to true change in wrap count gradient and those related to randomly applied model becomes more difficult, and thus the model determines gradients to be noise and classifies these as C0 .

Figure 2.7f shows that on the edge of the region where pixel gradients are classified mainly

as the minority class $C|2|$, the model has found it difficult to identify the pixel gradients that mark the points where the gradients transition from $C|2|$ to C1 or $C\bar{1}$. However, it is still remarkable that in Figure 2.7f, the model has been able to correctly differentiate any pixel gradients as $C|2|$ rather than noise given that, from looking at Figure 2.7c, it would be incredibly difficult for a human.

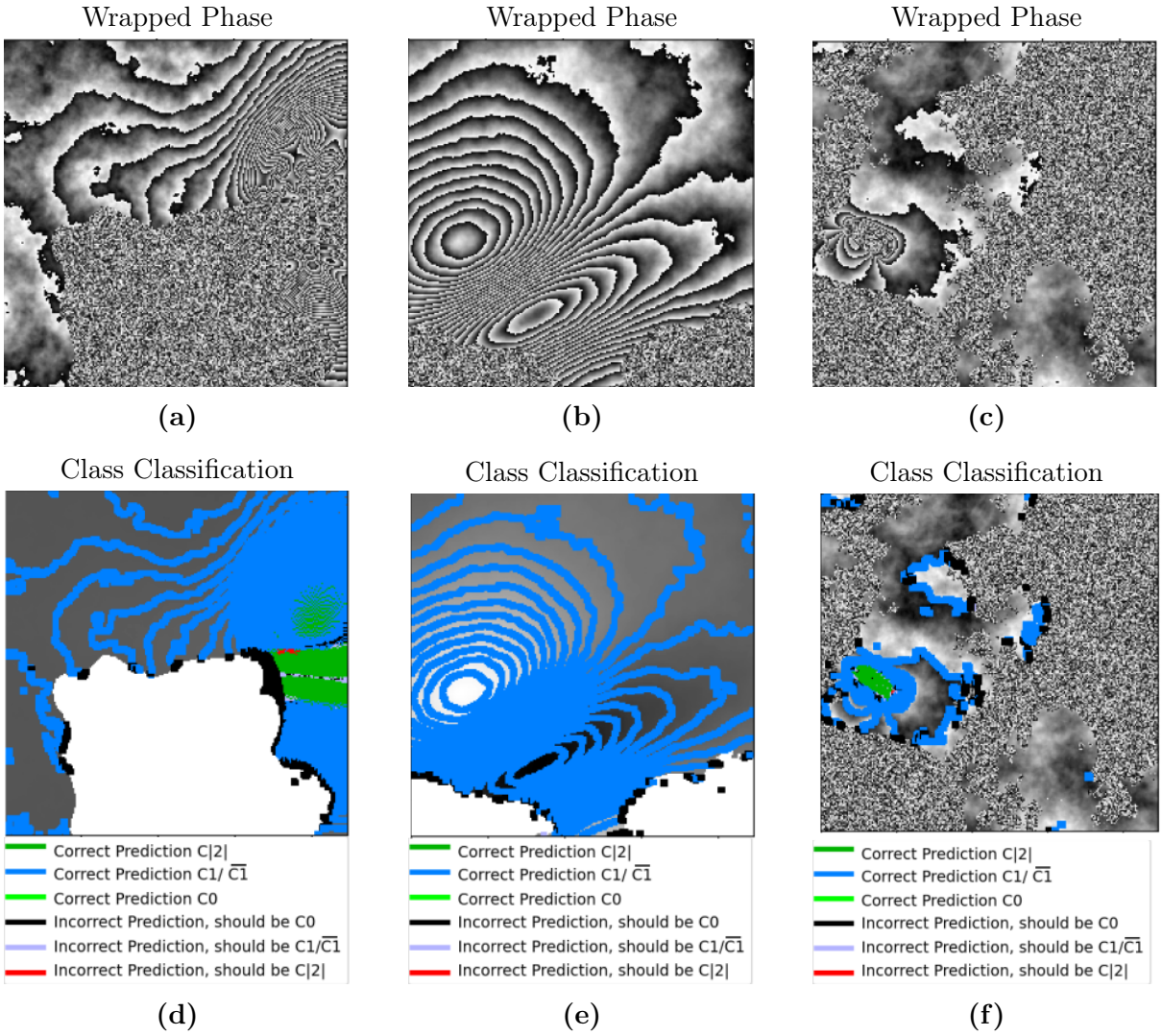


Figure 2.6 – Figure showing (a - c) : 3 examples of wrapped, synthetic interferograms, (d-f): the wrap count gradient predictions returned by MC[4]-GI in the horizontal direction with all incorrect pixels highlighted and correct pixel estimations for the minority class $C|2|$ shown.

Overall, the results shown in Figure 2.7 and Figure 2.6 indicate qualitatively that MC[4]-GI has a high degree of accuracy when classifying the majority and minority pixel gradients when varying degrees of uniform Gaussian noise are added.

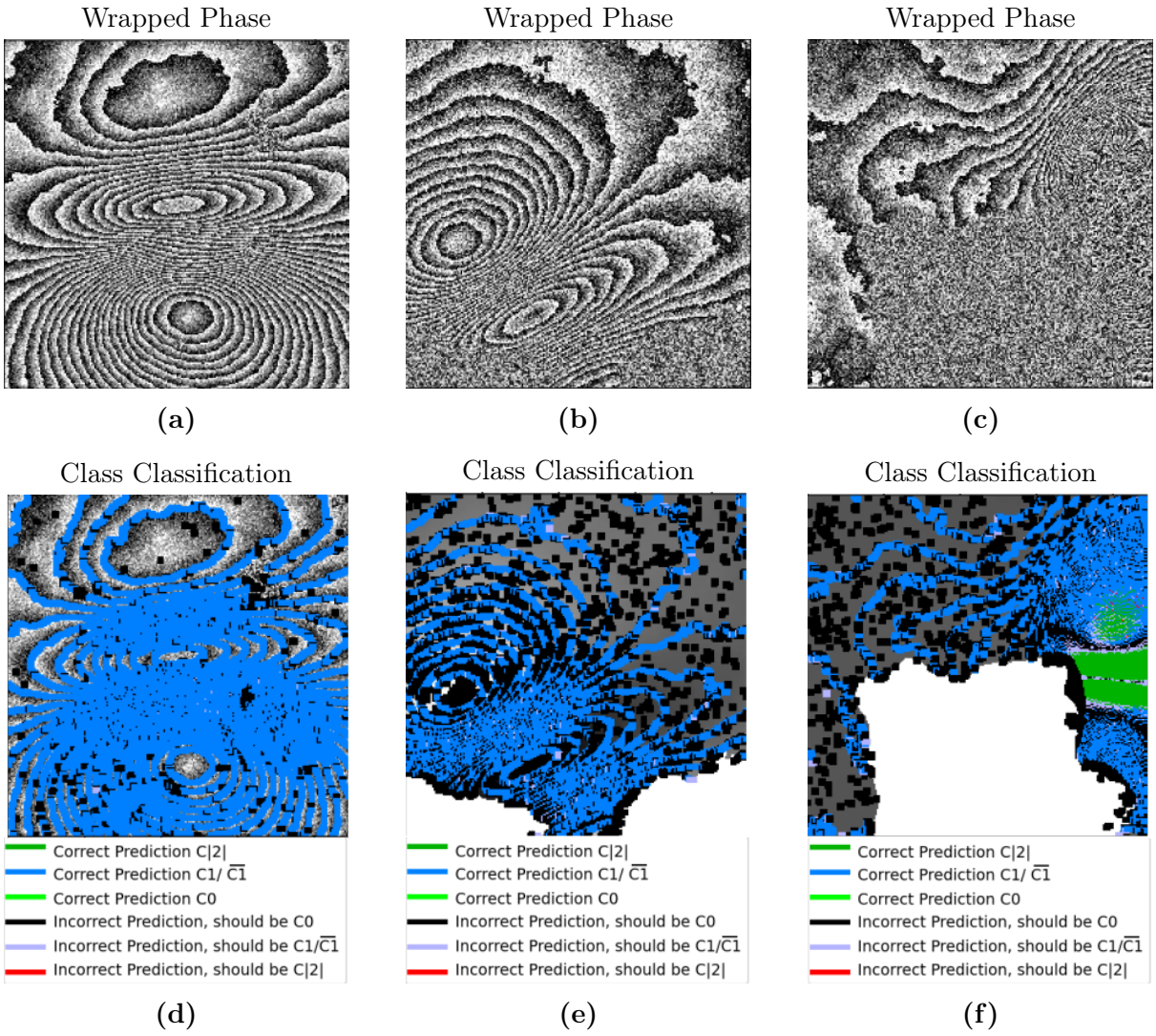


Figure 2.7 – Figure showing (a - c): 3 examples of wrapped, synthetic interferograms with varying levels of uniform Gaussian noise added, (d-f): MC[4]-GI wrap count gradient estimates in the horizontal direction with all incorrect pixels highlighted and correct pixel estimations for the minority class $C|2|$ shown.

2.4.2 Quantitive Assessment of MC-GI for Wrap Count Gradient Prediction on Synthetic Data

The MC[4]-GI classification performance was compared to SNAPHU. Wrapped interferograms were unwrapped using SNAPHU, and the wrap count gradients were subsequently calculated using the method discussed for the training data set target preparation. Gradients were grouped in line with a 4 class approach (C0 , C1 , $C\bar{1}$ and $C|2|$), with gradients grouped in line with a 4 class target; i.e an absolute wrap count gradient exceeding +1 is grouped as $C|2|$. When applying SNAPHU, no masking, additional weights, or cost file were used and the DefoMax was set to 1.2.

$$\text{Coherence}_{SNR} = \frac{SNR}{SNR+1} \tag{2.7}$$

(Hanssen 2009), where SNR (signal to noise ratio) is calculated as

$$SNR = 10log_{10}\frac{wrapped_{clean}}{wrapped_{noise}}. \tag{2.8}$$

Table 2.4 gives the performance of MC[4]-GI ($C|2|$, C1 , C0 , $C\bar{1}$) and SNAPHU using the measurement of Macro Mean Intersection over Union (MIoU), which is useful for measuring model performance where there is an imbalanced dataset. For the MIoU, the dice score (Equation 2.1) is calculated per class with macro-MIoU being the mean of all losses with each class contributing equally to the score. Therefore, if the model performs poorly at classifying the minority class, the overall score will be lowered rather than the minority class having little influence on the overall result if the class contribution were weighted by class size.

Table 2.5 show the measurements of macro-precision, macro-recall and macro-F1. The term macro before each statistic refers to each statistic being calculated on a per class

basis, where the class (for example C0) is treated as positive and all other classes are treated as negative (C1 , $C\bar{1}$, $C|2|$), and each class contributes equally to the overall final statistic (Pedregosa et al. 2011). The average statistic across all test interferograms is calculated to give the final score.

The IoU is calculated as

$$\text{Intersection over Union} = \frac{TP}{TP + FN + FP}, \tag{2.9}$$

where TP is true positive, FN is false negative and FP is false positive. The Intersection over Union is calculated per class, and the average taken over all classes, with each class contributing equally to the final score.

Macro-precision (Pedregosa et al. 2011) is the ratio of true positives to the total number of predictions of the positive class, where precision is calculated as

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{2.10}$$

The precision is calculated per class, and the average across all classes taken where each class contributes equally to the final score. It indicates whether the predictions of positive results are accurate and penalises wrongful positive case identification.

Macro-recall (Pedregosa et al. 2011) is the ratio of true positives to all positives in the groundtruth and measures the ability to capture all positive cases, and allows some tolerance of false positives. Recall is calculated as

$$\textit{Recall} = \frac{TP}{TP + FN}. \tag{2.11}$$

Like for Macro-Precision, the recall is calculated per class and averaged, with each class

contributing equally towards the final score. The score indicates whether the model has identified all positive cases, and penalises for cases where the positive cases are missed.

The Macro-F1 score (Pedregosa et al. 2011) is a combination of precision and recall, where F1 is calculated per class as

$$\mathrm{F1} = \frac{2TP}{2TP + FP + FN}, \tag{2.12}$$

and averaged across all classes, with each class contributing equally to the final score. A score of 1 indicates perfect results whilst a score of 0 is the worst case scenario.

For MC[4]-GI, the statistical scores were calculated relative to the number of classes within the corresponding target data set (4 classes), with each class giving equal weighting to the final score. SNAPHU was treated as a 4 class scenario where absolute gradients exceeding +2 were classified as $C|2|$.

Model	**Macro-MIoU Average**
SNAPHU	0.50
MC[4]-GI	0.89

Table 2.4 – Macro-MIoU is calculated in each of the horizontal and vertical directions and averaged for each of the models trained to classify gradients into 4 classes. SNAPHU was treated as a 4 class scenario ($\Delta k \geq 2 = 2$). Testing data set size: 400. Uniform Gaussian noise applied leading to a coherence range of 0.4 to 1.0.

Model	**Macro-Precision**	**Macro-Recall**	**Macro-F1**
SNAPHU	0.52	0.71	0.59
MC[4]-GI	0.91	0.98	0.94

Table 2.5 – Macro-Precision, Macro-Recall, and Macro-F1 scores for MC[4]-GI and SNAPHU tested on an unseen test data set. The level of noise varies across the test data set. SNAPHU was treated as equivalent to a 4 class classification task. Test data set size = 400

.

The MC[4]-GI approach outperforms SNAPHU across all metrics. The macro-recall for MC[4]-GI is 0.98 which indicates that the model performs well at classifying pixels into the majority classes (C0 , $C\bar{1}$, C1); a conclusion further confirmed by results shown in confusion matrixes in Figure 2.8, where the model has correctly classified more than

98% of C0 and the ±1 classes correctly. The correct classification of 94.9% of $C|2|$ by MC[4]-GI is a promising result, suggesting that for the majority of cases, larger phase changes will be correctly identified. Figure 2.8a demonstrates the use of MC[4]-GI is advantageous, given that SNAPHU has only correctly identified 0.2% of locations where pixel gradients would be classified as $C|2|$, with the majority identified to have a zero wrap count gradient.

Figure 2.8b conveys that less than 1% of the time, the majority classes C1 ,C0 and $C\bar{1}$ were misclassified as $C|2|$. The misclassification of majority classes into minority classes is preferable to the inverse scenario, simply because the minority class gradients are treated as likely sources of unwrapping error by the GI algorithm, and they will not be used in the subsequent integration step, meaning the risk of error introduction is reduced to 0. A potential issue may arise if unwrapping using the MC[4]-GI as if a substantial grouping of gradients are misclassified into the minority classes, leading to extensive regions of the interferogram being omitted from the subsequent integration step.

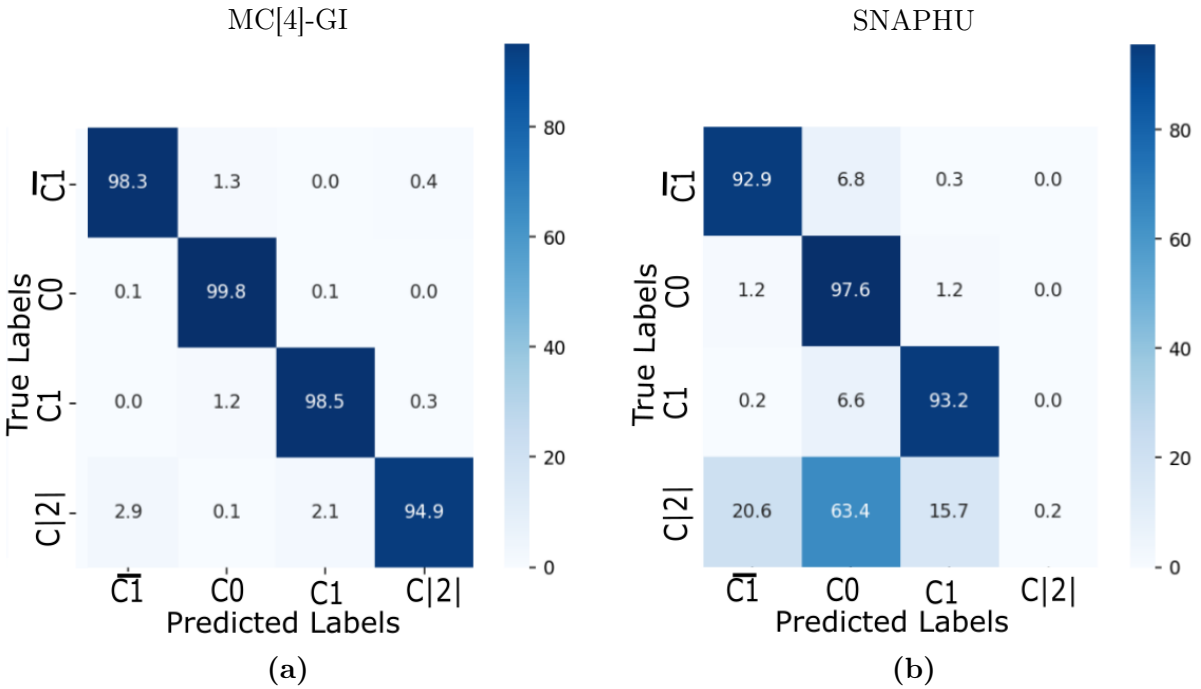
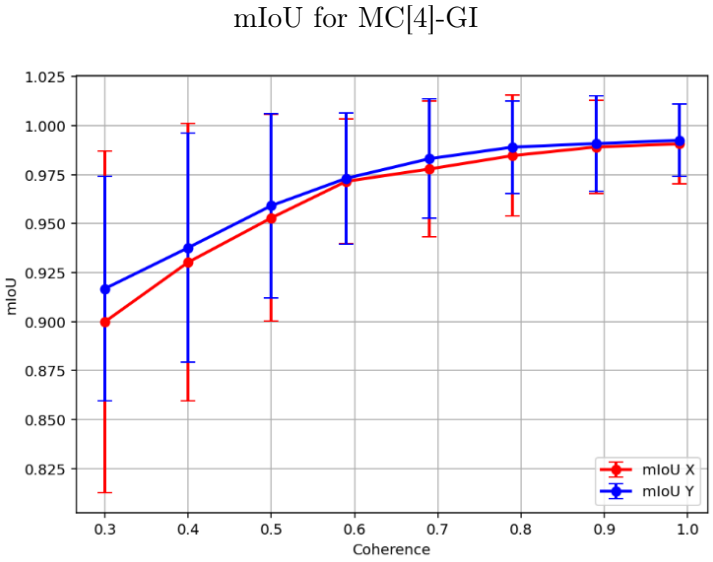
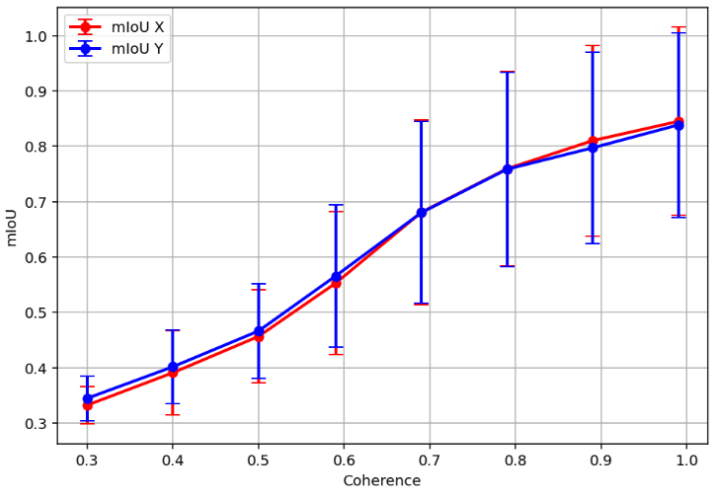


Figure 2.8 – Figure displaying the diffusion matrix for (a) MC[4]- GI and (b) SNAPHU. For MC[4]-GI model, the values are calculated based on the assumption that the absolute wrap count gradients exceeding 2 are $C|2|$. SNAPHU is treated as a 4 class case. Legend shows percentage.

Figure 2.9 indicates that MC[4]-GI always scores better than SNAPHU at all coherence levels. The accuracy of MC[4]-GI prediction declines more rapidly below 0.6 average coherence, however, in both the vertical and horizontal direction, the mIoU remains above 0.9 at all coherence levels. This indicates that MC[4]-GI has good noise robustness in the presence of increasing levels of uniform Gaussian noise.



(a)



(b)

Figure 2.9 – Comparison of macro-MIoU of wrap ground gradient estimation for (a) MC[4]-GI and (b) SNAPHU in the horizontal and vertical directions.

2.4.3 Assessment of the Full MC-GI Process

Ideally, the complete phase unwrapping approach, referred to as MC[4]-GI, would have been compared with other proposed deep learning methods. However, the code, pre-trained models and training dataset are not readily available and are non-trivial to re-implement. Instead, MC[4]-GI was evaluated against SNAPHU - a commonly used Statistical-cost, Network-flow Algorithm for Phase Unwrapping proposed by Chen et al. 2001, which maximises the conditional probability of unwrapped phase given the wrapped phase and coherence as conditions, a 3 class U-Net model (MC[3]-GI) and a 5 class U-Net model (MC[5]-GI). MC[5]-GI has C0 , C1 and $C\bar{1}$ but also expands upon MC[4]-GI by separating $C|2|$ into $C2^*$ ($\Delta k \geq 2$) and $C\bar{2}^*$ ($\Delta K \leq -2$).

It is important to acknowledge that there is a difficulty in quantitatively assessing the performance of unwrapping the synthetic interferograms. As highlighted before, the GI step of the proposed approach relies on pixels being connected and can lead to isolated regions being offset from one another. This will naturally increase the Root Mean Square Error (RMSE), despite the fact that MC-GI is better at unwrapping the deformation than using SNAPHU. For the purposes of this study, the most common rounded offset between the unwrapped result for each approach of MC-GI and the groundtruth unwrapped has been used as an offset factor to the MC-GI. I.e if the main difference between the unwrapped phase of the MC-GI approach and unwrapped groundtruth is 3, then 3 would be treated as the 0 difference. This ensures that the metric for measuring the MC-GI is not being unfairly influenced by an offset difference in wrap count value, and both the MC-GI and groundtruth unwrapped have the same baseline wrap count value.

Figure 2.10 highlights that across varying Gaussian noise levels, the MC-GI tends to outperform SNAPHU independent of the level of noise or number of classes of the MC-GI. The MC[3]-GI, MC[4]-GI and MC[5]-GI all have comparable performance when analysed quantitatively using Root Mean Squared Error.

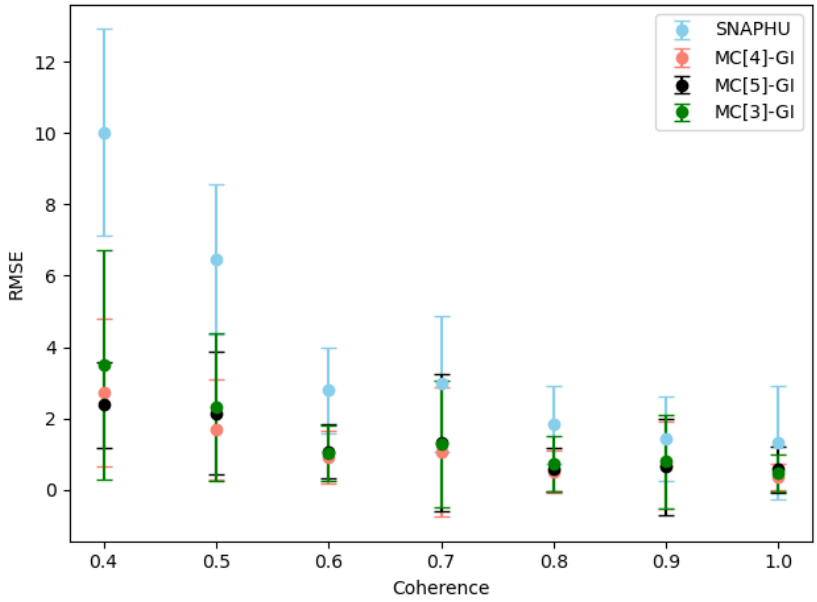
Figure 2.11 suggests that the MC-GI improves the accuracy of unwrapped phase when

large wrap count gradients are present compared with the unwrapped phase returned by SNAPHU. RMSE is calculated per interferogram between the groundtruth and unwrapped phase. The mean RMSE is plotted for all interferograms of a given coherence in Figure 2.11. MC[4]-GI most consistently leads to a lower RMSE result than MC[3]-GI and MC[5]-GI. This suggests MC[4]-GI is the most reliable option for phase unwrapping the synthetic interferograms.

When testing the approaches, I looked to identify whether a pattern existed between the deformation pattern and the number of classes best suited for an unwrapped phase accuracy. Figure 2.12 and the corresponding plots in Figure 2.13 demonstrate a representation of where each approach of MC[3]-GI, MC[4]-GI and MC[5]-GI each performed best at returning a highly accurate unwrapped phase when the wrapped interferograms contained minority class gradients. Figure 2.13 shows the RMSE for each wrapped interferogram as the noise level changed.

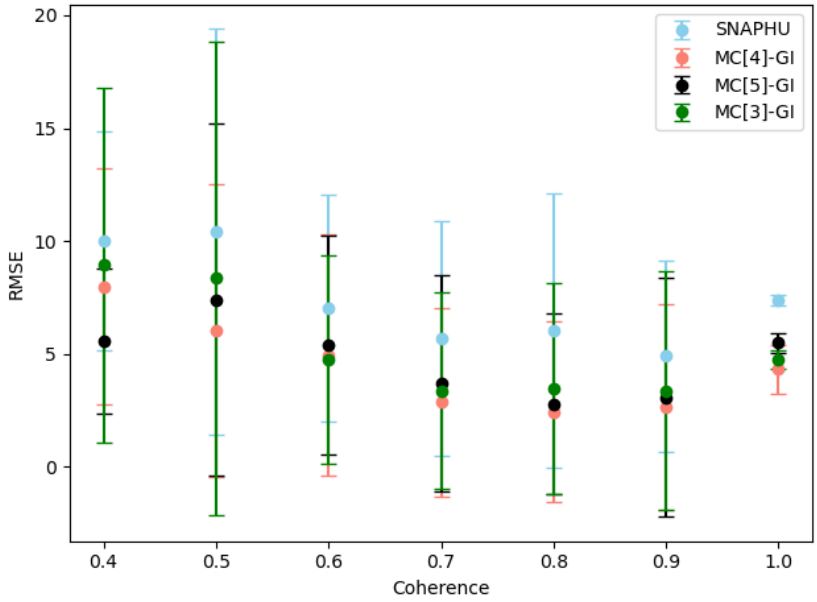
MC[3]-GI and MC[4]-GI both have comparable performances for the types of wrapped interferograms represented in Figure 2.13a-d, and consistently present lower RMSE for a given interferogram for all noise levels. In wrapped interferograms which contain missing values, such as that shown in Figure 2.12e, MC[4]-GI outperforms MC[3]-GI and MC[5]-GI, suggesting the use of a catch all two class better supports the GI approach to get the most accurate unwrapped result in this scenario. Where wrapped interferograms have a deformation pattern which appears to converge to a point, as seen in Figure 2.12b and c, there is no clear pattern as to which method results in the optimal result (see the unwrapped example in Figure 2.14). Overall, the results indicate that the inclusion of at least one additional class does not inhibit the GI performance and, in some cases, leads to an improvement in the accuracy of the unwrapped result for synthetic interferograms further emphasising that the MC[4]-GI approach is likely the most reliable unwrapping approach.

The MC-GI performance declines when Gaussian noise is generated to simulate low noise levels, though the model does still perform consistently better than SNAPHU (see Figure



(a)

Figure 2.10 – Figure of the mean Root Mean Squared Error against coherence for 400 wrapped interferograms of coherence 0.4 to 0.99. The error bars show 1 standard deviation.



(a)

Figure 2.11 – Figure of Root Mean Squared Error against coherence for 324 wrapped interferograms of coherence 0.4 to 0.99. Each interferogram contains examples of large wrap count gradient jumps ($C|2|$) in the horizontal and/or vertical direction. Plotted with the average RMSE per coherence and the error bars showing 1 standard deviation of the RMSE for the interferograms at a given coherence level.

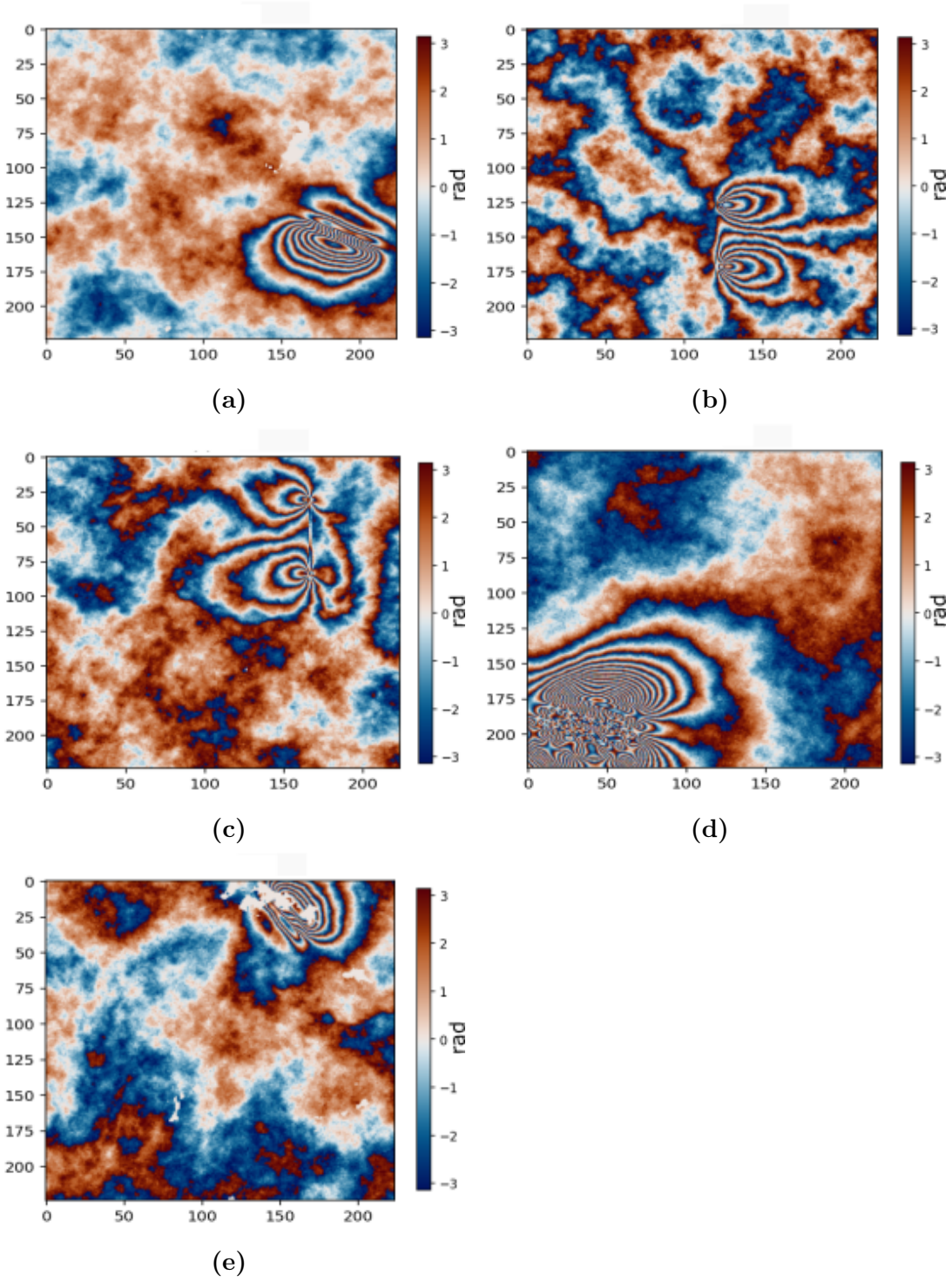


Figure 2.12 – Figure displaying different examples of interferograms where the minority class $C|2|$ is present.

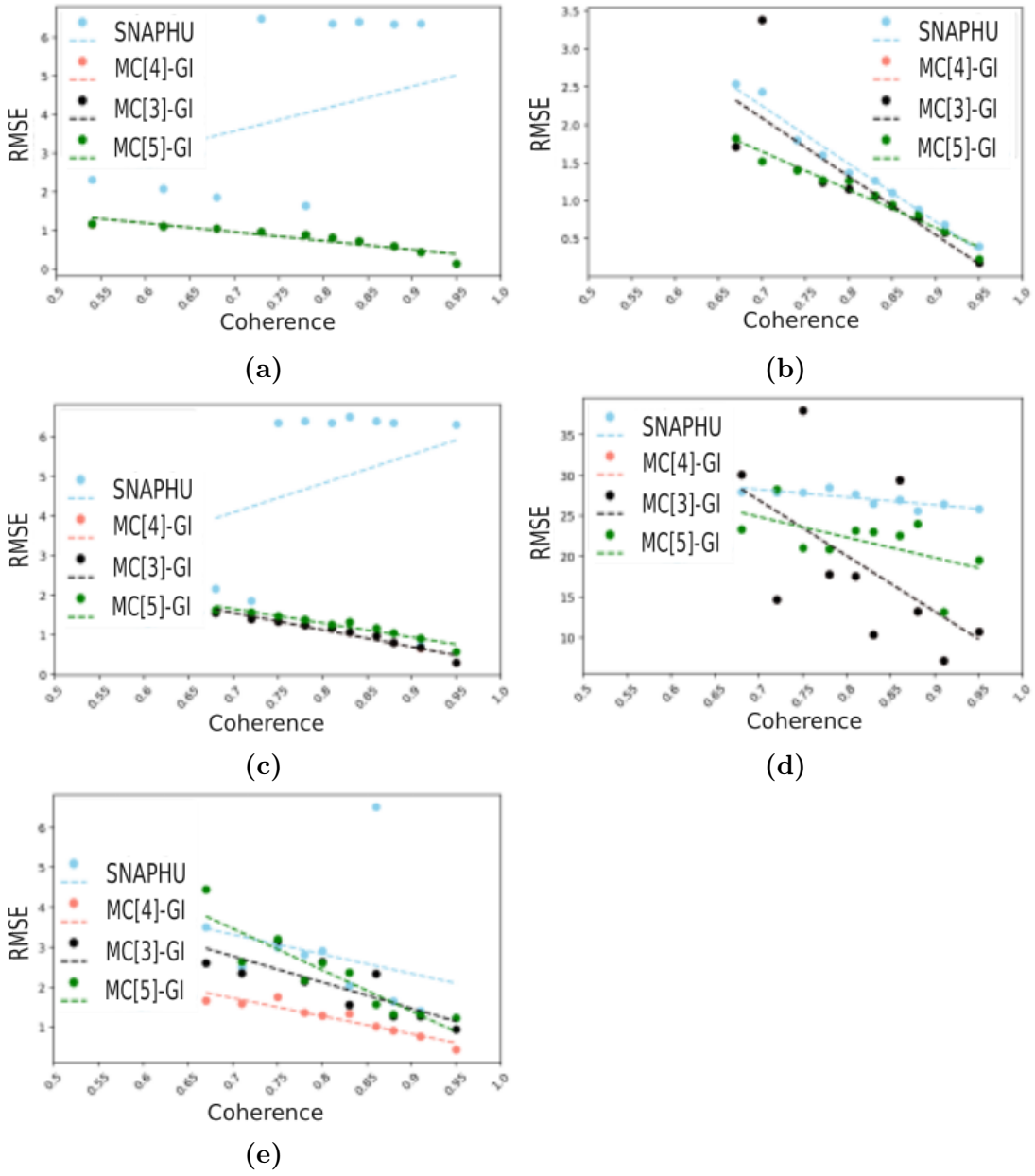


Figure 2.13 – Figure plotting the RMSE against coherence for each interferogram in Figure 2.12. For example, Figure 2.12a matches a in this figure. Unwrapping techniques applied are SNAPHU, MC[3]-GI, MC[4]-GI and MC[5]-GI, and a linear regression line plotted for each approach.

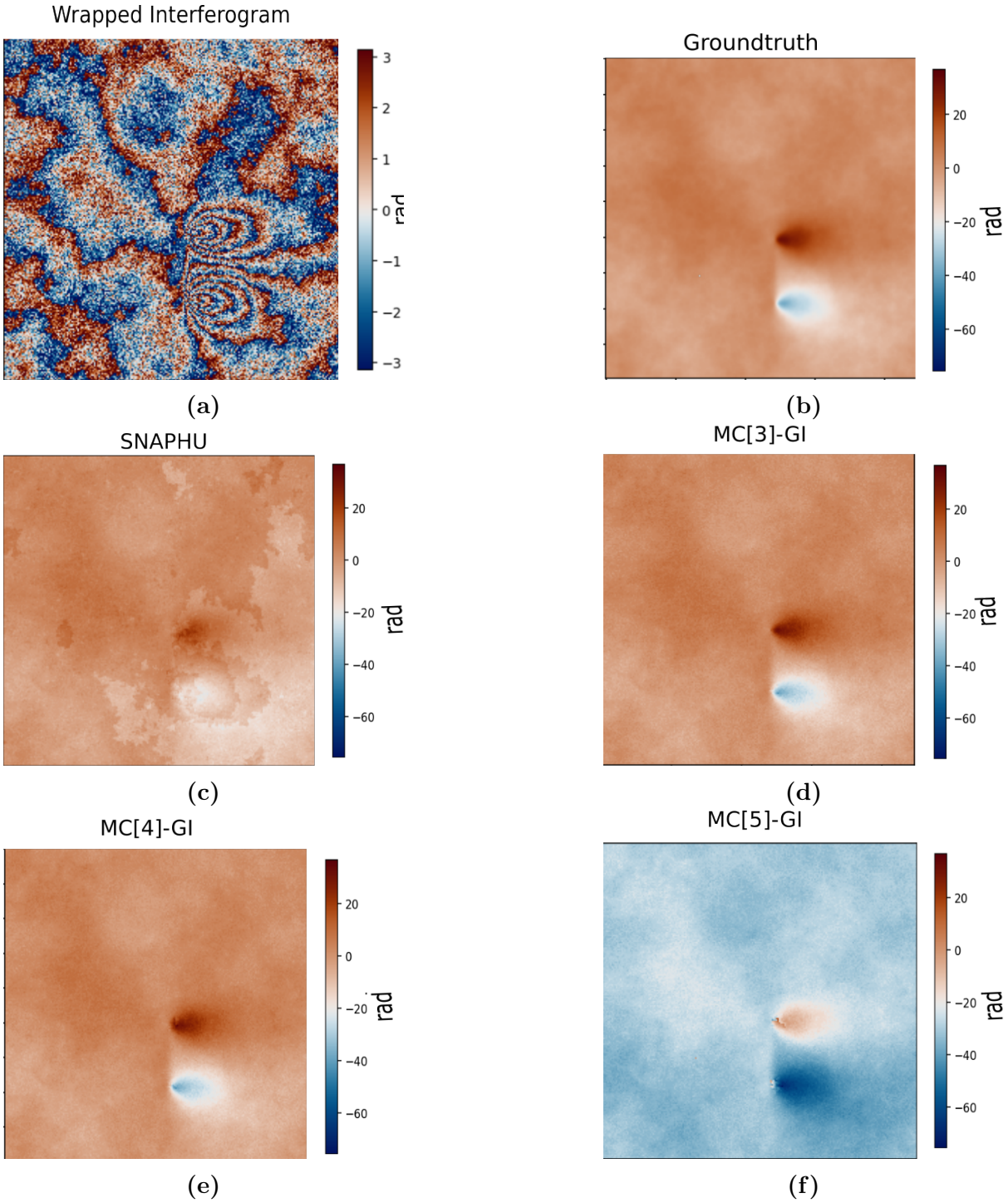


Figure 2.14 – Figure of (a) Wrapped interferogram, (b) Groundtruth unwrapped interferogram, (c) SNAPHU unwrapped interferogram, (d) MC[3]-GI unwrapped interferogram, (e) MC[4]-GI unwrapped interferogram, (f) MC[5]-GI unwrapped interferogram for a case where the accuracy of the unwrapped results are comparable for the MC-GI models.

2.9).

2.4.4 Further Assessment of the Complete MC[4]-GI approach on Synthetic Interferograms

Further analysis was performed on the MC[4]-GI performance in comparison to SNAPHU.

Figure 2.15 and 2.16 each demonstrate for various deformation patterns, fringe densities, large noise patch size and shape, when no Gaussian noise has been added, MC[4]-GI has a better unwrapped result than SNAPHU.

The MC[4]-GI method is particularly advantageous compared to SNAPHU, where dense fringes or complex fringe patterns exist. In Figure 2.15e, SNAPHU has unsuccessfully unwrapped the phase, mainly in the area where deformation has occurred next to a large noise region. The MC[4]-GI would be unaffected by the large noise patch given that it is ignored in the GI step, and this likely accounts for why the MC[4]-GI has been so successful in unwrapping such examples of deformation. Figure 2.16 highlights that although the MC[4]-GI has incorrectly unwrapped the densest area where the wrapped phase is beginning to alias at the centre of the deformation, presumably due to assigning $C|2|$ and thus the region being excluded from unwrapping leading to an incorrect result for these pixels, the use of the MC[4]-GI led to an overall improved unwrapped result in the wider deformation region compared to SNAPHU.

However, issues have arisen where a large noise field completely separates two coherent patches (Figure 2.17c) so that no connection is possible. This is a direct result of the MC-GI deliberately excluding noise pixels and minority class gradients, which may occur in large groupings in the GI step, thereby preventing connections from forming. Whilst each isolated region may be correctly unwrapped relative to pixels within the same region, the two disconnected regions are incorrectly offset relative to each other. However, Figure 2.17e demonstrates that SNAPHU also had this issue. To reduce the impact on the overall result, analysis has been performed on each isolated region, with results demonstrating that the isolated regions are correctly unwrapped relative to the points within the region

1341 and offset with respect to disconnect pixels.

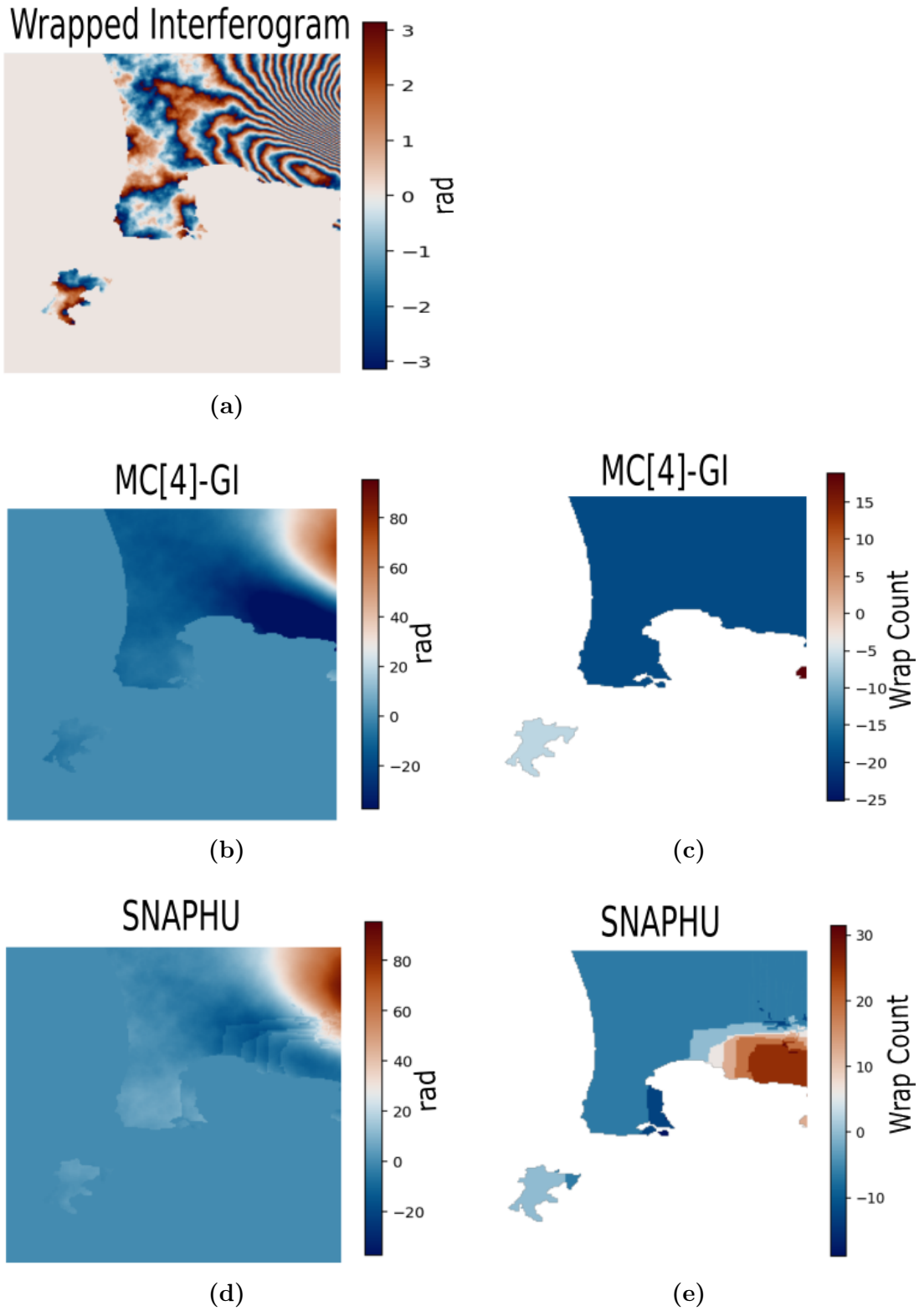


Figure 2.15 – Unwrapped example for (a) a wrapped interferogram with large noise patches, (b) shows the unwrapped results returned using MC[4]-GI, (c) shows the difference between the groundtruth wrap count and the predicted wrap count, (d) shows the groundtruth unwrapped phase, (e) shows the SNAPHU unwrapped phase prediction, (f) shows the difference between the SNAPHU wrap count and the groundtruth wrap count.

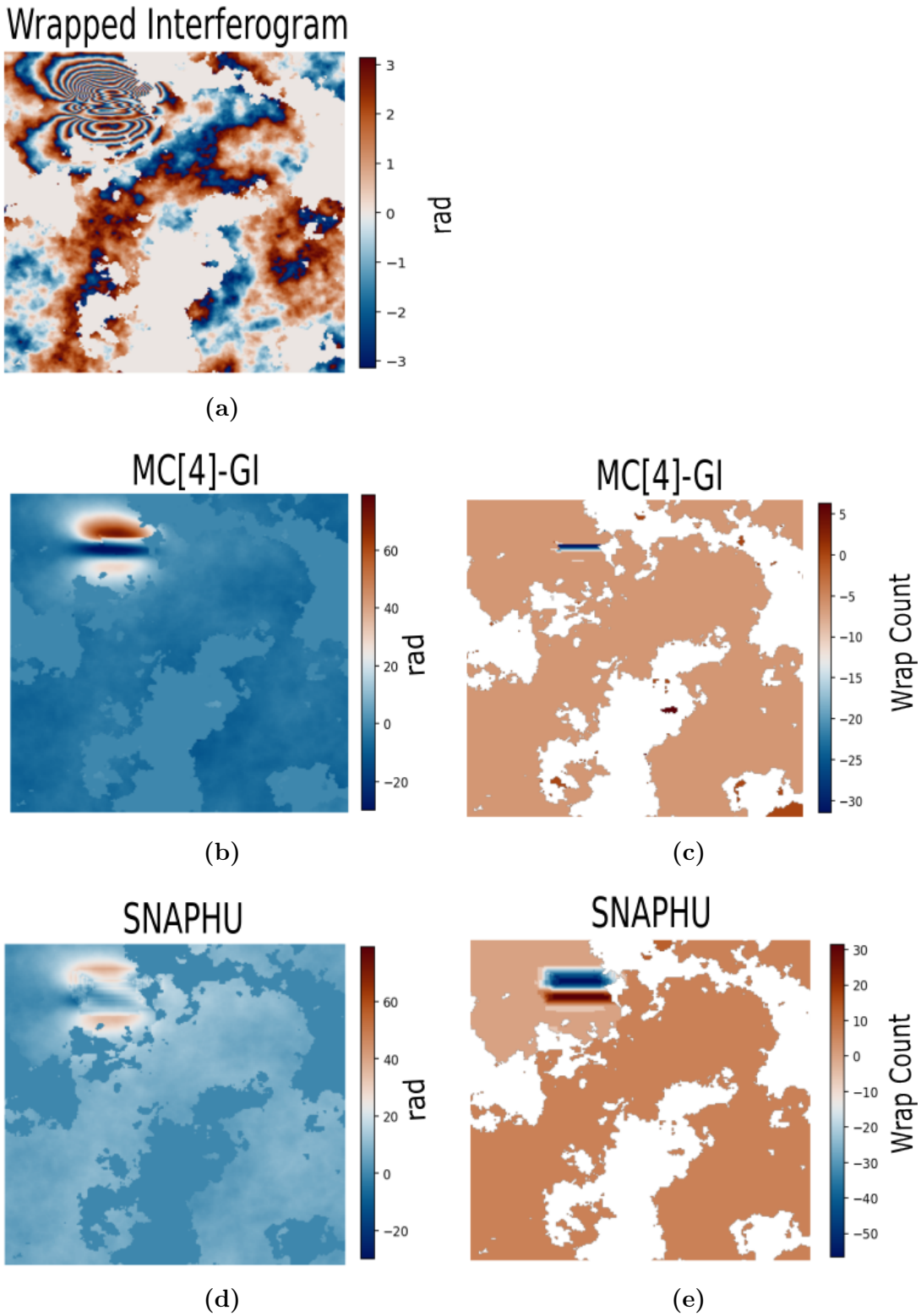


Figure 2.16 – Unwrapped example for (a) a wrapped interferogram with dense fringes, (b) shows the unwrapped result for MC[4]-GI (c) shows the difference between the groundtruth wrap count and the MC[4]-GI wrap count, (d) shows the SNAPHU unwrapped phase prediction, (e) shows the difference between the SNAPHU wrap count and the groundtruth wrap count.

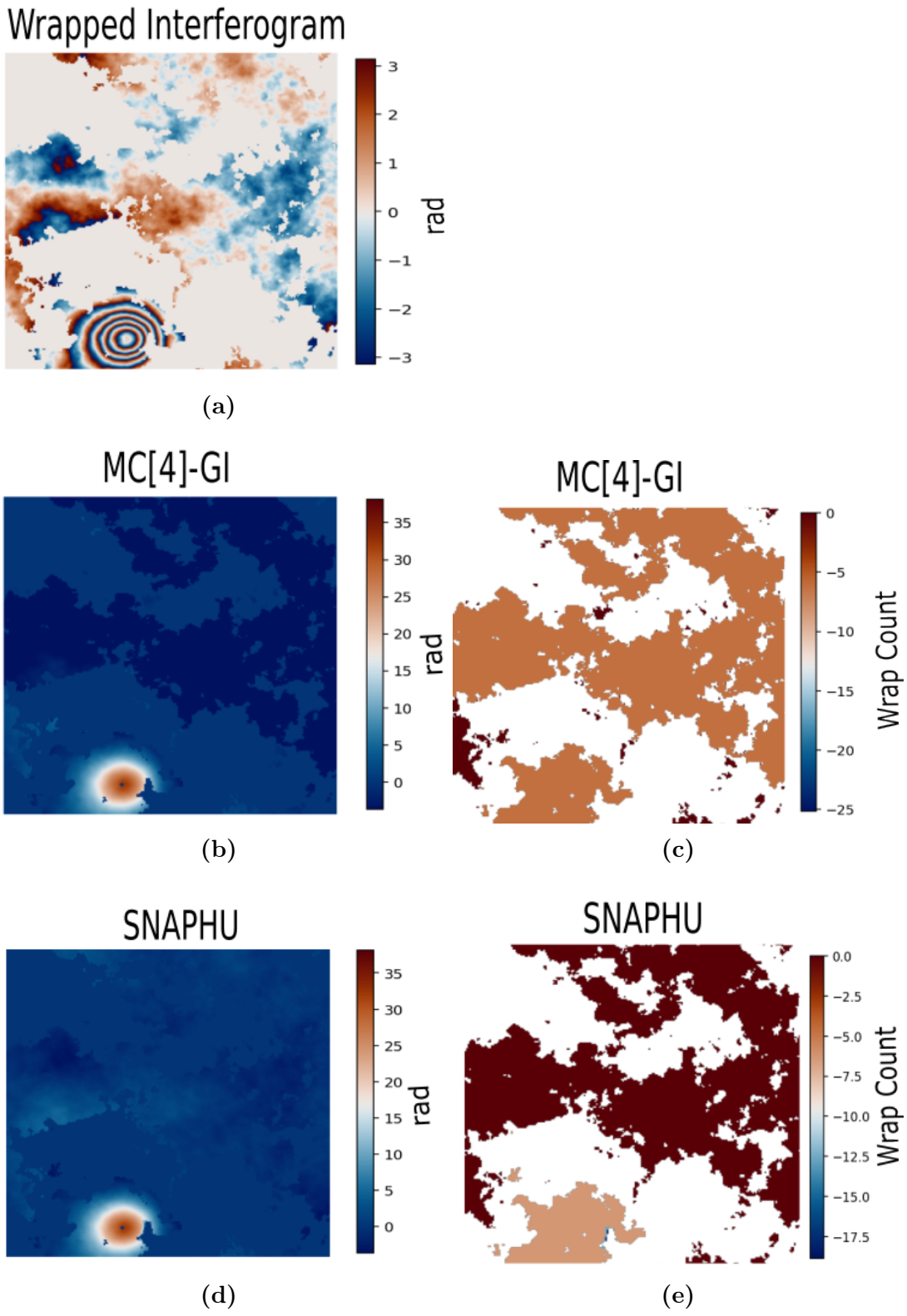


Figure 2.17 – Unwrapped example for (a) a wrapped interferogram where it has reduced unwrapped phase performance due to separated regions, (b) shows the unwrapped result from MC[4]-GI, (c) shows the difference between the groundtruth wrap count and the predicted wrap count for MC[4]-GI, (d) shows the SNAPHU unwrapped phase prediction, (e) shows the difference between SNAPHU wrap count and the groundtruth wrap count.

2.4.5 Loop Closure Error Analysis Explanation

Until now, the MC-GI approach has been assessed purely using synthetic data where a groundtruth is available. The groundtruth for real InSAR data, while not impossible to attain, is more difficult to attain due to its reliance on an accurately unwrapped interferogram, which is time-consuming and may not always be possible, particularly in cases where phase unwrapping errors are likely to occur. Instead, the unwrapping performance is assessed using the loop closure error approach.

Loop closure error calculations require 3 interferograms made from 3 SAR acquisitions taken over the same area at three timestamps (see the example shown in Table 2.6). Loop closure is calculated per pixel, where an absolute loop closure error $< \pi$ indicates correct phase unwrapping has occurred for that pixel across all 3 interferograms

$$\text{Loop Closure Error} = \text{Interferogram 1} + \text{Interferogram 3} - \text{Interferogram 2}. \quad (2.13)$$

An example case for Equation 2.13 is that Interferogram 1 is generated from SAR acquisitions at date 04/01/2022 and 28/01/2022, Interferogram 2 is generated from SAR acquisitions taken at time is 04/01/2022 and 21/02/2022 and Interferogram 3 is generated from SAR acquisitions taken at time 28/01/2022 and 21/02/2022 for some Frame (e.g: 115A 05409 131313). However, this assumes that errors in two of the interferograms do not cancel each other out.

Each of the interferograms has been separately unwrapped. Given that the unwrapped phase is equal to the wrapped phase plus some integer number offset (Equation 1.2, the offset could be any integer value, with the remaining interferogram unwrapped relative to this. This can result in each of the 3 interferograms used in the loop closure error calculation being offset to a different integer value, and thus a re-adjustment according

Frame	Time
115A 05409 131313	20220104 - 20220128
115A 05409 131313	20220104 - 20220221
115A 05409 131313	20220128 - 20220221

Table 2.6 – Table of frame id and time example

to some reference pixel is normally used to ensure that at that given point, the summation of the loop closure is 0. In theory, if the remaining interferogram is correctly unwrapped relative to that pixel, it should result in the loop closure error being 0 across all points. Rather than perform this adjustment prior to the loop closure calculation, loop closure was performed, followed by the addition of an integer value to shift the loop closure values so that the majority of pixels were within the 0 loop closure error range. Prior to this, pixels identified as having a coherence lower than the coherence threshold in at least one interferogram are masked and not used in the accuracy calculation.

Loop closure error is converted to a percentage of the interferogram, where the loop closure summation is 0 for the entire interferogram, by calculating the percentage of pixels where the loop closure error has not exceeded the tolerance threshold.

The unwrapping performance of SNAPHU and MC[4]-GI were analysed for real interferograms with varying levels of phase noise. The mask is generated by setting a coherence threshold of 0.1 based upon the coherence file available on the LicSAR portal (Lazecký et al. 2020).

2.4.6 Assessment of MC-GI unwrapping performance on Sentinel-1 Interferograms

The interferograms and coherence maps used in this section were taken from the LiCSAR portal, where they had been generated using the automated workflows from the COMET-LiCSAR system (Lazecký et al. 2020). A coherence threshold was used to mask the wrapped interferogram prior to being given to the MC-GI model and as input to

SNAPHU.

Figure 2.18 and 2.19 both demonstrate that for interferograms with little phase noise, the unwrapped result from the MC-GI and SNAPHU both have a loop closure error close to 0. In Figure 2.19j, it can be seen that whilst the MC[4]-GI method performed well generally across the entire interferogram, the unwrapped phase in the separated area has been demonstrated to have been impacted by being isolated from the main body of the wrapped phase (i.e being a separated regions). As seen in the synthetic testing dataset, the MC-GI requires improvement to correct for such eventualities.

When I consider the displacement caused by an earthquake illustrated in Figure 2.20a, it can be seen that MC[4]-GI wrap count map shows more defined fringe edges in Figure 2.20b than the SNAPHU wrap count map in Figure 2.20c. Where the pixels within the deformation region are not isolated, the MC[4]-GI has returned a gradient of deformation which is consistent with what is expected, given the wrapped interferogram, when compared with SNAPHU's unwrapped result over the same area. Promisingly, the loop closure error over the deformation area is 10.06% when unwrapped using MC[4]-GI, compared to a 12.08% loop closure error when unwrapped by SNAPHU. Though MC[4]-GI has only identified gradients to be C0 , C1 or $C\bar{1}$, which is likely resultant from the highest gradient region being masked prior to being given as input to the model (see Figure 2.21), MC[4]-GI still performs a more accurate unwrapping process than SNAPHU in the region where the highest deformation has occurred, indicated by the low loop closure error shown in Figure 2.20d.

The majority of the error when unwrapped by MC[4]-GI is focused within the boxed region of Figure 2.20a, unlike SNAPHU, where the error is concentrated around the central deformation area. This area does not have a clear fringe pattern, appearing noisy, and thus it is hard to identify exactly where the fringe edge should be. This likely led to incorrect unwrapping in the Greedy Phase Integrator, which relies on the clear fringe edge patterns for good results and whose performance declines when such attributes are not present. This further suggests that the Greedy Phase Integrator is limited in regions

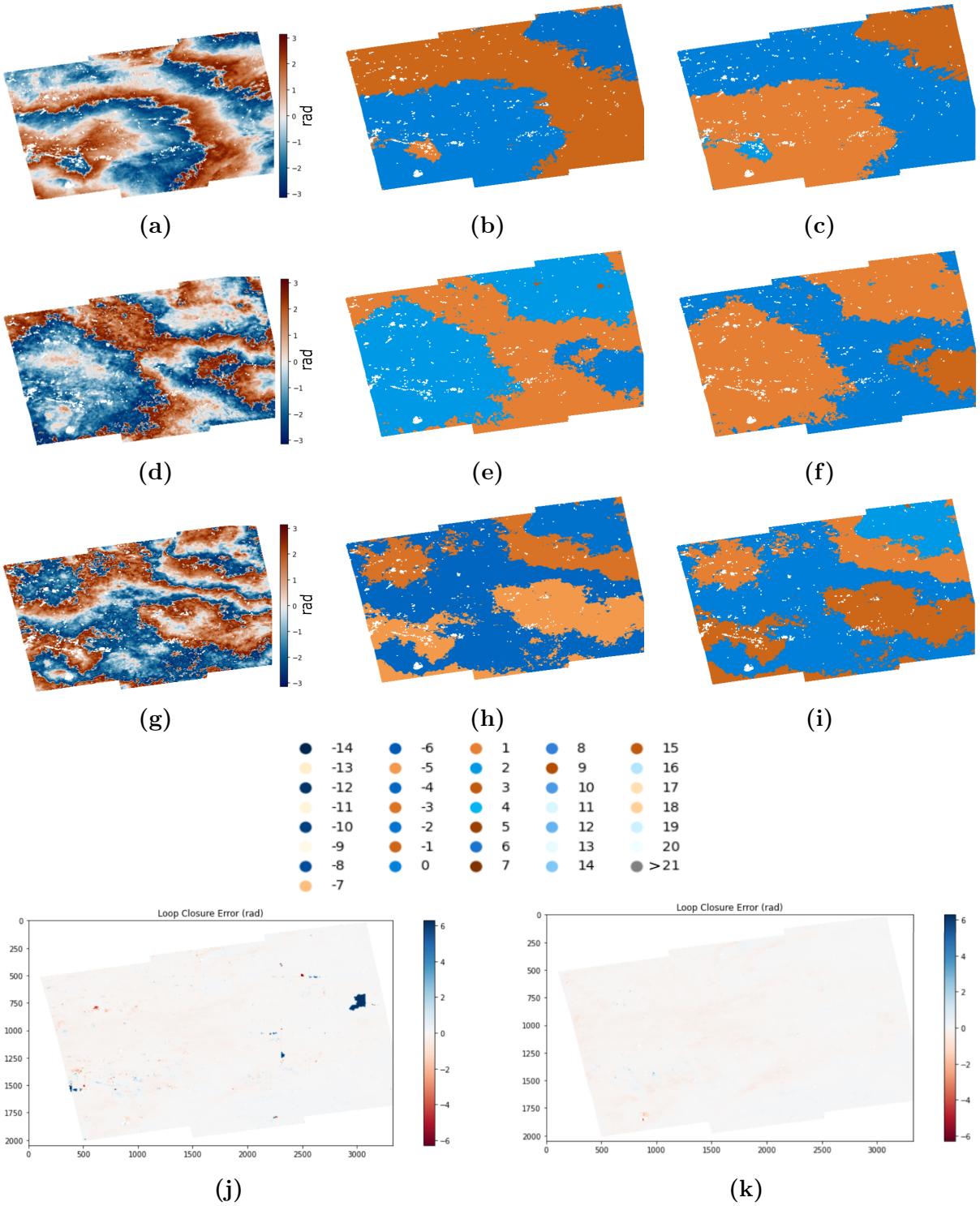


Figure 2.18 – Figure showing unwrapped interferograms for LicSAR Frame 172A 04842 091010, SAR acquisitions dated 2024/01/19, 2024/02/12 and 2024/02/24. (a) Wrapped Interferogram from acquisitions dated 2024/01/19 and 2024/02/12, (b) Wrap Count Result using MC[4]-GI for (a), (c) Wrap Count Result using SNAPHU for (a), (d) Wrapped Interferogram from acquisitions dated 2024/01/19 and 2024/02/24, (e) Wrap Count Results using MC[4]-GI for (d), (f) Wrap Count Results using SNAPHU for (d), (g) Wrapped Interferogram from acquisitions dated 2024/02/12 and 2024/02/24, (h) Wrap Count Results using MC[4]-GI for (g), (i) Wrap Count Results using SNAPHU for (g), (j) Loop Closure Error plot for the 3 interferograms unwrapped using MC[4]-GI (0.88%), (k) Loop Closure Error plot for the 3 interferograms unwrapped using SNAPHU (0.01%).

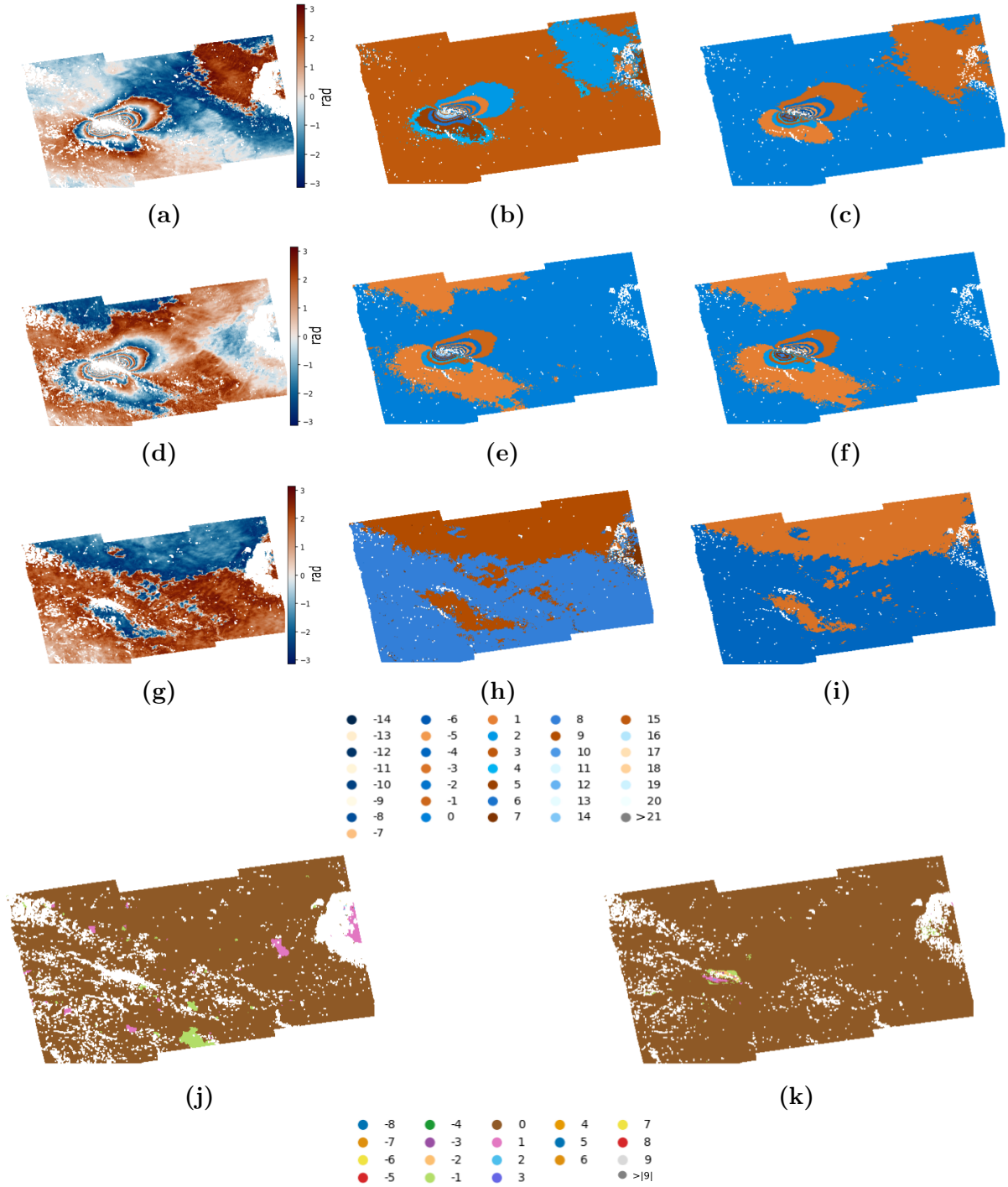


Figure 2.19 – Figure showing SNAPHU and MC[4]-GI approach output for the frame 128*A* 05172 131313. (a) Wrapped interferogram from SAR acquisitions 05/01/2022 and 17/01/2022, (b) Wrap count values returned from the MC[4]-GI when (a) is unwrapped, (c) Wrap count values returned by SNAPHU when (a) is unwrapped (d) Wrapped interferogram from SAR acquisitions 05/01/2022 and 29/01/2022, (e) Wrap count values returned by MC[4]-GI when (d) is unwrapped, (f) Wrap count values returned by SNAPHU when (d) is unwrapped, (g) Wrapped interferograms from SAR acquisitions 17/01/2022 and 29/01/2022, (h) Wrap count values returned by SNAPHU when (g) is unwrapped, (i) Wrap count values returned by SNAPHU when (g) is unwrapped, (j) Loop closure error when the 3 interferograms are unwrapped by MC[4]-GI (6.32% error), (k) Loop closure error when the 3 interferograms are unwrapped by SNAPHU (2.66% error).

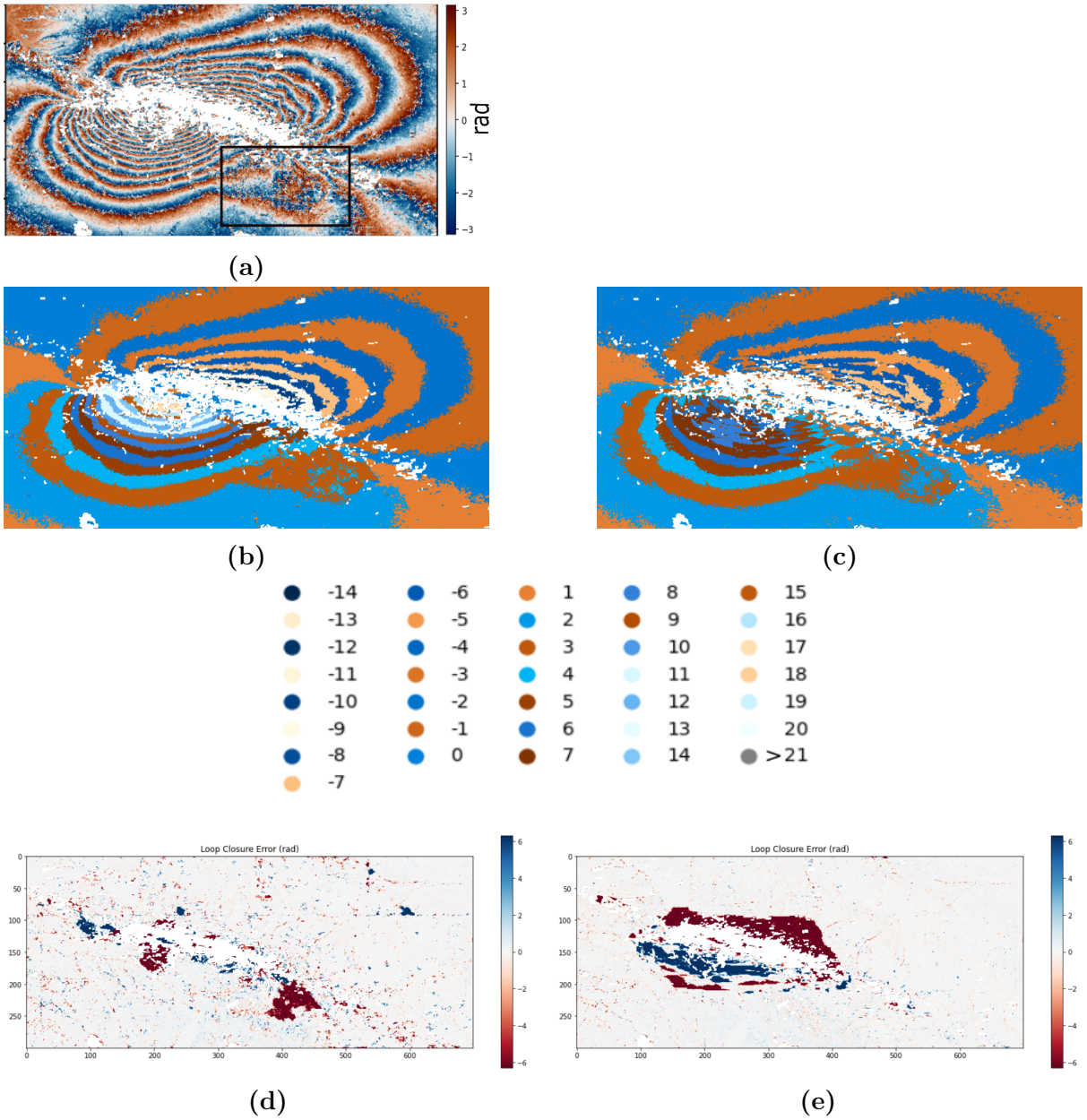
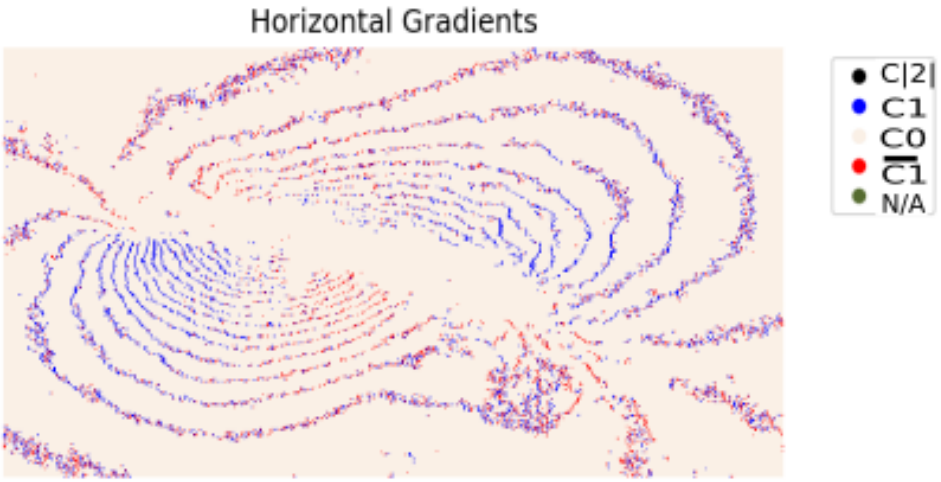
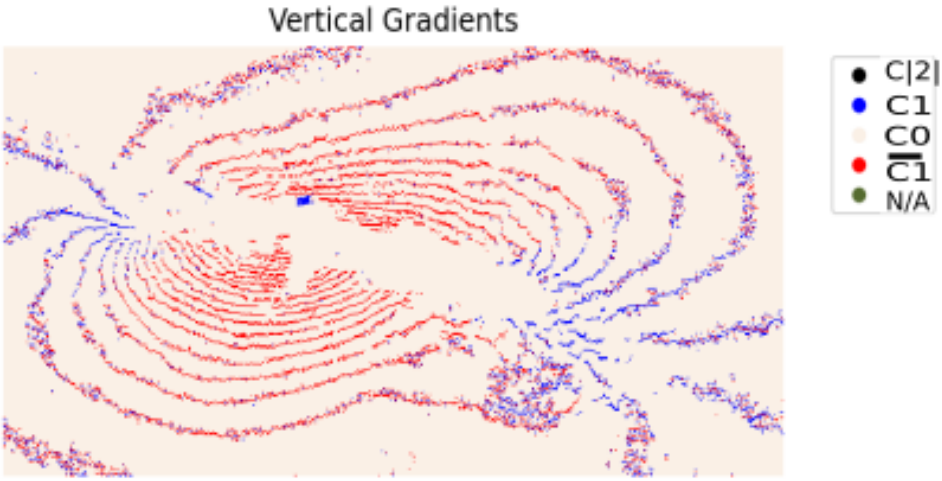


Figure 2.20 – Figure showing the wrap count values returned in the deformation by SNAPHU and MC[4]-GI for the interferogram frame 128*A* 05172 131313 dated 2022/01/05 – 2022/01/29. (a) Wrapped interferogram, (b) MC[4]-GI returned wrap count, (c) SNAPHU returned wrap count, (d) Loop Closure Error for b (10.06%) (e) Loop Closure Error for c (12.08%).



(a)



(b)

Figure 2.21 – Figure demonstrating the Multi-Class [4] Model classification for the horizontal and vertical wrap count gradients ((a) and (b) respectively) for Figure 2.20a

of high phase noise where it cannot identify and integrate across smooth fringe patterns. For MC[4]-GI to work optimally, some filtering is required; however, this can introduce smoothing through fringes, moving the problem from being a high C1 and $C\bar{1}$ occurrence to poor integration due to the loss of fringe edges.

Overall, MC[4]-GI has demonstrated improved capability in unwrapping high fringe rates at the centre of deformation in comparison to SNAPHU for interferograms where little phase noise is present.

The previous cases were relatively easy to unwrap as they had high average coherence

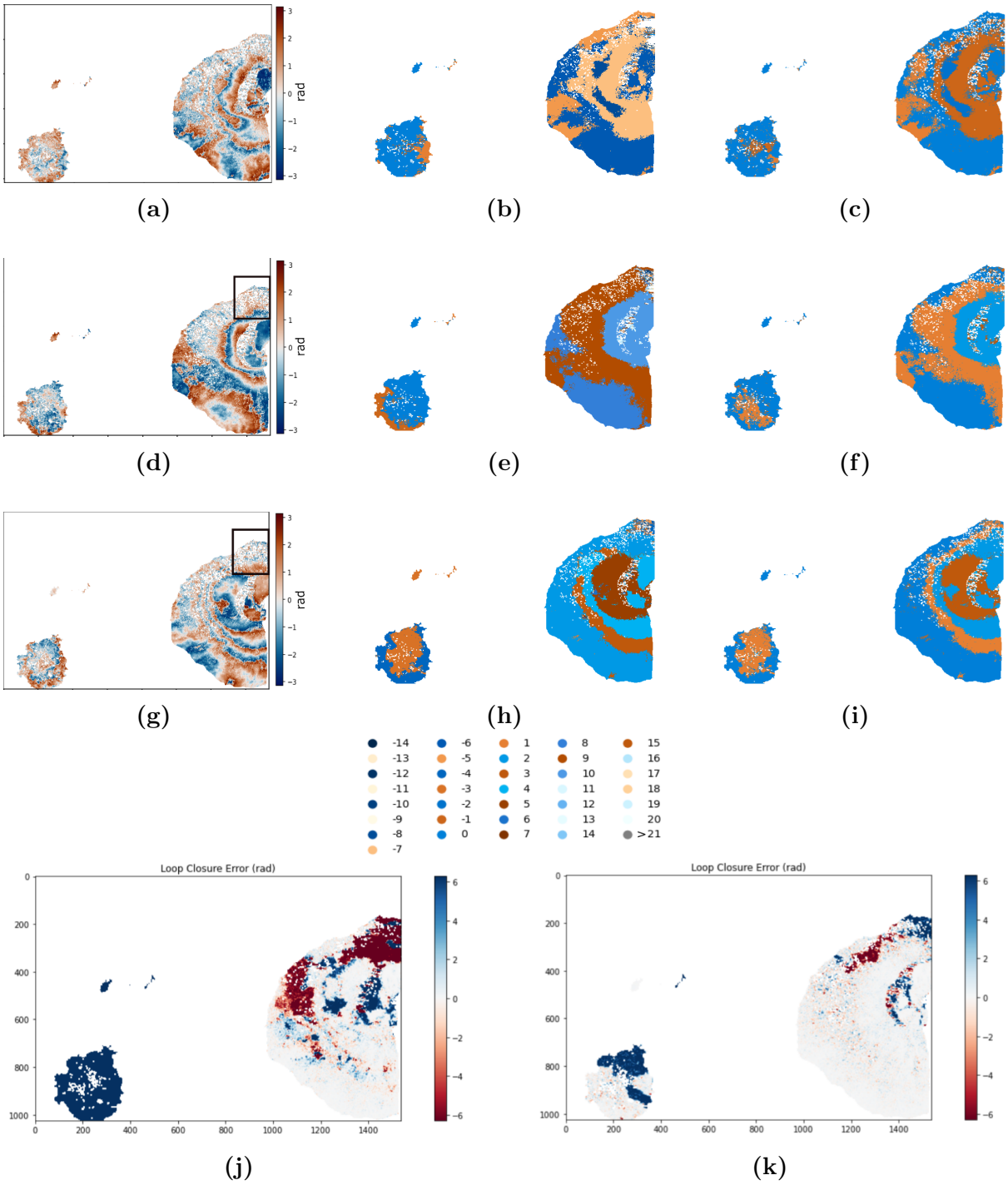


Figure 2.22 – Unwrapped results for Frame 140D SM FGbr s4 test (acquisitions dated 2015/06/12, 2015/09/16 and 2016/03/02). (a) Wrapped Interferogram (2015/06/12 - 2015/09/16), (b, c) Wrap Count returned when (a) is unwrapped using MC[4]-GI and SNAPHU respectively, (d) Wrapped Interferogram (2015/06/12 - 2016/03/02), (e, f) Wrap Count returned when (d) is unwrapped using MC[4]-GI and SNAPHU respectively, (g) Wrapped Interferogram (2015/09/16 - 2016/03/02), (h,i) Wrap Count returned when (g) is unwrapped using MC[4]-GI and SNAPHU respectively, (j) Loop Closure error for the 3 interferograms when unwrapped using MC[4]-GI (43.55%) (k) Loop Closure error for the 3 interferograms when unwrapped using SNAPHU (20.44%) Mean coherence for the 3 interferograms is 0.26.

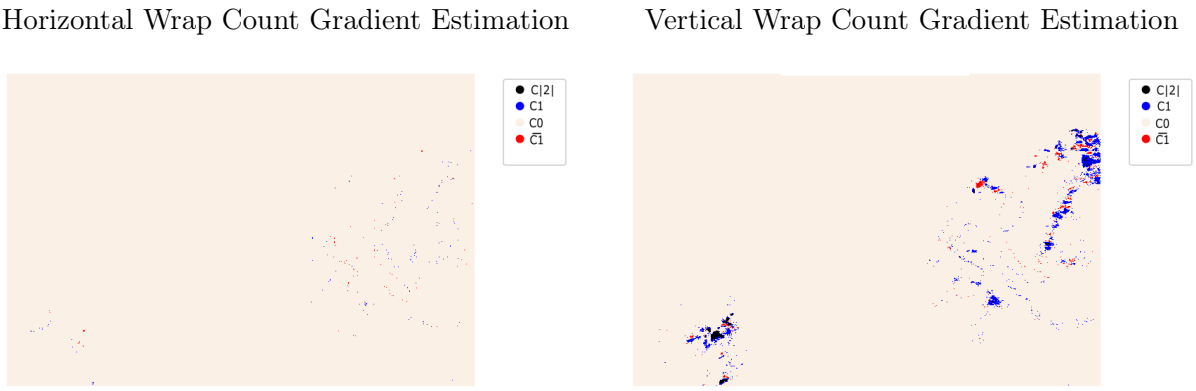


Figure 2.23 – Figure showing the MC[4]-GI predicted gradients in the (a) horizontal direction and (b) vertical direction for wrapped interferogram 1 (Figure 2.22a).

across the interferogram and deformation events where, whilst fringe rate is high, there is minimal additional phase noise. Figure 2.22 displays interferograms which are more challenging to unwrap, demonstrating clearly within the boxed region the negative impact of noise covering fringe edges. Again, the coherence threshold mask has been set to 0.01 to allow as much of the interferogram to be unwrapped as possible. In Figure 2.22i, SNAPHU has successfully unwrapped the expected deformation ring patterns on the larger island, whereas phase noise located in the boxed region of Figure 2.22g has negatively impacted the MC-GI to the extent that there is a disrupted wrap count ring in Figure 2.22h. Where coherence is higher outside the boxed region towards the bottom half of the larger island across all 3 interferograms (Figure 2.22a,d and g), MC-GI has unwrapped the phase with comparable success to SNAPHU.

Despite the smaller island having an average coherence of 0.06 for each of the 3 wrapped interferograms, MC[4]-GI returns spatial patterns with a consistent loop closure error, which indicates the phase has been unwrapped correctly across the 3 interferograms for the smaller island. The loop closure error value of 6 is due to being offset from the larger island, whereas when unwrapped using the SNAPHU approach, for the small island, there are two prominent loop closure error values (0 and 6), indicating the phase has been incorrectly unwrapped for at least 1 of the 3 interferograms.

In the boxed region of Figure 2.23, MC[4]-GI has classified the wrap count gradients as

C0 rather than as $C|2|$, C1 or $C\bar{1}$, which would indicate a fringe edge. The model has limited denoising capabilities due to not being trained to complete this task, leading to a dominance of C0 in the model wrap count gradient map shown in Figure 2.23, as MC[4]-GI is treating the region as a large noise patch. During the downstream Greedy Phase Integrator step, C0 indicates no change in wrap count. Thus, the large region where an edge is missing has led to an erroneous wrap count return due to the absence of a wrap count step change implementation and a smoothing of the region. The lack of distinction between one wrap count cycle start and finish ultimately means a phase unwrapping error has occurred and propagated as shown in the boxed region of Figure 2.22.

Overall, the MC[4]-GI model performs comparably to SNAPHU in regions of Sentinel-1 interferograms where there is a minimal amount of phase noise, and shows improved unwrapping accuracy when applied to large scale deformation events which have a high fringe rate at the centre of the deformation, as long as there is a limited level of phase noise.

2.5 Conclusion

In this Chapter, I have discussed my proposed MC[4]-GI phase unwrapping approach, which is a complete unwrapping process. This work has investigated the addition of minority classes ($C|2|$,C$\bar{2}^*$, $C2^*$) to the overall unwrapped result. With respect to gradient estimation, MC[4]-GI outperforms SNAPHU in the correct identification of locations where the wrap count gradient exceeds ± 1 when uniform Gaussian noise with a range of standard deviations to mimic various noise levels is applied. The MC[4]-GI is shown to differentiate between aliasing pixels and noise pixels, and de-noises uniform Gaussian noise during the pixel gradient identification step.

A novel Greedy Phase Integrator algorithm is proposed, which mimics human intuition by prioritising the highest quality pixel with the greatest associated model prediction

certainty pixels, followed by the next highest quality pixel, and so on. There is no need for wrap count pixels to be connected to previously returned wrap count pixels. MC[4]-GI shows good performance over a range of uniform Gaussian noise levels and was successfully able to denoise and identify gradients underneath the uniform Gaussian noise. The level of denoising capabilities is limited, as suggested by the decline in gradient prediction as the noise level increases. Such a decline in gradient classification accuracy in turn impacted the Greedy Phase Integrator algorithm. However, overall MC[4]-GI performed on average better than SNAPHU and a 3 class model input unwrapping a range of synthetic interferograms.

When applied to real interferograms of high average coherence, the model performs comparably to SNAPHU. However, in low coherence regions, MC[4]-GI exhibits a higher degree of unwrapping failure than SNAPHU. When exploring the gradient results for real interferograms, the model struggles to differentiate between noise and minority class gradients and is not able to denoise the wrapped phase. The MC-GI method is highly successful when little noise is present in the gradient data and the edges have been defined without large gaps. The gradient identification misclassifies cases where there is a rapid change between two wrap counts within small regions (high decorrelation). When taken against the good performance at denoising synthetic data generally, this is suggestive that the distribution of the noise data applied to synthetic data is different to the noise plus real InSAR signals, resulting in the difference in performance in these two scenarios.

Repeatedly demonstrated throughout this Chapter is the need for improvement in both the MC-GI approach and SNAPHU in cases where large patches of noise have isolated coherent wrapped pixels.

Thus, the following Chapters focus on work which attempted to rectify the two highlighted issues: noise and isolation of coherent pixel regions.

In the results section of Chapter 2, I demonstrate that MC[4]-GI achieves similar performance accuracy when unwrapping Sentinel-1 interferograms to SNAPHU in scenarios where there is limited phase noise and where there are limited occasions where high coherence regions have become isolated. MC-GI has been shown to yield better unwrapped results in areas of tight fringe patterns than SNAPHU for both synthetic and real interferograms.

Therefore, it is disappointing that the accuracy of MC-GI decreases in regions of high phase noise despite originally showing good results when tested upon synthetic interferograms with high levels of uniform Gaussian noise across the interferogram. In the following Chapters, I will present research investigating how to improve phase unwrapping for the cases where high noise has led to the type of model prediction failure described in Chapter 2. Whilst the model was able to filter Gaussian noise to identify underlying wrap count edges, when applied to real Sentinel-1 data, the presence of high levels of phase noise severely impacted both the model's performance and the subsequent Greedy Phase Integrator algorithm. The decrease in performance in this second step is likely due to its greedy nature, which relies on the assumption that high certainty, early returned wrap counts are definitely correct and that pixels with lower certainty are also those that are likely to introduce phase unwrapping errors. As the model had not been trained with realistic noise patterns (to be discussed in more depth in the following Chapter), the model prediction became increasingly poor with increasing levels of phase noise.

The proposed phase unwrapping method in the following three Chapters was undertaken with the motivation to improve the unwrapping of Sentinel-1 interferograms by training a deep learning model on more realistic circular Gaussian noise that better replicates the phase noise found in Sentinel-1 interferograms. A more robust noise integration method is then proposed, which utilises the cost file in SNAPHU to better integrate the gradient maps returned by the deep learning model. This work is presented across Chapters 3 and 4 for clearer communication of the research, where in Chapter 3, I discuss the model development, and in Chapter 4, I present the results of integrating the wrap count

gradient maps for both synthetic and real Sentinel-1 interferograms.

Chapter 3

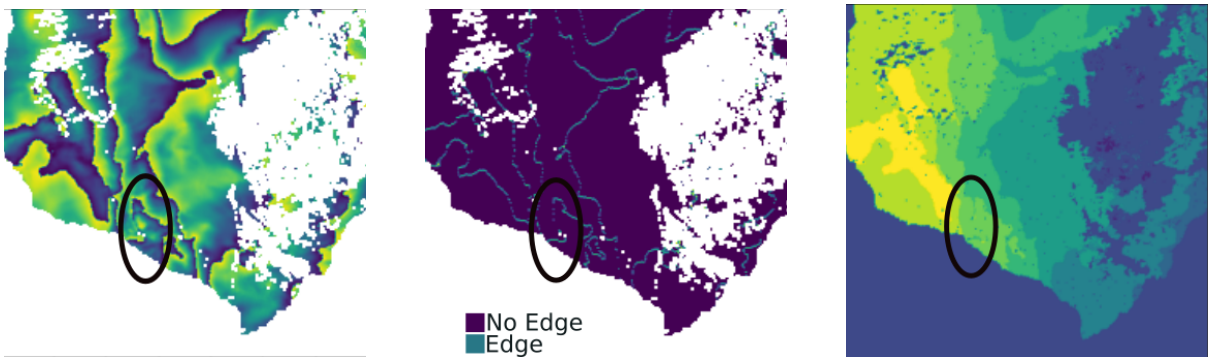
A Noisy Problem Part 1: Gradient Classification

3.1 Introduction

Interferometric Synthetic Aperture Radar (InSAR) is a well-established technique used to monitor natural hazards and anthropogenic activities by measuring surface deformation with millimetre per year scale precision (Morishita et al. 2020). Satellites send and receive microwaves, which have been reflected from Earth (Vijay Kumar et al. 2023), recording the returned signals as single look complex (SLC) images. Two SLC images taken over the same ground footprint can be used to make an interferogram, which is used to show the relative amount of displacement over time. Phase noise can be introduced from several sources into the interferograms. In the ionosphere, interactions with free electrons lead to variations in the travel path of the radar signal; this mainly affects L and P band SAR rather than C band (Bekaert et al. 2015). However, C band SAR is impacted in the lower 5km of the troposphere, where variations in pressure, temperature, and relative humidity can lead to signals in the interferogram of 15-20cm. Though spatially correlated, tropospheric signals are temporally uncorrelated (Zheng et al. 2021), and can

be corrected using weather models (Bekaert et al. 2015). In the unwrapping context, the tropospheric delay can be considered as a signal. Stochastic phase noise can be introduced from thermal noise and changes in the scattering properties of the reflector surface; for example, dense vegetation and water surface properties often change rapidly, impacting the strength and phase of reflected signals. These are more difficult to subtract from the wrapped phase and will likely negatively impact the downstream phase unwrapping (Zheng et al. 2021).

Given that phase noise is known to negatively impact phase unwrapping (Vijay Kumar et al. 2023), filtering approaches have been developed to reduce phase noise. Boxcar multi-look filtering takes an average phase from neighbouring pixels within a rectangular window (Pepin et al. 2024). Although this reduces phase noise, it can lead to a loss of resolution and phase-fringe edge distortion, meaning fringe edges such as that shown in Figure 3.1 may be lost. The MC[4]-GI model has been applied to filtered data taken from the LicSAR portal, where smoothing had led to a loss of the fringe edge. The lack of an edge class (C1 /$C\bar{1}$) results in the downstream Greedy Phase Integrator algorithm having insufficient information for integration, and it subsequently returns an incorrect wrap count.



(a) **(b)** **(c)**

Figure 3.1 – Impact of data filtering, which leads to smoothed fringe edges, increased ambiguity in the wrap count and affects the returned wrap count when the MC[4]-GI algorithm is applied. (a) Wrapped Filtered Interferogram, (b) Edge Prediction, (c) Returned Wrap Count

In order to retain as many true wrap count edge gradients as possible, it is favourable to perform enough filtering to remove the majority of phase noise, but not too much as to

remove the true wrap count gradient edges. The motivation of this work is to improve upon the MC-GI approach which I discuss in Chapter 1 to develop an unwrapping method which is more robust to phase noise and retains the addition of additional classification classes. Given that I highlight there was a discrepancy between the uniform Gaussian noise used to train MC-GI, and the phase noise present on Sentinel-1 data, I instead prepare a training data set where circular Gaussian noise is used to better simulate real phase noise. Moreover, rather than attempt to train a model which can both denoise the wrapped interferogram and correctly identify the underlying filtered gradients, the model is instead trained to classify all gradients at varying phase noise levels. While filtering can still be applied to the data, the goal of training a model in this manner is to minimise the level of filtering required when preparing the wrapped phase for unwrapping. Therefore, rather than filtering the wrapped interferograms prior to training, the decision was made to train the original data in order to develop a model that is robust to high levels of noise. This means that the noisier wrapped phase data is kept within the training dataset. This approach aim to enhance the model's ability to generalise effectively across a wide range of scenarios. This means that, even when filtering is necessary for real data, the model can operate effectively with reduced preprocessing, while still producing an accurate wrap count gradient map, with the hope this will reduce the number of occasions where smoothing through fringe edges has occured. This approach aims to preserve wrap count gradients corresponding to an absolute wrap count change of ± 1 or 2.

3.2 Methods

3.2.1 Noise Generation

Circular Gaussian noise, rather than uniform Gaussian noise, was added to the wrapped phase to ensure the wrapped phase has a realistic distribution. For cases where the phase value is close to $-\pi$, when phase noise is added, which reduces the phase value to less

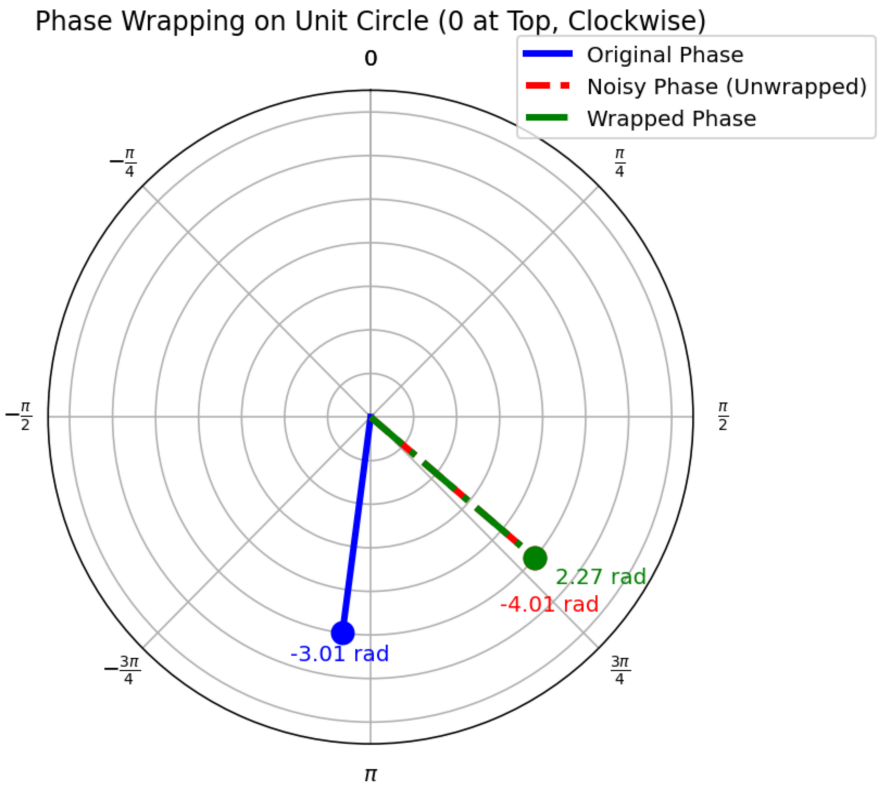


Figure 3.2 – Figure showing adding of noise in the complex domain. The wrapped phase value range is -π to π going clockwise around the unit circle. An example is given of the impact that adding phase noise at the $-\pi$ boundary can have.

than $-\pi$, the wrapped phase value will be wrapped to remain within the $-\pi$ to π range. For example, for a wrapped phase value of -3.01 radians, if noise were added which reduced the wrapped phase to -4.01 radians, the new wrapped phase value would be 2.27 radians, as illustrated in Figure 3.2.

This is also true for phase values close to π. A value of 3.01 radians which increased to 4.01 radians when phase noise is added would become -2.27 radians. This naturally occurs more often close to fringe edges where the phase values is close to the -π to π boundary, as highlighted in Figure 3.3, where Figure 3.3c demonstrates the biggest discrepancy in phase between clean (a) and the noisy (b) interferogram is at the fringe edges.

To add phase noise to the original wrapped phase signal of size 256×256, a signal to noise ratio is calculated based on some target coherence

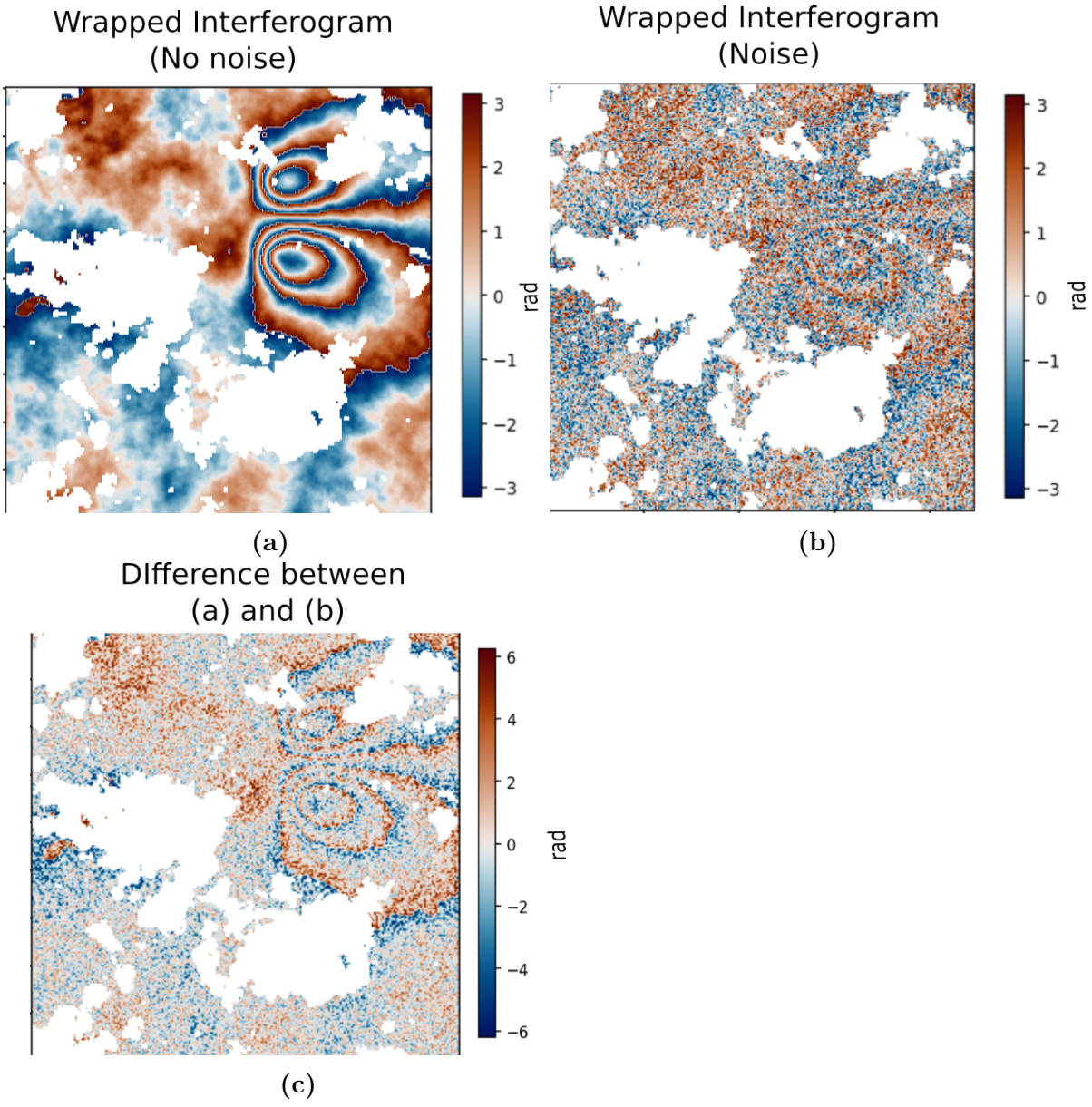


Figure 3.3 – Figure illustrating the impact of adding circular Gaussian noise to a wrapped interferogram. (a) shows the wrapped clean interferogram, (b) shows the wrapped noised interferogram with a target coherence of 0.3, (c) shows the difference between (a) and (b).

$$\text{SNR} = \frac{\text{Target Coherence}}{1 - \text{Target Coherence}}, \tag{3.1}$$

to determine the level of noise to add to the wrapped phase signal.

The standard deviation of the noise is calculated

$$\sigma^2 = \frac{1}{\sqrt{\text{SNR}}}, \tag{3.2}$$

which gives the level of the Gaussian noise to be added.

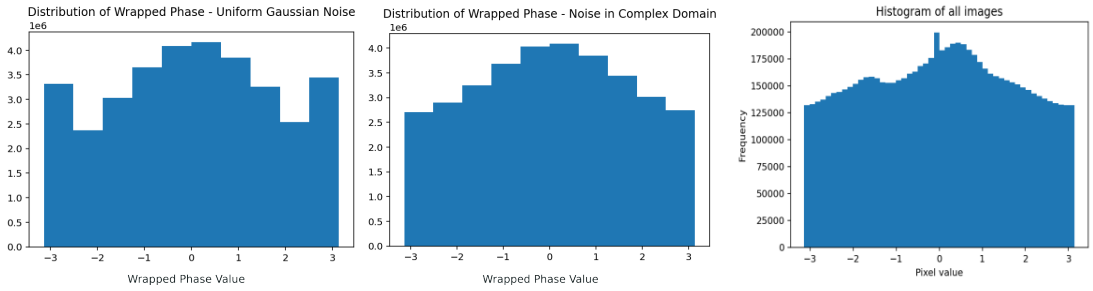
The phase noise to be added per pixel is prepared in an array of size $256 \times 256 \times 1$, where for each pixel, a random value is drawn from the Gaussian distribution with a mean of 0 and standard deviation of σ^2.

The phase noise is added to the original wrapped phase in the complex domain using

$$\psi_{noise} = \angle(e^{j\psi_{orig}} + e^{j\psi_{noise}}) \tag{3.3}$$

where $\angle(.)$ denotes the phase angle of the complex number, returning wrapped phase in the range [-π to π]. Adding the $e^{j\psi_{orig}}$ to $e^{j\psi_{noise}}$ guarantees the phase is added in the complex plane, ensuring circular Gaussian noise is used.

As can be seen in Figure 3.4, the data containing complex phase noise is more comparable in distribution to the expected distribution of real data. The spike seen in Figure 3.4 is due to one of the contributing interferograms containing a large area with no data, likely caused by masking of a water body during processing.



(a) **(b)** **(c)**

Figure 3.4 – Figure showing the difference in data distribution between (a) adding noise to wrapped phase directly and clipping noise to the $-\pi - \pi$ range as in Chapter 2 and (b) adding circular gaussian noise as in Chapter 3. (c) shows the data distribution of 4 real, unfiltered interferograms taken from the LicSAR portal.

3.2.2 Model Training

The U-Net architecture combined with a weighted categorical cross-entropy and Dice Loss led to excellent results for the training dataset in Chapter 1. The addition of circular Gaussian noise led to an increased difficulty training the U-Net model. Therefore, attention gates are added to the U-Net model architecture as they have been shown to be useful to improve the model results when included in U-Net which is trained to perform semantic segmentation tasks (Oktay et al. 2018). In the U-Net, downsampling results in a loss of spatial information. Whilst information is re-introduced through skip connections, it can mean information which is not useful is included from low-level encoder features and in turn can lead to poor model performance. To improve model performance, attention gates were added to take as input the upsampled decoder feature maps, and the encoder feature maps, and output an attention-weighted feature map. Attention gates support the identification of useful information regions, suppression of less useful information, and act as denoisers, leading to a generally improved feature extraction (Oktay et al. 2018). The spatial attention gate mechanisms were used to support the learning of finer scale features for semantic segmentation.

The model only classifies pixel wrap count gradients in either the horizontal or vertical direction, though this does introduce additional computational costs due to the requiring two models to be trained. The model was trained to convergence in all cases; early

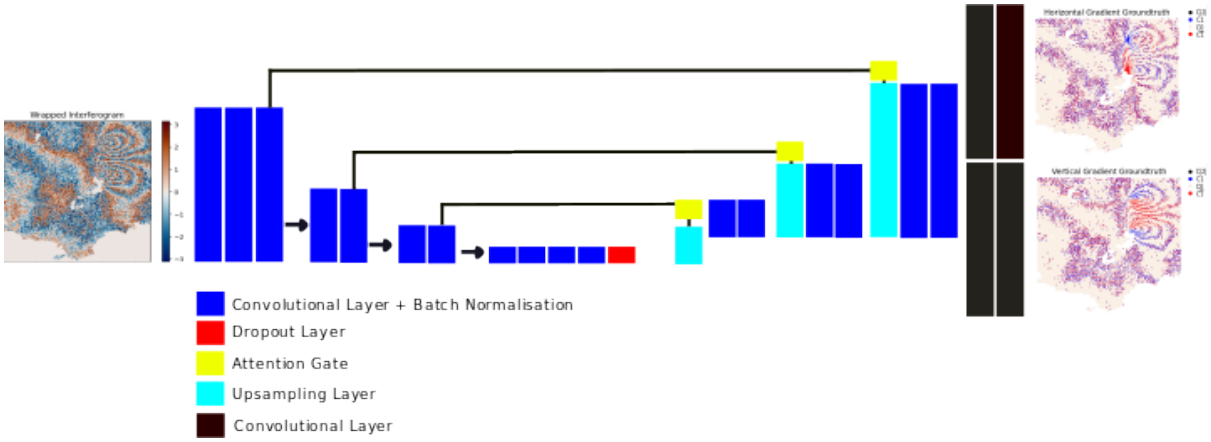


Figure 3.5 – Diagram of the model architecture used to train the deep learning model to identify horizontal and vertical wrap count gradients from noisy wrapped interferograms. Model architecture consists of convolutional layers, batch normalisation and max pooling layers in the encoder and convolutional layers, batch normalisation and concatenation layers in the decoder, combined with skip connections and the attention gate mechanisms. The wrapped phase input and corresponding vertical and horizontal wrap count gradient maps are shown, where there is a 4 class target output (C0 , C1 , $C\bar{1}$ and $C2^*$). For illustration purposes, the input and output are represented as if they are images, however model input and output each are greyscale, i.e one channel.

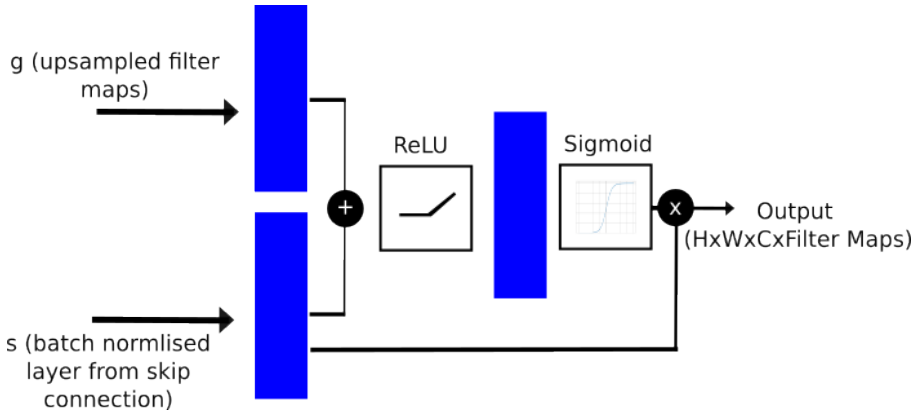


Figure 3.6 – Figure showing the attention-gate used in the U-Net

stopping was used to prevent overfitting.

3.2.3 Loss Functions

Tversky Loss

The Tversky loss is useful for training semantic segmentation models on imbalanced data sets. Whereas the Dice Loss treats false positives and false negatives equally (Abraham et al. 2018), the Tversky loss allows the adjustment of weightings for the false positives and false negatives using the weighting factors α and β respectively, finding by doing so, it leads to high sensitivity, improving performance for highly imbalanced data sets(Salehi et al. 2017). When classifying wrap count gradients, the training data set is heavily imbalanced, with large areas containing examples of a wrap count gradient of 0 or ± 1 in comparison with the much smaller areas containing absolute wrap count gradients which exceeds 1.

$$\text{Tversky Loss} = 1 - \frac{TP}{TP + \alpha FP + \beta FN + 1e^{-6}} \tag{3.4}$$

where TP is True positive, alpha and beta are variables to be set, FN is false negative, FP is false positive and $1e^{-6}$ was added to ensure no division by 0 (Azad et al. 2023). When α and β are both set to 0.5 and γ is 1, this is equivalent to using a Dice Loss. Values of α and β set to 0.3 and 0.7 respectively were found to work well by Salehi et al. 2017. When training the model, I applied logarithmic class weightings to the Tversky loss.

Weighted Sparse Categorical Cross-Entropy As previously stated in Chapter 2 (Equation 2.2), the weighted sparse categorical cross-entropy loss is used for multi-class semantic segmentation tasks with imbalanced classes (Spoorthi et al. 2020). Applying class weighing ensures that the misclassification of the minority class is more heavy penalised compared to majority class misclassification, forcing the model to not just classify

all pixels as the majority class for the lowest loss value. In Chapter 2, the weightings were calculated using

$$w_i = \text{total pixels}/(\text{C} * count_i), \tag{3.5}$$

where w_i is the computed weight for class i and $count_i$ is the number of samples in class i, and C is the total number of classes. Each weighting was inversely proportional to the sample size meaning a smaller sample size had a greater weighting. Though re-applying such weightings worked well for interferograms with a coherence above 0.5, for the interferograms of lower coherence, the model placed too heavy an emphasis on classifying the minority classes and tended to over-classify pixels as the $C|2|$ class.

Instead, applying a logarithmic scale of weighting, calculated as

$$W_c = log(1 + \frac{1}{\frac{n_c}{N} + \epsilon}), \tag{3.6}$$

where n_c is number of samples of class c, N is the total number of pixel samples and ϵ is $1e^{-6}$ to avoid division by 0, led to an overall improved classification result on the test data set. There is a reduced weighting of the minority class $C|2|$ with respect to C0 which improved the overall model classification performance.

3.2.4 Datasets

The training dataset was prepared using the software SyInterferopy (Gaddes et al. 2019) to generate the wrapped interferogram and unwrapped interferogram pairs. Phase noise was added using the approach discussed in the noise generation subsection, using the target coherence range of 0.3 to 1.0. The target output for each noised wrapped interferogram was the noised wrap count gradient in the horizontal (and vertical) direction, i.e: the gradients required to move from the noised wrapped phase to a clean unwrapped

phase output. Due to noise being added in the complex domain, this led to an increased number of C1 , $C\bar{1}$ and $C|2|$ around the fringe edges. The wrap count gradients in the horizontal direction were calculated as

$$\nabla_h(k_{i,j}) = k_{i+1,j} - k_{i,j}, \tag{3.7}$$

and the vertical wrap count gradient is given by

$$\nabla_v(k_{i,j}) = k_{i,j+1} - k_{i,j}, \tag{3.8}$$

where

$$k_{i,j} = round(\frac{(\phi_{i,j} - \psi_{i,j})}{2\pi}). \tag{3.9}$$

The NaN values which are automatically generated for the wrapped interferograms to mask large water body regions and simulate large noise patches in the SyInterferopy software (Gaddes et al. 2019), are replaced with the value 0 in the wrapped interferograms, meaning the wrap count gradient is also 0.

The test and training data sets were prepared the same way.

3.3 Results and Discussion

3.3.1 MC[4]-CGI Gradient Classification Performance

To explore the impact of varying the parameters α and β upon the accuracy of the multi-class wrap count gradient identification, henceforth referred to as MC-CGI, I performed an augmentation study. Given that in Chapter 2 the MC[4]-GI performed best at the

Model	Loss Parameters	Macro-MIoU	Macro-Precision	Macro-Recall	Macro-F1
1	α:0.5, β:0.5 (equivalent to Dice Loss)	0.76	**0.78**	**0.93**	**0.83**
2	α: 0.2, β: 0.8	0.76	0.76	0.81	0.79
3	α: 0.1, β: 0.9	0.76	0.74	0.86	0.77
4	α: 0.3, β: 0.7 (equvilent to Salehi et al. 2017)	0.72	0.74	0.90	0.79

Table 3.1 – Table showing the results of testing the MC[4]-CGI models on an unseen test data set of size $256 \times 256 \times 320$, where the coherence varies from 0.3 to 1.0. In model training, the Weighted Categorical Cross Entropy and Class Weighted Tversky Loss are used for applied. Model 4 uses the Tversky Loss and weighted categorical cross entropy. The combination of α, β and γ are varied. Weightings used for MC[4]-CGI: $C\bar{1}$: 3.3, C0 : 0.7, C1 : 3.3, $C|2|$: 8.4 for training. Scores are Macro-MIoU, Macro-Precision, Macro-Recall and Macro-F1. The score is calculated by averaging between the horizontal and vertical training model scores.

Model	Loss Parameters	Macro-MIoU	Macro-Precision	Macro-Recall	Macro-F1
1	α:0.5, β:0.5	**0.91**	**0.94**	0.96	**0.95**
2	α: 0.2, β: 0.8	0.90	0.92	0.97	0.94
3	α: 0.1, β: 0.9	0.88	0.90	**0.98**	0.94

Table 3.2 – Table showing the results of testing the MC[3]-CGI models on an unseen test data set of size $256 \times 256 \times 320$, where the coherence varies from 0.3 to 1.0. In model training, the Weighted Categorical Cross Entropy and Class Weighted Tversky Loss are applied. The combination of α, β and γ are varied. Weightings used for MC[3]-GI: $C\bar{1}^*$: 3.2, C0 : 0.7, $C1^*$ 3.2. Scores are Macro-MIoU, Macro-Precision, Macro-Recall and Macro-F1. The score is calculated by averaging between the horizontal and vertical training model scores.

Model	Loss Parameters	Macro-MIoU	Macro-Precision	Macro-Recall	Macro-F1
1	α:0 .5, β: 0.5	**0.57**	**0.60**	0.77	0.63
2	α: 0.2, β: 0.8	0.56	0.59	0.78	0.63
3	α: 0.1, β: 0.9	0.54	0.56	0.78	0.60

Table 3.3 – Table showing the results of testing the MC[3]-CGI models on an unseen test data set of size $256 \times 256 \times 320$, where the coherence varies from 0.3 to 1.0. In model training, the Weighted Categorical Cross Entropy and Class Weighted Tversky Loss are used for Models 1-3. The combination of α, β and γ are varied. Weightings used for MC[5]-GI: $C\bar{2}^*$: 8.8, $C\bar{1}$: 3.3, C0 : 0.7, C1 : 3.3, $C2^*$: 9.4. Scores are Macro-MIoU, Macro-Precision, Macro-Recall and Macro-F1. The score is calculated by averaging between the horizontal and vertical training model scores.

complete phase unwrapping task and including an additional class is the next most logical step from the 3 class models represented in the literature, it was chosen as a baseline for the investigation. Table 3.1 displays the parameters used to train the MC[4]-CGI model, the macro-MIoU, macro-precision, macro-recall and macro-f1 scores (please refer back to Chapter 2 for a detailed description of each statistic) when each model was tested on the test data set.

The results presented in Table 3.1 demonstrate that the parameter configuration used to train Model 1 achieved the best performance. Model performance progressively declined as the Tversky loss parameter setup deviated from the Dice Loss equivalent ($\alpha = 0.5, \beta = 0.5$). The parameters used to train the MC[4]-CGI models 1 -3 were used to train MC[3]-CGI and MC[5]-CGI, as generally these parameter settings led to the best overall metric score for the MC[4]-CGI model set.

Whilst the model performance metrics for the MC[3]-CGI and MC[5]-CGI are shown in Table 3.2 and Table 3.3 respectively, this is for completeness, not comparison purposes. This is because the best performing MC[3]-CGI and MC[5]-CGI are used for comparison purposes in work presented in the following Chapter, to understand the impact of varying class number on the accuracy of the complete phase unwrapping process. Comparison of the semantic segmentation accuracy between the MC[3]-CGI, MC[4]-CGI and MC[5]-CGI modes is inappropriate as the task is different every time the number of classes changes.

The *NaN* regions, automatically generated through masking in the SyInterferopy data generation step, were replaced with 0 values in the wrapped interferograms, and the gradients expected to classify the pixels as C0 ; the predicted gradients within these regions were included in the statistics scores. Each statistical measurement was calculated on a per class basis and the mean taken meaning each class contributed equally to the final score. Consequently, despite the smaller number of minority class instances, a low $C|2|$ value will disproportionately lower the final statistical score relative to the sample size in the test data set, indicating that the model does not perform equally well for each class.

Figure 3.7 presents the impact of changing the coherence on the macro-MIoU. All classification models show generally good classification ability, with a macro-MIoU score above 0.75, where coherence is equal to or above 0.6 coherence in both the vertical and horizontal direction. When the coherence is below 0.6, the macro-MIoU score consistently declines across all 4 models, indicating the level of noise has become significant enough to negatively impact the model's performance. The decline in performance may be due

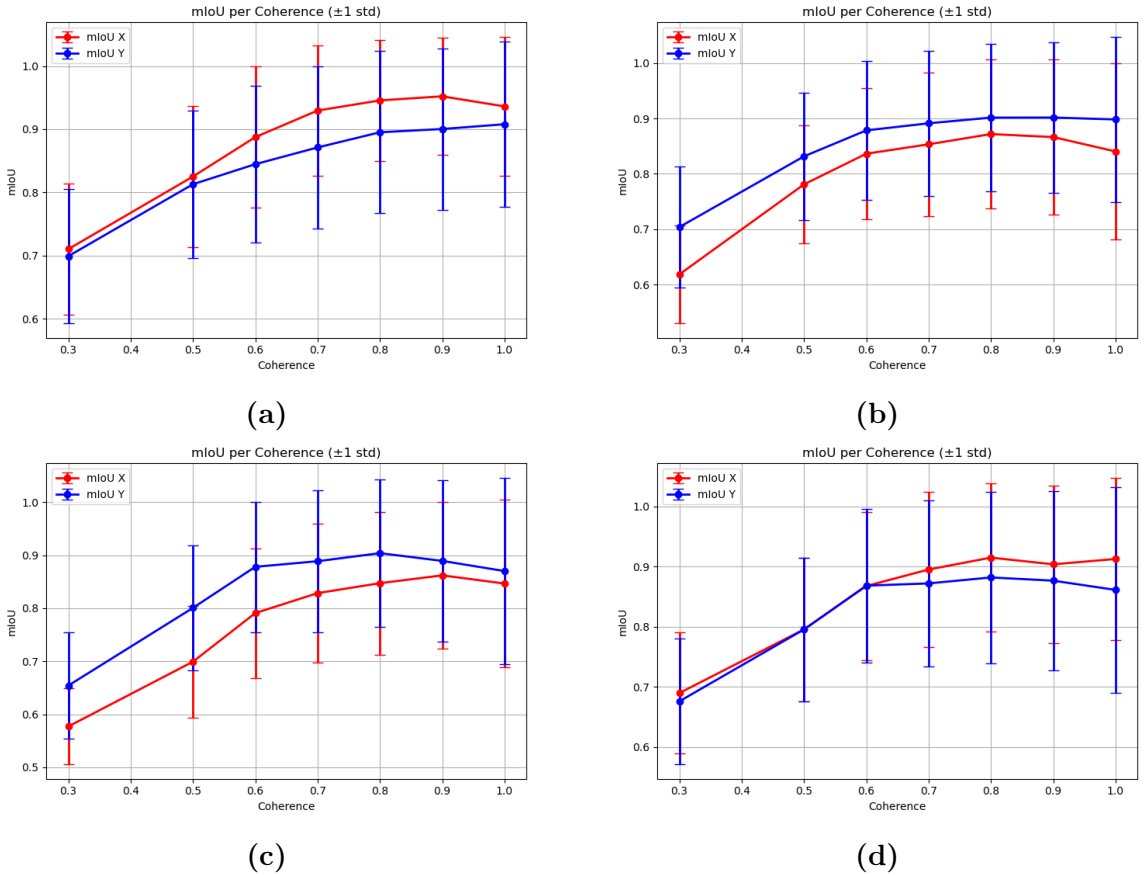


Figure 3.7 – Figure showing the Macro-MIoU scores for augmented models in the horizontal and vertical direction plotted against coherence for the MC[4]-CGI models from Table 3.1. (a) Model 1, (b) model 2, (c) model 3 and (d) model 4.

to increased noise, which has been randomly added to the interferogram. As the model is tasked with identifying both gradient edges associated with wrap count change and the gradients of the noise, the random distribution of the noise reduces the effectiveness of being able to use local pixel information to aid classification. Whilst neighbouring pixels to true wrap count gradients may still provide useful context, the randomness of noise introduces uncertainty, making the task harder.

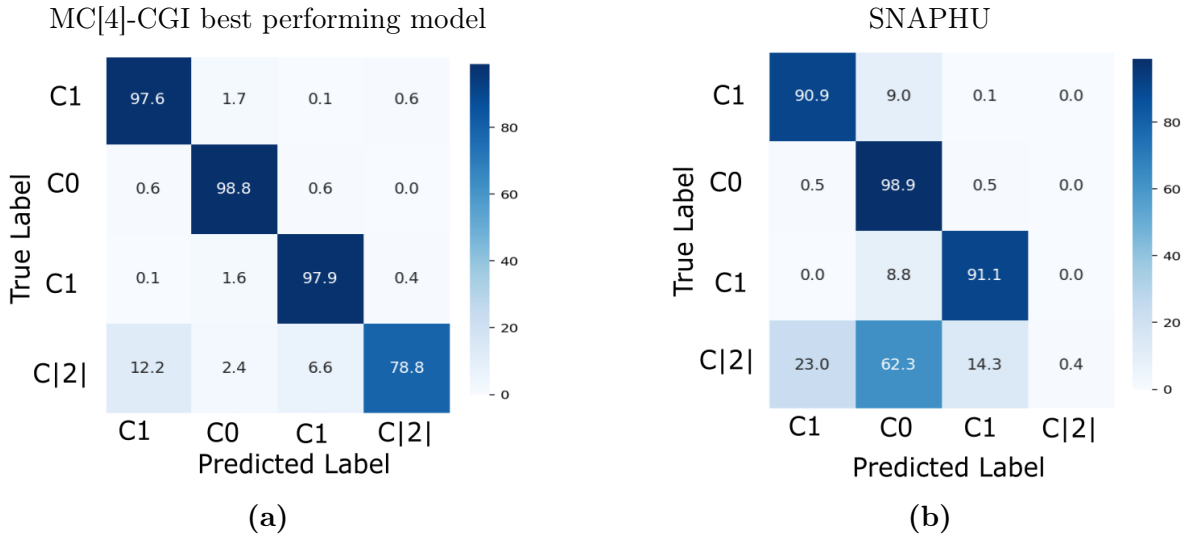


Figure 3.8 – Confusion matrices of (a) the best performing 4 class model and (b) SNAPHU as treated as a 4 class wrap count gradient prediction scenario. Correct prediction occurrence is given as a percentage of the total number of true positives for each class.

The confusion matrices in Figure 3.8 show the average percentage of correctly classified pixels in the horizontal and vertical direction for the best performing 4 class model and SNAPHU; SNAPHU is treated as a 4 class classification scenario.

Figure 3.8a indicates that on over 97% of occasions, the correct gradient classification was made by MC[4]-CGI for the majority classes. $C|2|$ predictions were correctly predicted on 79% of occasions by the MC[4]-CGI, which given the low occurrence of $C|2|$ was very encouraging. Where misclassification of $C|2|$ into an alternate class did occur, this was normally into C1 or $C\bar{1}$, which leaves an interesting conundrum. If the model does correctly identify the wrap count gradient direction but not the magnitude, when the wrap count gradients are integrated, the situation is equivalent to a 3 class model output. This means the outcome will be no worse than what is normally achieved through phase

unwrapping. Whilst SNAPHU identified 0.4% of jumps as $C|2|$ correctly, it identified over 90% of majority classes correctly, which whilst high is still a lower overall classification accuracy than the MC[4]-CGI model.

Overall MC[4]-CGI performs good classification, consistently performs well in regions where the deformation fringes are less dense, though does have a tendency to over-predict the presence of non C0 where fringes become more dense as shown in Figure 3.9.

3.4 Conclusion

To conclude, in this work I have developed 3 models which show good performance at correctly classifying wrap count gradients. The models are trained using synthetic data which contained circular Gaussian noise which better simulates phase noise than uniform gaussian noise with re-clipping to the $-\pi$ to π range. The benefit of training on noisy data is that it reduces the level of filtering required prior to performing phase unwrapping, reducing the potential for smoothing to lead to the loss of fringe edges. In Chapter 4, the MC-CGI wrap count gradient maps will be integrated using SNAPHU, where the cost file is updated a priori to initialising SNAPHU to include the model output information.

The work presented in Chapter 3 has built upon Chapter 2 by improving the model's robustness to more realistic noise. Initially, I attempted to train a model with the same architecture as the MC-GI model presented in Chapter 2; the trained model was unable to correctly identify wrap count gradients in the presence of circular gaussian noise, as the model architecture was too simple, and therefore insufficient, to correctly identify wrap count gradients. By including attention gates in the model architecture, I successfully trained the models to correctly identify vertical and horizontal wrap count gradients, even in the presence of circular gaussian noise. Therefore, there is an overall improvement from the models' trained in Chapter 2 from the situation where the models' increasingly struggle to correctly identify the wrap count gradients with increasing circular gaussian noise to being able to correctly identify wrap count gradients' a high proportion of the

time when attention gates are included in the model architecture.

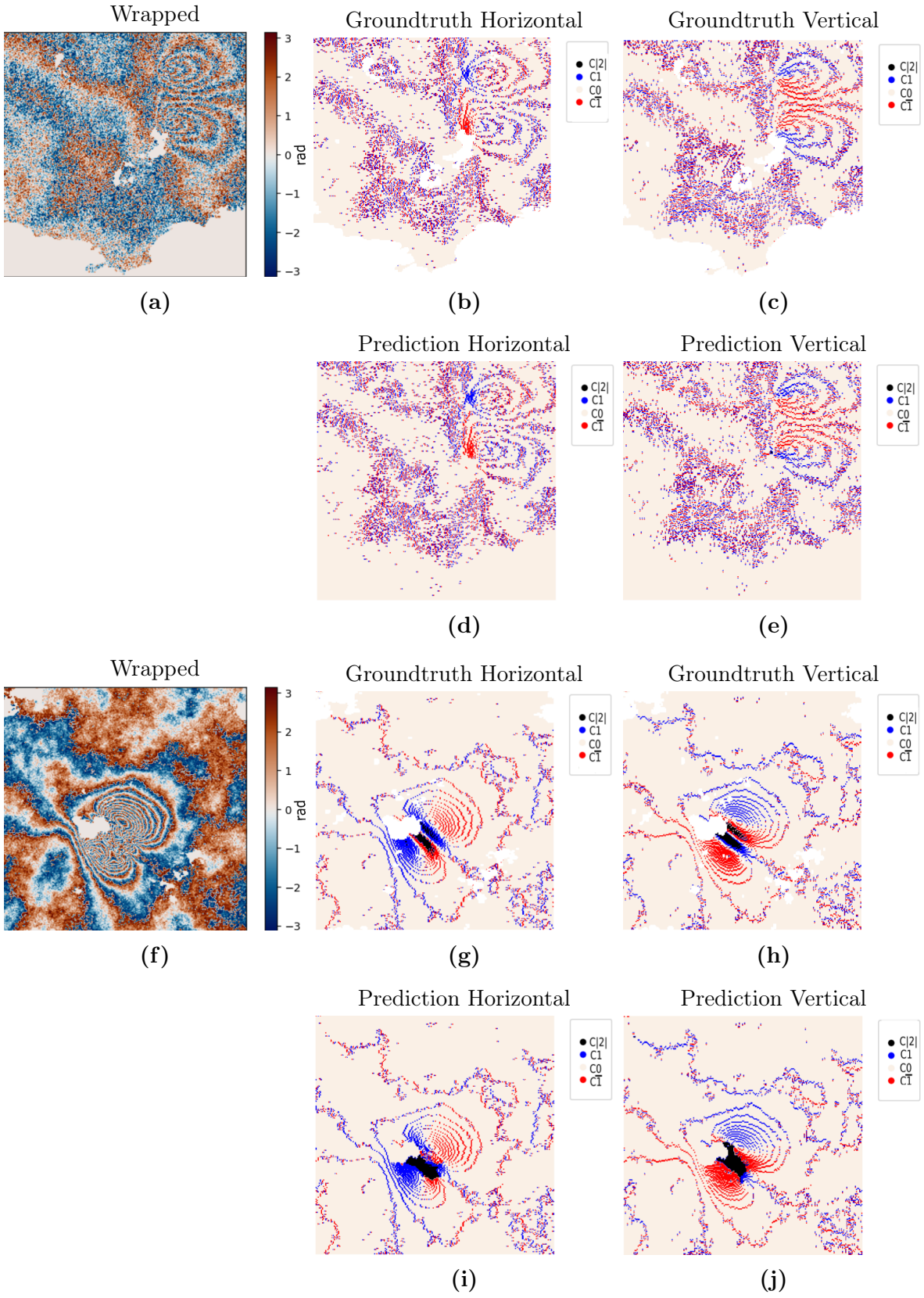


Figure 3.9 – Wrap Count gradient prediction made by the best performing MC[4]-CGI for two levels of noise (a) High noise level (coherence 0.5), (f) Low noise level (coherence 0.8). (a) Wrapped phase (high noise), (b) groundtruth classification in the horizontal direction (a), (c) groundtruth classification in the vertical direction (a), (d) predicted classification in the horizontal direction (a), (e) predicted classification in the vertical direction (a). (f) Wrapped phase (low noise), (g) groundtruth classification in the horizontal direction (f), (h) groundtruth classification in the vertical direction (f), (i) predicted classification in the horizontal direction (f), (j) predicted classification in the vertical direction (f).

Chapter 4

A Noisy Problem Part 2: The Integration Step

This Chapter is a continuation of Chapter 3; Chapter 3 presents the development of the model used to classify wrap count gradients. In this Chapter, I evaluate the entire unwrapping approach, which is referred to as Multi-Class Cost-Guided Integration (MC-CGI). I use the best performing MC[3]-CGI, MC[4]-CGI and MC[5]-CGI models in the unwrapping process. The information was separated into two chapters' in order to more clearly convey the information.

In this Chapter, I use the best performing 4 class model (MC[4]-CGI) trained in Chapter 3, and test whether the output, when combined with SNAPHU, leads to an overall improved unwrapped phase result in comparison to SNAPHU. The wrap count gradient maps output by the best performing MC[4]-CGI are integrated using SNAPHU, specifically using the cost-based integration approach. This is to provide a more robust method of integrating the model wrap count gradient maps than the Greedy Phase Integrator discussed in Chapter 2, whilst retaining the innovation of including additional wrap count classes exceeding the normal $0, \pm 1$ classes.

4.1 Introduction

The Greedy Phase Integrator proposed in Chapter 2 performs well when applied to real Sentinel-1 data if the phase noise is low enough to not be mistaken as a large noise patch and fringe edges remain uncovered. MC-GI shows particular improvement in the unwrapped phase result when compared to SNAPHU when unwrapping regions with high fringe rates. However, it is reliant on a differing range of prediction certainties across the interferogram, and the assumption is made that generally those pixels gradients of low certainty will be those likely to cause unwrapping errors. Issues arise when the predicted gradient is inaccurate, which generally increases in occurrences as the noise level also increases.

In classical phase unwrapping techniques, the integration path is influenced by a penalty that discourages large differences between the wrapped and unwrapped phase gradients. Additional information, such as coherence, can be used to improve the result, by adjusting the strength of the penalty; in regions of low coherence, the penalisation is lower because there is more uncertainty in the wrapped phase gradient. Whilst the GI does use coherence to prepare a noise mask, which determines which gradients should be excluded from the integration process, this is limited because the coherence threshold below which pixels should be masked needs to remain high enough to ensure that there are still pixels left to unwrap. Furthermore, the GI method is reliant upon few incorrectly classified gradients and is therefore intrinsically sensitive to noise. Though by working from the highest certainty to the lowest certainty gradients, some of the negative impacts of speckle gradients will likely be removed, if there is a large number of incorrectly classified gradients, an unrealistic patchwork effect can occur.

Thus, the GI approach is unable to completely negate the already recognised problem of high phase noise leading to incorrect gradient estimation and therefore phase unwrapping errors. As such, it is essential to explore alternate, more robust, integration approaches of the MC-CGI wrap count gradient maps which both increases the accuracy of unwrapped

1805 results for high phase noise regions where incorrect classification of phase gradients are
1806 more likely to occur but retains the benefit of identifying wrap count gradients and
1807 differentiating the identification of large gradient jumps. As a result, I instead combine
1808 the phase gradient estimates with SNAPHU to return the unwrapped phase.

1809 4.1.1 SNAPHU

1810 SNAPHU (Statistical-cost network flow algorithm for phase unwrapping) is a popular
1811 phase unwrapping technique. As has been stated in the introduction, the objective of
1812 many classical phase unwrapping approaches is to minimise the difference between the
1813 wrapped and unwrapped phase gradients to some penalisation power (Equation 1.4).
1814 Chapter 1.3.1 provides a description of SNAPHU, and for a full explanation of SNAPHU,
1815 please refer to Chen et al. 2001. SNAPHU is a popular phase unwrapping technique,
1816 however the accuracy of the unwrapped phase result is reliant on accurate pixel quality
1817 metrics, and accurate unwrapped phase gradient estimation. In areas of high fringe rate,
1818 SNAPHUs performance can be decline; though SNAPHU can allow for higher phase
1819 gradients, often the quality index of the high fringe rate regions will be low, meaning these
1820 areas will have a high degree of uncertainty associated with the estimated unwrapped
1821 phase gradient, and are more liable to phase unwrapping errors. High noise levels can
1822 also lead to a decline in the accuracy of unwrapped phase gradient.

1823 Deep learning can be used to enhance the accuracy of unwrapped phase gradient esti-
1824 mations. These improved phase gradient estimations are integrated with SNAPHU to
1825 improve the accuracy of the unwrapped phase result relative to using SNAPHU alone.
1826 As outlined in Chapter 2, I include an additional class for the explicit identification of
1827 large wrap count gradients (those which exceed an absolute gradient of ± 1).

1828 Given that the MC[4]-GI technique had poor robustness to noise, and that SNAPHU has
1829 poorer performance when wrap count gradients are larger, the combining of SNAPHU
1830 with the phase gradient estimations was explored with the expectation that such an

approach would lead to an overall improved performance. Testing was also performed to identify if combine the newly trained models with the greedy integration also had a better performance than using SNAPHU alone.

The wrap count gradient map outputs, together with their associated estimation certainties, from the best performing 4 class model (see Chapter 3) are used to inform the cost function used in SNAPHU's integration step. This strategy is designed to improve the accuracy of SNAPHU's unwrapped result in scenarios where fringe rates are high.

4.2 Methods

4.2.1 Unwrapping Process

In this Chapter, I combine the wrap count maps with SNAPHU, rather than with the Greedy Phase Integration algorithm presented in Chapter 2. This is due to the output of the Chapter 3 model differing from the output of the Chapter 2 model. In Chapter 2, the model is able to differentiate between uniform gaussian noise and fringe edges, returning a gradient map which has minimal additional wrap count gradients, other than those associated with fringe edges.

The model in Chapter 3 was trained on wrapped phase where circular gaussian noise has been added. Due to the model being unable to denoise the wrapped phase, the model instead classifies every wrap count gradient; it does not differentiate between gradients associated with fringe edges, and those associated with noise. As a result, the output wrap count gradient maps looks noisier. The Greedy Phase Integration algorithm relies on a gradient maps with minimal incorrectly classified gradients, and its accuracy declines as the number of incorrect gradients increase, leaving, in the worst case scenario, a wrap count map which is highly inaccurate. The wrap count gradients are instead integrated using SNAPHU. The model output was not originally combined with SNAPHU in Chap-

ter 2 as it is complex process to implement the wrap count gradient maps using cost files, though the increased robustness to noise is beneficial, as is the improved accuracy of the unwrapped result relative to both MC-GI and SNAPHU alone.

4.2.2 Cost file preparation using wrap count gradient estimations

SNAPHU optimisation routines are applied to search for the minimum total cost solution for each interferogram (Hooper 2010). Cost files are provided as a prior constraints, to more quickly guide the algorithm to the most accurate unwrapped result. To prepare the cost file on an interferogram by interferogram basis, I use the wrap count gradient maps returned by the MC-CGI model and the certainty in each prediction to determine the offset and variance.

Offset Calculation

The offset is calculated using the weighted μ,

$$\mu_{weighted} = \sum_{i=0}^{N} w_i \Delta k_i \tag{4.1}$$

where N is the number of classes and w_i is the weight of class i, k is the integer wrap count gradient of class i.

During the integration process, C0 , C1 and $C\bar{1}$ represent an integer wrap count offset of 0, +1 and -1 respectively. For MC[4]-CGI, $C|2|$ represents all occasions with an absolute wrap count gradient exceeding +1. I compare the complete unwrapping performance of MC[4]-CGI with a MC[5]-CGI and MC[3]-CGI process, to understand the impact of including additional classes on the accuracy of the unwrapped result.

It was expected that by directly using the model majority classes as the integer wrap count offset from $\Delta\psi$, it should result in the correct $\Delta\phi$. However, this did not work.

Consider the wrapped interferogram shown in Figure 4.1a, where the phase changes only in the horizontal direction. The expected unwrapped phase shown in Figure 4.1b shows a +6 wrap count change. Implementing a cost file where the offset is set to +6, the correct unwrapped phase is not returned, as shown in Figure 4.1c, and there are two phase unwrapping errors. The unwrapped phase gradient change associated with the +6 is -2.52 rather than the expected 13.2π.

Instead, $\Delta\psi$ must also be included in the offset value, where the offset value is calculated as

$$\text{offset} = (\mu_{weighted} - round(\frac{(\Delta\bar{\psi_{pc}} - \Delta\psi_{pc})}{2\pi})), \tag{4.2}$$

where $\Delta\psi_{pc}$ is the original wrapped phase gradient phase cycle and $\Delta\bar{\psi_{pc}}$ is the re-wrapped around phase gradient phase cycle given according to

$$\Delta\bar{\psi_{pc}} = \begin{cases} \Delta\psi_{\text{pc}} - 1, & \text{if } \Delta\psi_{\text{pc}} > 0.5 \\ \Delta\psi_{\text{pc}} + 1, & \text{if } \Delta\psi_{\text{pc}} < -0.5 \\ \Delta\psi_{\text{pc}}, & \text{otherwise.} \end{cases} \tag{4.3}$$

By using Equation 4.2, the phase is correctly unwrapped as shown in Figure 4.1f where there is no difference between the groundtruth wrap count and the SNAPHU returned wrap count associated with Figure 4.1a.

To demonstrate that using Equation 4.2 to prepare the offset value in the cost file does ensure an improved phase unwrapped solution for interferograms and not just a simple dummy examples shown Figure 4.1, Figure 4.2a was unwrapped. It is obvious that the unwrapped solution achieved by using SNAPHU combined with the cost file where the MC[4]-GI gradient maps have been used in Equation 4.2 (Figure 4.2d) is closer to the unwrapped groundtruth than the unwrapped solution achieved using SNAPHU alone

(Figure 4.2c).

Variance Calculation

The variance Var is calculated as

$$Var = \sum_{i=0}^{N} (w_i \cdot i^2) - \mu^2 \tag{4.4}$$

where N is the number of classes, i is the class, w_i is the probability of class i and μ is the weighted mean.

Laycost and DzMaz Calculation

The remaining values given in the cost file, Laycost and DzMax, were calculated as in the internal SNAPHU cost file (see (Chen et al. 2001)).

MC-CGI approach

Masked pixels are determined by a coherence map; below a user defined coherence threshold, the pixels are marked as masked and are normally given pre-determined values by SNAPHU in the cost file (offset: 0, variance = 32000, DzMax: 32000, Laycost: -32000). The majority of occasions in the remainder of this Chapter, I use the coherence threshold of 0.1.

In the MC[4]-CGI scenario $C|2|$ pixels are treated as masked pixels, as though we know the offset is larger that $\pm 1, 0$, the magnitude or direction of the wrap count gradient is unknown.

In the scenario where a MC[5]-CGI process is applied, $C2^*$ and $C\bar{2}^*$ represent all pixels where an absolute wrap count gradient change is equal to or exceeds 2 in the positive

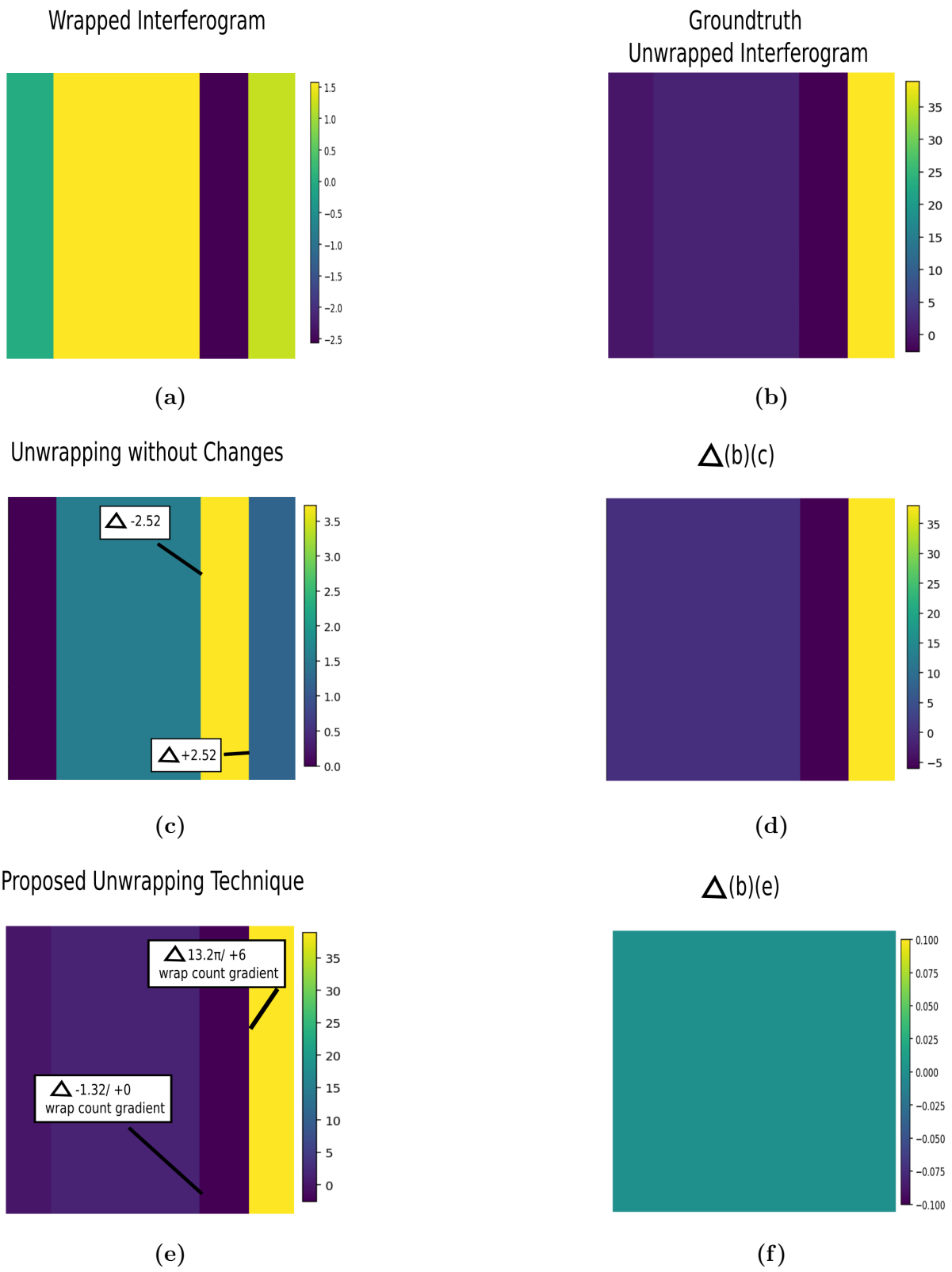


Figure 4.1 – Figure demonstrating the impact of applying predictions using Equation 4.2. (a) Wrapped interferogram, (b) unwrapped interferogram groundtruth, (c) Unwrapped interferogram where the offset have been directly set to the groundtruth wrap count gradient, (d) difference between groundtruth unwrapped phase (b) and unwrapped phase (c), (e) unwrapped phase applying Equation 4.2, (f) difference between groundtruth unwrapped phase (b) and unwrapped phase (e).

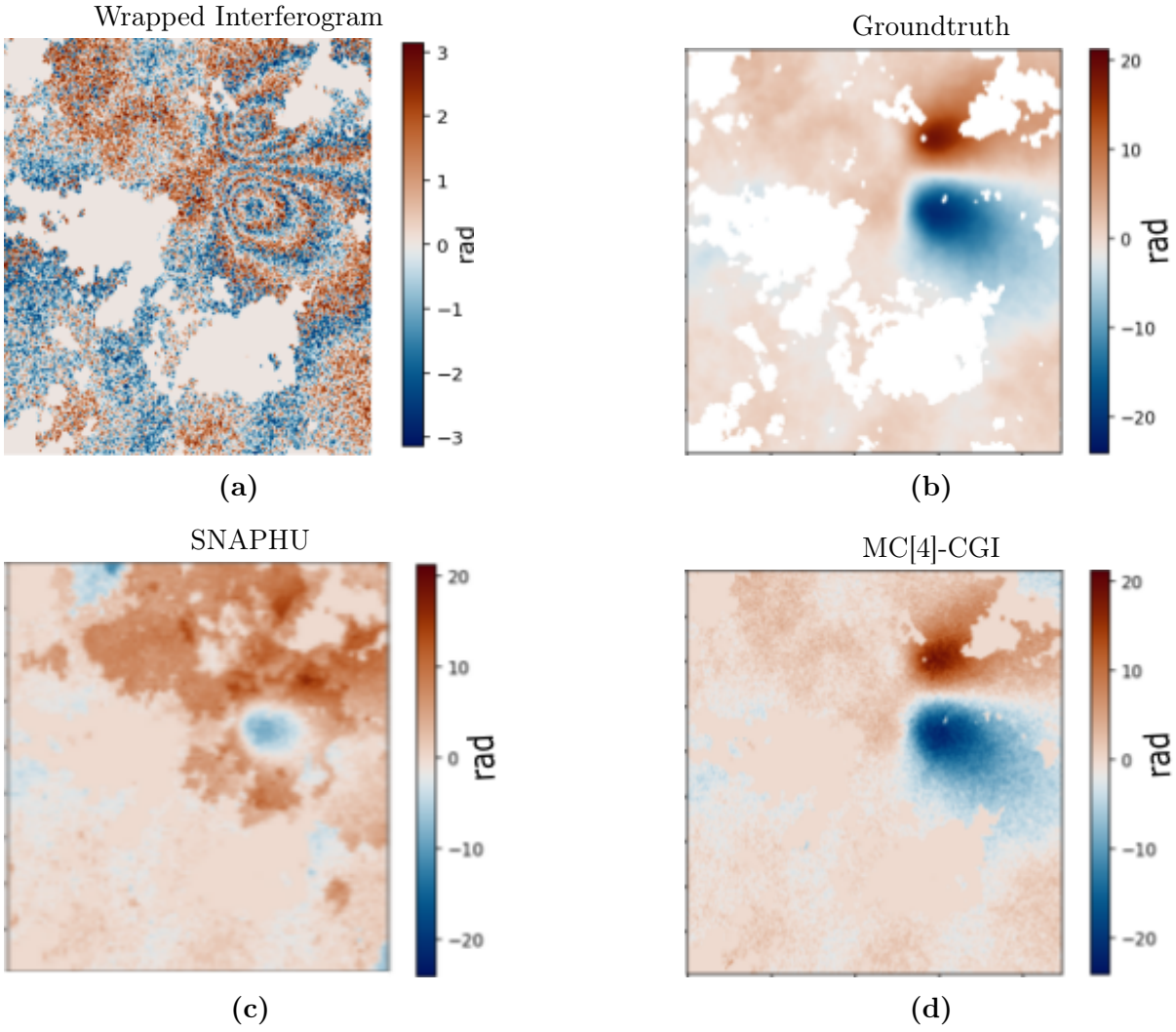


Figure 4.2 – (a) Wrapped Interferogram (b) Groundtruth Unwrapped Interferogram (c) SNAPHU unwrapped interferogram without cost file included (d) SNAPHU unwrapped interferogram when cost file is used to guide the phase unwrapping.

and negative direction respectively. For these minority classes, whilst it is known the offset is not $\pm 1, 0$, there is uncertainty in the magnitude of the shift. Such locations are considered to be probable discontinuity locations, likely to introduce phase unwrapping errors and therefore, they are treated as masked pixels during the preparation of the cost file.

The cost file is given as an optional input when initialising SNAPHU, and as previously stated, used as a priori information to increase the likelihood that SNAPHU will achieve the most accurate unwrapped solution. During experimentation, SNAPHU was initialised for all cases using a Minimum Spanning Tree algorithm.

4.3 Results and Discussion

4.3.1 Testing Unwrapping using Synthetic Interferograms

Testing the MC-CGI approach on the Synthetic Test Data Set

The best performing models for each classification task (MC[3]-CGI, MC[4]-CGI and MC[5]-CGI) from Chapter 3 are used to evaluate the complete unwrapped process. This process involves the models performing wrap count gradient classification, and the wrap count gradient maps then being integrated by SNAPHU. The gradient maps and model prediction certainty are used to determine the values in the cost files which are given as a priori information to SNAPHU prior to unwrapping. The MC-CGI method was tested on a synthetic test data set comprised of 320 wrapped interferograms, where 40 "clean" interferograms (i.e no noise added), had circular Gaussian noise added where the target coherence was increased in segments of 0.1 from 0.3 to 1.0.

To re-iterate, for the MC-CGI, the $C|2|$ gradients and for the MC[5]-CGI, the $C2^*$ and $C\bar{2}^*$, are treated as noise pixels given that the magnitude of the offset is unknown. For all cases, SNAPHU was applied with a mask but no coherence map, and DefoMax was set to 1.2. When calculating variance, for both the MC[5]-CGI and MC[4]-CGI, the minority classes were excluded. This was for a combination of reasons. As shown in Chapter 3, the accuracy for correctly classifying the minority classes is high. Furthermore, the magnitude of the offset for the minority classes is uncertain, whereas for C1 , C0 and $C\bar{1}$ it is known. Thus by excluding the minority classes, it gives an overall improved accuracy for the variance of the majority classes.

Figure 4.3a illustrates that using an MC[4]-CGI approach led to a lower RMSE compared with unwrapping synthetic interferograms using SNAPHU alone, demonstrating an overall improvement in the accuracy of the unwrapped phase. The RMSE scores when using SNAPHU alone are comparable to applying SNAPHU in combination with

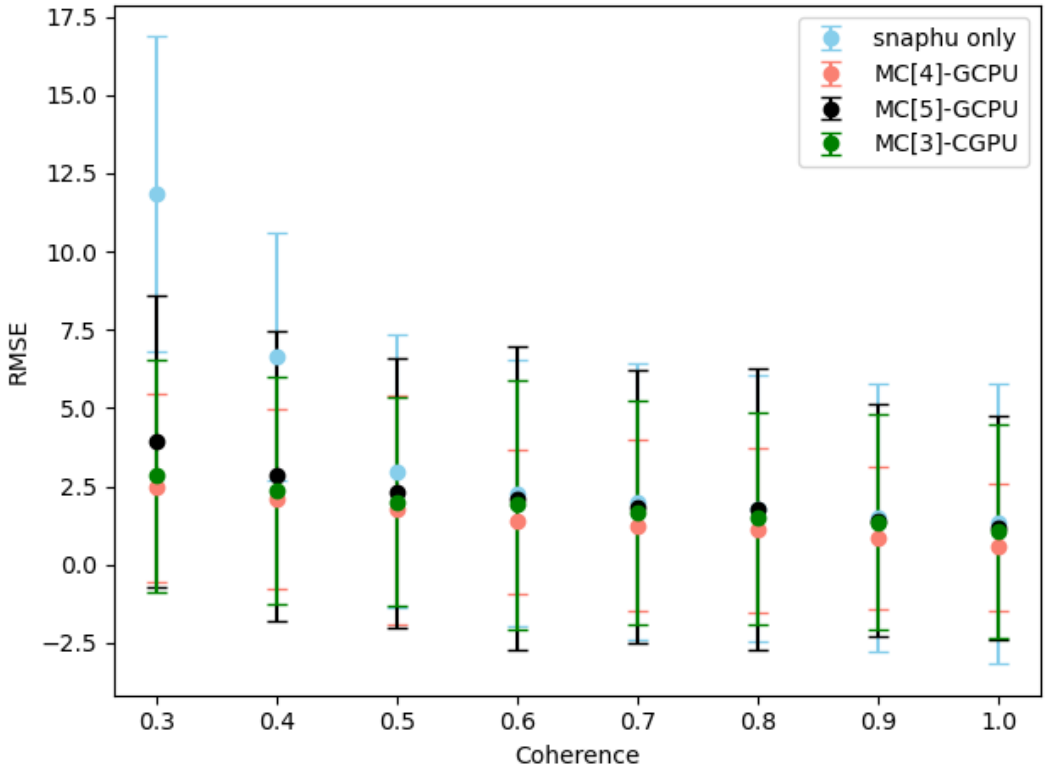
the MC[3]-CGI and MC[5]-CGI above a coherence level of 0.6; below 0.6, using the MC-CGI approach consistently outperforms applying SNAPHU alone. All the MC-CGI approaches have similar RMSE score, though MC[3]-CGI performs marginally better than MC[5]-CGI once the average coherence is below 0.6. Furthermore, whilst the RMSE of SNAPHU and the MC-CGI approaches all increase when coherence declines, the results in Figure 4.3a indicate that the dropoff in unwrapping performance for SNAPHU alone is much greater than the MC-CGI.

When considering that $C2^*$ and $\mathrm{C}\bar{2}^*$ are treated equivalently within the SNAPHU cost file, their combined contributions leads to a 68% and 80% (summed total of pixels classified as $C2^*$ and $\mathrm{C}\bar{2}^*$) being correctly identified as likely to be locations of discontinuities by MC[5]-CGI. This level of accuracy is comparable to the proportion correctly identified as $C|2|$ by the MC[4]-CGI, as shown for reference in Figure 3.8c. Consequently, it is likely the reduced accuracy by the MC[5]-CGI, particularly as coherence decreases, in identifying C1 and $C\bar{1}$ correctly, relative to the MC[3]-CGI and MC[4]-CGI classification accuracies for the same classes, led to the observed decline in unwrapping performance. Given this, further improvements to the performance of the MC[5]-CGI in gradient prediction could also lead to an overall performance of the unwrapped result.

Figure 4.4 and Figure 4.5 visually illustrate the conclusions drawn from Figure 4.3; the inclusion of the model gradient map information improves the overall output accuracy especially when in high phase noise level scenarios.

In Figure 4.5, the MC[4]-CGI combination is shown as the most reliable at high noise levels to improve unwrapped phase accuracy over the deformation region, whilst the MC-CGI generally is shown to be more accurate than using SNAPHU alone. As the phase noise level decreases, the SNAPHU with the MC[3]-CGI and MC[5]-CGI accuracy increases to become comparable to the MC[4]-CGI accuracy.

Figure 4.6b to e illustrates a case where the MC[4]-CGI approach led to the most significant improvement over the dense fringe region in the boxed area of the deformation.



(a) RMSE

Figure 4.3 – Figure shows Root Mean Square Error plotted against average coherence. RMSE is calculated on an interferogram basis for the entire test data set. The mean of the RMSE scores is calculated for all interferograms with the same average coherence. Coherence is not included in the SNAPHU configuration file. The comparison is made between unwrapping interferograms using (blue) SNAPHU alone, (pink) MC[4]-CGI, (black) MC[5]-CGI, (green) MC[3]-CGI. A line of best fit is included for each approach in the same colour as the points plotted. Coherence matches the coherence value used for the standard deviation in the test data generation step. Error bars are plotted as 1 standard deviation.

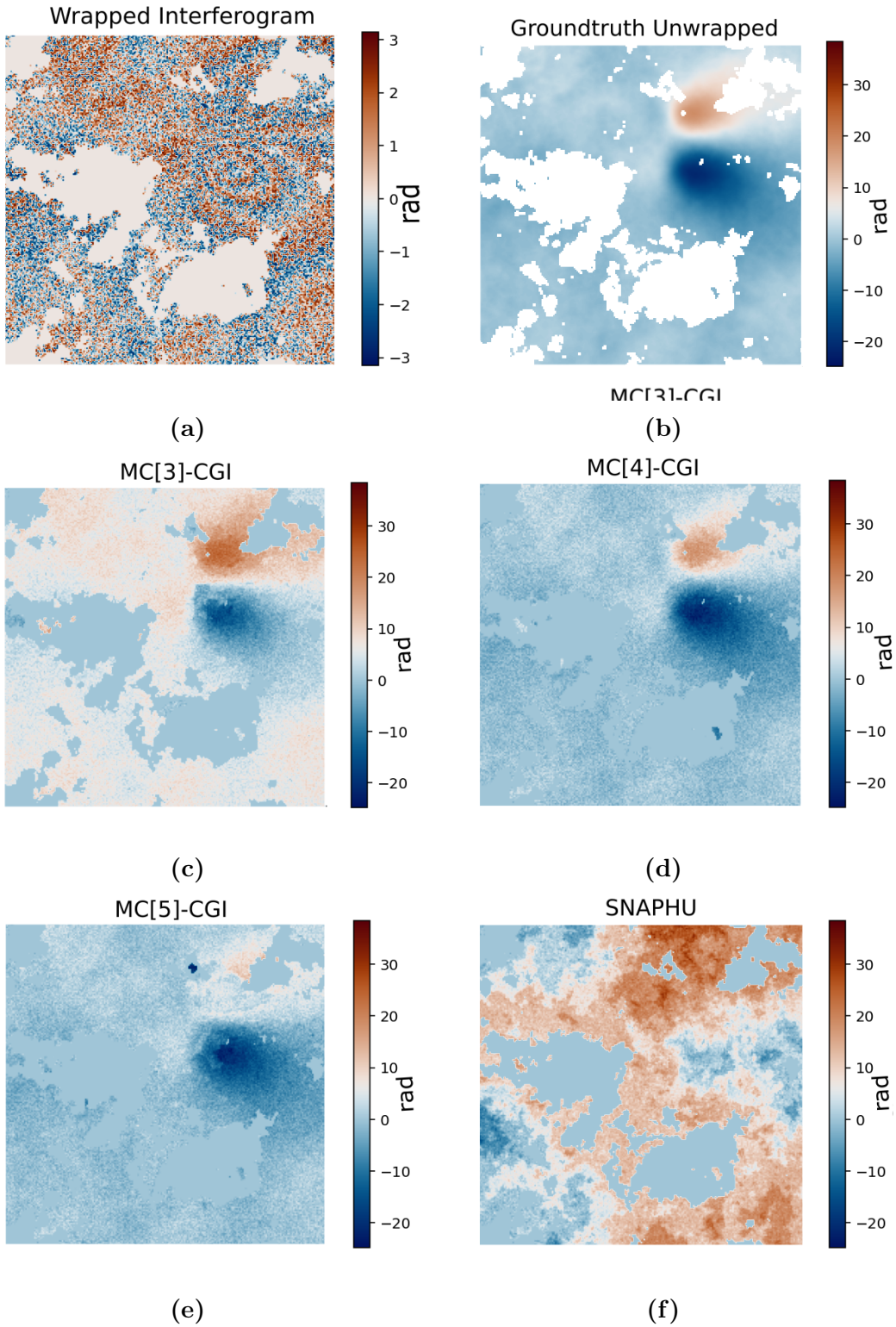


Figure 4.4 – Comparison of unwrapped results for the wrapped interferogram shown in (a). (a) Wrapped Interferogram, (b) Unwrapped Interferogram using MC[3]-CGI, (c) Unwrapped Interferogram using MC[4]-CGI, (d) Unwrapped Interferogram Groundtruth, (e) Unwrapped Interferogram using MC[5]-CGI, (f) Unwrapped Interferogram using SNAPHU only

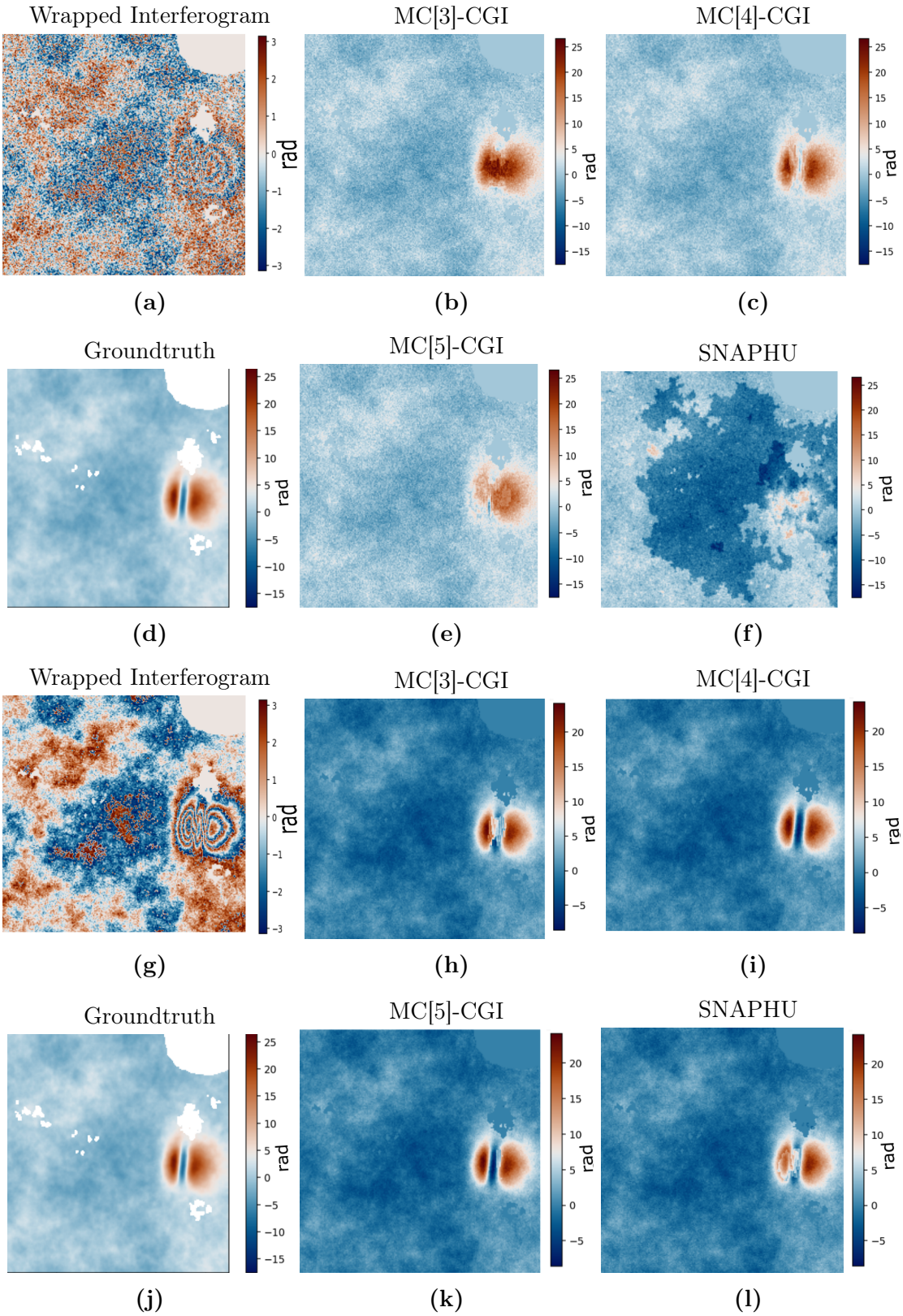


Figure 4.5 – Figure illustrating the MC-CGI led to an improved unwrapped result. (a) Wrapped Interferogram, (b-c, e-f) unwrapped result for (a) using MC[3]-CGI, MC[4]-CGI, MC[5]-CGI and SNAPHU respectively. (d) Unwrapped groundtruth of (a). (g) Wrapped Interferogram, (h - i, k - l) unwrapped result for (g) using MC[3]-CGI, MC[4]-CGI, MC[5]-CGI and SNAPHU respectively. (j) Unwrapped groundtruth of (g).

This region contained the highest number of non-zero gradients indicating a rapid change in the wrap count, caused by both the deformation and the phase noise, and thus is the area of the interferogram most likely to introduce unwrapping errors. The gradient maps shown in Figure 4.7 highlight that only MC[4]-CGI predicted the a high proportion of minority class $C|2|$ around the central region of high gradient change. Though MC[5]-CGI does have some minority class predictions, it is not so clearly associated with the central high fringe gradient region. Furthermore, the $C\bar{1}$ and C1 predictions in the centre of the deformation are in a less clear ringed pattern expected given the wrapped phase. This likely led to the overall poor unwrapped result shown in Figure 4.6.

For the wrapped interferogram shown in Figure 4.6f, where admittedly there is a higher overall coherence, MC[5]-CGI classifies pixels into the minority classes (Figure 4.7 in a similar location as the MC[4]-CGI model shown in Figure 4.7h, and also displays clearer fringe edge patterns (again similar to those made by the MC[4]-CGI model). The subsequent unwrapped interferogram which has been unwrapped using MC[5]-CGI, had a more accurate unwrapped result as shown in Figure 4.6i. As the MC[4]-CGI performed best in the deformation region for the high phase noise wrapped interferogram, especially around the high rate gradient change region where the majority of $C|2|$ predictions were made, it suggests that the identification of $C|2|$ can lead to an improved unwrapped result. The poor performance of MC[5]-CGI in the central deformation region for the wrapped interferogram with high phase noise likely relates to the poor performance at clearly identifying the edge of the deformation rings by the vertical model, and further improvement of the model will probably lead to an more improved unwrapped result.

Comparison with Chapter 2 Model and Algorithm

The purpose of utilising a cost-guided integration approach which used SNAPHU to integrate the wrap count gradient map rather than the MC-GI approach discussed in Chapter 2 was to increase the noise robustness of the integration whilst retaining the advantages of using wrap count gradient maps which explicitly identify large gradient

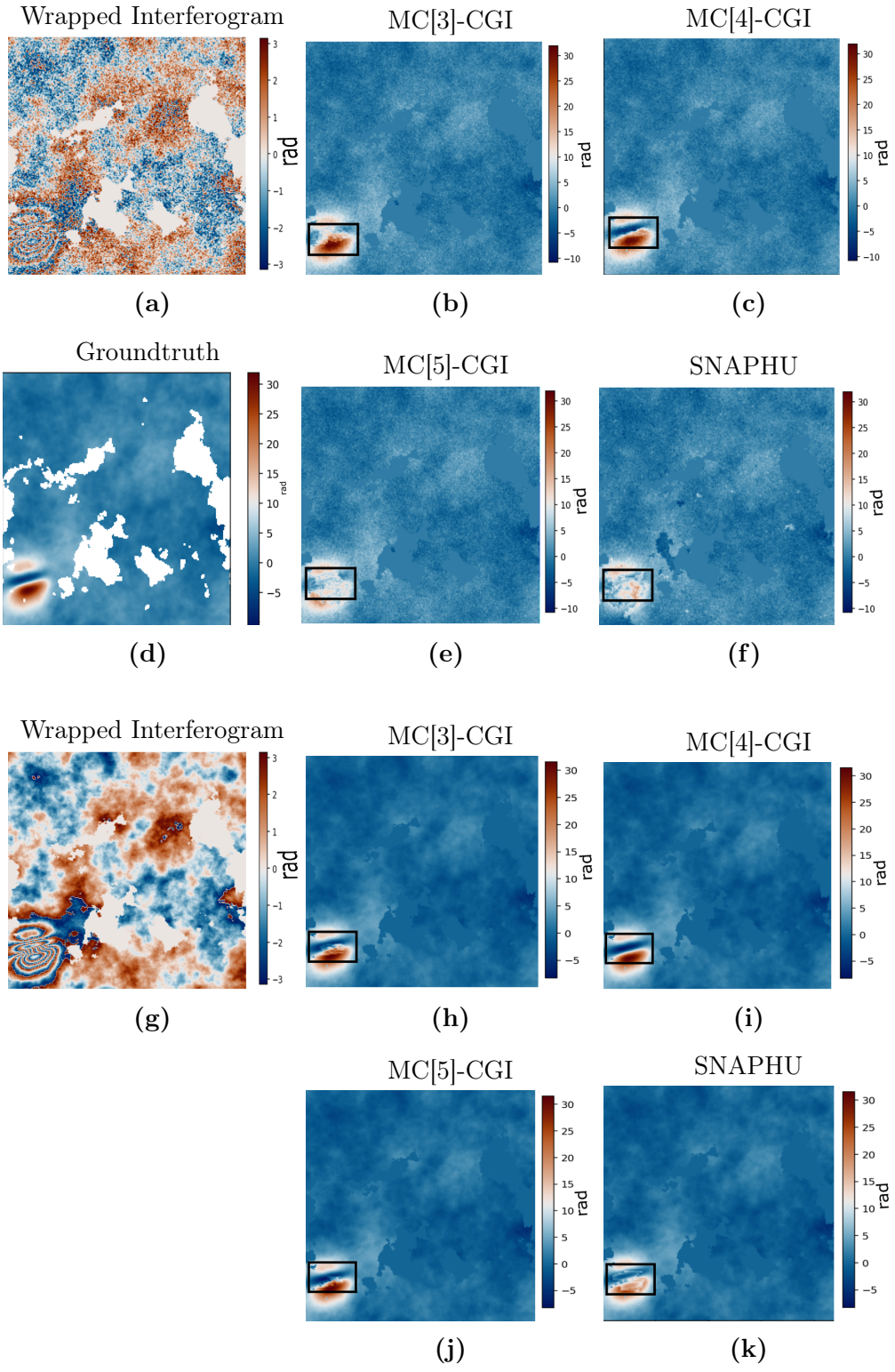


Figure 4.6 – Example of wrapped and unwrapped phase where a 4 class model has improved the unwrapped result in a high and low noise scenario (boxed area contains the highest fringe rate). (a) Wrapped phase in high noise scenario, (b, c, e, f) unwrapped phase for (a) using MC[3]-CGI, MC[4]-CGI, MC[5]-CGI and SNAPHU only respectively. (d) Groundtruth of (a), (g) Wrapped phase in low noise scenario, (h - k) unwrapped phase result for (g) using MC[3]-CGI, MC[4]-CGI, MC[5]-CGI and SNAPHU only respectively. A mask was set to True for the *NaN* regions (shown in white in the (a) and (g).

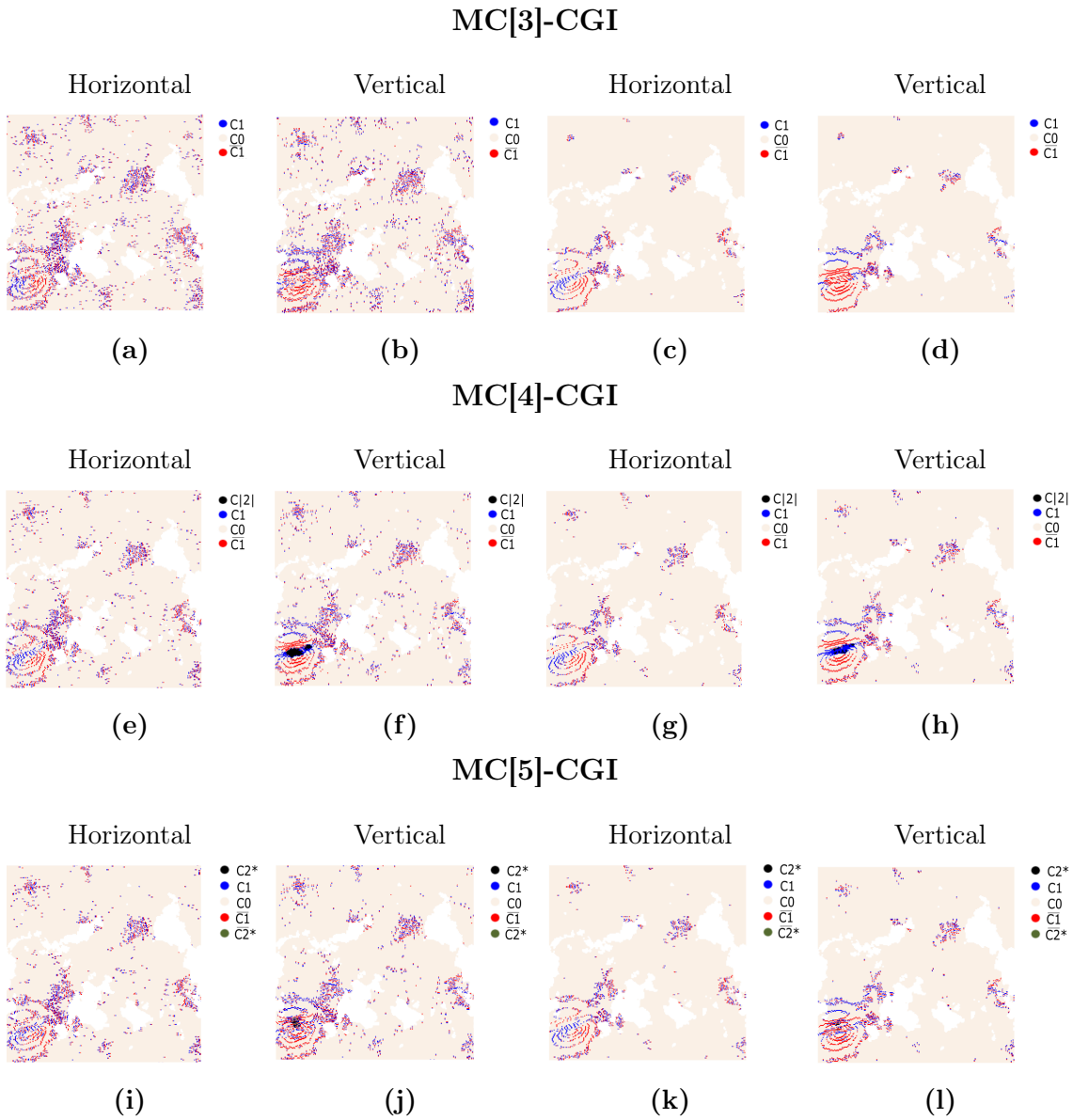


Figure 4.7 – Gradient maps used in the cost files to unwrap interferograms in Figure 4.6. Plots a,b,e,f,i and j correspond to Figure 4.6a. Plots c,d,g,h,k and l correspond to Figure 4.6g. (a) MC[3]-CGI gradient map in horizontal direction, (b) MC[3]-CGI gradient map in vertical, (c) MC[3]-CGI gradient map in the horizontal direction, (d) MC[3]-CGI gradient map for the vertical direction, (e) MC[4]-CGI gradient map for the horizontal direction, (f) MC[4]-CGI gradient map for the vertical direction, (g) MC[4]-CGI gradient map for the horizontal direction, (h) MC[4]-CGI gradient map for the vertical direction, (i) MC[5]-CGI gradient map for the horizontal direction, (j) MC[5]-CGI gradient map for the vertical direction, (k) MC[5]-CGI gradient map for the horizontal direction, (l) MC[5]-CGI gradient map for the vertical direction.

changes. I undertook analysis to understand how the each component of the MC[4]-CGI approach compared with the MC[4]-GI. The tests were performed using the synthetic dataset with the complex noise added.

Impact of MC[4]-GI Model Prediction Accuracy

Model	Macro-MIoU	Macro-Precision	Macro-Recall	Macro-F1
SNAPHU	0.26	0.36	0.29	0.31
MC[4]-GI	0.43	0.73	**0.52**	0.54

Table 4.1 – -
GI statistics when tested on synthetic data with complex noise]Table of statistical testing results when SNAPHU in a four case scenario and the MC[4]-GI model from Chapter 2 are applied to a test data set of size 320 prepared using circular Gaussian noise, where the standard deviation was varied to simulate coherences from 0.3 to 1.0.

As can be seen in Table 4.1, the MC-GI models trained in Chapter 2 show poor classification performance. This is expected when you consider that whilst the wrapped data used to train the models in Chapter 2 and Chapter 3 looked comparable, there is a noticeable difference in distribution between the training data sets.

The poor gradient classification performance, particularly at low coherence levels, means that the greedy algorithm performs poorly, as can be seen in the unwrapped result shown in Figure 4.8f. This is likely due to the greedy algorithm being sensitive to the accuracy of the classification seen in Figure 4.8c and d. Though guided by certainty during unwrapping, where a high proportion of gradients are inaccurate, the greedy algorithm is limited. As the level of noise decreases, the performance of the gradient prediction and the complete MC-GI process improves, as shown in Figure 4.9 and seen in the unwrapped result shown in Figure 4.10c,d and f. Given that the MC[4]-CGI has also been impacted by the poor model performance (Figure 4.8e), this indicates the model performance is limiting the overall performance.

Impact of MC[4]-CGI Model Prediction Accuracy

In the above examples, the complete MC[4]-GI and MC[4]-CGI processes utilised the model trained in Chapter 2, applied to the synthetic dataset with complex noise. Here,

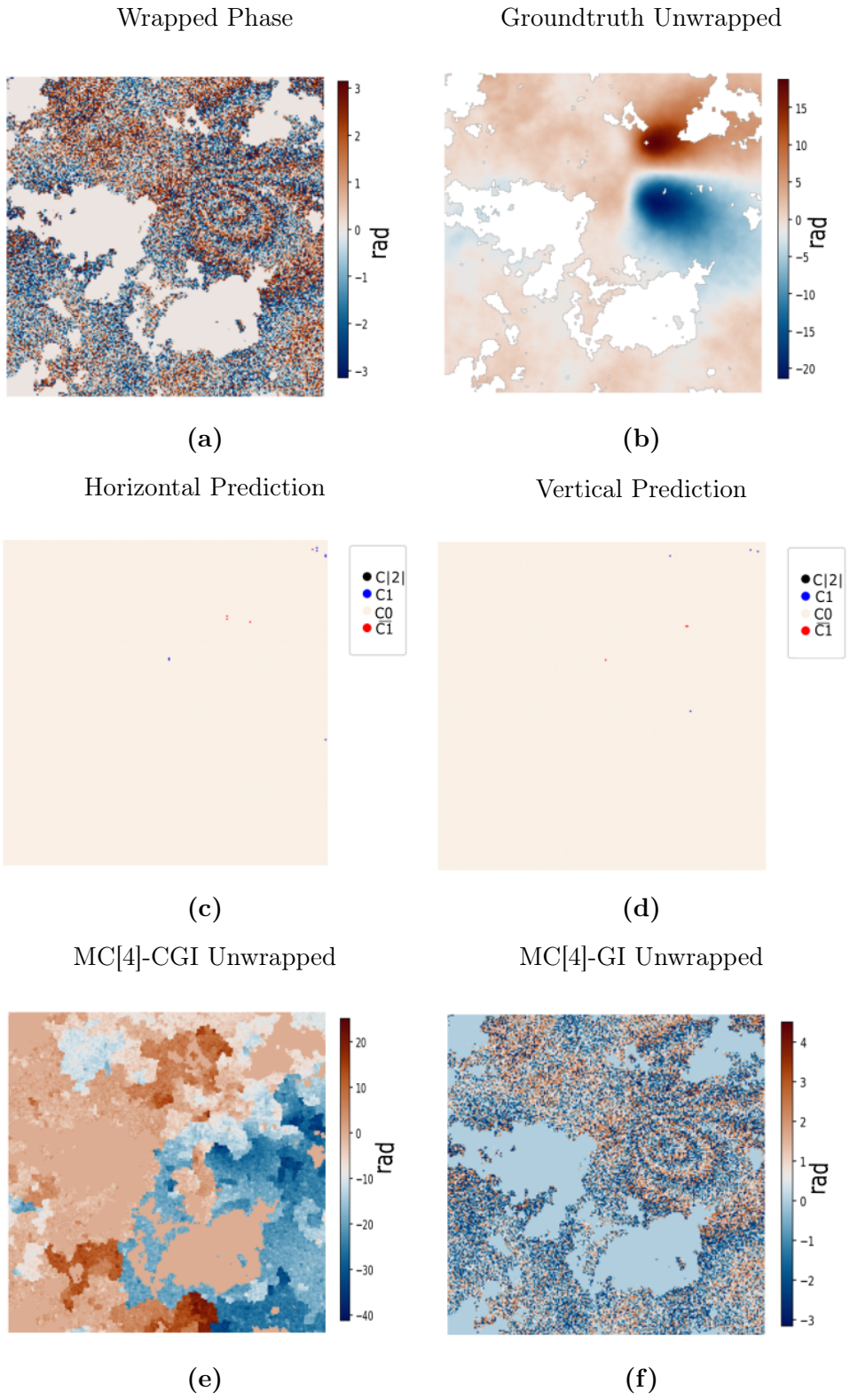


Figure 4.8 – Comparison of unwrapped results for the wrapped interferogram shown in (a). Chapter 2 model was used to predict the wrap count gradients. Coherence: 0.4. (a) Wrapped interferogram, (b) groundtruth unwrapped interferogram, (c) horizontal gradient prediction, (d) vertical gradient prediction, (e) unwrapped interferogram using MC[4]-CGI, (f) unwrapped interferogram using greedy algorithm.

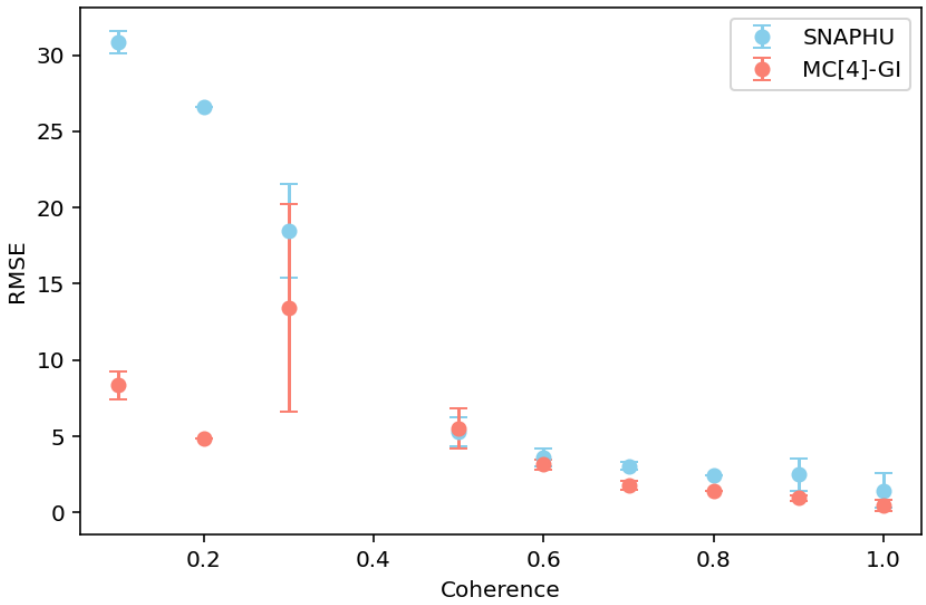


Figure 4.9 – Figure showing the RMSE for interferograms unwrapped using the Chapter 2 setup (MC[4]-GI) and SNAPHU.

I instead test the MC[4]-GI and MC[4]-CGI processes using the best performing 4 class model from Chapter 3.

When tested on a synthetic dataset, it is seen in Figure that the greedy integration algorithm approach achieves similar levels of accuracy during unwrapping as the cost guided approach when combined with the Chapter 3 model output, particularly at high coherence levels. In low coherence scenarios, Figure 4.11 suggests that a cost based approach may be more advantageous, given a lower RMSE is achieved.

The greedy algorithm is intrinsically sensitive to incorrectly labelled wrap count gradients, which are more likely to occur in high noise locations, and where high deformation rates are present. In such scenarios, whilst the MC[4]-CGI performance decreases, the MC[4]-GI more often incorrectly unwraps such areas (see Figure 4.12e vs. f). This is likely due to the increased risk of the classification model incorrectly classifying pixels in the area. However, more widely across the interferogram, the greedy algorithm in combination with the Chapter 3 classification model still performs surprisingly well, even in high noise scenarios, with less dense fringes (see Figure 4.13e and f).

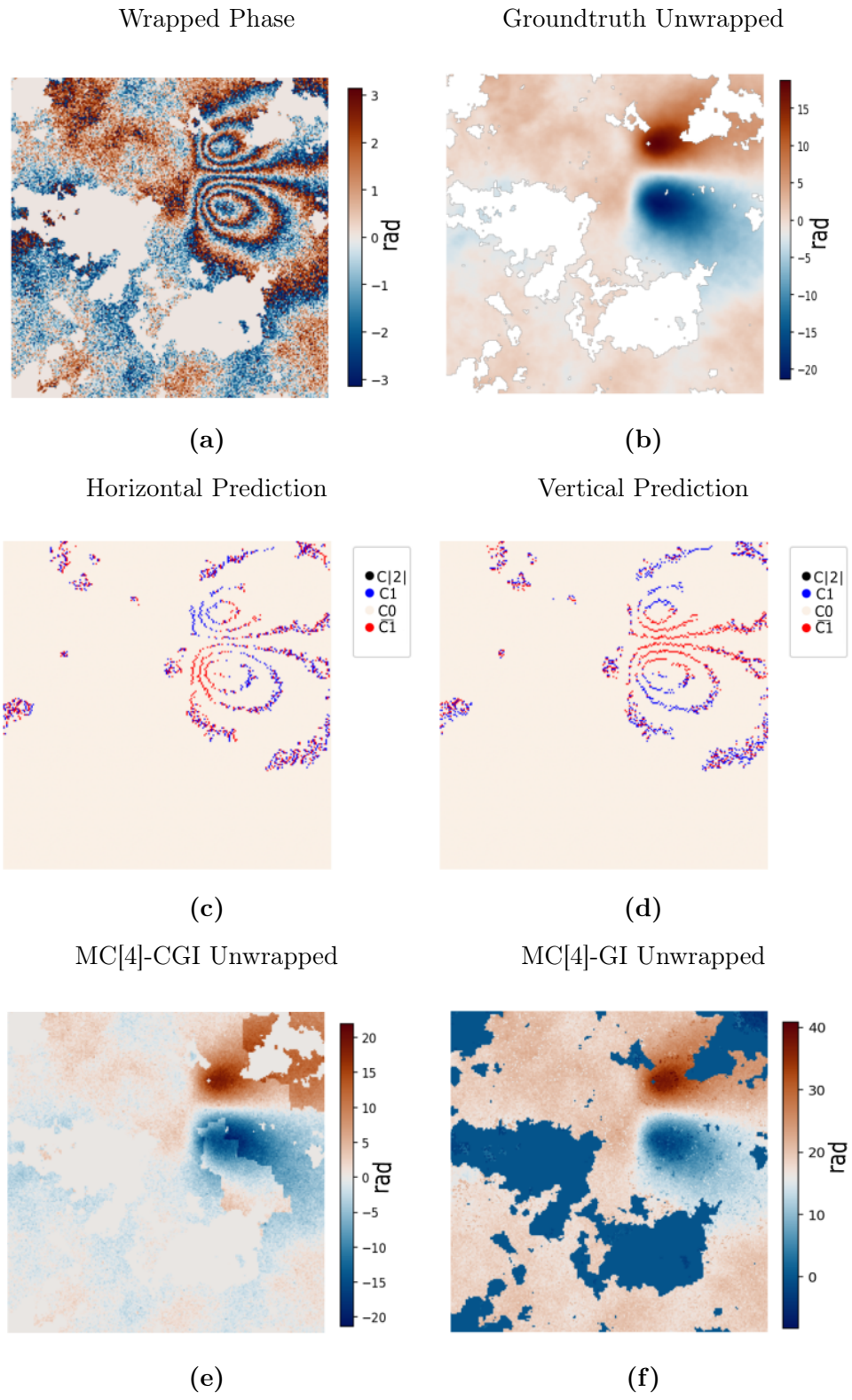


Figure 4.10 – Comparison of unwrapped results for the wrapped interferogram shown in (a). Chapter 2 model was used to predict the wrap count gradients. Coherence: 0.6. (a) Wrapped interferogram, (b) groundtruth unwrapped interferogram, (c) horizontal gradient prediction, (d) vertical gradient prediction, (e) unwrapped interferogram using MC[4]-CGI, (f) unwrapped interferogram using greedy algorithm.

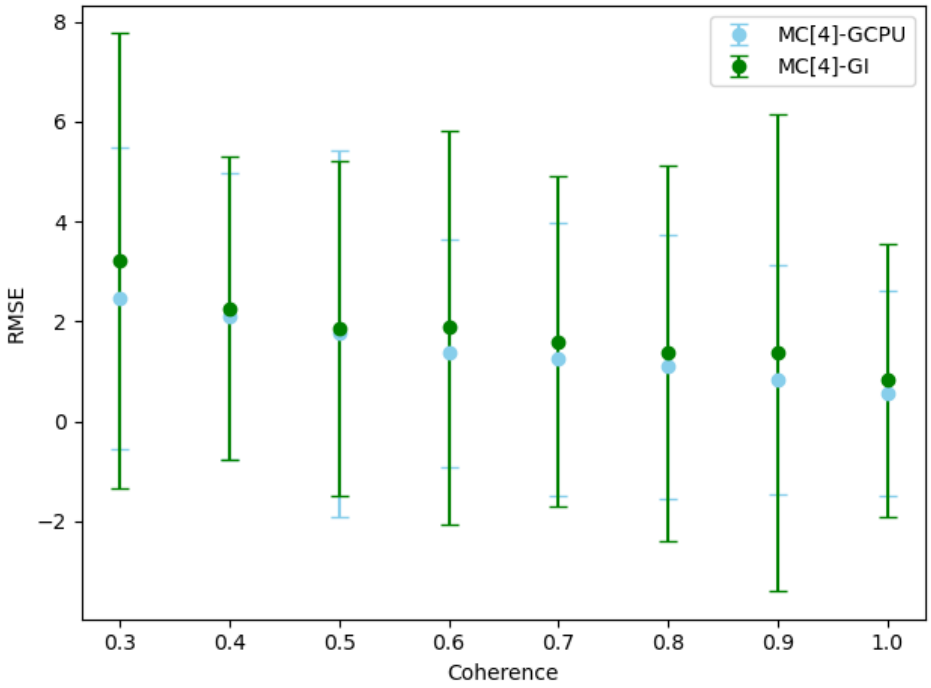


Figure 4.11 – Figure showing the RMSE at varying coherence when the best performing Chapter 3 model is applied in combination with the greedy algorithm (MC[4]-GI) and cost based unwrapping (MC[4]-CGI).

Therefore, both methods can be used to lead to a correctly unwrapped interferograms, though in areas with both high noise and high deformation, the MC[4]-CGI may be more suitable for a better unwrapped results over the deformation region.

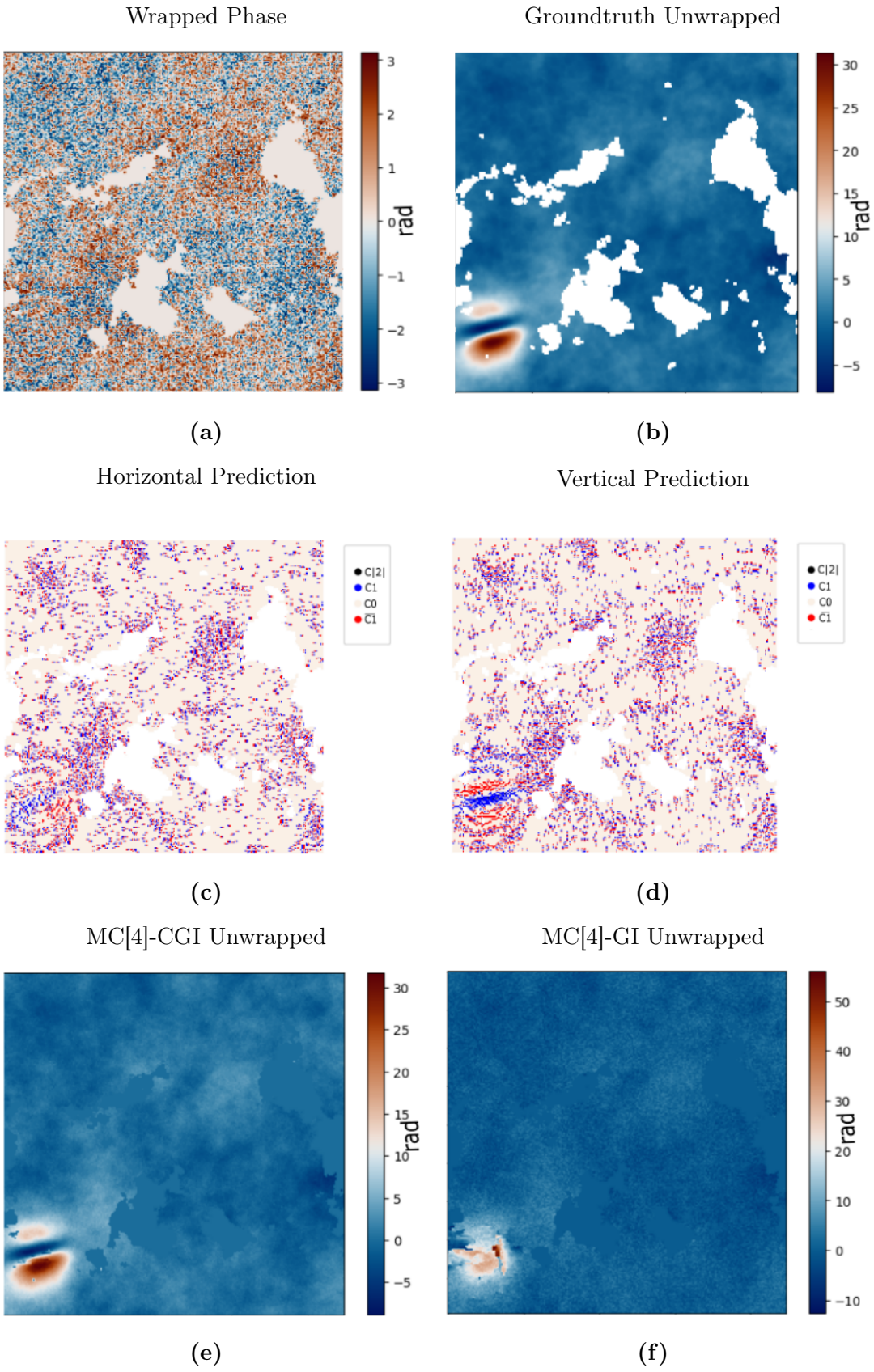


Figure 4.12 – Comparison of unwrapped results for the wrapped interferogram shown in (a). (a) Wrapped interferogram, (b) groundtruth unwrapped interferogram, (c) horizontal gradient prediction, (d) vertical gradient prediction, (e) unwrapped interferogram using MC[4]-CGI, (f) unwrapped interferogram using greedy algorithm.

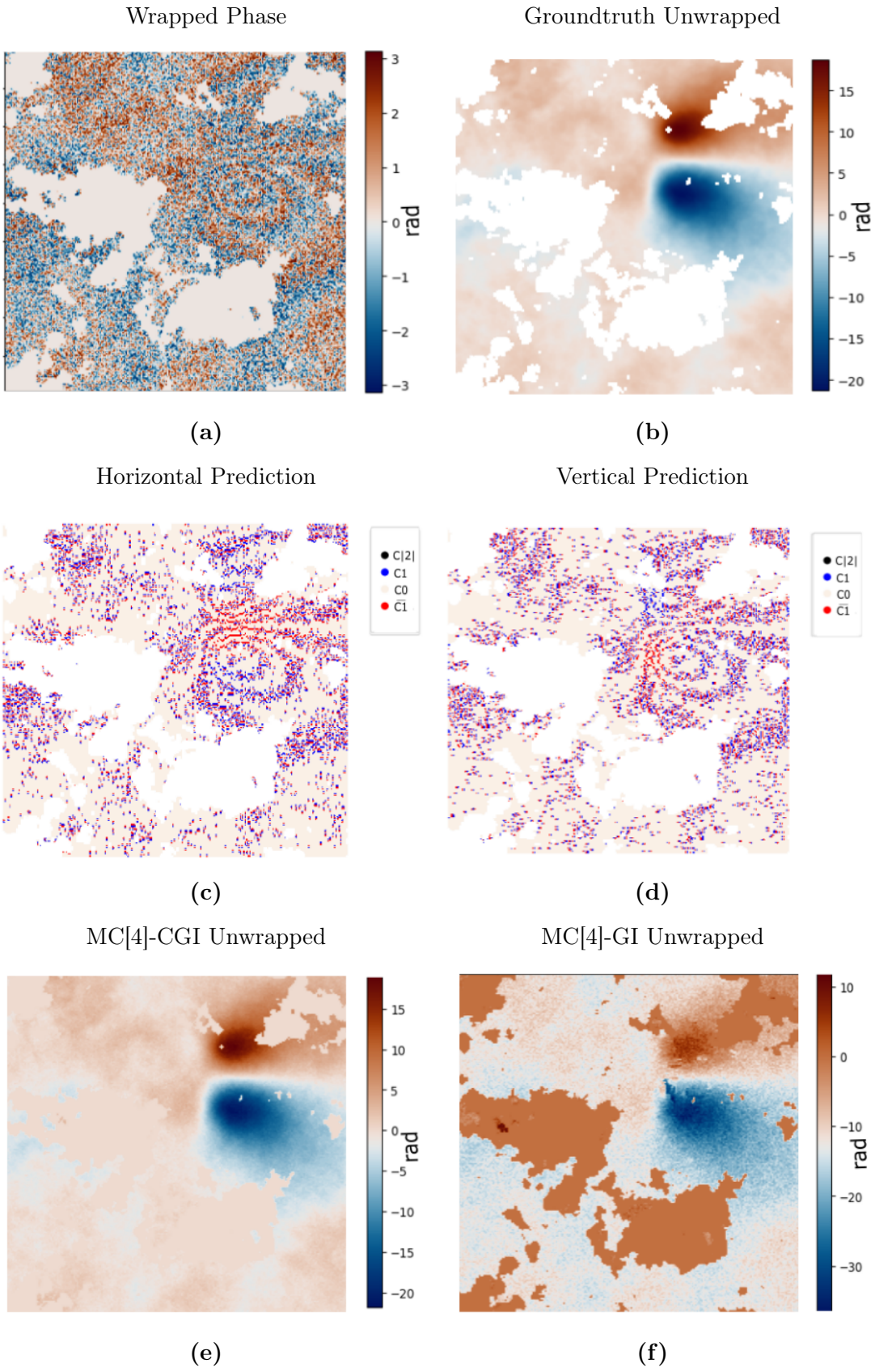


Figure 4.13 – Comparison of unwrapped results for the wrapped interferogram shown in (a). (a) Wrapped interferogram, (b) groundtruth unwrapped interferogram, (c) horizontal gradient prediction, (d) vertical gradient prediction, (e) unwrapped interferogram using MC[4]-CGI, (f) unwrapped interferogram using greedy algorithm.

4.3.2 Testing Unwrapping using Sentinel-1 Interferograms

The MC-CGI approach improves the unwrapped phase of synthetic interferograms when compared to SNAPHU, particularly in regions of high fringe rates. The results in the previous section also indicate the inclusion of $C|2|$ enhances the overall unwrapped result in the deformation region for synthetic interferograms. However, it is imperative that the improvements seen when MC-CGI is applied to synthetic data are transferred to real Sentinel-1 data. To assess whether this is the case, the MC-CGI approach was applied to Sentinel-1 interferograms and the results compared with applying SNAPHU alone using loop closure error (see Chapter 2 for a description).

Figure 4.14 demonstrates when the MC-CGI approach is applied, a reduction in loop closure error is achieved, and is independent of whether a 3,4 or 5 model is used, and only SNAPHU had a non-zero loop closure error in the region of deformation as highlighted in the boxed area of Figure 4.14p. This was due to incorrect unwrapping of the fringes (shown in the boxed region of Figure 4.14q and r), whereas in the remaining plots showing wrap count (Figures 4.14e,f,i,j,m and n), the fringes are clearly distinguishable from each other and are consistent with the wrapped phase fringe edges. In the boxed region, the majority of wrap count gradients were identified as the majority classes, which all models have high classification accuracy for (see Figure 4.15 for each corresponding wrap count gradient map in the horizontal and vertical direction). Given that the few $C|2|$ pixels predicted by the MC[4]-CGI in Figure 4.15f and m are circumventable during the integration process and that the wrap count map shown in Figure 4.14f and m are closely comparable to the MC[3]-CGI and MC[5]-CGI returned wrap counts, it is unlikely $C|2|$ is having a distinguishable impact upon the unwrapped result.

To identify whether in the current set-up, the choice of 3,4 or 5 class model impacted upon the unwrapped phase result, phase unwrapping was performed on a different section of Frame 123D 05292 242525, and the coherence lowered to 0.05 to ensure gradients were present which were likely to be exceed one phase cycle change. Whilst it was obvious

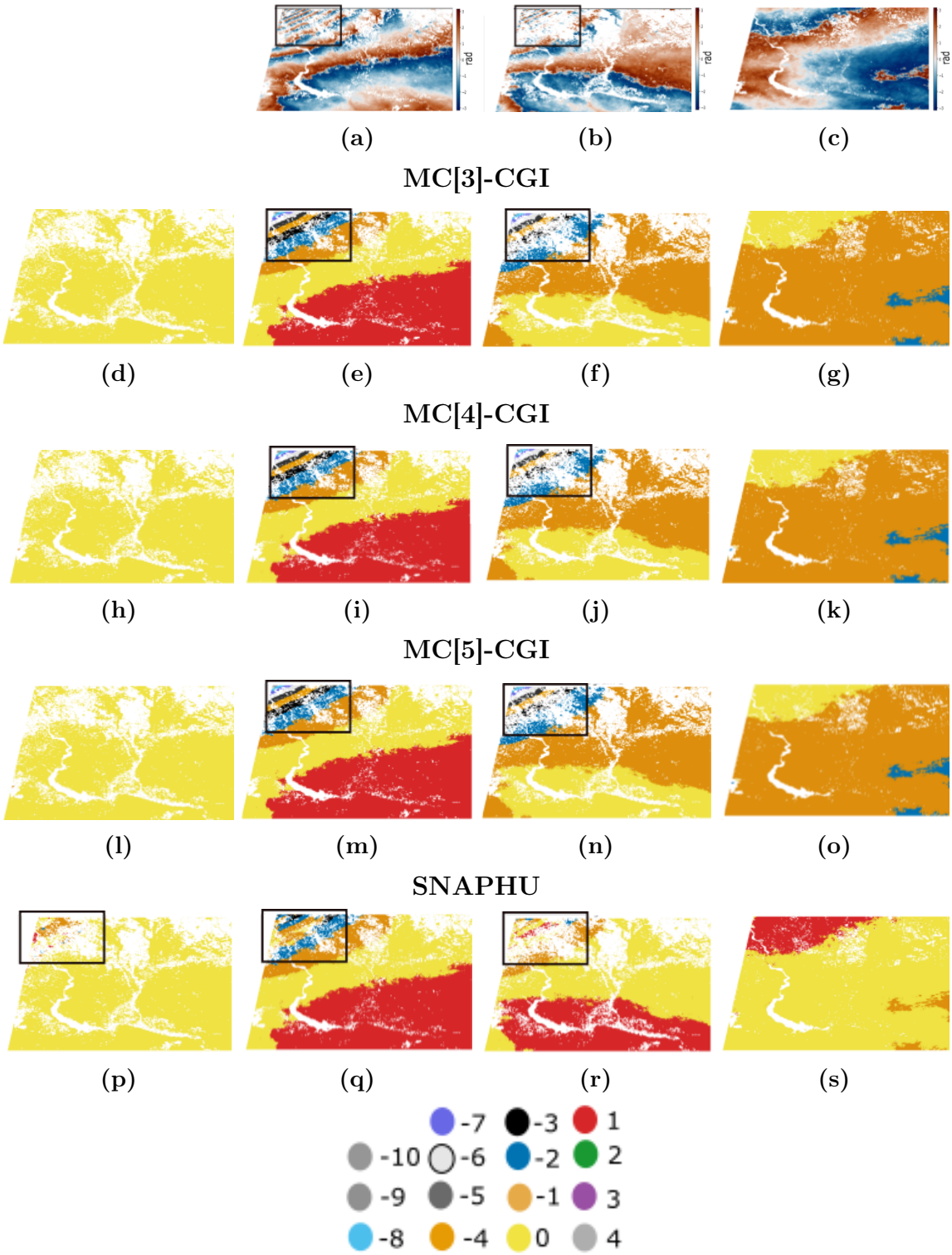


Figure 4.14 – Loop closure and wrap count for LiCSAR Frame 123D 05292 242525 test for interferograms dated: (a) 2023/01/12 - 2023/02/17, (b) 2023/01/12 - 2023/03/01 and (c) 2023/02/17 - 2023/03/01. (d) Loop Closure Error (1.5 %) for the MC[3]-CGI, (e - g) MC[3]-CGI wrap count results for wrapped interferograms a-c respectively, (h) Loop Closure Error (1.4 %) for the MC[4]-CGI, (i - k) MC[4]-CGI wrap count results for wrapped interferograms a-c respectively, (l) Loop Closure Error (1.5 %) for the MC[5]-CGI, (m - o) MC[5]-CGI wrap count results for wrapped interferograms a-c respectively, (p) Loop Closure Error (5.0%) where SNAPHU has been applied, (q-s) wrap count results using SNAPHU for the wrapped interferograms (a-c) respectively. SNAPHU: DefoMax = 1.2, mask applied (coherence threshold = 0.1). In (p-r), the boxed region highlights loop closure error source.

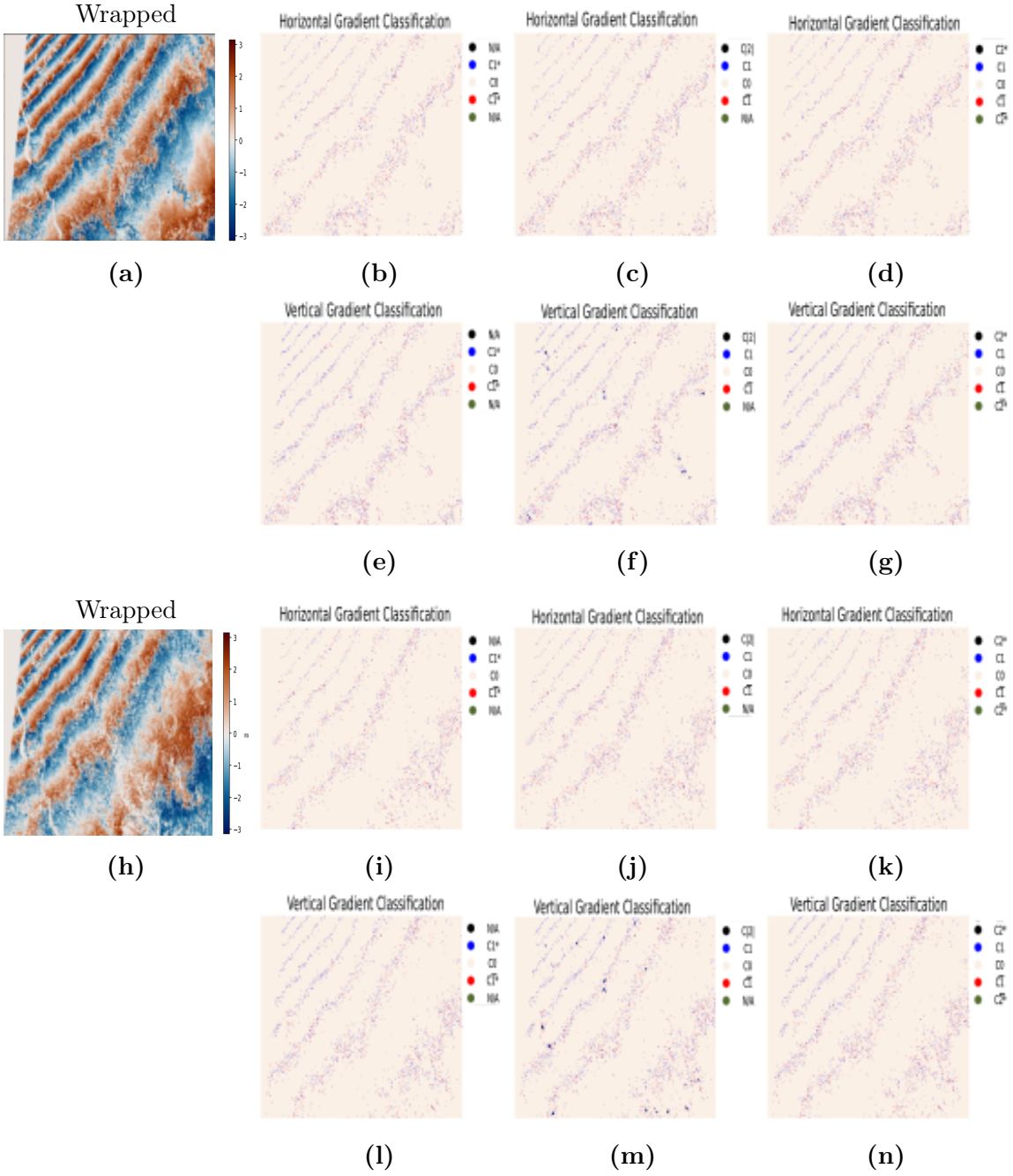


Figure 4.15 – Comparison of gradient predictions in the horizontal and vertical direction used to unwrap the LiCSAR Frame 123D 05292 242525 test for each of MC[3]-CGI, MC[4]-CGI and MC[5]-CGI for the boxed regions shown in Figure 4.14. Coherence threshold set to 0.1. (a) Boxed region from Figure 4.14a, (b - d) horizontal wrap count gradient estimations from MC[3]-CGI, MC[4]-CGI and MC[5]-CGI respectively for the region shown in (a). (e -g) Vertical wrap count gradient prediction for MC[3]-CGI, MC[4]-CGI and MC[5]-CGI respectively for the region shown in (a). (h) Boxed region of Figure 4.14b, (i - k) horizontal wrap count gradient estimation for MC[3]-CGI, MC[4]-CGI and MC[5]-CGI respectively for the region shown in (h). (l - n) Vertical wrap count gradient map for MC[3]-CGI, MC[4]-CGI and MC[5]-CGI respectively for the region shown in (h). Wrapped phase is in radians.

the use of wrap count gradient maps to guide SNAPHU decreased the loop closure error compared with SNAPHU alone, the model choice did not impact the overall loop closure error score, despite more $C|2|$ pixel gradients being identified.

As can be seen in Figure 4.16, the loop closure error is higher when SNAPHU only is used for phase unwrapping, with higher loop error along the top of the island, where the most noise is present. Whilst there are still phase unwrapping errors present in this region when the MC-CGI approach is taken (Figure 4.16p - r), it is much lower. Most of the loop closure error in Figure 4.16s is due to incorrect wrapping in Figure 4.16e, where there is a transition from clear signals returned from the bottom of the interferogram, to an increased level of phase noise towards the top of the interferogram. The results shown in Figure 4.16 indicate using SNAPHU to integrate the MC-CGI wrap count gradients rather than the MC-GI approach has increased the robustness of the unwrapping to phase noise leading to a lower loop closure error than results seen in Figure 2.22, despite less of the interferograms being masked. Secondly, the use of the MC-CGI approach, rather than using SNAPHU alone, leads to improved accuracy in the unwrapped result, particularly in areas of increased phase noise.

Finally, though MC[4]-CGI led to the best unwrapped results when applied to synthetic interferograms, for the case shown in Figure 4.16, the MC[3]-CGI and MC[5]-CGI approaches have a lower loop closure error. This difference is due to the change in phase shown in Figure 4.16a, where there is a large proportion of pixels masked, where a phase unwrapping error has been made over that transition by the MC[4]-CGI approach. This is likely due to the groupings of $C|2|$ impacting the transition, indicating that treating $C|2|$ as equvilent to masked pixels when groupings occur at transitions where there are high levels of masking already may lead to a similar effect as the isolated coherent pixel regions.

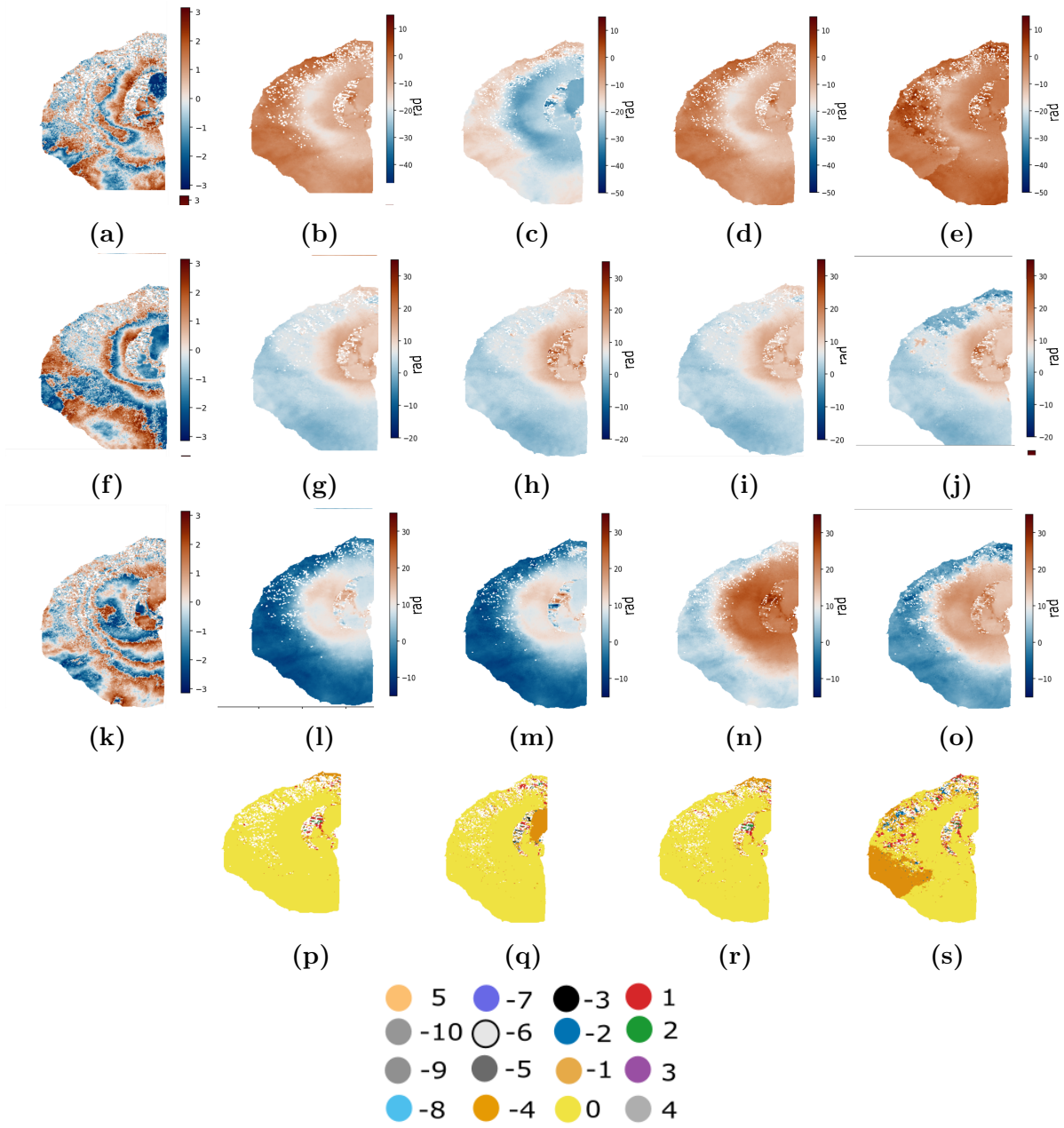


Figure 4.16 – Figure showing a case where the MC-CGI approach has led to improved phase unwrapped result compared with using SNAPHU alone in noisy regions. Coherence threshold = 0.01, DefoMax = 1.2, mask used, LiCSAR Frame: 140D SM FGbr s4 test. Dates of acquisitions: 2015/06/12, 2015/09/16 and 2016/03/02. (a) Wrapped Phase from acquisitions dated 2015/06/12 to 2015/09/16, (b) MC[3]-CGI unwrapped phase for (a), (c) MC[4]-CGI unwrapped phase for (a), (d) MC[5]-CGI unwrapped phase for (a), (e) SNAPHU unwrapped phase for (a), (f) Wrapped phase from acquisitions dated 2015/06/12 and 2016/03/02, (g) MC[3]-CGI unwrapped phase for (f), (h) MC[4]-CGI unwrapped phase for (f), (i) MC[5]-CGI unwrapped phase for (f), (j) SNAPHU unwrapped phase for (f), (k) Wrapped phase from acquisitions dated 2015/09/16 and 2016/03/02, (l) MC[3]-CGI unwrapped phase for (k), (m) MC[4]-CGI unwrapped phase for (k), (n) MC[5]-CGI unwrapped phase for (k), (o) SNAPHU unwrapped phase for (k), (p) loop closure error plot for MC[3]-CGI (13%), (q) loop closure error plot for MC[4]-CGI (16%), (r) loop closure error plot for MC[5]-CGI (13%), (s) loop closure error plot for SNAPHU (27%).

4.4 Conclusions

In this Chapter, I have proposed a Multi-Class Cost-Guided Integrator (MC-CGI) process where wrap count gradient maps in the horizontal and vertical direction are integrated using SNAPHU. In the process, the offset is determined by the wrap count gradients and the variance is calculated using the model prediction certainty and given as a priori knowledge to SNAPHU using cost files. This has expanded upon work in Chapter 2 both due to the model being trained on a synthetic data set with circular gaussian noise added and by using the more robust method to integrate the wrap count gradient maps, namely SNAPHU. The motivation for this work was to retain the benefit of using at least one additional class when training the model to explicitly identify large gradient changes, whilst more realistically simulating Sentinel-1 phase noise to improve the robustness of the approach.

Result indicate the MC-CGI approach leads to an overall improved accuracy in the unwrapped phase returned in the areas of deformation both for synthetic and real Sentinel-1 data in regions of high deformation compared to using SNAPHU alone. The MC[4]-CGI performed particularly well on synthetic data at unwrapping deformation. When applied to Sentinel-1 data, the accuracy of the unwrapped phase was shown to be comparable or better than SNAPHU applying the MC-CGI approach, though the choice of 3, 4 or 5 class model makes little impact on the unwrapped phase result.

As highlighted in Chapter 4's methods section, $C|2|$, $C2^*$ and $\mathrm{C}\bar{2}^*$ are treated as equivalent to masked noise pixels by SNAPHU due to the high degree of uncertainty associated with the magnitude of the phase change beyond knowing that the absolute phase cycle gradient exceeds +1. Therefore, there are several avenues for future work. The simplest involves retaining $C2^*$ and $\mathrm{C}\bar{2}^*$, while introducing a C3 class to encapsulate all absolute phase gradient changes exceeding 3, allowing the $C2^*$ and $\mathrm{C}\bar{2}^*$ to be used explicitly as useful information. This approach does raise an ongoing methodological challenge to determine the point at which the potential benefits of incorporating additional classes

are outweighed by the diminishing likelihood of such cases occurring. Moreover, as the number of classes increases, the imbalance of the training data set will become more pronounced, potentially compromising the model performance, especially as it is illustrated in the results that the overall classification accuracy minority classes as each additional class is added decreases.

An alternate avenue for future work involves exploring how to better utilise the current model classification certainty to influence the variance estimation. This approach would allow the use of the minority class probabilistic values. At present, the variance for the majority classes are calculated using the associated certainties, with minority classes excluded. For the MC-CGI, this would involve considering the probability of C1 and $C\bar{1}$. Under the assumption that where the next highest probability a $C|2|$ corresponds to is C1 , it is likely the gradient is 2 or more, whereas a second-most probable $C\bar{1}$ would suggest the gradient is -2 or less. This would also rely upon the absolute gradient change being close to 0, rather than a larger change in phase cycle, and that the likelihood of the phase gradient exceeding 2 increases as the magnitude of the gradient change becomes increasingly distant from 2. The same reasoning could be made for MC[5]-CGI, without the initial step to separate $C|2|$ into $C2^*$ and $C\bar{2}^*$. An initial validation study needs to be made to evaluate whether these assumptions are reasonable and the model probabilities are consistent with the assumptions before this could be implemented.

Chapter 5

Please Mind the Gap

At the end of Chapter 2, two limitations were highlighted when applying MC-GI to phase unwrapping. In this Chapter, I present work where I have explored how the use of a generative AI model can help to minimise the second limitation, namely how the isolation of coherent regions can lead to an offset in wrap count resulting in incorrect phase unwrapping. This issue has also been shown to arise when unwrapping using SNAPHU. To do this, a Brownian Bridge Diffusion Model (BBDM) (Bo et al. 2023) was trained to replace masked noise regions which isolated coherent regions by generating possible wrapped phase that realistically mimicked the patterns that might be expected to connect the coherent regions.

5.1 Introduction

The purpose of unwrapping interferograms is to restore a continuous phase from the wrapped phase. Phase unwrapping techniques generally apply the Itoh assumption; neighbouring pixels are broadly expected to differ by less than half a phase cycle (Itoh 1982). In regions of high phase noise or where there has been insufficient sampling relative to the level of deformation, the condition no longer holds and discontinuities are

present (Zhong et al. 2024). A discontinuity is defined as locations where the wrapped difference of wrapped phase differs from the true phase difference between adjacent pixels (Wu et al. 2022). For regions containing discontinuities, there is an increased risk that phase unwrapping errors will be introduced, and strategies are required to effectively minimise error introduction whilst integrating the wrapped phase.

A strategy employed to reduce phase unwrapping errors is the use of additional information prior to phase unwrapping which provides information about the pixel gradients and helps to guide the integration pathway. Quality guided methods rank pixels from high to low reliability (Zhong et al. 2014; Arevalillo-Herraez et al. 2016) such as coherence maps (Wang et al. 2021) or the density of residues (Arevalillo-Herraez et al. 2016). By leaving those pixels of lowest quality until last, theoretically the chance of error propagation will be reduced. Alternative unwrapping methods identify residues which are the locations at the end of, or along the discontinuities, and optimise the way to connect the residues by minimising the length of the connections (Goldstein et al. 1988; An et al. 2000) or cost associated with the connections (Wu et al. 2021a) with the overall goal of minimising the difference between the wrapped phase gradient and unwrapped phase gradient during the integration step.

SNAPHU (Statistical-cost, Network-flow Algorithm for Phase Unwrapping) (Chen 2001) has the option to use a mask and/ or coherence to inform the cost associated with an unwrapped gradient. A low cost is associated with each difference between the unwrapped and wrapped phase gradients in the masked and low coherence areas due to the high degree of uncertainty associated with the wrapped phase gradient. These are termed "neutral costs" by SNAPHU, meaning that they do not contribute towards the overall cost calculation for determining the best unwrapped result (Chen et al. 2001).

Existing methods rely on the assumption that there is neighbourhood continuity (Gao et al. 2022a) meaning that noisy, low coherence regions are not expected to isolate coherent pixel areas (Gao et al. 2023). This is not always the case; as highlighted in earlier Chapters, cases of coherent patches being isolated by noise patches can occur in Sentinel-

1 data, increasing the potential for incorrect phase unwrapping due to the regions being incorrectly offset relative to one another (Gao et al. 2023). The accuracy of unwrapping using MC-GI and SNAPHU was impacted by the isolation of coherent regions, as demonstrated in Figure 2.17c and e. Few studies have investigated alternative ways to "fill in" regions which are masked (masked regions are regions considered to contain little to no useful signal information), for the purpose of improving the accuracy of the final 2D phase unwrapped result (Gao et al. 2023; Gao et al. 2022a). It should be noted that filtering will help reduce the problem of noise somewhat by allowing smaller scale phase noise to be dealt with, however, this study looks at those situations where the area below a coherence threshold has isolated coherent regions, such as those shown in Figure 5.1.

Image inpainting replaces masked regions with plausible values based upon the expected data distribution. Rather than just using interpolation, nearest neighbour or assuming a constant value across the gap, I explore whether a generative AI model can be used to fill the masked regions to connect isolated coherent patches with a realistic looking signal. The overall aim is to improve the accuracy of the subsequent unwrapping by reducing the possibility of an offset being present between the isolated region.

Image-to-image translation is a deep learning technique where models are trained to map between the distributions of two domains (Bo et al. 2023), using an image input to generate an image output. Generative adversarial networks (GANS), described in Chapter 1, are early examples of a model architecture used for image-to-image translation, showing great success at generating synthetic images (Dhariwal et al. 2021). However, they were limited by sensitivity of the choice of starting parameters, restricted generalisation and the instability of training.

Variation auto-encoders (VAE) attempt to learn to reconstruct a data distribution which is close to the distribution of the original input (Cao et al. 2023). Whilst more generalisable, they over-smooth, leading to loss of finer scale details.

In recent years, Denoising Diffusion Probabilistic Models (DDPM) have been embraced

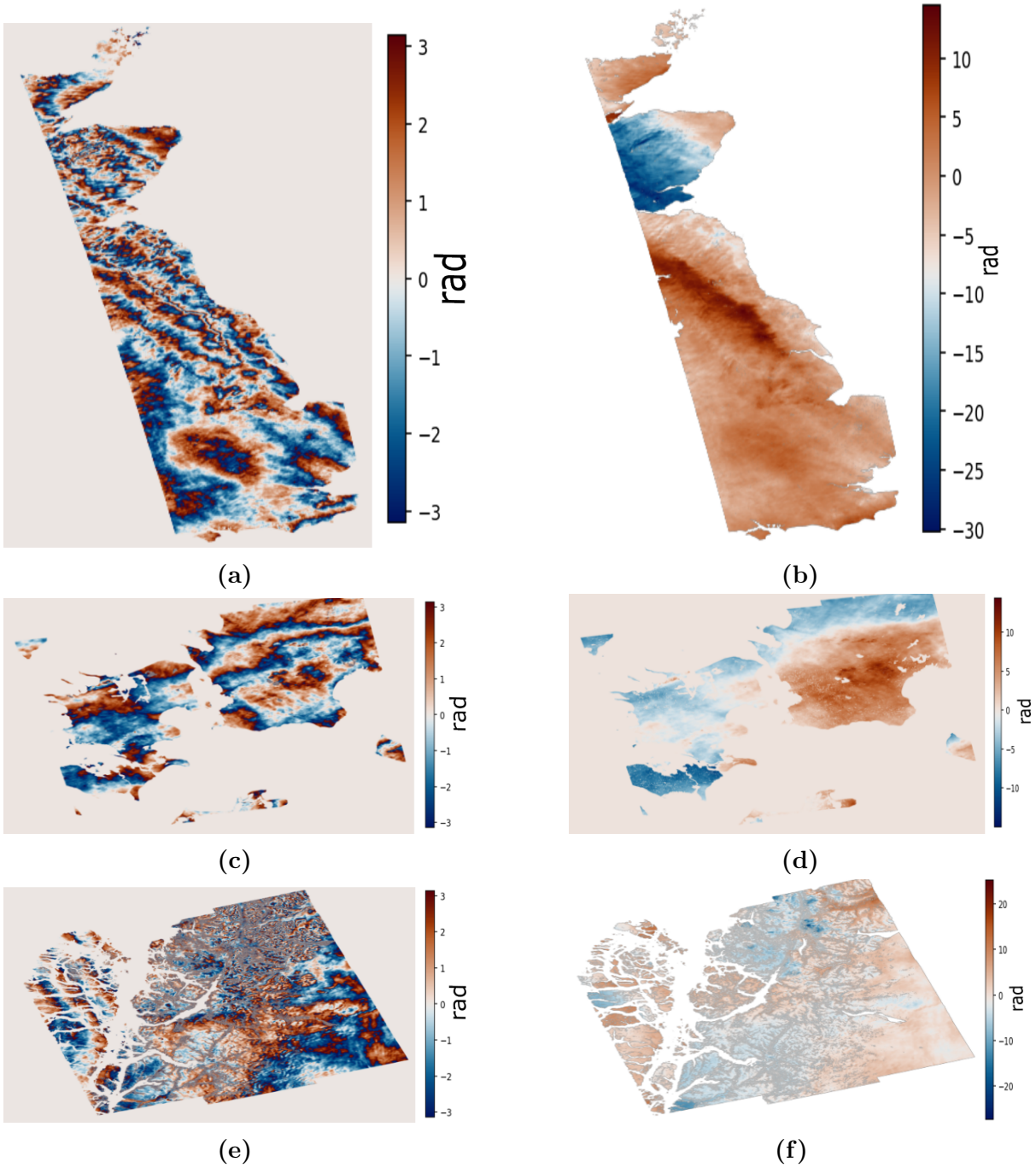


Figure 5.1 – Figure illustrating cases of erroneous phase unwrapping due to isolation of regions. (a, c, e) Wrapped Interferogram, (b, d, f) Unwrapped Interferogram for LiCSAR Frame 132A 03533 515322, 146A 03446 131312 and 062A 13470 131313. (a,b), (c,d) and (e,f) are each paired respectively

as a powerful image-generation tool (Dhariwal et al. 2021). These show an improved generalisability, and an ability to learn fine scale detail, reducing the level of smoothing and improving the overall image quality, when compared to the images generated by GANS and VAE. In DDPMs, a Markov chain progressively adds Gaussian noise to an image over multiple steps until it becomes nearly pure noise. During the reverse process, the DDPM is trained to gradually remove this noise, effectively learning the underlying data distribution of the original image. Diffusion models typically have a U-Net based architecture.

Conditional Diffusion Models build on the DDPM to allow the inclusion of additional information, which is used to guide the model towards generating a specific target with a specific distribution (Bo et al. 2023). In the conditional diffusion model, each step is influenced by the external condition, guiding the model towards a desired target. However there is no theoretical foundation to ensure the output matches the desired conditional distribution (Kim et al. 2025; Bo et al. 2023).

The Brownian Bridge Diffusion Model was proposed to ensure an explicit mathematical linkage between the input distribution and the desired target distribution (Bo et al. 2023), whilst keeping the advantage of the diffusion model when compared to GANs, namely more stable training and reduced smoothing. Bo et al. 2023 proposed to train the BBDM in the latent space rather than the pixel space, using VQGAN to encode the training data to a compressed, lower-dimensional representation of the input. Working within the latent space is advantageous as it has lower computational costs due to lower data dimensionality and also the models tend to be more generalisable. Since the BBDM proposal by Bo et al. 2023, the BBDM has been applied to the SAR domain, translating SAR images to optical images (Kim et al. 2025).

In this Chapter, I present work investigating whether a Brownian Bridge Diffusion Model can be used to inpaint masked regions by generating a realistic signal pattern to connect isolated coherent regions within the interferogram, with the motivation of reducing or removing completely the risk of phase unwrapping errors being introduced due to the

isolation of coherent pixels. I have chosen to trial the Brownian Bridge Diffusion Model to take advantage of the diffusion model attribute to not smooth, as a VAE is liable to do, and being easier to train than a GAN. Another reason the BBDM was chosen, rather than another form of diffusion model, was due to the open-source code being easy to access and adapt to the task of inpainting the wrapped phase images.

5.2 Method

In Bo et al. 2023, the encoder of a pre-trained VQGAN is used to reduce the images to the latent space which was used to train the BBDM; the corresponding decoder component of VQGAN is used to return the features from the latent space to an image. However, InSAR imagery has a very different distribution from the image data used to train the VQGAN. Rather than training a separate model to reduce the SAR images to the latent space, I instead trained the BBDM within the pixel space, which does increase computational costs for training the BBDM but means I am only training one model.

5.2.1 Dataset Preparation

When training deep learning models, the ideal scenario is to train with data which has a very close distribution to the data the model will subsequently be applied to. It is a significant advantage that real Sentinel-1 data can be used to train the model for image-to-image translation for inpainting.

Filtered and unfiltered wrapped interferograms, along with the corresponding coherence files, were sourced from the LiCSAR portal (Lazecký et al. 2020). These interferograms were selected from regions characterised as mainly containing atmospheric signals rather than large scale deformation, with minimal phase noise, ensuring a high quality input for model training.

The LiCSAR portal stores interferograms as large frames which often exceed dimensions of 1000×1000 pixels. Due to computational memory limitations, larger frames were cut into patches of size 64×64. The patch size 64×64 was applied due to Bo et al. (2023) applying the same patch size during training. Bo et al. (2023) found a model trained using patches of size 64×64 resulted in a better performance than if patches of 32×32 or 16×16 were used; it also ensured the patches were sufficiently small to not cause memory issues during training. Further data screening occured over two steps; an automatic step removing patches where the average patch coherence was below 0.3 followed by a manual step to remove remaining patches which contained regions of visibly high noise and *NaN* values. *NaN* values exist due to being outside the frame region or being masked already in the LiCSAR portal. This ensured the only noise present during training was that introduced to train the diffusion model, rather than the model learning to replicate the distribution of the phase noise. The data was normalised to a -1 to 1 range.

Coherence files, matching the areas that the atmospheric training samples were taken from, are available on the LicSAR portal. These were cut into patches of size 64×64, screened to ensure the average coherence of a patch is above 0.3 and manually checked to ensure regions of masked and unmasked areas would be present for each data sample. During training, a mask patch was randomly selected. Masks were prepared by using a 0.3 coherence threshold below which pixels were masked and adapted through the addition of a random number of quadrilateral shapes, each randomly sized and randomly placed. This ensured irregular masking shapes with random placement were represented within the training dataset and that data augmentation occured at for each rotation through the training data set. The training data set is of size 8236.

The validation and testing dataset were prepared using the above described method.

Parameter	Value
Optimiser	Adam
Learning Rate	1e-7
Batch Size	32
Top Model Epoch	732
Training Dataset size	8236
Coherence Threshold	0.3
Patch Size	64
Loss Type	Linear

Table 5.1 – Table of parameters used to train the BBDM.

5.2.2 Model Architecture and Training

The Brownian Bridge Diffusion Model U-Net architecture used by Bo et al. 2023 and subsequently within this work is built heavily on the the OpenAI diffusion models (Nichol et al. 2021). The BBDM model was chosen due to its availability and easy accessibility on GitHub [1]. The final model layer is changed to a Tanh function to ensure the values output would always be within a -1 to 1 range given that wrapped interferogram values are restricted to a $-\pi$ to π range.

During the Denoising Diffusion Probabilistic Model (DDPM) training, Gaussian noise is progressively added to the data sample x_0 in an iterative process. Via a reparametrisation trick, noise can be added from x_0 to x_t in one step:

$$x_t = \sqrt{\hat{\beta}_t} \cdot x_0 + \sqrt{1 - \hat{\beta}_t} \cdot z_t, \tag{5.1}$$

where $z_t \sim \mathcal{N}(0, \mathbf{I})$, $\hat{\beta}_t = \prod_{i=1}^{t} \alpha_i$ where $\alpha_t = 1 - \beta_t$, and β_t is the scheduler determining the rate at which noise is added (Bo et al. 2023). The z_t term is possible as all Gaussian distributions are merged to give a total noise term, and then scaled by the scheduler term.

When a DDPM is trained, at the endpoint of the forward process, the data is almost

[1] Code is available at: https://github.com/xuekt98/BBDM?tab=readme-ov-file (Bo et al. 2023)

Gaussian. In the reverse process, the model is trained to gradually denoise the data to restore the original data distribution in the reverse process.

For the Brownian Bridge Diffusion Model, each x_t is given by

$$x_t = (1 - m_t)x_0 + m_t y + \delta_t noise, \tag{5.2}$$

where y is the target output, x_0 is the input, m_t is $\frac{t}{T}$, t is diffusion step and T is the total number to diffusion steps and $noise$ is the random Gaussian noise generated. δ_t is defined as

$$\delta_t = 1 - ((1 - m_t)^2 + m_t^2) = 2(m_t - m_t^2), \tag{5.3}$$

The endpoint of the forward process is not an almost Gaussian distribution of data as in the DDPM, but instead the target image.

For a small diffusion step t, the primary contribution to noise arises from the randomly generated noise, with the initial influence of x_0 on x_t being greater than x_y. As the diffusion step reaches $\frac{T}{2}$, x_t will have a data distribution most similar to noise, with x_0 and x_y each contributing equally to x_t. As the diffusion step t nears T, the data distribution of x_t most closely aligns with the output data distribution, as x_0 makes increasingly smaller contribution to the x_t. Random Gaussian noise is added to each intermediate x_t to ensure that the intermediate is one possibility for a given time-step, and not just a weighted intermediate between the two distributions. This increases the model robustness and allows better generalisation, ensuring the trained BBDM is not limited to learning just how to get from input to output by a simple interpolation.

The aim of each step in the reverse process is to estimate the mean of the noise (μ_{noise}). This is done by optimising a simplified Evidence Lower Bound (ELBO), which is defined as

$$\mathbb{E}_{x_{\theta,y,\epsilon}}[(1-m_{t-1})\frac{\delta_{t|t-1}}{\delta_t}||m_t(y-x_0)+\sqrt{\delta_t},\epsilon-\epsilon_\theta(x_t,t)||^2] \tag{5.4}$$

where, as defined in Bo et al. 2023, m_t(y - x_0) + $\sqrt{\delta_t}\epsilon$ is the true noise the model tries to approximate and $\epsilon_\theta(x_t, t)$ is the neural networks noise prediction given the current diffusion time-step t and (1-m_{t-1})$\frac{\delta_{t|t-1}}{\delta_t}$ is a weighting term.

The model was trained using an early stopping mechanism for 778 epochs, for 2 days, 1 hour, 57 minutes using a NVIDIA RTX A5000 GI, with a batch size of 32, image size 64x64, Adam optimizer with learning rate 1e-7 and L1 loss type using code adapted from Bo et al. 2023. The top model used for subsequent testing occured at epoch 732.

5.3 Results and Discussion

5.3.1 Testing on a Test Data Set with a Groundtruth

Visual interrogation of the test results, using a test data set which is kept separate during trainings, indicates that qualitatively, the regions inpainted by the model look indistinguishable from the surrounding area. Figure 5.2a-c and Figure 5.2d-f demonstrate cases where the BBDM has created smooth atmospheric patterns and hard fringe edges respectively. The introduction of a realistic location for sharp fringe edges is particularly advantageous to the MC-GI model given that the MC-GI relies upon edge identification. Whilst the BBDM may add an additional area where the wrap count is changed, as in Figure 5.2e and f, the pattern is still realistic and the region does not have any ambiguous smoothing of edges, meaning it is unlikely to cause subsequent issues during unwrapping.

To quantitatively analyse the BBDM performance, before looking at the impact of inpainting on the complete phase unwrapping process, the metrics Mean Squared Error (MSE), Mean Absolute Error (MAE), Local structural Similarity Index Measure (SSIM)

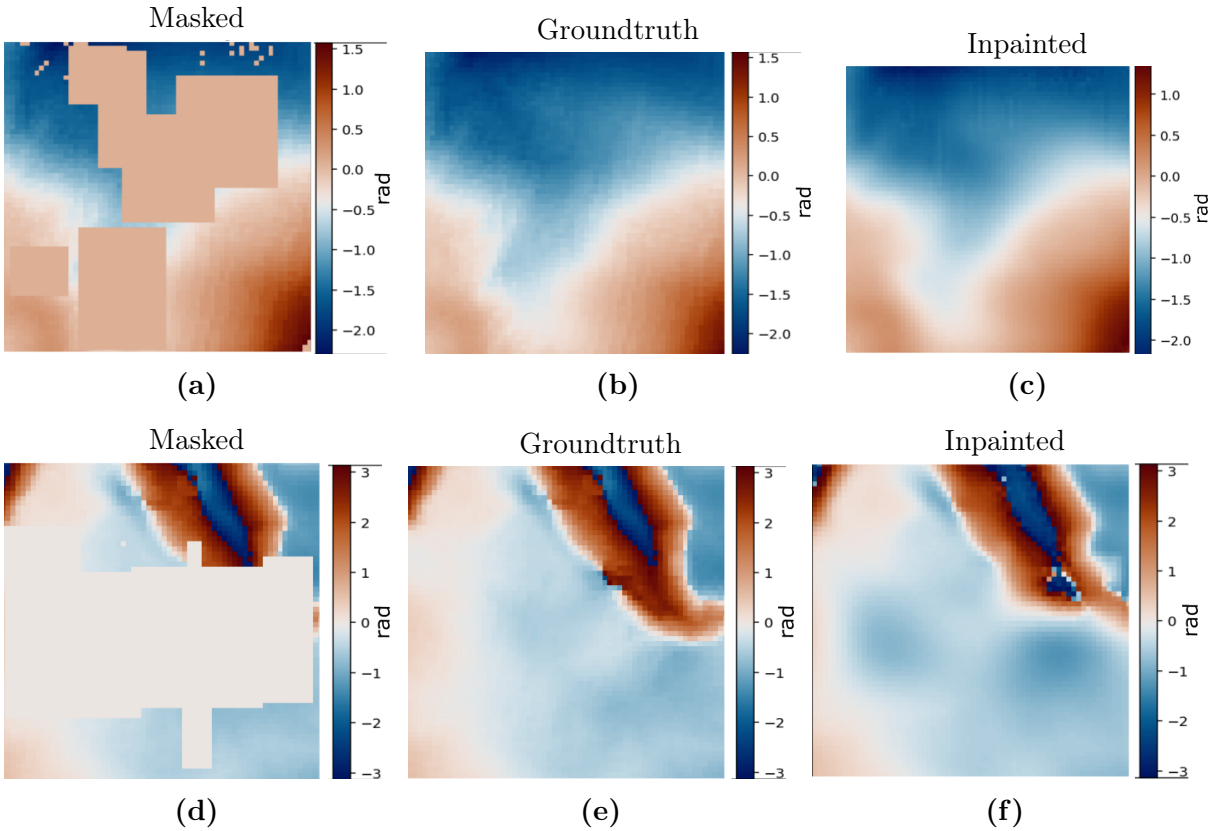


Figure 5.2 – Figure illustrating cases with (a-c) Good atmospheric inpainting case (d-f) Fringe edge inpainting case. (a) Masked Interferogram, (b) Groundtruth Interferogram, (c) Inpainted Interferogram for a, (d) Masked Interferogram, (e) Groundtruth Interferogram, (f) Inpainted Interferogram for d

and Peak Signal-to-Noise Ratio were used to evaluate the inpainting quality in the masked area compared with the original image. Often diffusion models are assessed using the Frechet Inception Distance (FID) and the Incepetion Score, however both rely on an InceptionNet model which has been pre-trained on the ImageNet dataset which has a very different distribution to the InSAR data. Furthermore, the Inception Score is sensitive to noise and small perturbations and whilst the FID better aligns with human judgement, it relies on the assumption of Gaussian distribution of the feature space distributions (Tuel et al. 2023).

Metric Explanations

Peak Signal-to-Noise Ratio: Peak signal-to-noise ratio (PSNR) measures the similarity between the groundtruth data and the generated data; a higher PSNR indicates the generated data distribution is close to the expected groundtruth. PSNR is used because with generative AI, you cannot expect to return the unmasked groundtruth as the model is learning a distribution and thus the returned sample is from the distribution conditioned on the unmasked areas.

$$PSNR = 10log_{10}\frac{Max^2}{MSE}, \tag{5.5}$$

where max^2 is the squared maximum value of the groundtruth interferogram data (π), and MSE is the mean squared error between the groundtruth and predicted interferogram data.

Mean Structured Similarity Index Measure: The Mean Structured Similarity Index Measure (SSIM) is widely used to measure the quality of the generated data compared with the groundtruth, comparing the data structure, comparing the luminance, contrast and structural similarity over a small window and the score is then averaged over the window (Wang et al. 2004; Hwang et al. 2022). Values range from 0 to 1 where 1

means a perfect match between the image and the generated image. The SSIM score was calculated using the Scikit-Image metric (Pedregosa et al. 2011).

$$SSIM_{x,y} = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5.6a}$$

$$MSSIM(X, Y) = \frac{1}{M}\sum_{j=1}^{M} SSIM_j \tag{5.6b}$$

where μ is the mean value, σ is the standard deviation, σ^2 is the variance, σ_{xy} is the covariance for the groundtruth (x) and generated image (y), C_1 and C_2 are constants that stop the denominator becoming 0, M is the number of windows and j refers to the index of the window (Hwang et al. 2022).

Mean Squared Error	**Local SSIM**	**Mean Absolute Error**	**Peak Signal to Noise Ratio**
0.50	0.79	0.18	69.08

Table 5.2 – Mean values of Mean Squared Error, Local SSIM, Mean Absolute Error, and Peak Signal to Noise Ratio over the testing dataset.

The distribution of MSE and MAE across the data set shown in Figure 5.3 suggests that the BBDM is able to closely approximate the expected data distribution in the gap-filled regions, aligning well with the ground truth. The majority of local SSIM values were above 0.79 and over 50% of inpainted regions have a SSIM above 0.8 (as shown in Figure 5.3b and Table 5.2, which suggests a close similarity in the data distribution of the inpainted region and the groundtruth for the inpainted region.

In each subplot of Figure 5.3, the maximum and minimum measurements (the lines at top and bottom of each plot respectively), the median and the mean are shown. Even for examples where there is a low PSNR values, such as those shown in Figure 5.4a - d, the low PSNR score of 23.6 was due to the fringe edge not being located in the expected place. However, the fringe edge was still placed in a reasonable location that perceptually looks

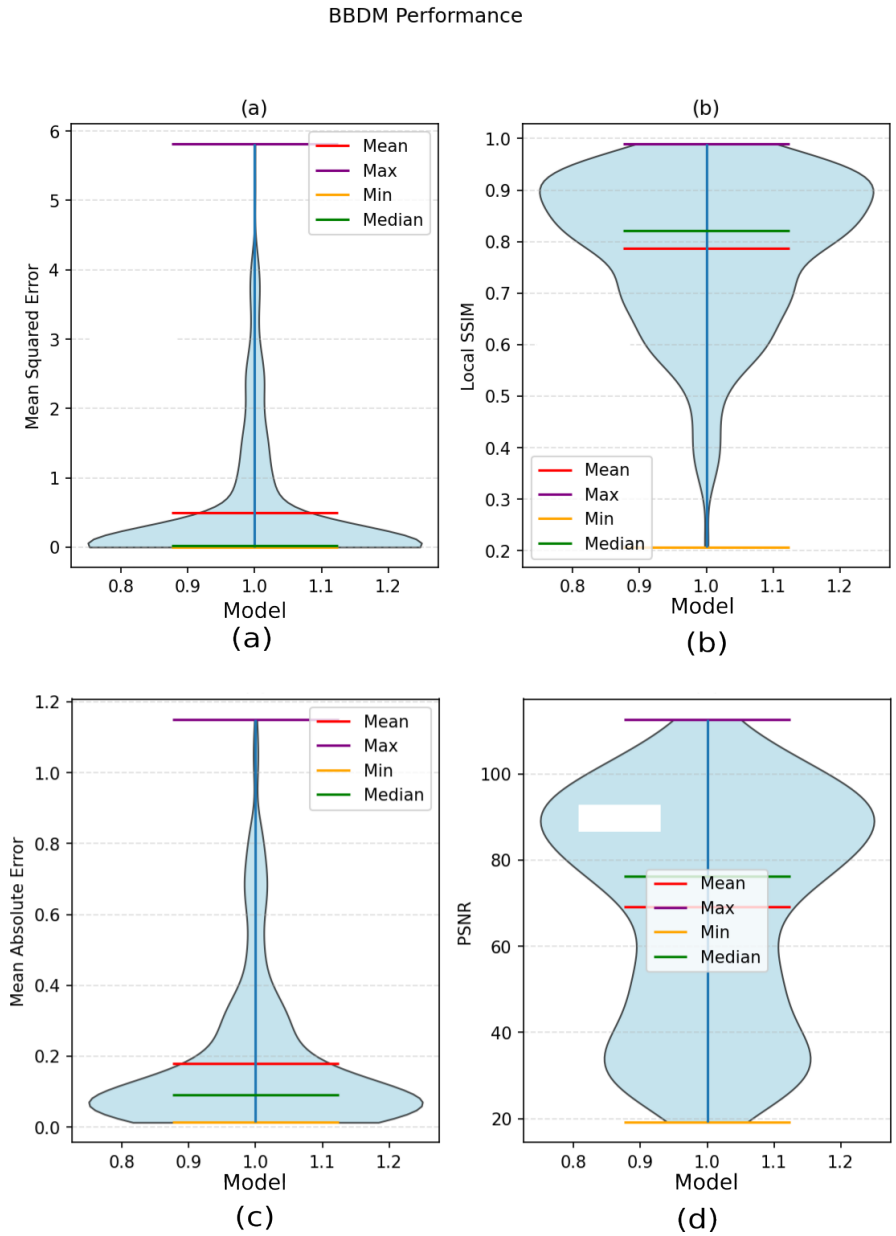


Figure 5.3 – Figure displaying violin plots of (a) Mean Squared Error, (b) Local SSIM, (c) Mean Absolute Error, (d) Peak Signal to Noise Ratio.

reasonable to a human. For Figure 5.4e-h, where the lowest SSIM value of 0.22 occured, this was due to the BBDM model introducing a gradual gradient in the atmosphere. However, from Figure 5.4g, the atmospheric pattern looks reasonable at the locations where there was the largest difference in absolute phase value was marked in Figure 5.4h.

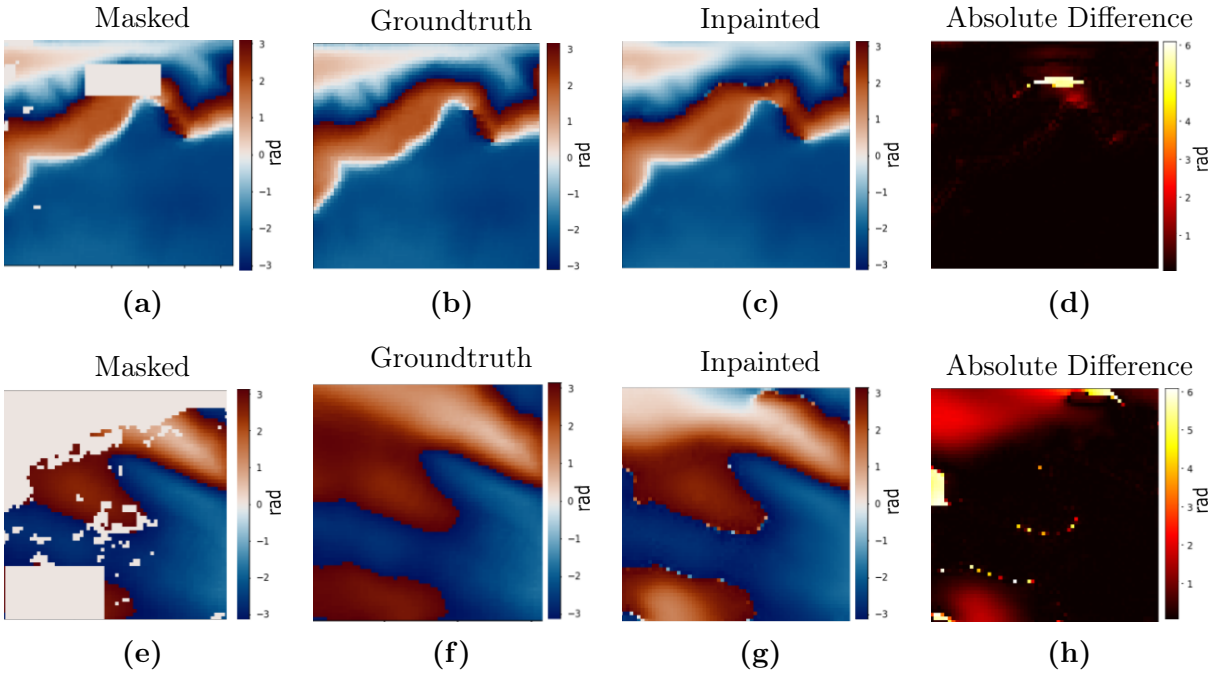


Figure 5.4 – Figure illustrating cases with (a-d) low PSNR score (23.6) and (e-h) low SSIM score (0.22). (a) Masked Interferogram, (b) Groundtruth Interferogram, (c) Inpainted Interferogram, (d) Absolute Phase Difference between b and c, (e) Masked Interferogram, (f) Groundtruth Interferogram, (g) Absolute Phase Difference between f and g.

5.3.2 Pixel-Wise Analysis of the Confidence in BBDM Prediction

Further assessment on a pixel-wise basis was performed using a Monte-Carlo approach. For each masked interferogram, the BBDM was run iteratively to give 10 difference possible inpainted results. The variance of the masked region was used to indicate the confidence of the model in the inpainted region; a high variance indicates high variability in the inpainted result and suggests low model confidence in the output, and vice versa for low variance. The testing was performed on an unseen dataset, where each interferogram contained atmospheric and synthetically generated deformation signals (size 256×256) to mimic the range of interferograms the BBDM will be applied to.

Over atmospheric regions, the variance is low, generally less than 0.1 indicating high confidence in the inpainting. The lowest certainty in the BBDM prediction is associated with where the fringe edges should be located; the further from the fringe edge, the higher the certainty in the model prediction.

5.3.3 Gap-filling larger interferograms

Due to memory capacity limitations, larger interferograms which exceed a dimension of 256×256 need to be patched, which complicates the gap-filling process, and means that there may be issues with an entire area being masked and containing few to no coherent signals. Offsetting is required to ensure there is a smooth transition between connected patches. Within this section, there is no groundtruth for the interferograms as I have chosen interferograms with regions of low coherence in order to mask the interferograms based upon coherence, rather than adding random shapes which replicates the scenarios where this method would be used to improve phase unwrapping. The results are analysed qualitatively.

The BBDM does work to inpaint interferogram patches where there is sufficient coherent signal present for the model to understand the general patterns and where the signals

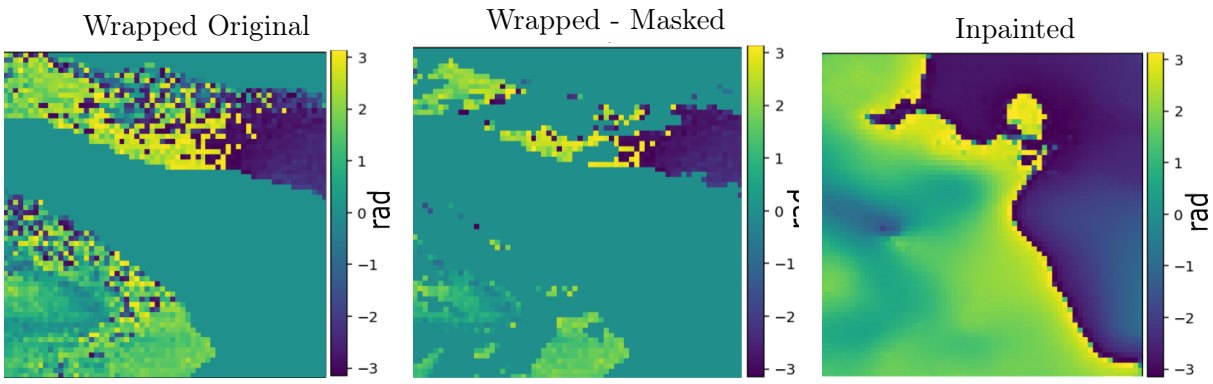


Figure 5.5 – An example illustrating where good inpainting has occurred. This is likely due to the similarity of the input to the training data. (a) Wrapped Interferogram, (b) Masked Interferogram, (c) Inpainted Interferogram. The coherence mask has a threshold of 0.3.

present are similar to those present within the training dataset as demonstrated. When patching a larger interferogram, the amount of masking will vary, which in turn influences the quality of the inpainted results for each file. It was found via experimentation that better performance is achieved both by increasing the number of pixels available and ensuring the available pixels are in large groupings. This is shown by comparing Figure 5.5 to Figure 5.6, which are both patches from the same interferogram; the more information available within the patch for the BBDM to base the inpainting on, the more accurate and realistic the inpainting is. In Figure 5.5 which has more available coherent pixels in the patch, and were the pixels are grouped together, the inpainting has been more successful than Figure 5.6, where fewer coherent pixels are available, and these pixels are located sparsely.

In Figure 5.6, where over 90% of the interferogram is masked, the inpainting has failed, by introducing unrealistic, incomplete fringes. Such fringes are likely caused by the BBDM assuming the variation between the small groupings of coherent pixels are caused due to each region having a different wrap count rather than the more likely scenario where each coherent region is within one wrap count phase cycle. Generally, an increase in patch size, or lower coherence threshold may allow more information to be present to inform the BBDM. However, a lower coherence threshold is unlikely to have improved the result

given the level of phase noise present meaning there is minimal useful signal for the BBDM to base its inpainting on. There is also no guarantee that the wider region around the tile would have sufficient coherent pixels to counteract the level of masking within the patch, and there is a limit as to the useable patch size due to memory limitations. The presence of incomplete fringes in the prediction will lead to downstream phase unwrapping issues, similar to those identified to be caused by filtering in Figure 3.1 for the Greedy Phase Integrator algorithm.

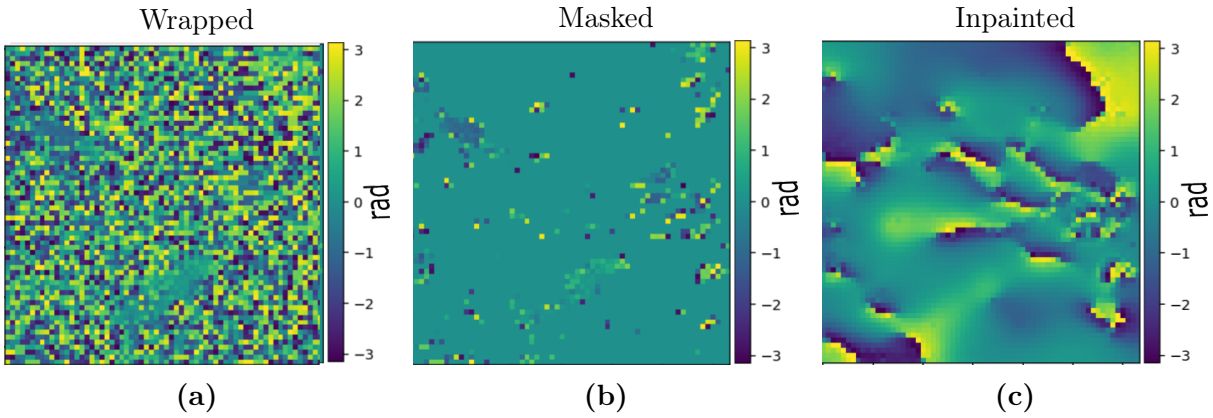
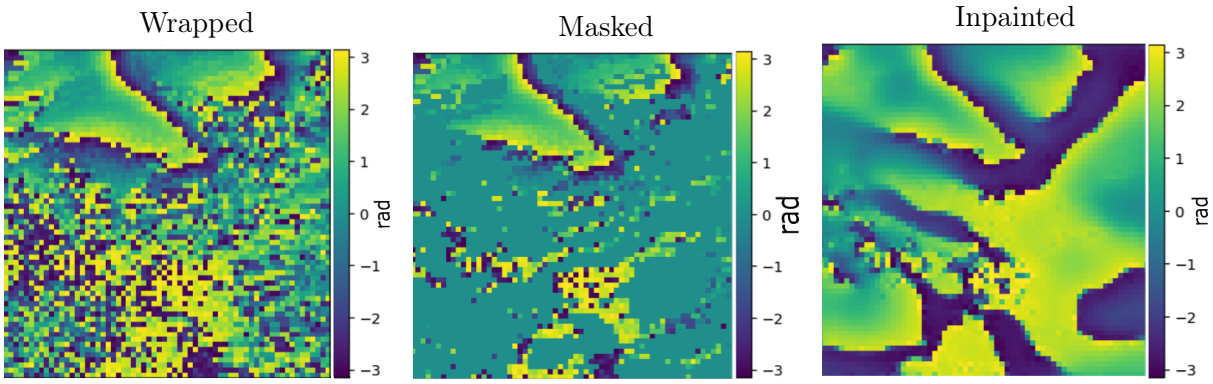


Figure 5.6 – Figure illustrating case where masking has led to poor inpainting (a) Wrapped Interferogram, (b) Masked Interferogram, (c) Inpainted Interferogram. The patch is masked using a coherence threshold of 0.3.

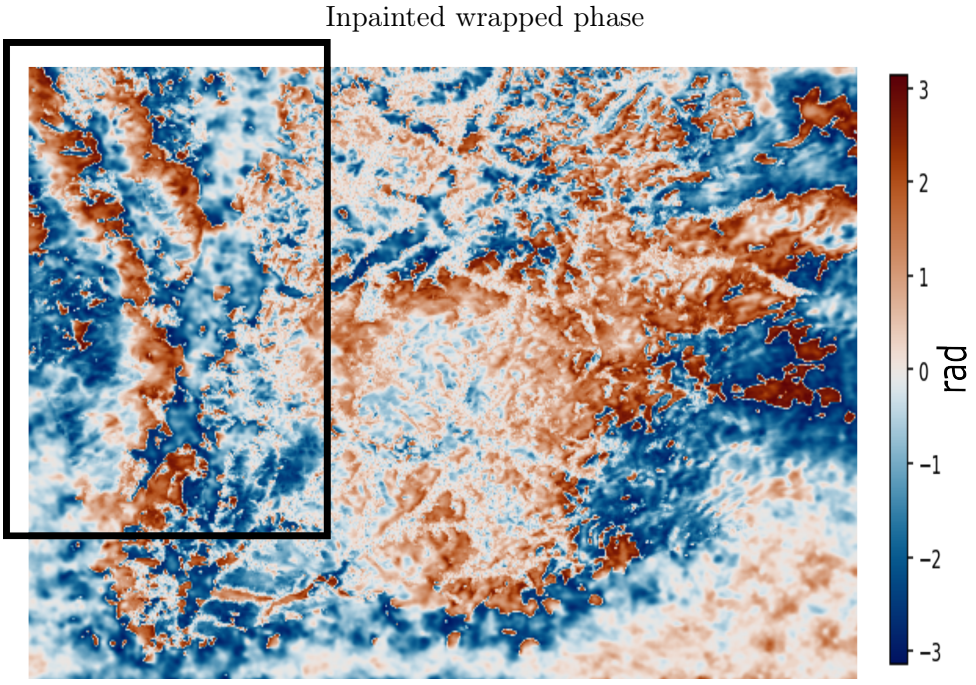
The BBDM does have a tendency to generate incomplete fringe edges, particularly when there are multiple fringes within a tile as in Figure 5.7. This is likely due to the training data set purposefully consisting of mainly large scale atmospheric signals, where the number of fringes within each patch was generally restricted to 1 or 2 edges. This was for two reasons; this study is to investigate whether inpainting could work to reduce phase unwrapping error and thus I chose to have the simplest scenarios and then increase the complexity should the approach work. The second reason is often there is more likely to be noisy signals around fringe edges associated with deformation, and given that in the initial study, I wanted to restrict noise introduction to only be the training introduced Gaussian noise, and not phase noise, I therefore restricted the signal types in the training data set.

To minimise the presence of patch edges impacting downstream phase unwrapping, over-



(a) **(b)** **(c)**

Figure 5.7 – Figure illustrating a case where poor inpainting has occured likely due to the dissimilarity between the test data and the training data (a) Wrapped Interferogram, (b) Masked Interferogram, (c) Inpainted Interferogram.



(a)

Figure 5.8 – Figure demonstrating the impact of tiling size on inpainted output for interferogram shown in Figure 5.9c. Tile size 256x256. Overlap = 128, via iterative left-right first, then right to left, always from the top to the bottom of the interferogram. LiCSAR Frame: 062A 13470 13131, dated 2022/03/25 - 2022/04/06.

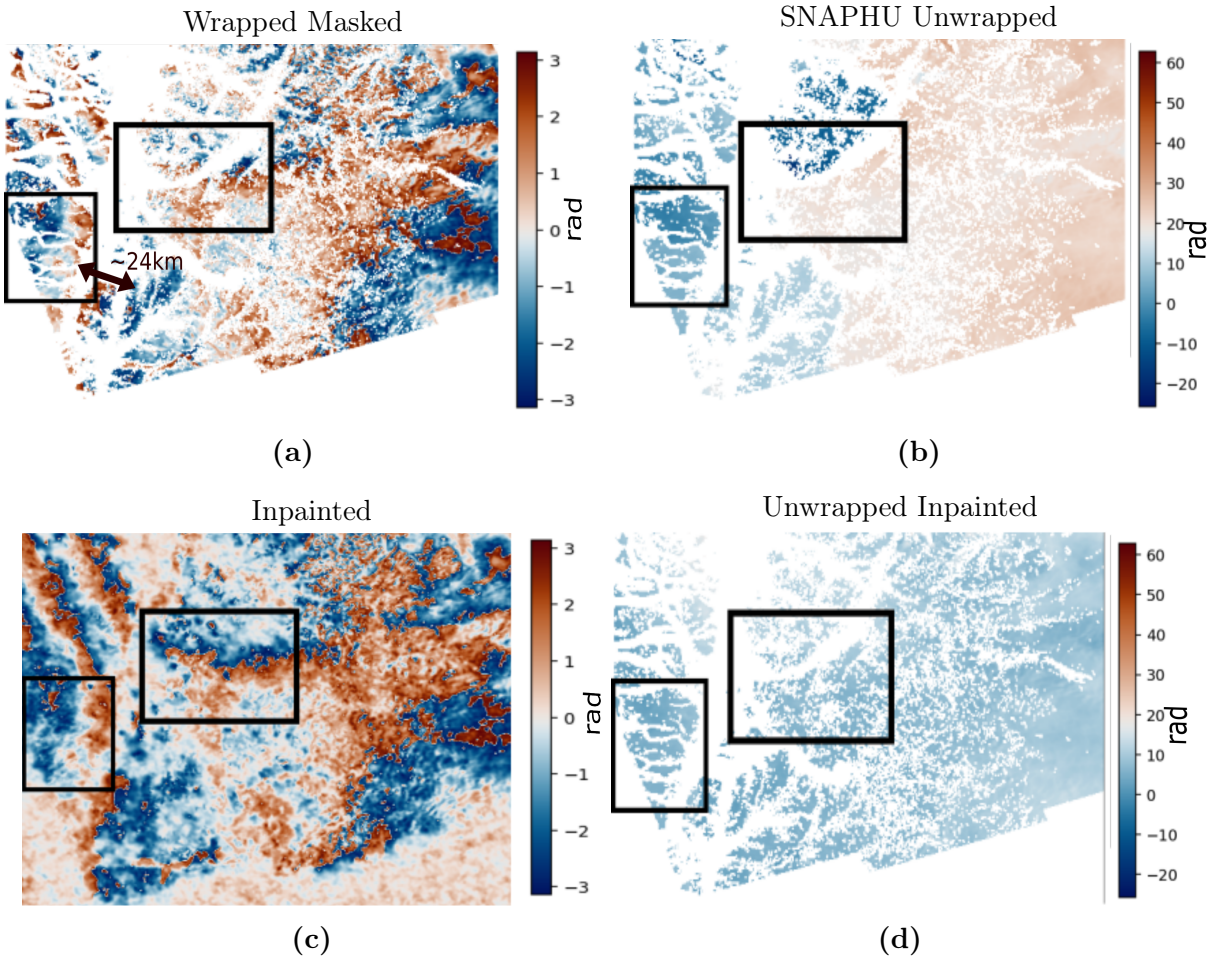


Figure 5.9 – Figure illustrating the application of BBDM to a wrapped interferogram where phase unwrapping errors have occured as a result of the isolation of coherent regions where SNAPHU has been used for unwrapping. LiCSAR Frame: 062A 13470 131313, dated 2022/03/25 - 2022/04/06 which has been patched to patch size 256x256 during unwrapping, coherence threshold:0.1, blow which inpainting occured, where (a) is the wrapped masked interferogram, (b) Unwrapped interferogram for wrapped interferogram (a) using a mask where threshold was set to 0.01 resulting in regions remaining separated, (c) inpainted wrapped phase version of (a), (d) Unwrapped phase of wrapped interferogram - no mask applied during unwrapping (c). The boxed regions highlight regions where the phase unwrapping has been impacted by isolation of coherent pixels and led to an improved phase unwrapped result. The mask for unwrapping is set to 0.01 to provide a better comparison between the unwrapped result for both the inpainted and not inpainted unwrapped result, otherwise too much of the interferogram remains masked for a reasonable comparison.

lapping patching was trialled. The interferogram was tiled with half overlapping regions in an iterative process. The inpainting was performed for large interferograms for patches generated from left to right first, with half overlapping tiles. Via experimentation, it was found that broadly patches were required to contain at least 25% coherent pixels to return a reasonable inpainted result. This was required to be in larger groupings, similar to those shown in Figure 5.7, rather than as isolated, smaller groupings. Information was included from previously inpainted tiles by setting a user defined overlapping area for the patch along both the top and left edges, thus meaning the previous gap filling was used alongside any coherent pixel information for the inpainting task. A second iteration of inpainting is then performed from right to left, in a similar process to inpaint any remaining masked regions to fill any remaining gaps. As can be seen in Figure 5.8, this ensures there are no artifacts left by patch edges.

Figure 5.9b shows that unwrapping errors have occured in the boxed regions due to the regions of coherent pixels being isolated. The BBDM achieves a realistic atmospheric pattern within the boxed area (Figure 5.9c), with placement of fringe edges where required. This means the isolated regions are connected across multiple patches making the inpainted regions indistinguishable from the actual signal, which supports subsequent phase unwrapping and reduces phase unwrapping errors.

In the boxed region of Figure 5.10b, the BBDM is shown to struggle to create the wrapped phase patterns associated with earthquakes, deforming volcanoes or mining. This is likely due to the training data set generally containing examples of wrapped phase with smooth transitions. Where fringe edges are present in the data set, these tend to be long fringes often associated with atmospheric signals rather than from large scale deformation events, meaning the rapidly changing fringe edges were not seen. Nevertheless, as can be seen in Figure 5.10d, the imperfect inpainting in the boxed region of Figure 5.10b has had no adverse affects on the unwrapped result, and the offset between the smaller island and larger island has been corrected for. Hence, whilst the BBDM can be applied to examples such as those shown in Figure 5.10, the interferograms should be checked before

2477 subsequent unwrapping to ensure the placement of edges won't inhibit the unwrapping
2478 process.

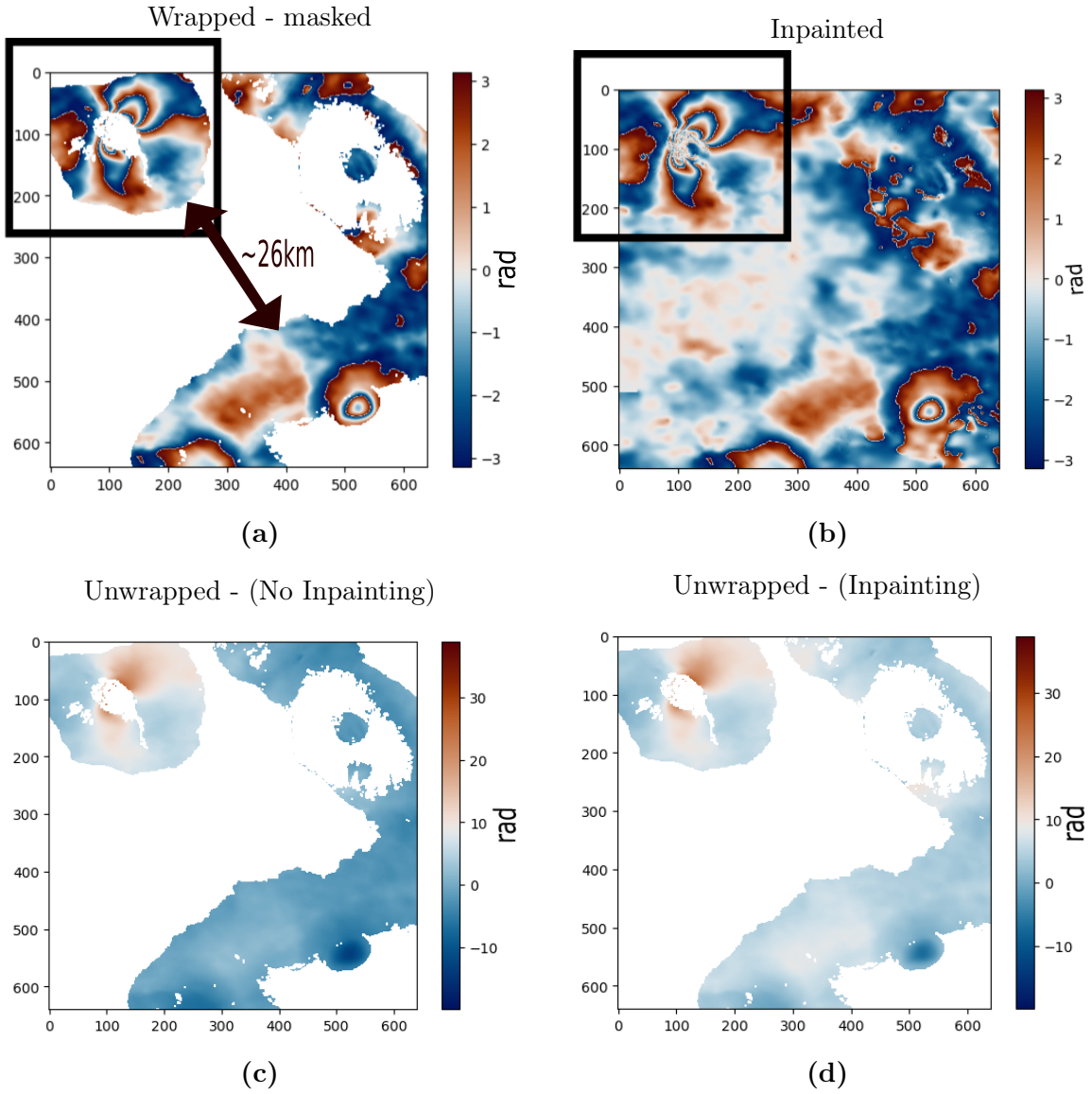


Figure 5.10 – Figure demonstrating that the BBDM can complete circular rings, however struggles where there are repeated fringe patterns often associated with deformation events. LicSAR Frame: 128D 09016 110500, date of acquisitions: 2024/02/24 and 2024/03/19. SNAPHU has been used to unwrap. Coherence set to 0.1, DefoMax set to 1.2, patch size 256x256, overlap size 128x128. (a) Masked Interferogram, (b) Inpainted Interferogram, (c) Unwrapped Interferogram of (a), (d) Unwrapped Interferogram of (b)

5.3.4 The Impact of Gap Filling on Phase Unwrapping

Given that it is known SNAPHU's unwrapping results are impacted by the isolation of coherent pixel regions, and to ensure any difference in the unwrapped results is due to the gap inpainting rather than the unwrapping technique, the technique was tested upon both SNAPHU and the MC-GI technique.

Figure 5.11 demonstrates a case where the use of gap filling has successfully led to a reduction in the loop closure error when the main error source was due to the offset between the unwrapped results of the two coherent and unconnected regions in Figure 5.11a from 12.3% to 0.2%. In this case, SNAPHU is used to unwrap both sets of interferograms. However, the inpainting connecting the two regions looks increasingly unrealistic as the gap size increases towards the top of the interferogram compared to the bottom. This is likely due the lower likelihood of each patch containing true signals as the gap size increases and the model is reliant upon propagating inpainted results. Given that in the test data set, most patches contained wrapped phase which varied minimally across the patch, it is unsurprising the inpainting has replicated such patterns.

To reduce the possibility that a patch contains a low proportion of signal, downsampling of the data was trialled. In the case shown in Figure 5.12, a downsampler from Scikit-Image was used to reduce the interferogram so that each dimension was halved; because of the dimensionality reduction, the tile size was reduced to 128x128 from 256x256 and the overlap was reduced to 64 from 128. As can be seen, the inpainting is more realistic, but the downsampling can impact on the final unwrapped result.

Figure 5.13 further demonstrates that inpainting can effectively reduce the loop closure error, decreasing, for this example, from 22.59% to 6.56%. However, as seen in Figure 5.11, the effectiveness of this technique appears to be sensitive to the gap size to fill. In cases where the isolated coherent regions are sufficiently distant, such that multiple patches have no coherent pixels as additional information, the transition between these patches can become unrealistic. This exists until sufficient coherent information is rein-

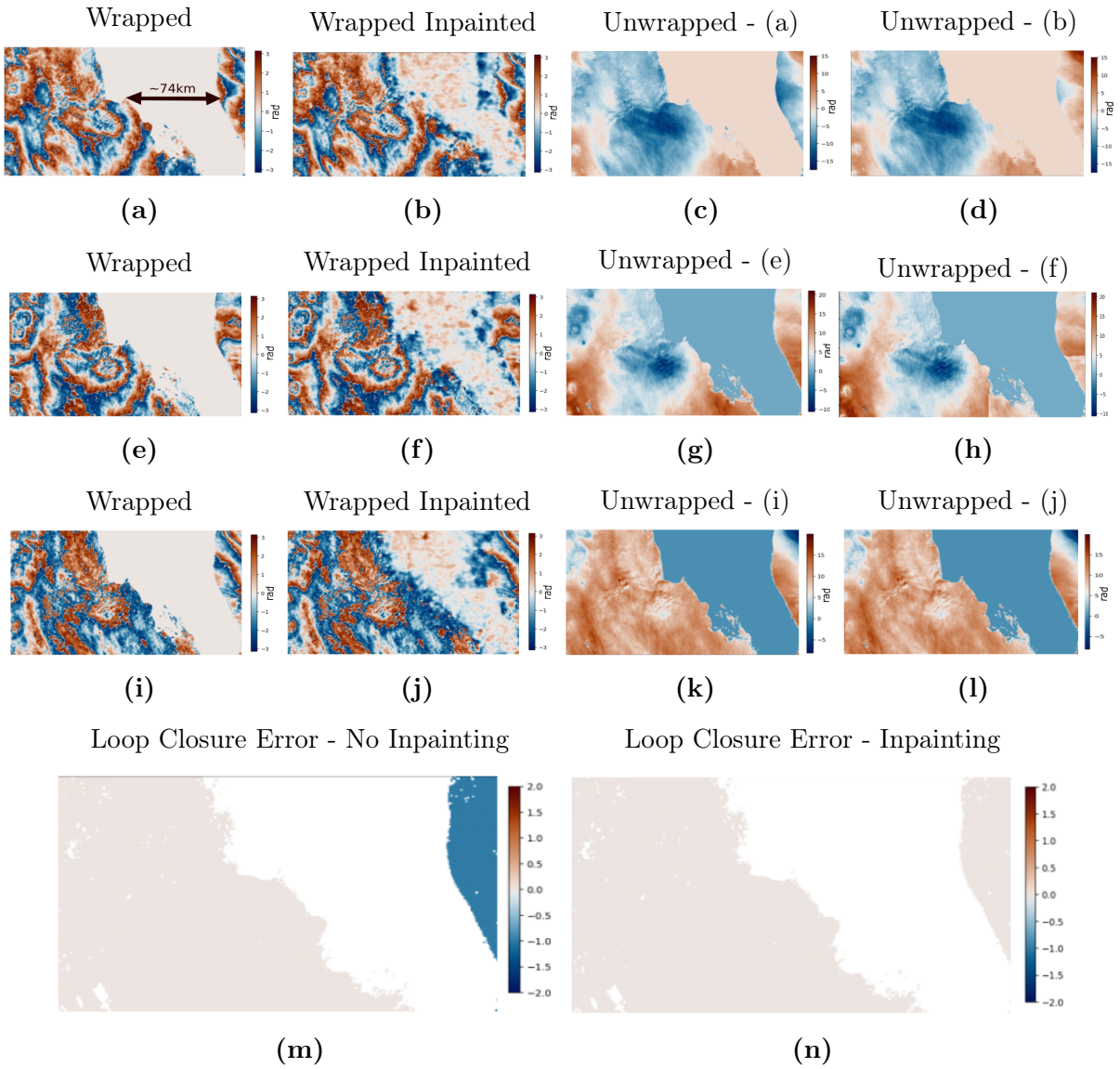


Figure 5.11 – Figure showing how inpainting has led to a reduced loop closure error due to improved unwrapping between two isolated coherent regions for the LicSAR Frame 006D 07728 131313. SNAPHU is used to unwrap. Inpainting was performed on tiles of size 256x256, with an overlap of 128x128 and coherence threshold of 0.1. (a) Wrapped Interferogram from dates 2022/11/29 and 2022/12/11, (b) Inpainted version of (a), (c) Unwrapped Interferogram of (a), (d) Unwrapped Interferogram of (b), (e) Wrapped Interferogram from dates 2022/11/29−2022/12/23, (f) Inpainted version of (e), (g) Unwrapped Interferogram of (e), (h) Unwrapped Interferogram of (f), (i) Wrapped Interferogram from dates 2022/12/11 − 2022/12/23, (j) Inpainted version of (i), (k) Unwrapped Interferogram of (i), (l) Unwrapped Interferogram of (j). (m) Loop Closure Error when no inpainting performed, (n) Loop Closure Error for inpainted results. For these cases, DefoMax is 1.2 and coherence masks are used, with coherences of 0 set to $1e-6$. Wrapped phase are in radians. Unwrapped phase is in rad.

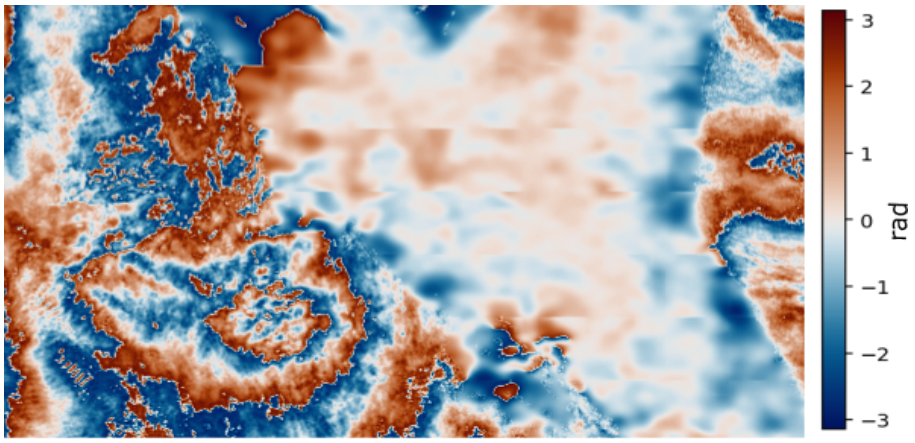


Figure 5.12 – Figure showing the inpainted result when the wrapped phase has been downsampled using mean averaging over every 2 pixels, which reduces the height and width dimensions by half. This is the inpainted, downsampled version of Figure 5.11f. There is more realistic inpainting in the large gap region, particularly towards the bottom of the wrapped interferogram.

troduced, as seen in Figure 5.13b. The reliance on sufficient wrapped phase information to gap fill is unsurprising when you consider that this is the information the BBDM uses to inform what the distribution in data should be, and the success of the inpainting impacts the phase unwrapping in turn. Between the interferograms the loop closure error has increased. As shown in Figure 5.13 to Figure 5.14, the coherence mask has been changed from 0.01 to 0.1 respectively, which reduces the information available for the BBDM.

The BBDM tends to predict smooth edges, rather than implementing hard fringe edges. This lack of hard edges negatively impacts unwrapping using MC[4]-GI, as shown in Figure 5.15. The incorrect unwrapping, particularly for the larger island in Figure 5.15b and d is likely directly related to the smooth edge seen in Figure 5.15a and c respectively. Whilst it is shown the BBDM can implement hard edges for changes in wrap count, such as at the top of the large island of Figure 5.15e, until the BBDM's ability to implement fringe edges but not hallucinate disappearing fringe edges improves, the phase unwrapping of the MC-GI models will still have poor accuracy, given its reliance on complete edges. However, when combined with SNAPHU, the BBDM reduces the occurrence of phase unwrapping errors.

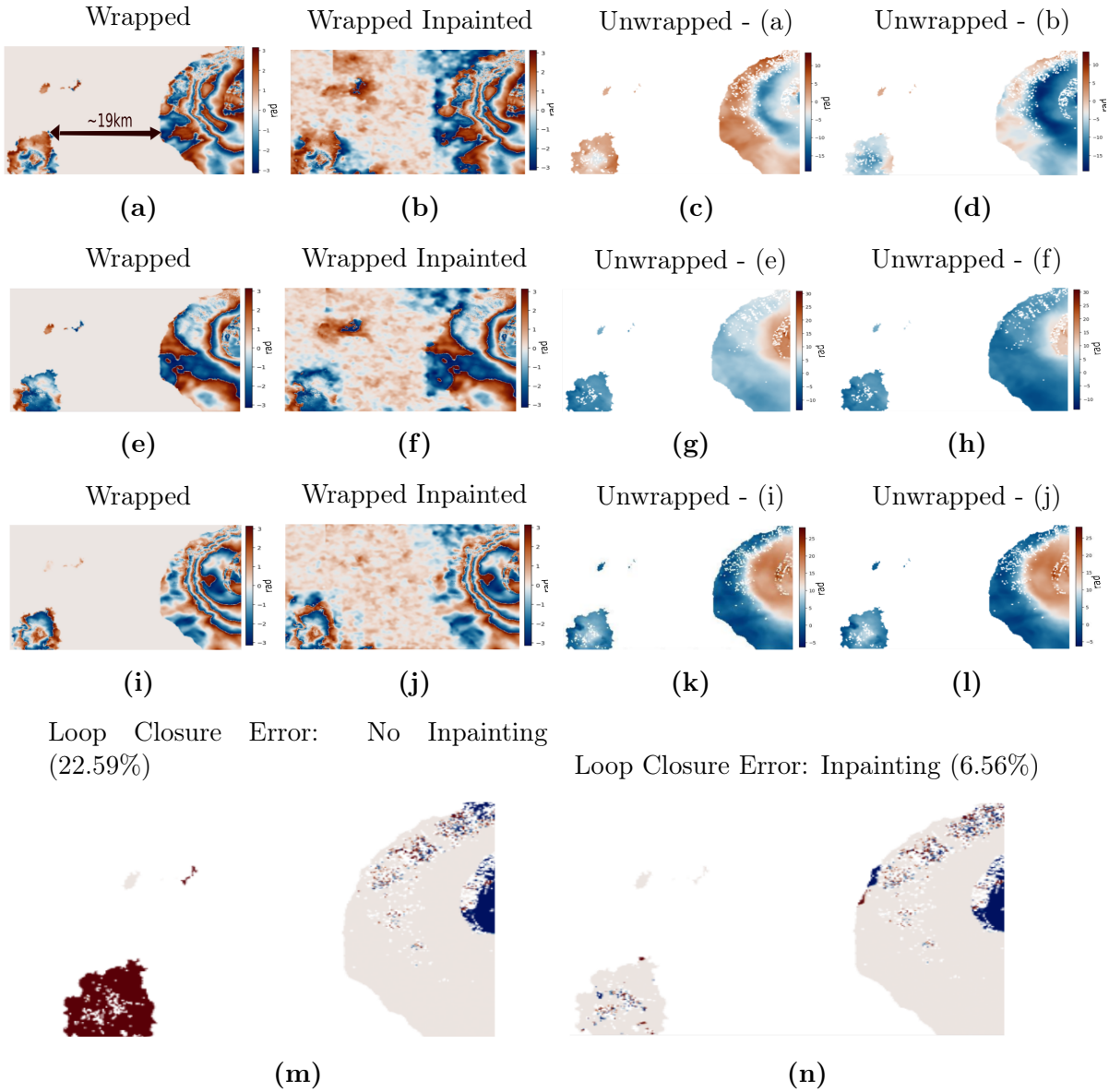


Figure 5.13 – Figure showing that inpainting has reduced loop closure error for LiCSAR Frame: 140D SM FGbr s4 test. Patch size: 256x256, Overlap: 128x128, Coherence threshold for inpainting mask: 0.01 (a) Wrapped Interferogram from dates 2015/06/12 and 2015/09/16, (b) Inpainted version of (a), (c) Unwrapped Interferogram of Wrapped Interferogram (a), (d) Unwrapped Interferogram of Wrapped Interferogram (b), (e) Wrapped Interferogram from dates 2015/06/12 – 2016/03/023, (f) Inpainted version of (e), (g) Unwrapped Interferogram of Wrapped Interferogram (e), (h) Unwrapped Interferogram of Wrapped Interferogram (f), (i) Wrapped Interferogram from dates 2015/09/16–2016/03/02, (j) Inpainted version of (i), (k) Unwrapped Interferogram of Wrapped Interferogram (i), (l) Unwrapped Interferogram of Wrapped Interferogram (j). No masking applied when unwrapping the inpainted result using SNAPHU. Mask applied for unwrapping masked interferograms using SNAPHU, coherence threshold: 0.01, DefoMax: 1.2. For the inpainted interferogram, the coherence map is adjusted so that pixels below 0.1 are set to 0.1.

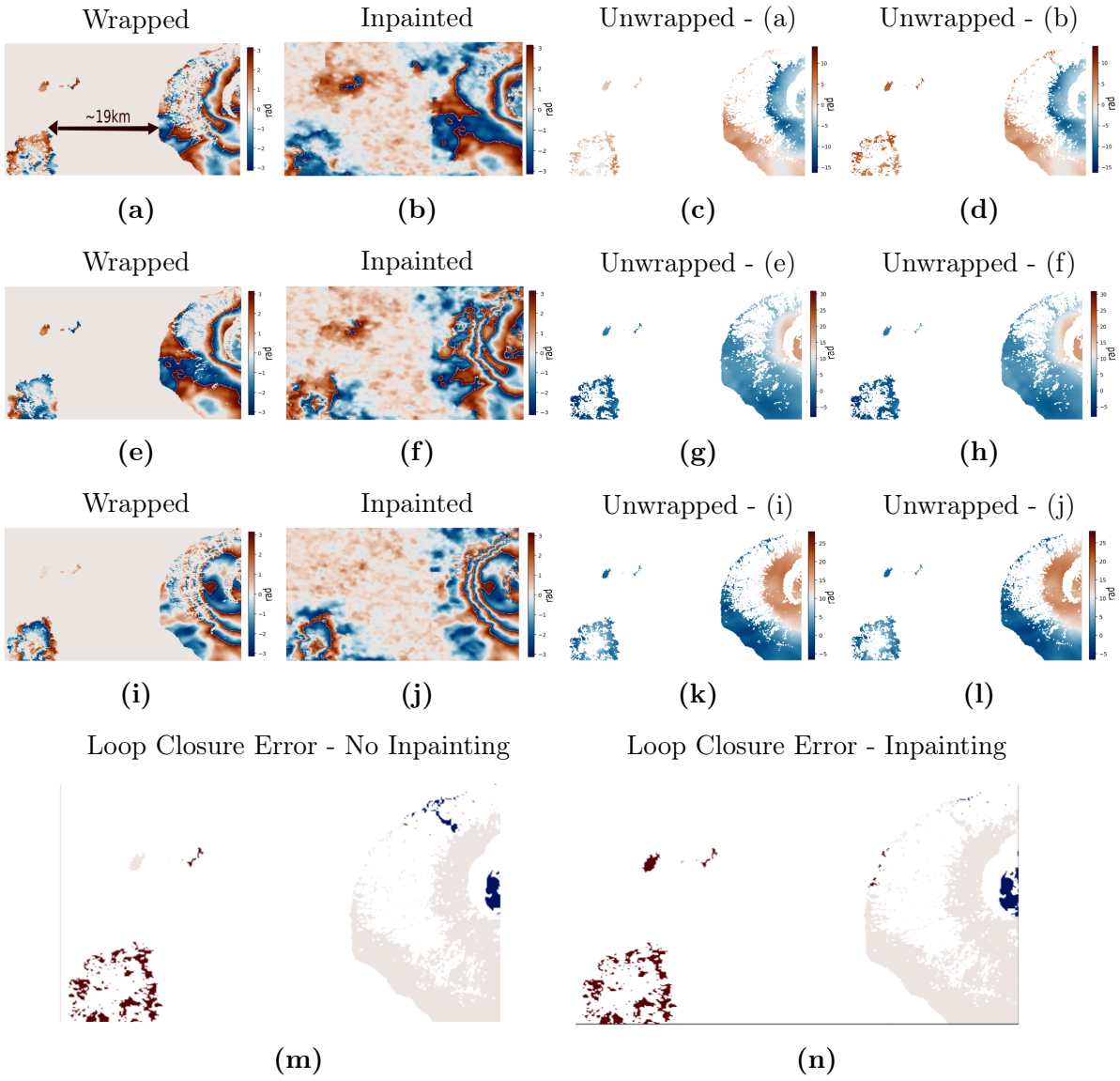


Figure 5.14 – Figure showing gap filling successfully reducing the loop closure error caused by two isolated coherent regions for the LicSAR Frame 140D SM FGbr s4 test (tile size: 256x256, overlap: 128x128, coherence threshold: 0.1) (a) Wrapped Interferogram (2015/06/12 - 2015/09/16), (b) Inpainted (a), (c) Unwrapped Interferogram (a), (d) Unwrapped Interferogram (b), (e) Wrapped Interferogram (2015/06/12 – 2016/03/023), (f) Inpainted (e), (g) Unwrapped Interferogram (e), (h) Unwrapped Interferogram (f), (i) Wrapped Interferogram from dates 2015/09/16 – 2016/03/02, (j) Inpainted (i), (k) Unwrapped Interferogram (i), (l) Unwrapped Interferogram (j). (m) Loop Closure Error when not inpainted(21.81%), (n) Loop Closure Error when inpainted (21.2%). SNAPHU settings: DefoMax - 1.2. For the inpainted region, the coherence is set as 0.1. A mask is applied when unwrapping the non-inpainted wrapped phase with a coherence threshold of 0.1.

Comparison to nearest neighbour

An alternate, non deep learning approach to gap fill is using nearest neighbour, where the 'gap' is predicted via interpolation from the nearest coherent point. The BBDM approach was compared to the nearest neighbour interpolation to explore whether an improved unwrapped result can be achieved in scenarios where two or more coherent regions are isolated.

A test dataset containing 46 synthetic interferograms of size $256 \times 256 \times 1$ was used; each interferogram contained at least two isolated regions with a large noise area. Whilst the training dataset included deformation event, I purposefully chose to minimise the number of cases where the deformation event had been partially masked at the edge of a coherent region. This is because the BBDM has been trained primarily on atmospheric signals and as discussed previously has not "seen" many examples of deformation events. If I were to have included such examples in the test dataset, the unwrapping may have been unfairly penalised for the BBDM in scenarios where it was required to inpaint a deformation event. Once the BBDM has been trained on a wider range of scenarios, this experiment should be repeated to include such cases to understand the full extent of the BBDM compared with the nearest neighbour approach.

I chose to test using synthetic interferograms rather than real interferograms as it meant I had a known, unwrapped groundtruth to compare the unwrapped results to. An additional advantage is that I could control the level of stochastic noise present, thus ensuring I was testing only the impact of the gap filling technique of the unwrapped result, rather than needing to consider whether the nearest neighbour or BBDM have increased sensitivity to stochastic noise. To generate the synthetic interferograms, SyInterferopy was used (Gaddes et al. 2019). Generation of synthetic interferograms using SyInterferopy is discussed in previous Chapters, and I refer the reader to these for additional information.

As can be seen in FIGURE, the BBDM and nearest neighbour achieve broadly comparable results. Both approaches successfully unwrap 41 interferograms, avoiding unwrapping

errors associated with the isolation of coherent regions. However, the BBDM approach incorrectly unwraps 2 interferograms, compared with 5 incorrectly unwrapped interferograms produced by the nearest neighbour approach. An example where the BBDM approach has performed better than the nearest neighbour approach is shown in Figure 5.17.

In Figure 5.18, the most likely source of error appears to be a fringe introduced by the inpainting between the two isolated regions. In the second, incorrectly unwrapped interferogram (Figure 5.19), the deformation signal is situated at the boundary of one of the isolated regions. Although the test dataset contains examples in which partially masked deformation events are successfully unwrapped (Figure 5.20), these cases generally involved only a tiny portion the deformation being masked. This highlights why efforts were made to minimise the inclusion of such cases within the test dataset, thereby avoiding penalisation of the BBDM's poor performance when inpainting deformation events during gap filling. However, further research is required to improve the model's capability in handling such scenarios.

Overall, the application of the BBDM to gap fill, rather than using nearest neighbour, has more consistently reduced the potential risk of the incorrect unwrapping of an interferogram due to the isolation of coherent regions. An additional benefit of the BBDM approach is that the wrapped phase looks more realistic when inpainted than when applying the nearest neighbour, which will improve visualisation of the wrapped phase to end-users. However, further work is required to expand the scenarios to include deformation events which are partially masked, which will increase the generalisability of the BBDM model.

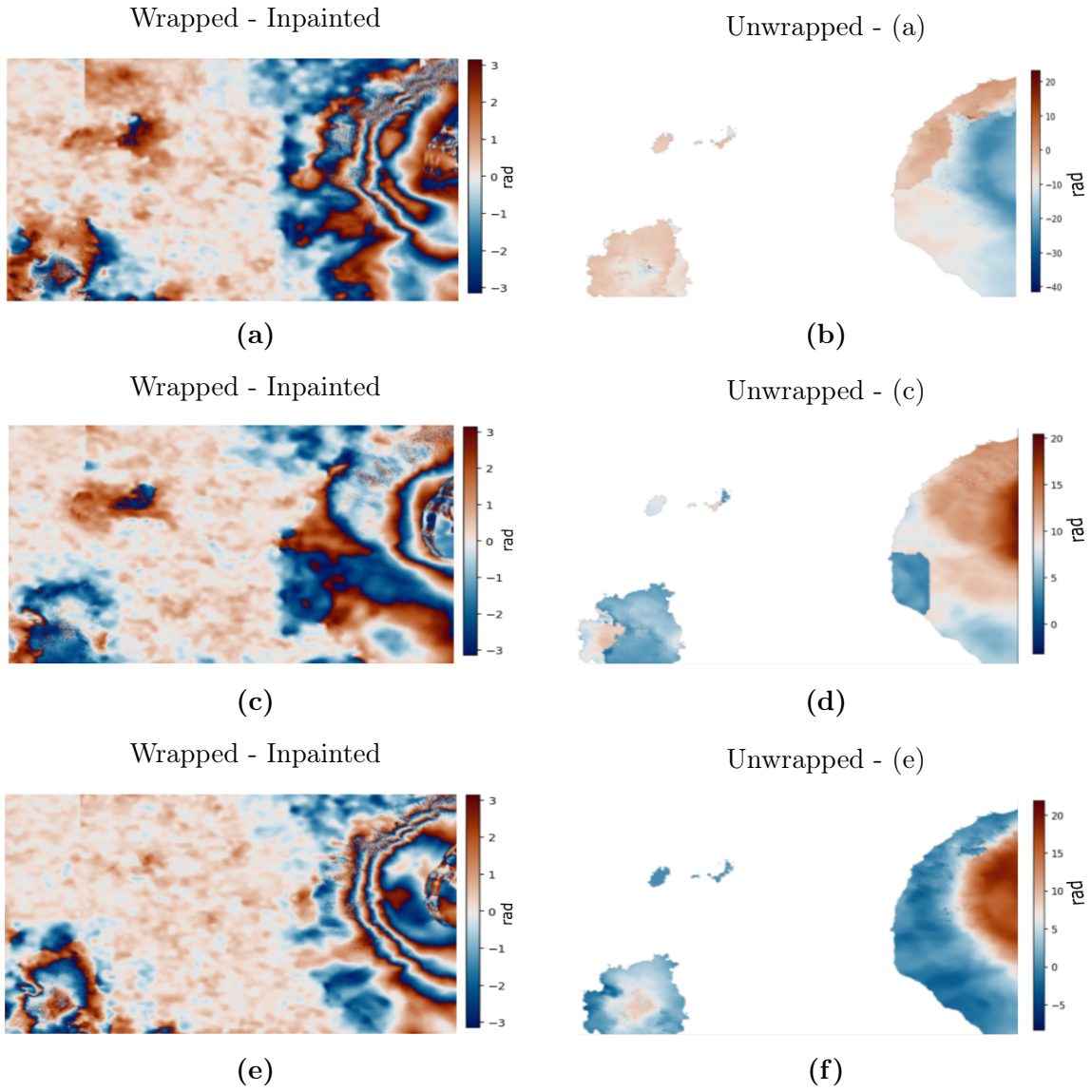


Figure 5.15 – Figure demonstrating that unwrapping is poor when MC[4]-GI is used to unwrap inpainted interferograms (LicSAR Frame 140D SM FGbr s4 test). The inpainting has been performed using a patch size of 256 × 256, an overlap of 128 × 128 and the coherence threshold is 0.1.

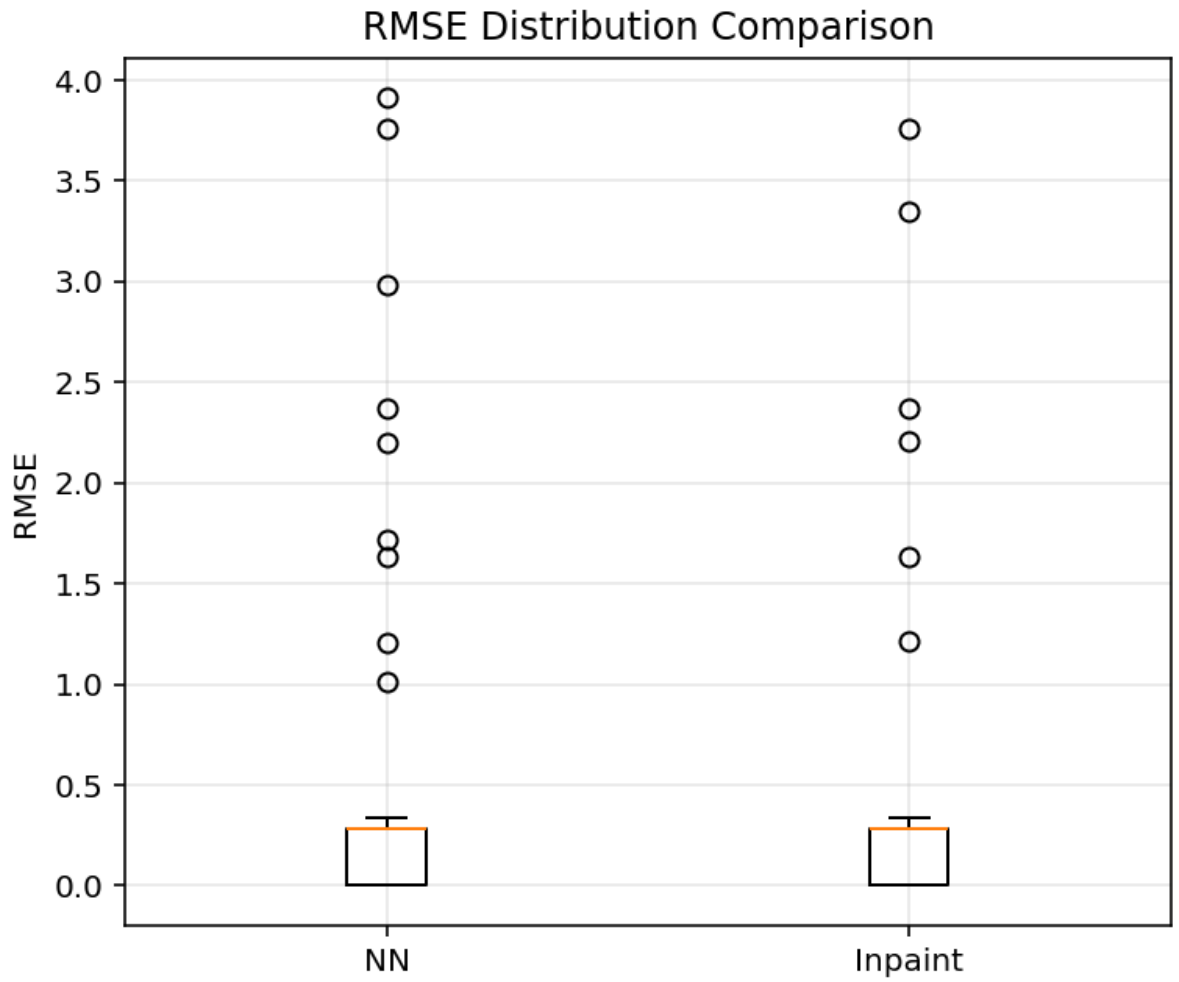


Figure 5.16 – Boxplots of the RMSE distribution for 46 unwrapped interferograms by the nearest neighbour (NN) and BBDM (Inpaint) approach. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap.

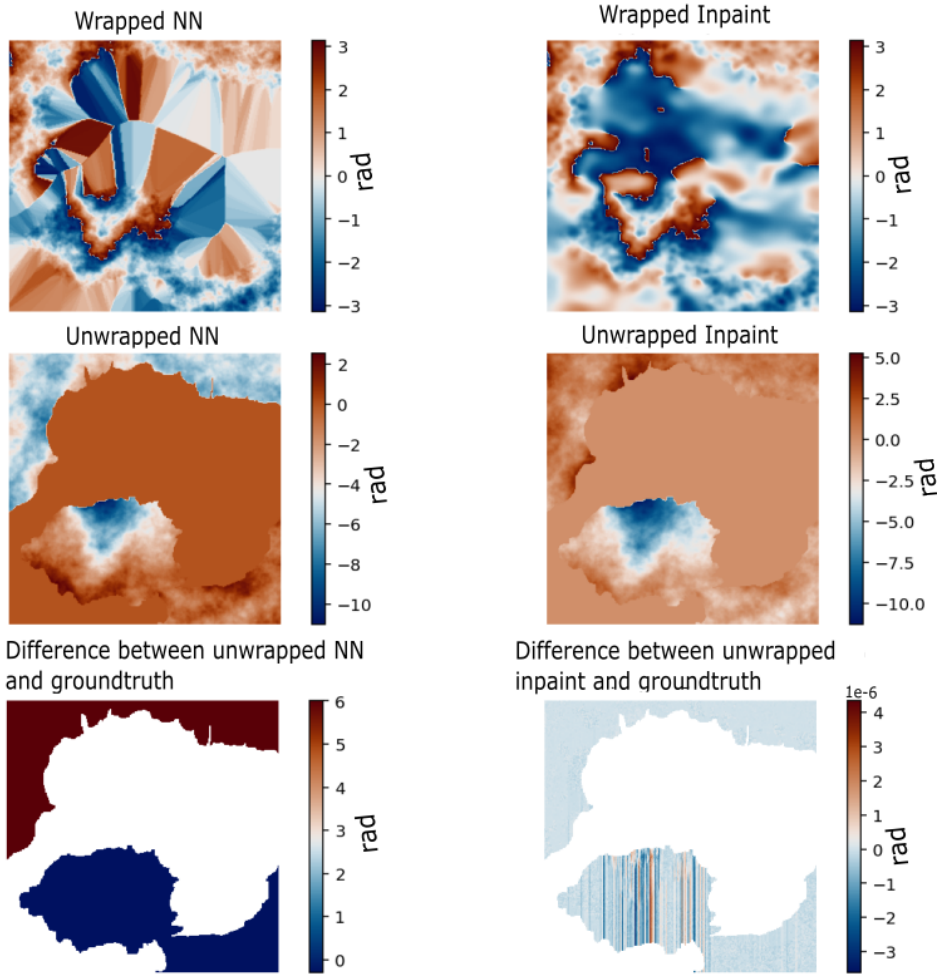


Figure 5.17 – Figure showing the wrapped phase, unwrapped phase and difference between the unwrapped groundtruth and unwrapped phase for the nearest neighbour (NN) and BBDM (inpaint) approach. This case shows where the BBDM approach has achieved better results than the NN approach. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap.

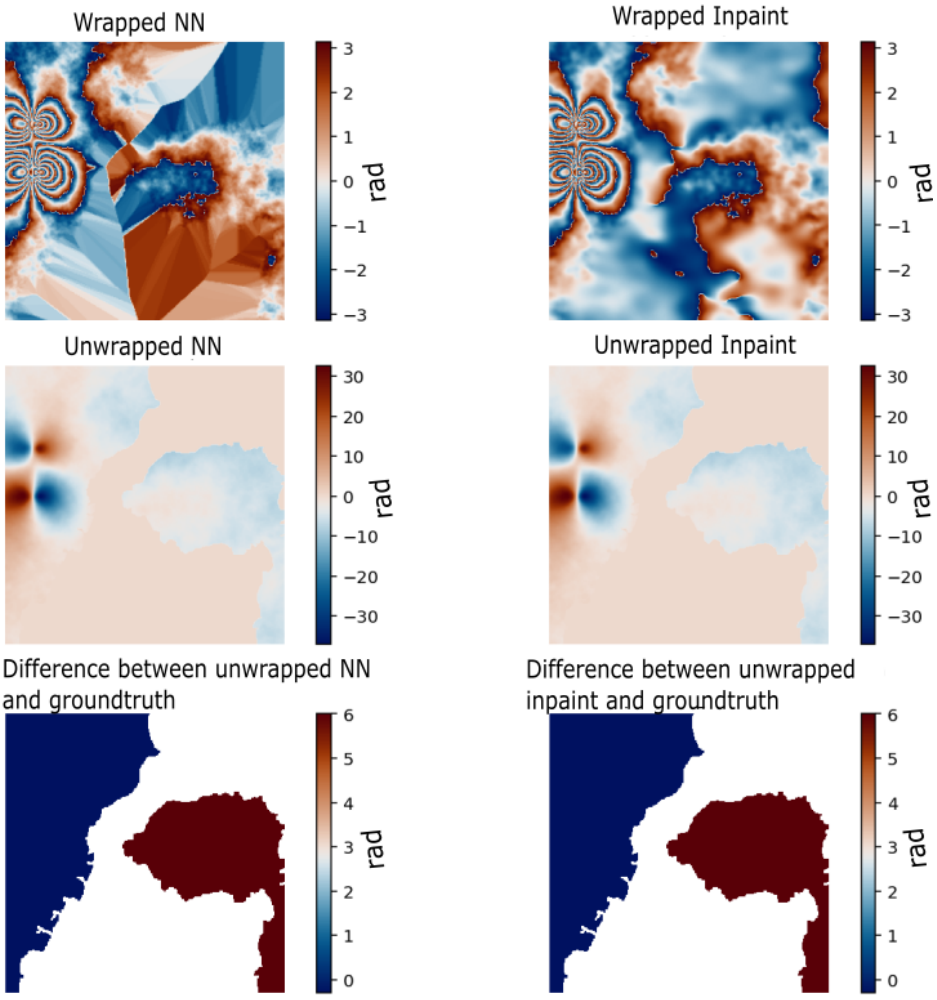


Figure 5.18 – Figure showing the wrapped phase, unwrapped phase and difference between the unwrapped groundtruth and unwrapped phase for the nearest neighbour (NN) and BBDM (inpaint) approach. This case shows where an error has likely occurred due to the introduction of a fringe between the two isolated regions in the BBDM approach. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap.

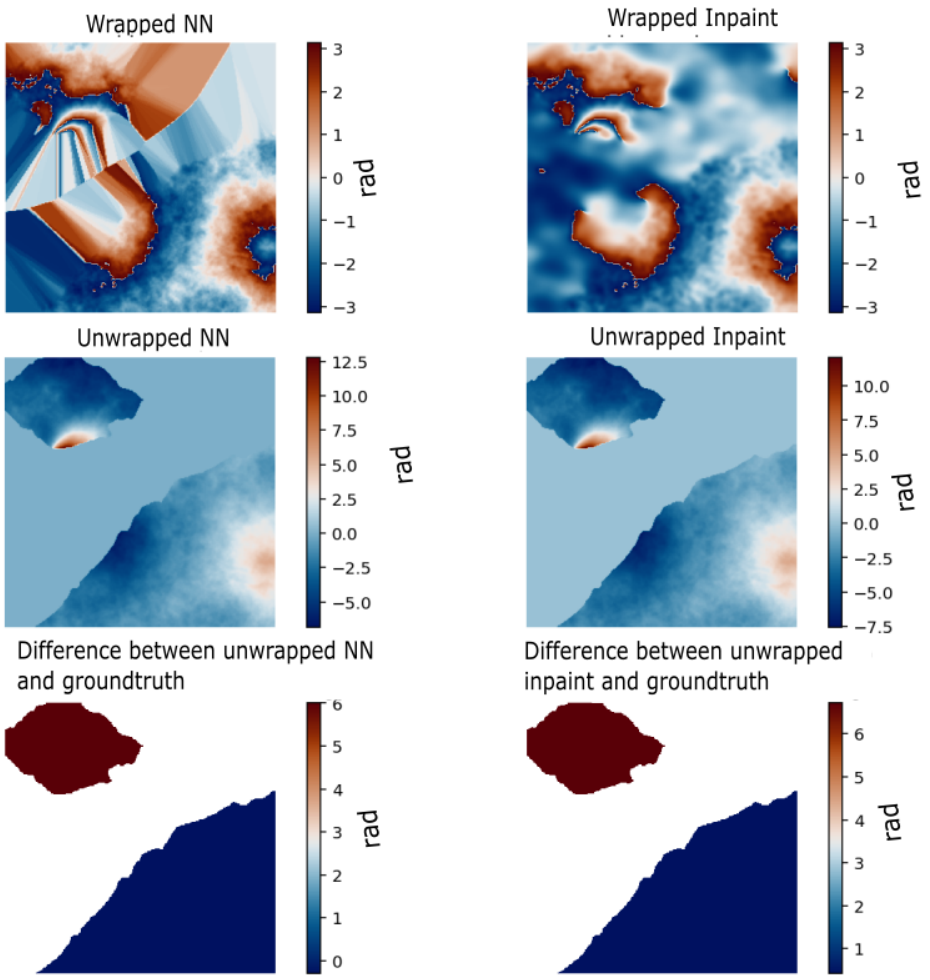


Figure 5.19 – Figure showing the wrapped phase, unwrapped phase and difference between the unwrapped groundtruth and unwrapped phase for the nearest neighbour (NN) and BBDM (inpaint) approach. This case shows where an error has likely occurred due a large portion of the deformation being masked in the BBDM approach. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap.

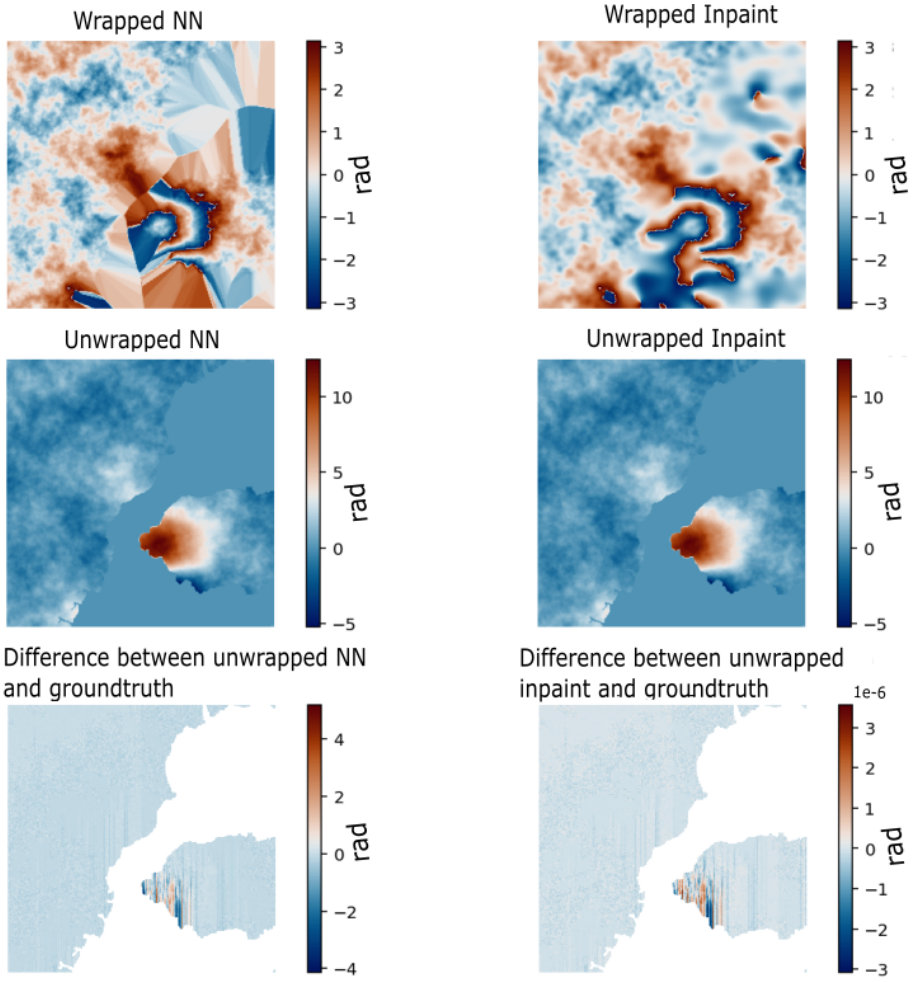


Figure 5.20 – Figure showing the wrapped phase, unwrapped phase and difference between the unwrapped groundtruth and unwrapped phase for the nearest neighbour (NN) and BBDM (inpaint) approach. This case shows where no error has been introduced despite the deformation being partially masked in the BBDM unwrapping. SNAPHU parameters: Mask: False, Coherence: 1 where there is no gap, 0.1 where for pixels within the gap.

5.4 Conclusion

In this Chapter, I use a Brownian Bridge Diffusion Model (Bo et al. 2023) to connect isolated coherent pixels with a reasonable estimation of the likely signal distribution to reduce phase unwrapping error caused by isolation of coherent regions. The model is trained using real Sentinel-1 interferograms containing mainly atmospheric signals rather than signal patterns associated with deformation events such as volcanic deformation or earthquakes. Whilst this limitation restricts the range of gap types and sizes to which BBDM can be effectively applied to, I am looking to apply this approach to connect isolated coherent regions rather than infill regions of deformation. This technique is successful at reducing loop closure error. Hence, this study provides a solid foundation for future research to build upon.

When applied to synthetic data, the complete unwrapping process using the BBDM to gap fill demonstrates more consistency than when using the nearest neighbour to gap fill, producing unwrapped interferograms with reduced error associated with the isolation of coherent regions.

Expanding the training data set to include examples of signals associated with large scale deformation events could further enhance the model's generalisation. As discussed in the results section, the success of the approach is very dependent upon the size of the gap and proportion of coherent pixels present to inform the gap filling process. For example, Figure 5.9 has a gap of around 24km diagonally with some horizontal gaps being of comparable size and has achieved good inpainting, but has coherent groupings of pixels from multiple sides resulting in a good inpainting result. In contrast, Figure 5.13 has a smaller horizontal gap of 19km, but is only gaining information from one side on some occasions resulting in a much poorer inpainted result. The data set was 64x64x8236, and whilst the mask shape and size varies each epoch, an increased data size and range of signals present as previously discussed increase the generalisation of the BBDM, reduce the occasions of a disappearing edge occurring, and minimise the risk of smoothing fringe

edges.

Overall, the results are promising, with examples shown where the gap filling has reduced the number of phase unwrapping errors associated with isolated coherent pixel regions. More work is required before the BBDM can be confidently applied during the phase unwrapping process to allow automation of the process.

Chapter 6

Conclusions

6.1 Summary of Research Outcomes

Improving phase unwrapping continues to be an active research field (Yan et al. 2025; Gao et al. 2024). Motivated by the success of deep learning both outside and within the field of phase unwrapping, in this research I explore the application of deep learning to improve phase unwrapping accuracy of InSAR data. I investigate the application of a semantic segmentation model to identify wrap count gradients, building upon previously published work by allowing the explicit classification of absolute wrap count gradients exceeding 1.

Across Chapter's 2 and 4, I discuss two integration methods to return the unwrapped phase from the phase gradient maps. In Chapter 2, I offer an L0 equivalent method which applies a Greedy algorithm, referred to as MC-GI; MC-GI prioritises the integration of high certainty predictions first. The 4 class approach, MC[4]-GI, works best on synthetic data, outperforming SNAPHU, MC[3]-GI and MC[5]-GI. For Sentinel-1 interferograms, the performance outcome is dependent on phase noise level; where little noise is present, the MC[4]-GI approach works well, unwrapping interferograms with high fringe rates but little noise better than SNAPHU. Nevertheless, two issues were identified with the MC-GI

approach.

Broadly in phase unwrapping, the continuation of phase is assumed (Gao et al. 2023), though for the MC-GI, this is not the case. Isolation of coherent regions can result in incorrect phase unwrapping due to the unwrapped regions being offset relative to each other. I demonstrate this type of unwrapping error is present in unwrapping results when the MC-GI and SNAPHU approaches are applied. A second limitation is resultant from the greedy nature of MC-GI. MC-GI relies on the assumption that the most probable step will to lead to the best unwrapped phase result without considering any future steps, meaning no backtracking of steps is allowed. MC-GI also relies on high model prediction accuracy and gradient maps with minimal noise. Whilst functioning well for data with minimal phase noise, the unwrapped result decreases in accuracy as noise increases, either due to large areas being classified as $C|2|$ and thus remaining wrapped, or due to the multi-class model identifying these regions as C0 , meaning the model considers the pixel gradients in an area to be comparable to large, random noise fields despite useful information sometimes being present, and often this will lead to unwrapping errors.

In Chapter 4, I present work which addresses the decrease in phase unwrapping result accuracy as phase noise increases. I present a cost-guided integration method, using the popular unwrapping approach SNAPHU, to improve the robustness of integration of wrap count gradient maps to high phase noise. Using a training data set, where circular Gaussian noise is applied to wrapped phase to simulate phase noise present in Sentinel-1 interferograms, I train a semantic segmentation model to classify wrap count gradients into 3, 4 and 5 classes. To improve robustness of integration to phase noise, I use the wrap count maps and model prediction certainties in combination with SNAPHU (Chen et al. 2001) as a cost-guided phase unwrapping approach, referring to the complete phase unwrapping technique as multi-class cost guided phase integration (MC-CGI). Though model gradient predictions have been used as weightings in combination with SNAPHU previously, and Wu et al. 2022 has converted the phase discontinuities into costs for SNAPHU to guide the MCF solver, a complete approach combining the model wrap

count gradient maps and model certainties to calculate the offset and variance to be given a priori to SNAPHU has not been previously proposed as far as this author is aware. Similar to the results presented in Chapter 2, the deep learning approach improves the accuracy of the unwrapped phase result compared to using SNAPHU alone for synthetic data. Unlike MC-GI, MC-CGI also consistently improves the unwrapped phase results of Sentinel-1 data, notably in regions of increased phase noise, though the choice of MC[3]-CGI, MC[4]-CGI or MC[5]-CGI has minimal impact on the overall unwrapped result. Where over prediction of $C|2|$ occurs, it can result in a worsening of unwrapped phase accuracy. In light of MC[4]-CGI having the best overall unwrapped result for synthetic interferograms, it does suggest the inclusion of the additional class should lead to an improvement in unwrapping performance. Thus, more work is required to optimally use the additional information provided by $C|2|$, though it would be interesting to explore whether by improving the MC[5]-CGI accuracy at gradient classification could also lead to an improvement in results due to the additional classes $C2^*$ and $C\bar{2}^*$.

Having illustrated that the isolation of coherent regions can lead to phase unwrapping errors, in Chapter 5 I present work exploring the use of inpainting to connect the regions with a realistic signal using a Brownian Bridge Diffusion Model, and highlight examples where unwrapping an inpainted wrapped interferogram has led to improved phase unwrapping. I apply a Brownian Bridge Diffusion Model (Bo et al. 2023) to connect the regions using inpainting. Comparing the results where SNAPHU is used to unwrap an inpainted wrapped phase and masked wrapped phase suggests that inpainting can reduce phase unwrapping errors associated with isolation of coherent regions. However, the realism of the inpainted results are strongly influenced by the gap size and proportion of original signal present; where insufficient real signal is present, the BBDM usually returns a result of very similar phase values, which when propagated across larger gaps looks unrealistic. This is unsurprising in view of the training data set containing mainly atmospheric signal where phase changes gradually across each training patch, meaning the BBDM is replicating this minimal phase change in the inpainted results. Further work is needed to increase the range of examples "seen" by the BBDM in the training

data set to better cope with such patterns. An increase of computational resources would allow larger patches to be used during training, increasing view size of the model and thus allowing for more phase change within a patch, though this may lead to the model being less accessible for general use given it would be reliant upon users also having access to large computer memory space.

6.2 Broader Summary

Phase unwrapping is a complicated process, liable to lead to errors in regions of high phase noise and where large gradients jumps have occured due the sampling level being insufficient relative to the level of sampling (Chen 2001). Within this thesis, I demonstrate that the inclusion of model identified gradients as a priori information in a cost-guided phase unwrapping approach does lead to a reduction in phase unwrapping errors in high noise regions. In previous publications, phase gradients have been restricted to 3 classes $(0,\pm1)$ which covers most scenarios, however did not allow the explicit identification of large gradient jump locations when doing so (Sica et al. 2022; Wang et al. 2021). Other studies identified discontinuity locations using a certainty based approach in which the residue locations and azimuth and range gradients had been pre-identified (Wu et al. 2022). I instead explore a one-stage wrap count gradient identification approach, which identifies both the wrap count gradient classes equivalent to those in previous publications, whilst also allowing the classification of gradients exceeding the assumed ±1 maximum through the inclusion of additional classes.

Whilst previous publications provide descriptions of the model training process, it is a non-trivial process to prepare the training data and model architecture and train the models multiple times. To provide some comparison of the impact of including an additional class to the unwrapped phase result, I train a 3 class, 4 class and 5 class model; when used to unwrap Sentinel-1 interferograms, there was little difference in the unwrapped result between each of the multi-class approaches, and groupings of $C|2|$ gradients led to

a similar effect as the isolation of coherent regions, likely due to the large gradient jumps being treated as equivalent to masked noise pixels. Given that for synthetic data, the inclusion of large gradients led to an improved unwrapped result, particularly over areas of rapid deformation, this indicates further development of the approach, whether by improving the distribution of noise across the synthetic interferogram to better simulate real Sentinel-1 noise distribution, or a more explicit use of the large gradient class sizing will likely lead to further improvement. Thus, though current results for Sentinel-1 data indicate little to no gain for the extra effort required to train the models to identify the large gradient classes, I suggest this is due to the limitations of the use method rather than truly reflecting the potential of including large gradient classes.

Throughout the thesis, I highlight the phase unwrapping errors related to the isolation of coherent pixel regions, which are present both in the results of MC-GI and MC-CGI but also when using SNAPHU alone. Gao et al. 2023 highlights both that there is a general assumption of phase continuity and that little research has been undertaken to reduce the occurrence of this sort of phase unwrapping error in 2D phase unwrapping. In Chapter 1 and 2, I discuss the use of Generative Adversarial Networks used as one-step phase unwrapping approaches. In theory, the GAN model architecture could be applied to gap fill the masked regions isolating coherent pixels, however, it has been raised that GANs are notoriously difficult to train, and can have smoothing in the resulting generated images (Dhariwal et al. 2021). Given that diffusion models are both less difficult to train and better at retaining the fine-scale details, I instead choose to use a BBDM, showing that the Brownian Bridge Diffusion Model can lead to reduced phase unwrapping errors. Though deep learning methods have been used to reduce phase unwrapping errors previously, to the best of this author's knowledge, this is the first time a diffusion based model has been used to address the isolation of coherent regions for wrapped interferograms using gap filling prior to phase unwrapping. Although the wrapped phase values predicted within the masked region are not the true returned signals, the values are representative of the expected distribution of values within the region, meaning that some guidance is present during the integration step as to where fringe edges may exist, and the relative

offsets between each isolated region. This work provides a foundation from which further study can build upon, demonstrating the tangibility of the gap filling method and that its application can lead to a reduction of phase unwrapping error, though currently the method is both computationally intensive and user intensive and further work is required to develop the method so that it can be confidently applied in an automated system.

6.3 Future Avenues

Within Chapter's 2, 3 and 4, I have explored how a deep learning model can be used to identify wrap count gradients with the intention of a more explicit identification of wrap count gradients which exceed one phase cycle. All of the MC-GI and MC-CGI models have shown good capability at identifying the minority classes, though with differing accuracy on synthetic data, however, it can be argued that the subsequent downstream processing of the wrap count gradient maps has not made full use of the minority classes $C|2|$, $C2^*$ and $C\bar{2}^*$. Given that over synthetic deformation areas particularly, the 4 class model MC-GI and MC-CGI approaches tended to lead to better unwrapped results than the respective 3 class versions, it suggests that the inclusion of the minority class is supportive in reducing the phase unwrapping errors, suggesting that further investigation exploring how to better utilise the minority class information, rather treating such pixel gradients as noise, may lead to improved phase unwrapped results. These investigations could take various forms.

The multi-class models could be expanded; currently, the 5 class model treats all wrap count gradients exceeding +2 as equivalent, and the same for -2 or less. In order to allow the inclusion of the wrap count gradients, an additional class could be included as a bin class. This would be equivalent to $C|2|$ for the MC[4] approaches however, instead include all wrap count gradient values exceeding +3. This does then cause the ongoing problem as to when the effort to train a model with additional classes is outweighed by the benefit of including a new class, especially given that as the class number increases, the classes

will become increasingly imbalanced due to the rarity of cases of each minority gradient. However, even the inclusion of the additional C3* class will allow for more cases to be covered during unwrapping, and mean the +2 and -2 gradients can be treated as such when calculating the offset values. Thus it is an interesting avenue to explore and may allow further improvement in regions of aliasing gradients which is an understudied area for 2D phase unwrapping (Pepin et al. 2024).

An alternate avenue of research is to use the MC-GI output as additional information in the MC-CGI approach. Given that MC-GI does perform better in regions of high fringe density than both SNAPHU alone and to a lesser extent the MC-CGI, it would be interesting to explore methods of identifying unwrapping cases where the enhanced unwrapping of MC-GI in the regions of high gradient deformation could be used in combination with the noise robustness of the MC-CGI approach. SNAPHU has an option to give a "first guess" of the unwrapped phase result. This does require exploring whether it does actually lead to an overall improved phase unwrapping result, and if such an MC-GI + MC-CGI approach is successful in reducing phase unwrapping errors, developing a processing pipeline to identify cases where the extra computational cost warrants the improvement in unwrapped result.

Early deep learning phase unwrapping research tested one-step approaches, going directly from wrapped to unwrapped phase, but suffered with over-smoothing. Given the development of diffusion models, which are better at generating fine-scale detail and reducing the occurrences of over-smoothing, and also given the speed at which the generative AI field itself generally is progressing, it is worthwhile analysing whether applying new generative AI models will lead to an improvement in the phase unwrapped result and an increase in generalisability for deformation.

The Brownian Bridge Diffusion Model was chosen due to its accessibility, both in respect to the code availability and the code being user friendly. The field of Generative AI is developing rapidly, meaning there are likely alternate models which would perform equally as well, or better than, the BBDM. Future work should explore the literature to

identify alternate methods and investigate the potential for gap filling for an InSAR data set to reduce the impact of isolated coherent regions leading to phase unwrapping.

The current, trained BBDM at minimum is capable of successfully inpainting small gaps, where real phase information is present on two edges and it is required to fill in the gap between. This has been shown to reduce the phase unwrapping error associated with two coherent regions being isolated from one another. However, due to the training data set being relatively small, containing mainly atmospheric signals and the patches used during training containing no more than two fringes, the number of scenarios the currently trained BBDM can be applied to is limited. Whilst it is capable of connecting two regions where relatively little deformation is occurring (e.g two islands which are separated by a water body), it would not currently be able to continue a series of fringe edges (e.g where an earthquake has occured). Therefore, the obvious next step is to increase both the size and variety of the data set. This should increase the generalisability of the model and reduce the occurrence of hallucinating and disappearing fringe edges.

Equally, thus far I have qualitatively identified that broadly a 25% coverage of grouped data generally leads to a realistic data distribution within the gap. Where data is more sporadic, the inpainted result looks unrealistic which has a subsequent effect on the unwrapping success. Further experimentation should investigate the impact of varying data resolution, gap size, patch size and training dataset size and variety; this will likely lead to an improved model capable of generating data which more closely approximates real world approximations. This would signify a step towards a including a generative model as an automated processing step. Once improved, an ensemble approach, running the inpainting several times from which to extract the best output for phase unwrapping, could be trialled. This is reliant on having sufficient computational resources, but may work well and enhance unwrapped results for small interferograms. As the initial results do indicate that the gap filling can reduce phase unwrapping errors due to isolation of coherent regions, further studies to improve on the above mentioned limitations seem worthwhile.

Once an improved BBDM is trained, it could be implemented with the improved wrap count gradient estimation models to provide a two-fold improvement to the phase unwrapping process. The new processing chain would start with the BBDM generating a series of possible inpainted interferograms, with the most common data distribution used for subsequent unwrapping. The inpainted wrapped interferogram would be given either to the MC-GI and MC-CGI models, dependent on the level of noise.

In most scenarios, the use of the MC-CGI approach would likely lead to the best unwrapped result. However, where a deformation has occured, which has a high fringe rate and relatively little noise, the MC-GI approach could be used.

The unwrapped output from MC-GI could then be used as an additional input to SNAPHU in the MC-CGI approach or taken as the unwrapped result.

Experimentation could be undertaken to design a processing chain which implements a series of parameters, or decision points, to decide for each interferogram which of the BBDM, MC-GI and MC-CGI should be included to achieve the best unwrapped result. Once trained, the models can be run on a CPU, meaning that users are not required to have GPUs and therefore making the models more accessible.

At present, the MC-GI and MC-CGI models take seconds to run. The time taken for the Greedy Integration Algorithm to return the wrap count is dependent on the patch size, with a patch of size 224×224 taking a few seconds, whilst an interferogram of size greater than 1000×1000 can take a number of hours. It would be relatively easy to implement within a wider system given that I have designed the code so that the user is required to provide the wrapped interferograms as input, and will receive the unwrapped phase as output. For those who would like to retrain the models for specific datasets, the most complex step is ensuring the wrapped phase is of a suitable size, and the additional information for further processing (i.e coherence mask) is available. With further development, a downloadable package for use in python could be created, which at the basic level requires the user to input the coherence and wrapped interferogram to

output an unwrapped interferogram, though be adaptable to allow user input for variables such as model choice or coherence threshold level (though another quality indexing system could be used instead).

If successfully implemented, this package could be easily implemented to replace current unwrappers. The complete method could be used to improve unwrapped phase by reducing the error associated with high fringe rates, inaccurate phase gradient estimation and isolation of coherent regions.

Bibliography

Abraham, Nabila and Naimul Mefraz Khan (2018). "A Novel Focal Tversky Loss Function With Improved Attention U-Net for Lesion Segmentation". In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pp. 683–687. URL: https://api.semanticscholar.org/CorpusID:53016422.

An, L, Q.S Xiang, and S Chavez (2000). "A fast implementation of the minimum spanning tree method for phase unwrapping". In: *IEEE Transactions on Medical Imaging* 19.8, pp. 805–808. DOI: 10.1109/42.876306.

Arevalillo-Herraez, M, F Villatoro, and M Gdeisat (2016). "A Robust and Simple Measure for Quality-Guided 2D Phase Unwrapping Algorithms". In: *IEEE Transactions on Image Processing* 25.6, pp. 2601–2609. DOI: 10.1109/TIP.2016.2551370.

Azad, Reza, Moein Heidary, Kadir Yilmaz, Michael Hüttemann, Sanaz Karimijafarbigloo, Yuli Wu, Anke Schmeink, and Dorit Merhof (2023). *Loss Functions in the Era of Semantic Segmentation: A Survey and Outlook*. arXiv: 2312.05391 [cs.CV]. URL: https://arxiv.org/abs/2312.05391.

Balmer, R and P Harti (1998). "Synthetic aperture radar interferometry". In: *Inverse Problems* 14.4. DOI: 10.1088/0266-5611/14/4/001.

Bekaert, David, Richard Walters, Tim Wright, Andy Hooper, and Doug Parker (Sept. 2015). "Statistical comparison of InSAR tropospheric correction techniques". In: *Remote Sensing of Environment* 170, pp. 40–47. DOI: 10.1016/j.rse.2015.08.035.

Bo, L, X Kaitao, L Bin, and L Yu-Kun (2023). *BBDM: Image-to-image Translation with Brownian Bridge Diffusion Models*. arXiv: 2205.07680 [cs.CV]. URL: https://arxiv.org/abs/2205.07680.

Bountos, Nikolaos Ioannis, Dimitrios Michail, and Ioannis Papoutsis (2022). "Learning from Synthetic Insar with Vision Transformers: The case of volcanic unrest detection". In: *IEEE Transactions on Geoscience and Remote Sensing* 60, pp. 1–12. DOI: 10.1109/tgrs.2022.3180891.

British Geological Survey (2024). *Volcanoes*. URL: https://www.bgs.ac.uk/discovering-geology/earth-hazards/volcanoes/ (visited on 12/06/2024).

Calvanese, Francesco, Francescopaolo Sica, Giuseppe Scarpa, and Paola Rizzoli (2020). "InSAR Phase Unwrapping using Convolutional Neural Network". In: *2020 IEEE Radar Conference (RadarConf20)*, pp. 1–6. DOI: 10.1109/RadarConf2043947.2020.9266485.

Calvari, Sonia, Alessandro Bonaccorso, Annalisa Cappello, Flora Giudicepietro, and Eugenio Sansosti (2022). "Volcanic Processes Monitoring and Hazard Assessment Using Integration of Remote Sensing and Ground-Based Techniques". In: *Remote Sensing* 14.15. ISSN: 2072-4292. DOI: 10.3390/rs14153626. URL: https://www.mdpi.com/2072-4292/14/15/3626.

Cao, Yihan, Siyu Li, Yixin Liu, Zhiling Yan, Yutong Dai, Philip S. Yu, and Lichao Sun (2023). *A Comprehensive Survey of AI-Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT*. arXiv: 2303.04226 [cs.AI]. URL: https://arxiv.org/abs/2303.04226.

Chen, C and H Zebker (2001). "Two-dimensional phase unwrapping with use of statistical models for cost functions in nonlinear optimization". In: *J. Opt. Soc. Am. A.* 18.2, pp. 338–351. DOI: https://doi.org/10.1364/JOSAA.18.000338.

Chen, C. (2001). "Statistical-cost network-flow approaches to two-dimensional phase unwrapping for radar interferometry". PhD thesis. Stanford University.

Chen, X, C He, and Y Huang (2023a). "An error distribution-related function-trained two-dimensional insar phase unwrapping method via U-GauNet". In: *Signal, Image and Video Processing*. DOI: 10.1007/s11760-022-02482-y.

Chen, X., C. He, and Y. Huang (2023b). "An error distribution-related function-trained two-dimensional InSAR phase unwrapping method via U-GauNet". In: *Signal, Image and Video Processing* 17.5, pp. 2653–2660.

Chen, Yu, Shuai Wang, Yandong Gao, Yanjian Sun, Jinqi Zhao, Kun Tan, and Peijun Du (2025). "DMP-PUNet: A novel network for two-dimensional InSAR phase unwrapping under severe noise and complex fringes conditions". In: *International Journal of Applied Earth Observation and Geoinformation* 139, p. 104519. ISSN: 1569-8432. DOI: https://doi.org/10.1016/j.jag.2025.104519. URL: https://www.sciencedirect.com/science/article/pii/S1569843225001669.

Copernicus SentiWiki (2026). *Sentinel-1 Mission*. URL: https://sentiwiki.copernicus.eu/web/s1-mission (visited on 03/08/2026).

Dhariwal, Prafulla and Alex Nichol (2021). *Diffusion Models Beat GANs on Image Synthesis*. arXiv: 2105.05233 [cs.LG]. URL: https://arxiv.org/abs/2105.05233.

Fan, Yanshuo, Juan Hiedra Cobo, Oliver Wang, Jainam Shroff, Aagyapal Kaur, and Zheng Liu (2025). "A Transformer-Based Architecture for InSAR Phase Unwrapping Under Noisy Conditions". In: *IEEE Transactions on Geoscience and Remote Sensing* 63, pp. 1–11. DOI: 10.1109/TGRS.2025.3626752.

Ferretti, Alessandro, Andrea Monti-Guarnieri, Claudio Prati, Fabio Rocca, and Didier Massonnet (Feb. 2007). *Part C: InSAR processing: a mathematical approach*. Tech. rep. TM-19. Printed in The Netherlands. Noordwijk, The Netherlands: ESA/ESTEC. URL: https://www.esa.int.

Flynn, T.J. (1998). "Phase unwrapping using discontinuity optimization". In: *IGARSS '98. Sensing and Managing the Environment. 1998 IEEE International Geoscience and Remote Sensing. Symposium Proceedings. (Cat. No.98CH36174)*. Vol. 1, 80–82 vol.1. DOI: 10.1109/IGARSS.1998.702805.

Gabriel, A. K., R. M. Goldstein, and H. A. Zebker (1989). "Mapping small elevation changes over large areas: differential radar interferometry". In: *Journal of Geophysical Research* 94.B7, pp. 9183–9191. DOI: 10.1029/JB094iB07p09183.

Gaddes, ME, A Hooper, and M Bagnardi (2019). "Using machine learning to automatically detect volcanic unrest in a time series of interferograms". In: *Journal of Geophysical Research: Solid Earth* 124.11, pp. 12304–12322. DOI: 10.1029/2019JB017519.

Gao, J, H Jiang, Z Sun, R Wang, and Y Han (2023). "A Parallel InSAR Phase Unwrapping Method Based on Separated Continuous Regions". In: *Remote Sensing* 15.5. ISSN: 2072-4292. DOI: 10.3390/rs15051370. URL: https://www.mdpi.com/2072-4292/15/5/1370.

Gao, Jian, Yunfei Guo, Shuohang Yang, and Zhongchang Sun (2022a). "Phase Unwrapping Method Based on Branch-Cuts of Region Residues". In: *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, pp. 2450–2453. DOI: 10.1109/IGARSS46834.2022.9884233.

Gao, Y, G Wang, G Wang, T Li, S Zhang, S Li, Y Zhang, and T Zhang (2022b). "Two-dimensional phase unwrapping method using a refined D-LinkNet-based unscented Kalman filter". In: *Optics and Lasers in Engineering* 152, p. 106948. DOI: https://doi.org/10.1016/j.optlaseng.2022.106948.

Gao, Yandong, Nanshan Zheng, Shijin Li, Yansuo Zhang, Qiuzhao Zhang, and Shubi Zhang (2024). "A Novel Two-Step DInSAR Phase Unwrapping for Large Gradient Mining Deformation". In: *IEEE Journal on Miniaturization for Air and Space Systems* 5.1, pp. 2–8. DOI: 10.1109/JMASS.2023.3305242.

García, Andrea (2025). "Greedy algorithms: a review and open problems". In: *Journal of Inequalities and Applications* 2025.1, p. 11.

Ghiglia, D and L Romero (1994). "Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods". In: *Journal of the Optical Society of America A* 11, pp. 107–117. DOI: https://doi.org/10.1364/JOSAA.11.000107.

– (1996). "Minimum Lp-norm two-dimensional phase unwrapping". In: *Journal of the Optical Society of America A* 13.10, pp. 1999–2013. DOI: https://doi.org/10.1364/JOSAA.13.001999.

Goldstein, R, H Zebker, and C Werner (1988). "Satellite radar interferometry: Two-dimensional phase unwrapping". In: *Advancing Earth and Space Science* 23.4, pp. 713–720. DOI: 10.1029/RS023i004p00713.

Hanssen, Rolf F. (2009). *Radar Interferometry: Data Interpretation and Error Analysis*. Berlin: Springer. ISBN: 978-3-540-85017-0.

Hooper, A (2010). "A statistical-cost approach to unwrapping the phase of InSAR time series". In: *Proceedings of the International Workshop on ERS SAR Interferometry, Frascati, Italy*. Vol. 30.

Hooper, Andrew (2008). "A multi-temporal InSAR method incorporating both persistent scatterer and small baseline approaches". In: *Geophysical research letters* 35.16.

Hooper, Andrew, Howard Zebker, Paul Segall, and Bert Kampes (2004). "A new method for measuring deformation on volcanoes and other natural terrains using InSAR persistent scatterers". In: *Geophysical research letters* 31.23.

Hwang, Jieon and Yoan Shin (2022). "SAR-to-Optical Image Translation Using SSIM Loss Based Unpaired GAN". In: *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 917–920. DOI: 10.1109/ICTC55196.2022.9952649.

Itoh, K (1982). "Analysis of the phase unwrapping algorithm". In: *Applied optics* 21.14, pp. 2470–2470.

Kester, W (2009). "What the Nyquist criterion means to your sampled data system design". In: *Analog Devices*, pp. 1–12.

Kim, Seon-Hoon and Daewon Chung (2025). "Conditional Brownian Bridge Diffusion Model for VHR SAR to Optical Image Translation". In: *IEEE Geoscience and Remote Sensing Letters* 22, pp. 1–5. ISSN: 1558-0571. DOI: 10.1109/lgrs.2025.3562401. URL: http://dx.doi.org/10.1109/LGRS.2025.3562401.

Lazecký, M, K Spaans, P.J González, Y Maghsoudi, Y Morishita, F Albino, N Elliott J Greenall, E Hatton, A Hooper, and D Juncu (2020). "LiCSAR: An automatic InSAR tool for measuring and monitoring tectonic and volcanic activity". In: *Remote Sensing* 12.15, p. 2430. DOI: https://doi.org/10.3390/rs12152430.

2985 LeCun, Y, Y Bengio, and G Hinton (2015). "Deep Learning". In: *Nature* 521.7553, pp. 436–
2986 444. DOI: 10.1038/nature14539.
2987 Li, L, H Zhang, Y Tang, C Wang, and F Gu (2022). "InSAR Phase Unwrapping by Deep
2988 Learning Based on Gradient Information Fusion". In: *IEEE Geoscience and Remote*
2989 *Sensing Letters* 19, pp. 1–5. DOI: 10.1109/LGRS.2021.3127318.
2990 Liu, Boyu, Lingda Wu, Xiaorui Song, Hongxing Hao, Ling Zou, and Yu Lu (Mar. 2023).
2991 "PUnet: A robust framework for phase unwrapping in interferometric SAR". In: *Fron-*
2992 *tiers in Environmental Science* 11, 1138399, p. 1138399. DOI: 10.3389/fenvs.2023.
2993 1138399.
2994 Ma, Zhangfeng, Jihong Liu, Xiaojie Liu, Cheng Zhou, Jia Hu, Yan Cui, Shuhong Qin,
2995 Teng Huang, and Zhen Li (2022). "Go Extra Miles: An Additional Error Correc-
2996 tion Procedure Aimed to Further Improve Phase Unwrapping Accuracy and Reduce
2997 Creep Model Uncertainty". In: *Journal of Geophysical Research: Solid Earth* 127.1,
2998 e2021JB022478.
2999 Marinkovic, P and R Hanssen (2004). "Advanced InSAR coregistration using point clus-
3000 ters". In: *IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing*
3001 *Symposium*. Vol. 1, pp. 489–492. DOI: 10.1109/IGARSS.2004.1369070.
3002 Montesinos López, Osval Antonio, Abelardo Montesinos López, and Jose Crossa (2022).
3003 "Fundamentals of Artificial Neural Networks and Deep Learning". In: *Multivariate Sta-*
3004 *tistical Machine Learning Methods for Genomic Prediction*. Cham: Springer Interna-
3005 tional Publishing, pp. 379–425. ISBN: 978-3-030-89010-0. DOI: 10.1007/978-3-030-
3006 89010-0_10. URL: https://doi.org/10.1007/978-3-030-89010-0_10.
3007 Morishita, Y, M Lazecky, T Wright, J Weiss, J Elliott, and A Hooper (2020). "LiCS-
3008 BAS: An open-source INSAR time series analysis package integrated with the LIC-
3009 SAR Automated Sentinel-1 Insar Processor". In: *Remote Sensing* 12.3, p. 424. DOI:
3010 10.3390/rs12030424.
3011 Mu, Jingqin, Yuzhu Wang, Sheng Zhan, Guoqing Yao, Kun Liu, Yueqin Zhu, and Lizhe
3012 Wang (2023). "A coherence-guided InSAR phase unwrapping method with cycle-consistent

adversarial networks". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17, pp. 2690–2704.

Nichol, Alex and Prafulla Dhariwal (2021). "Improved Denoising Diffusion Probabilistic Models". In: *CoRR* abs/2102.09672. arXiv: 2102.09672. URL: https://arxiv.org/abs/2102.09672.

Oktay, Ozan, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert (2018). *Attention U-Net: Learning Where to Look for the Pancreas*. arXiv: 1804.03999 [cs.CV]. URL: https://arxiv.org/abs/1804.03999.

Oveis, A.H, E Guisti, S Ghio, and M Martorella (2021). "A Survey on the Applications of Convolutional Neural Networks for Synthetic Aperture Radar: Recent Advances". In: *IEEE Aerospace and Electronic Systems Magazine*.

Payne, Jessica A, AR Watson, Y Maghsoudi, SK Ebmeier, R Rigby, M Lazeckỳ, M Thomas, and JR Elliott (2025). "Widespread extent of irrecoverable aquifer depletion revealed by country-wide analysis of land surface subsidence hazard in Iran". In: *Journal of Geophysical Research: Solid Earth* 130.9, e2024JB030367.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pepe, A and F Calò (2017). "A review of interferometric synthetic aperture RADAR (InSAR) multi-track approaches for the retrieval of Earth's surface displacements". In: *Applied Sciences* 7.12, pp. 1264–1303. DOI: 10.3390/app7121264.

Pepin, Karissa and Howard Zebker (2024). "Aliasing in InSAR 2-D phase unwrapping and time series". In: *IEEE Transactions on Geoscience and Remote Sensing* 62, pp. 1–17.

Pu, L, X Zhang, Z Zhou, L Li, L Zhou, K Shi, and S Wei (2021). "A Robust InSAR Phase Unwrapping Method via Phase Gradient Estimation Network". In: *Remote Sensing* 13.22, p. 4564. DOI: https://doi.org/10.3390/rs13224564.

Ronneberger, O, P Fischer, and T Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28.

Salehi, Seyed Sadegh Mohseni, Deniz Erdogmus, and Ali Gholipour (2017). "Tversky loss function for image segmentation using 3D fully convolutional deep networks". In: *CoRR* abs/1706.05721. arXiv: 1706.05721. URL: http://arxiv.org/abs/1706.05721.

Sica, F, F Calvanese, G Scarpa, and P Rizzoli (2022). "A CNN-Based Coherence-Driven Approach for InSAR Phase Unwrapping". In: *IEEE Geoscience and Remote Sensing Letters* 19, pp. 1–5. DOI: 10.1109/LGRS.2020.3029565.

Spoorthi, G, S Gorthi, and R.K Sai Subrahmanyam Gorthi (2019). "PhaseNet: A Deep Convolutional Neural Network for Two-Dimensional Phase Unwrapping". In: *IEEE Signal Processing Letters* 26.1, pp. 54–58. DOI: 10.1109/LSP.2018.2879184.

Spoorthi, G.E, R.K Sai Subrahmanyam Gorthi, and Gorthi S (2020). "PhaseNet 2.0: Phase Unwrapping of Noisy Data Based on Deep Learning Approach". In: *IEEE Transactions on Image Processing* 29, pp. 4862–4872. DOI: 10.1109/TIP.2020.2977213.

Tuel, Alexandre, Thomas Kerdreux, Claudia Hulbert, and Bertrand Rouet-Leduc (2023). *Diffusion Models for Interferometric Satellite Aperture Radar*. arXiv: 2308.16847 [cs.CV]. URL: https://arxiv.org/abs/2308.16847.

Vijay Kumar, Sachin, Xinyao Sun, Zheng Wang, Ryan Goldsbury, and Irene Cheng (2023). "A U-Net Approach for InSAR Phase Unwrapping and Denoising". In: *Remote Sensing* 15.21. ISSN: 2072-4292. DOI: 10.3390/rs15215081. URL: https://www.mdpi.com/2072-4292/15/21/5081.

Wang, H, J Hu, H Fu, C Wang, and Z Wang (2021). "A Novel Quality-Guided Two-Dimensional InSAR Phase Unwrapping Method via GAUNet". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14, pp. 7840–7856. DOI: 10.1109/JSTARS.2021.3099485.

Wang, K, Y Li, Q Kemao, J Di, and J Zheo (2019). "One-step robust deep learning phase unwrapping". In: *Optics Express* 27.10, pp. 15100–15115. DOI: https://doi.org/10.1364/OE.27.015100.

Wang, Zhou, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli (2004). "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4, pp. 600–612. DOI: 10.1109/TIP.2003.819861.

Wu, Z, T Wang, Y Wang, and D Ge (2021a). "A New Phase Unwrapping Method Combining Minimum Cost Flow with Deep Learning". In: *In 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 3177–3180. DOI: 10.1109/IGARSS47720.2021.9554886.

– (2022). "Deep-learning-based phase discontinuity prediction for two-dimensional phase unwrapping of SAR interferograms". In: *IEEE Transactions on Geoscience and Remote Sensing* 60, pp. 1–16. DOI: 10.1109/TGRS.2021.3121906.

Wu, Z, T Wang, Y Wang, R Wang, and D Ge (2021b). "Deep learning for the detection and phase unwrapping of mining-induced deformation in large-scale interferograms". In: *IEEE Transactions on Geoscience and Remote Sensing*. DOI: 10.1109/TGRS.2021.3121907.

Xin, Chen and Tang Jing-Song (2010). "MST phase unwrapping based on new weighted map". In: *2010 3rd International Congress on Image and Signal Processing*. Vol. 9, pp. 4408–4412. DOI: 10.1109/CISP.2010.5647889.

Yan, Chao, Tao Li, Yandong Gao, Shijin Li, Xiang Zhang, Xuefei Zhang, Di Zhang, and Huiqin Liu (2025). "A Novel Two-Stage Learning-Based Phase Unwrapping Algorithm via Multi-Model Fusion". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.

Yang, Kai, Zhihui Yuan, Xuemin Xing, and Lifu Chen (2023a). "Deep-learning-based mask-cut method for InSAR phase unwrapping". In: *IEEE Journal on Miniaturization for Air and Space Systems* 4.2, pp. 221–230.

Yang, L, Q Feng, and Z Wang (2006). "Optimized minimum spanning tree phase unwrapping algorithm for phase image of interferometric SAR". In: *2006 6th International Conference on ITS Telecommunications*, pp. 1240–1243. DOI: 10.1109/ITST.2006.288851.

Yang, Yu, Bingqian Chen, Zhenhong Li, Chen Yu, Chuang Song, and Fengcheng Guo (2023b). "A novel phase unwrapping method for low coherence interferograms in coal

mining areas based on a fully convolutional neural network". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17, pp. 601–613.

Zhang, Junkang and Qingguang Li (Mar. 2022). "EESANet: Edge-enhanced self-attention network for two-dimensional phase unwrapping". In: *Optics Express* 30.7, p. 10470. DOI: 10.1364/oe.444875.

Zhang, L, G Huang, Y Li, S Yang, L Lu, and W Huo (2023). "A Robust InSAR Phase Unwrapping Method via Improving the pix2pix Network". In: *Remote Sensing* 15.19. ISSN: 2072-4292. DOI: 10.3390/rs15194885. URL: https://www.mdpi.com/2072-4292/15/19/4885.

Zhang, Ziwen, Jiang Qian, Yong Wang, and Xiaobo Yang (2021). "An Improved Least Square Phase Unwrapping Algorithm Combined with Convolutional Neural Network". In: *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 3384–3387. DOI: 10.1109/IGARSS47720.2021.9555102.

Zheng, Yujie, Howard Zebker, and Roger Michaelides (2021). "A New Decorrelation Phase Covariance Model for Noise Reduction in Unwrapped Interferometric Phase Stacks". In: *IEEE Transactions on Geoscience and Remote Sensing* 59.12, pp. 10126–10135. DOI: 10.1109/TGRS.2021.3050087.

Zhong, H, J Tang, S Zhang, and X Zhang (2014). "A Quality-Guided and Local Minimum Discontinuity Based Phase Unwrapping Algorithm for InSAR/InSAS Interferograms". In: *IEEE Geoscience and Remote Sensing Letters* 11.1, pp. 215–219. DOI: 10.1109/LGRS.2013.2252880.

Zhong, Wenjie, Jia Li, Xin Li, Juanjuan Feng, Lei Guo, Zhiqiang Li, Junhui Wu, and Lingshuai Kong (2024). "A Discontinuity-Guided Two-Dimensional Phase Unwrapping Method for SAR Interferograms". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17, pp. 9309–9323. DOI: 10.1109/JSTARS.2024.3392637.

Zhou, L, H Yu, Y Lan, and M Xing (2022a). "Deep Learning-Based Branch-Cut Method for InSAR Two-Dimensional Phase Unwrapping". In: *IEEE Transactions on Geoscience and Remote Sensing* 60, pp. 1–15. DOI: 10.1109/TGRS.2021.3099997.

Zhou, L, H Yu, V Pascazio, and M Xing (2022b). "PU-GAN: An One-step 2-D InSAR Phase Unwrapping based on Conditional Generative Adversarial Network". In: *IEEE Transactions on Geoscience and Remote Sensing*. Part b, p. 1. DOI: 10.1109/TGRS.2022.3145342.

Zhou, Lifan and Hanwen Yu (2025). "MoDL-PU: Model-Based Deep Learning for InSAR Phase Unwrapping". In: *IEEE Transactions on Geoscience and Remote Sensing* 63, pp. 1–11. DOI: 10.1109/TGRS.2025.3549607.

Zhou, Lifan, Hanwen Yu, and Yang Lan (2020). "Deep Convolutional Neural Network-Based Robust Phase Gradient Estimation for Two-Dimensional Phase Unwrapping Using SAR Interferograms". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.7, pp. 4653–4665. DOI: 10.1109/TGRS.2020.2965918.

Zhou, Lifan, Hanwen Yu, Yong Wang, Yang Lan, and Mengdao Xing (2022c). "PG-BCNet: A neural network combined with the PGNet and BCNet for 2-D InSAR phase unwrapping". In: *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, pp. 2470–2473.